الجمهوريــة الجزائريــة الديمقراطيــة الشعبيــة
Democratic and Popular Republic of Algeria
وزارة التعلـيـم العـالي و البـحـث العـلـمـي
Ministry of Higher Education and Scientific Research

Mustapha Stambouli University
Mascara
Faculty of exact science
Departement of computer science

جامعة مصطفى اسطمبولي
معسكر

DOCTORAL THESIS

Speciality: Networks and distributed systems

Distributed system for image segmentation based on parallel metaheuristics

Presented by **:** Mezaghrani Ali.

On 16/06/2025.

In front of the committee composed of:

President	**REBBAH Mohammed**	**Professor**	**University of Mascara**
Examiner	**SMAIL Omar**	**Professor**	**University of Mascara**
Examiner	**MEKKAOUI Kheireddine**	**MCA**	**University of Saida**
Supervisor	**DEBAKLA Mohammed**	**MCA**	**University of Mascara**
Co-Supervisor	**DJEMAL Khalifa**	**Professor**	**University of Evry, France**

Abstract

In recent years, the integration of distributed systems with parallel processing techniques has significantly advanced the field of image processing. Distributed systems enable the efficient handling of large datasets by dividing the computational tasks across multiple nodes, improving both speed and scalability. Parallelization in image processing enhances performance by executing multiple tasks simultaneously, making it possible to process high-resolution images and complex datasets in real time. Metaheuristic algorithms have been widely adopted for optimization tasks in image processing due to their ability to explore large search spaces effectively. These algorithms, when coupled with machine learning (ML) models, provide powerful solutions for feature selection in classification tasks. Metaheuristics help identify the most relevant features from large datasets, thereby enhancing the classification performance of ML models. Further, parallel metaheuristics, deployed in a distributed environment, can optimize image segmentation processes by splitting the task across multiple computational units, thereby speeding up the process while maintaining or improving segmentation accuracy.

Bearing those in mind, we propose in this thesis, four hybrid methods focusing on feature selection, classification, and parallel image segmentation. The first method employs Grey Wolf Optimization (GWO) for selecting the most relevant features, followed by a Random Forest (RF) classifier to perform accurate classification. The second method integrates Correlation-based filtering with GWO to enhance the feature selection process, and applies various machine learning classifiers for comparative performance analysis. For parallel image segmentation, we designed two parallel approaches to improve efficiency and accuracy. The first is a Parallel Whale Optimization Algorithm (WOA) combined with K-Means clustering, implemented using multiprocessing to accelerate the segmentation process. The second method uses Grey Wolf Optimization in combination with Fuzzy C-Means (FCM), leveraging GPU acceleration for parallel execution. This approach significantly reduces computational time while maintaining high segmentation quality. All methods are evaluated using standard performance metrics. Our aim is to demonstrate the effectiveness of parallel metaheuristics and hybrid selection-classification strategies in medical image analysis.

In conclusion, this thesis shows that combining parallel computing, metaheuristic optimization, and machine learning offers an effective, accurate, and scalable solution to challenges in medical image analysis, contributing to the development of next-generation diagnostic systems.

Keywords: Distributed system, Parallel metaheuristic, Image segmentation, Classification, Grey Wolf Optimizer, Whale Optimizer Algorithm, Multiprocessing, Graphic Processing Unit.

Résumé

Ces dernières années, l'intégration de systèmes distribués avec des techniques de traitement parallèle a considérablement fait progresser le traitement d'images. Ces systèmes permettent de gérer efficacement de grands ensembles de données en répartissant les tâches de calcul sur plusieurs nœuds, améliorant ainsi la vitesse et l'évolutivité. La parallélisation du traitement d'images améliore les performances en exécutant plusieurs tâches simultanément, permettant ainsi de traiter des images haute résolution et des ensembles de données complexes en temps réel. Les algorithmes métaheuristiques ont été largement adoptés pour les tâches d'optimisation en traitement d'images en raison de leur capacité à explorer efficacement de vastes espaces de recherche. Couplés à des modèles d'apprentissage automatique (ML), ces algorithmes offrent des solutions performantes pour la sélection de caractéristiques dans les tâches de classification. Les métaheuristiques aident à identifier les caractéristiques les plus pertinentes parmi de grands ensembles de données, améliorant ainsi les performances de classification des modèles d'apprentissage automatique. De plus, les métaheuristiques parallèles, déployées dans un environnement distribué, peuvent optimiser les processus de segmentation d'images en répartissant la tâche sur plusieurs unités de calcul, accélérant ainsi le processus tout en maintenant, voire en améliorant, la précision de la segmentation.

Dans cette optique, nous proposons dans cette thèse quatre méthodes hybrides axées sur la sélection, la classification et la segmentation d'images parallèles. La première méthode utilise l'optimisation Grey Wolf (GWO) pour sélectionner les caractéristiques les plus pertinentes, suivie d'un classificateur Random Forest (RF) pour une classification précise. La seconde méthode intègre le filtrage par corrélation à GWO pour améliorer le processus de sélection des caractéristiques et applique divers classificateurs d'apprentissage automatique pour une analyse comparative des performances. Pour la segmentation d'images parallèles, nous avons conçu deux approches parallèles afin d'améliorer l'efficacité et la précision. La première est un algorithme d'optimisation Parallel Whale (WOA) combiné à un clustering K-Means, implémenté par multitraitement pour accélérer le processus de segmentation. La seconde méthode utilise l'optimisation Grey Wolf en combinaison avec Fuzzy C-Means (FCM), exploitant l'accélération GPU pour une exécution parallèle. Cette approche réduit considérablement le temps de calcul tout en maintenant une qualité de segmentation élevée. Toutes les méthodes sont évaluées à l'aide d'indicateurs de performance standard. Notre objectif est de démontrer l'efficacité des métaheuristiques parallèles et des stratégies hybrides de sélection-classification dans l'analyse d'images médicales.

En conclusion, cette thèse démontre que la combinaison du calcul parallèle, de l'optimisation métaheuristique et de l'apprentissage automatique offre une solution efficace, précise et évolutive aux défis de l'analyse d'images médicales, contribuant ainsi au développement de systèmes de diagnostic de nouvelle génération.

Mots-clés : Système distribué, Métaheuristique parallèle, Segmentation d'images, Classification, Algorithme d'optimization Grey Wolf, Algorithme d'optimisation Whale, Multitraitement, Graphic Processing Unit.

Acknowledgement

I would like to express my special appreciation and thanks to my supervisors Dr. DEBAKLA Mohammed and Prof. KHALIFA Djemal, not only for their tremendous support to my thesis writing, but also for the enlightenment on academic thinking which is the lifelong benefit. Without their consistent and illuminating instruction, this thesis could not have reached its present form.

I also want to say thanks to Prof. REBBAH Mohammed, Prof. SMAIL Omar Dr. MEKKAOUI Kheireddine and Prof. SALEM Mohammed, who gave me plenty of precious advice on my research.

Last, but not least, I appreciate the support of my family during my PH.D studies. My family has been a source of inspiration and strength for me whenever necessary, she has continuously provided me with her moral, spiritual, emotional, and financial support. Thanks to my brothers, sisters, relatives, mentors, friends, and classmates who expressed to me warm words of advice and encouragement to finish this study. I would also like to express my gratitude to anyone who has contributed, near or far, to the completion of this thesis.

Contents

Chapter 1 : Distributed system for image processing.

Chapter 2 : Metaheuristic for image processing

Chapter 3 : Machine learning and metaheuristic for image processing

Chapter 4 : Grey wolf optimizer for breast cancer classification

List of Figures

List of Tables

General Introduction

Image processing plays a fundamental role in a wide range of applications, including medical imaging, remote sensing, object recognition, and pattern analysis. Traditional image processing techniques often rely on handcrafted features and predefined algorithms, which struggle when confronted with complex patterns, noise, and variability in real-world images. With the emergence of machine learning (ML), particularly deep learning, the field has witnessed substantial advancements. ML models now enable automated, adaptive, and highly accurate image analysis by learning intricate patterns from large datasets. Techniques such as Convolutional Neural Networks (CNNs) [1], Support Vector Machines (SVMs)[2], and clustering algorithms like K-Means [3] and Fuzzy C-Means (FCM) [4] have demonstrated strong performance in tasks such as image segmentation, classification, and feature selection [5]-[10].

However, achieving optimal performance with ML techniques often requires careful selection of parameters, efficient feature extraction, and well-structured clustering, tasks that are inherently challenging due to high-dimensional data and the presence of irrelevant or redundant information. To address these challenges, metaheuristic algorithms have emerged as powerful tools for optimizing ML models. Inspired by natural and biological systems, metaheuristics like Grey Wolf Optimizer (GWO) [11], Ant Lion Optimizer (ALO) [12], Genetic Algorithms (GA)[13], and Whale Optimization Algorithm (WOA)[14] provide adaptive and robust search mechanisms that outperform traditional optimization methods in exploring complex, high-dimensional solution spaces. Their application in image processing has been particularly impactful in feature selection, hyperparameter tuning, and neural architecture optimization, especially in critical areas such as medical imaging where precision is vital.

Despite these advantages, one major limitation of metaheuristic algorithms is their computational cost, particularly when applied to large-scale image datasets or high-resolution medical scans. To overcome this bottleneck, parallel and distributed computing strategies have been increasingly adopted [15][16][17]. By leveraging multi-core processors, high-performance computing (HPC) infrastructures, and Graphics Processing Units (GPUs), researchers have been able to accelerate metaheuristic-driven image processing significantly. Parallel metaheuristics including parallelized swarm intelligence models, distributed genetic algorithms, and GPU-accelerated GWO or WOA have shown promising results in reducing execution time and enhancing scalability. In the context of medical image segmentation, this enables real-time or near real-time analysis of MRI or CT scans, supporting faster and more accurate diagnostic decisions.

By integrating ML with metaheuristics and leveraging the power of parallel computing, hybrid systems can achieve superior performance in image processing. These approaches are particularly beneficial in domains requiring high accuracy, robustness, and computational efficiency, such as tumor detection, organ segmentation, and disease classification in medical imaging.

Problem Statement

While traditional and metaheuristic-enhanced ML techniques have shown significant promise in medical image analysis, their scalability remains limited by computational constraints. Most segmentation and feature selection methods are implemented sequentially, which

becomes a performance bottleneck when applied to large datasets or high-resolution medical images. Moreover, current literature lacks comprehensive frameworks that integrate metaheuristic optimization, machine learning, and parallel computing to deliver both high accuracy and computational efficiency. This thesis addresses the need for such integrated and scalable solutions for feature selection, classification, and segmentation in medical imaging.

Objectives of the Thesis

This thesis aims to design, develop, and evaluate distributed and parallel approaches for medical image segmentation and classification based on metaheuristic algorithms. The main objectives are:

1. To study and implement metaheuristic algorithms for feature selection and image segmentation.
2. To integrate machine learning techniques with metaheuristic optimization to improve breast cancer classification.
3. To design and implement parallel and distributed versions of segmentation algorithms using multicore CPUs and GPUs.
4. To evaluate the effectiveness of the proposed methods in terms of segmentation accuracy, classification performance, and computational speed.

Major Contributions

The thesis makes the following contributions:

- **Feature Selection and Breast Cancer Classification Using Metaheuristics**
  Two feature selection methods are proposed based on the Grey Wolf Optimizer (GWO):
  1. A GWO-based feature selection combined with Random Forest (RF) classifier, applied to the WDBC dataset, which improves classification accuracy by selecting the most relevant features [18].
  2. A hybrid method that integrates Correlation-based Feature Selection (CFS) with GWO, followed by classification using Support Vector Machine (SVM), RF, and Naïve Bayes (NB)[19].

  These methods are validated through experimental results and published in international conferences and journals.

- **Parallel Image Segmentation Based on Metaheuristics**

  Two parallel segmentation frameworks are proposed:

  1. A multiprocessing-based approach that combines the Whale Optimization Algorithm (WOA) with K-means clustering, implemented on multicore CPUs to accelerate computation. This method was validated through experimental results and published in international conferences
  2. A GPU-accelerated segmentation approach named P-GWO-FCM, which parallelizes both GWO optimization and Fuzzy C-Means (FCM) clustering for

fast and accurate MRI segmentation.This GPU-based method is submitted to an international journal for publication.

Thesis structure

This dissertation is structured into five chapters:

- **Chapter 1** provides a general overview of image processing concepts and distributed systems, including a state-of-the-art review of distributed architectures in image processing.
- **Chapter 2** discusses metaheuristic algorithms in image processing, their optimization role in tasks such as segmentation and classification, and their parallelization strategies. It concludes with a review of parallel metaheuristic-based image segmentation techniques.
- **Chapter 3** focuses on the integration of machine learning and metaheuristics in image analysis, exploring common classifiers (SVM, RF, NB) and their synergy with metaheuristics for feature extraction and selection.
- **Chapter 4** presents two GWO-based feature selection methods for breast cancer classification, along with experimental validation using various machine learning classifiers.
- **Chapter 5** details two parallel segmentation methods: a multiprocessing-based WOA-KMeans hybrid algorithm and a GPU-accelerated P-GWO-FCM approach for MRI segmentation, highlighting performance improvements in accuracy and speed.

Chapter 1

Distributed system for image processing

1. Introduction

Image processing is a core field in computer science and engineering that focuses on performing operations on images to enhance, analyze, or extract meaningful information from them. It plays a critical role in various applications such as medical imaging, satellite data analysis, security and surveillance, industrial inspection, and machine vision. Image processing tasks include fundamental operations such as filtering, edge detection, segmentation, feature extraction, and classification. As image resolutions and dataset sizes have increased, especially with the advent of high-definition and hyperspectral imaging, processing such data in real-time or within a reasonable time frame has become a computationally demanding task. Traditional sequential processing methods often fail to meet these requirements, particularly when applied to large-scale or high-throughput image processing scenarios.

Distributed systems refer to a collection of independent computers that work together as a cohesive system to achieve a common goal. These computers, or nodes, communicate and coordinate their actions by passing messages over a network. The fundamental features of distributed systems include resource sharing, concurrency, fault tolerance, and scalability. In contrast to centralized systems, distributed systems offer improved performance and reliability by distributing workloads across multiple machines. They are widely adopted in modern computing environments, including cloud computing, edge computing, and high-performance computing clusters, where they support complex data-driven tasks such as real-time analytics, artificial intelligence, and large-scale simulations. Their ability to divide and parallelize tasks makes distributed systems ideal for solving problems that are too large or too slow to be addressed by a single machine.

In this chapter, we begin by introducing the fundamental concepts of image processing, with a particular focus on the need for parallelization. We then discuss distributed systems as a powerful platform for enabling parallel image processing. Finally, we review the existing literature on parallel and distributed approaches to image processing

2. Image representation

An image is a visual representation of an object, scene, or concept, typically captured or created using cameras, sensors, or graphic software. Images can be digital or analog, and they contain essential visual information that can be processed [20], analyzed, and interpreted by both humans and machines. In digital imaging, an image is composed of pixels, which are the

smallest units of a picture. Each pixel carries color and intensity information, forming a complete visual representation when arranged in a grid. Images can be categorized into various types, such as grayscale, binary, color, and multispectral images, depending on their composition and the number of color channels they contain. The definition and quality of an image depend on several factors, including resolution**,** bit depth, and compression methods. High-definition images contain more pixels and finer details, while low-definition images may appear pixelated or blurry. With advancements in technology, images are widely used in fields such as medical imaging, computer vision, remote sensing, and artificial intelligence, where they are analyzed for pattern recognition, segmentation, and classification.

An image is defined as a two-dimensional function, F(x,y), where x and y are spatial coordinates, and the amplitude of F at any pair of coordinates (x,y) is called the intensity of that image at that point. When x, y, and amplitude values of F are finite, we call it a digital image. In other words, an image can be defined by a two-dimensional array specifically arranged in rows and columns [20]. Digital Image is composed of a finite number of elements, each of which elements have a particular value at a particular location.These elements are referred to as picture elements, image elements, and pixels. A Pixel is most widely used to denote the elements of a Digital Image. There are four types of images:

1. Binary image: The binary image as its name suggests, contain only two pixel elements i.e 0 and 1, where 0 refers to black and 1 refers to white. This image is also known as Monochrome.
2. 8 bit color format: It is the most famous image format. It has 256 different shades of colors in it and commonly known as Grayscale Image. In this format, 0 stands for Black, and 255 stands for white, and 127 stands for gray.
3. 16 bit color format: It is a color image format. It has 65,536 different colors in it. It is also known as High Color Format. In this format the distribution of color is not as same as Grayscale image.A 16 bit format is actually divided into three further formats which are Red, Green and Blue. That famous RGB format.

3. Image processing

Image processing is the technique of using computational methods to manipulate, analyze, and interpret visual information in digital images. It involves a series of operations to enhance image quality, extract meaningful features, segment objects, and derive useful insights from visual data. In essence, image processing transforms raw image data (captured from sensors or cameras) into a form that is easier to analyze or visually appealing, enabling tasks such as object recognition, pattern detection, and scene interpretation. Applications span across fields such as medical imaging, computer vision, robotics, remote sensing, and digital photography.
Image Processing refers to the manipulation and analysis of digital images using algorithms and techniques to extract, enhance, or modify information from them. It involves applying mathematical and computational methods to improve the visual appearance of images or to extract useful features for tasks such as recognition, detection, and classification. The goal of image processing can be to improve image quality, enhance specific features, or prepare the

image for further analysis in fields such as medical imaging, computer vision, and remote sensing [20].

3.1. Image acquisition

Image acquisition is the process of capturing or obtaining a digital image from a physical scene [21], usually through a camera, scanner, or specialized sensor. This is the first step in the image processing workflow, where real-world visual information is converted into a digital format for analysis and manipulation.In image acquisition, devices may capture various types of images, such as grayscale, color, infrared, or multispectral, depending on the application. The quality and resolution of the acquired image play a significant role in the effectiveness of subsequent image processing tasks.

3.2. Image preprocessing

There are several methods commonly used for image preprocessing to enhance image quality and prepare data for analysis.. Each method plays a vital role in improving the performance of subsequent image processing or machine learning tasks.

3.2.1. Resizing

Resizing an image is the process of changing its dimensions, either width, height, or both while maintaining or adjusting the original aspect ratio. This operation involves either reducing (downsampling) or increasing (upsampling) the number of pixels in the image, which affects its resolution and quality.

3.2.2. Noise reduction

Noise reduction in image segmentation involves preprocessing an image to remove or minimize unwanted artifacts (noise) that can interfere with accurate segmentation. Noise can obscure boundaries, alter pixel values, and generally reduce the quality of segmentation results. Effective noise reduction ensures that the segmented regions correspond accurately to real structures or objects in the image, leading to clearer and more reliable segmentation (Gaussian Blur, Median Filter, Bilateral Filter,Non-Local Means, Wavelet Denoising, Anisotropic Diffusion (Perona-Malik Filtering)).

3.2.3. Normalization

Image normalization is the process of adjusting the range of pixel intensity values in an image to a standard scale. This technique enhances contrast and prepares images for more consistent and accurate analysis, especially in tasks like image classification, segmentation, and feature extraction.

3.2.4. Binarization

Binarization is the process of converting a grayscale or color image into a binary image, where each pixel is assigned one of two values: typically 0 (black) or 1 (white). This technique is commonly used in image processing to simplify images, making objects and backgrounds easier to distinguish by reducing the image to only two intensity levels (Global Thresholding, Adaptive Thresholding, Sauvola and Niblack Methods, Iterative and K-Means Thresholding, Deep Learning-Based Binarization).

3.2.5. Contrast enhancement

Contrast enhancement is an image preprocessing technique used to improve the visibility of features in an image by expanding the range of intensity values. Enhanced contrast makes objects, edges, and details more distinguishable, which is particularly valuable for applications in segmentation, object detection, and feature extraction (Histogram Equalization, Adaptive Histogram Equalization, Linear Contrast Stretching (Normalization), Gamma Correction…).

3.3. Image segmentation

Image segmentation is a fundamental technique in digital image processing and computer vision. It involves partitioning a digital image into multiple segments (regions or objects) to simplify and analyze an image by separating it into meaningful components, which makes the image processing more efficient by focusing on specific regions of interest [22]. Figure 1 represents image segmentation methods.

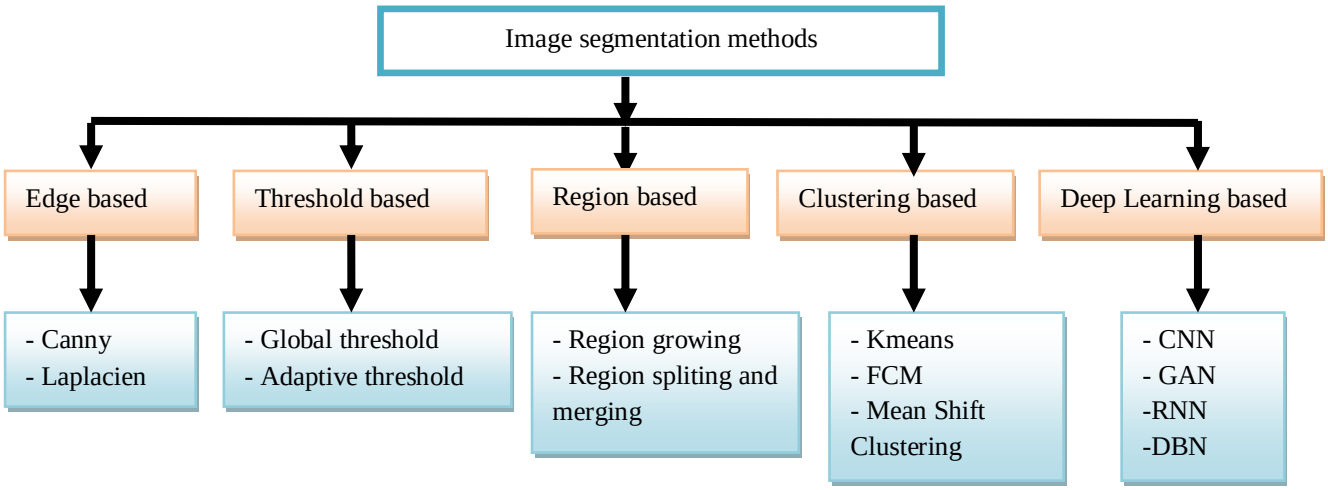


Figure 1. Image segmentation methods.

3.3.1. Edge based image segmentation

Edge-based image segmentation is a technique that divides an image into regions based on the detection of edges, which are the boundaries between different objects or regions within the image. An edge is a significant change in pixel intensity, often marking a shift in texture, color, or brightness. By identifying these boundaries, edge-based segmentation can outline objects or distinct regions effectively. This method usually involves applying edge-detection

algorithms such as Canny [23], or Laplacian of Gaussian [24] to highlight edges, which are then used to segment the image [25]. Edge-based segmentation is widely used in applications where clear boundaries are essential, such as medical imaging, object detection, and feature extraction.An example of edge-based segment is shown in Figure 2.

(a) (b)

Figure 2. Edge-based segment (a) original image, (b) the resulting image after segmentation.

3.3.2. Threshold based segmentation

Threshold-based segmentation is an image segmentation technique that partitions an image by grouping pixels based on their intensity values. It operates by selecting one or more threshold values to separate objects from the background or different regions within the image. Pixels with intensity values above a set threshold might represent objects, while those below might represent the background. This approach works well when there's a clear contrast between objects and the background.In literature there are several types of thresholding [26], Including Global Thresholding (Uses a single threshold value for the entire image, ideal for uniformly lit images), Adaptive Thresholding [27] (Calculates threshold values for smaller regions, which is useful for images with uneven lighting), and Otsu's Method [28] (An automated global thresholding technique that calculates an optimal threshold by minimizing the variance within regions).Threshold-based segmentation is widely used in various domain such as Medical imaging, to isolate organs or tissues, Document processing, to separate text from the background, and Industrial inspection, for defect detection.

For example: we have threshold level 128 so that it decide that all the pixels having intensity value greater than 128, it belong to some regions and those intensity values less than 128, it belong to some other region. Let an image be f(x, y). Suppose that this image consists the dark object against the bright background or viceversa.Therefore, intensity concentrate mainly on two regions,one towards the darker side (or lower intensity) and othertowards the brighter side (or higher intensity). The histogram with two peaks and valley at the bottom as shown in figure 3. Where T is called the threshold value. Figure 4, present leukemia image

segmentation using multi-thresholding method, where (a) represent the original image, and (b) represent the segmented image.

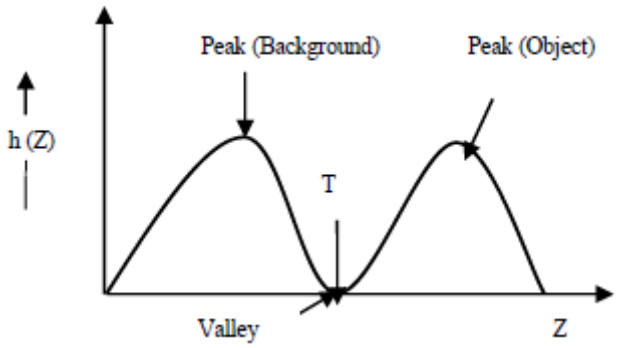
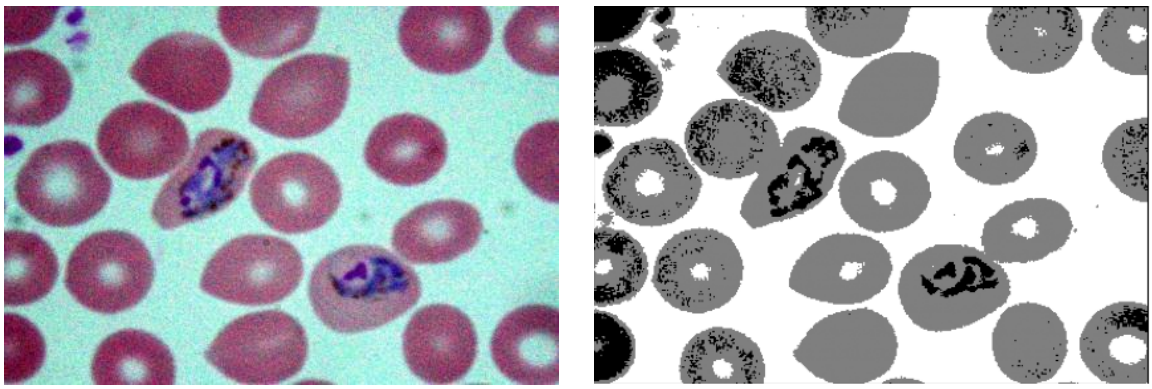


Figure 3. Histogram with two peaks and one valley [29].



(a) (b)

Figure 4. Leukemia image segmentation using multi-thresholding method, where (a) represent the original image, (b) the segmented image.

3.3.3. Region based image segmentation

Region-based image segmentation is a technique that segments an image by grouping neighboring pixels into regions based on predefined criteria, such as similarity in intensity, color, or texture. This approach assumes that pixels within a particular region are more similar to each other than to those in other regions. Region-based segmentation is widely used for medical imaging, such as segmenting organs or lesions. Satellite and aerial imagery, for analyzing terrain or vegetation. And Object detection in complex images. Figure 5, represent a region-based segmentation where (a) original image, (b) the resulting image after segmentationthe segmentation process.Below are some foundational methods in region-based segmentation:

- Region Growing [30]: Region growing starts with seed points and expands regions by adding neighboring pixels with similar properties until no more pixels meet the

similarity criterion. This method is intuitive and effective but can be sensitive to noise and initial seed selection.

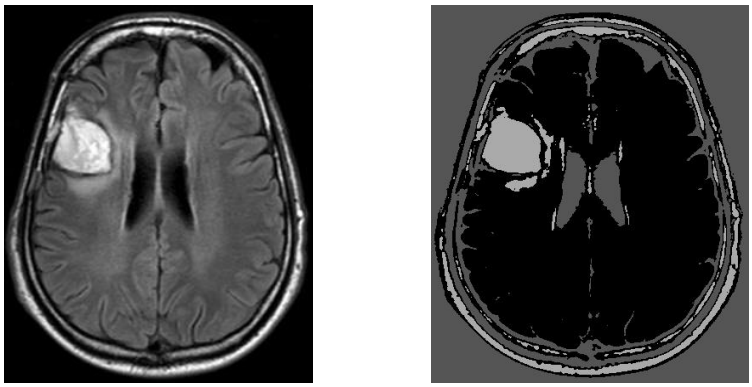
- Watershed Segmentation (Region-Based) [31]: A popular technique for separating touching objects by viewing image intensities as topographical surfaces. Watershed segmentation identifies “watershed lines” as boundaries between regions.
- Region Splitting and Merging [32]: This method recursively splits an image into regions based on homogeneity and merges similar adjacent regions, ensuring each final segment is homogenous.
- Markov Random Fields (MRF) [33]: MRF is a probabilistic model where each pixel’s label depends on neighboring labels, effectively capturing spatial dependencies. This model is commonly used in medical imaging.

Figure 5. Region-based segmentation, (a) original image, (b) the resulting image after segmentation [34].

3.3.4. Clustering based image segmentation

Clustering-based image segmentation is a technique that divides an image into segments by grouping pixels into clusters based on their similarity in features like color, intensity, or texture. This unsupervised approach relies on clustering algorithms to categorize pixels so that similar pixels belong to the same segment, while dissimilar ones are placed in separate segments. Clustering-based segmentation is particularly effective when there’s no prior knowledge of the image’s content or the number of objects present.

- K-Means Clustering [3]: One of the most popular clustering algorithms, K-Means groups pixels based on similarity in feature space (e.g., color or intensity) by iteratively updating cluster centroids. It is simple and effective for basic image segmentation tasks. Clustering-based segmentation is widely used for: Medical imaging, to cluster similar tissues or structures. Remote sensing, to classify land use or vegetation types. Image compression, to group similar pixels and reduce data size. Figure 6 show an exemple of segmentation using kmeans with number of cluster k=3



(a) (b)

Figure 6. Image segmentation using kmeans where (a) represent the original image and (b) represent the segmented image (number of cluster k=3).

- Fuzzy C-Means (FCM) Clustering [4]: An extension of K-Means, FCM allows partial membership of pixels to multiple clusters, which makes it more robust to noise and better suited for images with fuzzy boundaries.
- Mean Shift Clustering [35]: Mean Shift is a non-parametric clustering technique that identifies clusters by locating peaks in a density function. It's particularly useful for segmenting images with complex or multi-modal distributions.

3.3.5. Deep learning based image segmentation methods

Deep Learning (DL) has significantly advanced the field of image segmentation by enabling automatic, highly accurate pixel-wise classification of images [1], [36], [37]. Unlike traditional segmentation methods that rely on manual feature extraction or predefined rules, deep learning models, particularly Convolutional Neural Networks (CNNs), can learn complex, hierarchical features directly from the raw image data. This ability to learn spatial patterns and textures has made DL methods the go-to approach for many segmentation tasks, such as object detection, medical image analysis, and remote sensing. Architectures like U-Net [1], Fully Convolutional Networks (FCNs) [38], and Mask R-CNN have revolutionized segmentation by improving accuracy, robustness, and the ability to segment complex structures. Deep learning methods, especially those leveraging large labeled datasets and advanced architectures, are capable of generalizing well to a wide range of segmentation tasks, offering solutions that can adapt to new domains with minimal human intervention. As a result, DL has become indispensable in both research and industry for tasks requiring precise segmentation, even in challenging or noisy environments.

3.3.5.1. Convolutional Neural Networks (CNNs) for image segmentation

Convolutional Neural Networks (CNNs) are a class of deep learning algorithms designed to process and analyze visual data, particularly images. CNNs use convolutional layers to automatically detect patterns, features, and hierarchical structures in images, making them highly effective for image segmentation tasks. In image segmentation, CNNs classify each pixel in an image, assigning it a specific label (e.g., background or foreground), which

allows for precise segmentation of objects or regions in the image. In following an Example in Segmentation:

- U-Net [1]: A CNN architecture commonly used for semantic segmentation, where the image is divided into regions corresponding to different classes (such as foreground and background).
- Fully Convolutional Networks (FCNs) [38]: An extension of CNNs where the fully connected layers are replaced with convolutional layers to allow for pixel-wise classification in segmentation tasks.

3.3.5.2. Generative Adversarial Networks (GANs) for Image Segmentation

Generative Adversarial Networks (GANs) [39] consist of two neural networks: a generator and a discriminator. The generator creates synthetic data (such as segmented images), while the discriminator evaluates the authenticity of the generated data against real data. In the context of image segmentation, GANs can be used to generate high-quality segmentation masks or to improve segmentation performance through adversarial training. The generator tries to improve the segmentation mask it produces, while the discriminator ensures the result is realistic and accurate, pushing the model toward better segmentation outcomes. In following an Example in Segmentation [40]**:**

- Pix2Pix: A GAN-based model where the generator generates segmentation maps from input images, and the discriminator ensures that the generated segmentation masks are realistic.
- CycleGAN: Used in unpaired image-to-image translation tasks, such as generating segmentation masks without needing paired training data.

3.3.5.3. Recurrent Neural Networks (RNNs) for Image Segmentation

Recurrent Neural Networks (RNNs) are a class of neural networks designed to model sequential data by maintaining a memory of previous states [41]. While RNNs are traditionally used in tasks like natural language processing, they have been applied to image segmentation when there is temporal or spatial context that needs to be captured over a sequence of frames or pixels. For example, RNNs can be used in video segmentation, where each frame's segmentation can be influenced by previous frames or in situations where pixel dependencies (such as neighboring pixels) are crucial to achieving accurate segmentation. In following an Example in Segmentation:

- Long Short-Term Memory (LSTM) [42]: An advanced type of RNN that can capture long-range dependencies, making it effective for segmenting sequential images, video frames, or spatially dependent pixel information in images.
- CRNNs (Convolutional RNNs): Combining CNNs for feature extraction with RNNs for sequence modeling, particularly useful for sequential image segmentation tasks.

3.3.5.4. Deep Belief Networks (DBNs) for Image Segmentation

Deep Belief Networks (DBNs) [43] are a type of probabilistic generative model made up of multiple layers of stochastic, latent variables. They are composed of multiple Restricted Boltzmann Machines (RBMs) stacked together, where each RBM learns to model the data at a different level of abstraction. In the context of image segmentation, DBNs can be used to learn complex, hierarchical representations of the image, which can then be applied to segment objects or regions in the image. DBNs are typically used in scenarios where unsupervised feature learning is necessary before applying a discriminative model for segmentation.In following an Example in Segmentation:

- Feature Learning with DBNs: DBNs can be trained on image data to learn features that are then used to improve the segmentation of objects in images, particularly when there is limited labeled data available.
- DBN + SVM: A hybrid model where the DBN is used for unsupervised feature learning and the learned features are classified using a Support Vector Machine (SVM) to perform segmentation.

3.4. Feature extraction

Feature extraction is the process of transforming raw data into a set of measurable characteristics (features) that can be used to represent the essential aspects of that data. In the context of image processing, feature extraction involves identifying and isolating relevant information (such as color, texture, shape, or edges) from an image to simplify further analysis or machine learning tasks. The goal of feature extraction is to reduce the complexity of the data while preserving meaningful patterns that can aid in tasks like classification, recognition, and segmentation. Effective feature extraction enhances a model's accuracy, speeds up processing, and can improve its ability to generalize to new data. Feature extraction is used for in several domain like image classification, object detection, medical imaging, and facial recognition. In littérature, there are various techniques for feature extraction such as Gray-Level Co-occurrence Matrix (GLCM)[44], Local Binary Patterns (LBP)[45], Histogram of Oriented Gradients (HOG)[46], and Principal Component Analysis (PCA)[47].

3.5. Feature selection

Feature selection is a crucial preprocessing step in machine learning that aims to identify the most relevant and informative features from high-dimensional datasets while eliminating redundant or irrelevant ones. By selecting the optimal subset of features, feature selection improves model performance, reduces computational cost, and enhances interpretability. This process is particularly important in applications such as medical diagnostics, image processing, text classification, and bioinformatics, where datasets often contain hundreds or thousands of features, many of which may be noisy or non-contributory. Feature selection techniques can generally be categorized into three main approaches: filter methods, wrapper

methods, and embedded methods. Researchers have extensively explored a range of feature selection methods through various studies, as evidenced in the literature [5-10][48].

3.5.1. Filter Methods

Filter methods evaluate the importance of features based on statistical techniques without involving any specific machine learning model. These methods assess feature relevance by computing correlation coefficients, information gain, mutual information, or statistical tests. Popular filter techniques include:

- Correlation-based Feature Selection (CFS): Measures the correlation between input features and the target variable, removing highly correlated redundant features.
- Chi-Square Test: Evaluates the dependency between categorical features and the target class, commonly used in text classification.
- Information Gain (IG): Determines the amount of information a feature contributes to the class label, widely applied in decision trees.
- Principal Component Analysis (PCA): Although technically a dimensionality reduction technique rather than a selection method, PCA transforms features into uncorrelated components while retaining most of the data variance.

Filter methods are computationally efficient and work well for high-dimensional datasets, but they may not always select the optimal feature subset for a specific learning algorithm.

3.5.2. Wrapper Methods

Wrapper methods evaluate subsets of features by training and testing a machine learning model iteratively, selecting the subset that yields the best performance. These methods use search strategies such as forward selection, backward elimination, and recursive feature elimination (RFE). Common wrapper techniques include:

- Sequential Forward Selection (SFS): Starts with an empty feature set and iteratively adds features that improve model accuracy.
- Sequential Backward Selection (SBS): Begins with all features and removes the least significant ones step by step.
- Recursive Feature Elimination (RFE): Uses a model (e.g., SVM or Random Forest) to rank features and remove the least important ones iteratively.

Wrapper methods often achieve high accuracy but can be computationally expensive, especially for large datasets.

3.5.3. Embedded Methods

Embedded methods incorporate feature selection directly into the training process of machine learning models. These methods leverage regularization techniques to penalize less

important features, leading to automatic selection. Some widely used embedded techniques are:

- LASSO (Least Absolute Shrinkage and Selection Operator): A regression-based method that applies L1 regularization to shrink the coefficients of less significant features to zero, effectively removing them.
- Decision Tree-Based Methods: Tree algorithms like Random Forest and Gradient Boosting assign importance scores to features and allow pruning of irrelevant ones.
- Elastic Net: Combines LASSO (L1 regularization) and Ridge Regression (L2 regularization) to enhance feature selection in high-dimensional data.

Embedded methods strike a balance between efficiency and accuracy, making them suitable for real-world applications like medical diagnosis and fraud detection.

3.6. Classification

Image classification is a critical step in image processing where an algorithm assigns a category or label to an image based on its visual characteristics. It is widely used in various fields, including medical imaging, autonomous driving, security surveillance, and satellite imagery analysis. The goal of classification is to enable computers to recognize patterns and categorize images into predefined classes. This process typically involves extracting features from images and using machine learning or deep learning models to analyze and classify them [49].

4. Distributed systems

Parallelization refers to the process of breaking down a computational task into smaller sub-tasks that can be executed simultaneously across multiple processing units [50]. This process significantly accelerates computations, especially when handling large data sets or performing complex operations, such as scientific simulations, machine learning, and image processing. Parallelization can occur at various levels [50], including task-level parallelism, where distinct tasks are executed simultaneously, and data-level parallelism, where individual elements of a data set are processed concurrently. The primary advantage of parallelization is the significant reduction in processing time, enabling systems to handle complex computations more efficiently. In addition, parallel systems are scalable, allowing for greater computational power as more processors or cores are added. By utilizing available resources effectively, parallelization enhances overall system efficiency and can also provide fault tolerance by ensuring that tasks are redundantly executed across different processors. However, parallelization comes with limitations, such as the complexity of designing parallel algorithms, which may require careful consideration of dependencies between tasks and ensuring efficient synchronization and communication between processors. Additionally, the overhead from inter-process communication and the diminishing returns seen in performance when scaling up (due to communication bottlenecks and memory access limitations) can limit the effectiveness of parallel systems. The design of parallel algorithms must also take into

account data dependencies, as certain tasks are inherently sequential and cannot be parallelized.

A distributed system is a collection of independent computers that work together to achieve a common goal. These systems enable resource sharing, parallel processing, and fault tolerance by distributing tasks among multiple computing nodes. Unlike centralized systems, where a single machine handles all tasks, distributed systems improve scalability, performance, and reliability by coordinating multiple processors. They are used in various applications such as cloud computing, scientific simulations, big data analytics, and artificial intelligence. However, distributed systems also present challenges, such as maintaining consistency, managing network communication, and handling failures effectively.

Distributed systems operate by enabling multiple computing units, known as nodes, to communicate and collaborate over a network to achieve a common goal. Each node processes a portion of a task, and the system ensures synchronization, consistency, and coordination among them. Communication typically occurs through message passing or shared memory, allowing data to be exchanged efficiently. Middleware acts as a bridge between nodes, managing task allocation, load balancing, and failure recovery. One of the major advantages of distributed systems is their ability to provide high availability and fault tolerance, if one node fails, others can take over. However, a key challenge lies in maintaining consistency and synchronization across multiple nodes, as network failures and data replication conflicts can lead to system inconsistencies.

4.1. Flynn’s classification of parallel machines

There are several ways to classify parallel machines[50]. However, one classification has been widely used since 1966, namely Flynn's Taxonomy [51]. This classification distinguishes parallel architectures based on two independent parameters: instructions and data: each of these two parameters can have two possible states: Single or Multiple. Table 1 illustrates Flynn's classification.

	Single Data	Multiple Data
Single Instruction	SISD	SIMD
Multiple Instruction	MISD	MIMD

Table 1. Flynn classification of parallel machine.

Single instruction, single data (SISD)

A sequential machine that can execute only a single instruction stream in a single CPU clock cycle. Furthermore, only one data stream is used as input per clock cycle. Program execution is deterministic, and it is the oldest and most widespread type of machine today.

Single instruction, multiple data (SIMD)

This is a type of parallel machine whose processors execute the same instruction in a given clock cycle. However, each processing unit can operate on a different data element. This type of machine is well-suited for regular problems such as image processing and graphics

rendering. Program execution is synchronous and deterministic. Furthermore, the majority of current workstation processors and graphics processing units include a specialized SIMD processing unit, known as SWAR (SIMD Within A Register).

Multiple instruction, single data (MISD)

A single data stream feeds multiple processing units, and each processing unit operates on the data independently using a stream of independent instructions. Hennessy and Patterson in [52] state that no such machines have been designed, while Flynn in his 1996 article [53] classifies systolic architectures [54] in this category.

Multiple instruction, multiple data (MIMD)

This is currently the most common type of parallel machine. Each processor in these machines can execute a different instruction stream and operate on a different data stream. Execution can be synchronous or asynchronous, deterministic or nondeterministic. Examples include current supercomputers, networked clusters of parallel machines, computing grids, Symmetric Multi-Processors (SMPs), and multi-core processors. In addition, many of these machines contain SIMD processing units.

4.2. Memory Architectures of Parallel Machines

In the following, we classify parallel machines according to the type of their memory hierarchy[50]. This classification allows us to distinguish parallel machines from a perspective other than that of the CPU and also provides a better understanding of the motivations behind programming models for parallel machines.

4.2.1. Shared Memory Parallel Machines

There are several variants of these machines, but they all share a common property: the ability for all processors to access memory as a global address space. Thus, multiple processors can operate independently but share the same memory resource. A change made by one processor to a memory location is visible to all other processors. This class of machines can be divided into two subclasses based on memory access times: UMA (Figure 7) and NUMA (Figure 8).

Uniform Memory Access

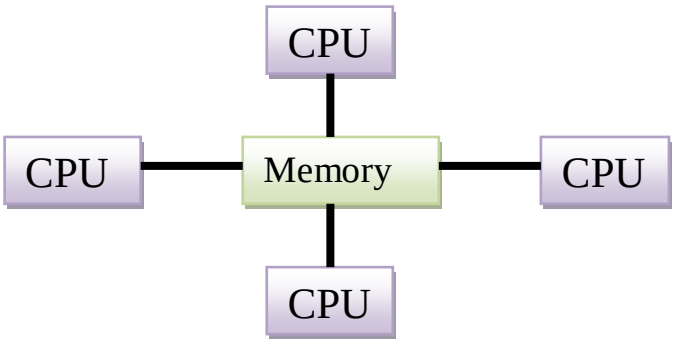


Figure 7. UMA Shared memory parallel machine.

These are mainly SMP-type machines that have multiple identical processors and can access memory equally and at the same time. They are sometimes referred to as CC-UMA (Cache Coherent UMA). Cache coherence means that if one processor updates a memory location, all other processors are aware of this change. This functionality is ensured at the hardware level.

Non-UMA

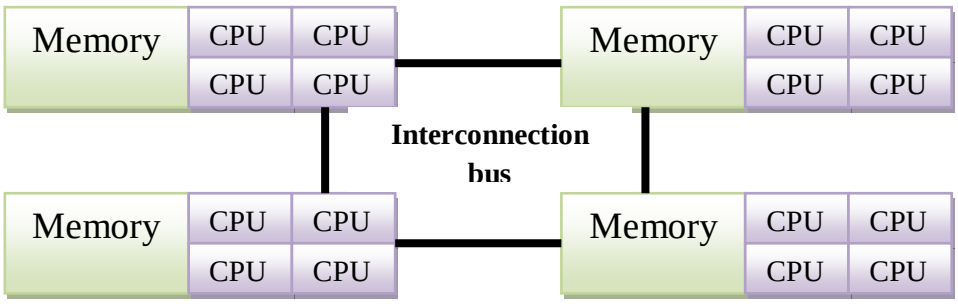


Figure 8. Non-UMA Shared memory parallel machine.

This type of machine is often designed by connecting two or more SMPs. One SMP can have direct access to the memory of another SMP. Access times to a given memory are not the same for all processors, and when a node is traversed, access is slower. If cache coherence is guaranteed, this is called CC-NUMA.

4.2.2. Distributed Memory Parallel Machines

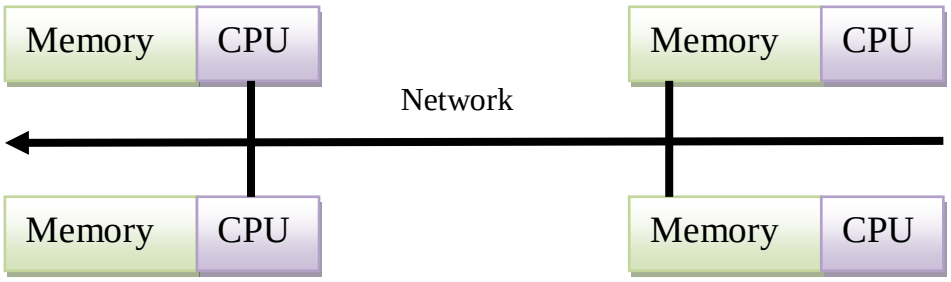


Figure 9. Parallel machines with distributed memory.

Like shared memory machines, distributed memory machines vary, but they share one thing in common: they require a communication network to connect the processors memories as we see in Figure 9. Each processor has its own local memory. The memory addresses of a given processor do not correspond to those of another, and therefore the concept of global memory does not exist. Since each processor has its own private memory, it operates independently. Indeed, any change made to its local memory has no effect on the memory of other processors, which precludes the concept of cache coherence. When a processor needs data from another processor's memory, the programmer is responsible for defining when and how the data is transferred. The programmer is also responsible for synchronization.

4.2.3. Hybrid Memory Parallel Machines

The fastest machines in the world employ so-called hybrid memory architectures (Figure 10), which combine the two previous types: shared and distributed. The shared memory

component is often an SMP machine. The distributed component consists of networking multiple SMP machines. The different SMPs can only address their own memory, and data

transfer between two SMPs requires network communications. The major difference between this type of architecture and NUMA SMPs is that the memory space is not shared, and inter-processor communication takes place over an interconnection network such as Ethernet or Infiniband.

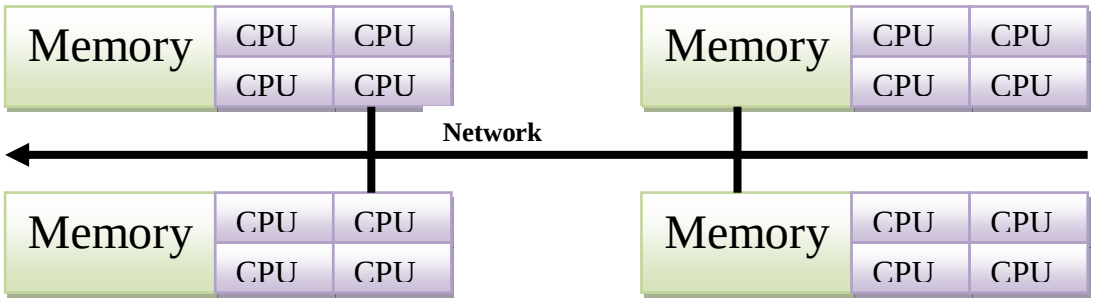


Figure 10. Hybrid memory parallel machine.

4.3. Parallel Programming Models

There are several programming models for parallel machines. These models exist at a level of abstraction above the hardware and memory architecture. Although at first glance, programming models are closely linked to the machine architecture, they are assumed to be implementable on any parallel machine, regardless of its characteristics. There is no ideal programming model, but some programming models are well-suited for a given application on a given machine[50]. Below, we describe the main parallel programming models.

4.3.1. The Shared Memory Model

In this programming model, tasks share a common address space to which they can read and write data asynchronously. Several mechanisms, such as locks and semaphores, can be used to control access to shared memory. This programming model is simplified from the user's perspective because there is no notion of data ownership by a task, which avoids explicit communications to transfer data from one task to another. However, in terms of performance, this last point is a disadvantage because it generates additional memory access, cache refresh, and bus traffic when multiple processors use the same data. Implementations of this model on shared memory machines are limited to the native compiler, which translates program variables into global memory addresses. However, there is no implementation of this model on distributed memory machines.

4.3.2. The Threaded Programming Model

In the threaded programming model, a single process can have multiple, concurrent execution paths. This concept can be thought of as a main program that includes a number of subroutines. The main program is scheduled for execution by the operating system, and it acquires all the system resources necessary for its execution. It then executes a set of instructions serially and creates a number of tasks (threads) that can be scheduled and executed concurrently by the OS. Each thread owns its local data but also shares the main

program's resources with other threads. Each thread has access to global memory because it shares the main program's address space. A thread's workload can be considered a subroutine of the main program, but it can run in parallel with another thread. Threads communicate with each other via global memory, which requires synchronization operations to guarantee exclusive access to a given location at a given time for a single thread.

Threads have variable lifetimes and can be created and destroyed throughout the program. The threaded programming model is often associated with shared-memory machines. Thread implementations typically include a library of functions or a series of directives buried within the parallel code. In both cases, the user is responsible for defining parallelism. There are several thread implementations, and most manufacturers have developed their own versions, which have affected the portability of parallel code. However, a standardization effort has given rise to two implementations that have become the standard today: POSIX threads [55] and OpenMP [56].

4.3.3. The Message Passing Programming Model

In this model, parallel programming is done by message passing. A set of tasks uses its own local memory during computation. Multiple tasks can reside on the same physical machine or on an arbitrary number of machines. Tasks exchange data through communications by sending and receiving messages. Data transfers require cooperative operations to be performed by each process. For example, a send operation must have a dual receive operation. Message Passing implementations take the form of a library of subroutines, and the programmer is responsible for detecting parallelism. As with any library, multiple versions have been developed, leading to compatibility issues. In 1992, the MPI Forum was founded with the goal of standardizing Message Passing implementations, including PVM [57]. Two standards were then developed: MPI [58] in 1994 and MPI-2 in 1996. Today, MPI is the most widely used programming model for message passing. In MPI implementations on shared-memory architectures, network communications are simply replaced by memory copies.

4.3.4. The Data Parallel Model

This model is based on data parallelism, which focuses parallel work on a set of data contained in an array or a multi-dimensional data structure. A set of tasks work collectively on the same data structure, but each task operates on a different partition of this structure. The tasks all perform the same operation on their data partition. On shared-memory architectures, all tasks can access the data structure via global memory. However, when the memory architecture is distributed, the data is divided into chunks that reside in each task's local memory. Programming with this model is generally done by writing code with data parallel constructs. These constructs can take the form of calls to library functions or directives recognized by a data parallel compiler. Implementations of this model are often in the form of compilers or extensions to them. Examples include Fortran compilers (F90 and F95) and their HPF (High Performance Fortran) extension [59], which support data parallel programming. HPF includes directives that control data distribution, assertions that can improve the optimization of the generated code, and data parallel constructs. Implementations of this

model on distributed memory architectures take the form of a compiler that converts standard code into Message Passing In (MPI) code, which distributes data across different processors, all transparently from the user's perspective. Despite its early popularity, HPF has not achieved the expected success, as evidenced by the analysis of its main author in [60].

4.3.5. Stream Computing Programming Model

This model, commonly called stream computing, is based on data parallelism. A single computing kernel is applied to a set of data. This model is the dominant model for graphics computing units. It is a model where parallelism is of the SIMD type, or multiple computing units (typically hundreds) execute the same instruction on a set of data in parallel. Machines supporting this type of model are GPUs, FPGAs, and certain specialized processors such as Stanford's Imagine [61] and Merrimac [62]. Several languages have also been developed to support this type of hardware, including StreamIt [63] and Brook [64]. CUDA [65] and OpenCL [66] are also implementations of this parallel programming model and are by far the most widely used today. The model consists of simplifying both the hardware and restricting the type of parallelism used.

4.4. Distributed system techniques

Distributed systems employ various techniques to achieve efficient parallel computation, scalability, and high performance. Below some distributed system :

4.4.1. High-Performance Computing (HPC)

HPC involves the use of supercomputers and clusters to perform complex computations at high speeds [67]. These systems utilize parallel processing techniques, where multiple processors work together on large-scale problems such as climate modeling, molecular simulations, and AI training. HPC clusters typically employ MIMD architectures and rely on frameworks like MPI and OpenMP for efficient task execution. While HPC provides unmatched computational power, it requires specialized hardware, high energy consumption, and complex software management.

4.4.2. Cloud Computing

Cloud computing offers on-demand access to computing resources such as servers, storage, and databases over the internet. It provides a scalable and cost-efficient alternative to traditional on-premise computing. Cloud computing has emerged as a transformative paradigm with the potential to revolutionize the implementation and delivery of IT services [68]. It offers a model that enables ubiquitous, convenient, and on-demand network access to a shared pool of configurable computing resources such as networks, servers, storage, applications, and services with minimal management effort or direct interaction with service providers [69]. Based on the nature of services delivered, cloud service providers are commonly categorized into three primary models: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) [70].

- **Infrastructure as a Service (IaaS):** Virtualized computing resources, including VMs and storage.
- **Platform as a Service (PaaS):** Development platforms that provide pre-configured environments for application deployment.
- **Software as a Service (SaaS):** Cloud-hosted software applications accessible via web browsers.

  Cloud computing reduces infrastructure costs and increases flexibility but presents challenges such as security risks, data privacy concerns, and vendor lock-in.

4.4.3. Graphic Processing Unit

GPU is a powerful multicore processor. GPUs have high-performance processing units for graphics processing. Initially, GPUs were designed to accelerate graphics rendering. They are currently used to parallelize general-purpose computations to reduce application runtime [71].GPUs are highly suited to implementing program execution with various data elements. This technique is referred to as data parallelism. Data parallelism distributes data components to parallel threads on GPUs.Data parallelism is most commonly used in 3D rendering, stereo vision, pattern recognition, image, video, and medical applications [72].

A significant performance difference exists between GPU and general-purpose multi-core CPU. Figure 11 shows an architectural comparison between CPU and GPU. CPUs are optimized for sequential programming. It uses advanced control logic to execute instructions from a single thread in parallel or out of sequential sequence while keeping the illusion of sequential execution.GPUs typically have several CPU cores, ALUs, control units, and memory types [72]

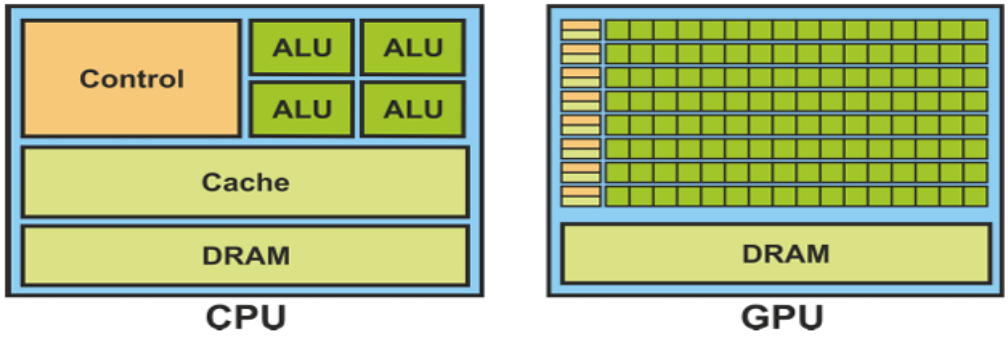


Figure 11. Comparison CPU vs GPU from source [73].

The GPU is a multi-core architecture that improves intense computing and frees up CPU resources. A GPU consists of global memory and streaming multiprocessors (SMs). Each SM includes a group of streaming processors (SP) connected to a local memory (register memory). SPs in an SM are linked to a shared memory [74, 75]. The architecture of GPUs require specif programming languages such as: OpenCL, OpenMP, OpenACC and CUDA [76]. The Compute Unified Device Architecture (CUDA) is a powerful hardware and software architecture for managing computations on GPUs. It treats the GPU as a data-parallel computing device, eliminating the requirement to translate computations to the graphics pipeline [77]. GPU computations are programmed as kernel functions. A kernel program defines the execution of a serial thread on a GPU. The host CPU launches the kernel

with given numbers of blocs and threads. A bloc represents a set of a specific number of threads, and all blocs in that kernel launch have the same number of threads.

4.4.4. Multiprocessing

Multiprocessing is a technique where multiple processors or cores within a single machine execute tasks simultaneously, improving computational performance and efficiency. It can be classified into Symmetric Multiprocessing (SMP), where all processors share the same memory and workload, and Asymmetric Multiprocessing (AMP), where one processor controls task distribution while others execute assigned processes. Additionally, multithreading enables multiple threads within a single process to execute concurrently, improving system responsiveness. The key advantage of multiprocessing is that it maximizes CPU utilization and speeds up task execution. However, it also introduces challenges such as increased complexity in task scheduling, potential race conditions, and overhead in managing shared resources among multiple processors.

4.4.5. MapReduce and Hadoop

MapReduce is a programming model for processing and generating large datasets with a parallel, distributed algorithm on a cluster. Hadoop, an open-source framework, implements the MapReduce model and provides a scalable and fault-tolerant system for distributed data processing. In the context of image processing, Hadoop and MapReduce are used to split large image datasets into smaller chunks that can be processed independently across different nodes in a cluster. The "Map" phase processes each chunk, such as applying filters, extracting features, or transforming image formats,while the "Reduce" phase aggregates the results, such as compiling extracted features or stitching processed image parts. This technique is particularly useful for batch-processing large image repositories, such as satellite or surveillance data. One of the key strengths of Hadoop is its ability to handle failures gracefully, by reassigning failed tasks to other nodes without disrupting the overall process. It is widely adopted in big data environments where scalability and data reliability are essential.

4.5. Advantages and Disadvantages of Distributed Systems

Distributed systems are widely used in modern computing due to their ability to connect multiple independent machines to work together as a single cohesive system. While they offer significant benefits for performance and scalability, they also come with a set of challenges that must be carefully managed.

4.5.1. Advantages

- **Scalability:** Distributed systems can dynamically scale by adding more nodes, accommodating increasing workloads efficiently.
- **Fault Tolerance:** Redundant nodes ensure that system failures do not lead to complete outages, improving reliability.

- **Cost Efficiency:** By utilizing multiple lower-cost machines instead of expensive supercomputers, distributed systems reduce infrastructure costs.
- **Resource Sharing:** Multiple users and applications can access shared resources, optimizing utilization.

4.5.2. Disadvantages

- **Complexity:** Managing distributed nodes, data synchronization, and task scheduling requires sophisticated algorithms.
- **Consistency Issues:** Ensuring data consistency across multiple nodes is challenging, especially in real-time applications.
- **Network Overhead:** Communication between nodes introduces latency, which may impact performance.
- **Security Concerns:** Data transmitted across networks is vulnerable to cyber threats and requires strong encryption protocols.

Distributed systems have revolutionized computing by enabling scalable, efficient, and resilient architectures for modern applications. By leveraging computing models such as shared memory, distributed memory, and hybrid approaches, distributed systems can handle complex computations across multiple nodes. Techniques such as HPC, cloud computing, GPU acceleration, and multiprocessing provide powerful tools for parallel processing, optimizing performance across various domains. Despite their advantages, distributed systems also pose challenges, including synchronization, fault tolerance, and security risks. As technology advances, ongoing research and innovations will continue to enhance distributed computing, making it a cornerstone of future computing paradigms.

5. Literature review about image processing using distributed system

In recent years, distributed systems have been increasingly applied to image processing tasks to overcome the limitations of sequential methods. Researchers have developed various distributed image processing frameworks and algorithms to leverage the computational power of multiple machines. For example, frameworks like Apache Hadoop and MapReduce have been used to parallelize tasks such as image filtering and segmentation by dividing images into smaller blocks and distributing them across different nodes. Similarly, Apache Spark has been employed for real-time image classification and feature extraction tasks, benefiting from its in-memory processing capabilities. In the field of deep learning, distributed systems have enabled the training of large convolutional neural networks (CNNs) for image classification and object detection. Frameworks like TensorFlow and PyTorch offer distributed training mechanisms that allow models to be trained across multiple GPUs or compute nodes. Furthermore, high-performance computing (HPC) environments using MPI (Message Passing Interface) and OpenMP have also been adopted for image processing, where speed and precision are important.

Real-time image processing remains a significant challenge, primarily due to the large amount of data contained in each image that must be processed rapidly and efficiently.

According to [78], interactive real-time processing and rendering (particularly on immersive, high-resolution displays) require highly sophisticated methods in computer graphics, efficient data handling, and advanced parallelization strategies. These tasks are computationally intensive and are further complicated by the projected growth of datasets in this field, which are expected to reach terabyte (TB) scale in the near future.

To address these demands, researchers have made considerable progress in developing parallel processing algorithms that utilize the computational power of Graphics Processing Units (GPUs) [79,80] and multi-core Central Processing Units (CPUs). These parallel architectures have significantly accelerated image segmentation and other image processing tasks. Notably, the introduction of GPUs has provided a powerful, cost-effective, and adaptable platform for parallel computing, which has been widely adopted in various successful studies across intelligent computing and image analysis domains [81].

This section presents a literature review on parallel image processing, highlighting various parallelization tools applied across different domains. In particular, the medical field has recognized the significant benefits of parallel processing algorithms, as demonstrated by real-world experiments conducted in several hospitals [82]. Traditional Central Processing Units (CPUs), however, struggle to efficiently handle the growing volume of medical image data, especially given the rapid increase in dataset sizes. In response to these limitations, Graphics Processing Units (GPUs) have emerged as a cutting-edge solution, offering substantial computational power to address complex challenges in medical image analysis [83].

In [84], a hybrid serial–parallel CNN-Transformer (SPCT) network was proposed for 3D medical image segmentation, integrating the strengths of Convolutional Neural Networks (CNNs) and Transformer architectures. The model incorporates a Cross-Window Self-Attention Transformer (CWST) module to capture global contextual information, along with a Multi-Scale Local Enhancement (MLE) module for effective feature fusion. Extensive evaluations on prostate, atrium, and pancreas MRI/CT datasets demonstrated that SPCT outperforms six state-of-the-art segmentation methods in terms of Dice Similarity Coefficient (DSC), Intersection over Union (IoU), and boundary accuracy, all while maintaining relatively low computational complexity. These results indicate that SPCT is a promising framework for accurate and efficient 3D medical image segmentation. However, its parallelization strategy primarily focused on enhancing feature learning, rather than accelerating the clustering or segmentation processes through full parallel optimization.

In [85], a deep learning-based framework was proposed for simultaneous MRI reconstruction and segmentation. The approach employs a calibrationless, parallel image-domain deep learning model designed to enhance image quality and improve segmentation robustness in the presence of distortions. By integrating Deep Structured Low-Rank (Deep-SLR) reconstruction with a dedicated segmentation network, the method effectively reduces aliasing and blurring artifacts, resulting in improved segmentation accuracy and computational efficiency. Experimental evaluations on brain MRI datasets show that the proposed framework outperforms existing parallel MRI reconstruction and segmentation methods, particularly under few-shot learning scenarios.

Another notable deep learning advancement is the Bidirectional Efficient Attention Parallel Network (BEAP-Net) [86], developed to enhance 3D medical image segmentation within a semi-supervised learning framework. BEAP-Net integrates Supreme Channel Attention

(SCA) and Parallel Spatial Attention (PSA) modules to effectively capture both spatial and channel-specific features. Experimental results on Left Atrium (LA) and pancreas datasets show that BEAP-Net surpasses eight state-of-the-art methods, achieving superior segmentation accuracy while maintaining computational efficiency. However, despite its impressive performance on public datasets, the model's reliance on semi-supervised learning may constrain its applicability in real-time clinical environments where fully annotated datasets are readily available.

Similarly, the Multi-Parallel Blocks UNet (MPB-UNet) [87] was proposed for automated brain tumor segmentation, enhancing the conventional UNet architecture through the integration of multiple parallel processing blocks inspired by the mechanisms of human visual perception. The model incorporates Atrous Spatial Pyramid Pooling (ASPP) to effectively capture multi-scale contextual information, thereby improving segmentation precision. The architecture was evaluated on the Low-Grade Glioma Segmentation Dataset, where MPB-UNet demonstrated superior performance, achieving an accuracy of 99.86% and significantly outperforming existing state-of-the-art methods.

In [88], parallel Fuzzy C-Means (FCM) clustering was investigated for brain tumor segmentation, leveraging GPU acceleration through CUDA. The approach utilized FLAIR MRI images and implemented parallelization for key computational steps, including cluster initialization, membership matrix calculation, and spatial function evaluation. Although the use of GPU significantly accelerated the segmentation process, the method lacked an optimization mechanism for refining cluster centroids, which limited its capability to escape local optima and potentially reduced segmentation accuracy.

The study presented in [89] conducted a comparative analysis of two parallel implementations (Bias-Corrected FCM (BCFCM) and Spatial FCM (SFCM))with a focus on enhancing robustness and efficiency in MRI image segmentation. Performance was evaluated in terms of both segmentation quality and processing speed. By utilizing GPU-based architectures, the implementations demonstrated substantial reductions in execution time while preserving high segmentation accuracy. The findings underscore the effectiveness of parallel processing in optimizing FCM algorithms for medical image analysis.

Adapting FCM for 3D medical image segmentation presents additional computational complexities. In [90], a hybrid parallel implementation was introduced to enhance segmentation accuracy while notably decreasing execution time. Experimental results on both real and simulated medical datasets showed a speedup of up to 5× compared to traditional sequential methods, highlighting its potential for large-scale medical imaging applications.

A novel knowledge-driven FCM approach, FCM-GENIUS, was proposed in [91] for efficient brain tissue segmentation from MRI scans. This method combines region of interest (ROI) selection, knowledge-based initialization, and optimization techniques to enhance centroid selection and minimize computational complexity. Furthermore, the use of CUDA-enabled GPU parallelization accelerates processing significantly. Experimental evaluations on the IBSR datasets demonstrated that FCM-GENIUS achieves a reduction in segmentation time of up to seven times, while maintaining accuracy on par with existing methods.

In [92], the authors introduced a novel parallel computational approach for image processing, specifically designed for a spectral analysis algorithm within a distributed environment for video surveillance systems. This method utilizes the ZeroMQ library to

distribute video frames from the surveillance stream across multiple computing nodes (multi-core processors), ensuring load balancing. Additionally, OpenMP technology is employed to leverage all available CPU cores for accelerating the spectral analysis of the images. The primary focus of the work involves two key aspects: first, the separation of the video stream into individual frames received from the camera, and second, the spectral analysis of these images on multi-core platforms.

Benchara, Y. [93] introduced a scalable distributed k-means algorithm using cloud microservices for high-performance computing (HPC). Their study highlighted the feasibility of leveraging cloud-based parallel processing for efficiently handling large-scale image data. Similarly, Enfedaque et al. [94] proposed a GPU-based implementation of bitplane coding, which provided high-performance parallel coefficient processing for image compression.

In [95], three key contributions are presented. The first contribution involves enhancing the performance of the Support Vector Machine (SVM) for breast cancer diagnosis by utilizing a modern Grey Wolf Optimizer (GWO). The second contribution introduces three efficient scaling techniques as alternatives to the traditional normalization method. The final contribution implements a parallel technique that employs task distribution to improve the efficiency of GWO. The parallelized version of the model demonstrates promising results, particularly in terms of execution time when run on four CPU cores.

In [96], the primary contribution of the paper is the implementation of a MapReduce programming algorithm to analyze large sets of fingerprint images that are typically too large to process due to limited physical memory. The approach aims to extract features from these images efficiently. Initially, the images are stored in an image data repository for preprocessing, followed by feature extraction for the biometric traits of each user, which are then stored in a database. The proposed algorithm simultaneously preprocesses and extracts key features, such as ridges and bifurcations, from multiple fingerprint images. Feature points are detected using the Crossing Number (CN) method. The algorithm is validated using data from the National Institute of Standards and Technology's (NIST) Special Database 4, which contains fingerprint images from various users. Experimental results demonstrate that the MapReduce approach significantly reduces processing time, achieving nearly a 50% decrease compared to traditional methods.

X. Tan et al. [97] proposed an adaptive Spark-based approach for remote sensing data processing, demonstrating enhanced efficiency through map-reduce-based remote processing. The developed model exhibited improved stability and performance within a cloud environment. A mapping and reducing strategy was applied to image tiles, leading to significant improvements in processing large volumes of remote sensing data. However, the Spark model was constrained to pixel-based classification, limiting its broader applicability.

Despite the effectiveness of using several parallel techniques for image processing, challenges such as communication overhead, fault tolerance, and load balancing remain critical concerns in distributed image processing. As datasets continue to grow in volume and complexity, the role of distributed systems in image processing will become increasingly essential, making it a dynamic and evolving research area with significant practical impact.

To sum up, the analysis of the literature presents the progress made in parallel image processing,facilitated by furthering the approaches to parallel algorithms, optimization principles, andcomputational platforms. The discussed studies can thus be regarded as a

critical starting ground forfuture investigations aimed at further enhancement and application of parallel approaches in the contextof image processing to enhance the prospects of the latter.

6. Conclusion

In this chapter, we explored the fundamental concepts of image processing and the role of parallelization in enhancing computational efficiency. Image processing encompasses a wide range of techniques, including filtering, segmentation, feature extraction and selection, which often require significant computational power. To address these challenges, parallelization methodologies have been developed to distribute processing tasks across multiple computing units, improving speed and scalability. Additionally, parallel machine architectures, such as multi-core processors, GPU-based processing, and distributed computing systems, provide the necessary infrastructure to handle large-scale image data efficiently. By integrating parallel computing techniques with advanced hardware architectures, modern image processing applications can achieve high performance, enabling real-time analysis and large-scale data processing in various fields, including medical imaging, satellite image analysis, and artificial intelligence. We have explored the concept of image processing, including its definition and various techniques used to manipulate and analyze digital images. We also discussed distributed systems, providing an overview of their definition and the methods employed to manage and process data across multiple interconnected systems. Both fields play crucial roles in modern computing, with image processing enabling efficient handling of visual data and distributed systems facilitating the management of large-scale computations and resources.

Chapter 2

Metaheuristic for image processing

1. Introduction

Image segmentation is a fundamental task in image processing and computer vision that involves partitioning an image into meaningful regions, making it easier to analyze or interpret. Effective segmentation is essential in various applications, such as medical imaging, remote sensing, object detection, and industrial inspection. However, due to the complex nature of real-world images which characterized by noise, low contrast, and intensity inhomogeneity, traditional segmentation methods often struggle to deliver optimal results.

Metaheuristic algorithms have emerged as powerful tools for tackling image segmentation problems, especially when classical approaches fall short. Inspired by natural phenomena such as evolution, swarm behavior, or physical processes, metaheuristics provide a flexible and efficient means to search for near-optimal solutions in large and complex search spaces. Unlike deterministic methods, metaheuristics do not guarantee the global optimum but often find sufficiently good solutions within reasonable computational time.

Popular metaheuristic algorithms applied to image segmentation include Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and more recent approaches like Grey Wolf Optimization (GWO) and Whale Optimization Algorithm (WOA). These methods are commonly used to optimize clustering objectives, thresholding techniques, or region merging criteria. Their ability to balance exploration and exploitation makes them particularly suitable for segmentation tasks where the objective function is nonlinear, multi-modal, or otherwise difficult to optimize analytically.

In this chapter, we begin by introducing the concept of metaheuristic algorithms, outlining their general principles and applications. We then explore how these algorithms are utilized specifically for image segmentation, highlighting their strengths in handling complex, high-dimensional problems. Following this, we delve into the parallelization of metaheuristic algorithms, discussing how parallel computing enhances their performance and scalability. We then present a comprehensive review of existing research on parallel metaheuristicsapplied to image segmentation. Finally, we conclude with a discussion that synthesizes the insights gained and suggests potential directions for future work.

2. Metaheuristic algorithms

In general, the complexity of real-world problems has been steadily increasing, rendering traditional mathematical programming techniques increasingly inadequate for solving and optimizing such problems. Most real-life optimization challenges are inherently nonlinear, highly

complex, and multimodal, often involving conflicting objective functions. These characteristics make the task of identifying optimal or even near-optimal solutions particularly difficult. In fact, even for seemingly simple or linear objective functions, achieving an optimal solution may be infeasible or non-existent. Consequently, there is often no guarantee of obtaining an optimal solution in practical scenarios [98][99].

In response to these challenges, metaheuristic optimization algorithms have emerged as a prominent and rapidly evolving area of research. These high-level strategies are designed to guide the search process toward high-quality solutions by selecting, combining, or adapting heuristics in an intelligent manner. Metaheuristics aim to efficiently explore complex search spaces and are capable of producing sufficiently good, improved, and robust solutions for a wide range of real-world optimization problems [100][101].

Metaheuristic algorithms represent a class of powerful optimization techniques specifically designed to tackle complex problems that are intractable for conventional mathematical or deterministic methods. These algorithms draw inspiration from various natural processes and phenomena, such as genetic evolution, swarm intelligence, and thermodynamic principles, in order to efficiently explore vast and complex search spaces in pursuit of globally optimal or near-optimal solutions. Prominent examples include Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Simulated Annealing (SA), and Tabu Search (TS), all of which have demonstrated considerable success across diverse domains including engineering, finance, and computer science [102].

One notable advantage of metaheuristic algorithms lies in their independence from initial solution requirements, making them particularly effective in scenarios where the starting conditions are unknown or poorly defined. Moreover, their robustness in navigating high-dimensional, multimodal, and non-convex search spaces distinguishes them from traditional optimization methods. Despite these strengths, metaheuristics are not without limitations. Due to their inherently stochastic nature, there is no guarantee that the global optimum will be found in every execution, and the quality of solutions can vary. Furthermore, their computational overhead can become significant, especially when dealing with large-scale or real-time applications [103].

In summary, while metaheuristic algorithms offer a flexible and effective approach for solving intricate optimization problems, their application must be guided by a deep understanding of the problem context and computational constraints. The selection of an appropriate metaheuristic should consider both the specific characteristics of the optimization problem and the resources available, as their performance is highly problem-dependent [104]. Although not infallible, metaheuristics remain indispensable tools in modern optimization, balancing exploration and exploitation to yield practical solutions where traditional methods fall short.

The term metaheuristic originates from the combination of "meta," meaning beyond or at a higher level, and "heuristic," which denotes a problem-solving approach based on trial-and-error or experiential strategies. Historically, algorithms incorporating stochastic elements were commonly referred to as heuristic methods. Metaheuristics, in this context, have evolved into overarching strategic frameworks that orchestrate and adapt lower-level heuristics, aiming to surpass the limitations of conventional local optimization methods. These strategies systematically blend elements of local search and randomized exploration to efficiently navigate complex solution spaces. While they are capable of yielding high-quality solutions to difficult optimization

problems within reasonable computational time, there is generally no formal guarantee of attaining the global optimum [105].

Metaheuristic algorithms are particularly effective in addressing large-scale and computationally intractable problems, such as those classified as NP-hard, or in environments characterized by uncertainty, incompleteness, or imprecision. Due to the enormity of the search space, it is typically infeasible to evaluate all potential solutions exhaustively. However, one of the key advantages of metaheuristics is their problem-independent design, requiring minimal assumptions about the underlying optimization model. This makes them highly adaptable across a wide range of domains and problem types. Unlike deterministic or iterative optimization techniques, metaheuristics do not ensure convergence to an optimal solution. Many rely on stochastic processes, meaning that the solutions obtained are influenced by probabilistic variables generated during the search process [106].Despite their probabilistic nature, metaheuristics have demonstrated the ability to find high-quality solutions in combinatorial optimization problems with significantly reduced computational overhead compared to exact algorithms, iterative solvers, or rudimentary heuristics. Their capacity to explore diverse regions of the solution space makes them valuable tools for solving complex optimization tasks [107]. A generic schematic illustrating the typical workflow of metaheuristic algorithms is presented in Figure 13.

Much of the scholarly work on metaheuristics is empirical, centered around computational experiments and performance benchmarking. Nonetheless, a body of theoretical research also exists, addressing aspects such as algorithmic convergence and conditions under which global optimality might be achieved. While the field has seen the emergence of numerous innovative and practically effective metaheuristic techniques, it has also been criticized for inconsistencies in scholarly rigor. Common issues include vague conceptual frameworks, inadequate experimental validation, and insufficient engagement with prior literature [108].

To facilitate better understanding and application, metaheuristics have been broadly categorized in the literature based on their core operational strategies and sources of inspiration, ranging from biological and physical processes to social and cognitive behaviors. These classifications are instrumental in elucidating the foundational principles of each algorithm and guiding practitioners in selecting suitable approaches for specific problem contexts. Figure 12 provides a comprehensive taxonomy of metaheuristic families, highlighting representative algorithms within each category.

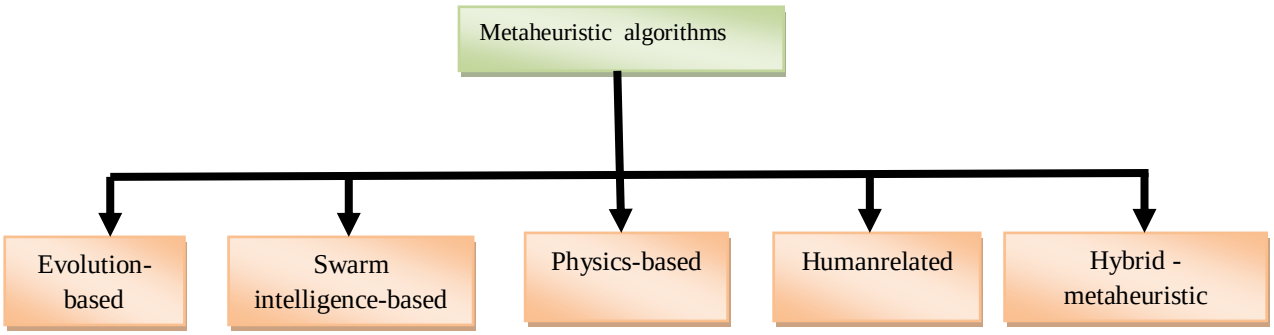


Figure 12.Classification of nature-inspired algorithms.

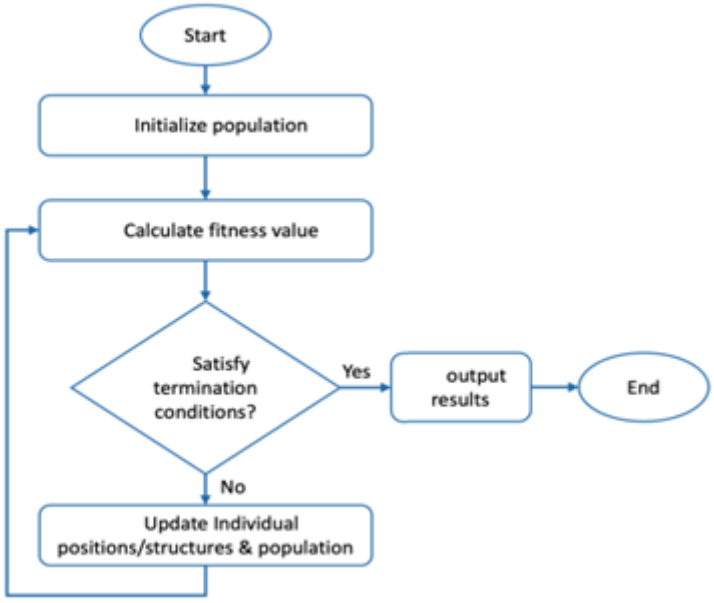


Figure 13. Generic flowchart of metaheuristics algorithms.

2.1. Evolution-based algorithms

Evolutionary Algorithms (EAs) constitute a family of optimization techniques inspired by the principles of natural selection as proposed in Darwin's theory of evolution, which posits that genetic variation arises randomly within a population and that only the fittest individuals survive and reproduce. Drawing upon this biological paradigm, EAs are designed to explore complex search spaces and identify near-optimal solutions through iterative processes. Each cycle of an EA, referred to as a generation, typically involves a sequence of key operations: parent selection, recombination (or crossover), mutation, and survivor selection. While crossover and mutation serve to diversify the population and explore the search space, parent and survivor selection mechanisms focus the search through exploitation of promising regions. Prominent representatives of this algorithmic class include Genetic Algorithms (GA) [13] and Differential Evolution (DE) [109]. These methods begin with a randomly initialized population of candidate solutions, which is iteratively improved by combining and modifying high-quality individuals via evolutionary operations. Among these, the Genetic Algorithm, explicitly modeled on the Darwinian process of evolution, remains the most widely applied and extensively studied. Building upon these foundations, more sophisticated variants such as Genetic Programming, and Differential Evolution have emerged, extending the evolutionary framework to address a broader range of optimization problems. Overall, evolutionary algorithms have demonstrated remarkable adaptability and efficacy across a wide array of application domains. Their capabilities have been successfully leveraged in areas such as image analysis, disease detection, wind speed prediction, and the identification of cancer-related symptoms, underscoring their value as versatile and robust tools for solving complex real-world problems.

2.2. Swarm intelligence-based algorithms

The second prominent category of metaheuristic algorithms the Swarm Intelligence (SI**),** SI is inspired by the collective behavior and decentralized communication observed in social

organisms. These algorithms emulate the way groups of animals, such as birds, fish, ants, and bees, interact and exchange information to guide collective decision-making and problem-solving. The fundamental principle underpinning swarm-based metaheuristics is that the behavioral dynamics of individual agents are influenced by shared knowledge within the group, which in turn directs their movement and convergence patterns during the optimization process. By regulating information exchange within the swarm, these algorithms achieve a balance between exploration of the search space and exploitation of promising regions.

Numerous bio-inspired algorithms fall under this category, each modeled on distinct forms of social or biological behavior. For instance, the BAT algorithm [110], inspired by the echolocation behavior of bats, adapts frequency tuning and signal loudness to explore the solution space effectively. The Cuckoo Search (CS) algorithm [111], modeled on the brood parasitism of cuckoo birds, has been widely applied to real-world optimization problems, with binary variants developed to handle discrete search spaces. Another noteworthy algorithm, the Grasshopper Optimization Algorithm (GOA) [112], simulates the locational dynamics and social interactions of grasshoppers to achieve optimization through a balance of attractive and repulsive forces. The Firefly Algorithm (FA) [113] draws on the bioluminescent communication of fireflies, leveraging perceived brightness and spatial distance to iteratively update solution candidates and converge toward optima—making it especially suitable for feature selection tasks. The Dragonfly Algorithm (DA) [114] replicates the static and dynamic swarming behaviors of dragonflies to solve optimization problems through local and global search strategies. Similarly, the Grey Wolf Optimizer (GWO) [11] models the social hierarchy and group hunting strategies of grey wolves, incorporating leadership dynamics to steer the population through a multi-dimensional search space. Another notable approach, the Flower Pollination Algorithm (FPA) [115], simulates pollination mechanisms in flowering plants to combine local exploitation and global exploration via probabilistic interactions. The Ant Lion Optimizer (ALO) [12] is based on the predatory behavior of ant lions and their interactions with ants, effectively modeling trapping mechanisms to guide optimization processes. Lastly, the Whale Optimization Algorithm (WOA) [14] mimics the bubble-net hunting strategies of humpback whales, incorporating encircling mechanisms and spiral-shaped movements to perform guided search and convergence.

Collectively, swarm intelligence algorithms have demonstrated considerable efficacy in solving a broad range of complex, multi-dimensional, and nonlinear optimization problems. Their adaptive, decentralized nature makes them especially well-suited for dynamic and uncertain environments, where traditional deterministic methods often fall short.

2.3. Physics-based algorithms

The third major category of metaheuristic algorithms encompasses physics-based optimization techniques, which are grounded in the simulation of physical laws and phenomena to guide the search for optimal solutions. These algorithms are inspired by fundamental principles from physics, such as thermodynamics, electromagnetism, and gravitational dynamics, and apply these concepts metaphorically to traverse complex solution spaces.

One of the earliest and most well-known examples is Simulated Annealing (SA) [116], modeled after the annealing process in metallurgy, where materials are heated and then slowly cooled to achieve a stable crystalline structure. SA mimics this process to escape local optima and converge

toward a global optimum, making it especially effective in solving multimodal and rugged optimization landscapes. Another notable algorithm is the Lightning Search Algorithm (LSA) [117], which draws inspiration from the unpredictable and powerful nature of lightning strikes. LSA integrates stochastic search mechanisms with both local and global exploration strategies, using metaphorical discharge paths to balance intensification and diversification in the search space. The Gravitational Search Algorithm (GSA) [118] simulates the laws of gravity and the motion of celestial bodies. In this framework, candidate solutions are treated as objects whose masses influence one another through gravitational attraction. Heavier (i.e., fitter) solutions exert a stronger pull, guiding the population toward more promising regions of the search space. Similarly, Electromagnetic Field Optimization (EFO) [119] emulates the interactions among charged particles within an electromagnetic field. This algorithm governs the movement of particles through attraction and repulsion forces, facilitating a dynamic and adaptive search process that can efficiently converge on optimal solutions.

Beyond these, several other physics-inspired metaheuristics have been developed, such as the Multi-Verse Optimizer, the Sine-Cosine Algorithm, and variants of GSA, each leveraging distinct physical metaphors to tackle high-dimensional, nonlinear, or combinatorial optimization problems. These techniques have shown particular promise in feature selection tasks across diverse datasets, offering robust and flexible tools for handling real-world complexity in data-driven environments.

2.4. Human-related algorithms

Human-inspired metaheuristic algorithms are a class of optimization techniques modeled on human social behaviors, cognitive processes, and learning mechanisms. These algorithms emulate how humans interact, learn, and collaborate to address complex problems, offering novel strategies for solving diverse optimization challenges. This category encompasses several algorithms that draw upon psychological and educational paradigms to enhance the exploration and exploitation of the solution space. One such approach is the Brainstorm Optimization (BSO) algorithm [120], which simulates the human process of idea generation in group discussions. Inspired by creative problem-solving sessions, BSO iteratively generates, evaluates, and refines candidate solutions through collaborative mechanisms, making it particularly effective for tasks such as data classification and high-dimensional optimization. Another notable method is Teaching–Learning-Based Optimization (TLBO) [121], which models the educational dynamics between a teacher and students in a classroom setting. This algorithm leverages the influence of a 'teacher' (the best solution in the population) to guide the learning of 'students' (other solutions) through two main phases: the teaching phase and the learning phase. These phases promote both exploration and exploitation by simulating knowledge dissemination and peer-to-peer learning, thereby enhancing convergence toward optimal solutions. The Gaining–Sharing Knowledge-Based Algorithm (GSK) [122] also follows a human-centric philosophy, replicating the way individuals gain, share, and assimilate knowledge within a community. This algorithm emphasizes cooperative learning and mutual exchange of information among candidate solutions, which facilitates a more informed and diversified search process. By modeling human knowledge transfer, the GSK algorithm improves adaptability and performance across a wide range of optimization problems.

In essence, human-based metaheuristics offer unique and flexible frameworks for addressing real-world optimization tasks by mimicking fundamental aspects of human cognition, learning, and cooperation.

2.5. Hybrid Metaheuristic

Hybrid metaheuristic algorithms have recently garnered significant attention for their efficacy in addressing complex optimization problems [123]. In particular, they have shown substantial promise in the domain of feature selection, where the goal is to extract the most relevant and optimal subset of features from high-dimensional datasets. These algorithms are constructed by strategically integrating the most effective components such as operators, strategies, or mechanisms from distinct metaheuristic frameworks. By combining complementary strengths of multiple algorithms, hybrid approaches are able to overcome common limitations such as premature convergence and entrapment in local optima. This integration enhances both the exploration (global search) and exploitation (local refinement) capabilities, facilitating more efficient traversal of the search space. As a result, hybrid algorithms are better equipped to deliver near-optimal or optimal solutions with improved robustness and adaptability. The synthesis of diverse algorithmic paradigms allows hybrid metaheuristics to achieve a superior balance between search intensification and diversification. This translates into improved convergence speed, enhanced solution quality, and greater computational efficiency. Ultimately, hybrid metaheuristics represent a powerful strategy in modern optimization, leveraging the strengths of multiple techniques to yield more effective and reliable outcomes across a wide array of application domains.

3. Image segmentation based-metaheuristic

Image segmentation is a crucial operation in image processing, where an image is partitioned into distinct regions based on various features, such as intensity, color, texture, or other characteristics. Traditional segmentation techniques, such as thresholding, clustering, and edge detection, often encounter difficulties when dealing with complex images, noise, or non-uniform illumination. To overcome these challenges, metaheuristic optimization algorithms have been increasingly applied to enhance segmentation accuracy and robustness. These algorithms optimize segmentation parameters by effectively balancing exploration and exploitation strategies, improving segmentation results. Recognized as the primary and most fundamental operation in image analysis, image segmentation plays a pivotal role in diverse computer vision applications, such as medical imaging [124], autonomous target recognition [125], geographic imaging [126], and robotic vision [127]. Image segmentation generally involves dividing an image into multiple segments, typically separating the foreground from the background, based on specific features like textures or grayscale values. In one notable contribution, Mandal introduced an enhanced version of image segmentation using Particle Swarm Optimization (PSO), which demonstrated significant improvement in segmentation performance [128]. Additionally, various nature-inspired optimization algorithms have been utilized in image segmentation to solve related optimization challenges and attain optimal solutions [129][130][131].

The subsequent sections offer a comprehensive review of the most widely used image segmentation techniques and their improvements through nature-inspired algorithms. Figure 14 presents statistical analysis of metaheuristic-based image segmentation in medical applications conducted from 2012 to 2022, based on data sourced from Scopus databases [132].

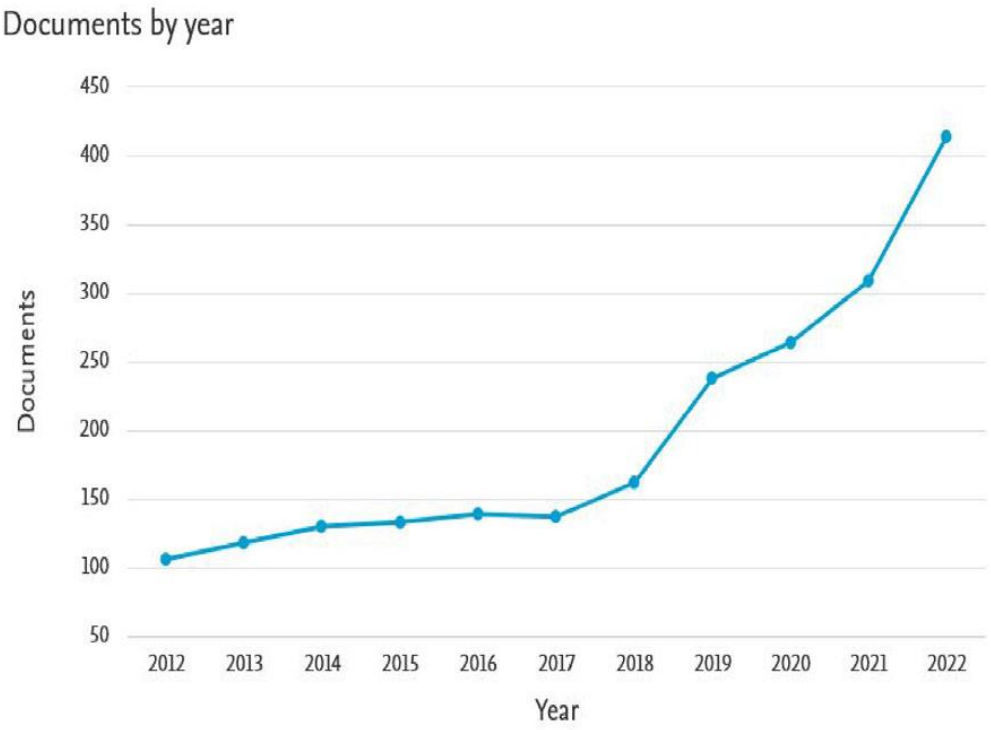


Figure 14. Histogram of publications of image segmentation using Metaheuristics in medical images [132].

A variety of methods have been introduced in the literature that apply metaheuristic algorithms to image segmentation tasks. These methods exploit the global search capabilities of metaheuristic techniques to address the shortcomings of conventional segmentation approaches, particularly in handling complex, noisy, or low-contrast images. Researchers have investigated a broad spectrum of algorithms, including those based on swarm intelligence, evolutionary strategies, and physics-inspired methods, to optimize key segmentation parameters such as threshold values, cluster centroids, or region boundaries. The expanding body of research underscores the effectiveness and flexibility of metaheuristic-based approaches in improving segmentation accuracy and robustness, demonstrating their applicability across diverse image types and practical scenarios.

3.1. Thresholding-based image segmentation using metaheuristic

The task of finding optimal threshold values in an image is often referred to as the thresholding problem. The image histogram is typically used to identify threshold points, with each image possessing its own set of optimal thresholds [133]. Otsu and Kapur methods [134] are widely recognized techniques for determining these thresholds. However, the challenge of multilevel thresholding (MTH) for image segmentation is inherently complex, with detailed discussions on these challenges found in [135, 136]. While Otsu and Kapur are effective for images with a small

number of thresholds, they become computationally expensive and time-consuming when dealing with images that require a large number of thresholds. As a result, nature-inspired optimization algorithms and Swarm Intelligence (SI) methods are employed to tackle such intricate segmentation problems. These metaheuristics mimic natural behaviors, such as those of animals, birds, and humans, to identify optimal solutions. Various metaheuristic algorithms have been applied to MTH problems, including Equilibrium Optimizer (EO) [137], Chimp Optimization Algorithm (ChOA) [138], Artificial Bee Colony (ABC) [139], Particle Swarm Optimization (PSO) [130], Bacterial Foraging Optimization (BFO) [141], and Cuckoo Search (CS) [142]. For medical image segmentation, Genetic Algorithm (GA) combined with Simulated Binary Crossover (SBX) [143] has been utilized to obtain optimal thresholds, showing superior performance in comparison with other algorithms. A modified version of Artificial Bee Colony (ABC), called CCABC, was proposed in [144] to enhance segmentation performance, especially when applied to COVID-19 X-ray image segmentation, outperforming other competitive algorithms. The ABC algorithm [145] was also successfully used to determine the optimal threshold for melanoma detection, with results showing superior performance over alternative methods. In [146], a novel hybrid approach combining the Slime Mold Algorithm (SMA) and Whale Optimization Algorithm (WOA)was introduced to address image segmentation problems for COVID-19 chest X-ray images. The results demonstrated that this hybrid method outperformed all comparison metrics. Dynamic Particle Swarm Optimization (DPSO) combined with Fuzzy C-Means (FCM) [147] was applied to MRI and synthetic images, demonstrating robustness against noise and better performance than other competing algorithms. The integration of Harris Hawks Optimization (HHO) with chaotic initialization and altruism [148] was proposed for thresholding during brain MRI segmentation, showing improved results compared to existing methods. Additionally, the Monarch Butterfly Optimization (MBO)algorithm [149] was employed for medical image segmentation at multiple thresholds, yielding superior accuracy and speed, particularly at thresholds 3 and 4. In [150], an improved Ant Colony Optimization (ACO) algorithm was introduced for COVID-19 X-ray segmentation, leveraging swarm intelligence for more accurate results. Furthermore, a Harris Hawks Optimization (HHO) and Otsu method combination [151] demonstrated significant reductions in computational costs and convergence time while maintaining optimal results. Lastly, an improved Sparrow Search Algorithm [152], incorporating Levy flight and nonlinear inertia weight, was proposed for image threshold segmentation, and its performance on benchmark functions showed it to be superior to other algorithms.

3.2. Clustering-based image segmentation using metaheuristic

Cluster-based image segmentation involves grouping similar pixels together, employing algorithms such as K-means clustering**,** fuzzy clustering, and others [153, 154]. In [155], the Modified Fuzzy K-Means (MFKM) algorithm, combined with Bacteria Foraging Optimization (BFO), was used to identify the tumor region in Magnetic Resonance (MR) brain images by distinguishing between edema and normal tissue regions. The results of this method were compared with traditional Modified Fuzzy K-Means (MFKM)**,** Particle Swarm Optimization-based Fuzzy C-Means (FCM based on PSO), and conventional FCM algorithms, showing superior performance in MR brain image segmentation. A novel approach in [156] involved the use of Red

Fox Optimization (DRFO) with Kernel Fuzzy C-Means to detect skin cancer from dermoscopy images in the ISIC 2020 database, with this method yielding the best results compared to other competitive algorithms. In [157], the Shuffled Shepherd Optimization Algorithm (SSOA)was combined with the Salp Swarm Algorithm (SSA) to create SSSOA, which was applied alongside the Generative Adversarial Network (GAN) model for lung cancer detection in CT images. The SSSOA-based GAN method outperformed other algorithms in terms of accuracy**,** similarity, and the Dice coefficient. The authors in [158] integrated the Social Ski Driver (SSD) algorithm with SSSOA to detect lung cancer in CT images, using the Deep Renyi Entropy Fuzzy Clustering (DREFC) algorithm to segment lung lobes. This proposed method significantly improved accuracy**,** specificity, and sensitivity compared to other algorithms. The use of PSO and Mahalanob is distance in [159] enhanced the Fuzzy C-Means (FCM) algorithm, resulting in the Improved Spatial Fuzzy C-Means (IFCMS) method for image segmentation using simulated brain MRI images from the McConnell Brain Imaging Center database, demonstrating the efficiency of the approach. In [160], a Hybrid Sea Lion Squirrel Search Optimization (HSLnSSO) technique was employed to improve Fused Optimal Centroid K-means with K-Mediods Clustering (FOC-KKC) for dental caries segmentation, showing superior performance compared to other competitive methods.

3.3. Edge-based image segmentation using metaheuristic

Edge detection (ED) plays a pivotal role in image processing by identifying boundaries between regions with distinct gray-level intensities. This process is critical in various applications, such as detecting retinal blood vessels [161]. Classical ED operators including Prewitt, Sobel, Canny, Wallis, Laplacian, and Kirsch, each one of them employ specific convolution masks to emphasize edge features while suppressing irrelevant information, preserving the most salient image characteristics.

Several research efforts have harnessed nature inspired optimization algorithms to enhance edge detection and segmentation performance. For instance, in [162], the authors proposed an ant colony optimization (ACO)-based segmentation technique for processing MRI and iris images. The proposed method demonstrated superior segmentation quality, especially in images with complex local textures, outperforming traditional techniques. In [163], the researchers introduced a geometric deformable model that integrates edge- and region-based information with prior shape knowledge. This model employed genetic algorithms during its training phase to optimize level set parameters, and then applied the learned model during testing. The approach yielded improved accuracy in segmenting anatomical structures across diverse biomedical imaging modalities compared to state-of-the-art methods. A modified watershed segmentation (MWS) algorithm was implemented on a Xilinx Virtex-5 FPGA in [164] to segment brain tumors from MRI images. The hardware-accelerated implementation achieved more accurate results than conventional algorithms. The authors in [165] utilized ensemble deep neural networks combined with Particle Swarm Optimization (PSO) for segmenting the optic disc (OD) in retinal images. Their approach included variations of PSO such as a refined super-ellipse method, random and average leader-based searches, and an accelerated super-ellipse action. By incorporating Mask R-CNN, the technique effectively addressed the biases of individual networks, achieving superior performance in both unimodal and multimodal segmentation tasks. This method also demonstrated high

accuracy in detecting diabetic macular edema, a critical concern for elderly patients at risk of vision loss. In [166], the authors proposed a novel OD segmentation technique based on Markowitz portfolio optimization, validated using four datasets including Messidor, HRF, DRIVE, and a private dataset from Hospital UniversitarioSant Joan de Reus in Spain. The results confirmed the robustness and superiority of the proposed approach over competing techniques. Furthermore, [167] presented a Canny edge detector-based method for curve detection in brain tumor MRI scans, using data from the Neoplastic Disease section of the Whole Brain Atlas (Harvard Medical School). This method showed significant improvements over traditional active contour models such as Chan-Vese (CV), Local Binary Fitting (LBF), and Local Intensity Fitting (LIF), particularly in accurately identifying tumor boundaries.

3.4. Region-based image segmentation using metaheuristic

Region-based image segmentation techniques partition an image into distinct regions by grouping neighboring pixels that share similar characteristics [168]. These methods aim to ensure that each segmented region exhibits uniform properties, such as intensity or texture, while maintaining clear boundaries between dissimilar regions. This approach is particularly valuable in medical imaging, where precise localization of anatomical structures or pathological areas is critical. In [169], the authors proposed a novel framework for liver segmentation in abdominal CT images during the portal phase. The method integrates a multilevel local region-based Sparse Shape Composition (SSC) model with a hierarchical deformable shape optimization algorithm. The framework achieved slightly superior performance when compared to existing liver segmentation techniques. A different study in [170] utilized multi-objective particle swarm optimization (MOPSO) to enhance brain MRI segmentation. This approach addresses the limitations of traditional region-based active contour models and fuzzy entropy clustering. The method was evaluated using datasets from the Internet Brain Segmentation Repository (IBSR), real MR images from the McConnell Brain Imaging Center, and synthetic MR data. The results demonstrated improved robustness and segmentation accuracy across all tested datasets. In [171], the authors combined particle swarm optimization (PSO) with a robust graph-based (RGB) segmentation technique to detect breast tumors in ultrasound images. This hybrid approach outperformed both traditional regional segmentation methods and standard RGB techniques, delivering more precise tumor localization in challenging ultrasound imagery.

3.5. Deep learning and metaheuristic-based image segmentation

Deep learning has revolutionized image segmentation by leveraging convolutional neural networks (CNNs), U-Net, and transformer-based architectures to achieve state-of-the-art accuracy. However, deep learning models often require extensive labeled data, suffer from overfitting, and are computationally expensive. To address these challenges, researchers have integrated metaheuristic algorithms with deep learning for optimized segmentation. Metaheuristics can enhance deep learning models by optimizing hyperparameters, improving feature selection, and refining segmentation masks. For instance, GWO and PSO have been used to fine-tune CNN parameters, ensuring optimal learning rates and filter sizes. Additionally, GA and WOA have been applied to refine segmentation post-processing by optimizing threshold values and boundary refinement in deep learning-generated masks. Another emerging approach is using metaheuristics

for active learning, where algorithms like ACO select the most informative samples for training deep networks, reducing the annotation burden. By combining deep learning's feature extraction capability with the global search ability of metaheuristics, hybrid models achieve more accurate, computationally efficient, and adaptive segmentation, especially in medical and remote sensing applications.

Integrating metaheuristic algorithms with deep learning techniques has shown promise in enhancing image segmentation tasks. These hybrid approaches leverage the strengths of both methodologies to improve segmentation accuracy and efficiency. Below are several notable studies that have explored this integration:

Recent advancements in biomedical image segmentation have seen a growing integration of metaheuristics with deep learning models, yielding promising results across various medical imaging modalities.

In [172], the authors introduced a Hybrid Metaheuristics with Deep Learning-based Fusion Model for Biomedical Image Analysis (HMDL-MFMBIA). This framework encompasses image preprocessing, segmentation using Swin-UNet, and feature extraction through a fusion of deep learning architectures, specifically Xception and ResNet. A Hybrid Salp Swarm Algorithm (HSSA) is used for optimal hyperparameter selection, significantly enhancing the performance of biomedical image classification and analysis.

Similarly, in [173], a local-area contrast-correcting preprocessing technique was proposed. This method uses a brightness-preserving transformation function based on local neighborhood mean and standard deviation. To optimize transformation results, Differential Evolution (DE) and Artificial Bee Colony (ABC) algorithms were used as metaheuristic estimators for decision variables. Evaluation on four publicly available datasets demonstrated that images processed with DE-ABC showed superior segmentation performance compared to raw input data.

In [174], an adaptive multi-objective convolutional neural network, termed AdaResU-Net, was introduced for medical image segmentation. This architecture combines the U-Net framework with residual learning, and employs a Multi-objective Evolutionary Algorithm (MEA) to optimize hyperparameters while balancing segmentation accuracy and model complexity. The model was validated using the Promise12 dataset and cardiac MRI sequences from York University, outperforming traditional U-Net [1] and ResNet [175] models.

Further, [176] presented a fully evolutionary DenseRes model, which leverages dense and residual blocks along with evolutionary algorithms to automatically design optimal network architectures for medical image segmentation. Tested on six public MRI and CT datasets, the model demonstrated high segmentation accuracy while using a minimal number of parameters, surpassing both manually and automatically designed counterparts.

Despite the advantages of using metaheuristics in image segmentation such as robustness, flexibility, and the ability to avoid local optima, these techniques often face challenges including high computational complexity, slow convergence, and scalability issues, particularly with large-scale or real-time image data. A promising solution to these limitations is the parallelization of metaheuristic algorithms, which distributes computational tasks across multiple processors or GPUs, thereby accelerating convergence and enabling real-time or near real-time processing.

4. Parallel metaheuristic for algorithms optimization

On one side, optimization problems are witnessing a significant rise in complexity, accompanied by escalating demands for computational resources. Real-world scenarios frequently involve NP-hard problems that are both CPU- and memory-intensive. Although metaheuristic techniques offer a pragmatic approach to mitigating the computational burden of exhaustive search strategies, they often remain computationally expensive, particularly when addressing high-dimensional search spaces or dealing with intricate objective functions and constraint formulations. This challenge is further amplified by the emergence of increasingly sophisticated metaheuristic frameworks, such as hybrid and multi-objective variants, which themselves are becoming resource-intensive.

Conversely, the rapid evolution of computing technologies has ushered in a new era of parallelism. Advances in processor design, including multicore and specialized processing architectures, alongside the development of high-throughput network infrastructures (e.g., Myrinet, InfiniBand for LANs; optical networks for WANs) and scalable storage systems, have made parallel computing a mainstream solution. The inherent limitations of sequential processing, bounded by physical constraints such as thermodynamic limits and signal propagation delays, have accelerated this transition. Today, multicore processors are standard in even modest computing platforms, such as laptops and personal workstations, representing accessible forms of parallel architecture. Additionally, the continuous decline in cost-to-performance ratios, coupled with the proliferation of high-performance devices and low-latency communication technologies, has significantly bolstered the feasibility and appeal of parallel computing paradigms.

The scientific community is showing growing interest in distributed and massively parallel programming. Parallel metaheuristic optimization is a crucial area in artificial intelligence and computational science, aiming to solve complex optimization problems more efficiently [177]. The objective of using parallel metaheuristics is to enhance the efficiency, scalability, and accuracy of optimization algorithms by leveraging parallel computing. Traditional metaheuristics, can be computationally expensive, especially when applied to complex tasks like image segmentation as a difficult step in image processing. By parallelizing key operations such as fitness evaluation, solution updates, these algorithms can significantly reduce execution time while improving solution quality. Parallel implementations, particularly on multi-core CPUs and GPUs, enable a more extensive exploration of the search space, prevent premature convergence, and facilitate the handling of large-scale data efficiently.

This approach is crucial in applications like MRI segmentation, where real-time processing and high segmentation accuracy are essential for medical diagnostics. Finally, parallel metaheuristics refer to optimization algorithms that exploit parallel computing architectures to enhance performance. These algorithms can be categorized based on their level of parallelism, synchronization strategy, and computational model. The primary goal is to accelerate convergence and improve exploration capabilities in high-dimensional search spaces.

4.1. Parallel metaheuristic strategies

Parallelizing metaheuristic algorithms is essential for improving their efficiency and scalability, particularly in computationally intensive tasks like image segmentation, enhancement, and restoration. Several strategies have been proposed to exploit parallel hardware, ranging from fine-grained fitness evaluations to coarse-grained population division. Each approach offers unique advantages and trade-offs, depending on the image processing application and computational architecture.

4.1.1. Intra-population parallelism (Fine-Grained Parallelism)

The first and most widely used strategy is intra-population parallelism**,** also known as fine-grained parallelism (Figure 15). Each processing element (PE) communicates directly with its immediate neighbors (up, down, left, and right) through dedicated links, as shown by the solid lines. The dotted lines indicate potential or logical communication paths beyond immediate neighbors, emphasizing the local and spatial nature of interactions in such models. In this model, the individuals within a population are evaluated independently and simultaneously. This is especially effective for image processing tasks where fitness functions are computationally expensive, for instance, in multilevel thresholding, each individual may represent a different set of thresholds whose quality is assessed using entropy or edge-based metrics. Since these evaluations are independent, they are ideal for execution on parallel architectures such as GPUs using CUDA, OpenCL, or PyTorch. Each thread can evaluate a separate candidate solution, leading to significant speedups. This strategy is simple to implement and efficient in terms of parallel resource utilization, although synchronization is required during selection and population update phases, which may introduce some overhead.

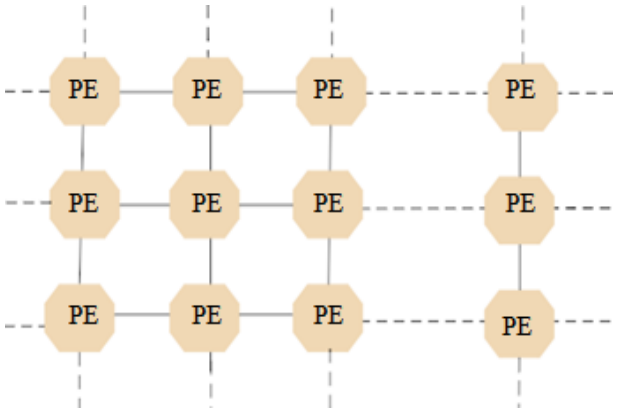


Figure 15. Fine-grained model.

4.1.2. Island strategy (Coarse-Gained Parallelsm)

Another popular method is the island model, which implements coarse-grained parallelism by dividing the population into multiple subpopulations (or islands) (see Figure 16). Each island runs an independent instance of the metaheuristic algorithm and explores the solution space autonomously. At regular intervals, a migration mechanism allows individuals to be exchanged

between islands to maintain diversity and avoid premature convergence. In image processing, the island model is particularly useful for high-dimensional tasks such as MRI segmentation, where each island may focus on different regions or resolution levels of the image. This model is well-suited for distributed computing environments, such as MPI-based clusters or cloud platforms, and offers excellent scalability. However, the effectiveness of this strategy depends on the design of the migration policy, as frequent communication between islands can increase overhead. The papers [178][179] uses the island model to parallelize their propositions.

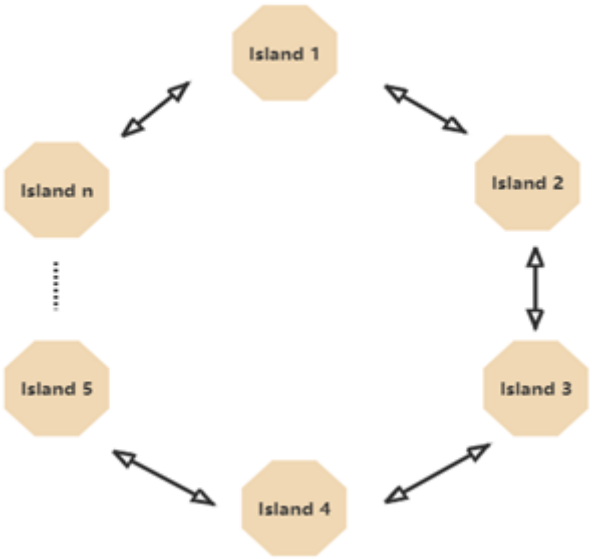


Figure 16. Coarse-gained model.

4.1.3. Master-Slave strategy

The master-slave model represents a task-parallelism approach in which a central master process manages the optimization flow, while several slave processes or threads are responsible for evaluating the fitness of individual solutions (Figure 17). This model is particularly advantageous when the fitness function is complex or time-consuming, such as in clustering-based segmentation or filter parameter optimization. The master generates and dispatches candidate solutions, and the slaves perform the necessary image processing computations in parallel. This model is often implemented using multicore CPUs with OpenMP or multiprocessing libraries in Python. It offers a straightforward architecture and centralized control, making it easier to implement and debug. However, the master can become a bottleneck when dealing with very large populations or high-frequency updates, which limits scalability. Authors in [180][181] used this mode of parallelization in their studies.

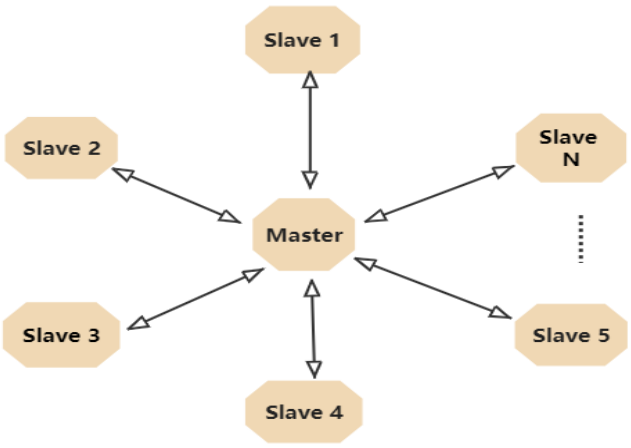


Figure 17. Master-slave model.

4.1.4. Hybrid Parallel Strategy

Finally, hybrid parallel strategies combine elements of both fine-grained and coarse-grained parallelism to exploit the full potential of modern computing architectures (Figure 18). For example, a hybrid model might use an island approach across multiple nodes, where each island internally performs intra-population parallelism using GPUs. This approach is highly adaptable and effective for large-scale image processing applications involving multi-objective optimization or multi-phase segmentation. Hybrid strategies are particularly beneficial when dealing with hierarchical image analysis, where different resolution levels or image patches can be processed in parallel at different granularity levels. However, these strategies are more complex to implement and require careful management of synchronization and communication overhead across different layers of parallelism.

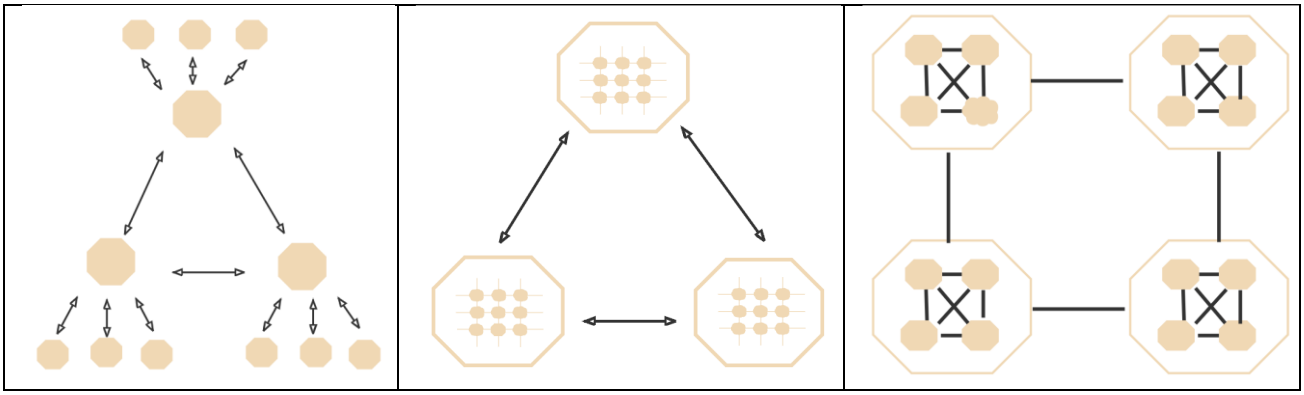


Figure 18. Hybrid model.

4.2. Parallel modelisation of metaheuristics

The reason that drives researchers to use parallel models in metaheuristics is the significant demand for computing power for the problems to be solved. Indeed, evaluating the objective function for each solution is generally the most costly operation in a metaheuristic [182]. We can

distinguish three parallelization models. Figure 19 describes the combination of these three models [183]:

4.2.1. The algorithm-level parallel model

This involves parallelizing algorithms (see Part 1 of Figure 19). This model does not depend on the problem being addressed. If the algorithms are independent of each other, parallelizing them only speeds up their execution. There is no improvement in the quality of the solutions found compared to the sequential model. On the other hand, if the algorithms are cooperative (as indicated by the two-way arrows in Figure 19, Part 1), parallelizing them can not only reduce their execution time but also improve the solutions found compared to the sequential model.

4.2.2. The iteration-level parallel model

This model consists of parallelizing neighbor generation at each iteration, regardless of the problem being addressed (see part 2 of Figure 19). It improves execution time, not the solutions found, compared to the sequential model. Its goal is to evaluate and generate neighbor solutions in parallel.

4.2.3. The parallel solution model

This model focuses on the parallel evaluation of a single solution (see part 3 of Figure 19). It depends on the specific problem being addressed. This is the lowest level, where individual solutions (or particles, individuals, configurations, etc.) within an iteration are evaluated or processed in parallel.

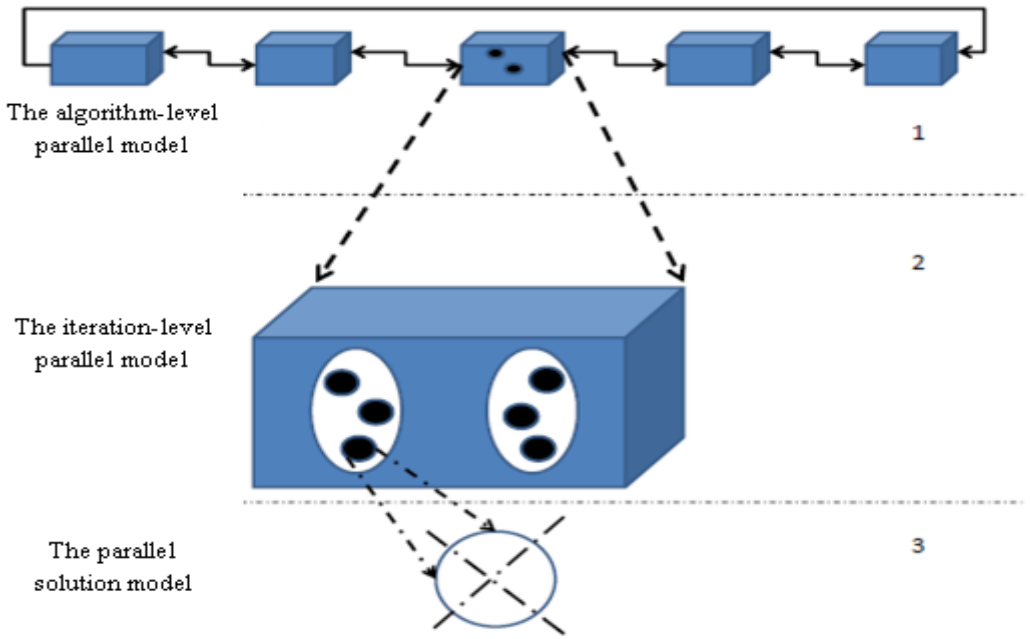


Figure 19. Combining the three parallel model.

4.3. Literature review about image segmentation based parallel metaheuristics

Parallel metaheuristic algorithms have become essential for image segmentation due to their ability to handle high-dimensional data and computational complexity efficiently. researchers have explored parallel implementations using multi-core CPUs, GPUs, and distributed computing frameworks to accelerate convergence and improve segmentation quality. Studies have shown that parallel optimization algorithm significantly reduce execution time while maintaining or enhancing segmentation accuracy. Similarly, hybrid parallel models combining metaheuristics with deep learning or fuzzy clustering have been proposed to improve segmentation robustness. These advancements highlight the importance of parallel metaheuristics in achieving fast, precise, and scalable image segmentation solutions.

Several recent studies have explored the integration of parallel computing and hybrid metaheuristic algorithms to improve the performance and scalability of image segmentation techniques, particularly in medical and high-resolution imaging domains.

In [15], a novel hybrid algorithm combining Harris Hawks Optimization (HHO) and Differential Evolution (DE) is presented for color image multilevel thresholding segmentation. The population is divided into two equal subpopulations, each optimized in parallel using HHO and DE. Otsu's method and Kapur's entropy serve as fitness functions for determining optimal threshold values. When benchmarked against seven state-of-the-art algorithms, the proposed HHO-DE method exhibited superior performance across various quality metrics, including average fitness, standard deviation (STD), Peak Signal to Noise Ration(PSNR**)**, Structure Similarity Index(SSIM), and Root Means Square Error (RMSE), establishing its effectiveness for multilevel thresholding.

In [16], the authors introduced a Parallel Multi-Verse Optimizer (PMVO) that incorporates a communication strategy into the original MVO algorithm. Initial solutions are randomly divided into groups, and information sharing is conducted periodically to mitigate premature convergence and local optima trapping. Tested on the CEC2013 test suite, PMVO outperformed conventional optimizers such as GWO, PSO, MVO, and Parallel PSO. When applied to multilevel image segmentation, PMVO consistently yielded higher-quality results than comparative methods.

A parallel compact Differential Evolution (pcDE) algorithm was proposed in [17] to improve optimization performance and was applied to image segmentation tasks. By dividing the population into multiple subgroups and introducing Optimal Elite (OE) and Mean Elite (ME**)** communication strategies, the algorithm enhanced convergence speed and solution stability. Results on standard benchmarks confirmed its superiority over traditional cDE. However, the parallel execution demanded considerable computational resources, especially for large-scale segmentation tasks.

In [184], a parallel Genetic Algorithm-based Fuzzy C-Means (GA-FCM) clustering algorithm was developed for brain MRI segmentation, using embedded GPUs. The method partitions the genetic population into subgroups and executes clustering in parallel across devices. The integration of Message Passing Interface (MPI) and CUDA ensured efficient load distribution. Experimental results showed up to 12× speedup over CPU-based FCM methods without compromising segmentation accuracy, making this approach highly suitable for large-scale medical imaging.

The work in [185] proposed a Big Data Architecture-based Biomedical Image Classification (BDA-BMIC) system, combining Genetic Algorithms (GA) and Gradient Approximation (GA) for feature selection and classification. The system is built on Apache Spark and Hadoop Distributed File System (HDFS), supporting parallel processing for large biomedical datasets. The use of parallelized SVMs and CNNs accelerated the classification process and achieved superior accuracy and computational efficiency, facilitating real-time image analysis.

Finally, in [186], a hybrid optimization technique is proposed by integrating the Coronavirus Optimization Algorithm (COVIDOA) with Harris Hawks Optimization (HHOA) for 2D and 3D medical image segmentation. The hybrid approach leverages the strengths of both algorithms to enhance convergence behavior. Otsu's method and Kapur's entropy are used as fitness criteria for optimal thresholding. The method is evaluated on IEEE CEC 2019 benchmark functions and various imaging modalities (MRI, CT, X-ray), showing improved results in terms of PSNR, SSIM, and NCC compared to seven other metaheuristic algorithms.

These studies collectively demonstrate that the integration of parallelism and hybrid metaheuristics not only improves segmentation accuracy but also significantly accelerates computational performance. As a result, such techniques are increasingly becoming essential for handling complex, high-resolution, and real-time biomedical image analysis tasks.

5. Conclusion

In this chapter, we presented the basic information about metaheuristics including definition and taxonomy of Metaheuristics algorithm, in addition, we have cited several works in which metaheuristics have been taken into account to improve the segmentation performance. Besides, current limitations of the segmentation methods have been pointed out, including the height computation time, from which we can have a clear view to go further. Even though there is a growing number of works in this field, using Parallel metaheuristics for solving the image segmentation problem has been proven to be successful and accelerate the execution time. By considering the limitations and taking advantages of current works, we propose in the following chapters some methods that are contributions in this area.

Parallelization is a powerful strategy to overcome the limitations of metaheuristic-based image segmentation. By distributing computations across multiple cores or GPUs, parallelization significantly reduces execution time, enhances scalability, and makes real-time processing feasible. Future research can focus on hybrid approaches that integrate deep learning with parallelized metaheuristics for even more robust image segmentation solutions.

Chapter 3

Machine learning and metaheuristic for image classification

1. Introduction

Image classification is a key challenge in computer vision, aiming to automatically assign labels to images based on their visual content. Several machine learning algorithms such as Support Vector Machines (SVM), Decision Trees, and k-Nearest Neighbors (k-NN) have long been applied to this task by learning patterns from hand-crafted features like color, texture, and shape descriptors. More recently, deep learning approaches particularly Convolutional Neural Networks (CNNs), have become the state-of-the-art due to their ability to automatically extract hierarchical features and deliver high classification accuracy. Despite their success, these models often suffer from limitations such as high computational cost, sensitivity to parameter settings, and the need for large labeled datasets. To address these challenges, metaheuristic optimization algorithms have been introduced as a complementary approach. Inspired by natural and evolutionary processes, metaheuristics like Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and Grey Wolf Optimizer (GWO) offer robust, flexible search strategies for solving complex optimization problems. In image classification, they have been effectively used to optimize feature selection, fine-tune model hyperparameters, and even enhance the training of neural networks. The integration of machine learning with metaheuristic optimization leads to hybrid systems that combine learning capabilities with global search efficiency, resulting in improved accuracy, faster convergence, and better generalization especially in scenarios involving high-dimensional data, imbalanced classes, or noisy inputs. This synergy has shown significant potential in domains such as medical diagnostics, satellite imagery analysis, and facial recognition, where reliable and efficient image classification is essential.

In this chapter, we explore the application of machine learning and metaheuristic algorithms in the field of image processing, with a particular focus on classification tasks. Machine learning techniques, including both traditional models and deep learning approaches, have proven effective in analyzing and interpreting visual data for various image processing applications. Metaheuristic algorithms, on the other hand, offer powerful optimization strategies that enhance model performance by selecting the most relevant features and tuning parameters efficiently. The integration of these two paradigms enables more accurate and robust image analysis systems. Following this general overview, we focus specifically on the use of a Modified Grey Wolf Optimizer (MGWO) for feature selection in the classification of

breast cancer. By employing MGWO, the aim is to identify the most informative features from medical images, thereby improving the accuracy and efficiency of the classification process. This hybrid approach demonstrates the potential of combining bio-inspired optimization with machine learning to address complex medical imaging problems.

2. Machine learning

Machine Learning (ML) is a subfield of artificial intelligence (AI) that focuses on developing algorithms and systems that can learn from data and improve their performance over time without being explicitly programmed. Instead of relying on hard-coded instructions, ML systems identify patterns and relationships within data, allowing them to make predictions, decisions, or classifications based on new inputs. ML spans several approaches, including supervised learning (learning from labeled data), unsupervised learning (finding structure in unlabeled data), semi-supervised learning (a mix of labeled and unlabeled), and reinforcement learning (learning through trial and error in an interactive environment). Its power lies in its adaptability. ML systems improve as they are exposed to more data, making them useful for a wide range of tasks such as image recognition, natural language processing, predictive analytics, and autonomous systems. In essence, machine learning enables computers to learn from experience, much like humans do, transforming how we solve complex, data-driven problems.

2.1. History of machine learning

The term Machine Learning was originally introduced by Arthur Samuel in 1952 [187]. Five years later, in 1957, Frank Rosenblatt at the Cornell Aeronautical Laboratory integrated Donald Hebb's theory on neural activity with Samuel's foundational ideas, leading to the development of the perceptron, a pioneering model in neural computation. By 1967, the introduction of the nearest neighbor algorithm marked a key advancement in early pattern recognition. This algorithm was notably applied to route mapping and emerged as one of the initial strategies for addressing the traveling salesman problem, aimed at determining the most efficient path. During the 1960s, research revealed that incorporating multiple layers within perceptron architectures significantly enhanced their computational capabilities. This breakthrough in multilayer neural networks opened new avenues in the field of neural network research [187].

As noted in [188], the evolution of machine learning is closely tied to the broader pursuit of artificial intelligence. From AI's inception as a scholarly field, a subset of researchers focused on enabling machines to learn from data. Their approaches ranged from symbolic reasoning methods to early neural network models, such as the perceptron. Many of these early neural approaches were later recognized as variations of generalized linear models from statistical theory. Additionally, probabilistic reasoning techniques, particularly in contexts like automated medical diagnostics, played a crucial role in early machine learning applications. Machine learning emerged as a distinct discipline and began to gain significant momentum during the 1990s. Its focus shifted from the broader and often elusive objective of achieving

artificial intelligence to addressing well-defined, practical problems. This evolution was marked by a departure from the symbolic reasoning approaches traditionally associated with AI, in favor of data-driven methods rooted in statistics and probability theory [189].

Although machine learning and data mining are distinct areas, they share considerable methodological overlap. Many data mining tasks utilize machine learning techniques, albeit often with different end goals. While data mining is primarily concerned with extracting patterns from large datasets, machine learning emphasizes the development of models that can make accurate predictions or decisions based on data.

A core aspect of machine learning is its strong connection to optimization theory. Most learning tasks are framed as the minimization of a loss function over a training dataset. This function quantifies the error or discrepancy between a model's predictions and the actual outcomes, such as assigning incorrect labels in a classification task. Although optimization techniques are capable of reducing loss within the training set, machine learning distinguishes itself through its emphasis on generalization: the model's ability to maintain accuracy on new, unseen data [190]. Machine learning algorithms are designed to ingest and analyze data to uncover underlying patterns related to individuals, organizational activities, transactions, or events. In the subsequent sections, we explore the different types of real-world data and categorize the major branches of machine learning algorithms.

2.2. Types of data

In most cases, the availability and accessibility of data are fundamental to the development of machine learning models and real-world data-driven systems [191, 192]. Data can exist in multiple formats including structured, semi-structured, and unstructured [193, 194]. Additionally, metadata, which refers to data that provides information about other data, plays a crucial role in data management and interpretation. The following provides a brief overview of these data types:

- **Structured Data**: This type of data adheres to a predefined schema and follows a consistent format, making it highly organized and easily searchable by humans and computational systems alike. Structured data is typically stored in relational databases using tabular formats. Examples include personal details (such as names, addresses, and birthdates), financial records (like credit card numbers and stock prices), and geographic coordinates.
- **Unstructured Data**: Unlike structured data, unstructured data lacks a standardized format, which poses challenges in terms of storage, processing, and analysis. This category includes a wide array of content, often in the form of text and multimedia. Examples encompass sensor readings, email content, blog posts, wiki pages, word documents, PDFs, audio clips, video files, digital images, slide presentations, web content, and various business documents.
- **Semi-structured Data**: Although not stored in traditional relational databases, semi-structured data contains organizational elements such as tags or markers that provide a level of structure. This facilitates easier processing compared to unstructured data.

Examples include HTML and XML files, JSON documents, and data stored in NoSQL databases.

- **Metadata**: Metadata is a special category that refers to information about other data, rather than being raw data itself. While data represent facts or values related to entities, metadata provides context or descriptors that enhance the interpretability of that data. For instance, the metadata of a document may include details such as its author, creation date, file size, and format, thereby enriching its informational value for users.

In the fields of machine learning and data science, researchers frequently utilize a variety of well-established datasets tailored to specific application domains. Examples include cybersecurity datasets such as UNSW-NB15 [195], ISCX'12 [196], CICDDoS2019 [197], and Bot-IoT [198], mobile device datasets like phone call logs [199, 200] and SMS logs [201], IoT-related datasets [202, 203], as well as data from sectors such as agriculture, e-commerce [204], and healthcare including datasets on heart disease [205], diabetes mellitus [206], and COVID-19 [207]. These datasets reflect the diversity of data types mentioned earlier (structured, semi-structured, unstructured, and metadata) which vary depending on the application context.

Effectively analyzing such datasets within their respective domains involves uncovering meaningful patterns and insights that can drive the development of intelligent, real-world systems. To achieve this, different machine learning methodologies are employed, each chosen based on their unique learning paradigms and capabilities. A detailed discussion of these machine learning approaches follows in the subsequent section.

2.3. Types of machine learning techniques

Machine learning algorithms are broadly classified into five major categories: supervised learning**,** unsupervised learning**,** semi-supervised learning**,** reinforcement learning**,** and deep learning [208], as illustrated in Figure 20. Each of these learning paradigms offers distinct methodologies and is suited to different types of real-world problem-solving scenarios. A brief overview of each category and its practical applications is provided below.

2.3.1. Supervised

Supervised learning refers to a fundamental machine learning paradigm where the goal is to learn a mapping function from inputs to outputs using a dataset composed of labeled examples. In this approach, the algorithm is trained on input-output pairs, enabling it to infer a function that can generalize to unseen data. This method is inherently task-driven, meaning it is designed to achieve specific predictive outcomes based on the nature of the input data. The two most common types of supervised learning tasks are classification, which involves assigning input data to discrete categories, and regression, which predicts continuous numeric values. For example, text classification, such as determining the sentiment of a tweet or categorizing a product review, the latter is a typical application of supervised learning. Figure

21 illustrates the essential stages involved in a supervised learning workflow, from training on labeled data to applying the learned model for prediction on new inputs.

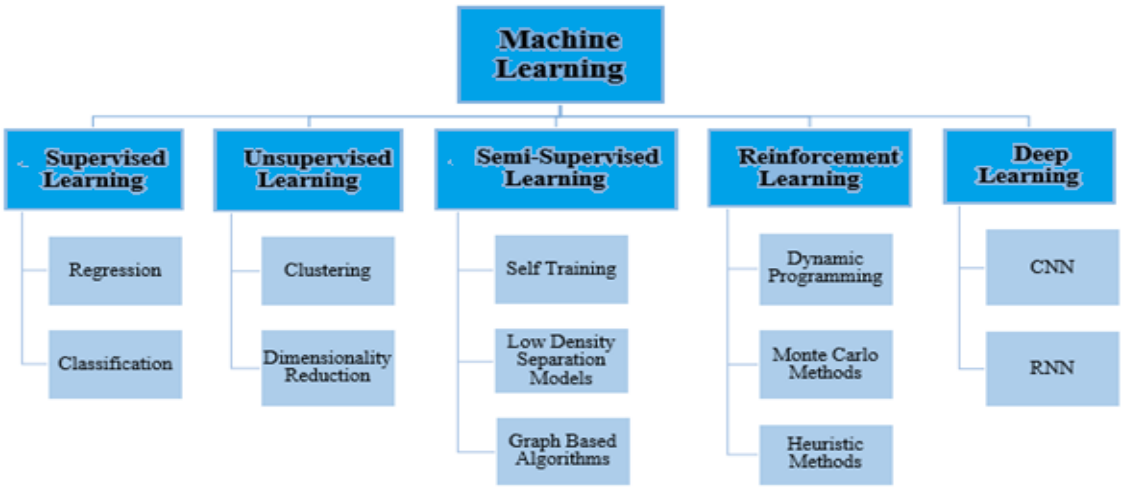


Figure 20. Various types of machine learning techniques [208].

Figure 21. Different stages of the supervised learning [208]

2.3.2. Unsupervised

Unsupervised learning involves the analysis of datasets that lack labeled outputs, relying entirely on the inherent structure of the data, making it a data-driven rather than task-driven approach. Unlike supervised learning, it does not require human-labeled examples, allowing algorithms to autonomously discover patterns, relationships, or structures within the data. This learning paradigm is particularly effective for tasks such as feature extraction, trend discovery, and exploratory data analysis, where the objective is to gain insights or organize information without predefined categories. Common unsupervised tasks include clustering (grouping similar data points), density estimation, dimensionality reduction, feature learning, association rule mining, and anomaly detection. Unsupervised learning is foundational in applications like customer segmentation, topic modeling, fraud detection, and recommendation systems, where understanding hidden structures is crucial for knowledge discovery. Figure 22 represent different steps of the unsupervised learning.

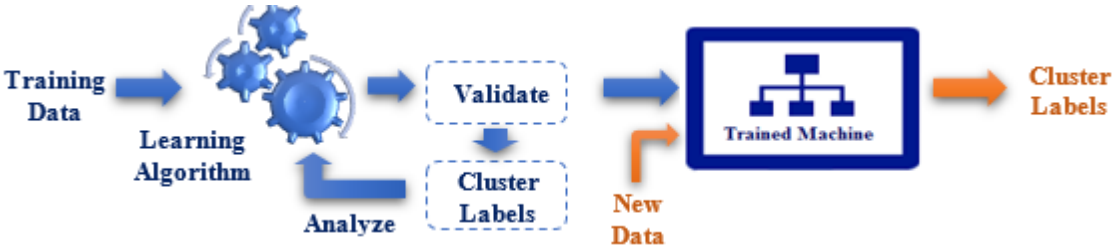


Figure 22. Different stages of the unsupervised learning [208].

2.3.3. Semi-supervised

Semi-supervised learning represents a middle ground between supervised and unsupervised learning approaches. It leverages a combination of both labeled and unlabeled data during training, effectively blending the strengths of each paradigm. This hybrid approach is particularly valuable in real-world scenarios where labeled data is scarce, expensive, or time-consuming to obtain, while large volumes of unlabeled data are readily available [209]. The primary objective of semi-supervised learning is to enhance predictive performance by utilizing the structural information from unlabeled data alongside the limited labeled data, achieving better results than would be possible using labeled data alone. Common application domains for semi-supervised learning include machine translation, fraud detection, automated data labeling, and text classification, among others. This approach is especially effective when unlabeled data can help the model learn the underlying distribution more accurately. Figure 23 illustrates the various stages involved in a semi-supervised machine learning process.

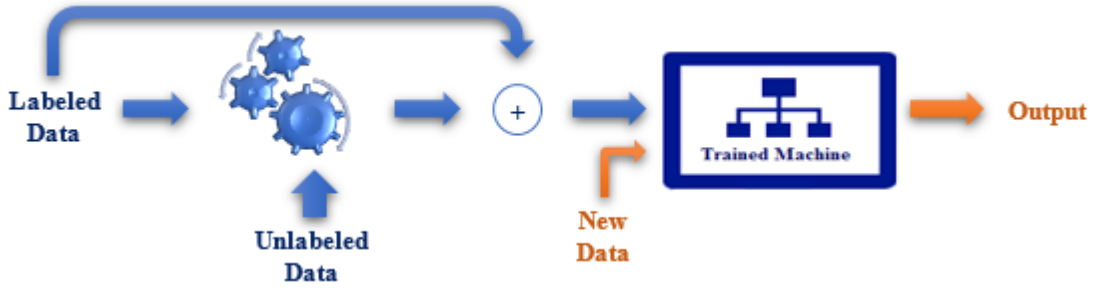


Figure 23. Different stages of the semi-supervised learning [208].

2.3.4. Reinforcement

Reinforcement learning (RL) is a machine learning paradigm in which software agents or machines learn to make optimal decisions through interaction with an environment [210]. Unlike supervised approaches, RL is environment-driven, relying on feedback in the form of rewards or penalties to guide learning. The agent's objective is to learn a strategy(or policy) that maximizes cumulative reward over time while minimizing potential risks [209]. This learning model is particularly effective in scenarios that require sequential decision-making, adaptation, and continuous improvement. RL is instrumental in powering advanced AI systems, especially in areas such as robotics, autonomous vehicles, intelligent manufacturing,

and logistics optimization. However, it is generally not suited for simple or well-defined problems, where traditional learning techniques may suffice. Ultimately, the effectiveness of any machine learning model (be it reinforcement-based or otherwise) depends on aligning the learning strategy with the nature of the data and the intended outcome. As such, choosing the appropriate learning technique is essential to building robust, intelligent systems across diverse application domains.

2.3.5. Deep learning

Since its resurgence in 2006, deep learning has rapidly gained momentum and become a cornerstone in hundreds of research efforts across diverse fields, from information processing to artificial intelligence. As a specialized branch of machine learning, deep learning relies on models that learn from multiple layers of abstraction, enabling them to capture intricate and non-linear relationships within data. At the heart of deep learning lies a hierarchical architecture, where high-level features are progressively built upon lower-level ones, allowing the system to automatically learn complex representations. This layered structure is what gives deep learning its "deep" designation. Notably, many deep learning frameworks are rooted in unsupervised learning representations [211]. Deep learning sits at the intersection of various disciplines, including neural networks, graphical models, optimization, artificial intelligence, pattern recognition, and signal processing. Its rise in popularity can be attributed to several key factors: the exponential growth in computational power (particularly through GPUs), the availability of vast amounts of training data, and its pivotal role in pushing the boundaries of modern machine learning especially in fields like computer vision, natural language processing, and speech recognition.

2.4. Machine learning workflow

Before diving into the detailed workflow of machine learning, it is important to understand the growing importance of intelligent systems in handling complex image processing tasks. With the rapid increase in visual data across fields such as healthcare, security, and remote sensing, there is a strong demand for automated methods capable of interpreting and analyzing images with high accuracy. Machine learning has emerged as a powerful tool in this regard, offering the ability to learn patterns from data and make informed predictions. This section describes steps involved in the machine learning workflow for image processing applications (Figure 24).

- **Data Collection:** The first step in any machine learning workflow is collecting relevant data, which forms the foundation upon which models are built. This data can come from a variety of sources, including internal databases, APIs, web scraping, IoT sensors, user interactions, or public datasets. The quality, quantity, and variety of data gathered have a direct impact on the model's ability to learn meaningful patterns. For instance, biased or incomplete data can lead to inaccurate or unfair models. It's also important to consider the format and structure of the data, whether it's structured (e.g., tables), semi-structured (e.g., JSON/XML), or unstructured (e.g., text, images). Ethical

considerations, such as user privacy and data consent, are especially critical during this phase, particularly when dealing with sensitive or personal information.

- **Data Preprocessing:** Raw data is often messy, inconsistent, and incomplete, which makes preprocessing a critical step. This stage involves cleaning the data by handling missing values, correcting inconsistencies, removing duplicates, and identifying outliers that may distort analysis. Next, the data is transformed into a format suitable for machine learning algorithms. Categorical variables are encoded into numerical values, numerical features are normalized or standardized, and irrelevant or redundant features may be dropped. Feature engineering is also key here, where new features are created from existing ones to enhance the model's predictive power. For high-dimensional data, techniques like Principal Component Analysis (PCA) may be used to reduce dimensionality, improving both computational efficiency and model performance.

- **Data Splitting:** Once preprocessing is complete, the dataset is divided into distinct subsets to enable objective model evaluation. The data is typically split into three parts: training, validation, and test sets. The training set is used to teach the model, allowing it to learn patterns from the data. The validation set helps fine-tune the model's hyperparameters and prevent overfitting by providing feedback on how well the model generalizes to unseen data during training. Finally, the test set acts as a final checkpoint to evaluate the model's real-world performance on completely unseen data. This separation ensures that the model's evaluation is fair, unbiased, and not influenced by data it has already encountered.

- **Model Selection:** With the data prepared, the next step is to choose an appropriate algorithm based on the nature of the problem, whether it's classification, regression, clustering, or reinforcement learning. For example, logistic regression, decision trees, support vector machines (SVM), and neural networks are commonly used for classification tasks, while linear regression is used for predicting continuous outcomes. In unsupervised settings, clustering algorithms like K-Means, DBSCAN, or hierarchical clustering help discover natural groupings in the data. The choice of algorithm depends on several factors, including the size of the dataset, the number of features, interpretability requirements, training time, and the expected accuracy. Often, multiple models are trained and compared before settling on the best-performing one.

- **Model Training:** Training a machine learning model involves feeding it the training data so it can learn to make predictions or decisions without being explicitly programmed. During this phase, the model adjusts its internal parameters to minimize the difference between its predictions and the actual outcomes, a process guided by a loss function. Optimization techniques such as gradient descent are used to iteratively improve the model's performance. Depending on the complexity of the model and the size of the dataset, training can range from seconds to hours or even days. It's essential

to monitor the training process to prevent issues like overfitting, where the model becomes too tailored to the training data and performs poorly on new data.

- **Model Evaluation:** After training, the model's effectiveness is assessed using performance metrics relevant to the task. For classification problems, metrics like accuracy, precision, recall, F1-score, and ROC-AUC are commonly used, while regression tasks may use mean squared error (MSE), mean absolute error (MAE), or R-squared. The evaluation is done on the test set, which contains data the model hasn't seen before, providing a true measure of how it might perform in real-world scenarios. To further validate the model's robustness, techniques like k-fold cross-validation are used, where the dataset is split into k parts, and the model is trained and tested k times on different combinations. This helps ensure the results are not biased by a specific data split.

- **Model Optimization:** Model optimization focuses on refining the model to achieve better performance. This includes tuning hyperparameters, predefined settings that influence the training process but are not learned from the data, such as learning rate, depth of trees, or number of neurons in a neural network. Techniques like grid search, random search, or Bayesian optimization are used to find the best hyperparameter combinations. Regularization methods like L1 (Lasso) and L2 (Ridge) are applied to reduce overfitting by penalizing overly complex models. Ensemble learning methods, such as bagging (Random Forests) or boosting (XGBoost), can be employed to combine multiple weak models into a stronger one. Effective optimization can significantly enhance both accuracy and generalization.

- **Model Deployment :** Once the model meets performance expectations, it is deployed into a real-world environment where it can provide predictions on new, incoming data. Deployment involves integrating the model into an application or service, often via REST APIs, microservices, or cloud platforms. The model must be scalable, secure, and able to respond in real time or batch mode depending on the use case. Monitoring infrastructure is also important at this stage to track system health, latency, and throughput. Additionally, considerations like version control, rollback mechanisms, and containerization (e.g., using Docker) ensure that the model operates reliably and can be updated or replaced as needed.

- **Model Maintenance:** The final step in the ML lifecycle is ongoing model maintenance. As the environment or input data evolves, the model's performance can degrade, a phenomenon known as concept drift. To combat this, models must be continuously monitored for accuracy, fairness, and reliability. If performance drops, retraining the model with recent data may be necessary. Maintenance also includes periodically re-evaluating features, retraining with additional data, updating dependencies, and testing against new edge cases. Tools for A/B testing can be used to compare new models with existing ones in live environments, ensuring only

improvements are deployed. A well-maintained model ensures long-term value and alignment with evolving user needs.

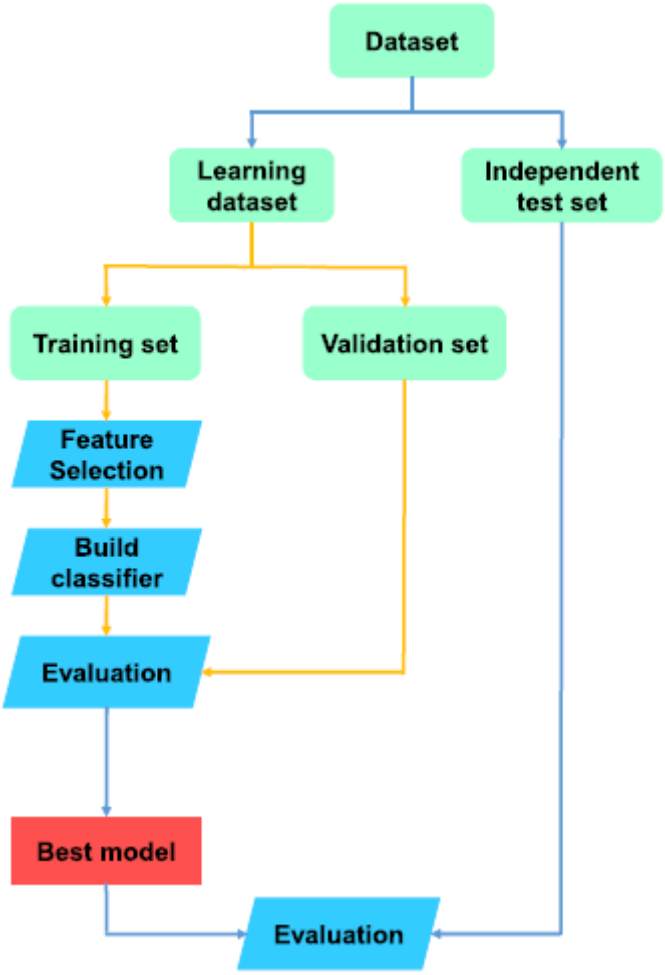


Figure 24. Flowchart of machine learning process.

2.5. Machine learning tasks

Machine learning (ML) encompasses a diverse range of tasks that allow models to learn from data and make intelligent decisions. These tasks can be broadly categorized into classification, regression, clustering, anomaly detection, dimensionality reduction, and reinforcement learning, each with unique applications across different domains.

- **Classification:** Classification is a supervised learning task where the model assigns input data to predefined categories. It is widely used in applications like spam detection (classifying emails as spam or not), medical diagnosis (predicting diseases based on symptoms), and image recognition (identifying objects in images). Algorithms such as Support Vector Machines (SVM), Random Forest (RF), and Neural Networks are commonly employed for classification tasks.

- **Regression:** Regression is another supervised learning task that predicts a continuous numerical value based on input features. It is commonly applied in house price prediction, stock market forecasting, and weather prediction. Techniques such as Linear Regression, Decision Trees, and Deep Learning-based regression models help establish relationships between variables to make accurate predictions.
- **Clustering:** Clustering is an unsupervised learning task that groups data points into clusters based on similarity. It is used in applications like customer segmentation (grouping customers by behavior), anomaly detection (identifying unusual patterns in network security), and gene expression analysis in bioinformatics. Popular clustering algorithms include K-Means, DBSCAN (Density-Based Spatial Clustering), and Hierarchical Clustering.
- **Anomaly Detection:** Anomaly detection identifies outliers or rare events that differ significantly from normal data patterns. It is widely used in fraud detection (identifying suspicious financial transactions), network security (detecting cyber-attacks), and industrial maintenance (predicting equipment failures). Methods such as One-Class SVM, Isolation Forests, and Autoencoders are commonly used for anomaly detection.
- **Dimensionality Reduction :** Dimensionality reduction is crucial for handling high-dimensional datasets by reducing the number of features while preserving essential information. It is commonly applied in imag**e** compression, text processing (reducing feature space in Natural Language Processing), and medical diagnostics (extracting key biomarkers from genomic data). Techniques such as Principal Component Analysis (PCA), t-Distributed Stochastic Neighbor Embedding (t-SNE), and Autoencoders help simplify complex datasets.
- **Reinforcement Learning:** Reinforcement learning (RL) is a task where agents learn optimal strategies by interacting with an environment and receiving rewards for desired actions. RL is extensively used in robotics (autonomous navigation), game playing (AlphaGo, OpenAI's Dota 2 bot), and recommendation systems (personalized content recommendations). Key RL algorithms include Q-Learning, Deep Q Networks (DQN), and Policy Gradient Methods.

Finally, machine learning tasks serve as the foundation for intelligent systems across multiple industries. The choice of ML task depends on the problem type, data characteristics, and desired outcomes. As ML research advances, hybrid approaches combining supervised, unsupervised, and reinforcement learning continue to enhance performance in real-world applications.

2.6. Classification analysis

Classification is a fundamental supervised learning approach in machine learning, often framed as a predictive modeling task wherein the objective is to assign predefined class labels to input instances [193]. Formally, it involves learning a function (F) that maps input features (X) to target outputs (Y), which may represent categories, labels, or classes. This process can be applied to both structured and unstructured datasets to determine the appropriate class of

new observations. A common example is email spam detection, where messages are categorized as either “spam” or “not spam”. In the following sections, we outline several prevalent classification tasks.

- **Binary Classification:** Binary classification involves predictive tasks where each instance is categorized into one of two distinct classes, such as “true/false” or “yes/no” [193]. Typically, one class represents the default or normal state, while the other denotes an abnormal or exceptional condition. For example, in medical diagnostics, “cancer not detected” may represent the normal condition, whereas “cancer detected” signifies an abnormal outcome. Similarly, the task of distinguishing between “spam” and “not spam” in email filtering exemplifies binary classification.
- **Multiclass Classification:** Multiclass classification extends the binary framework to problems involving more than two class labels [193]. Unlike binary classification, it does not rely on a dichotomy of normal versus abnormal outcomes. Instead, instances are assigned to one of several possible categories. A typical example is the classification of network intrusions in the NSL-KDD dataset [212], where attacks are categorized into four distinct classes: Denial of Service (DoS), User to Root (U2R), Remote to Local (R2L), and Probing.
- **Multi-label Classification:** Multi-label classification addresses scenarios where a single instance may simultaneously belong to multiple classes or categories, making it a generalization of multiclass classification. Here, labels are not mutually exclusive and may follow a hierarchical structure, such as in multi-level text classification. For example, a Google News article may be concurrently tagged under “technology,” “city name,” and “latest news.” This paradigm necessitates sophisticated learning algorithms capable of predicting multiple, potentially overlapping labels [213].

Numerous classification algorithms have been developed and extensively studied in the fields of machine learning and data science [214]. In the subsequent section, we provide an overview of the most widely adopted classification techniques across various application domains.

2.6.1. Support Vector Machine (SVM)

A Support Vector Machine (SVM) is a supervised learning algorithm widely utilized for both classification and regression tasks [2][215][216]. Although applicable to regression problems, SVM is particularly effective and predominantly used for classification. The core objective of the SVM algorithm is to determine the optimal hyperplane that best separates data points belonging to different classes within an N-dimensional feature space. This separation is achieved by maximizing the margin between the nearest data points of opposing classes, referred to as support vectors. The dimensionality of the hyperplane corresponds to the number of input features: for example, with two features, the hyperplane is a line with three features, it becomes a two-dimensional plane. As the number of features increases

beyond three, the geometric complexity of the hyperplane escalates, making it more challenging to visualize.

2.6.1.1. Support Vector Machine Terminology

In the context of Support Vector Machines (SVM), several key terminologies define the mechanics and theoretical foundation of the algorithm. At the core is the hyperplane, which is a decision boundary that separates the feature space into distinct classes. In an N-dimensional space, this hyperplane has N−1 dimensions and is mathematically represented by a linear equation. Support vectors are the data points that lie closest to the hyperplane, they are critical because they directly influence the position and orientation of the hyperplane. The margin refers to the distance between the hyperplane and the nearest support vectors from each class. The objective of the SVM algorithm is to maximize this margin, thereby improving the model's generalization ability and reducing overfitting. A hard margin SVM assumes that the data is linearly separable and aims for perfect classification with no misclassification, while a soft margin SVM allows for some misclassification in exchange for better performance on non-linearly separable data. The degree of tolerance to misclassification is controlled by a regularization parameter C, where a smaller value allows more misclassification and a larger value enforces stricter separation. In cases where data cannot be linearly separated, kernel functions are employed to implicitly map the original input space into a higher-dimensional space, where a linear separation is possible. Common kernel types include the linear kernel**,** polynomial kernel, and Radial Basis Function (RBF) kernel**.** The decision function is the model's output that determines which side of the hyperplane a new data point falls on, thereby assigning it to a specific class. Together, these components enable SVMs to be powerful and flexible tools for both linear and non-linear classification tasks.

2.6.1.2. Support vector machine algorithm working

In Support Vector Machines (SVM), an optimal hyperplane is typically defined as the one that maximizes the margin of separation between the two classes. This hyperplane, known as the maximum-margin hyperplane or hard margin, is constructed to achieve the greatest possible distance between itself and the closest data points from each class, these critical points are termed support vectors. The rationale behind maximizing the margin is to enhance the model's ability to generalize to unseen data by minimizing the risk of misclassification. A wider margin implies a more confident decision boundary, which in turn improves the robustness of the classifier, especially in linearly separable datasets.

The optimal hyperplane in a Support Vector Machine (SVM) is chosen such that the distance to the nearest data point from each class is maximized. This distance defines the margin, and the corresponding hyperplane is referred to as the maximum-margin hyperplane or hard margin, provided that the data is linearly separable. Among the candidate hyperplanes illustrated in Figure 25**,** the hyperplane labeled L2 satisfies this criterion by maintaining the greatest margin from the closest data points on either side. Therefore, L2 is selected as the optimal decision boundary.

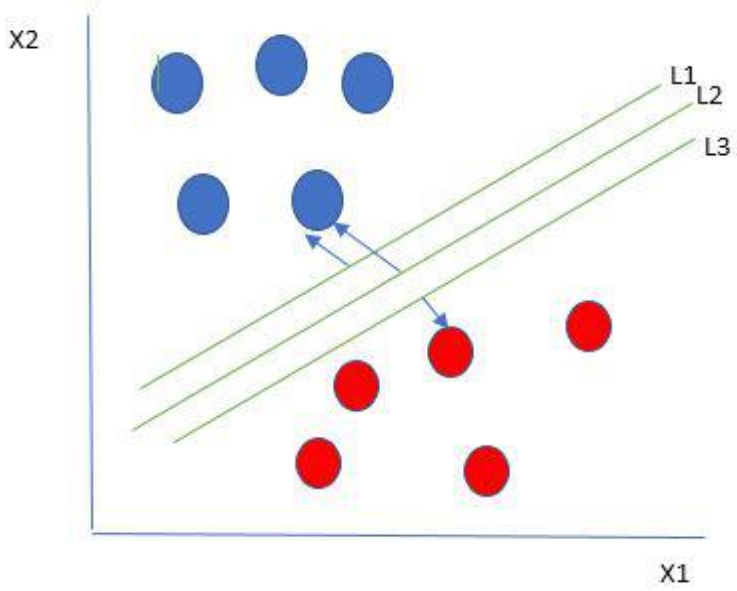


Figure 25. Multiple hyperplanes separate the data from two classes.

2.6.2. Decision Tree algorithm (DT)

A Decision Tree is a widely used supervised learning algorithm in machine learning, designed to predict outcomes based on input features [217]. Structurally, it resembles a tree, where internal nodes represent decision rules or tests on specific attributes, branches denote the outcomes of these tests (i.e., attribute values), and leaf nodes correspond to the final predictions or class labels. The algorithm operates by recursively partitioning the data into subsets based on the most significant feature at each step, thereby creating a model that is both interpretable and effective. Decision trees are versatile and can be applied to both classification and regression tasks, making them suitable for a wide range of applications in machine learning. Their ability to model non-linear relationships and handle both numerical and categorical data contributes to their popularity across various domains.

2.6.2.1. Decision tree terminologies

In the context of decision trees, several key terminologies define the structure and learning process of the model. At the top of the tree is the root node, which represents the initial feature or attribute upon which the first split is made. From this node, the data is recursively divided into subsets based on decision rules, which are derived from the feature values. The points where these splits occur are called internal nodes or decision nodes, and each one tests a specific attribute to determine the path the data should follow. The branches emerging from these nodes correspond to the possible outcomes or values of the tested attribute. The terminal points of the tree, known as leaf nodes or terminal nodes, represent the final output or prediction, which could be a class label in classification tasks or a numerical value in regression tasks. The path from the root node to a leaf node constitutes a decision path, encapsulating a series of rules that lead to a prediction. Another important concept is

information gain (or Gini impurity), which is used as a criterion to select the optimal attribute for splitting; it quantifies how well a given feature separates the data into distinct classes. Overfitting is a common challenge in decision trees, where the model becomes too complex and captures noise in the training data, leading to poor generalization. Techniques such as pruning (removing branches that add little predictive value) are employed to counteract this. Overall, understanding these terminologies is essential for interpreting, constructing, and optimizing decision tree models effectively.

2.6.2.2. Decision trees working

A Decision Tree is a popular machine learning model used for both classification and regression tasks. It constructs a tree-like hierarchical structure, where each internal node represents a decision based on a feature or attribute, each branch corresponds to the outcome of that decision, and each leaf node denotes the final prediction or result. The process of constructing a decision tree typically involves the following steps:

1. **Root Node Selection**:
    - Start with the entire dataset as the root.
    - Select the feature that best splits the dataset based on a specific metric (e.g., Gini Impurity, Information Gain, Mean Squared Error for regression).
2. **Splitting**:
    - Partition the dataset into subsets based on the selected feature and its threshold values.
    - Repeat the process for each subset, creating child nodes.
3. **Stopping Criteria**:
    - Stop splitting when:
        - All data points in a subset belong to the same class.
        - A predefined depth is reached.
        - Splitting no longer improves performance significantly.
4. **Prediction**:
    - For classification, the majority class in a leaf node is the predicted class.
    - For regression, the average of the target values in the leaf node is used.

2.6.3. Random forest algorithm

A Random Forest Algorithm [218][219] is a supervised machine learning algorithm that is extremely popular and is used for Classification and Regression problems in Machine Learning. We know that a forest comprises numerous trees, and the more trees more it will be robust. Similarly, the greater the number of trees in a Random Forest Algorithm [220], the higher its accuracy and problem-solving ability. Random Forest is a classifier that contains several decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset. It is based on the concept of ensemble learning which is a process of combining multiple classifiers to solve a complex problem and improve the performance of the model.

2.6.3.1. Random Forest Algorithm working

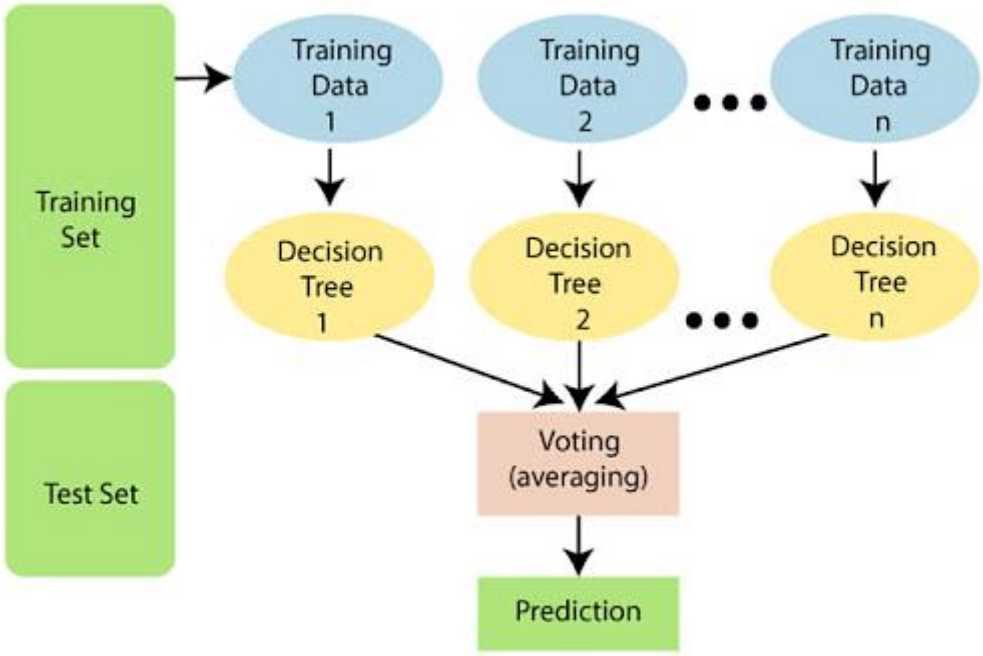


Figure 26. Random forest algorithm.

The following steps explain the working Random Forest Algorithm (Figure 26):

Step 1: Select random samples from a given data or training set.

Step 2: This algorithm will construct a decision tree for every training data.

Step 3: Voting will take place by averaging the decision tree.

Step 4: Finally, select the most voted prediction result as the final prediction result.

2.6.3.2. Difference between decision tree and random forest

Decision Trees	Random Forest
They usually suffer from the problem of overfitting if it's allowed to grow without any control.	Since they are created from subsets of data and the final output is based on average or majority ranking, the problem of overfitting doesn't happen here.
A single decision tree is comparatively faster in computation.	It is slower.
They use a particular set of rules when a data set with features are taken as input.	Random Forest randomly selects observations, builds a decision tree and then the result is obtained based on majority voting. No formulas are required here.

Table 2. Difference between decision tree and random forest algorithm.

2.6.3.3. Important hyperparameters

Hyperparameters are used in random forests to either enhance the performance and predictive power of models or to make the model faster. The following hyperparameters are used to enhance the predictive power:

- n_estimators: Number of trees built by the algorithm before averaging the products.
- max_features: Maximum number of features random forest uses before considering splitting a node.
- mini_sample_leaf: Determines the minimum number of leaves required to split an internal node.

The following hyperparameters are used to increase the speed of the model:

- n_jobs: Conveys to the engine how many processors are allowed to use. If the value is 1, it can use only one processor, but if the value is -1,, there is no limit.
- random_state: Controls randomness of the sample. The model will always produce the same results if it has a definite value of random state and if it has been given the same hyperparameters and the same training data.
- oob_score: OOB (Out Of the Bag) is a random forest cross-validation method. In this, one-third of the sample is not used to train the data but to evaluate its performance.

2.6.4. Naive bayes algorithm

The Naive Bayes classifier is a popular supervised machine learning algorithm used for classification tasks [221] [222]. It is a classification technique based on Bayes' Theorem with an independence assumption among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. It belongs to the family of generative learning algorithms, which means that it models the distribution of inputs for a given class or category. This approach is based on the assumption that the features of the input data are conditionally independent given the class, allowing the algorithm to make predictions quickly and accurately.

In statistics, naive Bayes are simple probabilistic classifiers that apply Bayes' theorem. This theorem is based on the probability of a hypothesis, given the data and some prior knowledge. The naive Bayes classifier assumes that all features in the input data are independent of each other, which is often not true in real-world scenarios. However, despite this simplifying assumption, the naive Bayes classifier is widely used because of its efficiency and good performance in many real-world applications.

Moreover, it is worth noting that naive Bayes classifiers are among the simplest Bayesian network models, yet they can achieve high accuracy levels when coupled with kernel density estimation. This technique involves using a kernel function to estimate the probability density function of the input data, allowing the classifier to improve its performance in complex scenarios where the data distribution is not well-defined. As a result, the naive Bayes classifier is a powerful tool in machine learning

3. Machine learning in image processing

Machine learning (ML) has revolutionized image processing by enabling computers to automatically analyze, interpret, and manipulate images with high accuracy and efficiency [223]. Traditional image processing techniques rely on manually designed filters and algorithms for tasks such as edge detection, noise reduction, and segmentation. In contrast, ML models learn patterns and features from large datasets, allowing them to perform complex image-related tasks with minimal human intervention.

One of the most powerful ML techniques in image processing is deep learning, particularly Convolutional Neural Networks (CNNs). CNNs are designed to recognize spatial hierarchies of features, making them highly effective for tasks like image classification, object detection, and segmentation. For example, CNNs are widely used in facial recognition, medical imaging (e.g., MRI and X-ray analysis), and autonomous driving, where they help identify objects in real-time. Other machine learning algorithms, such as Support Vector Machines (SVMs) and K-Means clustering, are used in applications like handwritten character recognition and image segmentation. Feature extraction techniques, such as Histogram of Oriented Gradients (HOG) and Scale-Invariant Feature Transform (SIFT), are often combined with machine learning models to enhance their ability to recognize objects and patterns in images. Another major application of ML in image processing is image segmentation, where an image is divided into meaningful regions. This is crucial in medical imaging, where tumor detection and tissue segmentation are required for diagnosis.

3.1. Machine learning for image segmentation

Machine learning has significantly advanced image segmentation, enabling precise and automated partitioning of images into meaningful regions, which is essential in fields such as medical imaging, autonomous driving, and satellite image analysis. Traditional machine learning techniques, such as K-means clustering, Support Vector Machines (SVMs), and Random Forests (RFs), rely on manually extracted features like color, texture, and edge information to classify pixels into different regions. For instance, K-means clustering has been widely used in remote sensing, where it groups satellite images into land cover types such as forests, water bodies, and urban areas based on spectral properties. Similarly, SVM-based segmentation has been applied in medical imaging, where it helps identify tumors in MRI and CT scans by classifying pixels based on texture and intensity features. Random Forest classifiers, which utilize an ensemble of decision trees, have also been used for histopathological image segmentation, distinguishing between different tissue structures for cancer diagnosis. However, these traditional approaches often struggle with high-dimensional, complex images because they depend on handcrafted feature extraction, which may not generalize well across diverse datasets. To overcome these limitations, deep learning has revolutionized image segmentation by enabling end-to-end learning without manual feature selection. CNNs and their advanced variants, such as Fully Convolutional Networks (FCNs) [22], are now widely used for pixel-wise classification, significantly improving segmentation accuracy. One of the most impactful architectures, U-Net [19], has shown exceptional

performance in medical image segmentation, such as detecting lung infections in COVID-19 CT scans or identifying brain tumors in MRI scans. Its encoder-decoder structure enables it to capture both low-level and high-level spatial information, making it highly effective for segmenting small and complex structures. More advanced models, such as Mask R-CNN, perform instance segmentation, where they not only classify pixels but also separate different objects of the same category, making them valuable in autonomous driving for detecting individual pedestrians and vehicles. DeepLabV3, another widely used segmentation model, employs atrous convolutions to capture multi-scale contextual information, which enhances its ability to segment complex scenes like urban landscapes and road environments. More recently, transformer-based models such as SETR (Segmenter Transformer) have been introduced, leveraging self-attention mechanisms to capture global dependencies in images, improving segmentation performance in applications like satellite imagery analysis and industrial defect detection. The integration of traditional machine learning approaches with deep learning-based techniques continues to improve segmentation precision, making it an essential tool in various domains, including biomedical imaging, agricultural monitoring, security surveillance, and robotics.

3.2. Machine learning for image classification

Machine learning techniques such as SVMs and RFs have been widely used for image classification due to their strong generalization capabilities and robustness [220]. SVM, a powerful supervised learning algorithm, is particularly effective for binary and multi-class classification tasks. It works by finding the optimal hyperplane that maximizes the margin between different classes in a high-dimensional feature space. For instance, SVMs have been successfully applied in medical imaging [8], such as classifying malignant and benign tumors based on texture and intensity features extracted from MRI scans. On the other hand, RF, an ensemble learning method, constructs multiple decision trees and aggregates their predictions to enhance classification accuracy and reduce overfitting. RF is particularly useful when handling high-dimensional datasets with irrelevant features. A common application is in remote sensing, where RF is used to classify land cover types from satellite imagery by analyzing spectral and spatial features [224]. While these traditional machine learning models require manual feature extraction, they remain highly effective in cases where deep learning may be computationally expensive or when labeled training data is limited. Their interpretability and ability to handle small datasets make them valuable tools for various classification tasks, including medical diagnostics, object recognition, and agricultural monitoring. In contrast, deep learning models like Convolutional Neural Networks (CNNs) have revolutionized image classification by automatically learning hierarchical features from raw images[225]. CNN architectures such as AlexNet, VGGNet, and ResNet have outperformed traditional models in benchmark datasets like ImageNet by recognizing intricate patterns and spatial hierarchies. For instance, CNN-based models have achieved state-of-the-art accuracy in medical diagnostics, such as detecting pneumonia from chest X-rays or classifying diabetic retinopathy from retinal images. More recent advancements, such as Vision Transformers (ViTs) and self-supervised learning, further enhance classification performance by capturing long-range dependencies and reducing reliance on large labeled

datasets. The shift from traditional machine learning models to deep learning approaches has significantly increased the accuracy, scalability, and automation of image classification across various fields, including healthcare, security, and autonomous systems.

3.3. Machine learning for feature selection

Feature selection in image processing using machine learning is a crucial step that enhances model performance by eliminating irrelevant or redundant features while preserving the most informative ones [5]-[10][38]. Unlike feature extraction, which transforms raw data into new feature representations, feature selection focuses on choosing the most significant features from the original dataset, reducing dimensionality and computational costs. Traditional filter-based methods, such as mutual information, chi-square tests, and correlation analysis, rank features based on their statistical importance before model training. These methods are commonly used in texture analysis and remote sensing, where spectral bands or pixel intensities are selected based on their correlation with classification labels. Wrapper methods, like Recursive Feature Elimination (RFE), iteratively remove less important features while training a classifier, ensuring that only the most relevant attributes are retained. A well-known example is gene expression analysis in medical imaging, where RFE is applied to select the most important biomarkers from high-dimensional MRI or CT scan data.

Embedded methods, such as LASSO (Least Absolute Shrinkage and Selection Operator) regression and tree-based algorithms like Random Forest (RF) and Gradient Boosting Machines (GBMs), perform feature selection during training, making them computationally efficient. In medical image segmentation, RF-based feature selection is used to identify the most relevant pixel intensity patterns for tumor detection. Deep learning models integrate feature selection through attention mechanisms, which allow neural networks to focus on the most informative regions of an image. For example, in retinal disease detection, attention-based CNNs highlight critical areas in fundus images, improving diagnostic accuracy. Additionally, auto-encoders, a type of neural network, perform unsupervised feature selection by learning compact, high-level representations of images, reducing noise and redundancy. In autonomous driving, feature selection techniques help identify key visual cues, such as lane markings and obstacles, while filtering out irrelevant background details. The integration of traditional and deep learning-based feature selection methods has led to advancements in biomedical imaging, satellite image classification, and industrial quality control, ensuring more accurate and efficient decision-making in complex image processing tasks.

4. Machine learning and metaheuristic for image processing

The integration of metaheuristic algorithms with machine learning has significantly advanced image processing by enhancing segmentation, classification, feature selection, and feature extraction. Metaheuristic algorithms, inspired by natural and biological processes, provide efficient global optimization techniques that improve the performance of machine learning models in handling complex, high-dimensional image data. These methods help

optimize hyperparameters, refine cluster selection, and improve accuracy in various computer vision tasks.

4.1. Machine learning and metaheuristics for image segmentation

Image segmentation is essential in medical imaging, remote sensing, and object detection, where precise boundary detection is required. Traditional clustering-based segmentation methods, such as FCM, can be improved by integrating metaheuristic optimization techniques. GWO, WOA, and FA have been widely applied to optimize segmentation by selecting the best cluster centroids. For instance, GWO-FCM hybrid models have been employed for MRI brain tumor segmentation, ensuring more precise region identification compared to traditional FCM. In deep learning, metaheuristic techniques like PSO and DE have been used to optimize Fully Convolutional Networks (FCNs) and U-Net architectures, leading to enhanced segmentation performance in medical image processing and autonomous driving applications(See chapter 2, section 4.3).

4.2. Machine learning and metaheuristics for image classification

Image classification is a fundamental task in image processing, where machine learning models categorize images into predefined classes. Traditional machine learning techniques, such as SVMs and RFs, rely on well-selected features, while deep learning models like CNNs automatically extract hierarchical features from images [225]. Metaheuristic algorithms play a crucial role in optimizing these models. For example, HHO and GA have been used to fine-tune hyperparameters in CNN architectures, improving classification accuracy in medical imaging and satellite image recognition. Additionally, ACO has been successfully applied to optimize the selection of relevant training samples, reducing computational complexity while maintaining high accuracy.

4.3. Machine learning and metaheuristics for feature selection

Feature selection is critical in reducing data dimensionality while retaining the most informative features for classification and segmentation tasks. Traditional feature selection methods, such as filter-based and wrapper-based approaches, often struggle with high-dimensional data. Metaheuristic algorithms, including GWO, PSO, and Artificial Bee Colony (ABC), have been extensively applied to improve feature selection in medical imaging and several field image classification [5][6][7][8]. For example, GWO-based feature selection has been utilized in breast cancer diagnosis, selecting the most relevant texture and intensity features from mammograms. Similarly, PSO-based feature selection has been applied to histopathological image classification, improving diagnostic accuracy by reducing redundant information.

4.4. Machine learning and metaheuristics for feature extraction

Feature extraction transforms raw image data into a set of meaningful representations, improving the performance of machine learning models. While deep learning architectures like CNNs can automatically extract features, metaheuristic algorithms can optimize this process by selecting the most relevant deep features. For instance, Hybrid CNN-GA models have been applied to remote sensing image analysis, where GA refines extracted CNN features to improve land cover classification. Additionally, ACO-based feature extraction has been employed in industrial defect detection, ensuring that only the most relevant image features are used for defect identification, reducing false positives and increasing reliability.

5. Conclusion

A wide range of applications in recent literature demonstrate the effectiveness of combining machine learning techniques with metaheuristic algorithms for optimization tasks. This hybrid approach is particularly valuable in scenarios where traditional machine learning methods struggle with high-dimensional data, local optima, or suboptimal parameter configurations. Metaheuristics Optimization algorithms have been extensively used to enhance various aspects of machine learning models, most notably for feature selection, hyperparameter tuning, and model structure optimization. In the field of medical diagnosis, this synergy has proven especially beneficial, enabling the development of more accurate and reliable predictive models. Among these applications, breast cancer classification has received significant attention due to its critical importance in early detection and treatment planning. In the next chapter, we focus on the integration of metaheuristic optimization with machine learning techniques to improve the classification performance in breast cancer diagnosis, highlighting relevant methods, experimental results, and their impact on clinical outcomes.

Chapter 4

Grey wolf optimizer for breast cancer classification

1. Introduction

The Grey Wolf Optimizer (GWO), inspired by the social hierarchy and hunting behavior of grey wolves, has proven to be an effective metaheuristic algorithm for solving complex optimization problems, In the context of breast cancer classification, GWO is particularly useful for feature selection, a critical step in reducing dimensionality and improving classifier performance. Medical datasets, such as those containing breast cancer features, often include redundant or irrelevant attributes that can negatively impact the accuracy and efficiency of classification models. By simulating the intelligent hunting strategy of grey wolves, GWO can explore the feature space effectively and identify the most relevant subset of features that contribute to accurate diagnosis. This not only reduces computational cost but also enhances the interpretability of the model. When integrated with machine learning classifiers such as Support Vector Machines (SVM) or neural networks, GWO-based feature selection has shown promising results in increasing classification accuracy and ensuring reliable detection of breast cancer at early stages.

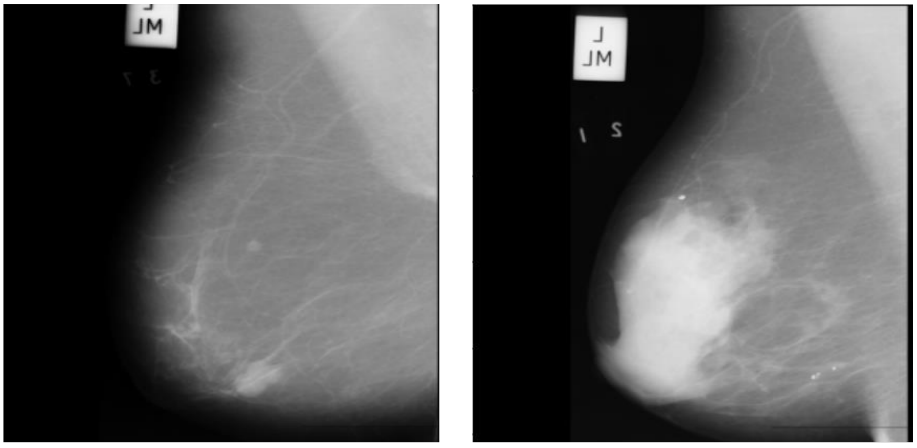
In order to develop an effective approach for precise breast cancer classification, we proposed two methods including a Modified Grey Wolf Optimizer combined with random forest (MGWO-RF) and Correlation technique combined with the Modified grey Wolf Optimizer (CMGWO) for feature selection step then the classification using SVm, RF and NB classifiers. We used the publicly available Wisconsin Breast Cancer Dataset (WBCD), and its features were computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe the characteristics of the cell nuclei present in the image.

2. Breast cancer disease

Breast cancer remains one of the leading causes of mortality among women worldwide [226][227]. Despite being largely preventable in its early stages, a significant number of cases are still diagnosed at advanced stages, reducing the effectiveness of treatment and survival outcomes. The development and implementation of accurate and efficient diagnostic techniques are therefore crucial for enabling personalized care and minimizing the risk of cancer recurrence. In clinical practice, healthcare professionals rely on diverse data sources, including electronic medical records, laboratory test results, and disease-specific studies to support accurate diagnosis and prognosis of breast cancer. Moreover, the integration of artificial intelligence (AI) technologies into the medical domain is increasingly gaining traction, offering promising potential to automate diagnostic processes and enhance the accuracy and efficiency of breast cancer detection.

Breast cancer develops in the cells of the breast tissue, typically originating in the fatty or fibrous connective tissues. It is a malignant tumor that can grow rapidly and progressively worsen, potentially leading to death if not detected and treated in time. While breast cancer predominantly affects women, it can also occur in men, though such cases are rare. Several risk factors contribute to the development of breast cancer, including age, genetic predisposition, and family history. Clinically, breast tumors are generally categorized into two primary types based on their origin and behavior [228]. Figure 27 depicts two sample images from the mammography image analysis society (MIAS) [229] dataset for cancer and normal cases.

- **Benign Tumors**: These are non-cancerous growths composed of cells that do not pose a serious threat to health. Benign tumors do not invade nearby tissues or spread to other parts of the body (a process known as metastasis). They are generally not harmful unless they exert pressure on surrounding tissues, nerves, or blood vessels, potentially causing discomfort or damage.
- **Malignant Tumors**: These tumors consist of cancerous cells capable of invading nearby tissues and spreading to other areas of the body through the bloodstream or lymphatic system, a process referred to as metastasis. Malignant tumors are more aggressive and potentially life-threatening if not treated promptly.



(a) (b)

Figure 27. Two sample images from the MIAS dataset for (a) cancerous, and (b) normal case [229].

Breast cancer can be classified into several subtypes based on the origin, cellular characteristics, and molecular markers:

- **Ductal Carcinoma In Situ (DCIS):** DCIS is a non-invasive or pre-invasive breast cancer where abnormal cells are confined within the milk ducts and have not invaded surrounding breast tissue. It is considered the earliest form of breast cancer (Stage 0) and is typically detected through mammography. While DCIS itself is not life-

threatening, it can increase the risk of developing invasive breast cancer if left untreated.

- **Invasive Ductal Carcinoma (IDC):** IDC is the most common type of breast cancer. It originates in the milk ducts and invades the surrounding breast tissue, with the potential to metastasize to other parts of the body. IDC may present as a lump in the breast and is typically diagnosed through imaging and biopsy.
- **Invasive Lobular Carcinoma (ILC):** Originates in the lobules and has a tendency to be more difficult to detect on mammograms [230]. ILC begins in the lobules, the glands responsible for milk production, and invades surrounding tissues. ILC can be more challenging to detect through imaging due to its growth pattern, which often results in a thickening or swelling rather than a distinct lump.
- **Inflammatory Breast Cancer (IBC):** A rare but aggressive form that blocks lymph vessels in the skin of the breast, causing swelling and redness [231]. IBC is a rare and aggressive form of breast cancer that blocks lymph vessels in the skin of the breast, leading to redness, swelling, and warmth. It often lacks a distinct lump and can be mistaken for an infection. IBC is typically diagnosed at a more advanced stage and requires a combination of chemotherapy, surgery, and radiation therapy.

However, medical imaging plays a critical role in breast cancer screening, diagnosis, and treatment planning [223]:

- **Mammography:** The gold standard for breast cancer screening; it uses low-dose X-rays to identify abnormal masses or calcifications (see figure 28) [232].

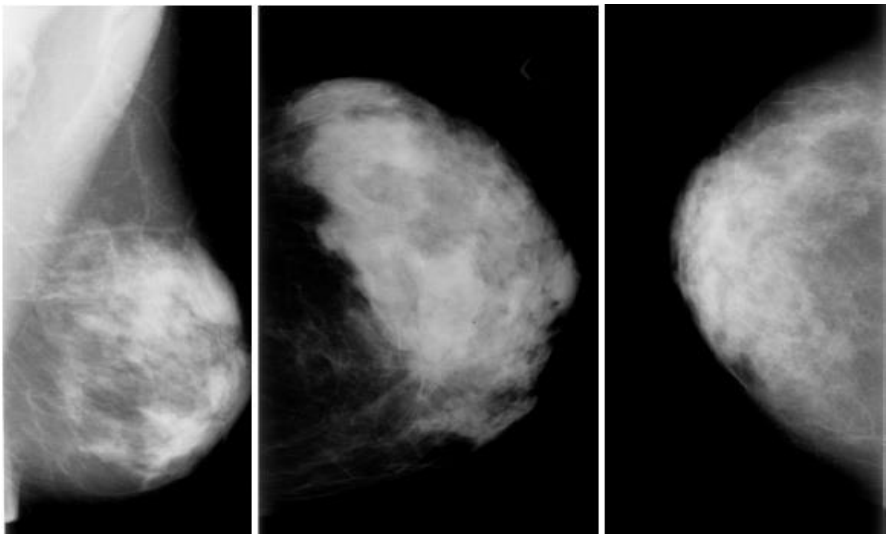


Figure 28. Mammography images from the digital database for screening mammography dataset from kaggle [228].

- **Ultrasound:** Useful for characterizing palpable lumps and distinguishing between solid and cystic masses; often used as a supplemental tool to mammography [233].

- **Magnetic Resonance Imaging (MRI):** Offers high sensitivity, especially in high-risk patients; it is useful for detecting multifocal and multicentric disease [234].
- **Digital Breast Tomosynthesis (DBT):** A 3D imaging technique that improves cancer detection rates and reduces false positives [235].
- **Positron Emission Tomography (PET)/CT:** Generally used for staging and detecting metastasis in advanced cases [236].

2.1. Publicly available datasets for breast cancer research

In the field of breast cancer detection and diagnosis, researchers extensively utilize several publicly available datasets to develop and evaluate machine learning and deep learning models. Among the most commonly used datasets is the Wisconsin Diagnostic Breast Cancer (WDBC) dataset, which contains features extracted from fine needle aspirate (FNA) images and supports binary classification of tumors into benign and malignant categories. Another widely used resource is the Digital Database for Screening Mammography (DDSM), which provides a large collection of mammographic images annotated by radiologists, including cases with verified pathology reports. In addition to these, datasets such as INbreast**,** CBIS-DDSM, and Breast Ultrasound Images Dataset offer rich image-based data for algorithm development across various imaging modalities like mammography, ultrasound, and MRI. The availability of these public datasets has been instrumental in accelerating progress in automated breast cancer detection, enabling reproducible research and benchmarking of algorithmic performance.

Breast cancer remains a major global health concern. The combination of early detection, advanced imaging techniques, and the use of publicly available datasets can significantly enhance the accuracy of diagnosis and treatment planning. These datasets also support the development of Computer Aided Diagnostic (CAD) based solutions to assist clinicians and researchers in fighting breast cancer more effectively.

2.2. Wisconsin diagnosis breast cancer dataset (WDBC)

The dataset used in this study is sourced from the UCI Machine Learning Repository and is known as the Wisconsin Diagnostic Breast Cancer (WDBC) dataset [237]. It comprises 569 instances, each representing a breast tumor diagnosis classified as either benign or malignant. Among these, 357 cases (62.74%) are labeled benign, while 212 cases (37.26%) are malignant. The class distribution is visually represented in Figure 29. The dataset includes 33 attributes in total. These consist of an ID number. The last feature, unnamed, which had the value null for all occurrences, and, a diagnosis label (where *M* denotes malignant and *B* denotes benign), and 30 real-valued features extracted from digitized images of fine needle aspirate (FNA) biopsies of breast masses. These features describe various morphological characteristics of the cell nuclei observed in the biopsy images.

Specifically, the 30 numeric attributes represent ten cell features, including radius, texture, perimeter, area, smoothness, compactness, concavity, concave points, symmetry, and fractal dimension. For each of these ten features, the dataset provides three computed values: the mean, standard error, and worst (for the largest value), leading to 30 numerical descriptors per

instance. The first column contains a unique identifier for each patient, while the second column holds the class label indicating the diagnosis. Columns 3 to 32 consist of the real-valued features, which are used to train and evaluate classification models that predict whether a tumor is benign (non-cancerous) or malignant (cancerous). A complete description of these features is provided in Table 3.

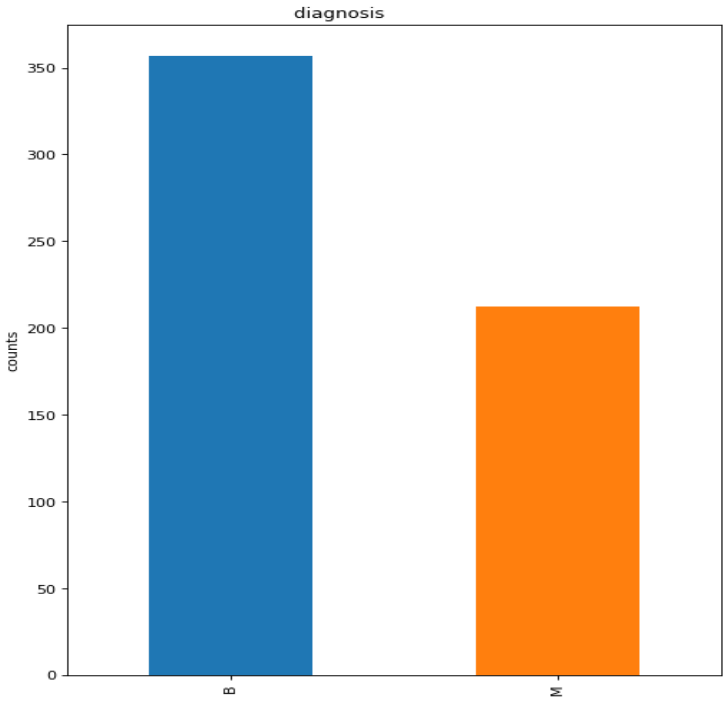


Figure 29. The distribution of the number of Benign and Malignant classes in the WDBC dataset.

Table 3. Wisconsin diagnosis breast cancer features description.

Number	Feature Group	Description
1	Radius	Distance from center to perimeter.
2	Texture	Standard deviation of gray-scale values.
3	Perimeter	Length of the cell boundary.
4	Area	Size of the cell nucleus.
5	Smoothness	Local variation in radius lengths.
6	Compactness	(Perimeter² / Area - 1.0).
7	Concavity	Severity of concave portions of the contour.
8	Concave Points	Number of concave portions of the contour.
9	Symmetry	Symmetry of the nucleus shape.
10	Fractal Dimension	Roughness of the contour (coastline approximation).

2.3. WDBC dataset pre-processing

Purification and modification of the dataset are required before applying ML algorithms to the dataset, it is a necessary step to pre-process the data. Performance and accuracy of the

predictive model are not only affected by the algorithms used, but also by the quality of the dataset and pre-processing. The phases of pre-processing used in this investigation are as follows:

- **Missing values checking:** The dataset contains 569 instances of 33 variables. However, it was discovered that the variable id had no effect on the dataset description or on disease prediction because it merely keeps a serial record of the instances. As a result, the dataset's id feature was removed. Additionally, while conducting additional preprocessing operations on the dataset, it was discovered that the last feature, unnamed: 32, had the value null for all occurrences. This might be a mistake in the data collection process, because of this the feature was also removed from the dataset.

- **Encoding data:** The performance of machine models depends on various aspects. One element that influences performance of the models are the methods used to analyze data and feed it to the model. As such, vital step in encoding data is turning data into categorical variables understood by ML models. Encoding data, elevates model quality and helps in feature engineering. The class label "diagnosis" was expressed as strings of (B= Benign, M= Malignant). This category characteristic must be converted to restricted numbers. This is done to transform data into a format that ML algorithms can understand. Label encoding was used to encode the diagnostic occurrences in this study, and the result was (M=1, B =0).

- **Outliers checking:** An outlier is a statistic or observation that deviates from a distribution's overall pattern. If few data are significantly different or not in range of main trend then those are termed outliers. There skewness results, affecting the mean and standard deviation of the distribution. In this work detects the existence of outliers in the dataset. As a result, outliers were identified and eliminated from their respective features.

- **Normalization :** Feature normalization or standardization is also performed to scale the values and eliminate bias caused by differing feature ranges. This step ensures that all features contribute equally to the learning process and helps algorithms converge more efficiently during training.

As a result, The Wisconsin Diagnostic Breast Cancer (WDBC) dataset is a widely used benchmark in medical machine learning tasks, particularly for classifying tumors as benign or malignant. Before training any model, data preprocessing is essential to ensure quality input. The dataset undergoes several preprocessing steps including handling missing values, normalization or standardization of the features to a common scale, and encoding the target variable into binary values (e.g., benign as 0 and malignant as 1). This helps to enhance model performance and prevent bias toward certain features due to scale differences.

3. Grey Wolf Optimizer Algorithm

The grey wolf (Canis lupus), a member of the canid family, is recognized as an apex predator due to its position at the top of the food chain. These wolves usually live in packs ranging from five to twelve members and exhibit strong social behavior, particularly in hunting and leadership dynamics. The structured social hierarchy within the pack, illustrated in Figure 30, inspired the development of the Grey Wolf Optimizer (GWO) algorithm [11].

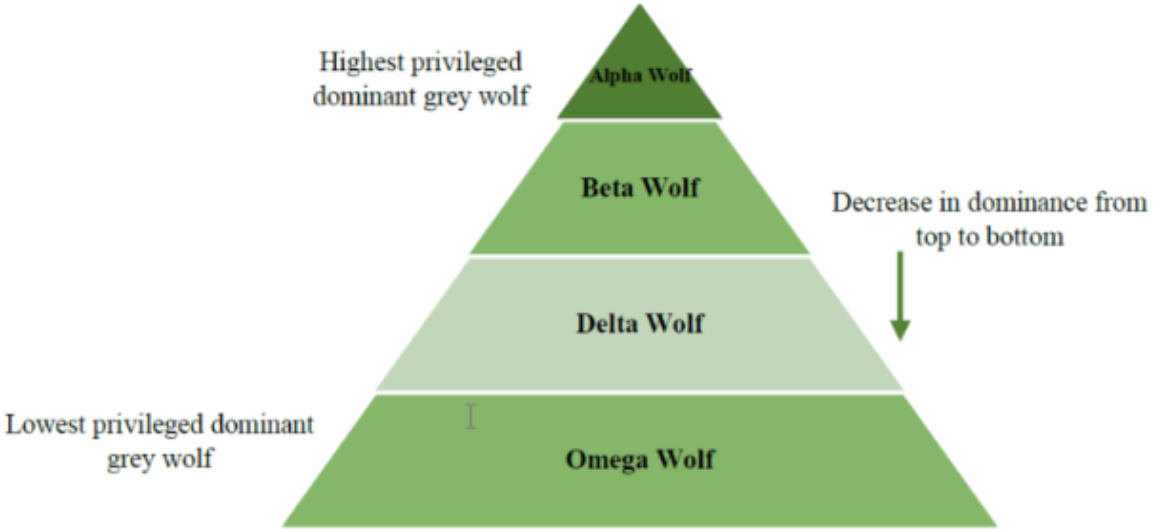


Figure 30. The hierarchy of Grey Wolf Optimizer.

The leader of the wolf pack, whether male or female, is referred to as the alpha. This individual is primarily responsible for making key decisions, such as when to hunt, where to rest, and when the pack should move. Interestingly, the alpha is not necessarily the strongest member physically, highlighting that leadership within the pack relies more on organization and discipline than on sheer strength.

In the grey wolf hierarchy, the beta occupies the second-highest rank, serving as the alpha's subordinate and trusted aide. Beta wolves assist the alpha in decision-making, managing pack responsibilities, and maintaining order among lower-ranking members. They also act as advisors to the alpha and reinforce its commands throughout the pack.

Below the betas are the deltas, who are subordinate to both alphas and betas but hold authority over the lowest-ranking wolves. This group includes roles such as scouts, sentinels, senior members, hunters, and caregivers.

At the bottom of the hierarchy is the omega, the least dominant member of the pack. Omegas must yield to all others, typically eat last, and often serve as the scapegoat or tension reliever within the group. Occasionally, they also take on the role of babysitters for the pack.

3.1. Mathematical model

The social hunting behavior of grey wolves is mathematically represented by utilizing the positions of the fittest wolves [11], namely the alpha, beta, and delta wolves, to guide the

search for the optimal solution. Meanwhile, omega wolves follow the dominant wolves during the hunting process.

3.1.1. Encircling the prey

For hunting, grey wolves chase and encircle the prey. Mathematically, it is modeled as given in Eq. (1) and (2):

$$\vec{X}(t+1) = \vec{X}_P(t) + \vec{A}.\vec{D} \quad (1)$$

$$\vec{D} = \left|\vec{C} \,.\, \vec{X}_P(t) - \vec{X}(t)\right| \quad (2)$$

here t is the indicated current iteration and t + 1 represents next iteration. $\vec{X}$ is the position vector of grey wolf and $\vec{X}_P$ is the position vector of prey. $\vec{A}$ and $\vec{C}$ are coefficient vectors where they are depicted as:

$$\vec{A} = 2\vec{a} \,.\, \vec{r_1} - \vec{a} \quad (3)$$

$$\vec{C} = 2 \,.\, \vec{r_2} \quad (4)$$

where $\vec{r_1}$ and $\vec{r_2}$ are the random vectors in the range of [0, 1] and components of $\vec{a}$ are linearly decreased from 2 to 0 over the courses of iterations [11]

3.1.2. Hunting

Once the prey's location is estimated, the grey wolves encircle it to begin the hunt. The hunting process is directed by the alpha, beta, and delta wolves. Although the exact location of the prey is unknown across the entire search space, it is assumed that the alpha, beta, and delta wolves have knowledge of the prey's position. As a result, the three best solutions are retained, and the remaining wolves (omega wolves) update their positions based on these optimal solutions. The hunting process is mathematically governed by Equation (7), which is derived from Equations (5) and (6).

$$\vec{D_\alpha} = \left|\vec{C_1}.\vec{X}_\alpha - \vec{X}\right|, \qquad \vec{D_\beta} = \left|\vec{C_2}.\vec{X}_\beta - \vec{X}\right| \;, \quad \vec{D_\delta} = \left|\vec{C_3}.\vec{X}_\delta - \vec{X}\right| \quad (5)$$

$$\vec{X_1} = \vec{X_\alpha} - A_1 \,.\, \left(\vec{D_\alpha}\right), \vec{X_2} = \vec{X_\beta} - A_2 \,.\, \left(\vec{D_\beta}\right), \vec{X_3} = \vec{X_\delta} - A_3 \,.\, \left(\vec{D_\delta}\right) \quad (6)$$

$$\vec{X}(t+1) = \frac{\vec{X_1} + \vec{X_2} + \vec{X_3}}{3} \quad (7)$$

Figure 31 demonstrates how a search agent updates its position based on the locations of the alpha, beta, and delta wolves within a 2D search space. As shown, the final position is randomly influenced by the positions of the alpha, beta, and delta agents. Therefore, these wolves collectively determine the prey's location, while the other wolves adjust their positions randomly around it.

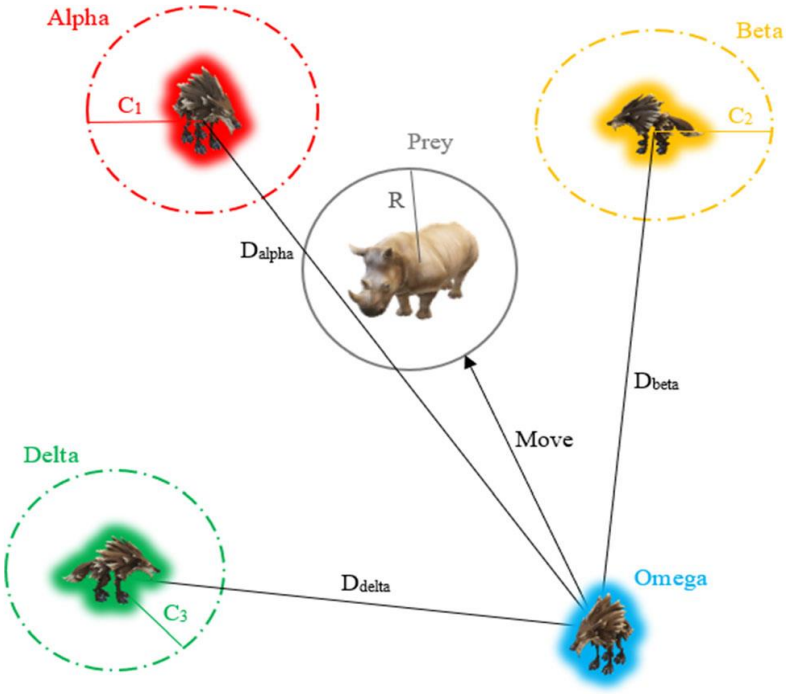


Figure 31. Position updating in the Grey Wolf Optimizer [11].

During the search and hunting process, exploration and exploitation are managed through the parameters $|\overrightarrow{A}|$ and $|\overrightarrow{C}|$. The parameter $|\overrightarrow{A}|$ gradually decreases from 2 to 0, ensuring a balance between exploration and exploitation. When $|\overrightarrow{A}|>1$, the grey wolves spread out to explore the search space for the prey, while when $|\overrightarrow{A}|<1$, they converge towards each other to attack the prey (See figure 32). The inherent randomness in this process helps prevent the wolves from becoming trapped in local minima, promoting a more effective global search.

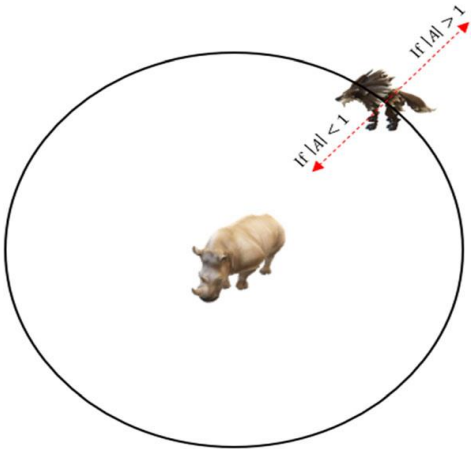


Figure 32.Attacking prey and searching prey [11].

Algorithm 1 : Pseudocode of GWO

Algorithm: Grey Wolf Optimizer (GWO)
Input: MaxIterations (T), Population Size (N), Search Space.
1. Initialize the population of grey wolves (X_i, i = 1, 2 ... N) randomly
2. Evaluate the fitness of each wolf
3. Identify the three best wolves:
- Alpha (X_α) -> Best solution
- Beta (X_β) -> Second best solution
- Delta (X_δ) -> Third best solution
4. For each iteration t = 1 to T:
For each wolf (X_i):
Update the control parameters a (linearly decreases from 2 to 0).
Compute coefficient vectors A and C using Equation 3 and 4.
Compute new positions relative to α, β, and δ Using Equation 5-7.
Evaluate new fitness value.
Update X_α, X_β, and X_δ if better solutions are found.
5. Return the best solution Xα

3.2. Literature review about Modified GWO

The Grey Wolf Optimizer (GWO) has emerged as a powerful and widely adopted metaheuristic algorithm inspired by the leadership hierarchy and hunting strategies of grey wolves in nature. Since its introduction, GWO has demonstrated impressive performance across various optimization tasks [238]. However, like many metaheuristics, it also faces challenges such as premature convergence, lack of population diversity, and difficulty in balancing exploration and exploitation. To address these limitations, numerous researchers have proposed enhanced and modified versions of GWO. These variations aim to improve the algorithm's adaptability, convergence speed, and overall performance by integrating concepts from reinforcement learning, random walks, evolutionary strategies, and swarm intelligence, among others. This literature review explores a range of such modified GWO approaches.

A study [239] introduced the Variable Weights-GWO, an enhanced version of the Grey Wolf Optimizer that incorporates variable weights to better preserve the social hierarchy within the wolf pack. In this approach, the weight assigned to the alpha position must always be equal to or greater than those of the beta and delta positions, with the beta's weight also required to be at least equal to that of the delta. Furthermore, a new formula was proposed to adjust the control parameter, aiming to reduce the chances of the algorithm getting stuck in local optima. The performance of VW-GWO was evaluated against ALO, PSO, BA, and the original GWO across 11 benchmark functions, and the results validated the effectiveness of the proposed method.

Another researcher [240] proposed an enhanced version of GWO called Improved Alpha-Guided GWO (IAgGWO), which incorporates a novel guidance mechanism along with a mutation operation to speed up convergence and prevent the algorithm from getting trapped in local optima. The use of scalar coefficients A and C simplifies the implementation of the

algorithm. The study demonstrated the superiority of IAgGWO by comparing it with four other algorithms across 35 benchmark functions and through its application to the engineering design problem of a two-stage operational amplifier.

A more precise model [241] was developed to mimic the hierarchy of authority and group hunting tactics used by grey wolves in the wild. According to this new model, each wolf moves straight in the direction of the prey's predicted location and the location of each wolf is dynamically evaluated by the leader wolves . Evaluations using the CEC2017 benchmark suite showed that the improved optimizer significantly outperforms the original GWO and its later variants in terms of both convergence speed and solution stability.

To enhance the wolf pack's ability to locate prey, a study [242] proposed a modified version of GWO known as Random-Walk-GWO, which incorporates random walks for the leading wolves. Experimental results based on the CEC 2014 benchmark revealed that RWGWO outperformed both the standard GWO and other metaheuristic algorithms.

Another proposed algorithm, RBGWO [243], aims to enhance the overall efficiency of the search process by effectively balancing exploration and exploitation. It introduces three consecutive improvement strategies, including a random walk guided by Student's t-distributed random values and a social hierarchy mechanism. The first strategy updates each grey wolf's position using weight-based variables. The second incorporates a random walk approach inspired by [242] to refine position updates. The third introduces a novel randomization technique to further boost the search efficiency and reinforce the random walk process. When tested on the CEC 2014 benchmark functions at various scales, RBGWO outperformed the standard GWO.

The Experienced GWO (EGWO) [244] integrates reinforcement learning techniques to determine the optimal actions to take during different phases of the optimization process and across various regions of the search space. A neural network is used to store and utilize this experiential knowledge. The proposed EGWO was evaluated against the original GWO, PSO, and GA in two key optimization tasks: feature selection and neural network weight adaptation. The results demonstrated that EGWO delivered significantly improved performance over the compared algorithms.

The Improved Grey Wolf Optimizer (I-GWO) [245] was designed to address global optimization and engineering design challenges. To tackle issues such as limited population diversity, imbalance between exploration and exploitation, and premature convergence in the original GWO, the I-GWO incorporates a novel mobility strategy known as the Dimension Learning-based Hunting (DLH) search method. Experimental results on various engineering design problems highlighted the algorithm's effectiveness and versatility.

An Enhanced Grey Wolf Optimizer (EGWO) was proposed in [246], incorporating Lévy flight and binomial crossover mechanisms to improve the grey wolves' hunting behavior. This enhanced strategy was also applied to optimize clustering processes. The EGWO's performance was evaluated using seven benchmark datasets from the UCI repository and compared against five other clustering algorithms. Empirical results demonstrated that EGWO is a robust and promising approach for efficient large-scale data clustering.

3.3. Modified GWO using weighted position update method

In the conventional GWO algorithm, the position of search agents (omega wolves) is updated by averaging the positions of the top three wolves including alpha, beta, and deltas wolves, as defined in Equation (7). While simple, this uniform averaging approach may lead to premature convergence and low-quality solutions, particularly in complex or high-dimensional search spaces. To overcome this limitation, a weighted position update mechanism is adopted, inspired by the work of S. Kumar and M. Singh [8]. This approach assigns varying weights to the contributions of the alpha, beta, and delta wolves based on their fitness, allowing the more optimal leaders to have a greater influence on the position updates. This modification enhances both the convergence speed and solution quality by dynamically adjusting the influence of each leading wolf.

The mathematical formulation for this technique is presented in Equations (8) and (9), which redefine the agents' movement in a more adaptive and fitness-aware manner.

$$W_1=A_1*C_1, \qquad W_2=A_2*C_2, \qquad W_3=A_3*C_3 \tag{8}$$

$$X(t+1) = (W_1*X_1 + W_2*X_2 + W_3*X_3) / (W_1+W_2+W_3) \tag{9}$$

3.4. Grey Wolf Optimizer for image processing

This section presents a review of relevant studies on image processing, with a particular focus on medical imaging applications. Various enhancements to the Grey Wolf Optimizer (GWO) have been proposed to improve segmentation, classification, and feature selection tasks. The following subsections provide an in-depth discussion of these approaches and their effectiveness in different image processing domains.

The rapid expansion of multimedia content, particularly images, on social media platforms has intensified interest in content-based image retrieval (CBIR) systems. Despite the emergence of various CBIR techniques, face recognition continues to present significant challenges. To address this, a study [247] introduced an enhanced version of the Grey Wolf Optimizer, called Varying Weight GWO (VW-GWO), for optimizing a Support Vector Machine (SVM)-based facial recognition model. Simulation results demonstrated that VW-GWO significantly improved classification accuracy and stability.

In another advancement, a Mixed GWO approach [248] was proposed to effectively handle optimization problems involving continuous, discrete, or mixed variables. Leveraging this bio-inspired technique, the study successfully performed simultaneous denoising and unmixing of multispectral images.

Furthermore, an Ensemble Grey Wolf Optimizer (EGWO) [249] was developed by integrating an elite-based search strategy and a modified position update equation. This method showed promising results when tested on 12 images from the USC-SIPI dataset.

In the realm of medical imaging, chest X-ray (CXR) images have become preferred over CT scans for COVID-19 detection, owing to their clearer representation of lung abnormalities. To enhance diagnostic accuracy and reduce reliance on manual interpretation, a

three-stage classification model CXGNet was introduced [250]. It combines an enhanced GWO with genetic algorithm (EGWO-GA) and deep learning-based convolutional neural networks (DLCNN) for optimal feature selection. This model outperformed traditional diagnostic methods such as RT-PCR, antigen, and serological tests in both speed and efficiency.

Segmentation, a critical stage in image processing, also benefits from GWO-based enhancements. Among the popular segmentation methods, histogram-based thresholding stands out for its simplicity and effectiveness. For multi-level thresholding tasks, researchers [251] proposed a Discrete Multi-Objective Shuffled GWO (D-MOSG) algorithm, which delivered superior segmentation performance across various benchmarks. Experimental results confirmed that the Discrete Multi-Objective Shuffled Grey Wolf Optimizer (D-MOSG) outperforms other algorithms in multi-level image thresholding tasks, delivering superior segmentation accuracy.

In a related study [252], a Modified Grey Wolf Optimizer (MGWO) was introduced to enhance the original GWO algorithm. This variant was applied to the segmentation of leaf spot diseases in maize using four distinct threshold levels. The results demonstrated that MGWO delivers competitive performance, highlighting its effectiveness as a robust optimizer for multi-threshold image segmentation applications.

4. Modified Grey Wolf Optimizer and Random Forest strategy

Using the Modified Grey Wolf Optimization (MGWO) algorithm for breast cancer classification enhances diagnostic accuracy by selecting the most relevant features from complex medical datasets. In this section we implement a effective method for breast cancer classification based on feature selection and classification by integrating GWO with Machine learning.

4.1. Modified GWO and random forest strategy

In this section, breast cancer classification was performed using the WDBC dataset, which includes various features extracted from digitized images of breast tissue samples. The process began with comprehensive data preprocessing, including handling missing values, normalizing features for consistency, and encoding target labels for binary classification (benign vs. malignant). Feature selection was then applied using a Modified Grey Wolf Optimization (MGWO) algorithm, which improves the standard GWO by enhancing its search capability to effectively identify the most relevant features while eliminating redundant or irrelevant ones. Classification was conducted using a Random Forest (RF) classifier, chosen for its robustness, ensemble learning approach, and efficiency in handling high-dimensional data. Two experiments were carried out: the first involved training the RF classifier with all original features, while the second used only the optimized feature subset selected by MGWO.

The final step involved evaluating the overall classification strategy using key performance metrics, including accuracy, precision, recall, and F1-score. The results demonstrated that the MGWO-based feature selection not only effectively reduced the dimensionality of the dataset

but also maintained or improved the classification performance. This highlights the advantage of integrating an intelligent feature selection method with a powerful classifier like Random Forest. Figure 33 illustrates the workflow of the two experimental scenarios proposed for breast cancer diagnosis.

In the first scenario, the process consists of three main phases. The first phase is data preprocessing, which is common to both scenarios. During this phase, data cleaning and filtering were carried out to eliminate noise and prevent the generation of ineffective rules or patterns. Specifically, the WDBC dataset was cleaned, and outliers were removed using the outer line (outlier detection) approach. The second phase involves classification using a Random Forest classifier trained on all the original features of the dataset. The third phase is the evaluation of classification performance using appropriate metrics.

The second scenario consists of four main phases. It begins with the same data preprocessing step as the first scenario. Next, a feature selection process is applied using the Modified Grey Wolf Optimization (MGWO) algorithm to identify the most significant features contributing to classification accuracy. In the third phase, a Random Forest classifier is again used, but this time trained only on the selected features. Finally, the fourth phase involves the evaluation of classification performance to compare the effectiveness of the reduced feature set against the full set used in the first scenario.

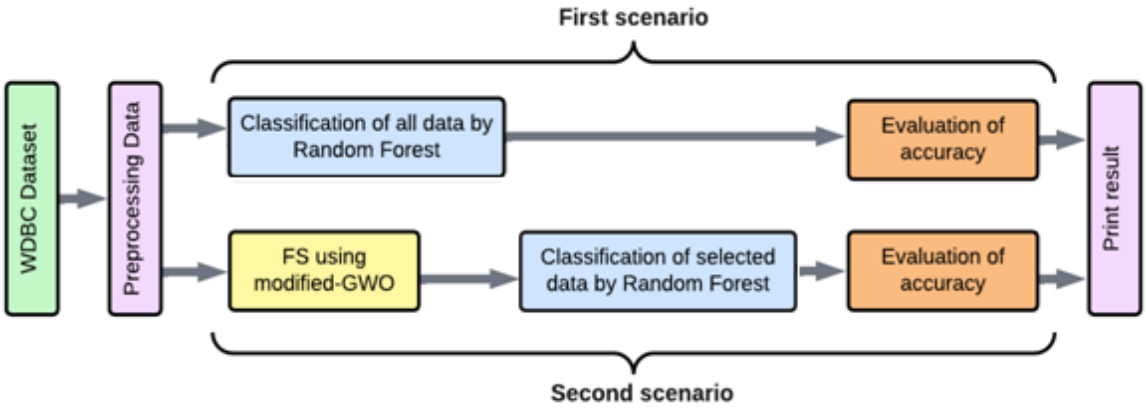


Figure 33. Proposed method for accurate breast cancer classification.

4.2. Experimental results

In this study, the primary objective was to enhance classification performance and improve diagnostic accuracy by reducing the feature dimensionality using the Modified Grey Wolf Optimization (MGWO) algorithm. During the experiments, the MGWO was configured to run for 20 iterations with 10 search agents. For the classification task, a Random Forest (RF) classifier was employed to distinguish between malignant and benign tumors, chosen for its high accuracy, robustness, and ability to handle complex datasets. The most promising results were achieved through a hybrid approach that combined MGWO for feature selection with the RF classifier for final prediction. As shown in Table 4, the proposed method demonstrated improved performance across all evaluation metrics, including sensitivity, specificity,

precision, F1-score, and accuracy, when dimensionality reduction was applied using MGWO prior to classification with Random Forest.

Table 4. Classification results of the proposed MGWO-RF approach using different performance measures.

Performance Measures	Classification results (%)	
	Without feature selection	Feature Selection using MGWO-RF
Sensitivity	96,3	98,1
Specificity	97,8	98,9
Precision	96,3	98,1
F1-score	96,3	98,1
Accuracy	97,2	98,6

4.2.1. Comparing the classification results between the modified GWO-RF and the base GWO-RF

Table 5 presents a performance comparison between the proposed Modified GWO-RF approach and the baseline GWO-RF method. This comparison aims to evaluate the impact of integrating a weighted position update mechanism into the original GWO algorithm. The results clearly demonstrate that the modified version significantly enhances classification performance. Specifically, the Modified GWO-RF approach achieved an accuracy of 98.6%, an F1-score of 98.1%, and a sensitivity of 98.1%, outperforming the baseline across these key metrics. These improvements highlight the effectiveness of the proposed enhancements in optimizing feature selection and boosting diagnostic accuracy.

Table 5. Comparing classification results between the modified GWO-RF approach and the base GWO-RF approache.

Performance Measures	Classification results (%)	
	Modified GWO with RF	Original GWO with RF
Sensitivity	98,1	96,3
Specificity	98,9	98,9
Precision	98,1	98,1
F1-score	98,1	97,2
Accuracy	98,6	97,9

4.2.2. Comparing classification results between the modified GWO-RF and existing feature selection approaches

Table 6 provides a comparative analysis of the classification performance between the proposed MGWO-RF approach and several existing feature selection-based methods for breast cancer detection. The comparison highlights how different techniques perform in terms of key evaluation metrics such as accuracy, sensitivity, and F1-score. From the results, it is

evident that the proposed MGWO-RF method consistently outperforms the other approaches, demonstrating superior effectiveness in selecting the most relevant features and enhancing classification performance. These findings validate the strength of integrating the Modified Grey Wolf Optimization with the Random Forest classifier for accurate and reliable breast cancer diagnosis.

Table 6. Comparing results of the modified GWO-RF approach with existing feature selection approaches.

Approaches	Authors	Years	Number of features	Accuracy %
FS-KNN	Sayed et al.[5]	2019	14	90,28
FS – GBDT	Rao et al. [6]	2019	14	92.80
FS-KNN	Abdel- Basset et al.[7]	2020	16	94,82
FS + EGWO-SVM	S. Kumar & M. Singh[8]	2021	6	98,24
Proposed approach	Proposed	2022	12	98,60

5. Hybrid algorithm using correlation and Modified GWO based feature selection

Feature selection is a crucial step in many machine learning tasks, including classification, where the goal is to identify a subset of relevant features that enhance model performance while reducing computational complexity. In high-dimensional datasets, such as the Wisconsin Diagnostic Breast Cancer (WDBC) dataset, feature selection becomes even more significant due to the risk of overfitting, computational inefficiencies, and the presence of noisy or irrelevant features.

Traditional feature selection methods often rely on either filter, wrapper, or embedded techniques. Filter methods evaluate the relevance of features based on their statistical properties, while wrapper methods assess subsets of features by evaluating model performance. However, these methods have their limitations, particularly when dealing with correlated features that can result in redundancy and hinder classification performance.

To address these challenges, a hybrid approach that combines correlation-based feature selection with the optimization power of the Modified Grey Wolf Optimizer (MGWO) is proposed. This hybrid algorithm aims to first remove highly correlated features, ensuring that only independent and non-redundant features are retained, and then further optimize this subset using MGWO to identify the most relevant features for classification. By combining correlation-based feature selection with the Modified Grey Wolf Optimizer, this hybrid algorithm aims to achieve a balance between reducing the feature space and maintaining or enhancing classification accuracy, providing an efficient and reliable method for breast cancer classification and other similar high-dimensional classification tasks.

The experimental results indicate that the proposed method successfully achieves a balance between feature relevance and diversity, resulting in improved performance across multiple evaluation metrics.

5.1. Pearson Correlation technique

The Pearson Correlation Coefficient is a statistical measure used to evaluate the linear relationship between two continuous variables [253], [254]. It helps determine the degree to which one variable can be predicted based on the behavior of another. In the context of feature selection, this method is commonly used to assess the correlation between input features and the target variable. Ideally, the selected features should exhibit a strong correlation with the target variable, while maintaining minimal correlation with one another to avoid redundancy. When two features are highly correlated with each other, they carry overlapping information, and retaining both may lead to unnecessary complexity in the model. In such cases, only one of the correlated features is typically retained, as the other does not contribute additional predictive value. The correlation coefficient (r) ranges from -1 to +1, a value of +1 indicates a perfect positive correlation, -1 indicates a perfect negative correlation, and 0 signifies no linear correlation. The closer the absolute value of the coefficient is to 1, the stronger the linear relationship between the two variables.

5.2. Feature selection and classification approach using Correlation and Modified Grey Wolf Optimizer (MGWO)

The proposed method aims to enhance the classification accuracy for the Wisconsin Diagnostic Breast Cancer (WDBC) dataset by performing feature selection in two level: first using correlation-based feature selection to remove redundant features, and then applying the Modified Grey Wolf Optimizer (MGWO) to optimize the remaining uncorrelated features. The resulting feature set is subsequently fed into classifiers such as Support Vector Machine (SVM)**,** Random Forest (RF)**,** and Naïve Bayes (NB) for classification (see Figure 34).

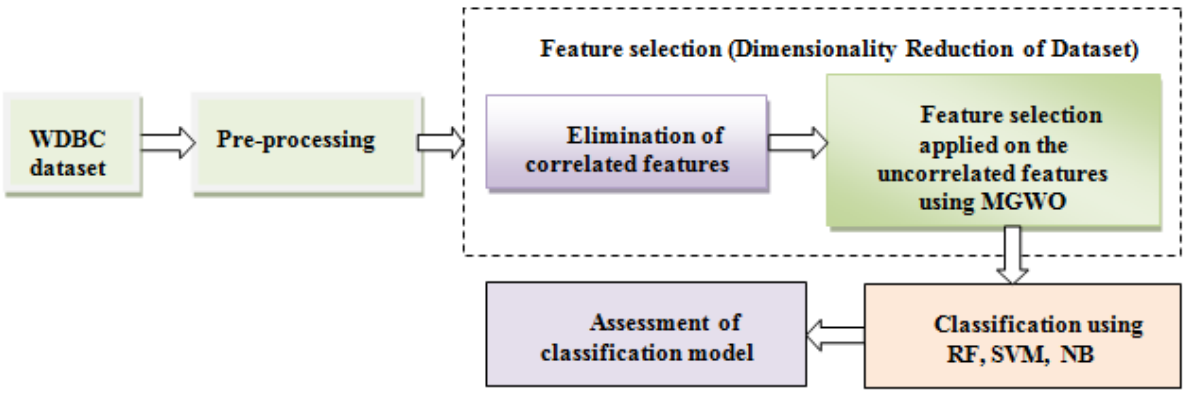


Figure 34. Flowchart of the suggested breast cancer classification method based feature selection.

In our proposed model, the WDBC dataset was utilized, and a preprocessing step was performed to clean the data by removing irrelevant or unused features. The feature selection (FS) process employed a two-step strategy that combined a correlation-based method with the Modified Grey Wolf Optimization (MGWO) algorithm to identify the most relevant attributes. For the classification task, we implemented multiple machine learning algorithms,

including Support Vector Machine (SVM), Random Forest (RF), and Naïve Bayes (NB), to evaluate the model's performance.

The proposed approach for breast cancer classification follows a two-stage process: feature selection and classification. The first stage focuses on dimensionality reduction using a combination of filter and wrapper methods. Initially, correlation-based feature selection is applied to the WDBC dataset to remove highly correlated features both among themselves and with the target variable (cancer tumor or not). This technique helps in reducing redundancy and improving the feature space by selecting only the most independent and relevant features. As a result, the number of features is reduced from 30 to 16, which enhances the efficiency and performance of subsequent steps (refer to Section 5.2.1 for further details).

Once the correlated features are eliminated, the Modified Grey Wolf Optimizer (MGWO) is applied to the remaining 16 non-correlated features. The MGWO algorithm is designed to select the most significant and relevant features, optimizing the feature set further. By applying the MGWO to 16 features instead of the original 30, the algorithm can operate more efficiently, providing better results with fewer variables. This step ensures that the selected features contribute maximally to the classification process and improve the overall accuracy of the model (as described in Section 5.2.2).

In the second stage of the process, the reduced and optimized feature set is used to classify the breast cancer data using three different machine learning classifiers: Support Vector Machine (SVM), Random Forest (RF), and Naïve Bayes (NB). These classifiers are chosen for their robustness in handling high-dimensional datasets and their ability to provide reliable predictions for breast cancer classification. The overall process is represented in Algorithm 2, which outlines the two main stages of the proposed method: the feature selection process, which combines correlation-based selection and MGWO, followed by the classification stage, where machine learning classifiers are used to perform the final classification task.

Algorithm 2. Pseudocode of the proposed method.
Phase 1: Input: upload WDBC Dataset Step1 : Preprocessing and removing unused features from Dataset. Step2 : Feature selection with correlation technique and removing correlated features from original dataset (see Section 5.2.1). Step3 : Feature selection applied on uncorrelated features using MGWO algorithm (see Section 5.2.2). Output : Selected features Phase 2: Classification of breast cancer using selected features based on the output of Phase1 and assessment the accuracy of classification.

5.2.1. Feature selection using correlation

As shown in Figure 35, a heat map was employed to analyze the correlations between the features of the dataset. The analysis revealed a strong correlation between the features "radius-mean", "parametric-mean", and "area-mean", as all these features provide similar information regarding the size of breast cancer cells. Given the redundancy in the information conveyed by these features, only the "area-mean" feature was selected to effectively represent the size of the breast cancer cells, streamlining the feature set while preserving essential information.

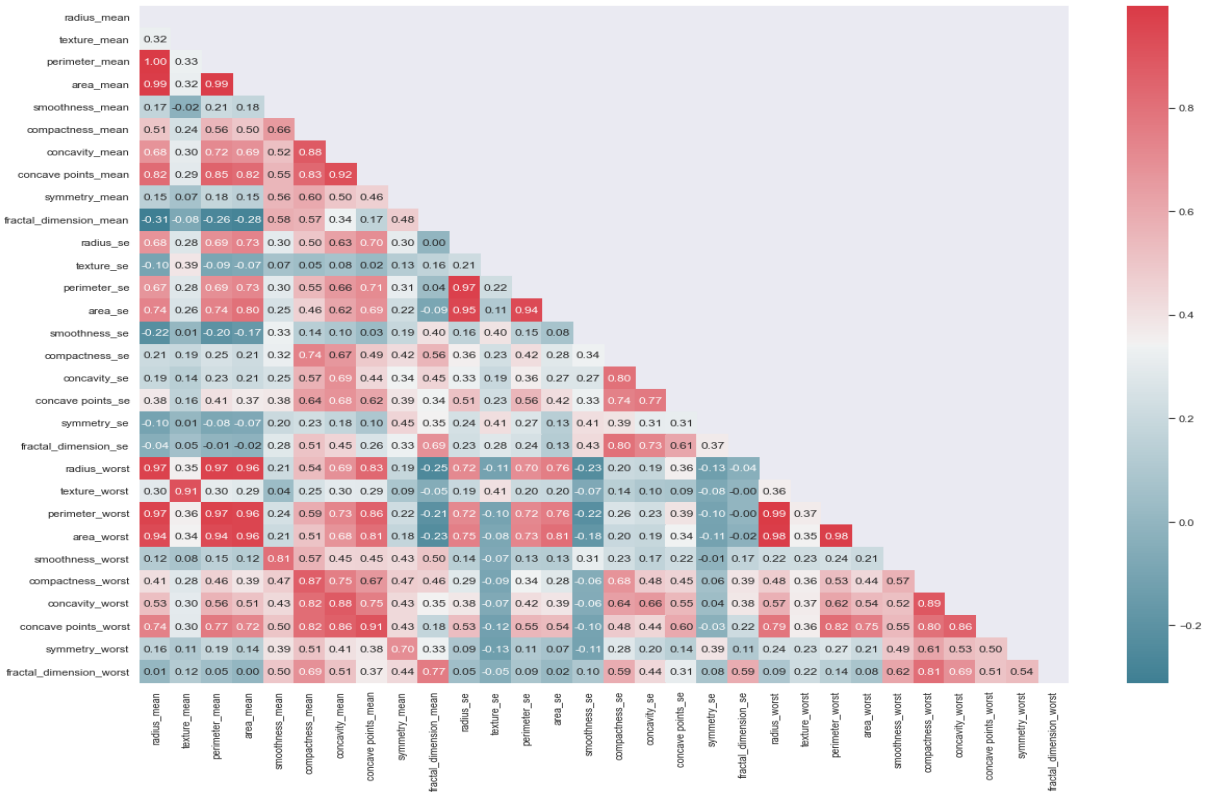


Figure 35. Heat-map plot showing the correlations among all features of WDBC.

A total of 14 features were removed, including 'perimeter-mean', 'radius-mean', 'compactness-mean', 'concave points-mean', 'radius-se', 'perimeter-se', 'radius-worst', 'perimeter-worst', 'compactness-worst', 'concave points-worst', 'compactness-se', 'concave points-se', 'texture-worst', and 'area-worst'. Following this feature elimination process, 16 features remained for further analysis. The relationships between these selected features are depicted in Figure 36.

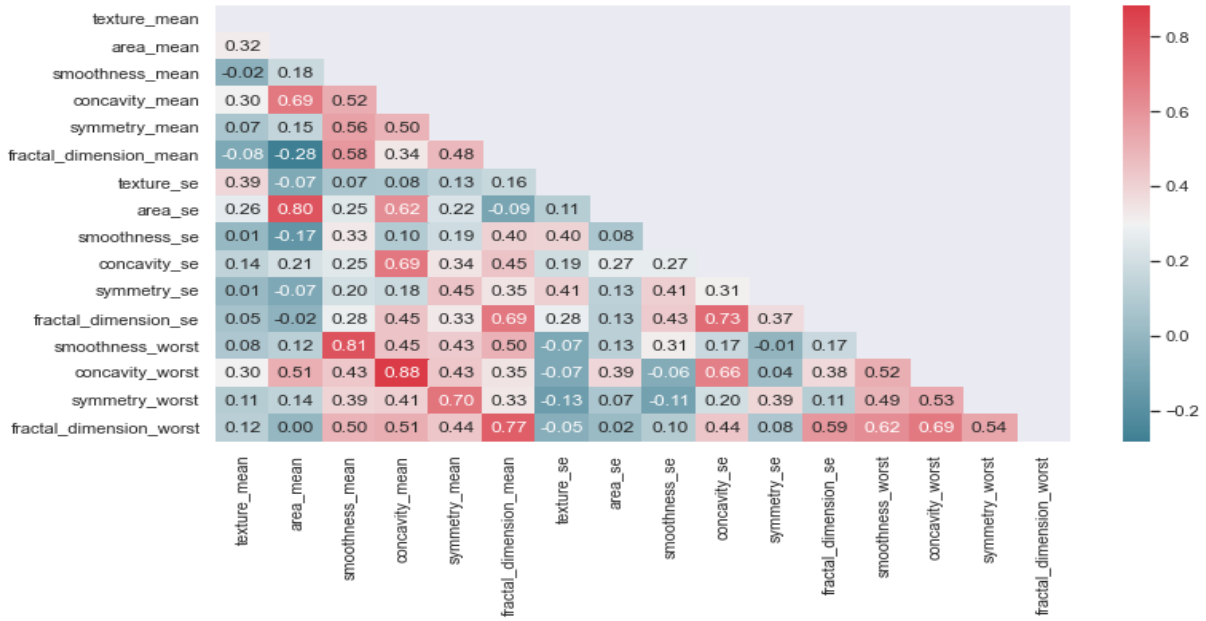


Figure 36. Heat-map plot showing the correlations among selected features of WDBC.

5.2.2. Feature Selection using Modified GWO

To effectively detect breast cancer tumors in our study, we leveraged the strengths of the Modified Grey Wolf Optimizer (MGWO) algorithm to identify the most relevant subset of features. Algorithm 3 presents the pseudo code of the MGWO algorithm. As previously mentioned, in the MGWO approach, Equation (8) is employed instead of Equation (7) to generate more accurate and relevant results. Various classifiers were then trained using the feature subset determined by the MGWO algorithm. An illustrative example of the position vector used by the alpha search agent in the MGWO algorithm for feature selection is depicted in Figure 37. The position vector consists of binary values (1 or 0) for each feature. For an n-dimensional problem, the position vector contains n bits. A feature is excluded from the subset if its corresponding position in the vector is 0, while it is selected if the value is 1. Therefore, the number of selected features corresponds to the number of 1s in the position vector, representing the optimal subset of features chosen by the algorithm. Algorithm 4 further details how the MGWO algorithm effectively identifies the optimal feature subset. The most important features selected through this method, after applying MGWO to the uncorrelated features, include texture-mean, area-mean, concavity-mean, symmetry-mean, fractal-dimension-mean, area-se, concavity-se, smoothness-worst, and fractal-dimension-worst. These nine features were found to be the most significant for efficiently identifying breast cancer and achieving optimal classification accuracy.

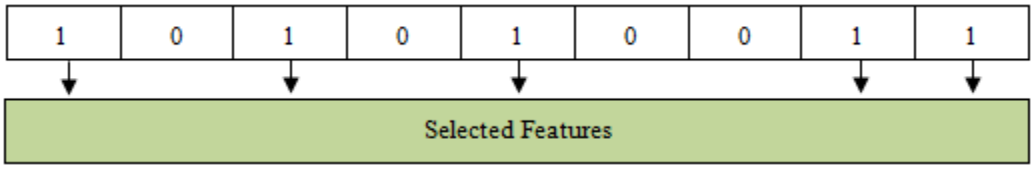


Figure 37. Representation of feature selection technique with MGWO.

Algorithm 3. Pseudocode of Modified GWO
Input: - Dataset, Number of features (Dim), Population Number of Iteration Output: Minimum number of selected features by MGWO initialize alpha, beta, and delta positions Initialize alpha pos, beta pos, and delta pos Initialize the positions of search agents For each Iteration For each Searchagent no - Calculate objective function for each search agent - Update Alpha pos, Beta pos, and Delta pos end For For each Searchagent no For each features Update the Position of search agents including omegas using Equations (1)-(6) and Equation (8) end For end For end For return Alpha pos.

Algorithm 4. The optimal subset of features using modified GWO.

For each feature in alpha pos[i] (i=1,2,. . . ,Dim)
if (alpha pos[i] > 0, 5)
alpha pos[i] =1
Else if (alpha pos[i] < 0, 5)
alpha pos[i] = 0
End if
End for

5.3. Breast cancer classification steps

Following the feature selection process, the classification step aims to accurately distinguish between malignant and benign breast cancer cases using the most relevant features. In this work, three popular supervised machine learning classifiers were applied:

Support Vector Machine (SVM), Random Forest (RF), and Naïve Bayes (NB). Each classifier was trained and evaluated using the features selected by the metaheuristic-based optimization process. SVM was used due to its strong performance in high-dimensional spaces and its ability to find optimal hyperplanes for classification. Naïve Bayes, despite its simplicity and assumptions of feature independence, was included for its fast computation and effectiveness on small datasets. Random Forest, an ensemble learning method based on decision trees, was chosen for its robustness against overfitting and its ability to model complex, non-linear relationships.

The performance of each classifier was assessed using several standard evaluation metrics, including accuracy, precision, sensitivity, specificity, F1-score, and the area under the ROC curve (AUC). The results showed that the Random Forest classifier consistently outperformed both SVM and NB across these metrics. Its ensemble nature allows it to better capture interactions among features, making it particularly effective when working with the optimized feature subset. These findings indicate that RF is a suitable and reliable choice for breast cancer classification when combined with a robust feature selection method, offering both high classification accuracy and generalization capability.

5.4. Experimental Results

In this study, feature selection (FS) was executed using a correlation-based technique integrated with a Modified Grey Wolf Optimization (GWO) algorithm. To evaluate the effectiveness of the selected features, multiple machine learning classifiers, including Support Vector Machine (SVM), Random Forest (RF), and Naïve Bayes (NB) were employed. The proposed hybrid approach was validated using the Wisconsin Diagnostic Breast Cancer (WDBC) dataset.

The experimental setup was developed in Python, with the Modified GWO configured to run for 20 iterations using 10 search agents. All simulations were conducted on a system equipped with an Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz and 8GB of RAM. This configuration was chosen to balance computational efficiency with the capability to process feature-rich biomedical data. The results demonstrate the robustness and classification accuracy improvements achieved by incorporating the enhanced feature selection strategy.

5.4.1. Comparison of different performance metrics between different classifiers

The table 7 presents a comparative evaluation of three machine learning classifiers including SVM, RF, and NB based on five standard performance metrics: precision, F1-score, sensitivity, specificity, and accuracy. This analysis specifically emphasizes the performance of the Correlation–Modified GWO in the context of breast cancer classification. Among the evaluated classifiers, combining Correlation-MGWO with Random Forest demonstrates the most balanced and reliable performance in breast cancer classification. It achieves the highest accuracy (99.12%), indicating superior overall predictive power. Furthermore, its high precision, sensitivity, and F1-score, along with its strong specificity, confirm its robustness and efficacy in distinguishing between malignant and benign tumors. These results suggest that Correlation-MGWO with Random Forest is a highly effective model for clinical decision

support systems aimed at breast cancer diagnosis, offering a compelling combination of accuracy, reliability, and interpretability.

Table 7. Comparison of different performance metris between different classifiers for breast cancer classification using the data of Confusion Matrix.

Evaluation measurement	SVM	RF	NB
Precision	**100%**	97,6%	100%
F1-score	**96,4%**	95,2%	92,5%
Sensitivity	93%	93%	86%
Specificity	**100%**	98,6%	100%
Accuracy	97,4%	**99,12%**	96,5%

5.4.2. Comparing the classification accuracy between CBGWO (Correlation + Base GWO) and CMGWO (Correlation + Modified GWO)

Table 8 presents a comparative analysis of classification accuracy for three machine learning algorithms including RF, SVM, and NB, the algorithm evaluated under three experimental conditions: (i) without feature selection, (ii) with feature selection using Correlation and Base Grey Wolf Optimizer (CBGWO), and (iii) with feature selection using Correlation and Modified Grey Wolf Optimizer (CMGWO). This analysis illustrates the effectiveness of the Modified Grey Wolf Optimizer-based feature selection techniques in enhancing classification performance, particularly for breast cancer diagnosis.

Overall, the results clearly demonstrate that feature selection plays a crucial role in improving classifier performance. Among the feature selection techniques, the Correlation + Modified GWO (CMGWO) consistently yields the highest accuracy across all classifiers, confirming its superiority in identifying informative and non-redundant features. Random Forest shows the best absolute performance across all scenarios, but the most substantial relative improvement is observed in SVM. These findings validate the effectiveness of the proposed CMGWO approach as a robust feature selection strategy for breast cancer classification tasks.

Table 8. Comparison of classification accuracy using proposed approach between CBGWO and CMGWO.

Classifiers	Without Feature selection	CBGWO	CMGWO
RF	97.07%	98.83%	**99.12%**
SVM	92.10%	92.98%	97.36%
NB	94.40%	93.85%	96.50%

5.4.3. Comparison of the classification accuracy between different classifiers using ROC curve (receiver operating characteristic curve)

ROC curve helps to better understand the power of a machine learning algorithm. We can easily observe in Figure 31 that RF is the perfect classifier. The Area Under the Curve (AUC) is the measure of the ability of a classifier to distinguish between classes, and it is used as a summary of the ROC curve. The higher the AUC, the better the performance among classifiers. From Figure 38, we see that RF gives good results compared with SVM and NB classifier in terms of ROC-AUC metric by achieving an AUC criterion equal to 99,3%.

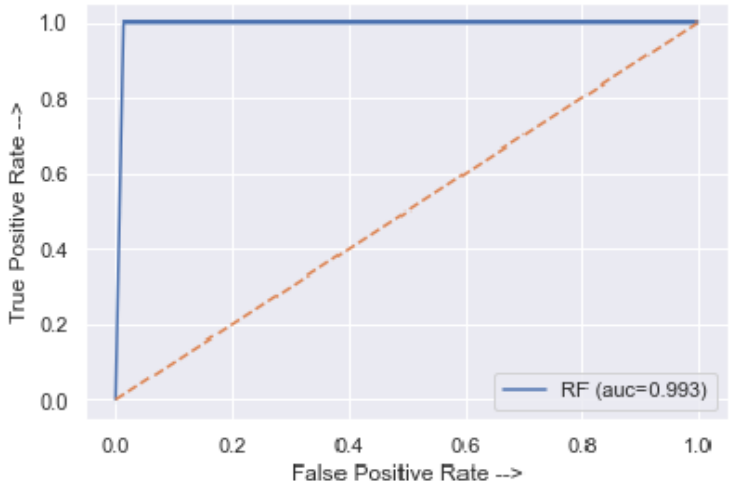


Figure 38. ROC curve metric of RF classifier.

The second best classifier was SVM by obtaining 97% as shown in Figure 39. Figure 40 represents the ROC-AUC metric obtaining by NB classifier and achieving 94,6%.

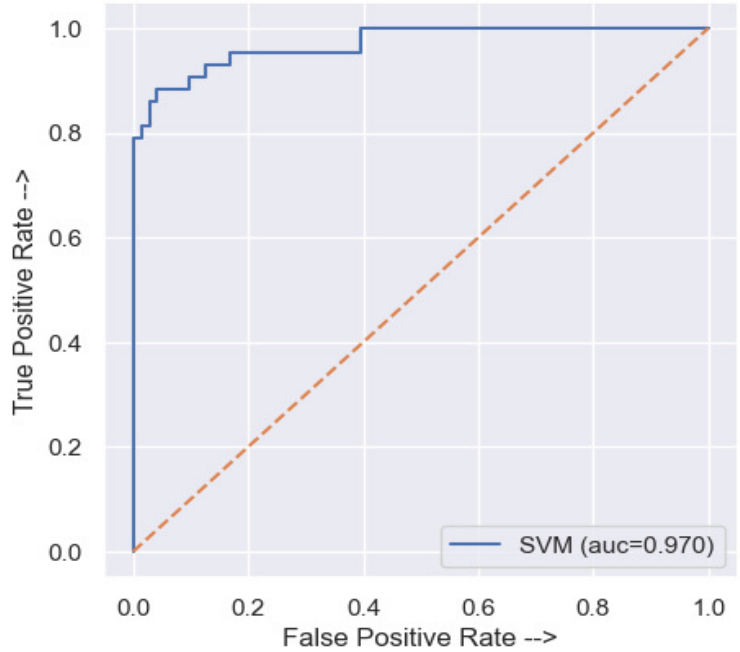


Figure 39. ROC curve metric of SVM classifier.

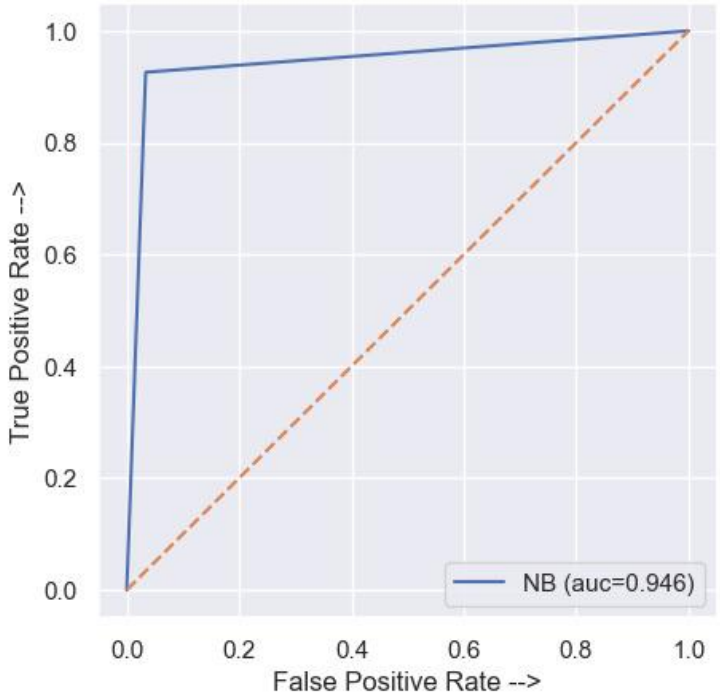


Figure 40. ROC curve metric of NB classifier.

5.4.4. Comparison of the suggested method with existing works

Table 9 presents a comparison of the classification accuracy achieved by various feature selection methods used in conjunction with different classifiers for the task of breast cancer classification. The table includes results from prior studies, as well as the performance of the proposed approach, which utilizes Correlation + Modified Grey Wolf Optimizer (CMGWO)**.**

The comparison in Table 9 shows that the proposed CMGWO method with Random Forest achieves the highest classification accuracy (99.12%**),** surpassing all other existing feature selection methods and classifiers. Although the performance of the proposed method with SVM and Naïve Bayes is slightly lower compared to other feature selection techniques like GOA, the overall results highlight the robust nature of the proposed CMGWO method, especially when used with Random Forest. This suggests that CMGWO is an effective feature selection technique that can significantly enhance classification performance, particularly with more complex classifiers like Random Forest.

Table 9. Evaluation of the proposed method by comparing results with existing feature selection methods.

Authors	Feature Selection Technique	Classifier	Accuracy (%)
Darzi et al. [9]	Genetic Algorithm	Case-based reasoning (CBR)	97.37
A. Rahmani et al. [10]	Feature Selection with GOA	SVM	98.83
S. Kumar and M. Singh [8]	Feature Selection with Enhanced GWO-SVM	SVM	98.24
Ibrahim et Nazir. [48]	Correlation + Principal Component Analysis	Ensemble machine learning	98.24
Proposed	Proposed-CMGWO	SVM	97.36
		NB	96.5
		RF	**99.12**

6. Conclusion

Integrating machine learning with metaheuristic algorithms has proven to be an effective strategy for solving a wide range of complex problems across various domains, particularly in image processing. This hybrid approach leverages the predictive power of machine learning models and the global optimization capabilities of metaheuristics to improve accuracy, robustness, and adaptability in tasks such as image segmentation, classification, and feature selection.

In this chapter, we presented an effective approach for breast cancer classification by integrating a feature selection method based on the Modified Grey Wolf Optimization (MGWO) algorithm with various machine learning classifiers. The MGWO was employed to identify the most relevant features, reducing data dimensionality while preserving critical diagnostic information. Several classifiers were tested, with random forest showing particularly strong performance when combined with MGWO. On the other hand, we also proposed a novel hybrid approach that combines correlation-based analysis with the MGWO for feature selection in breast cancer classification. In this method, correlation is first used to eliminate redundant or highly correlated features that may negatively impact the performance of the classifier. Subsequently, MGWO is employed to optimize the selection of the most relevant subset of features, enhancing the discriminative power of the model. This combination leverages the simplicity and effectiveness of correlation filtering with the robust exploration and exploitation capabilities of MGWO, resulting in improved classification accuracy and reduced computational complexity. The experimental results demonstrated that the proposed hybrid approaches improves classification accuracy and overall diagnostic reliability. These findings confirm the potential of intelligent feature selection in enhancing machine learning-based medical diagnosis systems.

However, one of the major challenges of this integration is the high computational cost, especially when dealing with large-scale image datasets or high-dimensional feature spaces. To address this issue, we explore the use of parallel metaheuristic algorithms deployed on distributed systems. By distributing the computation across multiple processing nodes, we aim to significantly reduce execution time while maintaining or even improving solution quality. This parallelization strategy allows for more efficient exploration of the search space, making it feasible to apply hybrid Metaheuristic-ML approaches in real-time or large-scale image processing applications.

Chapter 5

Parallel metaheuristic for image segmentation

1. Introduction

Image segmentation methods often struggle with the computational demands of high-resolution data and the complexity of extracting meaningful regions. To overcome these challenges, this section proposes two parallel metaheuristic-based segmentation approaches designed to improve both processing speed and segmentation quality. The first approach involves a Parallel Whale Optimization Algorithm (WOA) combined with K-Means clustering, implemented using Python's multiprocessing module. By distributing the optimization and clustering tasks across multiple CPU cores, this method accelerates the segmentation process while effectively exploring the solution space to enhance accuracy. The second approach utilizes a Parallel GWO integrated with Fuzzy C-Means (FCM) for MRI brain image segmentation, with key computations such as membership updates and centroid adjustments offloaded to the GPU. This GPU-based strategy leverages the parallel processing power of modern hardware to significantly reduce computation time and enable more iterations, resulting in improved convergence and segmentation quality. Overall, both parallel implementations demonstrate that harnessing multi-core CPUs and GPUs can minimize execution time and boost the effectiveness of metaheuristic-driven image segmentation techniques.

To accelerate the segmentation process, parallel computing techniques are employed using multiprocessing and GPU acceleration. Multiprocessing enables concurrent execution of segmentation tasks across multiple CPU cores, reducing processing time for large-scale images. Meanwhile, GPU-based parallelism significantly speeds up iterative optimization and clustering processes, making it feasible to handle terabyte-scale datasets efficiently. Frameworks such as pytorch, tensorFlow, and CUDA facilitate GPU acceleration, allowing deep learning models and optimization algorithms to execute in parallel. By leveraging machine learning, metaheuristic optimization, and parallel computing, this research aims to advance high-performance image segmentation for applications requiring large-scale data processing, such as medical imaging, remote sensing, and real-time object detection.

2. Parallel Whale Optimization Algorithm-Kmeans for image segmentation using Multiprocessing

To improve the quality and efficiency of image segmentation, the present work targets two core objectives including optimizing cluster centroids and accelerating the segmentation process. First, the Whale Optimization Algorithm (WOA) [14] is employed to determine the optimal centroids for each cluster, providing a strong initialization for the subsequent

segmentation step. These centroids are then utilized by the K-means algorithm, which performs the actual image segmentation based on the optimized positions identified by WOA.

To address the second objective which is computational acceleration, we introduce a parallelized implementation of this hybrid strategy using the multiprocessing framework. By distributing the WOA optimization across multiple processing units, we significantly reduce execution time while maintaining segmentation accuracy.

This section is structured as follows: we begin with a comprehensive overview of the WOA algorithm and its mathematical formulation, followed by a description of the K-means clustering technique. Next, we detail the integration of these methods into the hybridWOA-K-means framework, and finally, we elaborate on the parallelization strategy employed to enhance performance on multi-core systems.

2.1. Whale Optimization Algorithm (WOA)

To address numerical optimization problems, the Whale Optimization Algorithm (WOA) was introduced by Mirjalili and Lewis in 2016 [14]. This algorithm is inspired by the social behavior and bubble-net hunting strategy of humpback whales, humpback whales consider as one of the largest mammal species on Earth. The distinctive hunting technique used by humpback whales, known as bubble-net feeding, serves as the foundation for WOA's design. Algorithm 5 presents the pseudo-code for the whale optimization algorithm. The algorithm is based on three core phases: encircling prey, the bubble-net attacking method, and searching for prey. The mathematical models corresponding to these strategies are detailed in the following subsections.

2.1.1. Encircling Prey. At first, whales detect the position of prey and encircle it. this process is simulated by WOA. The global optimal solution is treated as the prey, while the other candidate solutions modify their places in reference to the global optimal solution, the location of the candidate solution, $\vec{X}\ (t+1)$ is calculated by the two following equations:

$$\vec{D} = |\vec{C}\ .\overrightarrow{X^*}(t) - \vec{X}(t)| \tag{11}$$

$$\vec{X}(t+1) = \overrightarrow{X^*}(t) - \vec{A}\ .\vec{D} \tag{12}$$

where $\overrightarrow{X^*}(t)$ is the global optimal solution (the best position recorded), $\vec{X}(t)$ denotes the position of candidate solution in the current generation (t) (denotes the best position recorded), t refers to the number of current iterations, and $\vec{A}$ and $\vec{D}$ are coefficient vectors, which are calculated as follows:

$$\vec{A} = 2\vec{a}\ .\vec{r} - \vec{a} \tag{13}$$

$$\vec{C} = 2\vec{r} \tag{14}$$

where $\vec{a}$ decreases linearly from 2 to 0 over iterations and $\vec{r}$ is randomly generated vector range in [0, 1]. the global optimal solution gained in the current generation can be used to

adjust the position of the candidate solution. The position of candidate solution could be updated via altering the values of $\vec{A}$ and $\vec{C}$.

2.1.2. Bubble-Net Attacking Method. Two methods are used in this step and they are designed as follows :

- **Shrinking Encircling Mechanism**: Equations (12) and (13) define the mathematical model of the shrinking encircling process. The value of $\vec{A}$ depends on the change of $\vec{a}$. In other words, by assigning a random number to $\vec{A}$ in [−1, 1], the new position of the candidate solution will be found between the current solution and global optimal solution.
- **Spiral Updating Position:** this technique calculates the new position of the candidate solution. The distance between the present candidate solution and global optimal solution is first calculated by Equation (15). Then the new position is generated by Equation (16).

$$\overrightarrow{D'} = |\overrightarrow{X^*}(t) - \vec{X}(t) \tag{15}$$

$$\vec{X}(t+1) = \overrightarrow{D'} . e^{bl} . \cos(2\pi l) + \overrightarrow{X^*}(t) \tag{16}$$

where b are constants that define the geometry of the logarithmic spiral and l is randomly generated range in [−1, 1], respectively.

there is a 50% probability of successful exploitation (attacking the prey) by using either a shrinking mechanism or a spiral model, and this is manipulated by a random number pro ∈ [0, 1]. The mathematical formula is :

$$\vec{X}(t+1) = \begin{cases} \overrightarrow{X^*}(t) - \vec{A} . \vec{D} & \text{pro} < 0.5 \\ \vec{X}(t+1) = \overrightarrow{D'} . e^{bl} . \cos(2\pi l) + \overrightarrow{X^*}(t) & \text{pro} > 0.5 \end{cases} \tag{17}$$

2.1.3. Search for Prey. Whales search at random based on their position. By using Equation (13), these processes can be utilized in WOA. $\vec{A}$ is designed as a random value less than 1 or higher than −1; this makes a chance for WOA to execute a random search. Thus, the purpose of this method is to enhance WOA's exploratory capabilities. When $\vec{A}$is greater than 1, it enables WOA to execute a wide search, the following formula is the model:

$$\vec{D} = |\vec{C} . \vec{X}_{rand}(t) - \vec{X}(t)| \tag{18}$$

$$\vec{X}(t+1) = \vec{X}_{rand}(t) - \vec{A} . \vec{D} \tag{19}$$

where $\vec{X}_{rand}$ denotes a random position of whale, $\vec{A}$ is less than 1, and the global optimal solution of the current iteration is selected for updating candidate solutions.

Algorithm 5 : Pseudo-code of the WOA algorithm

```
Initialize the whales population Xi(i=1,2, …,n)
Calculate the fitness of each solution
X*=The best search agent.
While i< maximum number of iteration do
        For every solution do
                Update a, A,C and p
                If (p<0.5) then
                        If (|A|<1) then
                                Update the postion of the current solution using Equation (12)
                          Else If (|A|>1) then
                                Random solution is generated
                                Update the position of the current solution using Equation (19)
                Else if (p>=0.5) then
                        Update the position of the current solution using Equation (16)
        End
        Check whether any solution exceeds the search space and adjust it
        Compute the fitness of every solution
        Update X* if there is a better solution
        t=t+1
End
Return X*
```

2.2. Hybrid Whale Optimization Algorithm -Kmeans

In this work, an enhanced image segmentation technique is introduced through a hybridization of the Whale Optimization Algorithm (WOA) with the K-means clustering method. This integrated approach leverages the global search capability of WOA, which is an evolutionary, nature-inspired metaheuristic, to identify optimal solution regions within the search space, while the K-means algorithm subsequently refines cluster centroids to achieve high quality segmentation. By combining these two strategies, the proposed method aims to enhance segmentation effectiveness across multiple evaluation metrics. The algorithm's structure is detailed in Algorithm 6. To steer the optimization process, the Sum of Squared Errors (SSE) is employed as the objective function, which focuses on minimizing intra-cluster variance to ensure pixel homogeneity within each cluster. The segmentation procedure unfolds in five systematically organized stages:

1. **Initialization**: Each whale is initialized with a randomly generated set of centroids, serving as candidate solutions for image segmentation.
2. **Fitness Evaluation**: The SSE is computed for each whale's current centroid configuration, providing a quantitative assessment of clustering effectiveness.

3. **WOA Global Search**: The whale population is updated using WOA's position-update strategies, including the encircling prey model and spiral movement, to explore the solution space broadly.
4. **K-means Local Refinement**: Following each global search phase, K-means is applied to refine the centroid positions of each whale, thereby enhancing local clustering precision.
5. **Termination**: The algorithm concludes either upon reaching the predefined maximum number of iterations or when the SSE exhibits negligible improvement across successive iterations.

2.3. Parallel Whale Optimization Algorithm -Kmeans strategy

In the present study, a Parallel Whale Optimization Algorithm–K-means (PWOA-Kmeans) framework was developed with the dual objectives of accelerating the image segmentation process and enhancing key performance metrics such as accuracy, Peak Signal-to-Noise Ratio (PSNR), Root Mean Square Error (RMSE), and Structural Similarity Index (SSIM). The core concept involves executing the Whale Optimization Algorithm (WOA) in a parallel computing environment to optimize the cluster centroids for the K-means algorithm more efficiently.

In this parallel design, each computational process is tasked with optimizing a single whale, where a whale is represented by a unique set of candidate centroids. Thus, for a population size of N whales, Nparallel processes are launched, each Computing Processing Unit core independently refining the centroid positions of one whale. For example, if the number of whales is set to eight, eight concurrent processes are executed, each dedicated to the optimization of one distinct whale. A schematic overview of this parallel structure is depicted in Figure 41, where whales (W) are mapped to their corresponding processes (P).

The procedure begins with image loading and preprocessing, which includes flattening and normalizing the input image to prepare it for segmentation. Subsequently, a pool of worker processes is instantiated using the multiprocessing library, typically matching the number of available CPU cores to maximize computational throughput.

Next, the parallelized WOA-Kmeans algorithm is executed. The WOA global position update and the K-means local refinement steps are performed concurrently for each whale across the distributed processes. Upon completion of all processes, the solution yielding the lowest Sum of Squared Errors (SSE) is identified as the optimal whale. Finally, a refined K-means algorithm is applied using the selected optimized centroids, producing the segmented output image.

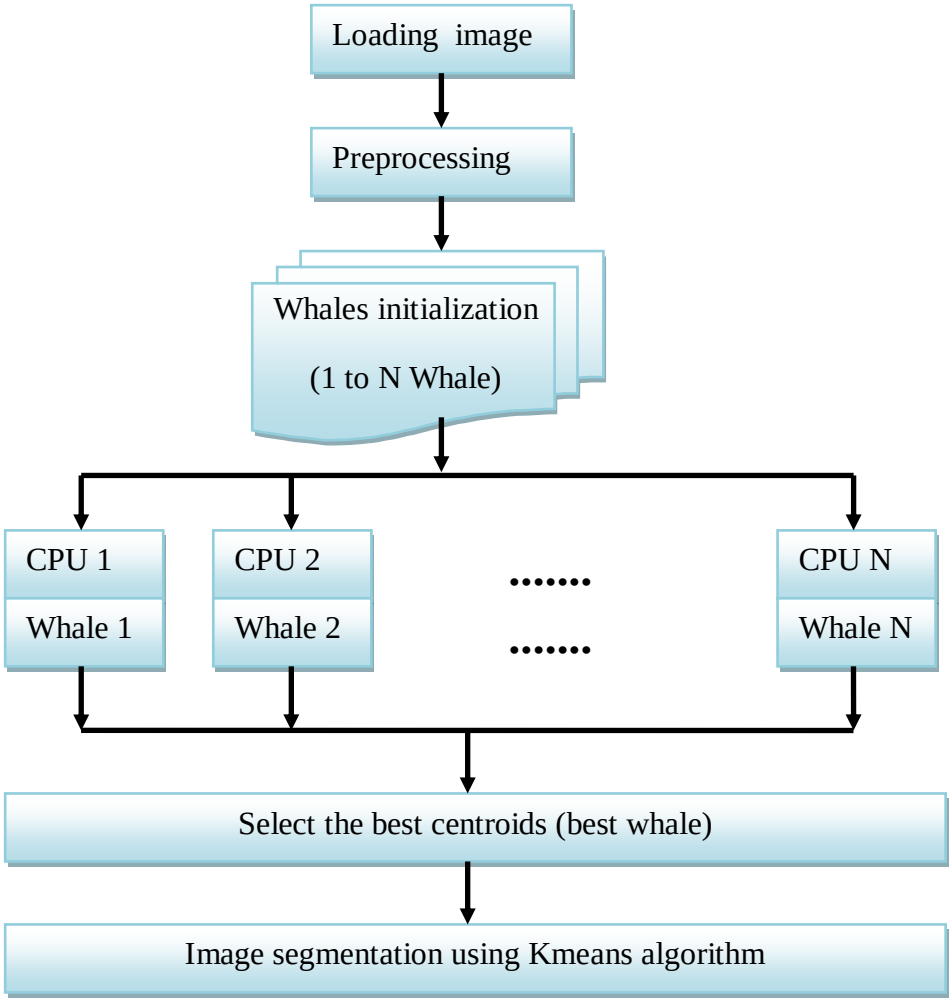


Figure 41. Flowchart of parallel WOA-Kmeans method.

2.4. Experimental results

To validate the robustness and efficiency of the proposed Parallel WOA–K-means (PWOA-Kmeans) approach, a comprehensive comparative analysis was conducted against its sequential counterpart under identical parameter settings. Both implementations were executed using 30 whales and 30 iterations. The parallel version leveraged Python's multiprocessing module and was deployed on an Intel(R) Core(TM) i7-1065G7 processor featuring 8 cores, each operating at 1.30 GHz.

The results underscore the advantages of parallelization, as the segmentation process significantly benefited from the integration of WOA with K-means in a parallel execution environment. To further demonstrate the method's generalizability and robustness, it was tested across a diverse set of grayscale images. These included three widely used benchmark images from the Berkeley Segmentation Dataset [255], including, Cameraman, Lena, and Baboon, as we see in figure 42 where images represented as Image 1, Image 2, and Image 3 respectively, as well as a Leukemia cell image (Image 4 from figure 42) obtained from the ALL-IDB medical imaging database[256].

Figure 42, Figure 43 and Figure 44 provide a visual representation of the segmentation results. Figure 42 displays the original input images, figure 43 presents the segmented outputs

using the sequential WOA–K-means approach, and figure 44 illustrates the segmented results obtained from the parallel PWOA–Kmeans method.

Initially, segmentation was performed without parallelization to establish a baseline. Subsequently, the same images were segmented using the multiprocessing-based parallel implementation. The final stage involved a comparative evaluation between the two models, focusing on various performance indicators such as segmentation accuracy, PSNR, RMSE, and SSIM to assess improvements in both computational efficiency and segmentation quality.

Image 1 Image 2 Image 3 Image 4

Figure 42.The original input images.

Image 1 Image 2 Image 3 Image 4

Figure 43.The segmented outputs using the sequential WOA–Kmeans approach.

Image 1 Image 2 Image 3 Image 4

Figure 44.The segmented results obtained from the parallel WOA–Kmeans method.

2.4.1. Comparing the time between sequential and parallel approachfor different images

As illustrated in Table 10, utilizing Parallel WOA to optimize the K-means clustering algorithm by identifying the optimal centroids for each cluster significantly reduces execution time across various test images. The results clearly demonstrate that the Parallel WOA-K-means consistently outperforms other approaches in terms of computational speed, delivering the fastest segmentation times for all tested images. In contrast, the sequential WOA-K-means exhibited the longest execution times, making it the least efficient method among those evaluated. These findings validate the effectiveness of parallelization in enhancing both performance and scalability of the hybrid segmentation framework.

Table 10. Comparing the computation time between sequential and parallel algorithm.

Images	Time (second)	
	Sequentiel approach	Parallel approach
Image 1	186,11	16.82
Image 2	131	16.13
Image 3	156	17
Image 4	120	11.5

2.4.2. Comparing several metric between sequential and parallel approach for different images

In this experiment, the effectiveness of the proposed approach is evaluated with a focus on its ability to determine optimal cluster centroids for improved image segmentation quality. As presented in Table 11, a comparative analysis between the Parallel WOA-Kmeansand itssequential counterpart is conducted using key performance metrics: Accuracy**,** RMSE**,** PSNR**,** and SSIM. The results clearly demonstrate that the Parallel WOA-Kmeans consistently outperforms the sequential version, delivering superior segmentation accuracy**,** lower reconstruction error**,** higher image fidelity, and enhanced structural preservation. This highlights the advantage of incorporating parallel optimization into the segmentation pipeline for both performance and quality enhancement.

Table 11. Comparing the performance metrics between parallel and sequential model.

	Parallel approach				Sequential approach			
	Accuracy	RMSE	PSNR	SSIM	Accuracy	RMSE	PSNR	SSIM
Image 1	98,37	7,52	30,60	0,86	98,52	7,53	30,59	0,85
Image 2	94,14	7,67	30,43	0,84	91,19	8,28	29,76	0,81
Image 3	96,97	8,22	29,82	0,88	95,09	8,19	29,85	0,86
Image 4	100	3,92	36,25	0,91	100	3,94	36,21	0,90

3. Parallelized Grey Wolf Optimizer-Based Fuzzy C-Means for MRI Segmentation on GPU

In the field of medical image analysis, accurate and efficient brain MRI segmentation plays a critical role in the diagnosis and treatment of neurological disorders. Traditional clustering-based segmentation methods often struggle with the complexity and noise inherent in MRI data. To address these challenges, we propose a parallel implementation of a hybrid Grey Wolf Optimizer–Fuzzy C-Means (GWO-FCM) algorithm for brain MRI segmentation using GPU acceleration. The Grey Wolf Optimizer (GWO), inspired by the leadership hierarchy and hunting behavior of grey wolves, is employed to optimize the initial cluster centers and improve the convergence of the FCM algorithm. By integrating GWO with FCM, the method achieves enhanced segmentation accuracy and robustness against intensity inhomogeneity and noise. Furthermore, to overcome the high computational cost typically associated with metaheuristic-based clustering, the proposed approach leverages the parallel processing capabilities of Graphics Processing Units (GPUs). GPU-based parallelization is used to accelerate both the GWO optimization process and the iterative updates of the FCM membership matrix and cluster centers. This parallel GWO-FCM framework significantly reduces execution time while maintaining high segmentation quality, making it well-suited for large-scale medical image analysis and real-time clinical applications.

3.1. Parallel MRI image segmentation using GPU

In medical literature, magnetic resonance imaging (MRI) is recognized as the most widely used modality for brain imaging, followed by computed tomography (CT), positron emission tomography (PET), and ultrasound [257, 258]. MRI is particularly favored due to its ability to provide detailed anatomical visualization of the entire brain, including the spinal cord and vascular structures, thanks to its superior contrast capabilities [259]. Unlike CT, MRI is non-ionizing, where MRI uses strong magnetic fields and radio waves to create detailed images of the body's internal structures. These radio waves do not carry enough energy to ionize atoms or molecules, making MRI a safer imaging option, especially for repeated use. as it does not expose patients to harmful radiation [260]. Among the most commonly utilized MRI sequences are T1-weighted, T2-weighted, and FLuid Attenuated Inversion Recovery (FLAIR) [261, 262].

Despite its advantages, MRI comes with certain challenges, where it requires expensive, high-performance machinery, and data acquisition and image reconstruction are often time-intensive, making efficient and accurate processing techniques essential for timely clinical decision-making [263–265]. Moreover, MRI brain images often suffer from artifacts, rendering segmentation a particularly complex and critical task in medical image analysis. Medical image segmentation [266] has garnered significant attention in recent years and remains a central focus in the field of biomedical imaging research [267]. It plays a pivotal role in delineating anatomical structures such as tumors, bones, organs, and critical brain regions. A wide array of algorithms has been developed to support this task, including thresholding, clustering, level set methods, active contours, and region-growing techniques [268], each offering unique advantages for specific imaging scenarios.

Even with these advancements, brain MRI segmentation continues to pose substantial challenges, largely due to the presence of imaging artifacts, intensity inhomogeneities, and anatomical variability. These complexities make it difficult to identify a universal segmentation strategy capable of consistently delivering optimal results across diverse datasets. Consequently, there is no one-size-fits-all solution that can comprehensively address the computational demands of brain image segmentation. To overcome these limitations, GPU-accelerated segmentation methods have emerged as a powerful alternative. These approaches aim to fulfill three main objectives: (1) enabling the comparative analysis of multiple segmentation algorithms, (2) facilitating the rapid and automated segmentation of large-scale medical image datasets, and (3) providing interactive visualization and segmentation tools that operate in real time. Leveraging the massively parallel architecture of modern GPUs, which can house hundreds of cores and support thousands of concurrent threads, technologies like CUDA-based parallel programming significantly enhance performance and scalability. As a result, GPU computing has become an indispensable tool for solving computationally intensive problems in medical imaging, particularly in the domain of segmentation.

In this section, we introduce a GPU-accelerated Parallel Grey Wolf Optimization-based Fuzzy C-Means (P-GWO-FCM) clustering framework tailored for efficient and accurate MRI image segmentation. The proposed method synergistically combines the exploratory strength of Grey Wolf Optimization (GWO) with the clustering precision of Fuzzy C-Means (FCM) to overcome the limitations associated with poor centroid initialization, a common drawback in traditional FCM. To further elevate segmentation quality, we incorporate Fuzzy Entropy as a fitness function, providing a more robust measure of uncertainty inherent in medical imaging data. This not only promotes the formation of well-defined and compact clusters but also enhances resilience to noise and intensity inhomogeneity commonly observed in MRI scans. On the other hand, the iterative nature of both GWO and FCM poses computational challenges, especially when dealing with large-scale, high-resolution images. To address this, the algorithm is parallelized using GPU architecture, enabling concurrent execution of key operations such as GWO position updates, fitness evaluations, and FCM membership calculations. This parallel strategy dramatically reduces computational time, accelerates convergence, and facilitates the practical deployment of the method in real-time or large-scale medical image analysis scenarios.

3.2. Fuzzy Entropy Clustering (FEC)

Fuzzy Entropy Clustering (FEC) is an advanced clustering technique that incorporates entropy-based regularization into fuzzy clustering frameworks to more effectively manage uncertainty and address the challenges of overlapping clusters [269, 270]. Unlike conventional fuzzy clustering approaches, FEC leverages an entropy measure to quantify the ambiguity in pixel-to-cluster assignments, thereby enhancing the algorithm’s sensitivity to vague boundaries and noise inherent in complex image data. At the core of FEC lies the principle of entropy minimization, where the fuzziness of the membership matrix is systematically reduced throughout the clustering process. This is achieved by iteratively computing the degree of membership for each data point relative to the cluster centroids,

while simultaneously minimizing the entropy to encourage more definitive assignments. The algorithm continues to refine the cluster centers and membership values until it converges to an optimal configuration. By embedding entropy as a guiding metric, FEC promotes clearer segmentation boundaries, robustly handles uncertainty, and yields well-separated, compact clusters. The mathematical formulation for fuzzy entropy, which plays a pivotal role in the optimization objective, is provided in Equation (20).

$$E = -\sum_{i=1}^{N}\sum_{j=1}^{C} u_{ij}\log(u_{ij}) \quad (20)$$

Where N represent the number of pixels, C the number of clusters and u_{ij} is the membership value of pixel i to cluster j.

3.3. Fuzzy-C Mean Grey Wolf Optimizer algorithm

We propose a hybrid segmentation framework that synergistically combines Grey Wolf Optimization (GWO) with Fuzzy Entropy (FE) as the objective function to refine the performance of the Fuzzy C-Means (FCM) clustering algorithm. By exploiting GWO's robust global search capabilities, this integration aims to optimize the selection of cluster centers, while the incorporation of fuzzy entropy significantly improves the algorithm's ability to handle uncertainty and overlapping regions which is a critical challenge in medical image segmentation.

Within this hybrid model, the GWO algorithm guides the optimization process using its biologically inspired mechanisms: encircling prey, hunting strategies, and the final attack phase, which together maintain a dynamic balance between global exploration and local exploitation. These iterative position updates ensure convergence toward more discriminative and stable cluster configurations. As a result, the enhanced FCM model benefits from sharper segmentation boundaries, increased resilience to noise, and improved accuracy. The conceptual structure of this hybrid GWO-FE-FCM method is visually depicted in Figure 45.

The segmentation process commences with the initialization of a wolf population, where each wolf encodes a candidate solution namely a potential set of cluster centroids. During the fitness evaluation phase, each solution is assessed using Fuzzy Entropy Clustering (FEC), which quantifies the uncertainty associated with cluster memberships. The objective is to minimize the fuzzy entropy, thereby promoting crisp cluster boundaries and improving the interpretability of segmentation results. Following fitness assessment, Grey Wolf Optimization (GWO) drives the position update phase, where the wolves' positions (i.e., centroids) are iteratively adjusted based on the hierarchy of the alpha, beta, and delta wolves. This biologically inspired mechanism guides the swarm toward promising regions of the search space through controlled exploration and exploitation. Each iteration refines the centroid configuration with the dual goal of minimizing entropy and improving segmentation fidelity. These two steps, the fitness computation and position adjustment are cyclically repeated until a termination condition is met, either upon reaching the predefined number of iterations or when convergence is achieved (i.e., negligible changes in fitness values across iterations). Once optimization concludes, the final refined centroids are fed into the Fuzzy C-Means (FCM) algorithm to perform the actual segmentation. Pixels are assigned to clusters

based on their fuzzy membership degrees, yielding a segmented image that delineates distinct regions according to intensity variations. This entire procedure is summarized in Algorithm 7.

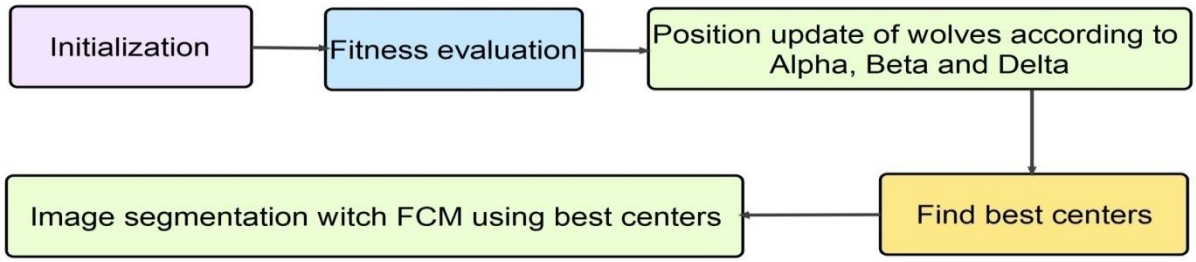


Figure 45. Diagram of hybrid GWO-FCM-FE.

Algorithm 7 : Pseudo-code of Hybrid GWO-FCM-FEC.

Input:
Grayscale image I of size M× N; Number of clusters C; Maximum number of iterations MaxIter

Step1:Initialization

- Randomly initialize the positions of N wolves (set of cluster) in the image intensity range.

Step 2: Iterative Optimization

For each iteration from t=1to MaxIter:

1. Evaluate Fitness:
   - For each wolf (cluster center set):
     - Compute fuzzy memberships μ_{ij} for each pixel based on the distance to the cluster centers.
     - Calculate the fuzzy entropy *E* using Equation 20.
     - Assign E as the fitness for the current wolf.
2. Update Leaders:
   - Identify α, β, δ: the three best wolves with the lowest entropy values.
3. Update Wolf Positions:
   - For each wolf :
     - Calculate the distance D to alpha wolf, Beta and delta wolf using Equation 11.
     - Update position using Equation 12.
     - Clip the updated position to stay within the search space(valid intensity range).

Step3:Segmentation

- After the final iteration, use the best wolf () cluster centers as best initial centroids for FCM algorithm.

3.4. Parallel GWO and Fuzzy C-Mean using Graphic Processing Unit

The proposed methodology integrates Parallel Grey Wolf Optimization (PGWO) and Parallel Fuzzy C-Means (PFCM) in a two-stage sequential framework to effectively optimize cluster centroids for MRI image segmentation, as illustrated in Figure 46. In the first phase, PGWO is employed to perform a global search for optimal centroids, using Fuzzy Entropy as the fitness function. Leveraging GPU acceleration, the algorithm parallelizes key components such as wolf position updates and fitness evaluations, enabling the simultaneous assessment of multiple candidate solutions across the search space.

Upon identifying the most promising set of centroids, the second phase executes the Parallel FCM algorithm using also GPU-accelerated. This stage benefits from fine-grained parallelism where membership matrix computations and cluster center updates are performed concurrently. Specifically, each pixel's membership degree is computed in parallel, while centroid updates utilize parallel reduction operations, substantially minimizing computational overhead and accelerating convergence.

The sequential deployment of PGWO and PFCM capitalizes on the strengths of both algorithms: PGWO provides a robust global search mechanism to initialize the clustering process effectively, while PFCM delivers precise local refinement to enhance segmentation quality. This hybrid approach strategically balances exploration and exploitation, resulting in improved segmentation performance and computational efficiency when compared to traditional, non-parallel clustering methods.

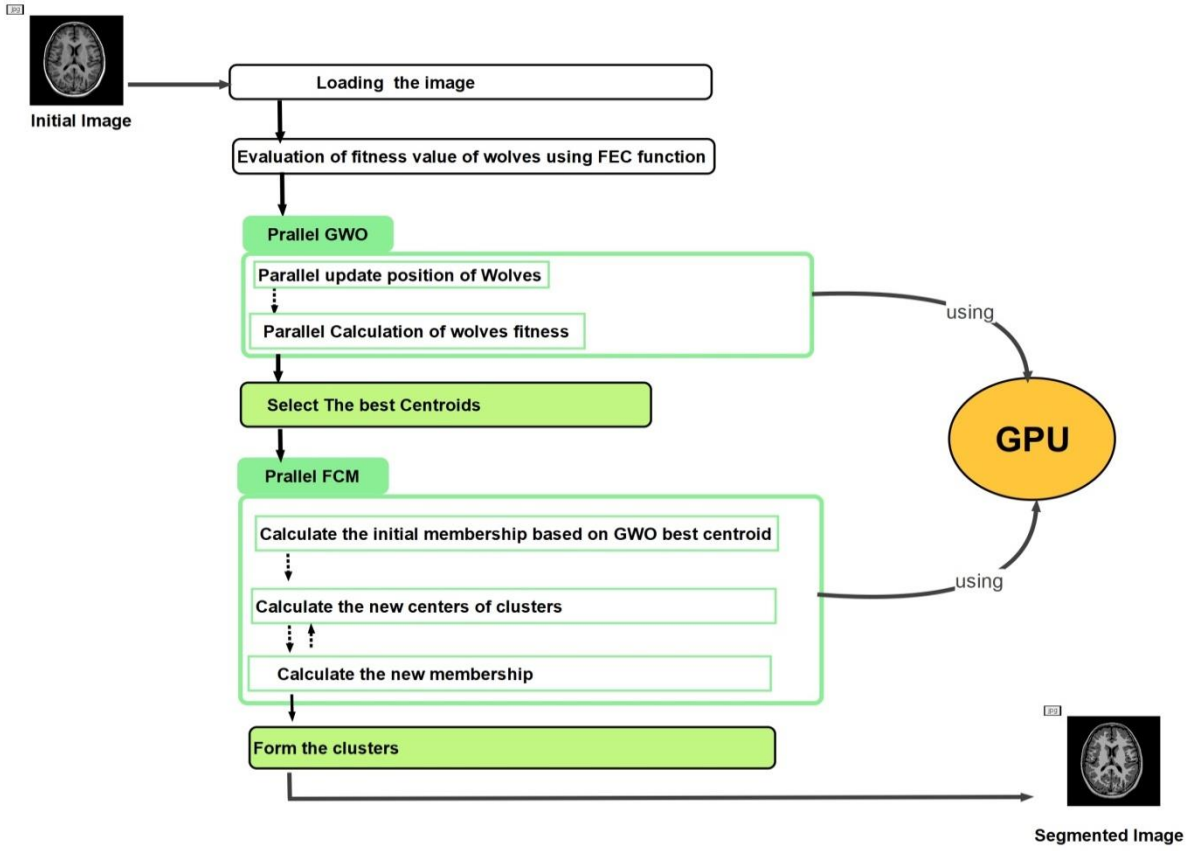


Figure 46. The process of proposed Parallel GWO and Parallel FCM.

Figure 47 presents a schematic diagram of the proposed hybrid approach, integrating Parallel PGWO and PFCM for MRI image segmentation. The framework begins with the execution of PGWO, guided by Fuzzy Entropy (FE) as the fitness function, to systematically explore the search space and determine optimal cluster centroids. This stage follows a well-defined sequence of operations designed to maximize search efficiency through GPU-parallelized computation.

Following the identification of candidate centroids by PGWO, the Parallel FCM algorithm is employed to perform refined segmentation. This phase further enhances the clustering precision by leveraging parallel computation for key operations such as membership updates and centroid recalculation.The comprehensive implementation details of both PGWO and PFCM are outlined in Algorithm 8 and Algorithm 9, respectively, offering a step-by-step procedural breakdown of the parallel mechanisms employed in each stage of the segmentation pipeline.

Algorithm 8 : Pseudo-code of PGWO.

For each iteration from t=1 to MaxIter:

Step 1: Parallel fuzzy membership calculation steps:

- Assign each pixel (or block of pixels) to a thread.
- Compute distance between the pixel intensity and all cluster centers.
- Update fuzzy membership μ_{ij} for the pixel *i* in each cluster *j* using :

$$\mu_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{d_{ij}}{d_{ik}}\right)^{2/(m-1)}}$$

Where *m* is the fuzziness factor.

- store μ_{ij} in global memory.

Step 2: Parallel fitness evaluation steps:

- Assign each wolf (set of centers) to a thread.
- Compute fuzzy entropy E using Equation (20).
- Store E in global memory for each wolf.

Step 3: Parallel GWO position update steps:

- Assign each wolf to a thread.
- Update positions using GWO algorithm Equations (11) and (12).
- Clip values to ensure valid intensity range.

End-for

Select Best wolf (Best Centers).

Algorithm 9 : Pseudo-code of PFCM.

Initialization using best centroid selected by P-GWO

For each iteration from t=1 to MaxIter:

Step1:

Initialize Membership Matrix in Parallel :

For Each pixel x_i is assigned an initial membership value for each cluster k.

Step2:

Compute Cluster Centers in Parallel:

For each cluster k:

- Compute weighted sum of all pixels based on membership.

Step3:

Compute New Membership Matrix in Parallel:

For each pixel x_i and cluster k:

- Compute distances d_{ki}
- Compute new membership values using the fuzzy rule.

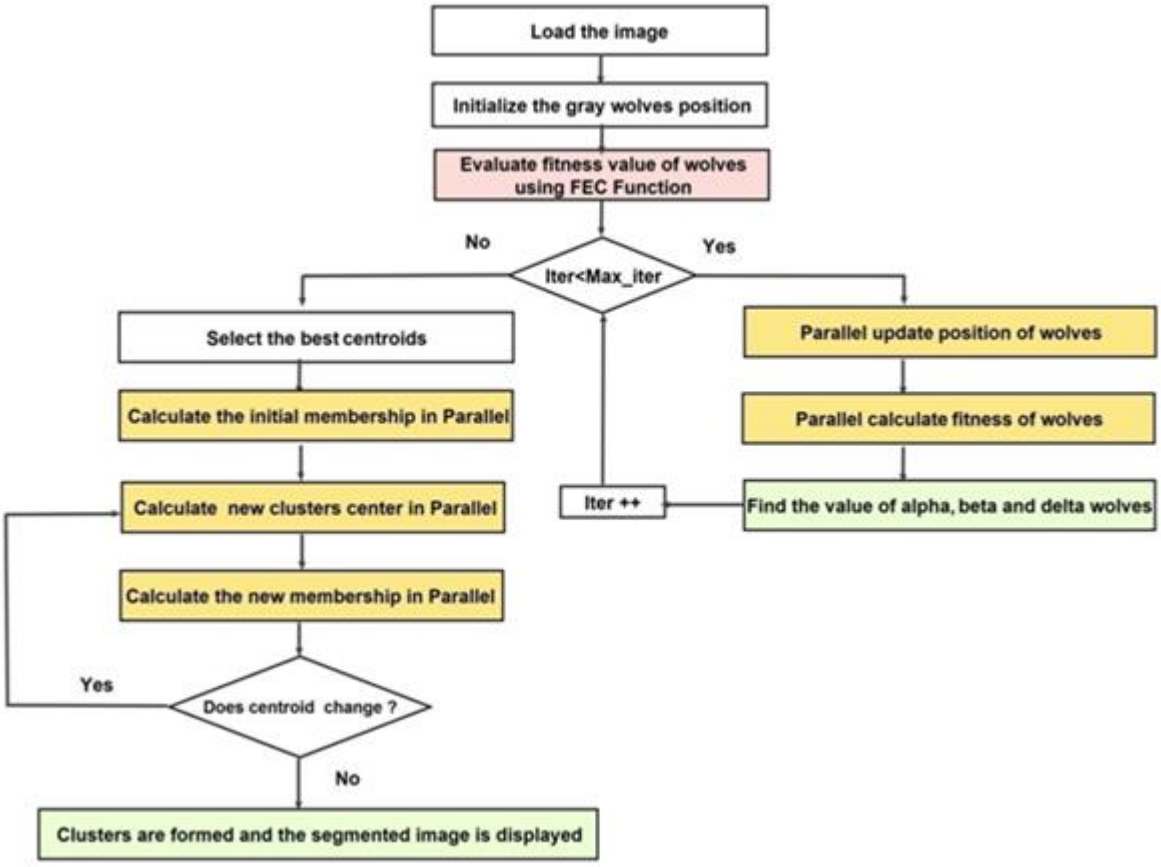


Figure 47. Diagram of the proposed P-GWO-FCM.

3.5. Experimental results

The experimental evaluation of the proposed Parallel Grey Wolf Optimization with Fuzzy C-Means (P-GWO-FCM) approach was conducted using three distinct datasets: a simulated brain tumor dataset [271], a real-world clinical MRI dataset sourced from Kaggle [272], and the dataset from the Radiological Society of NorthAmerica (RSNA), provided as part of a recent Kaggle competition [273]. These datasets were chosen to comprehensively assess the segmentation capability of the method across both synthetic and clinical imaging scenarios.

To quantitatively evaluate segmentation performance, several well-established metrics were employed: the Jaccard Index, Davies-Bouldin Index (DBI), Partition Coefficient Index (PCI), and Partition Entropy Index (PEI). These metrics collectively provide insight into segmentation accuracy, intra-cluster compactness, inter-cluster separation, and membership fuzziness. The proposed P-GWO-FCM technique was benchmarked against traditional Fuzzy C-Means (FCM), Sequential GWO-FCM, and other relevant state-of-the-art methods. This comparative analysis highlights the improvements achieved through parallelization and hybridization. The mathematical definitions and formulations of each evaluation metric are detailed below.

1. **Jaccard Index:** The Jaccard Index (JI) measures the similarity between two sets, commonly used for evaluating segmentation accuracy:

$$\mathrm{JI} = \frac{|A \cap B|}{|A \cup B|} \tag{21}$$

Where A is the ground truth and B is the segmented region.

2. **Partition Coefficient Index (PCI)**: PCI evaluates the compactness of clusters in fuzzy clustering, defined as:

$$\mathrm{PCI} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} u_{ij}^2 \tag{22}$$

where u_{ij} is the membership value of pixel iin cluster j, and N is the total number of pixels.

3. **Davies-Bouldin Index (DBI):** DBI assesses cluster compactness and separation, given by:

$$DBI = \frac{1}{C}\sum_{i=1}^{C} max_{i \neq j} \frac{s_i + s_j}{d_{ij}} \tag{23}$$

Where s_i is the dispersion of cluster i , and d_{ij} is the distance between cluster centroids.

4. **Partition Entropy Index (PEI):** PEI measures the fuzziness of cluster memberships:

$$PEI = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} u_{ij} \log u_{ij} \tag{24}$$

Whereu_{ij} is the membership value of pixel *i* in cluster *j*.

5. **Dice coefficient measure**: is a statistic measure used for comparing the similarity between two samples and ranges between 0 and 1:

$$Dice = \frac{2|A \cap B|}{|A| + |B|} \quad (25)$$

Where A and B are the segmented image and the ground truth image.

3.5.1. Evaluation on Simulated Brain Tumor Dataset

The effectiveness of the proposed P-GWO-FCM approach was rigorously evaluated using a simulated brain tumor dataset, with a focus on accurately segmenting key brain tissues including White Matter (WM), Gray Matter (GM), and Cerebrospinal Fluid (CSF). To quantitatively assess segmentation performance, the Jaccard Index was calculated for each tissue type, capturing the degree of spatial overlap between the segmented outputs and their corresponding ground truth regions. The evaluation was performed on a T1-weighted brain MRI scan with a resolution of 217 × 181 pixels. This setup was designed to test the robustness and precision of the segmentation under practical imaging conditions. Accurately delineating WM, GM, and CSF is of critical importance in clinical neuro imaging, particularly for diagnostic assessments in neurology and radiology.

Figure 48 offers a comprehensive visual comparison of the segmentation results across different methods. The original brain image is displayed in Figure 48*(a)*, while the manually annotated ground truth for WM, GM, and CSF is shown in Figure 48*(b)*. The segmentation outputs generated by traditional FCM, sequential GWO-FCM, and the proposed P-GWO-FCM method are illustrated in Figure 48(c), *48*(d), and *48*(e), respectively. This visual analysis underscores the performance enhancements introduced by incorporating GWO and parallel processing into the clustering framework.

Figure 48 effectively illustrates the superior performance of the P-GWO-FCM method in maintaining regional homogeneity, producing segmentation results that are both uniform and structurally coherent. Notably, the proposed approach demonstrates a remarkable ability to preserve fine anatomical details from the original MRI scans, an essential feature in medical image analysis, where subtle tissue variations can carry significant diagnostic implications. To quantitatively assess this performance, the Jaccard Similarity (JS) was computed for each of the evaluated methods: conventional FCM, Sequential GWO-FCM, and the proposed Parallel GWO-FCM. As summarized in Table 12, the average JS values obtained using the P-GWO-FCM method were consistently higher across all tissue classes (WM, GM, CSF), underscoring its enhanced segmentation accuracy.

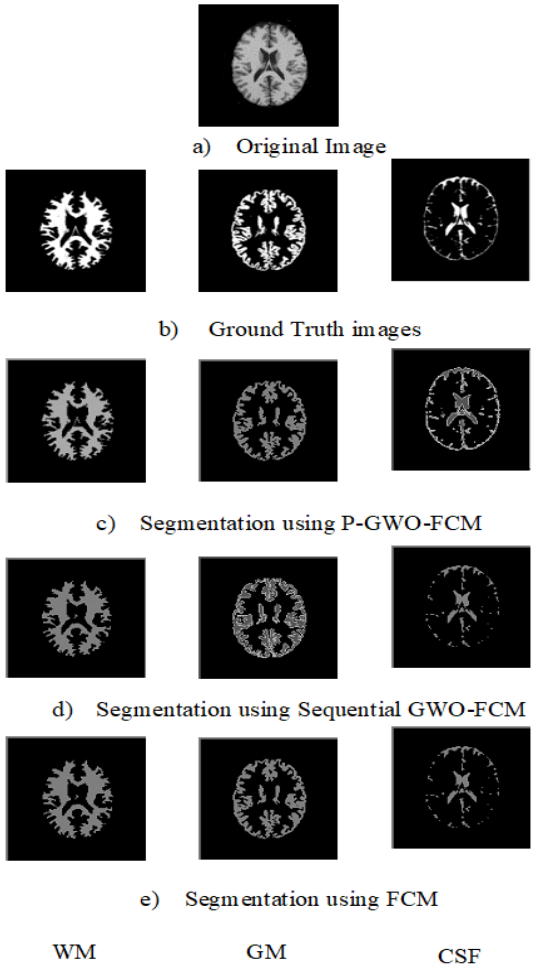


Figure 48. Segmentation results of WM, GM, CSF using FCM, sequential-GWO-FCM, and P-GWO-FCM.

Table 12 presents a comprehensive comparison of the segmentation outcomes based on the JS metric, which measures the degree of spatial overlap between the segmented outputs and the ground truth. A higher JS score directly correlates with improved accuracy, and the results clearly indicate that the proposed parallelized hybrid approach outperforms both the standalone FCM and its sequential hybrid variant.

Table 12. Jaccard similarity values for the three methods on simulated MR images.

Method	Jaccard Measure			Average
	WM	GW	CSF	
P-GWO-FCM	0,94	0,89	0,93	**0,92**
Sequential GWO-FCM	0,89	0,90	0,92	0,90
FCM	0,88	0,80	0,90	0,86

The P-GWO-FCM method clearly outperforms competing algorithms, attaining the highest Jaccard Similarity (JS) scores across all examined tissue classes: 0.94 for White Matter (WM), 0.89 for Gray Matter (GM), and 0.93 for Cerebrospinal Fluid (CSF). These individual results contribute to an impressive average JS value of 0.92, which exceeds the performance metrics of both traditional FCM and its sequential hybrid variant.

This superior performance highlights the efficacy of integrating Grey Wolf Optimization (GWO) with Fuzzy Entropy (FE) to enhance the FCM framework for brain MRI segmentation. The global search capabilities of GWO significantly mitigate the limitations of poor initialization and susceptibility to local optima, which are common challenges in conventional clustering approaches. Simultaneously, the incorporation of fuzzy entropy introduces greater robustness in uncertain or ambiguous regions, preserving fine-grained structural variations crucial for medical diagnosis.

Moreover, the parallel implementation of the algorithm delivers substantial computational speedups without compromising segmentation precision. This makes the proposed method particularly suitable for large-scale and real-time medical imaging applications, a benefit further demonstrated in the subsequent experimental results.

In addition to the internal comparisons, the proposed P-GWO-FCM method was benchmarked against established segmentation techniques, namely FCM-GENIUS [274] and Deep-JCR (Deep Joint Calibrationless Reconstruction) [275], both of which utilize the Dice Similarity Coefficient as the primary performance metric. As shown in Table 13, P-GWO-FCM consistently outperforms these state-of-the-art approaches, achieving superior Dice scores of 0.93 for White Matter (WM), 0.89 for Gray Matter (GM), and **0.95** for Cerebrospinal Fluid (CSF).

These results underscore the robustness and precision of the proposed hybrid method, which not only enhances segmentation accuracy but also maintains computational efficiency through parallel processing. The comparative advantage of P-GWO-FCM reaffirms the value of integrating metaheuristic global search and fuzzy entropy-based refinement into the FCM framework for high-resolution brain MRI segmentation tasks.

Table 13. Comparing the dice coefficient between the proposed method and the existing methods.

Methods	Dice coefficient		
	WM	GM	CSF
P-GWO-FCM	**0,93**	**0,89**	**0,95**
Sequential GWO-FCM	0,93	0,88	0,87
FCM	0,88	0,88	0,78
FCM-GENIUS [274]	0.73	0.76	0.19
Deep-JCR [275]	0.913	0.855	0.805

3.5.2. Evaluation on Clinical brain MRI Dataset

To further assess the algorithm's performance, experiments were conducted on a clinical MRI dataset [272] from Kaggle, which includes a variety of tumor patterns with different intensity distributions. The segmentation performance of the P-GWO-FCM method was compared to that of FCM and sequential GWO-FCM. The effectiveness of these three methods was evaluated using the DBI, PEI and PCI metrics, with the results presented in Table 14. Figure 49 show cases a selection of MRI images from the clinical dataset used in the experiments, displaying 10 images of varying sizes to demonstrate the robustness of the proposed approach. These images were segmented using all three algorithms. Figure 50 illustrates the segmentation results obtained for these 10 brain MRI images using the P-GWO-FCM method.

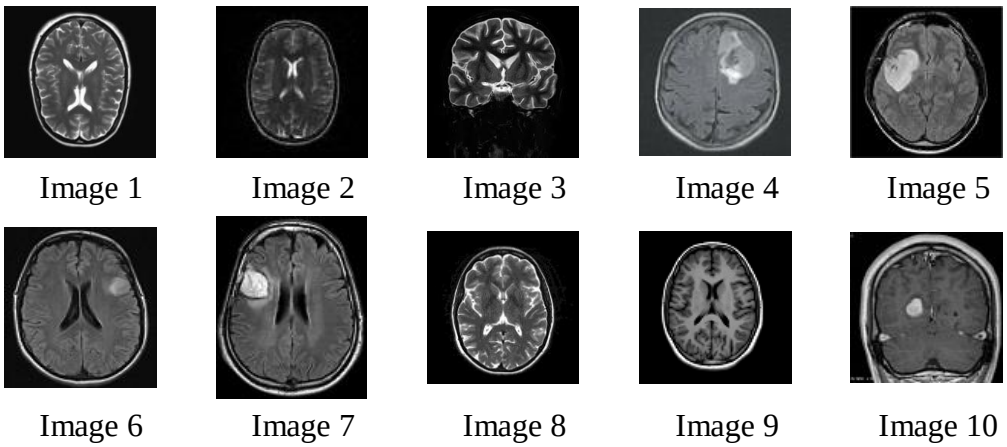


Figure 49. Samples of brain MR images from clinical dataset.

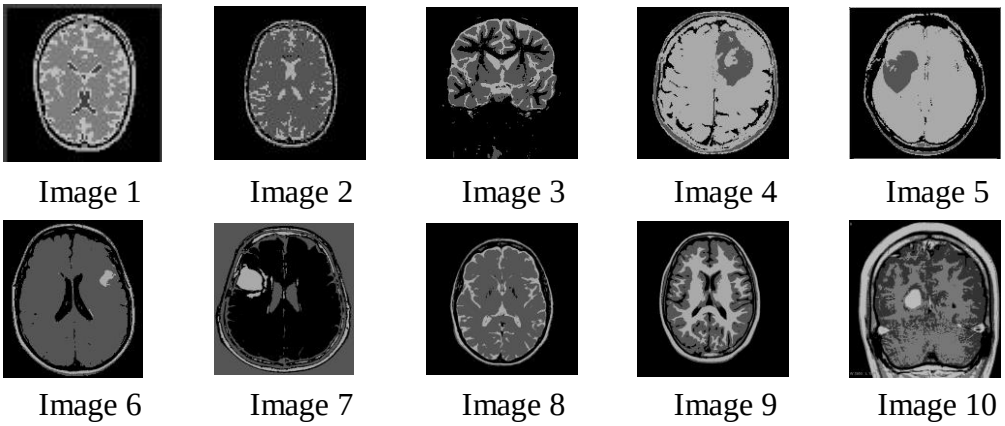


Figure 50.Segmentation results on the clinical brain MR Images with P-GWO-FCM.

A visual analysis of the segmented images reveals that the P-GWO-FCM method excels in clarity, detail preservation, and precise delineation of tissue boundaries. The results in Table 14 provide a comprehensive comparison of P-GWO-FCM, traditional FCM, and Sequential-GWO-FCM, demonstrating that P-GWO-FCM consistently outperforms the other methods across various evaluation metrics. These metrics assess clustering quality, particularly in

terms of cluster compactness, separation, and the clarity and reliability of cluster assignments. Regarding the DBI, which evaluates clustering quality by measuring compactness and separation, P-GWO-FCM achieved an average DBI value of 0.30, significantly lower than those of FCM and sequential GWO-FCM. This result indicates better cluster definition and separation, improving segmentation quality. The PEI, which quantifies uncertainty in membership assignments, further supports the robustness of our approach, as P-GWO-FCM attained an exceptionally low average PEI value of 0.25.This low PEI value reflects the minimal overlap between clusters, demonstrating the algorithm's ability to assign data points with higher confidence and precision. Additionally, the PCI measures clustering fuzziness and further demonstrates the superiority of P-GWO-FCM. The algorithm achieved an impressive average PCI value of 0.91, indicating clearer partitioning with reduced fuzziness. This high PCI value, consistent across all test images, confirms that cluster memberships are predominantly close to 0 or 1, leading to more definitive segmentation.

Table 14. Comparing of FCM, sequential GWO-FCM and P-GWO-FCM using Different metrics.

Images	FCM			Sequential-GWO-FCM			P-GWO-FCM		
	DBI	PEI	PCI	DBI	PEI	PCI	DBI	PEI	PCI
Image1	0.32	0.20	0.90	0.30	0.18	0 .88	0.30	0.18	0.92
Image2	0.35	0.22	0.87	0.33	0.20	0.89	0.24	0.17	0.93
Image3	0.29	0.31	0.89	0.19	0.31	0 .90	0.15	0.27	0.90
Image4	0. 39	0.31	0.86	0.42	0.31	0.86	0.43	0.30	0.91
Image5	0.25	0.28	0.91	0.25	0.30	0.93	0.20	0.28	0.92
Image6	0.35	0.26	0.88	0.33	0.26	0.90	0.31	0.26	0.90
Image7	0.36	0.24	0.90	0.36	0.24	0.91	0.30	0.19	0.91
Image8	0.40	0.30	0.87	0.37	0.30	0.87	0.37	0.35	0.92
Image9	0.39	0.32	0.87	0.40	0.32	0.89	0.35	0.24	0.89
Image10	0.42	0.30	0.86	0.39	0.33	0.87	0.36	0.33	0.90
Average	0.35	0.27	0.88	0.33	0.27	0.89	0.30	0.25	0.91

Table 15 presents a comparison between the sequential GWO-FCM and the P-GWO-FCM approach in terms of execution time. The results clearly show that P-GWO-FCM outperforms the sequential algorithm, achieving significantly faster execution. In contrast, with sequential GWO-FCM, larger images require more time for segmentation. However, the proposed parallel approach maintains a consistently low segmentation time regardless of image size, as illustrated in Figure 51.The execution time remains low even when using images with larger dimensions due to GPU acceleration, which efficiently processes parallel computations. The GPU's ability to handle multiple operations simultaneously distributes the workload across thousands of cores, significantly reducing processing time compared to the sequential approach.

Table 15. Comparing Time between sequential and P-GWO-FCM.

Images	Dimension x*y	Time (second)	
		Sequential-GWO-FCM	P-GWO-FCM
Image1	79*78	2,01	0,60
Image2	100*100	2,18	0,62
Image3	200*200	10,36	0,62
Image4	223*226	11,26	0,62
Image5	225*225	11,71	0,64
Image6	291*340	23,28	0,87
Image7	374*456	41,65	0,78
Image8	442*442	48,02	0,84
Image9	728*725	155,71	0,86
Image10	911*938	250,10	0,89

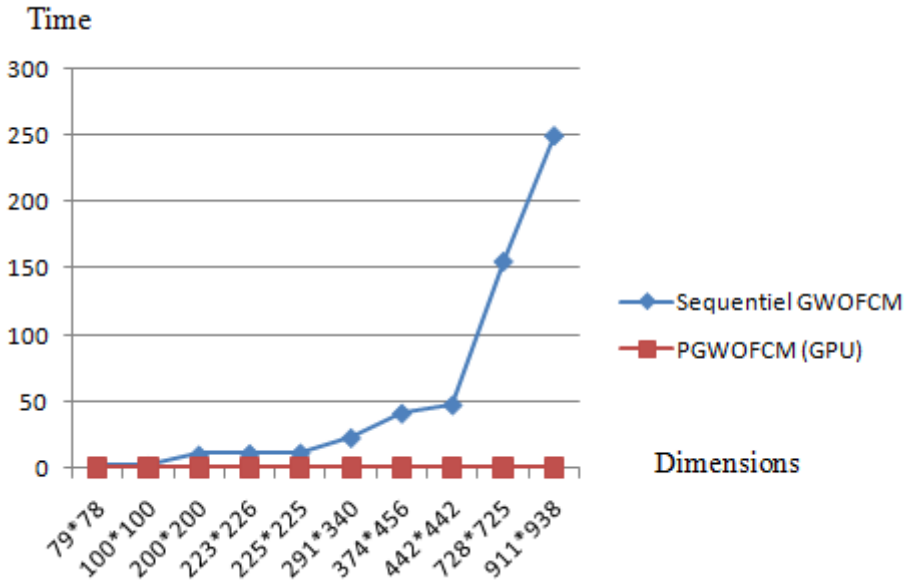


Figure 51. Comparing time between sequential-GWO-FCM and P-GWO-FCM on different sizes images.

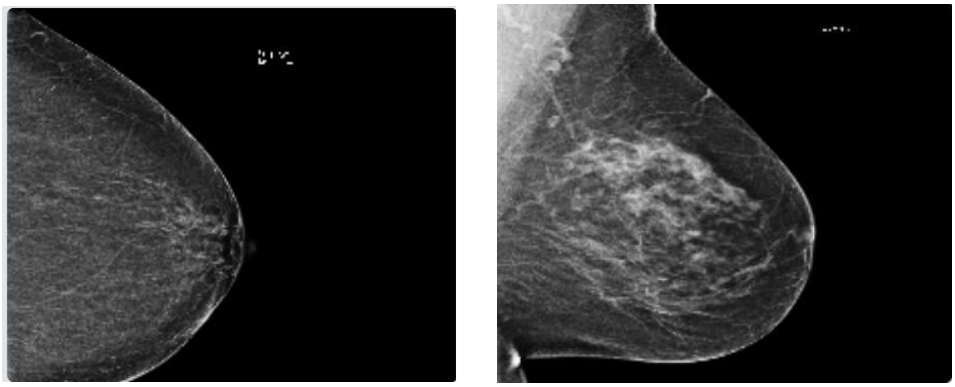
3.5.3. Evaluation on clinical breast cancer disease dataset

Another experimentation was conducted in this section, where the primary dataset utilized in this experiments is sourced from the Radiological Society of North America (RSNA), provided as part of a recent Kaggle competition [273]. This extensive dataset comprises 54,713 DICOM-format breast imaging studies, collected from approximately 11,000 patients. For each patient, a minimum of four images is included, captured from different breast laterality (left and right) and viewing angles, specifically craniocaudal (CC) and mediolateral oblique (MLO) views. The dataset is characterized by considerable diversity in both image

resolution and format, encompassing standard image formats such as JPEG and JPEG2000, and DICOM-specific pixel representations including monochrome1 and monochrome2. This heterogeneity presents a realistic challenge for preprocessing and model generalization. Figure 52 illustrates two representative samples from the RSNA dataset, showcasing one cancerous and one non-cancerous case, thereby highlighting the visual variation between healthy and pathological breast tissues.

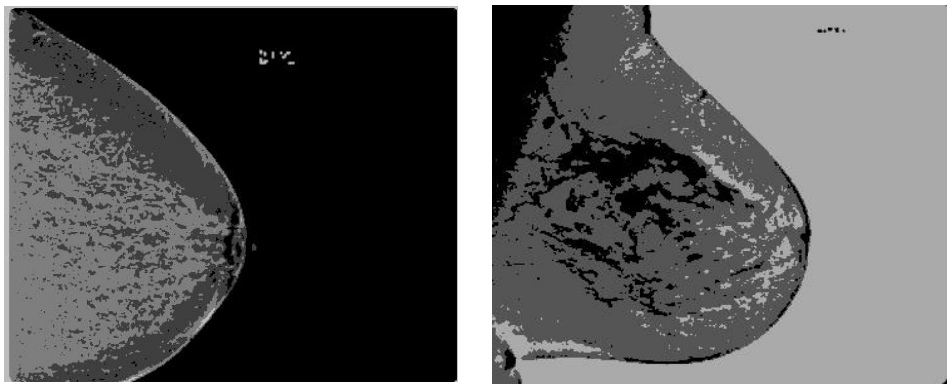
The visual results of the segmentation process are illustrated in Figures 52, 53, and 54. As shown in Figure 52, representative sample image from the dataset are presented to provide context for the segmentation task. Figure 53 displays the segmentation outcomes obtained using Sequential-GWO-FCM, highlighting its effectiveness in delineating key image regions. In contrast, Figure 54 presents the results produced by the P-GWO-FCM, allowing for a visual comparison between the two approaches in terms of segmentation quality and accuracy.

The results from Table 16 demonstrate that P-GWO-FCM method for image segmentation significantly outperforms the sequential method in terms of quality and computation time. Specifically, the parallel method achieved a Peak Signal-to-Noise Ratio (PSNR) of 31,97 and a Root Mean Square Error (RMSE) of 3.31, indicating a higher fidelity reconstruction and lower error compared to the sequential method. These performance metrics highlight the superiority of the parallel method, as a higher PSNR and lower RMSE generally reflect better segmentation quality. On the other, in terms of computational efficiency, P-GWO-FCM demonstrated superior performance with the shortest segmentation time of 1,74 second, outperforming Sequential GWO-FCM, which required 38,4 seconds. This noticeable difference in processing time highlights the efficiency of P-GWO-FCM, making it a more suitable choice for applications where rapid image segmentation is critical. The reduced execution time also suggests better scalability and responsiveness, especially in real-time or large-scale medical imaging scenarios.



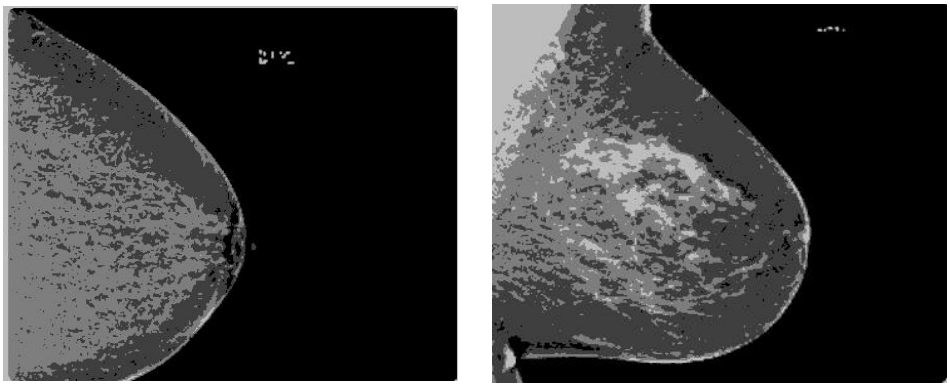
(a) (b)

Figure 52. Samples from the RSNA dataset, where (a) non-cancerous image, (b) cancerous image.



(a) (b)

Figure 53. The segmentation results obtained using Sequential-GWO-FCM where (a) non-cancerous image, (b) cancerous image.



(a) (b)

Figure 54. The segmentation results obtained using Parallel-GWO-FCMwhere (a) non-cancerous image, (b) cancerous image.

Table 16. Comparing segmentation results between the sequential-GWO-FCM and P-GWO-FCM on breast cancer images.

	Sequential-GWO-FCM	P-GWO-FCM
RMSE	6,42	3,31
PSNR	29,8	31,97
Time (second)	38,4	1,74

4. Conclusion

Parallel computing techniques have proven to be highly effective in addressing the computational challenges associated with large-scale image segmentation. By integrating machine learning, metaheuristic optimization, and parallel processing, significant improvements in segmentation accuracy and efficiency can be achieved. The use of multiprocessing on CPUs and GPU acceleration allows for the real-time processing of large datasets, making advanced segmentation techniques more accessible for practical applications. The combination of metaheuristic algorithms with ML-based segmentation offers a robust and adaptive approach to image segmentation. Methods such as parallel GWO-FCM on GPU and Parallel WOA-Kmeans using Multiprocessing demonstrate enhanced performance in terms of convergence speed and segmentation quality. Providing a scalable solution for diverse imaging domains. Future work can explore further optimizations in parallel computing frameworks, including distributed deep learning models, cloud-based parallel processing, and hybrid CPU-GPU architectures. Additionally, integrating explainable AI techniques with segmentation algorithms can improve interpretability and reliability in critical applications such as medical imaging.

In conclusion, parallel image segmentation methods offer a powerful approach for processing vast amounts of image data with improved speed, accuracy, and scalability. By continuing to advance parallel ML and metaheuristic techniques, researchers can further enhance image segmentation solutions across various domains, including healthcare, remote sensing, and real-time surveillance.

Conclusion and Future Work

Image segmentation is a fundamental process in medical image analysis, enabling the delineation of anatomical structures such as tumors, organs, and tissues. Traditional segmentation methodslike thresholding, edge detection, and region growingoften struggle with complex structures, noise, and high-resolution data, limiting their effectiveness in clinical applications.

To enhance segmentation accuracy and adaptability, machine learning (ML) techniques have been widely adopted. Algorithms such as Support Vector Machines (SVM), Random Forests (RF), K-Means, and Fuzzy C-Means (FCM) have been applied successfully. However, their performance heavily depends on optimal parameter settings and relevant feature selection.

To address these optimization challenges, metaheuristic algorithmssuch as Grey Wolf Optimizer (GWO), Whale Optimization Algorithm (WOA), and Genetic Algorithms (GA)offer effective global search strategies. They have proven useful for enhancing segmentation accuracy, selecting key features, and optimizing ML model parameters.

Nevertheless, metaheuristics are computationally intensive. This has motivated the use of parallel computing techniques, including CPU multiprocessing and GPU acceleration, to expedite metaheuristic-based image processing. Parallel metaheuristics enable faster convergence and scalability, making them well-suited for large-scale and real-time medical imaging tasks.

This thesis was driven by the need to develop scalable and accurate image processing systems capable of handling large medical datasets. Specifically, the thesis addresses the limitations of sequential metaheuristic algorithms in feature selection and image segmentation, proposing parallel and hybrid solutions that integrate metaheuristics, machine learning, and high-performance computing.The research explores the intersection of these domains to tackle two primary challenges:

- The selection of relevant features for accurate classification of medical conditions such as breast cancer,
- the segmentation of complex medical images, particularly MRI scans, with high precision and computational efficiency.

The first contribution of this work lies in the development of two robust metaheuristic-based feature selection approaches for breast cancer classification. The first method combines Grey Wolf Optimizer (GWO) with Random Forest (RF), achieving high classification accuracy (up to 98.6% on the WDBC dataset). The second method integrates Correlation-based Feature Selection (CFS) with GWO, followed by classification using RF, Support Vector Machine (SVM), and Naïve Bayes (NB). This hybrid strategy, termed CMGWO-RF, further improves performance, reaching 99.12% accuracy, and proves effective in reducing dimensionality while enhancing model interpretability.

The second major contribution concerns the design and implementation of parallel image segmentation frameworks tailored for medical imaging. Twostrategies were proposed:

1. Parallel WOA-KMeans using multiprocessing, which leverages CPU cores to accelerate the segmentation process while improving clustering quality. The approach demonstrated superior performance on grayscale test images and medical datasets, with reduced execution times (e.g., from 186s to 16s) and better segmentation metrics such as PSNR, SSIM, and accuracy.
2. P-GWO-FCM for MRI segmentation, which integrates GWO with Fuzzy C-Means (FCM) optimized by Fuzzy Entropy, and implements the entire pipeline on GPU using CUDA. This method significantly improves segmentation quality, achieving a Jaccard Similarity (JS) score of 0.92, and Dice coefficients of 0.93, 0.89, and 0.95 for WM, GM, and CSF tissues, respectively. Compared to traditional FCM, sequential GWO-FCM, FCM-GENIUS, and Deep-JCR, the P-GWO-FCM method outperformed all in both accuracy and execution time (from 250s down to 0.89s on large images).

Together, these contributions demonstrate the potential of combining metaheuristics, machine learning, and parallelism to develop scalable and intelligent image analysis systems, particularly in domains requiring high precision such as medical diagnostics. The results confirm that parallel and hybrid metaheuristic strategies not only improve the effectiveness of image segmentation and classification, but also address the computational bottlenecks associated with large-scale image processing.

Building upon the results of this thesis, several future research directions can be pursued:

- Extending the proposed approaches to multi-modal and 3D medical images, integrating modalities like PET, and CT
- Developing hybrid deep learning-metaheuristic systems, combining CNNs or Vision Transformers with GWO, PSO, or WOA.
- Implementing distributed computing and cloud deployment of segmentation pipelines for real-time analysis in clinical settings.
- Applying multi-objective optimization frameworks to balance segmentation accuracy, processing time, and memory usage.

- Ali Mezaghrani, Mohammed Debakla, Khalifa Djemal, (2024). Nove1 feature selection method for accurate breast cancer classification using Correlation coefficient and Modified GWO Algorithm, Computer science journal of moldova, v o1.32, no .2(9 5), 2024,

- A. Mezaghrani, M. Debakla and K. Djemal, "Parallel Whale Optimization Algorithm-Kmeans for Image Segmentation using Multiprocessing,*" 2025 International Symposium on iNnovative Informatics of Biskra (ISNIB)*, Biskra, Algeria, 2025, pp. 1-6, doi: 10.1109/ISNIB64820.2025.10983906.

- Ali Mezaghrani, Mohamed Debakla, and Khalifa Djemal « Robust Method for Breast cancer classification Based on Feature selection using RGWO Algorithm » First International conference, on Aftificiallntelligence: Theories and Applications ICÆTA 2022 Mascara, Algeria, November 7-8' 7022;

- Ali Mezaghrani, Mohamed Debakla, and Khalifa, « Feature selection using correlation method for breast cancer classification», First National Conference on Science and Technology, June 2022, Mascara, Algeria.

Bibliography

[1] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. « U-net : Convolutional networks for biomedical image segmentation.» International Conference on Medical image computing and computer-assisted intervention. Springer, Cham, 2015.

[2] Cortes, C., and Vapnik, V. «Support-vector networks. Machine Learning », 20 (3), 273–297,1995. DOI:10.1007/BF00994018.

[3] MacQueen, J. Some methods for classification and analysis of multivariate observations. Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, 1, 281–297,1967.

[4] Bezdek, J. C. Pattern Recognition with Fuzzy Objective Function Algorithms. Springer,1981.

[5] G. L. Sayed, A. E. Hassanien and A. T. Azar, Feature selection via a novel chaotic crow search algorithm, Neural Comput. Appl. (31) 171–188. 2019.

[6]. H. Rao, X. Shi, A. Rodrigue, J. Feng and Y. Xia, Feature selection based on artificial bee colony and gradient boosting decision tree, Appl. Soft Comput, (74) 634–642 . 2019.

[7]. M. Abdel-Basset, D. El-Shahat, I. El-henawy and S. Mirjalili, A new fusion of grey wolf optimizer algorithm with a two-phase mutation for feature selection, Exp. Syst. Appl, (139) 112824 . 2020.

[8]. S. Kumar and M. Singh, «Breast Cancer Detection Based on Feature Selection Using Enhanced Grey Wolf Optimizer and Support Vector Machine Algorithms»,Vietnam Journal ofComputer Science, vol. 8, no. 2, pp. 177–197, 2021, DOI: 10.1142/S219688882150007X.

[9] M. Darzi, A. AsgharLiaei, M. Hosseini, and H. Asghari, «Feature selection for breast cancer diagnosis: A case-based wrapper approach »,World Academy of Science, Engineering and Technology,International Journal of Medical, Health, Biomedical, Bioengineeringand Pharmaceutical Engineering, vol. 5, pp. 220–223, 2011. https://api.semanticscholar.org/CorpusID:51751976.

[10] A.E. Rahmani, M. Katouli, «Breast cancer detection improvement by grasshopper optimization algorithm and classification SVM», Revue d'Intelligence Artificielle, vol.34, no.2, pp. 195–202, 2020, DOI: https://doi.org/10.18280/ria.340210.

[11] Mirjalili S, Mirjalili SM, Lewis A,«Grey wolf optimizer». Adv Eng Softw 69:46–61, 2014.

[12] Mirjalili S,«The ant lion optimizer». Adv Eng Softw 83:80–98, 2015.

[13] wHolland J (1991) «Adaptation in natural and artificial systems», 1975

[14] Mirjalili S, Lewis A,«The whale optimization algorithm». Adv Eng Softw 95:51–67, 2016.

[15] Bao X, Jia H, Lang C, «A novel hybrid Harris Hawks optimization for color image multilevel thresholding segmentation ». IEEE Access 7:76529–76546. 2019. https://doi.org/10.1109/ACCESS.2019.2921545.

[16] X.-p. Wang, J.-S. Pan and S.-C. Chu, «A parallel multi-verse optimizer for application in multilevel image segmentation », Ieee Access, pp. 32018-32030, 2020.

[17] Xiao Sui, Shu-Chuan Chu, Jeng-Shyang Pan and HaoLuo. «Parallel Compact Differential Evolution for Optimization Applied to Image Segmentation». Appl. Sci. 10, 2195, 2020, doi:10.3390/app10062195.

[18] Ali Mezaghrani, Mohamed Debakla, and Khalifa Djemal « Robust Method for Breast cancer classification Based on Feature selection using RGWO Algorithm » First International conference, on Aftificial lntelligence: Theories and Applications ICÆTA 2022 Mascara, Algeria, November 7-8' 7022.

[19] Ali Mezaghrani, Mohammed Debakla, Khalifa Djemal. « Nove1 feature selection method for accurate breast cancer classification using Correlation coefficient and Modified GWO Algorithm ». Computer science journal of moldova, v o1.32, no .2(9 5), 2024.

[20] Gonzalez, R. C., & Woods, R. E. « Digital Image Processing (2nd ed.) ». Prentice Hall, 2002.

[21] Moustakidesa,G.,Briassoulisb,D.,E. Psarakisa,E.,Dimasb, «3D image acquisition and NURBS based geometry modelling of natural objects », Advances in Engineering Software, 955–969,2000.

[22] Haralick, R. M. & Shapiro, L. G. «Image Segmentation Techniques». Computer Vision, Graphics, and Image Processing. 1985.

[23] Canny, J. «A Computational Approach to Edge Detection ». IEEE Transactions on Pattern Analysis and Machine Intelligence ,1986.

[24] Marr, D., and Hildreth, E. «Theory of edge detection. Proceedings of the Royal Society of London». Series B. Biological Sciences, 207(1167), 187–217.1980.

[25] D. Sangeetha, and P. Deepa, «FPGA implementation of cost-effective robust Canny edge detection algorithm », Journal of Real-Time Image Processing, vol.16, no.4, pp.957-970, 2019.

[26] Sezgin, M., and Sankur, B. «Survey over Image Thresholding Techniques and Quantitative Performance Evaluation ». Journal of Electronic Imaging, 2004.

[27] Bradley, D., and Roth, G. « Adaptive thresholding using the integral image » . Journal of Graphics, GPU, and Game Tools, 12(2), 13–21, 2007.

[28] Otsu, N.« A Threshold Selection Method from Gray-Level Histograms » . IEEE Transactions on Systems, Man, and Cybernetics, 1979.

[29] Gurjeet Kaur, Kuldeep Singh, « A comparative study of image segmentation using thresholding techniques», Conference Paper , March 2013, Gurgaon College of Engineering, Gurgaon, 2013.

[30] Adams, R., and Bischof, L. « Seeded region growing ». IEEE Transactions on Pattern Analysis and Machine Intelligence, 16(6), 641–647, 1994.

[31] Vincent, L., and Soille, P. « Watersheds in digital spaces: An efficient algorithm based on immersion simulations ». IEEE Transactions on Pattern Analysis and Machine Intelligence, 13(6), 583–598, 1991.

[32] Horowitz, S. L., and Pavlidis, T. « Picture segmentation by a tree traversal algorithm ». Journal of the ACM (JACM), 23(2), 368–388, 1976.

[33] Geman, S., & Geman, D. « Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images » . IEEE Transactions on Pattern Analysis and Machine Intelligence, 6(6), 721–741, 1984.

[34] B. R. Lee, «An image segmentation approach for fruit defect detection using k-means clustering and graph-based algorithm », Vietnam Journal of Computer Science, vol. 2, no. 1, pp. 25-33, 2015.

[35] Comaniciu, D., and Meer, P. « Mean shift: A robust approach toward feature space analysis ». IEEE Transactions on Pattern Analysis and Machine Intelligence, 24(5), 603–619, 2002.

[36] He, K., Gkioxari, G., Dollár, P., andGirshick, R. « Mask R-CNN ». Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2961–2969, 2017.

[37] Wang, J., Sun, K., Cheng, T., et al.« Deep high-resolution representation learning for visual recognition ». IEEE Transactions on Pattern Analysis and Machine Intelligence, 43(10), 3349–3364, 2020.

[38] J. Long, E. Shelhamer, and T. Darrell, «Fully convolutional networksfor semantic. Segmentation »,in IEEE Conference on Computer Visionand Pattern Recognition, pp. 3431-3440, 2015.

[39] Goodfellow I , Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y, «Generative adversarial nets». In: Advances in neural information processing systems,2014.

[40] Hamed Mohammadi Azni, Mohsen Afsharchi, Armin Allahverdi, «Improving brain tumor segmentation performance using CycleGAN based feature extraction », Multimedia Tools and Applications, 82:18039–18058, 2023 ,https://doi.org/10.1007/s11042-022-14174-3.

[41] Merity S, « Single headed attention rnn: Stop thinking with your head». 2019, arXiv preprint.arXiv: 1911.11423

[42] Ning J, Jiaxian W, Xiang M, Ke Y, Yuchang M, « Multitask learning model based on multi-scale cnn and lstm for sentiment classification ». IEEE Access 8:77060–77072, 2020.

[43] Chong Z, Pin L, Kai Qin A, Kay Chen T, « Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics ». IEEE Trans Neural Networks Learn Syst 28(10):2306–2318, 2016.

[44] X. Zhang, J. Cui, W. Wang, and C. Lin, «A study for texture feature extraction of high-resolution satellite images based on a direction measure and gray level co-occurrence matrix fusion algorithm »,Sensors, vol. 17, no. 7, p. 1474, 2017.

[45] T. Ojala, M. Pietikäinen, and T. Maenpaa, «Multiresolution gray-scale and rotation invariant texture classification with local binary patterns »,IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 971–987, 2002.

[46] Dalal, N., & Triggs, B. « Histograms of Oriented Gradients for Human Detection ». IEEE CVPR, 2005.

[47] Herv ´e Abdi1∗ and Lynne J. Williams2, « Principal component analysis », Volume 2, July/August 2010.DOI: 10.1002/wics.101

[48] Sara Ibrahim, Saima Nazir and Sergio A. Velastin, «Feature Selection Using Correlation Analysis and Principal Component Analysis for Accurate Breast Cancer Diagnosis, »J.Imaging, vol. 7, no. 11, Article No. 225, 2021. [Online]. Available: https://doi.org/10.3390/jimaging7110225.

[49] D. Jaswal, S. V, and K. P. Soman, «Image Classification Using Convolutional Neural Networks,» Int. J. Sci. Eng. Res., vol. 5, no. 6, pp. 1661–1668, 2014, doi: 10.14299/ijser.2014.06.002.

[50] Tarik Saidani. Multi-level Optimization of an Image Processing Application on Parallel Machines. Other [cond-mat.other]. Université Paris Sud - Paris XI, 2012. French. NNT: 2012PA112268. tel-00776111

[51] Michael J. Flynn. Some computer organizations and their effectiveness. IEEE Transactions on Computers, 21(9) :948–960, September 1972.

[52] J.L. Hennessy and D.A. Patterson. Computer Architecture : A Quantitative Approach. The Morgan Kaufmann Series in Computer Architecture and Design. Elsevier Science, 2011.

[53] Michael J. Flynn and Kevin W. Rudd. Parallel architectures. ACM Comput. Surv., 28(1) :67–70, March 1996.

[54] H.T. Kung, C.E. Leiserson, and Carnegie-Mellon University. Dept. of Computer Science. Systolic Arrays for (VLSI). CMU-CS. Carnegie-Mellon University, Department of Computer Science, 1978.

[55] IEEE. IEEE Standard . 1003.1c-1995 thread extensions. Technical report, IEEE, 1995.

[56] L. Dagum and R. Menon. OpenMP : an industry standard API for shared-memory programming. IEEE Computational Science and Engineering, 5(1) :46–55, 1998.

[57] V. S. Sunderam. PVM: a framework for parallel distributed computing. Concurrency : Pract. Exper.,2(4) :315–339, November 1990.

[58] Message Passing Forum. MPI : A Message-Passing Interface Standard. Technical report, Message Passing Interface Forum, Knoxville, TN, USA, 1994.

[59] CORPORATE Rice University. High performance fortran language specification. SIGPLAN Fortran Forum, 12(4) :1–86, December 1993.

[60] Ken Kennedy, Charles Koelbel, and Hans Zima. The rise and fall of high performance fortran : an historical object lesson. In Proceedings of the third ACM SIGPLAN conference on History of programming languages, HOPL III, pages 7–1–7–22, New York, NY, USA, 2007. ACM.

[61] Ujval Kapasi, William J. Dally, Scott Rixner, John D. Owens, and Brucek Khailany. The Imagine stream processor. In Proceedings 2002 IEEE International Conference on Computer Design, pages 282–288, September 2002.

[62] W. J. Dally, P. Hanrahan, M. Erez, T. J. Knight, F. Labonte, J-H A., N. Jayasena, U. J. Kapasi, A. Das, J. Gummaraju, and I. Buck. Merrimac : Supercomputing with streams. In SC'03, Phoenix, Arizona, November 2003.

[63] William Thies, Michal Karczmarek, and Saman Amarasinghe. Streamit : A language for streaming applications. In International Conference on Compiler Construction, Grenoble, France, Apr 2002.

[64] Ian Buck, Tim Foley, Daniel Horn, Jeremy Sugerman, Kayvon Fatahalian, Mike Houston, and Pat Hanrahan. Brook for gpus : stream computing on graphics hardware. ACM Trans. Graph., 23(3) :777–786, August 2004.

[65] John Nickolls, Ian Buck, Michael Garland, and Kevin Skadron. Scalable parallel programming with cuda. Queue, 6 :40–53, March 2008.

[66] Khronos OpenCLWorking Group. The OpenCL Specification, version 1.0.29, 8 December 2008.

[67] M Vijayaraj, R.Malar Vizhi, P.Chandrakala, Laith H. Alzubaidi, Khasanov Muzaffar, R Senthilkumar,"Parallel and Distributed Computing for High- Performance Applications",E3S Web of Conferences 399, 04039 , 2023, https://doi.org/10.1051/e3sconf/202339904039.

[68]Alismaili, S. Z., Li, M., Shen, J., Huang, P., He, Q., and Zhan, W. Organisational-level assessment of cloud computing adoption: Evidence from the Australian SMEs. Journal of Global Information Management, 28(2), 73–89. 2020, doi:10.4018/JGIM.2020040104

[69] Mell, P., and Grance, T. Perspectives on cloud computing and standards. National Institute of Standards and Technology (NIST). Information Technology Laboratory,2009.

[70] Rimal, B. P., Jukan, A., Katsaros, D., and Goeleven, Y. Architectural requirements for cloud computing systems: An enterprise cloud approach. Journal of Grid Computing, 9(1), 3–26,2011. doi:10.1007/s10723-010-9171-y

[71] Collange, S., Dandass, Y.S., Daumas, M. and Defour, D. 'Using graphics processors for parallelizing hash-based data carving', in HICCS, Hawaii International Conference on System Sciences, pp.1–10. 2009.

[72]Kalaiselvi, T., Sriramakrishnan, P., and Somasundaram, K. Survey of using GPU CUDA programming model in medical image analysis. Informatics in Medicine Unlocked, 9, 133–144, 2017. doi:10.1016/j.imu.2017.08.001

[73] Paz-Gallardo, Abel and Plaza, Antonio. A New Morphological Anomaly Detection Algorithm for Hyperspectral Images and its GPU Implementation. Proceedings of SPIE - The International Society for Optical Engineering. 8157. 10.1117/12.892282. 2011.

[74] Glaskowsky, P.N. (2009) NVIDIA's Fermi: The First CompleteGPU Computing Architecture [online] http://www.nvidia.com/content/pdf/fermi_white_papers/p.glaskowsky_nvidia's_fermithe_first_complete_gpu_architecture.pdf

[75] ImaneZouaneb, M.Belarbi, A.Chouarfia. “Multi approach for realtimesystems specification: case study of GPU parallel systems,” in International Journal of Big Data Intelligence IJBDI, vol.3,pp.122-141. 2016

[76] Memeti, S., Li, L., Pllana, S., Kołodziej, J., & Kessler, C.Benchmarking OpenCL, OpenACC, OpenMP, and CUDA. Proceedings of the 2017 Workshop on Adaptive Resource Management and Scheduling for Cloud Computing - ARMS-CC ’17. 2017. doi:10.1145/3110355.3110356

[77] Liu, W., Schmidt, B., Voss, G., & Müller-Wittig, W.Accelerating molecular dynamics simulations using Graphics Processing Units with CUDA. Computer Physics Communications, 179(9), 634–641. 2008. doi:10.1016/j.cpc.2008.05.008

[78] Scholl, I.; Aach, T.; Deserno, T.M.; Kuhlen, T. Challenges of Medical Image Processing. Comput. Sci. Res. Dev, 26, 5–13. 2011.

[79] Sancho, J.; Sutradhar, P.; Rosa, G.; Chavarrías, M.; Perez-Nunez, A.; Salvador, R.; Lagares, A.; Juárez, E.; Sanz, C. GoRG: Towards a GPU-Accelerated Multiview Hyperspectral Depth Estimation Tool for Medical Applications. Sensors 2021, 21, 4091.

[80] Graca, C.; Falcao, G.; Figueiredo, I.N.; Kumar, S. Hybrid Multi-GPU Computing: Accelerated Kernels for Segmentation and Object Detection with Medical Image Processing Applications. J. Real-Time Image Process. 2017, 13, 227–244.

[81] De, A.; Zhang, Y.; Guo, C. A Parallel Adaptive Segmentation Method Based on SOM and GPU with Application to MRI Image Processing. Neurocomputing 2016, 198, 180–189.

[82] Celik, B.; Gul, S.; Celik, M. A Stochastic Programming Approach to Surgery Scheduling under Parallel Processing Principle. Omega 2023, 115, 102799.

[83] Kalaiselvi, T.; Sriramakrishnan, P.; Somasundaram, K. Survey of Using GPU CUDA Programming Model in Medical Image Analysis. Inform. Med. Unlocked 2017, 9, 133–144.

[84] Bin Yu, Quan Zhou, Li Yuan, Huageng Liang, Pavel Shcherbakov, and Xuming Zhang, “3D medical image segmentation using the serial–parallel convolutional neural network and transformer based on crosswindowselfattention”, CAAI Transactions on Intelligence Technology, 2024, DOI: 10.1049/cit2.12411

[85] Aniket Pramanik, Xiaodong Wu, and Mathews Jacob, “Joint Calibrationless Reconstruction and Segmentation of Parallel MRI” ,ieee transactions on medical imaging, vol. Xx, no. Xx, xxxx 2020.

[86] Dongsheng Wang, TiezhenXv, Jiehui Liu, Jianshen Li, Lijie Yang and Jinxi Guo, “Bidirectional Efficient Attention Parallel Network for Segmentation of 3D Medical Imaging”, Electronics 2024, 13, 3086. https://doi.org/10.3390/electronics13153086

[87] Fatma Chahbar, Medjeded Merati, and Said Mahmoudi,”MPB-UNet: Multi-Parallel Blocks UNet for MRI Automated Brain Tumor Segmentation”, Electronics 2025, 14, 40, https://doi.org/10.3390/electronics14010040

[88] Sanjay Saxena , Suraj Shama, “Brain Tumor Segmentation by Texture Feature Extraction with the Parallel Implementation of Fuzzy C-Means using CUDA on GPU”, 5th IEEE International Conference on Parallel, Distributed and Grid Computing(PDGC-2018), 20-22 Dec, 2018, Solan, India

[89] Noureddine Ait Ali, BouchaibCherradi, Ahmed El Abbassi, Omar Bouattane, and Mohamed Youssfi, “Parallel Implementation and Performance Evaluation of some Supervised Clustering Algorithms for MRI Images Segmentation”, Association for Computing Machinery(2019), BDIoT'19, October 23–24, 2019, Rabat, Morocco, 2019. https://doi.org/10.1145/3372938.3373007.

[90] Shadi AlZuÕbi, Mohammed Shehab, Mahmoud Al-Ayyoub, Yaser Jararweh, Brij Gupta, Parallel Implementation for 3D Medical Volume Fuzzy Segmentation, Pattern RecognitionLetters, 2018, doi: 10.1016/j.patrec.2018.07.026

[91] PrajoonaValsalan, P. Sriramakrishnan, S. Sridhar, G. Charlyn Pushpa Latha, A. Priya, S. Ramkumar, A. Robert Singh, and T. Rajendran, "Knowledge based fuzzy c‑means method for rapid brain tissues segmentation of magnetic resonance imaging scans with CUDA enabled GPU machine", Journal of Ambient Intelligence and Humanized Computing, May 2020, https://doi.org/10.1007/s12652-020-02132-6.

[92] Rakhimov, M., Mamadjanov, D., and Mukhiddinov, A. A High-Performance Parallel Approach to Image Processing in Distributed Computing. 2020 IEEE 14th International Conference on Application of Information and Communication Technologies (AICT), 2020. doi:10.1109/aict50176.2020.936884

[93]. Benchara, F. Z., and Youssfi, M. A new scalable distributed k-means algorithm based on Cloudmicro-services for High-performance computing. Parallel Computing, 101, 102736, 2021.

[94]. Enfedaque, P., Aulı-Llinas, F., and Moure, J. C.GPU implementation of bitplane coding with parallel coefficient processing for high performance image compression. IEEE Transactions onParallel and Distributed Systems, 28(8), 2272-2284. 2017.

[95] Elsayed Badr, Sultan Almotairi , Mustafa Abdul Salam , Hagar Ahmed, New Sequential and Parallel Support Vector Machine with Grey Wolf Optimizer for Breast Cancer Diagnosis, Alexandria Engineering Journal (2022) 61, 2520–2534.

[96] Sawsan M. Mahmoud, Rokaia Shalal Habeeb, "Analysis of Large Set of Images Using MapReduce Framework", I.J. Modern Education and Computer Science, 2019, 12, 47-52 Published Online December 2019 in MECS (http://www.mecs-press.org/) DOI: 10.5815/ijmecs.2019.12.05

[97]. Tan, X.; Di, L.; Zhong, Y.; Yao, Y.; Sun, Z.; Ali, Y. Spark-based adaptive Mapreduce data processing method for remote sensing imagery. Int. J. Remote Sens. 2021, 42, 191–207.

[98] S. Almufti, R. Asaad and B. Salim, "Review on Elephant Herding Optimization Algorithm Performance in Solving Optimization Problems", Sci-encepubco.com, 2019. [Online]. Available: https://www.sciencepubco.com/index.php/ijet/article/view/28473. [Accessed: 26- May- 2019].

[99] S. Almufti, "U-Turning Ant Colony Algorithm powered by Great Deluge Algorithm for the solution of TSP Problem", Hdl.handle.net, 2018.

[100] S. Almufti, "Using Swarm Intelligence for solving NPHard Problems," Academic Journal of Nawroz University, vol. 6, no. 3, pp. 46–50, 2017. https://doi.org/10.25007/ajnu.v6n3a78.

[101] S. Almufti and A. Shaban, "U-Turning Ant Colony Algorithm for Solving Symmetric Traveling Salesman Problem", Academic Journal of Nawroz University, vol. 7, no. 4, pp. 45-49, 2018. https://doi.org/10.25007/ajnu.v6n4a270.

[102] Fister, I., Yang, X.-S., Fister, I., Brest, J., and Fister, D. (2013a). A Brief Review of Nature-Inspired Algorithms for Optimization. . (2013a). http://arxiv.org/abs/1307.4186

[103] Almufti, S., Marqas, R., & Asaad, R. Comparative study between elephant herding optimization (EHO) and U-turning ant colony optimization (U-TACO) in solving symmetric traveling salesman problem (STSP). Journal Of Advanced Computer Science and Technology, 8(2), 32.2019.

[104] Rai, D., & Tyagi, K.Bio-inspired optimization techniques. ACM SIGSOFT Software Engineering Notes, 38(4), 1–7.2013.https://doi.org/10.1145/2492248.2492271

[105] Almufti, S.Using Swarm Intelligence for solving NPHard Problems. Academic Journal of Nawroz University, 6(3), 46–50. 2017.https://doi.org/10.25007/ajnu.v6n3a78.

[106] Ihsan, R. R., Almufti, S. M., Ormani, B. M., Asaad, R. R., and Marqas, R. B. (2021). A survey on Cat Swarm Optimization algorithm. Asian J. Res. Comput. Sci, 10, 22-32. 2021.

[107] Sadeeq, H. T., Abdulazeez, A. M., Kako, N. A., Zebari, D. A., and Zeebaree, D. Q. A New Hybrid Method for Global Optimization Based on the Bird Mating Optimizer and the Differential Evolution. 2021 7th

International Engineering Conference “Research and Innovation amid Global Pandemic" (IEC), 54–60. 2021.https://doi.org/10.1109/IEC52205.2021.9476147

[108] Agrawal, P., Abutarboush, H. F., Ganesh, T., andMohamed, A. W. Metaheuristic Algorithms on Feature Selection: A Survey of One Decade of Research (2009-2019). IEEE Access, 9, 26766–26791.2021. https://doi.org/10.1109/ACCESS.2021.3056407.

[109] Storn R, Price K, Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11(4):341–359. 1997.

[110] Yang XS A new metaheuristic bat-inspired algorithm. In: Nature Inspired Cooperative Strategies for Optimization. Springer, (2010), p 65–74

[111] Yang XS, Deb S Cuckoo search via levy flights. In: 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), (2009),IEEE, pp 210–214

[112] Saremi S, Mirjalili S, Lewis A Grasshopper optimisation algorithm: theory and application. Adv Eng Softw , (2017),105:30–47

[113] Yang XS. Firefly algorithms for multimodal optimization. In: International symposium on stochasticalgorithms, Springer, pp 169–178.2009.

[114] Mirjalili S. Dragonfly algorithm: a new meta-heuristic optimization technique for solving singleobjective, discrete, and multi-objective problems. Neural Comput Appl 27(4):1053–1073. 2016.

[115] Yang XS. Flower pollination algorithm for global optimization. In: International Conference on Unconventional Computing and Natural Computation, Springer, pp 240–249.2012.

[116] Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671–680

[117] Shareef H, Ibrahim AA, Mutlag AH. Lightning search algorithm. Appl Soft Comput 36:315–333. 2015.

[118] Rashedi E, Nezamabadi-Pour H, Saryazdi S. Gsa: a gravitational search algorithm. Inf Sci 179(13):2232–2248.2009.

[119] Abedinpourshotorban H, Shamsuddin SM, Beheshti Z et al. Electromagnetic field optimization: A physics-inspired metaheuristic optimization algorithm. Swarm Evol Comput 26:8–22. 2016.

[120] Shi Y. Brain storm optimization algorithm. In: International Conference in Swarm Intelligence, Springer, pp 303–309. 2011.

[121] Rao RV, Savsani VJ, Vakharia D, Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput Aided Des 43(3):303–315.2011.

[122] Mohamed AW, Hadi AA, Mohamed AK, Gaining-sharing knowledge based algorithm for solving optimization problems: a novel nature-inspired algorithm. Int J Mach Learn Cybern 11(7):1501–1529.2020.

[123] Amir Seyyedabbasi, Wadhah Zeyad Tareq Tareq, Nebojsa Bacanin, An Efective Hybrid Metaheuristic Algorithm for Solving Global Optimization Algorithms, Multimedia Tools and Applications (2024) 83:85103–85138 https://doi.org/10.1007/s11042-024-19437-9

[124] Lee, R.S., Dunnmon, J.A., He, A., Tang, S., Re, C., Rubin, D.L.: Comparison of segmentation-free and segmentation-dependent computer-aided diagnosis of breast masses on a public mammography dataset. J. Biomed. Inform. 113, 103656 (2021).

[125] Jiang, D., Li, G., Tan, C., Huang, L., Sun, Y., Kong, J.: Semantic segmentation for multiscale target based on object recognition using the improved faster-rcnn model. Futur. Gener. Comput. Syst. 123, 94–104 .2021.

[126] Menglin, W., Cai, X., Chen, Q., Ji, Z., Niu, S., Leng, T., Rubin, D.L., Park, H.: Geographic atrophy segmentation in sd-oct images using synthesized fundus autofluorescence imaging. Comput. Methods Programs Biomed. 182, 105101 .2019.

[127] Jia, W., Tian, Y., Luo, R., Zhang, Z., Lian, J., Zheng, Y.: Detection and segmentation of overlapped fruits based on optimized mask r-cnn application in apple harvesting robot. Comput. Electron. Agric. 172, 105380 .2020.

[128] Mandal, D., Chatterjee, A., Maitra, M.: Robust medical image segmentation using particle swarm optimization aided level set based global fitting energy active contour approach. Eng. Appl. Artif. Intell. 35, 199–214 .2014.

[129] Singh, V.: Sunflower leaf diseases detection using image segmentation based on particle swarm optimization. Artif. Intell. Agric. 3, 62–68 .2019.

[130] Singh, S., Mittal, N., Thakur, D., Singh, H., Oliva, D.: Nature and biologically inspired image segmentation techniques. Arch. Comput. Methods Eng. 1, 28 .2021.

[131] Farshi, T.R., Drake, J.H., O¨ zcan, E.: A multimodal particle swarm optimization-based approach for image segmentation. Expert Syst. Appl. 149, 113233 .2020.

[132] Essam H. Houssein, Gaber M. Mohamed, Youcef Djenouri,Yaser M. Wazery, Ibrahim A. Ibrahim. “Nature inspired optimization algorithms for medical image segmentation: a comprehensive review”, Cluster Computing , 27:14745–14766 .2014. https://doi.org/10.1007/s10586-024-04601-5(0123456789().

[133] Oliva, D., Elaziz, M.A., Hinojosa, S.: Multilevel thresholding for image segmentation based on metaheuristic algorithms. In: Metaheuristic Algorithms for Image Segmentation: Theory and Applications, pages 59–69. Springer, 2019.

[134] Sathya, P.D., Kalyani, R., Sakthivel, V.P.: Color image segmentation using Kapur, Otsu and minimum cross entropy functions based on exchange market algorithm. Expert Syst.
Appl. 172, 114636 ,2021.

[135] Elaziz, M.A., Heidari, A.A., Fujita, H., Moayedi, H.: A competitive chain-based Harris hawks optimizer for global optimization and multi-level image thresholding problems. Appl. Soft Comput. 95, 106347 , 2020.

[136] Aziz, M.A.E., Ewees, A.A., Hassanien, A.E.: Whale optimization algorithm and moth-flame optimization for multilevel thresholding image segmentation. Expert Syst. Appl. 83, 242–256 , 2017.

[137] Houssein, E.H., El-dinHelmy, B., Oliva, D., Pradeep, J., Premkumar, M., Elngar, A.A., Shaban, H.: An efficient multithresholding based covid-19 ct images segmentation approach using an improved equilibrium optimizer. Biomed. Signal Process. Control 73, 103401, 2022.

[138] Houssein, E.H., Emam, M.M., Ali, A.A.: An efficient multilevel thresholding segmentation method for thermography breast cancer imaging based on improved chimp optimization algorithm. Expert Syst. Appl. 185, 115651, 2021.

[139] Gao, H., Zheng, F., Pun, C.-M., Haidong, H., Lan, R.: A multilevel thresholding image segmentation based on an improved artificial bee colony algorithm. Comput. Electr. Eng. 70, 931–938, 2018.

[140] Sri Madhava Raja, N., Arockia Sukanya, S., Nikita, Y.: Improved pso based multi-level thresholding for cancer infected breast thermal images using Otsu. Proced. Comput. Sci. 48, 524–529, 2015

[141] Mohamad Amin Bakhshali and Mousa Shamsi: Segmentation of color lip images by optimal thresholding using bacterial foraging optimization (bfo). J. Comput. Sci. 5(2), 251–257, 2014.

[142] Agrawal, S., Panda, R., Bhuyan, S., Panigrahi, B.K.: Tsallis entropy based optimal multilevel thresholding using cuckoo search algorithm. Swarm Evol. Comput. 11, 16–30, 2013.

[143] Manikandan, S., Ramar, K., Willjuice Iruthayarajan, M., Srinivasagan, K.G.: Multilevel thresholding for segmentation of medical brain images using real coded genetic algorithm. Measurement 47, 558–568, 2014.

[144] Hang, S., Zhao, D., Fanhua, Yu., Heidari, A.A., Zhang, Yu., Chen, H., Li, C., Pan, J., Quan, S.: Horizontal and vertical search artificial bee colony for image segmentation of covid-19 x-ray images. Comput. Biol. Med. 142, 105181, 2022.

[145] Aljanabi, M., O¨ zok, Y.E., Rahebi, J., Abdullah, A.S.: Skin lesion segmentation method for dermoscopy images using artificial bee colony algorithm. Symmetry 10(8), 347, 2018.

[146] Abdel-Basset, M., Chang, V., Mohamed, R.: Hsma_woa: a hybrid novel slime Mould algorithm with whale optimization algorithm for tackling the image segmentation problem of chest x-ray images. Appl. Soft Comput. 95, 106642, 2020.

[147] Nameirakpam Dhanachandra and Yambem Jina Chanu: An image segmentation approach based on fuzzy c-means and dynamic particle swarm optimization algorithm. Multimed. Tools Appl. 79(25), 18839–18858 , 2020.

[148] Bandyopadhyay, R., Kundu, R., Oliva, D., Sarkar, R.: Segmentation of brain mri using an altruistic Harris hawks' optimization algorithm. Knowl.-Based Syst. 232, 107468 , 2021.

[149] Dorgham, O.M., Mohammed Alweshah, M.H., Ryalat, J.A., Khader, M., Alkhalaileh, S.: Monarch butterfly optimization algorithm for computed tomography image segmentation. Multimed. Tools Appl. 80(20), 30057–30090, 2021.

[150] Qi, A., Zhao, D., Fanhua, Yu., Heidari, A.A., Zongda, W., Cai, Z., Alenezi, F., Mansour, R.F., Chen, H., Chen, M.: Directional mutation and crossover boosted ant colony optimization with application to covid-19 x-ray image segmentation. Comput. Biol. Med. 148, 105810, 2022.

[151] Ryalat, M.H., Dorgham, O., Tedmori, S., Al-Rahamneh, Z., Al- Najdawi, N., Mirjalili, S.: Harris hawks optimization for covid- 19 diagnosis based on multi-threshold image segmentation. Neural Comput. Appl. 35(9), 6855–6873, 2023.

[152] Dongmei, W., Yuan, C.: Threshold image segmentation based on improved sparrow search algorithm. Multimed. Tools Appl. 81(23), 33513–33546, 2022.

[153] Dhanachandra, N., Manglem, K., Chanu, Y.J.: Image segmentation using k-means clustering algorithm and subtractive clustering algorithm. Proced. Comput. Sci. 54, 764–771, 2015.

[154] Rastgarpour, M., Shanbehzadeh, J., Soltanian-Zadeh, H.: A hybrid method based on fuzzy clustering and local region-based level set for segmentation of inhomogeneous medical images. J. Med. Syst. 38(8), 1–15, 2014.

[155] Anitha Vishnuvarthanan, M., Rajasekaran, P., Govindaraj, V., Zhang, Y., Thiyagarajan, A.: An automated hybrid approach using clustering and nature inspired optimization technique for improved tumor and tissue segmentation in magnetic resonance brain images. Appl. Soft Comput. 57, 399–426, 2017.

[156] Zexian, F., An, J., Yang, Q., Yuan, H., Sun, Y., Ebrahimian, H.: Skin cancer detection using kernel fuzzy c-means and developed red fox optimization algorithm. Biomed. Signal Process. Control71, 103160 , 2022.

[157] Jain, S., Indora, S., Atal, D.K.: Lung nodule segmentation using Salp shuffled shepherd optimization algorithm-based generative adversarial network. Comput. Biol. Med. 137, 104811, 2021.

[158] Ajai, A.K., Anitha, A.: Clustering based lung lobe segmentation and optimization based lung cancer classification using ct images. Biomed. Signal Process. Control 78, 103986, 2022.

[159] Benaichouche, A.N., Oulhadj, H., Siarry, P.: Improved spatial fuzzy c-means clustering for image segmentation using pso initialization, Mahalanobis distance and post-segmentation correction. Digital Signal Process. 23(5), 1390–1400, 2013.

[160] Ramana Kumari, A., Rao, S.N., Ramana Reddy, P.: Design of hybrid dental caries segmentation and caries detection with meta-heuristic-based resnext-rnn. Biomed. Signal Process. Control 78, 103961, 2022.

[161] Nguyen, U.T.V., Bhuiyan, A., Park, L.A.F., Ramamohanarao, K.: An effective retinal blood vessel segmentation method using multi-scale line detection. Pattern Recogn. 46(3), 703–715, 2013.

[162] Nadipally, M.: Optimization of methods for image-texture segmentation using ant colony optimization. In: Intelligent data analysis for biomedical applications, pages 21–47. Elsevier, 2019.

[163] Mesejo, P., Valsecchi, A., Marrakchi-Kacem, L., Cagnoni, S., Damas, S.: Biomedical image segmentation using geometric deformable models and metaheuristics. Comput. Med. Imaging Graph. 43, 167–178 , 2015.

[164] Sivakumar, V., Janakiraman, N.: A novel method for segmenting brain tumor using modified watershed algorithm in mri image with fpga. Biosystems 198, 104226, 2020.

[165] Li Zhang and Chee Peng Lim: Intelligent optic disc segmentation using improved particle swarm optimization and evolving ensemble models. Appl. Soft Comput. 92, 106328, 2020.

[166] Escorcia-Gutierrez, J., Torrents-Barrena, J., Gamarra, M., Romero-Aroca, P., Valls, A., Puig, D.: A color fusion model based on Markowitz portfolio optimization for optic disc segmentation in retinal images. Expert Syst. Appl. 174, 114697, 2021.

[167] Aghazadeh, N., Moradi, P., Castellano, G., Noras, P.: An automatic mri brain image segmentation technique using edge-region- based level set. J. Supercomput. 79(7), 7337–7359, 2023.

[168] Liu, S., Peng, Y.: A local region-based Chan-Vese model for image segmentation. Pattern Recogn. 45(7), 2769–2779, 2012.

[169] Shi, C., Cheng, Y., Liu, F., Wang, Y., Bai, J., Tamura, S.: A hierarchical local region-based sparse shape composition for liver segmentation in ct scans. Pattern Recogn. 50, 88–106, 2016.

[170] Pham, T.X., Siarry, P., Oulhadj, H.: A multi-objective optimization approach for brain mri segmentation using fuzzy entropy clustering and region-based active contour methods. Magn. Reson. Imaging 61, 41–65 ,2019.

[171] Huang, Q., Bai, X., Li, Y., Jin, L., Li, X.: Optimized graph based segmentation for ultrasound images. Neurocomputing 129, 216–224 ,2014.

[172] marwa obayya , muhammad kashif saeed , nuha alruwais , saud s. alotaibi , mohammed assiri , and ahmed s. salama. Hybrid Metaheuristics with Deep Learning based Fusion Model for Biomedical Image Analysis. Digital Object Identifier 10.1109/ACCESS.2023.3326369, VOLUME 11, 2023.

[173] Malik, S.; Akram, T.; Ashraf, I.; Rafiullah, M.; Ullah, M.; Tanveer, J. A Hybrid Preprocessor DE-ABC for Efficient Skin-Lesion Segmentation with Improved Contrast. Diagnostics12, 2625, 2022. https://doi.org/ 10.3390/diagnostics12112625

[174] Baldeon-Calisto, Maria, and Susana K. Lai-Yuen. ”AdaResU-Net: Multiobjective adaptive convolutional neural network for medical image segmentation.” Neurocomputing 392: 325-340, 2020.

[175] Targ, Sasha, Diogo Almeida, and Kevin Lyman. ”Resnet in resnet: Generalizing residual architectures.” arXiv preprint arXiv:1603.08029 , 2016.

[176] Hassanzadeh, Tahereh, Daryl Essam, and Ruhul Sarker. ”EvoU-Net: An evolutionary deep fully convolutional neural network for medical image segmentation.” Proceedings of the 35th Annual ACM Symposium on Applied Computing. 2020.

[177] Dawid Połap I , Karolina K˛esik, Marcin Wo´zniak, and Robertas Damaševi˘cius, Parallel Technique for the Metaheuristic Algorithms Using Devoted Local Search and Manipulating the Solutions Space, Appl. Sci.8, 293, 2018. doi:10.3390/app8020293

[178] T. Dokeroglu and A. Cosar, “Optimization of one-dimensional bin packing problem with island parallel grouping genetic algorithms,” Computers & Industrial Engineering, vol. 75, pp. 176–186, 2014.

[179] E. Alba and M. Tomassini, “Parallelism and evolutionary algorithms,” IEEE Transactions on Evolutionary Computation, vol. 6, no. 5, pp. 443–462, 2002.

[180] H.-C. Huang, “Fpga-based parallel metaheuristic pso algorithm and its application to global path planning for autonomous robot navigation,” Journal of Intelligent and Robotic Systems, vol. 76, no. 3,pp. 475–488, 2014.

[181] S, . G¨ulc¨u and H. Kodaz, “A novel parallel multi-swarm algorithm based on comprehensive learning particle swarm optimization,” Engineering Applications of Artificial Intelligence, vol. 45, pp. 33–45,2015.

[182] Luong, T. V. Métaheuristique parallèles sur GPU. Ph.D. thesis Ecole doctorale sciences Pour l’ingenieur Universite Lille Nord-de-France, 2011.

[183] Talbi, E. G. Metaheuristics : from design to implementation. John Wiley and Sons, 624 pages, 2009.

[184] Che-Lun Hung, Yuan-Huai Wu. Parallel genetic-based algorithm on multiple embedded graphic processing units for brain magnetic resonance imaging segmentation 2016. Computers and Electrical Engineering, 2016, 1–11, http://dx.doi.org/10.1016/j.compeleceng.2016.09.028.

[185] Laila Almutairi, Ahed Abugabah, Hesham Alhumyani, Ahmed A. Mohamed. Intelligent biomedical image classification in a big data architecture using metaheuristic optimization and gradient approximation. Wireless Networks ,2024, 30:7087–7108, https://doi.org/10.1007/s11276-023-03573-5(0123456789().,-volV)(0123456789,-().volV)

[186] Khalid M. Hosny, Asmaa M. Khalid, Hanaa M. Hamza, SeyedaliMirjalili. Multilevel segmentation of 2D and volumetric medical images using hybrid Coronavirus Optimization Algorithm. Computers in Biology and Medicine ,2022,150, https://doi.org/10.1016/j.compbiomed.2022.106003 .

[187] Keith B. History of Machine Learning. Keith D. Foote blog on March 26, 2019. Available on https://www.dataversity.net/a-brief-history-of-machine-learning. Accessed on December 20, 2020 .

[188]. Russell S., Norvig P. Artificial Intelligence: A Modern Approach (2nd ed.). Prentice Hall. ISBN 978-0137903955, 2003.

[189] Langley, P., Simon, H., Bradshaw, G. and Zytkow, J. Scientific Discovery: Computational Explorations of the Creative Processes. MIT Press, Cambridge, 1987.

[190]. Lee, H., and Choi, B. Knowledge management enablers, processes, and organizational performance: An integrative view and empirical examination. Journal of Management Information Systems, 20(1), 179– 228. 2003.

[191] Sarker IH, Hoque MM, MdK Uddin, Tawfeeq A. Mobile data science and intelligent apps: concepts, ai-based modeling and research directions. Mob Netw Appl, pages 1–19, 2020.

[192] Sarker IH, Kayes ASM, Badsha S, Alqahtani H, Watters P, Ng A. Cybersecurity data science: an overview from machine learning perspective. J Big Data.7(1):1–29, 2020.

[193] Han J, Pei J, Kamber M. Data mining: concepts and techniques. Amsterdam: Elsevier; 2011.

[194] McCallum A. Information extraction: distilling structured data from unstructured text. Queue. 3(9):48–57. 2005.

[195] Moustafa N, Slay J. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). Military communications and information systems conference (MilCIS), 2015;pages 1–6. IEEE, 2015.

[196] Canadian institute of cybersecurity, university of new brunswick, iscx dataset, http:// www. unb. ca/ cic/ datas ets/ index. html/ (Accessed on 20 October 2019).

[197] Cic-ddos2019 [online]. available: https:// www. unb. ca/ cic/ datas ets/ ddos- 2019. html/ (Accessed on 28 March 2020).

[198] Koroniotis N, Moustafa N, Sitnikova E, Turnbull B. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: bot-iot dataset. Fut Gen Comput Syst.100:779–96, 2019.

[199] Santi P, Ram D, Rob C, Nathan E. Behavior-based adaptive call predictor. ACM Trans Auton Adapt Syst. 6(3):21:1–21:28, 2011.

[200] Sarker IH, Colman A, Kabir MA, Han J. Phone call log as a context source to modeling individual user behavior. In: Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (Ubicomp): Adjunct, Germany, pages 630–634. ACM, 2016.

[201] Eagle N, Pentland AS. Reality mining: sensing complex social systems. Person Ubiquit Comput. 10(4):255–68, 2006.

[202] Balducci F, Impedovo D, Pirlo G. Machine learning applications on agricultural datasets for smart farm enhancement. Machines. 6(3):38, 2018.

[203] Khadse V, Mahalle PN, Biraris SV. An empirical comparison of ,supervised machine learning algorithms for internet of things data. In: 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), IEEE. 1–6, 2018.

[204] Zikang H, Yong Y, Guofeng Y, Xinyu Z. Sentiment analysis of agricultural product ecommerce review data based on deep learning. In: 2020 International Conference on Internet of Things and Intelligent Applications (ITIA), IEEE, 2020; pages 1–7

[205] Safdar S, Zafar S, Zafar N, Khan NF. Machine learning based decision support systems (dss) for heart disease diagnosis: a review. Artif Intell Rev.50(4):597–623, 2018.

[206] Perveen S, Shahbaz M, Keshavjee K, Guergachi A. Metabolic syndrome and development of diabetes mellitus: predictive modeling based on machine learning techniques. IEEE Access. 7:1365–75, 2018.

[207] Harmon SA, Sanford TH, Sheng X, Turkbey EB, Roth H, Ziyue X, Yang D, Myronenko A, Anderson V, Amalou A, et al. Artificial intelligence for the detection of covid-19 pneumonia on chest ct using multinational datasets. Nat Commun. 11(1):1–7, 2020.

[208] ALI BOU NASSIF, ISMAIL SHAHIN, IMTINAN ATTILI, MOHAMMAD AZZEH, AND KHALED SHAALAN, Speech Recognition Using Deep Neural Networks: A Systematic Review, Article in IEEE Access · February 2019 DOI: 10.1109/ACCESS.2019.2896880

[209] Mohammed M, Khan MB, Bashier Mohammed BE. Machine learning: algorithms and applications. CRC Press; 2016.

[210] Kaelbling LP, Littman ML, Moore AW. Reinforcement learning: a survey. J Artif Intell Res. 1996;4:237–85.

[211] Y. Cho and L. K. Saul, ``Kernel methods for deep learning," in Proc. Adv. Neural Inf. Process. Syst. (NIPS), vol. 22, 2009, pp. 342_350.

[212] Tavallaee M, Bagheri E, Lu W, Ghorbani AA. A detailed analysis of the kdd cup 99 data set. In. IEEE symposium on computational intelligence for security and defense applications. IEEE.2009:1–6. 2009.

[213] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, et al. Scikit-learn: machine learning in python. J Mach Learn Res. 12:2825–30, 2011.

[214] Witten IH, Frank E. Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, 2005.

[215] Hsu, C. W., Chang, C. C., & Lin, C. J. A practical guide to support vector classification. Technical Report, Department of Computer Science, National Taiwan University. 2010.

[216] Burges, C. J. C. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, 2, 121–167. 1998. DOI:10.1023/A:1009715923555

[217] Quinlan, J. R. "Induction of Decision Trees." Introduces ID3, one of the earliest algorithms for decision tree construction. 1986.

[218] "Random Forests" by Leo Breiman. This seminal paper introduces the Random Forest algorithm, detailing its construction, advantages, and performance metrics. 2001.

[219] "WildWood: A New Random Forest Algorithm" by Gaïffas et al. « Introduces an enhanced Random Forest variant that improves prediction accuracy and computational efficiency » (2021) .

[220] Jehad Ali , Rehanullah Khan , Nasir Ahmad , Imran Maqsood, « Random Forests and Decision Trees », IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 5, No 3, September 2012 ISSN (Online): 1694-0814 www.IJCSI.org

[221] I.Rich , « An empirical study of the naive Bayes classifier », T.J. Watson Research Center, 2001.

[222] Indika Wickramasinghe ,and Harsha Kalutarage, « Naive Bayes: Applications, Variations and Vulnerabilities - A Review of Literature with Code Snippets for Implementation », Methodologies and Application, Volume 25, pages 2277–2293, 2021.

[223] Jalloul, R.; Chethan, H.K.; Alkhatib, R. A Review of Machine Learning Techniques for the Classification and Detection of Breast Cancer from Medical Images. Diagnostics 2023, 13, 2460. https:// doi.org/10.3390/diagnostics13142460

[224] Yongguang Zhai , Zhongyi Qu, and Lei Hao ,Land Cover Classification Using Integrated Spectral, Temporal, and Spatial Features Derived from Remotely Sensed Images, Remote Sens. 2018, 10(3), 383; https://doi.org/10.3390/rs10030383

[225] Kurdi, S.Z.; Ali, M.H.; Jaber, M.M.; Saba, T.; Rehman, A.; Damaševi˘cius, R. Brain Tumor Classification Using Meta-Heuristic Optimized Convolutional Neural Networks. J. Pers. Med. 2023, 13, 181. https://doi.org/10.3390/ jpm13020181

[226]. UCI. Breast Cancer Wisconsin Dataset. Available online: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+ (Diagnostic) (accessed on 9 June 2019).

[227]. Coccia, M. The increasing risk of mortality in breast cancer: A socioeconomic analysis between countries. J. Soc. Adm. Sci. 2019, 6 218–230.

[228] Jalloul, R. Chethan, H.K., Alkhatib, R. A Review of Machine Learning Techniques for the Classification and Detection of Breast Cancer from Medical Images. Diagnostics 2023, 13, 2460. https:// doi.org/10.3390/diagnostics13142460

[229] Jafari, Z.; Karami, E. Breast Cancer Detection in Mammography Images: A CNN-Based Approach with Feature Selection. Information 2023, 14, 410. https://doi.org/ 10.3390/info14070410.

[230] Breastcancer.org. (2023). Types of Breast Cancer. https://www.breastcancer.org

[231] National Cancer Institute. (2022). Inflammatory Breast Cancer. https://www.cancer.gov/types/breast/ibc

[232] Pisano, E. D., et al. Diagnostic performance of digital versus film mammography for breast-cancer screening. NEJM, 353(17), 1773–1783, 2005.

[233] Berg, W. A., et al. Combined screening with ultrasound and mammography vs mammography alone in women at elevated risk of breast cancer. JAMA, 299(18), 2151–2163, 2008.

[234] Mann, R. M., et al. Breast MRI: state of the art. Radiology, 277(3), 520–536, 2015.

[235] Rafferty, E. A., et al. Breast cancer screening using tomosynthesis in combination with digital mammography. JAMA, 309(22), 2343–2349, 2013.

[236] Groheux, D., et al. 18F-FDG PET/CT in staging, restaging, and treatment response assessment of breast cancer. Journal of Nuclear Medicine, 54(9), 1309–1321, 2013.

[237] A. B. Yusuf, R. M. Dima, and S. K. Aina, "Optimized Breast Cancer Classification using Feature Selection and Outliers Detection," Journal of the Nigerian Society of Physical Sciences, vol. 3, no. 4, pp. 298–307, 2021, DOI: 10.46481/jnsps.2021.331.

[238] Yunyun Liu, Azizan As'arry, Mohd Khair Hassan, Abdul Aziz Hairuddin, Hesham Mohamad, Review of the grey wolf optimization algorithm: variants and applications, Neural Computing and Applications (2024) 36:2713–2735, 2024. https://doi.org/10.1007/s00521-023-09202

[239] Gao ZM, Zhao J. An improved grey Wolf optimization algorithm with variable weights. Comput Intell Neurosci. 2019. https:// doi.org/10.1155/2019/2981282

[240] Hu P, Chen S, Huang H, Zhang G, Liu L. Improved alpha-guided grey wolf optimizer. IEEE Access 7:5421–5437. 2019. https://doi.org/10.1109/ACCESS.2018.2889816

[241] Luo K. Enhanced grey wolf optimizer with a model for dynamically estimating the location of the prey. Appl Soft Comput J 77:225–235. 2019.https://doi.org/10.1016/j.asoc.2019.01.025

[242] Gupta S, Deep K. A novel random walk grey wolf optimizer. Swarm Evol Comput 44:101–112. 2019.https://doi.org/10. 1016/j.swevo.2018.01.001

[243] Adhikary J, Acharyya S. randomized balanced grey wolf optimizer (RBGWO) for solving real life optimization problems. Appl Soft Comput. 2022. https://doi.org/10.1016/j.asoc.2022.108429

[244] Emary E, Zawbaa HM, Grosan C. Experienced gray wolf optimization through reinforcement learning and neural networks. IEEE Trans Neural Netw Learn Syst 29(3):681–694. 2018.https://doi.org/10.1109/TNNLS.2016.2634548

[245] Nadimi-Shahraki MH, Taghian S, Mirjalili S. An improved grey wolf optimizer for solving engineering problems. Expert Syst Appl. 2021. https://doi.org/10.1016/j.eswa.2020.113917

[246] Tripathi AK, Sharma K, Bala M. A novel clustering method using enhanced grey wolf optimizer and mapreduce. Big Data Res 14:93–100. 2018.https://doi.org/10.1016/j.bdr.2018.05.002

[247] Barman B, Dewang RK, Mewada A. Facial recognition using grey wolf optimization. Mater Today Proc 58:273–285. 2022. https://doi.org/10.1016/j.matpr.2022.02.161

[248] Martin B, Marot J, Bourennane S. Mixed grey wolf optimizer for the joint denoising and unmixing of multispectral images. Appl Soft Comput J 74:385–410. 2019.https://doi.org/10.1016/j.asoc.2018.10.019

[249] Yu X, Wu X. Ensemble grey wolf optimizer and its application for image segmentation. Expert Syst Appl. 2022. https:// doi.org/10.1016/j.eswa.2022.118267

[250] Gopatoti A, Vijayalakshmi P, CXGNet: a tri-phase chest X-ray image classification for COVID-19 diagnosis using deep CNN with enhanced grey-wolf optimizer. Biomed Signal Process Control. 2022. https://doi.org/10.1016/j.bspc.2022.103860

[251] Karakoyun M, Gu¨lcu¨ S¸ , Kodaz H (2021) D-MOSG: discrete multi-objective shuffled gray wolf optimizer for multi-level image thresholding. Eng Sci Technol Int J 24(6):1455–1466. 2021, https://doi.org/10.1016/j.jestch.2021.03.011

[252] Yu H et al, Image segmentation of leaf spot diseases on maize using multi-stage Cauchy-enabled grey wolf algorithm. Eng Appl Artif Intell. 2022, https://doi.org/10.1016/j.engappai.2021. 104653

[253] Ali M. Alsaqr, "Remarks on the use of Pearson's and Spearman's correlation coefficients in assessing relationships in ophthalmic data," African Vision and Eye Health, vol. 80, no. 1, Article No. 612, 2021, DOI: 10.4102/aveh.v80i1.612.

[254] H. Akoglu, "User's guide to correlation coefficients," Turkish Journal of Emergency Medicine, vol. 18, no. 3, pp. 91–93, 2018, DOI:https://doi.org/10.1016/j.tjem.2018.08.001.

[255] 14 . Martin, D. , Fowlkes, C. , Tal, D. , and Malik, J. (2001). A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In Computer vision, 2001. ICCV 2001. Proceedings. Eighth IEEE international conference on: Vol. 2 (pp. 416–423). IEEE .

[256] Labati RD, Piuri V, Scotti F. All-Idb: the acute lymphoblastic leukemia image database For image processing Ruggero Donida Labati IEEE Member, Vincenzo Piuri IEEE Fellow, Fabio Scotti IEEE Member Universit ' a degli Studi di Milano, Department of Information Technologies,. IEEE Int Conf Image Process 2045–2048. 2011.

[257]. Laiton-Bonadiez, C.; Sanchez-Torres, G.; Branch-Bedoya, J. Deep 3D Neural Network for Brain Structures Segmentation Using Self-Attention Modules in MRI Images. Sensors 2022, 22, 2559.

[258]. Liu, Y.; Unsal, H.S.; Tao, Y.; Zhang, N. Automatic Brain Extraction for Rodent MRI Images. Neuroinformatics 2020, 18, 395–406.

[259]. Kontos, D.; Megalooikonomou, V.; Gee, J.C. Morphometric Analysis of Brain Images with Reduced Number of Statistical Tests: A Study on the Gender-Related Differentiation of the Corpus Callosum. Artif. Intell. Med. 2009, 47, 75–86.

[260]. Munir, K.; Frezza, F.; Rizzi, A. Deep Learning Hybrid Techniques for Brain Tumor Segmentation. Sensors 2022, 22, 8201.

[261]. Widmann, G.; Henninger, B.; Kremser, C.; Jaschke, W. MRI Sequences in Head & Neck Radiology–State of the Art. Fortschr. Rontgenstr. 2017, 189, 413–422.

[262]. Dong, Q.;Welsh, R.C.; Chenevert, T.L.; Carlos, R.C.; Maly-Sundgren, P.; Gomez-Hassan, D.M.; Mukherji, S.K. Clinical Applications of Diffusion Tensor Imaging. Magn. Reson. Imaging 2004, 19, 6–18

[263]. Sun, L.; Zu, C.; Shao, W.; Guang, J.; Zhang, D.; Liu, M. Reliability-Based Robust Multi-Atlas Label Fusion for Brain MRI Segmentation. Artif. Intell. Med. 2019, 96, 12–24.

[264]. Richard, N.; Dojat, M.; Garbay, C. Automated Segmentation of Human Brain MR Images Using a Multi-Agent Approach. Artif.Intell. Med. 2004, 30, 153–176.

[265]. González-Villà, S.; Oliver, A.; Valverde, S.; Wang, L.; Zwiggelaar, R.; Lladó, X. A Review on Brain Structures Segmentation in Magnetic Resonance Imaging. Artif. Intell. Med. 2016, 73, 45–69.

[266]. Cao, R.; Ning, L.; Zhou, C.; Wei, P.; Ding, Y.; Tan, D.; Zheng, C. CFANet: Context Feature Fusion and Attention Mechanism Based Network for Small Target Segmentation in Medical Images. Sensors 2023, 23, 8739.

[267]. Anusooya, G.; Bharathiraja, S.; Mahdal, M.; Sathyarajasekaran, K.; Elangovan, M. Self-Supervised Wavelet-Based AttentionNetwork for Semantic Segmentation of MRI Brain Tumor. Sensors 2023, 23, 2719.

[268]. Lu, F.; Tang, C.; Liu, T.; Zhang, Z.; Li, L. Multi-Attention Segmentation Networks Combined with the Sobel Operator for Medical Images. Sensors 2023, 23, 2546.

[269] FU Hai-Jun, WU Xiao-Hong, MAO Han-Ping, WU Bin,"Fuzzy Entropy Clustering Using Possibilistic Approach", Procedia Engineering 15 (2011) 1993 – 1997, doi:10.1016/j.proeng.2011.08.372

[270] Li R P, Mukaidono M. A Maximum-Entropy Approach to Fuzzy Clustering. International Joint Conference of the Fourth IEEE International Conference on Fuzzy Systems and The Second International Fuzzy Engineering Symposium., Proceedings of 1995 IEEE International Conference on Fuzzy Systems, 1995, Vol 4, pp.2227-2232.

[271] C.A. Cocosco, V. Kollokian, R.K.-S. Kwan, A.C. Evans : "BrainWeb: Online Interface to a 3D MRI Simulated Brain Database"NeuroImage, vol.5, no.4, part 2/4, S425, 1997 -- Proceedings of 3-rd International Conference on Functional Mapping of the Human Brain, Copenhagen, May 1997

[272] Praveen Kumar Ramtekkar, Anjana Pandey, Mahesh Kumar Pawar, "Accurate detection of brain tumor using optimized feature selection based on deep learning techniques", Vol.:(0123456789) Multimedia Tools and Applications, April 2023, https://doi.org/10.1007/s11042-023-15239-7.

[273] Carr, C.; Kitamura, F.; Partridge, G.; Kalpathy-Cramer, J.; Mongan, J.; Andriole, K.; Lavender Vazirabad, M.; Riopel, M.; Ball, R.; Dane, S.; et al. RSNA Screening Mammography Breast Cancer Detection, Kaggle 2022. Available online: https: //kaggle.com/competitions/rsna-breast-cancer-detection (accessed on 1 December 2022).

[274] PrajoonaValsalan, P. Sriramakrishnan, S. Sridhar, G. Charlyn Pushpa Latha, A. Priya, S. Ramkumar, A. Robert Singh, and T. Rajendran, "Knowledge based fuzzy c‐means method for rapid brain tissues segmentation of magnetic resonance imaging scans with CUDA enabled GPU machine", Journal of Ambient Intelligence and Humanized Computing, May 2020, https://doi.org/10.1007/s12652-020-02132-6.

[275] Aniket Pramanik, Xiaodong Wu, and Mathews Jacob.(2023). Joint Calibrationless Reconstruction and Segmentation of Parallel MRI. ieee transactions on medical imaging. ECCV 2022 Workshops, LNCS 13803, pp. 437–453. https://doi.org/10.1007/978-3-031-25066-8_24.

Abstract

In recent years, the integration of distributed systems with parallel processing techniques has significantly advanced the field of image processing. Distributed systems enable the efficient handling of large datasets by dividing the computational tasks across multiple nodes, improving both speed and scalability. Parallelization in image processing enhances performance by executing multiple tasks simultaneously, making it possible to process high-resolution images and complex datasets in real time. Metaheuristic algorithms have been widely adopted for optimization tasks in image processing due to their ability to explore large search spaces effectively. These algorithms, when coupled with machine learning (ML) models, provide powerful solutions for feature selection in classification tasks. Metaheuristics help identify the most relevant features from large datasets, thereby enhancing the classification performance of ML models. Further, parallel metaheuristics, deployed in a distributed environment, can optimize image segmentation processes by splitting the task across multiple computational units, thereby speeding up the process while maintaining or improving segmentation accuracy.

Bearing those in mind, we propose in this thesis, four hybrid methods focusing on feature selection, classification, and parallel image segmentation. The first method employs Grey Wolf Optimization (GWO) for selecting the most relevant features, followed by a Random Forest (RF) classifier to perform accurate classification. The second method integrates Correlation-based filtering with GWO to enhance the feature selection process, and applies various machine learning classifiers for comparative performance analysis. For parallel image segmentation, we designed two parallel approaches to improve efficiency and accuracy. The first is a Parallel Whale Optimization Algorithm (WOA) combined with K-Means clustering, implemented using multiprocessing to accelerate the segmentation process. The second method uses Grey Wolf Optimization in combination with Fuzzy C-Means (FCM), leveraging GPU acceleration for parallel execution. This approach significantly reduces computational time while maintaining high segmentation quality. All methods are evaluated using standard performance metrics. Our aim is to demonstrate the effectiveness of parallel metaheuristics and hybrid selection-classification strategies in medical image analysis.

In conclusion, this thesis shows that combining parallel computing, metaheuristic optimization, and machine learning offers an effective, accurate, and scalable solution to challenges in medical image analysis, contributing to the development of next-generation diagnostic systems.

Keywords: Distributed system, Parallel metaheuristic, Image segmentation, Classification, Grey Wolf Optimizer, Whale Optimizer Algorithm, Multiprocessing, Graphic Processing Unit.

Résumé

Ces dernières années, l'intégration de systèmes distribués avec des techniques de traitement parallèle a considérablement fait progresser le traitement d'images. Ces systèmes permettent de gérer efficacement de grands ensembles de données en répartissant les tâches de calcul sur plusieurs nœuds, améliorant ainsi la vitesse et l'évolutivité. La parallélisation du traitement d'images améliore les performances en exécutant plusieurs tâches simultanément, permettant ainsi de traiter des images haute résolution et des ensembles de données complexes en temps réel. Les algorithmes métaheuristiques ont été largement adoptés pour les tâches d'optimisation en traitement d'images en raison de leur capacité à explorer efficacement de vastes espaces de recherche. Couplés à des modèles d'apprentissage automatique (ML), ces algorithmes offrent des solutions performantes pour la sélection de caractéristiques dans les tâches de classification. Les métaheuristiques aident à identifier les caractéristiques les plus pertinentes parmi de grands ensembles de données, améliorant ainsi les performances de classification des modèles d'apprentissage automatique. De plus, les métaheuristiques parallèles, déployées dans un environnement distribué, peuvent optimiser les processus de segmentation d'images en répartissant la tâche sur plusieurs unités de calcul, accélérant ainsi le processus tout en maintenant, voire en améliorant, la précision de la segmentation.

Dans cette optique, nous proposons dans cette thèse quatre méthodes hybrides axées sur la sélection, la classification et la segmentation d'images parallèles. La première méthode utilise l'optimisation Grey Wolf (GWO) pour sélectionner les caractéristiques les plus pertinentes, suivie d'un classificateur Random Forest (RF) pour une classification précise. La seconde méthode intègre le filtrage par corrélation à GWO pour améliorer le processus de sélection des caractéristiques et applique divers classificateurs d'apprentissage automatique pour une analyse comparative des performances. Pour la segmentation d'images parallèles, nous avons conçu deux approches parallèles afin d'améliorer l'efficacité et la précision. La première est un algorithme d'optimisation Parallel Whale (WOA) combiné à un clustering K-Means, implémenté par multitraitement pour accélérer le processus de segmentation. La seconde méthode utilise l'optimisation Grey Wolf en combinaison avec Fuzzy C-Means (FCM), exploitant l'accélération GPU pour une exécution parallèle. Cette approche réduit considérablement le temps de calcul tout en maintenant une qualité de segmentation élevée. Toutes les méthodes sont évaluées à l'aide d'indicateurs de performance standard. Notre objectif est de démontrer l'efficacité des métaheuristiques parallèles et des stratégies hybrides de sélection-classification dans l'analyse d'images médicales.

En conclusion, cette thèse démontre que la combinaison du calcul parallèle, de l'optimisation métaheuristique et de l'apprentissage automatique offre une solution efficace, précise et évolutive aux défis de l'analyse d'images médicales, contribuant ainsi au développement de systèmes de diagnostic de nouvelle génération.

Mots-clés : Système distribué, Métaheuristique parallèle, Segmentation d'images, Classification, Algorithme d'optimization Grey Wolf, Algorithme d'optimisation Whale, Multitraitement, Graphic Processing Unit.

Acknowledgement

I would like to express my special appreciation and thanks to my supervisors Dr. DEBAKLA Mohammed and Prof. KHALIFA Djemal, not only for their tremendous support to my thesis writing, but also for the enlightenment on academic thinking which is the lifelong benefit. Without their consistent and illuminating instruction, this thesis could not have reached its present form.

I also want to say thanks to Prof. REBBAH Mohammed, Prof. SMAIL Omar Dr. MEKKAOUI Kheireddine and Prof. SALEM Mohammed, who gave me plenty of precious advice on my research.

Last, but not least, I appreciate the support of my family during my PH.D studies. My family has been a source of inspiration and strength for me whenever necessary, she has continuously provided me with her moral, spiritual, emotional, and financial support. Thanks to my brothers, sisters, relatives, mentors, friends, and classmates who expressed to me warm words of advice and encouragement to finish this study. I would also like to express my gratitude to anyone who has contributed, near or far, to the completion of this thesis.

Contents

Chapter 1 : Distributed system for image processing.

Chapter 2 : Metaheuristic for image processing

Chapter 3 : Machine learning and metaheuristic for image processing

Chapter 4 : Grey wolf optimizer for breast cancer classification

List of Figures

List of Tables

General Introduction

Image processing plays a fundamental role in a wide range of applications, including medical imaging, remote sensing, object recognition, and pattern analysis. Traditional image processing techniques often rely on handcrafted features and predefined algorithms, which struggle when confronted with complex patterns, noise, and variability in real-world images. With the emergence of machine learning (ML), particularly deep learning, the field has witnessed substantial advancements. ML models now enable automated, adaptive, and highly accurate image analysis by learning intricate patterns from large datasets. Techniques such as Convolutional Neural Networks (CNNs) [1], Support Vector Machines (SVMs)[2], and clustering algorithms like K-Means [3] and Fuzzy C-Means (FCM) [4] have demonstrated strong performance in tasks such as image segmentation, classification, and feature selection [5]-[10].

However, achieving optimal performance with ML techniques often requires careful selection of parameters, efficient feature extraction, and well-structured clustering, tasks that are inherently challenging due to high-dimensional data and the presence of irrelevant or redundant information. To address these challenges, metaheuristic algorithms have emerged as powerful tools for optimizing ML models. Inspired by natural and biological systems, metaheuristics like Grey Wolf Optimizer (GWO) [11], Ant Lion Optimizer (ALO) [12], Genetic Algorithms (GA)[13], and Whale Optimization Algorithm (WOA)[14] provide adaptive and robust search mechanisms that outperform traditional optimization methods in exploring complex, high-dimensional solution spaces. Their application in image processing has been particularly impactful in feature selection, hyperparameter tuning, and neural architecture optimization, especially in critical areas such as medical imaging where precision is vital.

Despite these advantages, one major limitation of metaheuristic algorithms is their computational cost, particularly when applied to large-scale image datasets or high-resolution medical scans. To overcome this bottleneck, parallel and distributed computing strategies have been increasingly adopted [15][16][17]. By leveraging multi-core processors, high-performance computing (HPC) infrastructures, and Graphics Processing Units (GPUs), researchers have been able to accelerate metaheuristic-driven image processing significantly. Parallel metaheuristics including parallelized swarm intelligence models, distributed genetic algorithms, and GPU-accelerated GWO or WOA have shown promising results in reducing execution time and enhancing scalability. In the context of medical image segmentation, this enables real-time or near real-time analysis of MRI or CT scans, supporting faster and more accurate diagnostic decisions.

By integrating ML with metaheuristics and leveraging the power of parallel computing, hybrid systems can achieve superior performance in image processing. These approaches are particularly beneficial in domains requiring high accuracy, robustness, and computational efficiency, such as tumor detection, organ segmentation, and disease classification in medical imaging.

Problem Statement

While traditional and metaheuristic-enhanced ML techniques have shown significant promise in medical image analysis, their scalability remains limited by computational constraints. Most segmentation and feature selection methods are implemented sequentially, which

becomes a performance bottleneck when applied to large datasets or high-resolution medical images. Moreover, current literature lacks comprehensive frameworks that integrate metaheuristic optimization, machine learning, and parallel computing to deliver both high accuracy and computational efficiency. This thesis addresses the need for such integrated and scalable solutions for feature selection, classification, and segmentation in medical imaging.

Objectives of the Thesis

This thesis aims to design, develop, and evaluate distributed and parallel approaches for medical image segmentation and classification based on metaheuristic algorithms. The main objectives are:

1. To study and implement metaheuristic algorithms for feature selection and image segmentation.
2. To integrate machine learning techniques with metaheuristic optimization to improve breast cancer classification.
3. To design and implement parallel and distributed versions of segmentation algorithms using multicore CPUs and GPUs.
4. To evaluate the effectiveness of the proposed methods in terms of segmentation accuracy, classification performance, and computational speed.

Major Contributions

The thesis makes the following contributions:

- **Feature Selection and Breast Cancer Classification Using Metaheuristics**
  Two feature selection methods are proposed based on the Grey Wolf Optimizer (GWO):
  1. A GWO-based feature selection combined with Random Forest (RF) classifier, applied to the WDBC dataset, which improves classification accuracy by selecting the most relevant features [18].
  2. A hybrid method that integrates Correlation-based Feature Selection (CFS) with GWO, followed by classification using Support Vector Machine (SVM), RF, and Naïve Bayes (NB)[19].

  These methods are validated through experimental results and published in international conferences and journals.

- **Parallel Image Segmentation Based on Metaheuristics**

  Two parallel segmentation frameworks are proposed:

  1. A multiprocessing-based approach that combines the Whale Optimization Algorithm (WOA) with K-means clustering, implemented on multicore CPUs to accelerate computation. This method was validated through experimental results and published in international conferences
  2. A GPU-accelerated segmentation approach named P-GWO-FCM, which parallelizes both GWO optimization and Fuzzy C-Means (FCM) clustering for

fast and accurate MRI segmentation.This GPU-based method is submitted to an international journal for publication.

Thesis structure

This dissertation is structured into five chapters:

- **Chapter 1** provides a general overview of image processing concepts and distributed systems, including a state-of-the-art review of distributed architectures in image processing.
- **Chapter 2** discusses metaheuristic algorithms in image processing, their optimization role in tasks such as segmentation and classification, and their parallelization strategies. It concludes with a review of parallel metaheuristic-based image segmentation techniques.
- **Chapter 3** focuses on the integration of machine learning and metaheuristics in image analysis, exploring common classifiers (SVM, RF, NB) and their synergy with metaheuristics for feature extraction and selection.
- **Chapter 4** presents two GWO-based feature selection methods for breast cancer classification, along with experimental validation using various machine learning classifiers.
- **Chapter 5** details two parallel segmentation methods: a multiprocessing-based WOA-KMeans hybrid algorithm and a GPU-accelerated P-GWO-FCM approach for MRI segmentation, highlighting performance improvements in accuracy and speed.

Chapter 1

Distributed system for image processing

1. Introduction

Image processing is a core field in computer science and engineering that focuses on performing operations on images to enhance, analyze, or extract meaningful information from them. It plays a critical role in various applications such as medical imaging, satellite data analysis, security and surveillance, industrial inspection, and machine vision. Image processing tasks include fundamental operations such as filtering, edge detection, segmentation, feature extraction, and classification. As image resolutions and dataset sizes have increased, especially with the advent of high-definition and hyperspectral imaging, processing such data in real-time or within a reasonable time frame has become a computationally demanding task. Traditional sequential processing methods often fail to meet these requirements, particularly when applied to large-scale or high-throughput image processing scenarios.

Distributed systems refer to a collection of independent computers that work together as a cohesive system to achieve a common goal. These computers, or nodes, communicate and coordinate their actions by passing messages over a network. The fundamental features of distributed systems include resource sharing, concurrency, fault tolerance, and scalability. In contrast to centralized systems, distributed systems offer improved performance and reliability by distributing workloads across multiple machines. They are widely adopted in modern computing environments, including cloud computing, edge computing, and high-performance computing clusters, where they support complex data-driven tasks such as real-time analytics, artificial intelligence, and large-scale simulations. Their ability to divide and parallelize tasks makes distributed systems ideal for solving problems that are too large or too slow to be addressed by a single machine.

In this chapter, we begin by introducing the fundamental concepts of image processing, with a particular focus on the need for parallelization. We then discuss distributed systems as a powerful platform for enabling parallel image processing. Finally, we review the existing literature on parallel and distributed approaches to image processing

2. Image representation

An image is a visual representation of an object, scene, or concept, typically captured or created using cameras, sensors, or graphic software. Images can be digital or analog, and they contain essential visual information that can be processed [20], analyzed, and interpreted by both humans and machines. In digital imaging, an image is composed of pixels, which are the

smallest units of a picture. Each pixel carries color and intensity information, forming a complete visual representation when arranged in a grid. Images can be categorized into various types, such as grayscale, binary, color, and multispectral images, depending on their composition and the number of color channels they contain. The definition and quality of an image depend on several factors, including resolution**,** bit depth, and compression methods. High-definition images contain more pixels and finer details, while low-definition images may appear pixelated or blurry. With advancements in technology, images are widely used in fields such as medical imaging, computer vision, remote sensing, and artificial intelligence, where they are analyzed for pattern recognition, segmentation, and classification.

An image is defined as a two-dimensional function, F(x,y), where x and y are spatial coordinates, and the amplitude of F at any pair of coordinates (x,y) is called the intensity of that image at that point. When x, y, and amplitude values of F are finite, we call it a digital image. In other words, an image can be defined by a two-dimensional array specifically arranged in rows and columns [20]. Digital Image is composed of a finite number of elements, each of which elements have a particular value at a particular location.These elements are referred to as picture elements, image elements, and pixels. A Pixel is most widely used to denote the elements of a Digital Image. There are four types of images:

1. Binary image: The binary image as its name suggests, contain only two pixel elements i.e 0 and 1, where 0 refers to black and 1 refers to white. This image is also known as Monochrome.
2. 8 bit color format: It is the most famous image format. It has 256 different shades of colors in it and commonly known as Grayscale Image. In this format, 0 stands for Black, and 255 stands for white, and 127 stands for gray.
3. 16 bit color format: It is a color image format. It has 65,536 different colors in it. It is also known as High Color Format. In this format the distribution of color is not as same as Grayscale image.A 16 bit format is actually divided into three further formats which are Red, Green and Blue. That famous RGB format.

3. Image processing

Image processing is the technique of using computational methods to manipulate, analyze, and interpret visual information in digital images. It involves a series of operations to enhance image quality, extract meaningful features, segment objects, and derive useful insights from visual data. In essence, image processing transforms raw image data (captured from sensors or cameras) into a form that is easier to analyze or visually appealing, enabling tasks such as object recognition, pattern detection, and scene interpretation. Applications span across fields such as medical imaging, computer vision, robotics, remote sensing, and digital photography.
Image Processing refers to the manipulation and analysis of digital images using algorithms and techniques to extract, enhance, or modify information from them. It involves applying mathematical and computational methods to improve the visual appearance of images or to extract useful features for tasks such as recognition, detection, and classification. The goal of image processing can be to improve image quality, enhance specific features, or prepare the

image for further analysis in fields such as medical imaging, computer vision, and remote sensing [20].

3.1. Image acquisition

Image acquisition is the process of capturing or obtaining a digital image from a physical scene [21], usually through a camera, scanner, or specialized sensor. This is the first step in the image processing workflow, where real-world visual information is converted into a digital format for analysis and manipulation.In image acquisition, devices may capture various types of images, such as grayscale, color, infrared, or multispectral, depending on the application. The quality and resolution of the acquired image play a significant role in the effectiveness of subsequent image processing tasks.

3.2. Image preprocessing

There are several methods commonly used for image preprocessing to enhance image quality and prepare data for analysis.. Each method plays a vital role in improving the performance of subsequent image processing or machine learning tasks.

3.2.1. Resizing

Resizing an image is the process of changing its dimensions, either width, height, or both while maintaining or adjusting the original aspect ratio. This operation involves either reducing (downsampling) or increasing (upsampling) the number of pixels in the image, which affects its resolution and quality.

3.2.2. Noise reduction

Noise reduction in image segmentation involves preprocessing an image to remove or minimize unwanted artifacts (noise) that can interfere with accurate segmentation. Noise can obscure boundaries, alter pixel values, and generally reduce the quality of segmentation results. Effective noise reduction ensures that the segmented regions correspond accurately to real structures or objects in the image, leading to clearer and more reliable segmentation (Gaussian Blur, Median Filter, Bilateral Filter,Non-Local Means, Wavelet Denoising, Anisotropic Diffusion (Perona-Malik Filtering)).

3.2.3. Normalization

Image normalization is the process of adjusting the range of pixel intensity values in an image to a standard scale. This technique enhances contrast and prepares images for more consistent and accurate analysis, especially in tasks like image classification, segmentation, and feature extraction.

3.2.4. Binarization

Binarization is the process of converting a grayscale or color image into a binary image, where each pixel is assigned one of two values: typically 0 (black) or 1 (white). This technique is commonly used in image processing to simplify images, making objects and backgrounds easier to distinguish by reducing the image to only two intensity levels (Global Thresholding, Adaptive Thresholding, Sauvola and Niblack Methods, Iterative and K-Means Thresholding, Deep Learning-Based Binarization).

3.2.5. Contrast enhancement

Contrast enhancement is an image preprocessing technique used to improve the visibility of features in an image by expanding the range of intensity values. Enhanced contrast makes objects, edges, and details more distinguishable, which is particularly valuable for applications in segmentation, object detection, and feature extraction (Histogram Equalization, Adaptive Histogram Equalization, Linear Contrast Stretching (Normalization), Gamma Correction…).

3.3. Image segmentation

Image segmentation is a fundamental technique in digital image processing and computer vision. It involves partitioning a digital image into multiple segments (regions or objects) to simplify and analyze an image by separating it into meaningful components, which makes the image processing more efficient by focusing on specific regions of interest [22]. Figure 1 represents image segmentation methods.

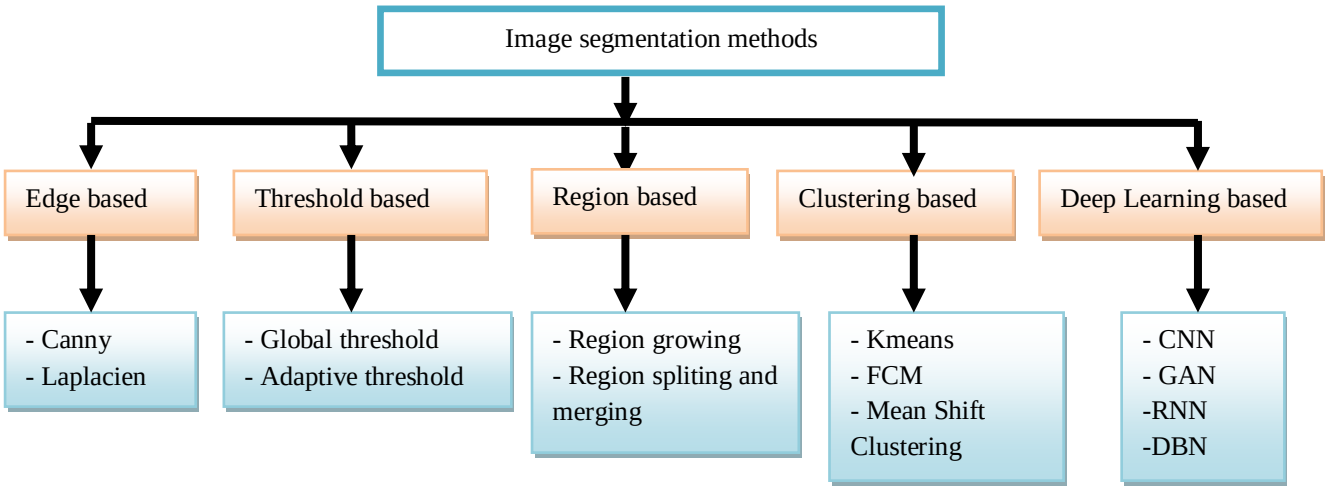


Figure 1. Image segmentation methods.

3.3.1. Edge based image segmentation

Edge-based image segmentation is a technique that divides an image into regions based on the detection of edges, which are the boundaries between different objects or regions within the image. An edge is a significant change in pixel intensity, often marking a shift in texture, color, or brightness. By identifying these boundaries, edge-based segmentation can outline objects or distinct regions effectively. This method usually involves applying edge-detection

algorithms such as Canny [23], or Laplacian of Gaussian [24] to highlight edges, which are then used to segment the image [25]. Edge-based segmentation is widely used in applications where clear boundaries are essential, such as medical imaging, object detection, and feature extraction.An example of edge-based segment is shown in Figure 2.

(a) (b)

Figure 2. Edge-based segment (a) original image, (b) the resulting image after segmentation.

3.3.2. Threshold based segmentation

Threshold-based segmentation is an image segmentation technique that partitions an image by grouping pixels based on their intensity values. It operates by selecting one or more threshold values to separate objects from the background or different regions within the image. Pixels with intensity values above a set threshold might represent objects, while those below might represent the background. This approach works well when there's a clear contrast between objects and the background.In literature there are several types of thresholding [26], Including Global Thresholding (Uses a single threshold value for the entire image, ideal for uniformly lit images), Adaptive Thresholding [27] (Calculates threshold values for smaller regions, which is useful for images with uneven lighting), and Otsu's Method [28] (An automated global thresholding technique that calculates an optimal threshold by minimizing the variance within regions).Threshold-based segmentation is widely used in various domain such as Medical imaging, to isolate organs or tissues, Document processing, to separate text from the background, and Industrial inspection, for defect detection.

For example: we have threshold level 128 so that it decide that all the pixels having intensity value greater than 128, it belong to some regions and those intensity values less than 128, it belong to some other region. Let an image be f(x, y). Suppose that this image consists the dark object against the bright background or viceversa.Therefore, intensity concentrate mainly on two regions,one towards the darker side (or lower intensity) and othertowards the brighter side (or higher intensity). The histogram with two peaks and valley at the bottom as shown in figure 3. Where T is called the threshold value. Figure 4, present leukemia image

segmentation using multi-thresholding method, where (a) represent the original image, and (b) represent the segmented image.

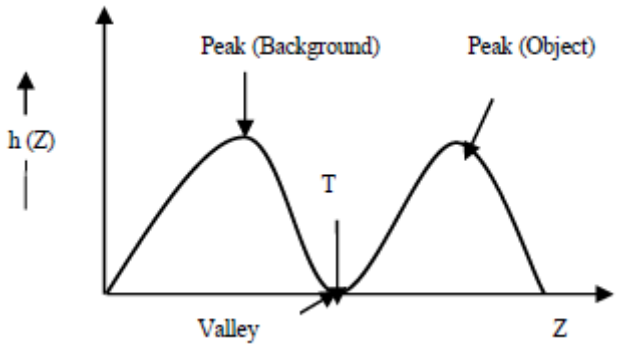
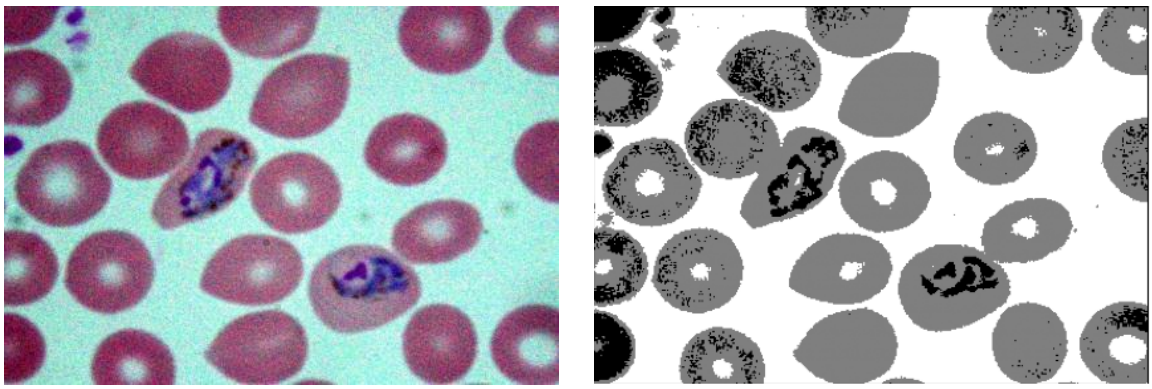


Figure 3. Histogram with two peaks and one valley [29].



(a) (b)

Figure 4. Leukemia image segmentation using multi-thresholding method, where (a) represent the original image, (b) the segmented image.

3.3.3. Region based image segmentation

Region-based image segmentation is a technique that segments an image by grouping neighboring pixels into regions based on predefined criteria, such as similarity in intensity, color, or texture. This approach assumes that pixels within a particular region are more similar to each other than to those in other regions. Region-based segmentation is widely used for medical imaging, such as segmenting organs or lesions. Satellite and aerial imagery, for analyzing terrain or vegetation. And Object detection in complex images. Figure 5, represent a region-based segmentation where (a) original image, (b) the resulting image after segmentationthe segmentation process.Below are some foundational methods in region-based segmentation:

- Region Growing [30]: Region growing starts with seed points and expands regions by adding neighboring pixels with similar properties until no more pixels meet the

similarity criterion. This method is intuitive and effective but can be sensitive to noise and initial seed selection.

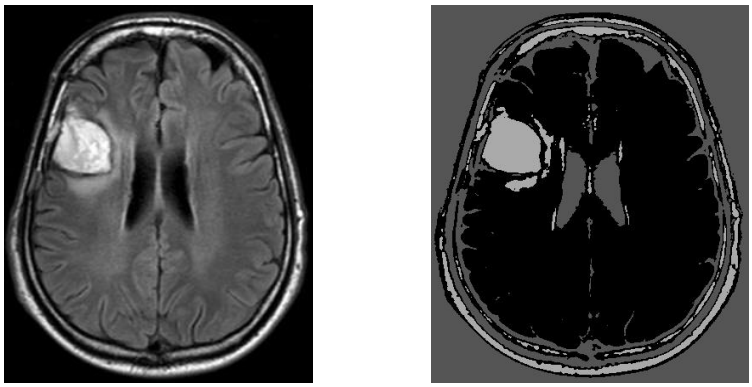
- Watershed Segmentation (Region-Based) [31]: A popular technique for separating touching objects by viewing image intensities as topographical surfaces. Watershed segmentation identifies “watershed lines” as boundaries between regions.
- Region Splitting and Merging [32]: This method recursively splits an image into regions based on homogeneity and merges similar adjacent regions, ensuring each final segment is homogenous.
- Markov Random Fields (MRF) [33]: MRF is a probabilistic model where each pixel’s label depends on neighboring labels, effectively capturing spatial dependencies. This model is commonly used in medical imaging.

Figure 5. Region-based segmentation, (a) original image, (b) the resulting image after segmentation [34].

3.3.4. Clustering based image segmentation

Clustering-based image segmentation is a technique that divides an image into segments by grouping pixels into clusters based on their similarity in features like color, intensity, or texture. This unsupervised approach relies on clustering algorithms to categorize pixels so that similar pixels belong to the same segment, while dissimilar ones are placed in separate segments. Clustering-based segmentation is particularly effective when there’s no prior knowledge of the image’s content or the number of objects present.

- K-Means Clustering [3]: One of the most popular clustering algorithms, K-Means groups pixels based on similarity in feature space (e.g., color or intensity) by iteratively updating cluster centroids. It is simple and effective for basic image segmentation tasks. Clustering-based segmentation is widely used for: Medical imaging, to cluster similar tissues or structures. Remote sensing, to classify land use or vegetation types. Image compression, to group similar pixels and reduce data size. Figure 6 show an exemple of segmentation using kmeans with number of cluster k=3



(a) (b)

Figure 6. Image segmentation using kmeans where (a) represent the original image and (b) represent the segmented image (number of cluster k=3).

- Fuzzy C-Means (FCM) Clustering [4]: An extension of K-Means, FCM allows partial membership of pixels to multiple clusters, which makes it more robust to noise and better suited for images with fuzzy boundaries.
- Mean Shift Clustering [35]: Mean Shift is a non-parametric clustering technique that identifies clusters by locating peaks in a density function. It's particularly useful for segmenting images with complex or multi-modal distributions.

3.3.5. Deep learning based image segmentation methods

Deep Learning (DL) has significantly advanced the field of image segmentation by enabling automatic, highly accurate pixel-wise classification of images [1], [36], [37]. Unlike traditional segmentation methods that rely on manual feature extraction or predefined rules, deep learning models, particularly Convolutional Neural Networks (CNNs), can learn complex, hierarchical features directly from the raw image data. This ability to learn spatial patterns and textures has made DL methods the go-to approach for many segmentation tasks, such as object detection, medical image analysis, and remote sensing. Architectures like U-Net [1], Fully Convolutional Networks (FCNs) [38], and Mask R-CNN have revolutionized segmentation by improving accuracy, robustness, and the ability to segment complex structures. Deep learning methods, especially those leveraging large labeled datasets and advanced architectures, are capable of generalizing well to a wide range of segmentation tasks, offering solutions that can adapt to new domains with minimal human intervention. As a result, DL has become indispensable in both research and industry for tasks requiring precise segmentation, even in challenging or noisy environments.

3.3.5.1. Convolutional Neural Networks (CNNs) for image segmentation

Convolutional Neural Networks (CNNs) are a class of deep learning algorithms designed to process and analyze visual data, particularly images. CNNs use convolutional layers to automatically detect patterns, features, and hierarchical structures in images, making them highly effective for image segmentation tasks. In image segmentation, CNNs classify each pixel in an image, assigning it a specific label (e.g., background or foreground), which

allows for precise segmentation of objects or regions in the image. In following an Example in Segmentation:

- U-Net [1]: A CNN architecture commonly used for semantic segmentation, where the image is divided into regions corresponding to different classes (such as foreground and background).
- Fully Convolutional Networks (FCNs) [38]: An extension of CNNs where the fully connected layers are replaced with convolutional layers to allow for pixel-wise classification in segmentation tasks.

3.3.5.2. Generative Adversarial Networks (GANs) for Image Segmentation

Generative Adversarial Networks (GANs) [39] consist of two neural networks: a generator and a discriminator. The generator creates synthetic data (such as segmented images), while the discriminator evaluates the authenticity of the generated data against real data. In the context of image segmentation, GANs can be used to generate high-quality segmentation masks or to improve segmentation performance through adversarial training. The generator tries to improve the segmentation mask it produces, while the discriminator ensures the result is realistic and accurate, pushing the model toward better segmentation outcomes. In following an Example in Segmentation [40]**:**

- Pix2Pix: A GAN-based model where the generator generates segmentation maps from input images, and the discriminator ensures that the generated segmentation masks are realistic.
- CycleGAN: Used in unpaired image-to-image translation tasks, such as generating segmentation masks without needing paired training data.

3.3.5.3. Recurrent Neural Networks (RNNs) for Image Segmentation

Recurrent Neural Networks (RNNs) are a class of neural networks designed to model sequential data by maintaining a memory of previous states [41]. While RNNs are traditionally used in tasks like natural language processing, they have been applied to image segmentation when there is temporal or spatial context that needs to be captured over a sequence of frames or pixels. For example, RNNs can be used in video segmentation, where each frame's segmentation can be influenced by previous frames or in situations where pixel dependencies (such as neighboring pixels) are crucial to achieving accurate segmentation. In following an Example in Segmentation:

- Long Short-Term Memory (LSTM) [42]: An advanced type of RNN that can capture long-range dependencies, making it effective for segmenting sequential images, video frames, or spatially dependent pixel information in images.
- CRNNs (Convolutional RNNs): Combining CNNs for feature extraction with RNNs for sequence modeling, particularly useful for sequential image segmentation tasks.

3.3.5.4. Deep Belief Networks (DBNs) for Image Segmentation

Deep Belief Networks (DBNs) [43] are a type of probabilistic generative model made up of multiple layers of stochastic, latent variables. They are composed of multiple Restricted Boltzmann Machines (RBMs) stacked together, where each RBM learns to model the data at a different level of abstraction. In the context of image segmentation, DBNs can be used to learn complex, hierarchical representations of the image, which can then be applied to segment objects or regions in the image. DBNs are typically used in scenarios where unsupervised feature learning is necessary before applying a discriminative model for segmentation.In following an Example in Segmentation:

- Feature Learning with DBNs: DBNs can be trained on image data to learn features that are then used to improve the segmentation of objects in images, particularly when there is limited labeled data available.
- DBN + SVM: A hybrid model where the DBN is used for unsupervised feature learning and the learned features are classified using a Support Vector Machine (SVM) to perform segmentation.

3.4. Feature extraction

Feature extraction is the process of transforming raw data into a set of measurable characteristics (features) that can be used to represent the essential aspects of that data. In the context of image processing, feature extraction involves identifying and isolating relevant information (such as color, texture, shape, or edges) from an image to simplify further analysis or machine learning tasks. The goal of feature extraction is to reduce the complexity of the data while preserving meaningful patterns that can aid in tasks like classification, recognition, and segmentation. Effective feature extraction enhances a model's accuracy, speeds up processing, and can improve its ability to generalize to new data. Feature extraction is used for in several domain like image classification, object detection, medical imaging, and facial recognition. In littérature, there are various techniques for feature extraction such as Gray-Level Co-occurrence Matrix (GLCM)[44], Local Binary Patterns (LBP)[45], Histogram of Oriented Gradients (HOG)[46], and Principal Component Analysis (PCA)[47].

3.5. Feature selection

Feature selection is a crucial preprocessing step in machine learning that aims to identify the most relevant and informative features from high-dimensional datasets while eliminating redundant or irrelevant ones. By selecting the optimal subset of features, feature selection improves model performance, reduces computational cost, and enhances interpretability. This process is particularly important in applications such as medical diagnostics, image processing, text classification, and bioinformatics, where datasets often contain hundreds or thousands of features, many of which may be noisy or non-contributory. Feature selection techniques can generally be categorized into three main approaches: filter methods, wrapper

methods, and embedded methods. Researchers have extensively explored a range of feature selection methods through various studies, as evidenced in the literature [5-10][48].

3.5.1. Filter Methods

Filter methods evaluate the importance of features based on statistical techniques without involving any specific machine learning model. These methods assess feature relevance by computing correlation coefficients, information gain, mutual information, or statistical tests. Popular filter techniques include:

- Correlation-based Feature Selection (CFS): Measures the correlation between input features and the target variable, removing highly correlated redundant features.
- Chi-Square Test: Evaluates the dependency between categorical features and the target class, commonly used in text classification.
- Information Gain (IG): Determines the amount of information a feature contributes to the class label, widely applied in decision trees.
- Principal Component Analysis (PCA): Although technically a dimensionality reduction technique rather than a selection method, PCA transforms features into uncorrelated components while retaining most of the data variance.

Filter methods are computationally efficient and work well for high-dimensional datasets, but they may not always select the optimal feature subset for a specific learning algorithm.

3.5.2. Wrapper Methods

Wrapper methods evaluate subsets of features by training and testing a machine learning model iteratively, selecting the subset that yields the best performance. These methods use search strategies such as forward selection, backward elimination, and recursive feature elimination (RFE). Common wrapper techniques include:

- Sequential Forward Selection (SFS): Starts with an empty feature set and iteratively adds features that improve model accuracy.
- Sequential Backward Selection (SBS): Begins with all features and removes the least significant ones step by step.
- Recursive Feature Elimination (RFE): Uses a model (e.g., SVM or Random Forest) to rank features and remove the least important ones iteratively.

Wrapper methods often achieve high accuracy but can be computationally expensive, especially for large datasets.

3.5.3. Embedded Methods

Embedded methods incorporate feature selection directly into the training process of machine learning models. These methods leverage regularization techniques to penalize less

important features, leading to automatic selection. Some widely used embedded techniques are:

- LASSO (Least Absolute Shrinkage and Selection Operator): A regression-based method that applies L1 regularization to shrink the coefficients of less significant features to zero, effectively removing them.
- Decision Tree-Based Methods: Tree algorithms like Random Forest and Gradient Boosting assign importance scores to features and allow pruning of irrelevant ones.
- Elastic Net: Combines LASSO (L1 regularization) and Ridge Regression (L2 regularization) to enhance feature selection in high-dimensional data.

Embedded methods strike a balance between efficiency and accuracy, making them suitable for real-world applications like medical diagnosis and fraud detection.

3.6. Classification

Image classification is a critical step in image processing where an algorithm assigns a category or label to an image based on its visual characteristics. It is widely used in various fields, including medical imaging, autonomous driving, security surveillance, and satellite imagery analysis. The goal of classification is to enable computers to recognize patterns and categorize images into predefined classes. This process typically involves extracting features from images and using machine learning or deep learning models to analyze and classify them [49].

4. Distributed systems

Parallelization refers to the process of breaking down a computational task into smaller sub-tasks that can be executed simultaneously across multiple processing units [50]. This process significantly accelerates computations, especially when handling large data sets or performing complex operations, such as scientific simulations, machine learning, and image processing. Parallelization can occur at various levels [50], including task-level parallelism, where distinct tasks are executed simultaneously, and data-level parallelism, where individual elements of a data set are processed concurrently. The primary advantage of parallelization is the significant reduction in processing time, enabling systems to handle complex computations more efficiently. In addition, parallel systems are scalable, allowing for greater computational power as more processors or cores are added. By utilizing available resources effectively, parallelization enhances overall system efficiency and can also provide fault tolerance by ensuring that tasks are redundantly executed across different processors. However, parallelization comes with limitations, such as the complexity of designing parallel algorithms, which may require careful consideration of dependencies between tasks and ensuring efficient synchronization and communication between processors. Additionally, the overhead from inter-process communication and the diminishing returns seen in performance when scaling up (due to communication bottlenecks and memory access limitations) can limit the effectiveness of parallel systems. The design of parallel algorithms must also take into

account data dependencies, as certain tasks are inherently sequential and cannot be parallelized.

A distributed system is a collection of independent computers that work together to achieve a common goal. These systems enable resource sharing, parallel processing, and fault tolerance by distributing tasks among multiple computing nodes. Unlike centralized systems, where a single machine handles all tasks, distributed systems improve scalability, performance, and reliability by coordinating multiple processors. They are used in various applications such as cloud computing, scientific simulations, big data analytics, and artificial intelligence. However, distributed systems also present challenges, such as maintaining consistency, managing network communication, and handling failures effectively.

Distributed systems operate by enabling multiple computing units, known as nodes, to communicate and collaborate over a network to achieve a common goal. Each node processes a portion of a task, and the system ensures synchronization, consistency, and coordination among them. Communication typically occurs through message passing or shared memory, allowing data to be exchanged efficiently. Middleware acts as a bridge between nodes, managing task allocation, load balancing, and failure recovery. One of the major advantages of distributed systems is their ability to provide high availability and fault tolerance, if one node fails, others can take over. However, a key challenge lies in maintaining consistency and synchronization across multiple nodes, as network failures and data replication conflicts can lead to system inconsistencies.

4.1. Flynn’s classification of parallel machines

There are several ways to classify parallel machines[50]. However, one classification has been widely used since 1966, namely Flynn's Taxonomy [51]. This classification distinguishes parallel architectures based on two independent parameters: instructions and data: each of these two parameters can have two possible states: Single or Multiple. Table 1 illustrates Flynn's classification.

	Single Data	Multiple Data
Single Instruction	SISD	SIMD
Multiple Instruction	MISD	MIMD

Table 1. Flynn classification of parallel machine.

Single instruction, single data (SISD)

A sequential machine that can execute only a single instruction stream in a single CPU clock cycle. Furthermore, only one data stream is used as input per clock cycle. Program execution is deterministic, and it is the oldest and most widespread type of machine today.

Single instruction, multiple data (SIMD)

This is a type of parallel machine whose processors execute the same instruction in a given clock cycle. However, each processing unit can operate on a different data element. This type of machine is well-suited for regular problems such as image processing and graphics

rendering. Program execution is synchronous and deterministic. Furthermore, the majority of current workstation processors and graphics processing units include a specialized SIMD processing unit, known as SWAR (SIMD Within A Register).

Multiple instruction, single data (MISD)

A single data stream feeds multiple processing units, and each processing unit operates on the data independently using a stream of independent instructions. Hennessy and Patterson in [52] state that no such machines have been designed, while Flynn in his 1996 article [53] classifies systolic architectures [54] in this category.

Multiple instruction, multiple data (MIMD)

This is currently the most common type of parallel machine. Each processor in these machines can execute a different instruction stream and operate on a different data stream. Execution can be synchronous or asynchronous, deterministic or nondeterministic. Examples include current supercomputers, networked clusters of parallel machines, computing grids, Symmetric Multi-Processors (SMPs), and multi-core processors. In addition, many of these machines contain SIMD processing units.

4.2. Memory Architectures of Parallel Machines

In the following, we classify parallel machines according to the type of their memory hierarchy[50]. This classification allows us to distinguish parallel machines from a perspective other than that of the CPU and also provides a better understanding of the motivations behind programming models for parallel machines.

4.2.1. Shared Memory Parallel Machines

There are several variants of these machines, but they all share a common property: the ability for all processors to access memory as a global address space. Thus, multiple processors can operate independently but share the same memory resource. A change made by one processor to a memory location is visible to all other processors. This class of machines can be divided into two subclasses based on memory access times: UMA (Figure 7) and NUMA (Figure 8).

Uniform Memory Access

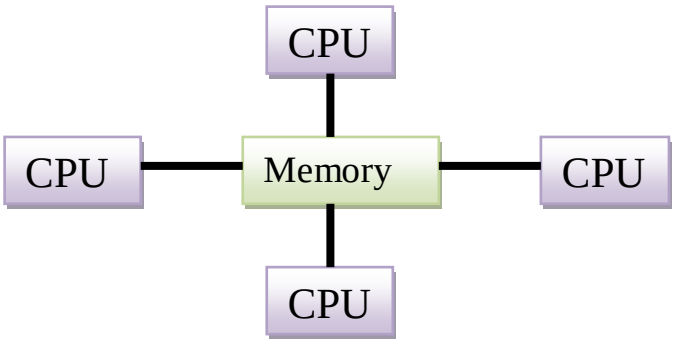


Figure 7. UMA Shared memory parallel machine.

These are mainly SMP-type machines that have multiple identical processors and can access memory equally and at the same time. They are sometimes referred to as CC-UMA (Cache Coherent UMA). Cache coherence means that if one processor updates a memory location, all other processors are aware of this change. This functionality is ensured at the hardware level.

Non-UMA

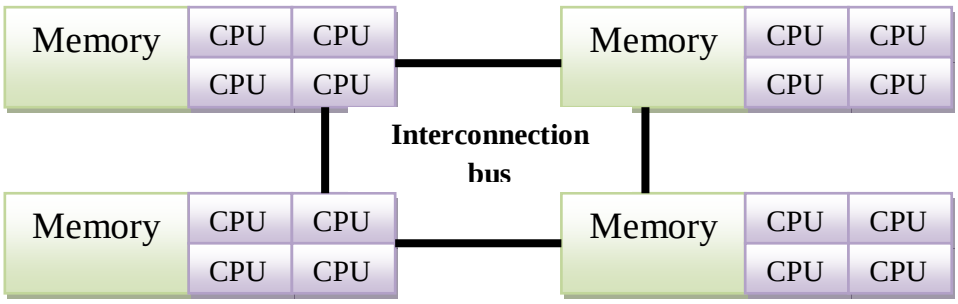


Figure 8. Non-UMA Shared memory parallel machine.

This type of machine is often designed by connecting two or more SMPs. One SMP can have direct access to the memory of another SMP. Access times to a given memory are not the same for all processors, and when a node is traversed, access is slower. If cache coherence is guaranteed, this is called CC-NUMA.

4.2.2. Distributed Memory Parallel Machines

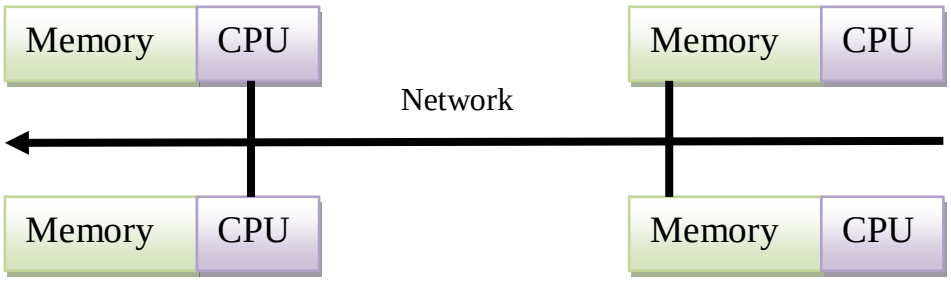


Figure 9. Parallel machines with distributed memory.

Like shared memory machines, distributed memory machines vary, but they share one thing in common: they require a communication network to connect the processors memories as we see in Figure 9. Each processor has its own local memory. The memory addresses of a given processor do not correspond to those of another, and therefore the concept of global memory does not exist. Since each processor has its own private memory, it operates independently. Indeed, any change made to its local memory has no effect on the memory of other processors, which precludes the concept of cache coherence. When a processor needs data from another processor's memory, the programmer is responsible for defining when and how the data is transferred. The programmer is also responsible for synchronization.

4.2.3. Hybrid Memory Parallel Machines

The fastest machines in the world employ so-called hybrid memory architectures (Figure 10), which combine the two previous types: shared and distributed. The shared memory

component is often an SMP machine. The distributed component consists of networking multiple SMP machines. The different SMPs can only address their own memory, and data

transfer between two SMPs requires network communications. The major difference between this type of architecture and NUMA SMPs is that the memory space is not shared, and inter-processor communication takes place over an interconnection network such as Ethernet or Infiniband.

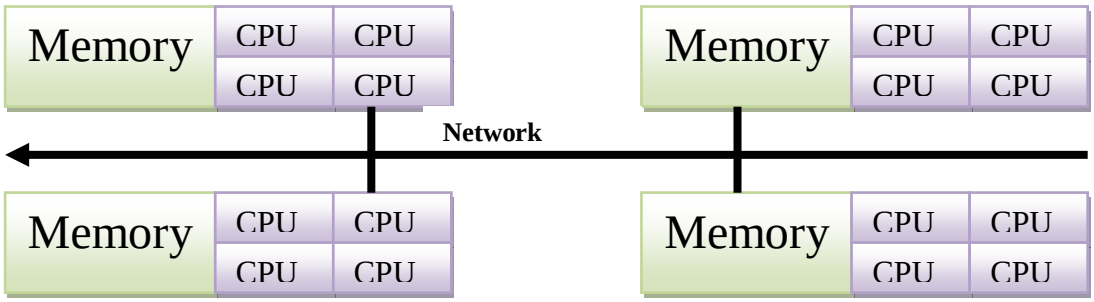


Figure 10. Hybrid memory parallel machine.

4.3. Parallel Programming Models

There are several programming models for parallel machines. These models exist at a level of abstraction above the hardware and memory architecture. Although at first glance, programming models are closely linked to the machine architecture, they are assumed to be implementable on any parallel machine, regardless of its characteristics. There is no ideal programming model, but some programming models are well-suited for a given application on a given machine[50]. Below, we describe the main parallel programming models.

4.3.1. The Shared Memory Model

In this programming model, tasks share a common address space to which they can read and write data asynchronously. Several mechanisms, such as locks and semaphores, can be used to control access to shared memory. This programming model is simplified from the user's perspective because there is no notion of data ownership by a task, which avoids explicit communications to transfer data from one task to another. However, in terms of performance, this last point is a disadvantage because it generates additional memory access, cache refresh, and bus traffic when multiple processors use the same data. Implementations of this model on shared memory machines are limited to the native compiler, which translates program variables into global memory addresses. However, there is no implementation of this model on distributed memory machines.

4.3.2. The Threaded Programming Model

In the threaded programming model, a single process can have multiple, concurrent execution paths. This concept can be thought of as a main program that includes a number of subroutines. The main program is scheduled for execution by the operating system, and it acquires all the system resources necessary for its execution. It then executes a set of instructions serially and creates a number of tasks (threads) that can be scheduled and executed concurrently by the OS. Each thread owns its local data but also shares the main

program's resources with other threads. Each thread has access to global memory because it shares the main program's address space. A thread's workload can be considered a subroutine of the main program, but it can run in parallel with another thread. Threads communicate with each other via global memory, which requires synchronization operations to guarantee exclusive access to a given location at a given time for a single thread.

Threads have variable lifetimes and can be created and destroyed throughout the program. The threaded programming model is often associated with shared-memory machines. Thread implementations typically include a library of functions or a series of directives buried within the parallel code. In both cases, the user is responsible for defining parallelism. There are several thread implementations, and most manufacturers have developed their own versions, which have affected the portability of parallel code. However, a standardization effort has given rise to two implementations that have become the standard today: POSIX threads [55] and OpenMP [56].

4.3.3. The Message Passing Programming Model

In this model, parallel programming is done by message passing. A set of tasks uses its own local memory during computation. Multiple tasks can reside on the same physical machine or on an arbitrary number of machines. Tasks exchange data through communications by sending and receiving messages. Data transfers require cooperative operations to be performed by each process. For example, a send operation must have a dual receive operation. Message Passing implementations take the form of a library of subroutines, and the programmer is responsible for detecting parallelism. As with any library, multiple versions have been developed, leading to compatibility issues. In 1992, the MPI Forum was founded with the goal of standardizing Message Passing implementations, including PVM [57]. Two standards were then developed: MPI [58] in 1994 and MPI-2 in 1996. Today, MPI is the most widely used programming model for message passing. In MPI implementations on shared-memory architectures, network communications are simply replaced by memory copies.

4.3.4. The Data Parallel Model

This model is based on data parallelism, which focuses parallel work on a set of data contained in an array or a multi-dimensional data structure. A set of tasks work collectively on the same data structure, but each task operates on a different partition of this structure. The tasks all perform the same operation on their data partition. On shared-memory architectures, all tasks can access the data structure via global memory. However, when the memory architecture is distributed, the data is divided into chunks that reside in each task's local memory. Programming with this model is generally done by writing code with data parallel constructs. These constructs can take the form of calls to library functions or directives recognized by a data parallel compiler. Implementations of this model are often in the form of compilers or extensions to them. Examples include Fortran compilers (F90 and F95) and their HPF (High Performance Fortran) extension [59], which support data parallel programming. HPF includes directives that control data distribution, assertions that can improve the optimization of the generated code, and data parallel constructs. Implementations of this

model on distributed memory architectures take the form of a compiler that converts standard code into Message Passing In (MPI) code, which distributes data across different processors, all transparently from the user's perspective. Despite its early popularity, HPF has not achieved the expected success, as evidenced by the analysis of its main author in [60].

4.3.5. Stream Computing Programming Model

This model, commonly called stream computing, is based on data parallelism. A single computing kernel is applied to a set of data. This model is the dominant model for graphics computing units. It is a model where parallelism is of the SIMD type, or multiple computing units (typically hundreds) execute the same instruction on a set of data in parallel. Machines supporting this type of model are GPUs, FPGAs, and certain specialized processors such as Stanford's Imagine [61] and Merrimac [62]. Several languages have also been developed to support this type of hardware, including StreamIt [63] and Brook [64]. CUDA [65] and OpenCL [66] are also implementations of this parallel programming model and are by far the most widely used today. The model consists of simplifying both the hardware and restricting the type of parallelism used.

4.4. Distributed system techniques

Distributed systems employ various techniques to achieve efficient parallel computation, scalability, and high performance. Below some distributed system :

4.4.1. High-Performance Computing (HPC)

HPC involves the use of supercomputers and clusters to perform complex computations at high speeds [67]. These systems utilize parallel processing techniques, where multiple processors work together on large-scale problems such as climate modeling, molecular simulations, and AI training. HPC clusters typically employ MIMD architectures and rely on frameworks like MPI and OpenMP for efficient task execution. While HPC provides unmatched computational power, it requires specialized hardware, high energy consumption, and complex software management.

4.4.2. Cloud Computing

Cloud computing offers on-demand access to computing resources such as servers, storage, and databases over the internet. It provides a scalable and cost-efficient alternative to traditional on-premise computing. Cloud computing has emerged as a transformative paradigm with the potential to revolutionize the implementation and delivery of IT services [68]. It offers a model that enables ubiquitous, convenient, and on-demand network access to a shared pool of configurable computing resources such as networks, servers, storage, applications, and services with minimal management effort or direct interaction with service providers [69]. Based on the nature of services delivered, cloud service providers are commonly categorized into three primary models: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) [70].

- **Infrastructure as a Service (IaaS):** Virtualized computing resources, including VMs and storage.
- **Platform as a Service (PaaS):** Development platforms that provide pre-configured environments for application deployment.
- **Software as a Service (SaaS):** Cloud-hosted software applications accessible via web browsers.

  Cloud computing reduces infrastructure costs and increases flexibility but presents challenges such as security risks, data privacy concerns, and vendor lock-in.

4.4.3. Graphic Processing Unit

GPU is a powerful multicore processor. GPUs have high-performance processing units for graphics processing. Initially, GPUs were designed to accelerate graphics rendering. They are currently used to parallelize general-purpose computations to reduce application runtime [71].GPUs are highly suited to implementing program execution with various data elements. This technique is referred to as data parallelism. Data parallelism distributes data components to parallel threads on GPUs.Data parallelism is most commonly used in 3D rendering, stereo vision, pattern recognition, image, video, and medical applications [72].

A significant performance difference exists between GPU and general-purpose multi-core CPU. Figure 11 shows an architectural comparison between CPU and GPU. CPUs are optimized for sequential programming. It uses advanced control logic to execute instructions from a single thread in parallel or out of sequential sequence while keeping the illusion of sequential execution.GPUs typically have several CPU cores, ALUs, control units, and memory types [72]

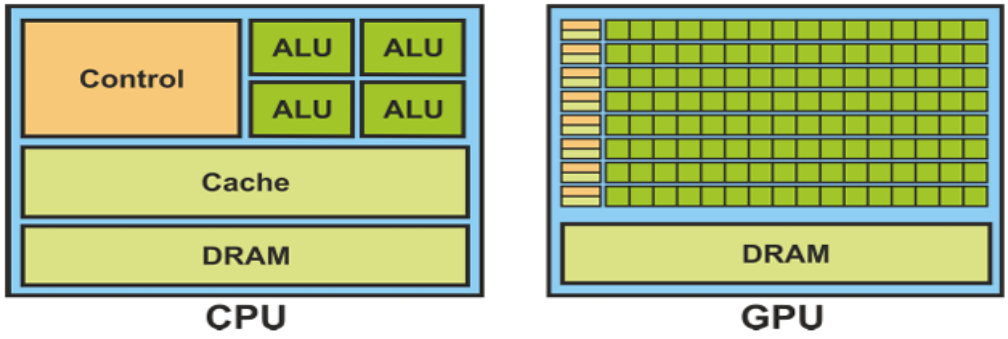


Figure 11. Comparison CPU vs GPU from source [73].

The GPU is a multi-core architecture that improves intense computing and frees up CPU resources. A GPU consists of global memory and streaming multiprocessors (SMs). Each SM includes a group of streaming processors (SP) connected to a local memory (register memory). SPs in an SM are linked to a shared memory [74, 75]. The architecture of GPUs require specif programming languages such as: OpenCL, OpenMP, OpenACC and CUDA [76]. The Compute Unified Device Architecture (CUDA) is a powerful hardware and software architecture for managing computations on GPUs. It treats the GPU as a data-parallel computing device, eliminating the requirement to translate computations to the graphics pipeline [77]. GPU computations are programmed as kernel functions. A kernel program defines the execution of a serial thread on a GPU. The host CPU launches the kernel

with given numbers of blocs and threads. A bloc represents a set of a specific number of threads, and all blocs in that kernel launch have the same number of threads.

4.4.4. Multiprocessing

Multiprocessing is a technique where multiple processors or cores within a single machine execute tasks simultaneously, improving computational performance and efficiency. It can be classified into Symmetric Multiprocessing (SMP), where all processors share the same memory and workload, and Asymmetric Multiprocessing (AMP), where one processor controls task distribution while others execute assigned processes. Additionally, multithreading enables multiple threads within a single process to execute concurrently, improving system responsiveness. The key advantage of multiprocessing is that it maximizes CPU utilization and speeds up task execution. However, it also introduces challenges such as increased complexity in task scheduling, potential race conditions, and overhead in managing shared resources among multiple processors.

4.4.5. MapReduce and Hadoop

MapReduce is a programming model for processing and generating large datasets with a parallel, distributed algorithm on a cluster. Hadoop, an open-source framework, implements the MapReduce model and provides a scalable and fault-tolerant system for distributed data processing. In the context of image processing, Hadoop and MapReduce are used to split large image datasets into smaller chunks that can be processed independently across different nodes in a cluster. The "Map" phase processes each chunk, such as applying filters, extracting features, or transforming image formats,while the "Reduce" phase aggregates the results, such as compiling extracted features or stitching processed image parts. This technique is particularly useful for batch-processing large image repositories, such as satellite or surveillance data. One of the key strengths of Hadoop is its ability to handle failures gracefully, by reassigning failed tasks to other nodes without disrupting the overall process. It is widely adopted in big data environments where scalability and data reliability are essential.

4.5. Advantages and Disadvantages of Distributed Systems

Distributed systems are widely used in modern computing due to their ability to connect multiple independent machines to work together as a single cohesive system. While they offer significant benefits for performance and scalability, they also come with a set of challenges that must be carefully managed.

4.5.1. Advantages

- **Scalability:** Distributed systems can dynamically scale by adding more nodes, accommodating increasing workloads efficiently.
- **Fault Tolerance:** Redundant nodes ensure that system failures do not lead to complete outages, improving reliability.

- **Cost Efficiency:** By utilizing multiple lower-cost machines instead of expensive supercomputers, distributed systems reduce infrastructure costs.
- **Resource Sharing:** Multiple users and applications can access shared resources, optimizing utilization.

4.5.2. Disadvantages

- **Complexity:** Managing distributed nodes, data synchronization, and task scheduling requires sophisticated algorithms.
- **Consistency Issues:** Ensuring data consistency across multiple nodes is challenging, especially in real-time applications.
- **Network Overhead:** Communication between nodes introduces latency, which may impact performance.
- **Security Concerns:** Data transmitted across networks is vulnerable to cyber threats and requires strong encryption protocols.

Distributed systems have revolutionized computing by enabling scalable, efficient, and resilient architectures for modern applications. By leveraging computing models such as shared memory, distributed memory, and hybrid approaches, distributed systems can handle complex computations across multiple nodes. Techniques such as HPC, cloud computing, GPU acceleration, and multiprocessing provide powerful tools for parallel processing, optimizing performance across various domains. Despite their advantages, distributed systems also pose challenges, including synchronization, fault tolerance, and security risks. As technology advances, ongoing research and innovations will continue to enhance distributed computing, making it a cornerstone of future computing paradigms.

5. Literature review about image processing using distributed system

In recent years, distributed systems have been increasingly applied to image processing tasks to overcome the limitations of sequential methods. Researchers have developed various distributed image processing frameworks and algorithms to leverage the computational power of multiple machines. For example, frameworks like Apache Hadoop and MapReduce have been used to parallelize tasks such as image filtering and segmentation by dividing images into smaller blocks and distributing them across different nodes. Similarly, Apache Spark has been employed for real-time image classification and feature extraction tasks, benefiting from its in-memory processing capabilities. In the field of deep learning, distributed systems have enabled the training of large convolutional neural networks (CNNs) for image classification and object detection. Frameworks like TensorFlow and PyTorch offer distributed training mechanisms that allow models to be trained across multiple GPUs or compute nodes. Furthermore, high-performance computing (HPC) environments using MPI (Message Passing Interface) and OpenMP have also been adopted for image processing, where speed and precision are important.

Real-time image processing remains a significant challenge, primarily due to the large amount of data contained in each image that must be processed rapidly and efficiently.

According to [78], interactive real-time processing and rendering (particularly on immersive, high-resolution displays) require highly sophisticated methods in computer graphics, efficient data handling, and advanced parallelization strategies. These tasks are computationally intensive and are further complicated by the projected growth of datasets in this field, which are expected to reach terabyte (TB) scale in the near future.

To address these demands, researchers have made considerable progress in developing parallel processing algorithms that utilize the computational power of Graphics Processing Units (GPUs) [79,80] and multi-core Central Processing Units (CPUs). These parallel architectures have significantly accelerated image segmentation and other image processing tasks. Notably, the introduction of GPUs has provided a powerful, cost-effective, and adaptable platform for parallel computing, which has been widely adopted in various successful studies across intelligent computing and image analysis domains [81].

This section presents a literature review on parallel image processing, highlighting various parallelization tools applied across different domains. In particular, the medical field has recognized the significant benefits of parallel processing algorithms, as demonstrated by real-world experiments conducted in several hospitals [82]. Traditional Central Processing Units (CPUs), however, struggle to efficiently handle the growing volume of medical image data, especially given the rapid increase in dataset sizes. In response to these limitations, Graphics Processing Units (GPUs) have emerged as a cutting-edge solution, offering substantial computational power to address complex challenges in medical image analysis [83].

In [84], a hybrid serial–parallel CNN-Transformer (SPCT) network was proposed for 3D medical image segmentation, integrating the strengths of Convolutional Neural Networks (CNNs) and Transformer architectures. The model incorporates a Cross-Window Self-Attention Transformer (CWST) module to capture global contextual information, along with a Multi-Scale Local Enhancement (MLE) module for effective feature fusion. Extensive evaluations on prostate, atrium, and pancreas MRI/CT datasets demonstrated that SPCT outperforms six state-of-the-art segmentation methods in terms of Dice Similarity Coefficient (DSC), Intersection over Union (IoU), and boundary accuracy, all while maintaining relatively low computational complexity. These results indicate that SPCT is a promising framework for accurate and efficient 3D medical image segmentation. However, its parallelization strategy primarily focused on enhancing feature learning, rather than accelerating the clustering or segmentation processes through full parallel optimization.

In [85], a deep learning-based framework was proposed for simultaneous MRI reconstruction and segmentation. The approach employs a calibrationless, parallel image-domain deep learning model designed to enhance image quality and improve segmentation robustness in the presence of distortions. By integrating Deep Structured Low-Rank (Deep-SLR) reconstruction with a dedicated segmentation network, the method effectively reduces aliasing and blurring artifacts, resulting in improved segmentation accuracy and computational efficiency. Experimental evaluations on brain MRI datasets show that the proposed framework outperforms existing parallel MRI reconstruction and segmentation methods, particularly under few-shot learning scenarios.

Another notable deep learning advancement is the Bidirectional Efficient Attention Parallel Network (BEAP-Net) [86], developed to enhance 3D medical image segmentation within a semi-supervised learning framework. BEAP-Net integrates Supreme Channel Attention

(SCA) and Parallel Spatial Attention (PSA) modules to effectively capture both spatial and channel-specific features. Experimental results on Left Atrium (LA) and pancreas datasets show that BEAP-Net surpasses eight state-of-the-art methods, achieving superior segmentation accuracy while maintaining computational efficiency. However, despite its impressive performance on public datasets, the model's reliance on semi-supervised learning may constrain its applicability in real-time clinical environments where fully annotated datasets are readily available.

Similarly, the Multi-Parallel Blocks UNet (MPB-UNet) [87] was proposed for automated brain tumor segmentation, enhancing the conventional UNet architecture through the integration of multiple parallel processing blocks inspired by the mechanisms of human visual perception. The model incorporates Atrous Spatial Pyramid Pooling (ASPP) to effectively capture multi-scale contextual information, thereby improving segmentation precision. The architecture was evaluated on the Low-Grade Glioma Segmentation Dataset, where MPB-UNet demonstrated superior performance, achieving an accuracy of 99.86% and significantly outperforming existing state-of-the-art methods.

In [88], parallel Fuzzy C-Means (FCM) clustering was investigated for brain tumor segmentation, leveraging GPU acceleration through CUDA. The approach utilized FLAIR MRI images and implemented parallelization for key computational steps, including cluster initialization, membership matrix calculation, and spatial function evaluation. Although the use of GPU significantly accelerated the segmentation process, the method lacked an optimization mechanism for refining cluster centroids, which limited its capability to escape local optima and potentially reduced segmentation accuracy.

The study presented in [89] conducted a comparative analysis of two parallel implementations (Bias-Corrected FCM (BCFCM) and Spatial FCM (SFCM))with a focus on enhancing robustness and efficiency in MRI image segmentation. Performance was evaluated in terms of both segmentation quality and processing speed. By utilizing GPU-based architectures, the implementations demonstrated substantial reductions in execution time while preserving high segmentation accuracy. The findings underscore the effectiveness of parallel processing in optimizing FCM algorithms for medical image analysis.

Adapting FCM for 3D medical image segmentation presents additional computational complexities. In [90], a hybrid parallel implementation was introduced to enhance segmentation accuracy while notably decreasing execution time. Experimental results on both real and simulated medical datasets showed a speedup of up to 5× compared to traditional sequential methods, highlighting its potential for large-scale medical imaging applications.

A novel knowledge-driven FCM approach, FCM-GENIUS, was proposed in [91] for efficient brain tissue segmentation from MRI scans. This method combines region of interest (ROI) selection, knowledge-based initialization, and optimization techniques to enhance centroid selection and minimize computational complexity. Furthermore, the use of CUDA-enabled GPU parallelization accelerates processing significantly. Experimental evaluations on the IBSR datasets demonstrated that FCM-GENIUS achieves a reduction in segmentation time of up to seven times, while maintaining accuracy on par with existing methods.

In [92], the authors introduced a novel parallel computational approach for image processing, specifically designed for a spectral analysis algorithm within a distributed environment for video surveillance systems. This method utilizes the ZeroMQ library to

distribute video frames from the surveillance stream across multiple computing nodes (multi-core processors), ensuring load balancing. Additionally, OpenMP technology is employed to leverage all available CPU cores for accelerating the spectral analysis of the images. The primary focus of the work involves two key aspects: first, the separation of the video stream into individual frames received from the camera, and second, the spectral analysis of these images on multi-core platforms.

Benchara, Y. [93] introduced a scalable distributed k-means algorithm using cloud microservices for high-performance computing (HPC). Their study highlighted the feasibility of leveraging cloud-based parallel processing for efficiently handling large-scale image data. Similarly, Enfedaque et al. [94] proposed a GPU-based implementation of bitplane coding, which provided high-performance parallel coefficient processing for image compression.

In [95], three key contributions are presented. The first contribution involves enhancing the performance of the Support Vector Machine (SVM) for breast cancer diagnosis by utilizing a modern Grey Wolf Optimizer (GWO). The second contribution introduces three efficient scaling techniques as alternatives to the traditional normalization method. The final contribution implements a parallel technique that employs task distribution to improve the efficiency of GWO. The parallelized version of the model demonstrates promising results, particularly in terms of execution time when run on four CPU cores.

In [96], the primary contribution of the paper is the implementation of a MapReduce programming algorithm to analyze large sets of fingerprint images that are typically too large to process due to limited physical memory. The approach aims to extract features from these images efficiently. Initially, the images are stored in an image data repository for preprocessing, followed by feature extraction for the biometric traits of each user, which are then stored in a database. The proposed algorithm simultaneously preprocesses and extracts key features, such as ridges and bifurcations, from multiple fingerprint images. Feature points are detected using the Crossing Number (CN) method. The algorithm is validated using data from the National Institute of Standards and Technology's (NIST) Special Database 4, which contains fingerprint images from various users. Experimental results demonstrate that the MapReduce approach significantly reduces processing time, achieving nearly a 50% decrease compared to traditional methods.

X. Tan et al. [97] proposed an adaptive Spark-based approach for remote sensing data processing, demonstrating enhanced efficiency through map-reduce-based remote processing. The developed model exhibited improved stability and performance within a cloud environment. A mapping and reducing strategy was applied to image tiles, leading to significant improvements in processing large volumes of remote sensing data. However, the Spark model was constrained to pixel-based classification, limiting its broader applicability.

Despite the effectiveness of using several parallel techniques for image processing, challenges such as communication overhead, fault tolerance, and load balancing remain critical concerns in distributed image processing. As datasets continue to grow in volume and complexity, the role of distributed systems in image processing will become increasingly essential, making it a dynamic and evolving research area with significant practical impact.

To sum up, the analysis of the literature presents the progress made in parallel image processing,facilitated by furthering the approaches to parallel algorithms, optimization principles, andcomputational platforms. The discussed studies can thus be regarded as a

critical starting ground forfuture investigations aimed at further enhancement and application of parallel approaches in the contextof image processing to enhance the prospects of the latter.

6. Conclusion

In this chapter, we explored the fundamental concepts of image processing and the role of parallelization in enhancing computational efficiency. Image processing encompasses a wide range of techniques, including filtering, segmentation, feature extraction and selection, which often require significant computational power. To address these challenges, parallelization methodologies have been developed to distribute processing tasks across multiple computing units, improving speed and scalability. Additionally, parallel machine architectures, such as multi-core processors, GPU-based processing, and distributed computing systems, provide the necessary infrastructure to handle large-scale image data efficiently. By integrating parallel computing techniques with advanced hardware architectures, modern image processing applications can achieve high performance, enabling real-time analysis and large-scale data processing in various fields, including medical imaging, satellite image analysis, and artificial intelligence. We have explored the concept of image processing, including its definition and various techniques used to manipulate and analyze digital images. We also discussed distributed systems, providing an overview of their definition and the methods employed to manage and process data across multiple interconnected systems. Both fields play crucial roles in modern computing, with image processing enabling efficient handling of visual data and distributed systems facilitating the management of large-scale computations and resources.

Chapter 2

Metaheuristic for image processing

1. Introduction

Image segmentation is a fundamental task in image processing and computer vision that involves partitioning an image into meaningful regions, making it easier to analyze or interpret. Effective segmentation is essential in various applications, such as medical imaging, remote sensing, object detection, and industrial inspection. However, due to the complex nature of real-world images which characterized by noise, low contrast, and intensity inhomogeneity, traditional segmentation methods often struggle to deliver optimal results.

Metaheuristic algorithms have emerged as powerful tools for tackling image segmentation problems, especially when classical approaches fall short. Inspired by natural phenomena such as evolution, swarm behavior, or physical processes, metaheuristics provide a flexible and efficient means to search for near-optimal solutions in large and complex search spaces. Unlike deterministic methods, metaheuristics do not guarantee the global optimum but often find sufficiently good solutions within reasonable computational time.

Popular metaheuristic algorithms applied to image segmentation include Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and more recent approaches like Grey Wolf Optimization (GWO) and Whale Optimization Algorithm (WOA). These methods are commonly used to optimize clustering objectives, thresholding techniques, or region merging criteria. Their ability to balance exploration and exploitation makes them particularly suitable for segmentation tasks where the objective function is nonlinear, multi-modal, or otherwise difficult to optimize analytically.

In this chapter, we begin by introducing the concept of metaheuristic algorithms, outlining their general principles and applications. We then explore how these algorithms are utilized specifically for image segmentation, highlighting their strengths in handling complex, high-dimensional problems. Following this, we delve into the parallelization of metaheuristic algorithms, discussing how parallel computing enhances their performance and scalability. We then present a comprehensive review of existing research on parallel metaheuristicsapplied to image segmentation. Finally, we conclude with a discussion that synthesizes the insights gained and suggests potential directions for future work.

2. Metaheuristic algorithms

In general, the complexity of real-world problems has been steadily increasing, rendering traditional mathematical programming techniques increasingly inadequate for solving and optimizing such problems. Most real-life optimization challenges are inherently nonlinear, highly

complex, and multimodal, often involving conflicting objective functions. These characteristics make the task of identifying optimal or even near-optimal solutions particularly difficult. In fact, even for seemingly simple or linear objective functions, achieving an optimal solution may be infeasible or non-existent. Consequently, there is often no guarantee of obtaining an optimal solution in practical scenarios [98][99].

In response to these challenges, metaheuristic optimization algorithms have emerged as a prominent and rapidly evolving area of research. These high-level strategies are designed to guide the search process toward high-quality solutions by selecting, combining, or adapting heuristics in an intelligent manner. Metaheuristics aim to efficiently explore complex search spaces and are capable of producing sufficiently good, improved, and robust solutions for a wide range of real-world optimization problems [100][101].

Metaheuristic algorithms represent a class of powerful optimization techniques specifically designed to tackle complex problems that are intractable for conventional mathematical or deterministic methods. These algorithms draw inspiration from various natural processes and phenomena, such as genetic evolution, swarm intelligence, and thermodynamic principles, in order to efficiently explore vast and complex search spaces in pursuit of globally optimal or near-optimal solutions. Prominent examples include Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Simulated Annealing (SA), and Tabu Search (TS), all of which have demonstrated considerable success across diverse domains including engineering, finance, and computer science [102].

One notable advantage of metaheuristic algorithms lies in their independence from initial solution requirements, making them particularly effective in scenarios where the starting conditions are unknown or poorly defined. Moreover, their robustness in navigating high-dimensional, multimodal, and non-convex search spaces distinguishes them from traditional optimization methods. Despite these strengths, metaheuristics are not without limitations. Due to their inherently stochastic nature, there is no guarantee that the global optimum will be found in every execution, and the quality of solutions can vary. Furthermore, their computational overhead can become significant, especially when dealing with large-scale or real-time applications [103].

In summary, while metaheuristic algorithms offer a flexible and effective approach for solving intricate optimization problems, their application must be guided by a deep understanding of the problem context and computational constraints. The selection of an appropriate metaheuristic should consider both the specific characteristics of the optimization problem and the resources available, as their performance is highly problem-dependent [104]. Although not infallible, metaheuristics remain indispensable tools in modern optimization, balancing exploration and exploitation to yield practical solutions where traditional methods fall short.

The term metaheuristic originates from the combination of "meta," meaning beyond or at a higher level, and "heuristic," which denotes a problem-solving approach based on trial-and-error or experiential strategies. Historically, algorithms incorporating stochastic elements were commonly referred to as heuristic methods. Metaheuristics, in this context, have evolved into overarching strategic frameworks that orchestrate and adapt lower-level heuristics, aiming to surpass the limitations of conventional local optimization methods. These strategies systematically blend elements of local search and randomized exploration to efficiently navigate complex solution spaces. While they are capable of yielding high-quality solutions to difficult optimization

problems within reasonable computational time, there is generally no formal guarantee of attaining the global optimum [105].

Metaheuristic algorithms are particularly effective in addressing large-scale and computationally intractable problems, such as those classified as NP-hard, or in environments characterized by uncertainty, incompleteness, or imprecision. Due to the enormity of the search space, it is typically infeasible to evaluate all potential solutions exhaustively. However, one of the key advantages of metaheuristics is their problem-independent design, requiring minimal assumptions about the underlying optimization model. This makes them highly adaptable across a wide range of domains and problem types. Unlike deterministic or iterative optimization techniques, metaheuristics do not ensure convergence to an optimal solution. Many rely on stochastic processes, meaning that the solutions obtained are influenced by probabilistic variables generated during the search process [106].Despite their probabilistic nature, metaheuristics have demonstrated the ability to find high-quality solutions in combinatorial optimization problems with significantly reduced computational overhead compared to exact algorithms, iterative solvers, or rudimentary heuristics. Their capacity to explore diverse regions of the solution space makes them valuable tools for solving complex optimization tasks [107]. A generic schematic illustrating the typical workflow of metaheuristic algorithms is presented in Figure 13.

Much of the scholarly work on metaheuristics is empirical, centered around computational experiments and performance benchmarking. Nonetheless, a body of theoretical research also exists, addressing aspects such as algorithmic convergence and conditions under which global optimality might be achieved. While the field has seen the emergence of numerous innovative and practically effective metaheuristic techniques, it has also been criticized for inconsistencies in scholarly rigor. Common issues include vague conceptual frameworks, inadequate experimental validation, and insufficient engagement with prior literature [108].

To facilitate better understanding and application, metaheuristics have been broadly categorized in the literature based on their core operational strategies and sources of inspiration, ranging from biological and physical processes to social and cognitive behaviors. These classifications are instrumental in elucidating the foundational principles of each algorithm and guiding practitioners in selecting suitable approaches for specific problem contexts. Figure 12 provides a comprehensive taxonomy of metaheuristic families, highlighting representative algorithms within each category.

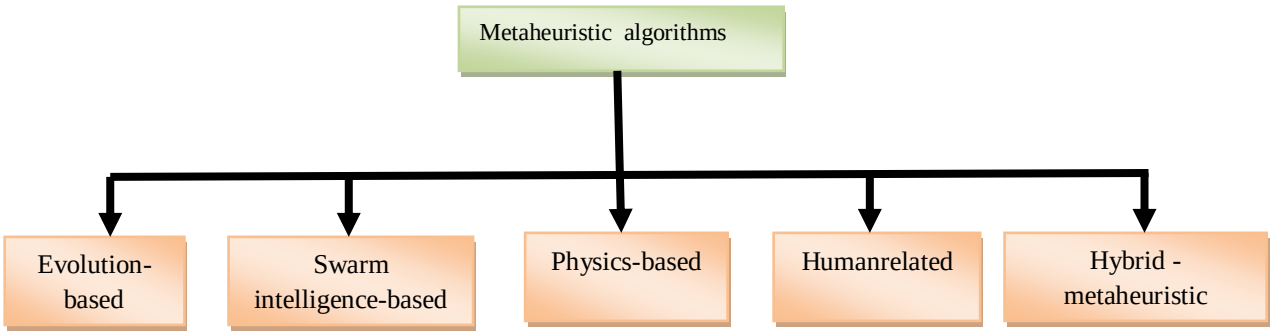


Figure 12.Classification of nature-inspired algorithms.

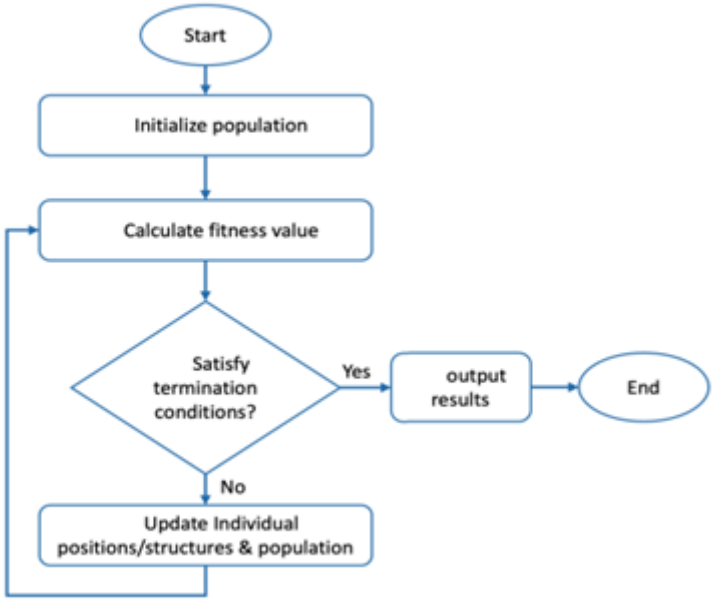


Figure 13. Generic flowchart of metaheuristics algorithms.

2.1. Evolution-based algorithms

Evolutionary Algorithms (EAs) constitute a family of optimization techniques inspired by the principles of natural selection as proposed in Darwin's theory of evolution, which posits that genetic variation arises randomly within a population and that only the fittest individuals survive and reproduce. Drawing upon this biological paradigm, EAs are designed to explore complex search spaces and identify near-optimal solutions through iterative processes. Each cycle of an EA, referred to as a generation, typically involves a sequence of key operations: parent selection, recombination (or crossover), mutation, and survivor selection. While crossover and mutation serve to diversify the population and explore the search space, parent and survivor selection mechanisms focus the search through exploitation of promising regions. Prominent representatives of this algorithmic class include Genetic Algorithms (GA) [13] and Differential Evolution (DE) [109]. These methods begin with a randomly initialized population of candidate solutions, which is iteratively improved by combining and modifying high-quality individuals via evolutionary operations. Among these, the Genetic Algorithm, explicitly modeled on the Darwinian process of evolution, remains the most widely applied and extensively studied. Building upon these foundations, more sophisticated variants such as Genetic Programming, and Differential Evolution have emerged, extending the evolutionary framework to address a broader range of optimization problems. Overall, evolutionary algorithms have demonstrated remarkable adaptability and efficacy across a wide array of application domains. Their capabilities have been successfully leveraged in areas such as image analysis, disease detection, wind speed prediction, and the identification of cancer-related symptoms, underscoring their value as versatile and robust tools for solving complex real-world problems.

2.2. Swarm intelligence-based algorithms

The second prominent category of metaheuristic algorithms the Swarm Intelligence (SI**),** SI is inspired by the collective behavior and decentralized communication observed in social

organisms. These algorithms emulate the way groups of animals, such as birds, fish, ants, and bees, interact and exchange information to guide collective decision-making and problem-solving. The fundamental principle underpinning swarm-based metaheuristics is that the behavioral dynamics of individual agents are influenced by shared knowledge within the group, which in turn directs their movement and convergence patterns during the optimization process. By regulating information exchange within the swarm, these algorithms achieve a balance between exploration of the search space and exploitation of promising regions.

Numerous bio-inspired algorithms fall under this category, each modeled on distinct forms of social or biological behavior. For instance, the BAT algorithm [110], inspired by the echolocation behavior of bats, adapts frequency tuning and signal loudness to explore the solution space effectively. The Cuckoo Search (CS) algorithm [111], modeled on the brood parasitism of cuckoo birds, has been widely applied to real-world optimization problems, with binary variants developed to handle discrete search spaces. Another noteworthy algorithm, the Grasshopper Optimization Algorithm (GOA) [112], simulates the locational dynamics and social interactions of grasshoppers to achieve optimization through a balance of attractive and repulsive forces. The Firefly Algorithm (FA) [113] draws on the bioluminescent communication of fireflies, leveraging perceived brightness and spatial distance to iteratively update solution candidates and converge toward optima—making it especially suitable for feature selection tasks. The Dragonfly Algorithm (DA) [114] replicates the static and dynamic swarming behaviors of dragonflies to solve optimization problems through local and global search strategies. Similarly, the Grey Wolf Optimizer (GWO) [11] models the social hierarchy and group hunting strategies of grey wolves, incorporating leadership dynamics to steer the population through a multi-dimensional search space. Another notable approach, the Flower Pollination Algorithm (FPA) [115], simulates pollination mechanisms in flowering plants to combine local exploitation and global exploration via probabilistic interactions. The Ant Lion Optimizer (ALO) [12] is based on the predatory behavior of ant lions and their interactions with ants, effectively modeling trapping mechanisms to guide optimization processes. Lastly, the Whale Optimization Algorithm (WOA) [14] mimics the bubble-net hunting strategies of humpback whales, incorporating encircling mechanisms and spiral-shaped movements to perform guided search and convergence.

Collectively, swarm intelligence algorithms have demonstrated considerable efficacy in solving a broad range of complex, multi-dimensional, and nonlinear optimization problems. Their adaptive, decentralized nature makes them especially well-suited for dynamic and uncertain environments, where traditional deterministic methods often fall short.

2.3. Physics-based algorithms

The third major category of metaheuristic algorithms encompasses physics-based optimization techniques, which are grounded in the simulation of physical laws and phenomena to guide the search for optimal solutions. These algorithms are inspired by fundamental principles from physics, such as thermodynamics, electromagnetism, and gravitational dynamics, and apply these concepts metaphorically to traverse complex solution spaces.

One of the earliest and most well-known examples is Simulated Annealing (SA) [116], modeled after the annealing process in metallurgy, where materials are heated and then slowly cooled to achieve a stable crystalline structure. SA mimics this process to escape local optima and converge

toward a global optimum, making it especially effective in solving multimodal and rugged optimization landscapes. Another notable algorithm is the Lightning Search Algorithm (LSA) [117], which draws inspiration from the unpredictable and powerful nature of lightning strikes. LSA integrates stochastic search mechanisms with both local and global exploration strategies, using metaphorical discharge paths to balance intensification and diversification in the search space. The Gravitational Search Algorithm (GSA) [118] simulates the laws of gravity and the motion of celestial bodies. In this framework, candidate solutions are treated as objects whose masses influence one another through gravitational attraction. Heavier (i.e., fitter) solutions exert a stronger pull, guiding the population toward more promising regions of the search space. Similarly, Electromagnetic Field Optimization (EFO) [119] emulates the interactions among charged particles within an electromagnetic field. This algorithm governs the movement of particles through attraction and repulsion forces, facilitating a dynamic and adaptive search process that can efficiently converge on optimal solutions.

Beyond these, several other physics-inspired metaheuristics have been developed, such as the Multi-Verse Optimizer, the Sine-Cosine Algorithm, and variants of GSA, each leveraging distinct physical metaphors to tackle high-dimensional, nonlinear, or combinatorial optimization problems. These techniques have shown particular promise in feature selection tasks across diverse datasets, offering robust and flexible tools for handling real-world complexity in data-driven environments.

2.4. Human-related algorithms

Human-inspired metaheuristic algorithms are a class of optimization techniques modeled on human social behaviors, cognitive processes, and learning mechanisms. These algorithms emulate how humans interact, learn, and collaborate to address complex problems, offering novel strategies for solving diverse optimization challenges. This category encompasses several algorithms that draw upon psychological and educational paradigms to enhance the exploration and exploitation of the solution space. One such approach is the Brainstorm Optimization (BSO) algorithm [120], which simulates the human process of idea generation in group discussions. Inspired by creative problem-solving sessions, BSO iteratively generates, evaluates, and refines candidate solutions through collaborative mechanisms, making it particularly effective for tasks such as data classification and high-dimensional optimization. Another notable method is Teaching–Learning-Based Optimization (TLBO) [121], which models the educational dynamics between a teacher and students in a classroom setting. This algorithm leverages the influence of a 'teacher' (the best solution in the population) to guide the learning of 'students' (other solutions) through two main phases: the teaching phase and the learning phase. These phases promote both exploration and exploitation by simulating knowledge dissemination and peer-to-peer learning, thereby enhancing convergence toward optimal solutions. The Gaining–Sharing Knowledge-Based Algorithm (GSK) [122] also follows a human-centric philosophy, replicating the way individuals gain, share, and assimilate knowledge within a community. This algorithm emphasizes cooperative learning and mutual exchange of information among candidate solutions, which facilitates a more informed and diversified search process. By modeling human knowledge transfer, the GSK algorithm improves adaptability and performance across a wide range of optimization problems.

In essence, human-based metaheuristics offer unique and flexible frameworks for addressing real-world optimization tasks by mimicking fundamental aspects of human cognition, learning, and cooperation.

2.5. Hybrid Metaheuristic

Hybrid metaheuristic algorithms have recently garnered significant attention for their efficacy in addressing complex optimization problems [123]. In particular, they have shown substantial promise in the domain of feature selection, where the goal is to extract the most relevant and optimal subset of features from high-dimensional datasets. These algorithms are constructed by strategically integrating the most effective components such as operators, strategies, or mechanisms from distinct metaheuristic frameworks. By combining complementary strengths of multiple algorithms, hybrid approaches are able to overcome common limitations such as premature convergence and entrapment in local optima. This integration enhances both the exploration (global search) and exploitation (local refinement) capabilities, facilitating more efficient traversal of the search space. As a result, hybrid algorithms are better equipped to deliver near-optimal or optimal solutions with improved robustness and adaptability. The synthesis of diverse algorithmic paradigms allows hybrid metaheuristics to achieve a superior balance between search intensification and diversification. This translates into improved convergence speed, enhanced solution quality, and greater computational efficiency. Ultimately, hybrid metaheuristics represent a powerful strategy in modern optimization, leveraging the strengths of multiple techniques to yield more effective and reliable outcomes across a wide array of application domains.

3. Image segmentation based-metaheuristic

Image segmentation is a crucial operation in image processing, where an image is partitioned into distinct regions based on various features, such as intensity, color, texture, or other characteristics. Traditional segmentation techniques, such as thresholding, clustering, and edge detection, often encounter difficulties when dealing with complex images, noise, or non-uniform illumination. To overcome these challenges, metaheuristic optimization algorithms have been increasingly applied to enhance segmentation accuracy and robustness. These algorithms optimize segmentation parameters by effectively balancing exploration and exploitation strategies, improving segmentation results. Recognized as the primary and most fundamental operation in image analysis, image segmentation plays a pivotal role in diverse computer vision applications, such as medical imaging [124], autonomous target recognition [125], geographic imaging [126], and robotic vision [127]. Image segmentation generally involves dividing an image into multiple segments, typically separating the foreground from the background, based on specific features like textures or grayscale values. In one notable contribution, Mandal introduced an enhanced version of image segmentation using Particle Swarm Optimization (PSO), which demonstrated significant improvement in segmentation performance [128]. Additionally, various nature-inspired optimization algorithms have been utilized in image segmentation to solve related optimization challenges and attain optimal solutions [129][130][131].

The subsequent sections offer a comprehensive review of the most widely used image segmentation techniques and their improvements through nature-inspired algorithms. Figure 14 presents statistical analysis of metaheuristic-based image segmentation in medical applications conducted from 2012 to 2022, based on data sourced from Scopus databases [132].

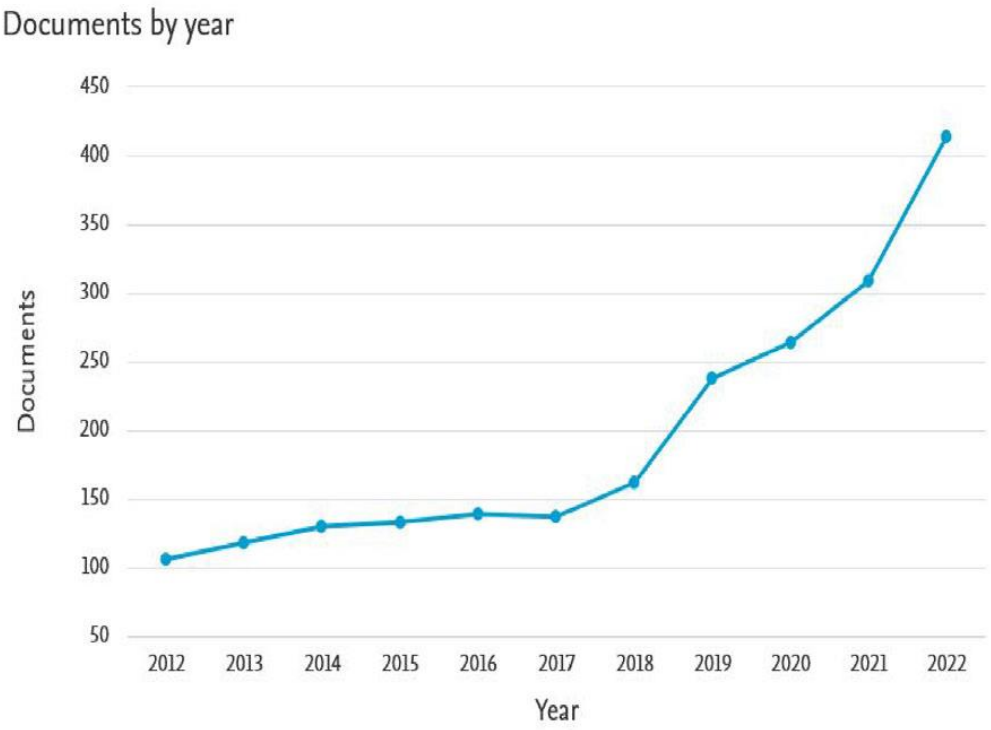


Figure 14. Histogram of publications of image segmentation using Metaheuristics in medical images [132].

A variety of methods have been introduced in the literature that apply metaheuristic algorithms to image segmentation tasks. These methods exploit the global search capabilities of metaheuristic techniques to address the shortcomings of conventional segmentation approaches, particularly in handling complex, noisy, or low-contrast images. Researchers have investigated a broad spectrum of algorithms, including those based on swarm intelligence, evolutionary strategies, and physics-inspired methods, to optimize key segmentation parameters such as threshold values, cluster centroids, or region boundaries. The expanding body of research underscores the effectiveness and flexibility of metaheuristic-based approaches in improving segmentation accuracy and robustness, demonstrating their applicability across diverse image types and practical scenarios.

3.1. Thresholding-based image segmentation using metaheuristic

The task of finding optimal threshold values in an image is often referred to as the thresholding problem. The image histogram is typically used to identify threshold points, with each image possessing its own set of optimal thresholds [133]. Otsu and Kapur methods [134] are widely recognized techniques for determining these thresholds. However, the challenge of multilevel thresholding (MTH) for image segmentation is inherently complex, with detailed discussions on these challenges found in [135, 136]. While Otsu and Kapur are effective for images with a small

number of thresholds, they become computationally expensive and time-consuming when dealing with images that require a large number of thresholds. As a result, nature-inspired optimization algorithms and Swarm Intelligence (SI) methods are employed to tackle such intricate segmentation problems. These metaheuristics mimic natural behaviors, such as those of animals, birds, and humans, to identify optimal solutions. Various metaheuristic algorithms have been applied to MTH problems, including Equilibrium Optimizer (EO) [137], Chimp Optimization Algorithm (ChOA) [138], Artificial Bee Colony (ABC) [139], Particle Swarm Optimization (PSO) [130], Bacterial Foraging Optimization (BFO) [141], and Cuckoo Search (CS) [142]. For medical image segmentation, Genetic Algorithm (GA) combined with Simulated Binary Crossover (SBX) [143] has been utilized to obtain optimal thresholds, showing superior performance in comparison with other algorithms. A modified version of Artificial Bee Colony (ABC), called CCABC, was proposed in [144] to enhance segmentation performance, especially when applied to COVID-19 X-ray image segmentation, outperforming other competitive algorithms. The ABC algorithm [145] was also successfully used to determine the optimal threshold for melanoma detection, with results showing superior performance over alternative methods. In [146], a novel hybrid approach combining the Slime Mold Algorithm (SMA) and Whale Optimization Algorithm (WOA)was introduced to address image segmentation problems for COVID-19 chest X-ray images. The results demonstrated that this hybrid method outperformed all comparison metrics. Dynamic Particle Swarm Optimization (DPSO) combined with Fuzzy C-Means (FCM) [147] was applied to MRI and synthetic images, demonstrating robustness against noise and better performance than other competing algorithms. The integration of Harris Hawks Optimization (HHO) with chaotic initialization and altruism [148] was proposed for thresholding during brain MRI segmentation, showing improved results compared to existing methods. Additionally, the Monarch Butterfly Optimization (MBO)algorithm [149] was employed for medical image segmentation at multiple thresholds, yielding superior accuracy and speed, particularly at thresholds 3 and 4. In [150], an improved Ant Colony Optimization (ACO) algorithm was introduced for COVID-19 X-ray segmentation, leveraging swarm intelligence for more accurate results. Furthermore, a Harris Hawks Optimization (HHO) and Otsu method combination [151] demonstrated significant reductions in computational costs and convergence time while maintaining optimal results. Lastly, an improved Sparrow Search Algorithm [152], incorporating Levy flight and nonlinear inertia weight, was proposed for image threshold segmentation, and its performance on benchmark functions showed it to be superior to other algorithms.

3.2. Clustering-based image segmentation using metaheuristic

Cluster-based image segmentation involves grouping similar pixels together, employing algorithms such as K-means clustering**,** fuzzy clustering, and others [153, 154]. In [155], the Modified Fuzzy K-Means (MFKM) algorithm, combined with Bacteria Foraging Optimization (BFO), was used to identify the tumor region in Magnetic Resonance (MR) brain images by distinguishing between edema and normal tissue regions. The results of this method were compared with traditional Modified Fuzzy K-Means (MFKM)**,** Particle Swarm Optimization-based Fuzzy C-Means (FCM based on PSO), and conventional FCM algorithms, showing superior performance in MR brain image segmentation. A novel approach in [156] involved the use of Red

Fox Optimization (DRFO) with Kernel Fuzzy C-Means to detect skin cancer from dermoscopy images in the ISIC 2020 database, with this method yielding the best results compared to other competitive algorithms. In [157], the Shuffled Shepherd Optimization Algorithm (SSOA)was combined with the Salp Swarm Algorithm (SSA) to create SSSOA, which was applied alongside the Generative Adversarial Network (GAN) model for lung cancer detection in CT images. The SSSOA-based GAN method outperformed other algorithms in terms of accuracy**,** similarity, and the Dice coefficient. The authors in [158] integrated the Social Ski Driver (SSD) algorithm with SSSOA to detect lung cancer in CT images, using the Deep Renyi Entropy Fuzzy Clustering (DREFC) algorithm to segment lung lobes. This proposed method significantly improved accuracy**,** specificity, and sensitivity compared to other algorithms. The use of PSO and Mahalanob is distance in [159] enhanced the Fuzzy C-Means (FCM) algorithm, resulting in the Improved Spatial Fuzzy C-Means (IFCMS) method for image segmentation using simulated brain MRI images from the McConnell Brain Imaging Center database, demonstrating the efficiency of the approach. In [160], a Hybrid Sea Lion Squirrel Search Optimization (HSLnSSO) technique was employed to improve Fused Optimal Centroid K-means with K-Mediods Clustering (FOC-KKC) for dental caries segmentation, showing superior performance compared to other competitive methods.

3.3. Edge-based image segmentation using metaheuristic

Edge detection (ED) plays a pivotal role in image processing by identifying boundaries between regions with distinct gray-level intensities. This process is critical in various applications, such as detecting retinal blood vessels [161]. Classical ED operators including Prewitt, Sobel, Canny, Wallis, Laplacian, and Kirsch, each one of them employ specific convolution masks to emphasize edge features while suppressing irrelevant information, preserving the most salient image characteristics.

Several research efforts have harnessed nature inspired optimization algorithms to enhance edge detection and segmentation performance. For instance, in [162], the authors proposed an ant colony optimization (ACO)-based segmentation technique for processing MRI and iris images. The proposed method demonstrated superior segmentation quality, especially in images with complex local textures, outperforming traditional techniques. In [163], the researchers introduced a geometric deformable model that integrates edge- and region-based information with prior shape knowledge. This model employed genetic algorithms during its training phase to optimize level set parameters, and then applied the learned model during testing. The approach yielded improved accuracy in segmenting anatomical structures across diverse biomedical imaging modalities compared to state-of-the-art methods. A modified watershed segmentation (MWS) algorithm was implemented on a Xilinx Virtex-5 FPGA in [164] to segment brain tumors from MRI images. The hardware-accelerated implementation achieved more accurate results than conventional algorithms. The authors in [165] utilized ensemble deep neural networks combined with Particle Swarm Optimization (PSO) for segmenting the optic disc (OD) in retinal images. Their approach included variations of PSO such as a refined super-ellipse method, random and average leader-based searches, and an accelerated super-ellipse action. By incorporating Mask R-CNN, the technique effectively addressed the biases of individual networks, achieving superior performance in both unimodal and multimodal segmentation tasks. This method also demonstrated high

accuracy in detecting diabetic macular edema, a critical concern for elderly patients at risk of vision loss. In [166], the authors proposed a novel OD segmentation technique based on Markowitz portfolio optimization, validated using four datasets including Messidor, HRF, DRIVE, and a private dataset from Hospital UniversitarioSant Joan de Reus in Spain. The results confirmed the robustness and superiority of the proposed approach over competing techniques. Furthermore, [167] presented a Canny edge detector-based method for curve detection in brain tumor MRI scans, using data from the Neoplastic Disease section of the Whole Brain Atlas (Harvard Medical School). This method showed significant improvements over traditional active contour models such as Chan-Vese (CV), Local Binary Fitting (LBF), and Local Intensity Fitting (LIF), particularly in accurately identifying tumor boundaries.

3.4. Region-based image segmentation using metaheuristic

Region-based image segmentation techniques partition an image into distinct regions by grouping neighboring pixels that share similar characteristics [168]. These methods aim to ensure that each segmented region exhibits uniform properties, such as intensity or texture, while maintaining clear boundaries between dissimilar regions. This approach is particularly valuable in medical imaging, where precise localization of anatomical structures or pathological areas is critical. In [169], the authors proposed a novel framework for liver segmentation in abdominal CT images during the portal phase. The method integrates a multilevel local region-based Sparse Shape Composition (SSC) model with a hierarchical deformable shape optimization algorithm. The framework achieved slightly superior performance when compared to existing liver segmentation techniques. A different study in [170] utilized multi-objective particle swarm optimization (MOPSO) to enhance brain MRI segmentation. This approach addresses the limitations of traditional region-based active contour models and fuzzy entropy clustering. The method was evaluated using datasets from the Internet Brain Segmentation Repository (IBSR), real MR images from the McConnell Brain Imaging Center, and synthetic MR data. The results demonstrated improved robustness and segmentation accuracy across all tested datasets. In [171], the authors combined particle swarm optimization (PSO) with a robust graph-based (RGB) segmentation technique to detect breast tumors in ultrasound images. This hybrid approach outperformed both traditional regional segmentation methods and standard RGB techniques, delivering more precise tumor localization in challenging ultrasound imagery.

3.5. Deep learning and metaheuristic-based image segmentation

Deep learning has revolutionized image segmentation by leveraging convolutional neural networks (CNNs), U-Net, and transformer-based architectures to achieve state-of-the-art accuracy. However, deep learning models often require extensive labeled data, suffer from overfitting, and are computationally expensive. To address these challenges, researchers have integrated metaheuristic algorithms with deep learning for optimized segmentation. Metaheuristics can enhance deep learning models by optimizing hyperparameters, improving feature selection, and refining segmentation masks. For instance, GWO and PSO have been used to fine-tune CNN parameters, ensuring optimal learning rates and filter sizes. Additionally, GA and WOA have been applied to refine segmentation post-processing by optimizing threshold values and boundary refinement in deep learning-generated masks. Another emerging approach is using metaheuristics

for active learning, where algorithms like ACO select the most informative samples for training deep networks, reducing the annotation burden. By combining deep learning's feature extraction capability with the global search ability of metaheuristics, hybrid models achieve more accurate, computationally efficient, and adaptive segmentation, especially in medical and remote sensing applications.

Integrating metaheuristic algorithms with deep learning techniques has shown promise in enhancing image segmentation tasks. These hybrid approaches leverage the strengths of both methodologies to improve segmentation accuracy and efficiency. Below are several notable studies that have explored this integration:

Recent advancements in biomedical image segmentation have seen a growing integration of metaheuristics with deep learning models, yielding promising results across various medical imaging modalities.

In [172], the authors introduced a Hybrid Metaheuristics with Deep Learning-based Fusion Model for Biomedical Image Analysis (HMDL-MFMBIA). This framework encompasses image preprocessing, segmentation using Swin-UNet, and feature extraction through a fusion of deep learning architectures, specifically Xception and ResNet. A Hybrid Salp Swarm Algorithm (HSSA) is used for optimal hyperparameter selection, significantly enhancing the performance of biomedical image classification and analysis.

Similarly, in [173], a local-area contrast-correcting preprocessing technique was proposed. This method uses a brightness-preserving transformation function based on local neighborhood mean and standard deviation. To optimize transformation results, Differential Evolution (DE) and Artificial Bee Colony (ABC) algorithms were used as metaheuristic estimators for decision variables. Evaluation on four publicly available datasets demonstrated that images processed with DE-ABC showed superior segmentation performance compared to raw input data.

In [174], an adaptive multi-objective convolutional neural network, termed AdaResU-Net, was introduced for medical image segmentation. This architecture combines the U-Net framework with residual learning, and employs a Multi-objective Evolutionary Algorithm (MEA) to optimize hyperparameters while balancing segmentation accuracy and model complexity. The model was validated using the Promise12 dataset and cardiac MRI sequences from York University, outperforming traditional U-Net [1] and ResNet [175] models.

Further, [176] presented a fully evolutionary DenseRes model, which leverages dense and residual blocks along with evolutionary algorithms to automatically design optimal network architectures for medical image segmentation. Tested on six public MRI and CT datasets, the model demonstrated high segmentation accuracy while using a minimal number of parameters, surpassing both manually and automatically designed counterparts.

Despite the advantages of using metaheuristics in image segmentation such as robustness, flexibility, and the ability to avoid local optima, these techniques often face challenges including high computational complexity, slow convergence, and scalability issues, particularly with large-scale or real-time image data. A promising solution to these limitations is the parallelization of metaheuristic algorithms, which distributes computational tasks across multiple processors or GPUs, thereby accelerating convergence and enabling real-time or near real-time processing.

4. Parallel metaheuristic for algorithms optimization

On one side, optimization problems are witnessing a significant rise in complexity, accompanied by escalating demands for computational resources. Real-world scenarios frequently involve NP-hard problems that are both CPU- and memory-intensive. Although metaheuristic techniques offer a pragmatic approach to mitigating the computational burden of exhaustive search strategies, they often remain computationally expensive, particularly when addressing high-dimensional search spaces or dealing with intricate objective functions and constraint formulations. This challenge is further amplified by the emergence of increasingly sophisticated metaheuristic frameworks, such as hybrid and multi-objective variants, which themselves are becoming resource-intensive.

Conversely, the rapid evolution of computing technologies has ushered in a new era of parallelism. Advances in processor design, including multicore and specialized processing architectures, alongside the development of high-throughput network infrastructures (e.g., Myrinet, InfiniBand for LANs; optical networks for WANs) and scalable storage systems, have made parallel computing a mainstream solution. The inherent limitations of sequential processing, bounded by physical constraints such as thermodynamic limits and signal propagation delays, have accelerated this transition. Today, multicore processors are standard in even modest computing platforms, such as laptops and personal workstations, representing accessible forms of parallel architecture. Additionally, the continuous decline in cost-to-performance ratios, coupled with the proliferation of high-performance devices and low-latency communication technologies, has significantly bolstered the feasibility and appeal of parallel computing paradigms.

The scientific community is showing growing interest in distributed and massively parallel programming. Parallel metaheuristic optimization is a crucial area in artificial intelligence and computational science, aiming to solve complex optimization problems more efficiently [177]. The objective of using parallel metaheuristics is to enhance the efficiency, scalability, and accuracy of optimization algorithms by leveraging parallel computing. Traditional metaheuristics, can be computationally expensive, especially when applied to complex tasks like image segmentation as a difficult step in image processing. By parallelizing key operations such as fitness evaluation, solution updates, these algorithms can significantly reduce execution time while improving solution quality. Parallel implementations, particularly on multi-core CPUs and GPUs, enable a more extensive exploration of the search space, prevent premature convergence, and facilitate the handling of large-scale data efficiently.

This approach is crucial in applications like MRI segmentation, where real-time processing and high segmentation accuracy are essential for medical diagnostics. Finally, parallel metaheuristics refer to optimization algorithms that exploit parallel computing architectures to enhance performance. These algorithms can be categorized based on their level of parallelism, synchronization strategy, and computational model. The primary goal is to accelerate convergence and improve exploration capabilities in high-dimensional search spaces.

4.1. Parallel metaheuristic strategies

Parallelizing metaheuristic algorithms is essential for improving their efficiency and scalability, particularly in computationally intensive tasks like image segmentation, enhancement, and restoration. Several strategies have been proposed to exploit parallel hardware, ranging from fine-grained fitness evaluations to coarse-grained population division. Each approach offers unique advantages and trade-offs, depending on the image processing application and computational architecture.

4.1.1. Intra-population parallelism (Fine-Grained Parallelism)

The first and most widely used strategy is intra-population parallelism**,** also known as fine-grained parallelism (Figure 15). Each processing element (PE) communicates directly with its immediate neighbors (up, down, left, and right) through dedicated links, as shown by the solid lines. The dotted lines indicate potential or logical communication paths beyond immediate neighbors, emphasizing the local and spatial nature of interactions in such models. In this model, the individuals within a population are evaluated independently and simultaneously. This is especially effective for image processing tasks where fitness functions are computationally expensive, for instance, in multilevel thresholding, each individual may represent a different set of thresholds whose quality is assessed using entropy or edge-based metrics. Since these evaluations are independent, they are ideal for execution on parallel architectures such as GPUs using CUDA, OpenCL, or PyTorch. Each thread can evaluate a separate candidate solution, leading to significant speedups. This strategy is simple to implement and efficient in terms of parallel resource utilization, although synchronization is required during selection and population update phases, which may introduce some overhead.

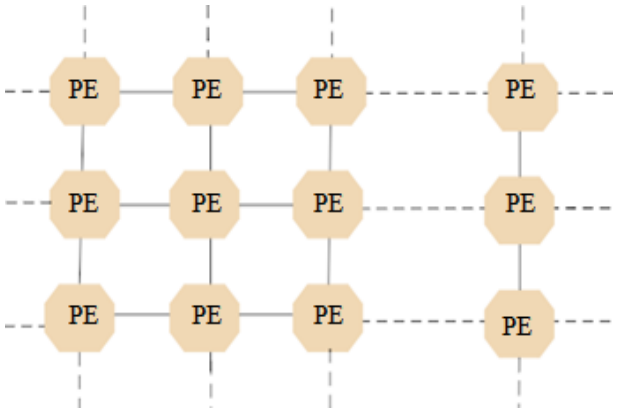


Figure 15. Fine-grained model.

4.1.2. Island strategy (Coarse-Gained Parallelsm)

Another popular method is the island model, which implements coarse-grained parallelism by dividing the population into multiple subpopulations (or islands) (see Figure 16). Each island runs an independent instance of the metaheuristic algorithm and explores the solution space autonomously. At regular intervals, a migration mechanism allows individuals to be exchanged

between islands to maintain diversity and avoid premature convergence. In image processing, the island model is particularly useful for high-dimensional tasks such as MRI segmentation, where each island may focus on different regions or resolution levels of the image. This model is well-suited for distributed computing environments, such as MPI-based clusters or cloud platforms, and offers excellent scalability. However, the effectiveness of this strategy depends on the design of the migration policy, as frequent communication between islands can increase overhead. The papers [178][179] uses the island model to parallelize their propositions.

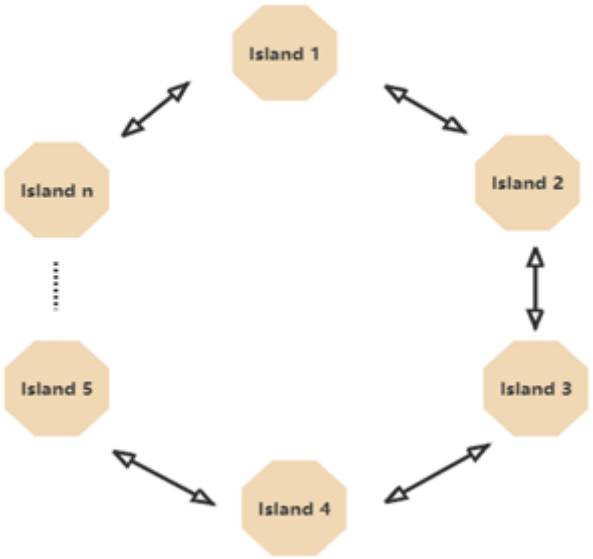


Figure 16. Coarse-gained model.

4.1.3. Master-Slave strategy

The master-slave model represents a task-parallelism approach in which a central master process manages the optimization flow, while several slave processes or threads are responsible for evaluating the fitness of individual solutions (Figure 17). This model is particularly advantageous when the fitness function is complex or time-consuming, such as in clustering-based segmentation or filter parameter optimization. The master generates and dispatches candidate solutions, and the slaves perform the necessary image processing computations in parallel. This model is often implemented using multicore CPUs with OpenMP or multiprocessing libraries in Python. It offers a straightforward architecture and centralized control, making it easier to implement and debug. However, the master can become a bottleneck when dealing with very large populations or high-frequency updates, which limits scalability. Authors in [180][181] used this mode of parallelization in their studies.

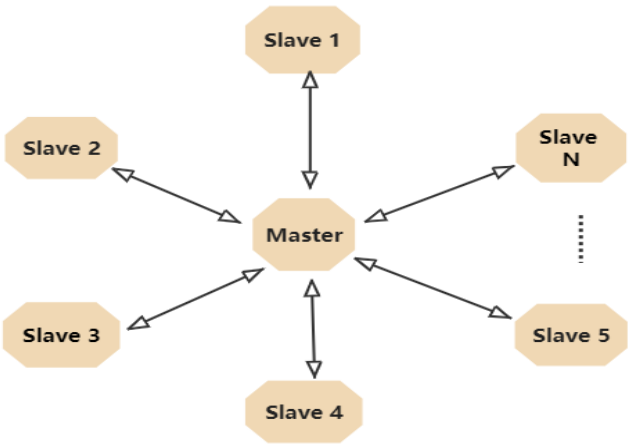


Figure 17. Master-slave model.

4.1.4. Hybrid Parallel Strategy

Finally, hybrid parallel strategies combine elements of both fine-grained and coarse-grained parallelism to exploit the full potential of modern computing architectures (Figure 18). For example, a hybrid model might use an island approach across multiple nodes, where each island internally performs intra-population parallelism using GPUs. This approach is highly adaptable and effective for large-scale image processing applications involving multi-objective optimization or multi-phase segmentation. Hybrid strategies are particularly beneficial when dealing with hierarchical image analysis, where different resolution levels or image patches can be processed in parallel at different granularity levels. However, these strategies are more complex to implement and require careful management of synchronization and communication overhead across different layers of parallelism.

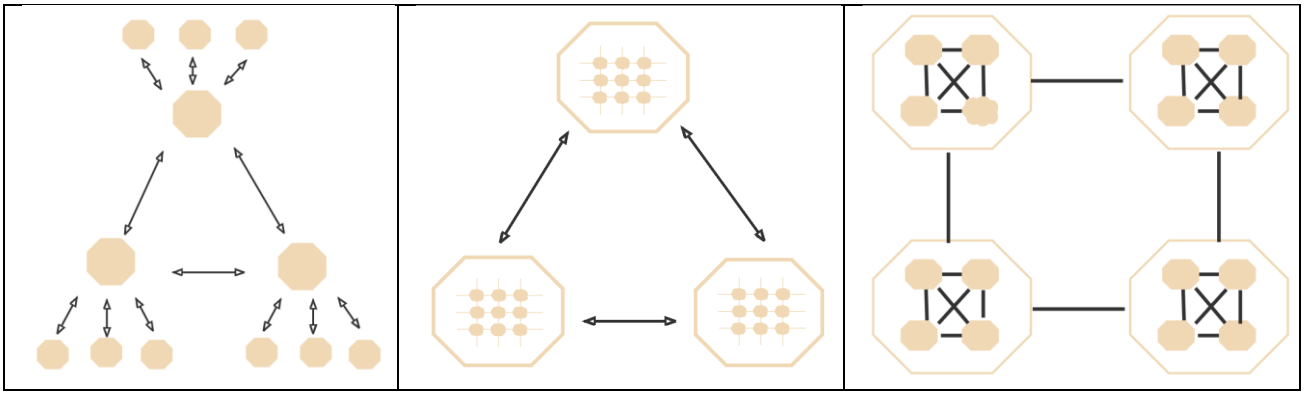


Figure 18. Hybrid model.

4.2. Parallel modelisation of metaheuristics

The reason that drives researchers to use parallel models in metaheuristics is the significant demand for computing power for the problems to be solved. Indeed, evaluating the objective function for each solution is generally the most costly operation in a metaheuristic [182]. We can

distinguish three parallelization models. Figure 19 describes the combination of these three models [183]:

4.2.1. The algorithm-level parallel model

This involves parallelizing algorithms (see Part 1 of Figure 19). This model does not depend on the problem being addressed. If the algorithms are independent of each other, parallelizing them only speeds up their execution. There is no improvement in the quality of the solutions found compared to the sequential model. On the other hand, if the algorithms are cooperative (as indicated by the two-way arrows in Figure 19, Part 1), parallelizing them can not only reduce their execution time but also improve the solutions found compared to the sequential model.

4.2.2. The iteration-level parallel model

This model consists of parallelizing neighbor generation at each iteration, regardless of the problem being addressed (see part 2 of Figure 19). It improves execution time, not the solutions found, compared to the sequential model. Its goal is to evaluate and generate neighbor solutions in parallel.

4.2.3. The parallel solution model

This model focuses on the parallel evaluation of a single solution (see part 3 of Figure 19). It depends on the specific problem being addressed. This is the lowest level, where individual solutions (or particles, individuals, configurations, etc.) within an iteration are evaluated or processed in parallel.

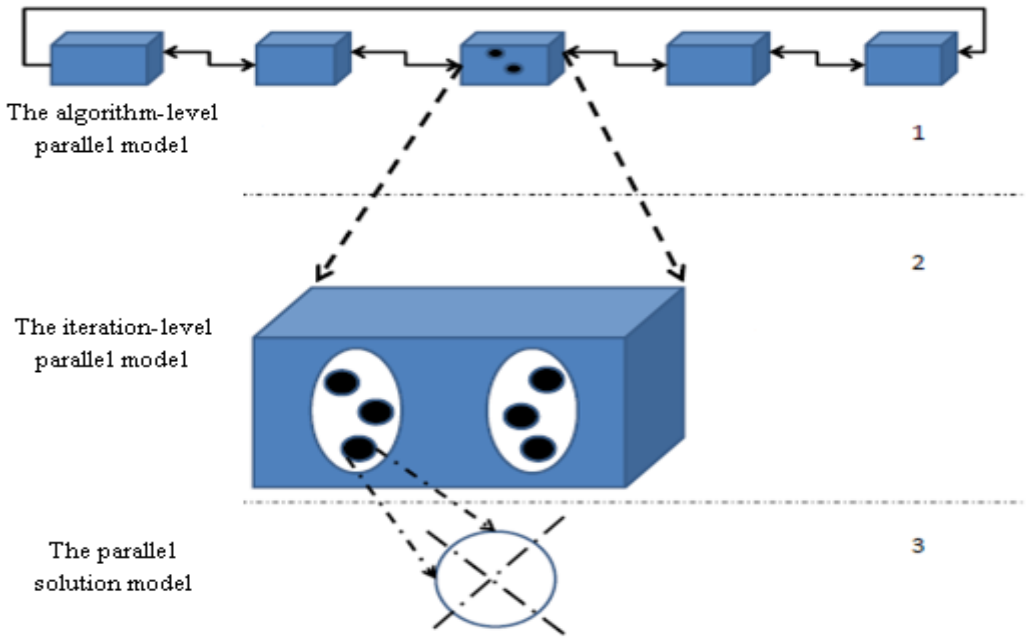


Figure 19. Combining the three parallel model.

4.3. Literature review about image segmentation based parallel metaheuristics

Parallel metaheuristic algorithms have become essential for image segmentation due to their ability to handle high-dimensional data and computational complexity efficiently. researchers have explored parallel implementations using multi-core CPUs, GPUs, and distributed computing frameworks to accelerate convergence and improve segmentation quality. Studies have shown that parallel optimization algorithm significantly reduce execution time while maintaining or enhancing segmentation accuracy. Similarly, hybrid parallel models combining metaheuristics with deep learning or fuzzy clustering have been proposed to improve segmentation robustness. These advancements highlight the importance of parallel metaheuristics in achieving fast, precise, and scalable image segmentation solutions.

Several recent studies have explored the integration of parallel computing and hybrid metaheuristic algorithms to improve the performance and scalability of image segmentation techniques, particularly in medical and high-resolution imaging domains.

In [15], a novel hybrid algorithm combining Harris Hawks Optimization (HHO) and Differential Evolution (DE) is presented for color image multilevel thresholding segmentation. The population is divided into two equal subpopulations, each optimized in parallel using HHO and DE. Otsu's method and Kapur's entropy serve as fitness functions for determining optimal threshold values. When benchmarked against seven state-of-the-art algorithms, the proposed HHO-DE method exhibited superior performance across various quality metrics, including average fitness, standard deviation (STD), Peak Signal to Noise Ration(PSNR**)**, Structure Similarity Index(SSIM), and Root Means Square Error (RMSE), establishing its effectiveness for multilevel thresholding.

In [16], the authors introduced a Parallel Multi-Verse Optimizer (PMVO) that incorporates a communication strategy into the original MVO algorithm. Initial solutions are randomly divided into groups, and information sharing is conducted periodically to mitigate premature convergence and local optima trapping. Tested on the CEC2013 test suite, PMVO outperformed conventional optimizers such as GWO, PSO, MVO, and Parallel PSO. When applied to multilevel image segmentation, PMVO consistently yielded higher-quality results than comparative methods.

A parallel compact Differential Evolution (pcDE) algorithm was proposed in [17] to improve optimization performance and was applied to image segmentation tasks. By dividing the population into multiple subgroups and introducing Optimal Elite (OE) and Mean Elite (ME**)** communication strategies, the algorithm enhanced convergence speed and solution stability. Results on standard benchmarks confirmed its superiority over traditional cDE. However, the parallel execution demanded considerable computational resources, especially for large-scale segmentation tasks.

In [184], a parallel Genetic Algorithm-based Fuzzy C-Means (GA-FCM) clustering algorithm was developed for brain MRI segmentation, using embedded GPUs. The method partitions the genetic population into subgroups and executes clustering in parallel across devices. The integration of Message Passing Interface (MPI) and CUDA ensured efficient load distribution. Experimental results showed up to 12× speedup over CPU-based FCM methods without compromising segmentation accuracy, making this approach highly suitable for large-scale medical imaging.

The work in [185] proposed a Big Data Architecture-based Biomedical Image Classification (BDA-BMIC) system, combining Genetic Algorithms (GA) and Gradient Approximation (GA) for feature selection and classification. The system is built on Apache Spark and Hadoop Distributed File System (HDFS), supporting parallel processing for large biomedical datasets. The use of parallelized SVMs and CNNs accelerated the classification process and achieved superior accuracy and computational efficiency, facilitating real-time image analysis.

Finally, in [186], a hybrid optimization technique is proposed by integrating the Coronavirus Optimization Algorithm (COVIDOA) with Harris Hawks Optimization (HHOA) for 2D and 3D medical image segmentation. The hybrid approach leverages the strengths of both algorithms to enhance convergence behavior. Otsu's method and Kapur's entropy are used as fitness criteria for optimal thresholding. The method is evaluated on IEEE CEC 2019 benchmark functions and various imaging modalities (MRI, CT, X-ray), showing improved results in terms of PSNR, SSIM, and NCC compared to seven other metaheuristic algorithms.

These studies collectively demonstrate that the integration of parallelism and hybrid metaheuristics not only improves segmentation accuracy but also significantly accelerates computational performance. As a result, such techniques are increasingly becoming essential for handling complex, high-resolution, and real-time biomedical image analysis tasks.

5. Conclusion

In this chapter, we presented the basic information about metaheuristics including definition and taxonomy of Metaheuristics algorithm, in addition, we have cited several works in which metaheuristics have been taken into account to improve the segmentation performance. Besides, current limitations of the segmentation methods have been pointed out, including the height computation time, from which we can have a clear view to go further. Even though there is a growing number of works in this field, using Parallel metaheuristics for solving the image segmentation problem has been proven to be successful and accelerate the execution time. By considering the limitations and taking advantages of current works, we propose in the following chapters some methods that are contributions in this area.

Parallelization is a powerful strategy to overcome the limitations of metaheuristic-based image segmentation. By distributing computations across multiple cores or GPUs, parallelization significantly reduces execution time, enhances scalability, and makes real-time processing feasible. Future research can focus on hybrid approaches that integrate deep learning with parallelized metaheuristics for even more robust image segmentation solutions.

Chapter 3

Machine learning and metaheuristic for image classification

1. Introduction

Image classification is a key challenge in computer vision, aiming to automatically assign labels to images based on their visual content. Several machine learning algorithms such as Support Vector Machines (SVM), Decision Trees, and k-Nearest Neighbors (k-NN) have long been applied to this task by learning patterns from hand-crafted features like color, texture, and shape descriptors. More recently, deep learning approaches particularly Convolutional Neural Networks (CNNs), have become the state-of-the-art due to their ability to automatically extract hierarchical features and deliver high classification accuracy. Despite their success, these models often suffer from limitations such as high computational cost, sensitivity to parameter settings, and the need for large labeled datasets. To address these challenges, metaheuristic optimization algorithms have been introduced as a complementary approach. Inspired by natural and evolutionary processes, metaheuristics like Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and Grey Wolf Optimizer (GWO) offer robust, flexible search strategies for solving complex optimization problems. In image classification, they have been effectively used to optimize feature selection, fine-tune model hyperparameters, and even enhance the training of neural networks. The integration of machine learning with metaheuristic optimization leads to hybrid systems that combine learning capabilities with global search efficiency, resulting in improved accuracy, faster convergence, and better generalization especially in scenarios involving high-dimensional data, imbalanced classes, or noisy inputs. This synergy has shown significant potential in domains such as medical diagnostics, satellite imagery analysis, and facial recognition, where reliable and efficient image classification is essential.

In this chapter, we explore the application of machine learning and metaheuristic algorithms in the field of image processing, with a particular focus on classification tasks. Machine learning techniques, including both traditional models and deep learning approaches, have proven effective in analyzing and interpreting visual data for various image processing applications. Metaheuristic algorithms, on the other hand, offer powerful optimization strategies that enhance model performance by selecting the most relevant features and tuning parameters efficiently. The integration of these two paradigms enables more accurate and robust image analysis systems. Following this general overview, we focus specifically on the use of a Modified Grey Wolf Optimizer (MGWO) for feature selection in the classification of

breast cancer. By employing MGWO, the aim is to identify the most informative features from medical images, thereby improving the accuracy and efficiency of the classification process. This hybrid approach demonstrates the potential of combining bio-inspired optimization with machine learning to address complex medical imaging problems.

2. Machine learning

Machine Learning (ML) is a subfield of artificial intelligence (AI) that focuses on developing algorithms and systems that can learn from data and improve their performance over time without being explicitly programmed. Instead of relying on hard-coded instructions, ML systems identify patterns and relationships within data, allowing them to make predictions, decisions, or classifications based on new inputs. ML spans several approaches, including supervised learning (learning from labeled data), unsupervised learning (finding structure in unlabeled data), semi-supervised learning (a mix of labeled and unlabeled), and reinforcement learning (learning through trial and error in an interactive environment). Its power lies in its adaptability. ML systems improve as they are exposed to more data, making them useful for a wide range of tasks such as image recognition, natural language processing, predictive analytics, and autonomous systems. In essence, machine learning enables computers to learn from experience, much like humans do, transforming how we solve complex, data-driven problems.

2.1. History of machine learning

The term Machine Learning was originally introduced by Arthur Samuel in 1952 [187]. Five years later, in 1957, Frank Rosenblatt at the Cornell Aeronautical Laboratory integrated Donald Hebb's theory on neural activity with Samuel's foundational ideas, leading to the development of the perceptron, a pioneering model in neural computation. By 1967, the introduction of the nearest neighbor algorithm marked a key advancement in early pattern recognition. This algorithm was notably applied to route mapping and emerged as one of the initial strategies for addressing the traveling salesman problem, aimed at determining the most efficient path. During the 1960s, research revealed that incorporating multiple layers within perceptron architectures significantly enhanced their computational capabilities. This breakthrough in multilayer neural networks opened new avenues in the field of neural network research [187].

As noted in [188], the evolution of machine learning is closely tied to the broader pursuit of artificial intelligence. From AI's inception as a scholarly field, a subset of researchers focused on enabling machines to learn from data. Their approaches ranged from symbolic reasoning methods to early neural network models, such as the perceptron. Many of these early neural approaches were later recognized as variations of generalized linear models from statistical theory. Additionally, probabilistic reasoning techniques, particularly in contexts like automated medical diagnostics, played a crucial role in early machine learning applications. Machine learning emerged as a distinct discipline and began to gain significant momentum during the 1990s. Its focus shifted from the broader and often elusive objective of achieving

artificial intelligence to addressing well-defined, practical problems. This evolution was marked by a departure from the symbolic reasoning approaches traditionally associated with AI, in favor of data-driven methods rooted in statistics and probability theory [189].

Although machine learning and data mining are distinct areas, they share considerable methodological overlap. Many data mining tasks utilize machine learning techniques, albeit often with different end goals. While data mining is primarily concerned with extracting patterns from large datasets, machine learning emphasizes the development of models that can make accurate predictions or decisions based on data.

A core aspect of machine learning is its strong connection to optimization theory. Most learning tasks are framed as the minimization of a loss function over a training dataset. This function quantifies the error or discrepancy between a model's predictions and the actual outcomes, such as assigning incorrect labels in a classification task. Although optimization techniques are capable of reducing loss within the training set, machine learning distinguishes itself through its emphasis on generalization: the model's ability to maintain accuracy on new, unseen data [190]. Machine learning algorithms are designed to ingest and analyze data to uncover underlying patterns related to individuals, organizational activities, transactions, or events. In the subsequent sections, we explore the different types of real-world data and categorize the major branches of machine learning algorithms.

2.2. Types of data

In most cases, the availability and accessibility of data are fundamental to the development of machine learning models and real-world data-driven systems [191, 192]. Data can exist in multiple formats including structured, semi-structured, and unstructured [193, 194]. Additionally, metadata, which refers to data that provides information about other data, plays a crucial role in data management and interpretation. The following provides a brief overview of these data types:

- **Structured Data**: This type of data adheres to a predefined schema and follows a consistent format, making it highly organized and easily searchable by humans and computational systems alike. Structured data is typically stored in relational databases using tabular formats. Examples include personal details (such as names, addresses, and birthdates), financial records (like credit card numbers and stock prices), and geographic coordinates.
- **Unstructured Data**: Unlike structured data, unstructured data lacks a standardized format, which poses challenges in terms of storage, processing, and analysis. This category includes a wide array of content, often in the form of text and multimedia. Examples encompass sensor readings, email content, blog posts, wiki pages, word documents, PDFs, audio clips, video files, digital images, slide presentations, web content, and various business documents.
- **Semi-structured Data**: Although not stored in traditional relational databases, semi-structured data contains organizational elements such as tags or markers that provide a level of structure. This facilitates easier processing compared to unstructured data.

Examples include HTML and XML files, JSON documents, and data stored in NoSQL databases.

- **Metadata**: Metadata is a special category that refers to information about other data, rather than being raw data itself. While data represent facts or values related to entities, metadata provides context or descriptors that enhance the interpretability of that data. For instance, the metadata of a document may include details such as its author, creation date, file size, and format, thereby enriching its informational value for users.

In the fields of machine learning and data science, researchers frequently utilize a variety of well-established datasets tailored to specific application domains. Examples include cybersecurity datasets such as UNSW-NB15 [195], ISCX'12 [196], CICDDoS2019 [197], and Bot-IoT [198], mobile device datasets like phone call logs [199, 200] and SMS logs [201], IoT-related datasets [202, 203], as well as data from sectors such as agriculture, e-commerce [204], and healthcare including datasets on heart disease [205], diabetes mellitus [206], and COVID-19 [207]. These datasets reflect the diversity of data types mentioned earlier (structured, semi-structured, unstructured, and metadata) which vary depending on the application context.

Effectively analyzing such datasets within their respective domains involves uncovering meaningful patterns and insights that can drive the development of intelligent, real-world systems. To achieve this, different machine learning methodologies are employed, each chosen based on their unique learning paradigms and capabilities. A detailed discussion of these machine learning approaches follows in the subsequent section.

2.3. Types of machine learning techniques

Machine learning algorithms are broadly classified into five major categories: supervised learning**,** unsupervised learning**,** semi-supervised learning**,** reinforcement learning**,** and deep learning [208], as illustrated in Figure 20. Each of these learning paradigms offers distinct methodologies and is suited to different types of real-world problem-solving scenarios. A brief overview of each category and its practical applications is provided below.

2.3.1. Supervised

Supervised learning refers to a fundamental machine learning paradigm where the goal is to learn a mapping function from inputs to outputs using a dataset composed of labeled examples. In this approach, the algorithm is trained on input-output pairs, enabling it to infer a function that can generalize to unseen data. This method is inherently task-driven, meaning it is designed to achieve specific predictive outcomes based on the nature of the input data. The two most common types of supervised learning tasks are classification, which involves assigning input data to discrete categories, and regression, which predicts continuous numeric values. For example, text classification, such as determining the sentiment of a tweet or categorizing a product review, the latter is a typical application of supervised learning. Figure

21 illustrates the essential stages involved in a supervised learning workflow, from training on labeled data to applying the learned model for prediction on new inputs.

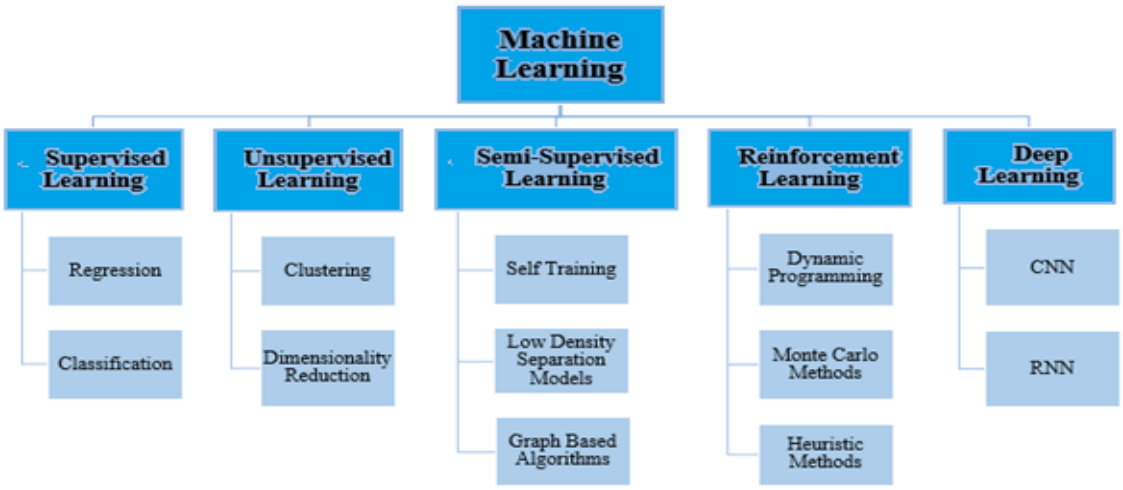


Figure 20. Various types of machine learning techniques [208].

Figure 21. Different stages of the supervised learning [208]

2.3.2. Unsupervised

Unsupervised learning involves the analysis of datasets that lack labeled outputs, relying entirely on the inherent structure of the data, making it a data-driven rather than task-driven approach. Unlike supervised learning, it does not require human-labeled examples, allowing algorithms to autonomously discover patterns, relationships, or structures within the data. This learning paradigm is particularly effective for tasks such as feature extraction, trend discovery, and exploratory data analysis, where the objective is to gain insights or organize information without predefined categories. Common unsupervised tasks include clustering (grouping similar data points), density estimation, dimensionality reduction, feature learning, association rule mining, and anomaly detection. Unsupervised learning is foundational in applications like customer segmentation, topic modeling, fraud detection, and recommendation systems, where understanding hidden structures is crucial for knowledge discovery. Figure 22 represent different steps of the unsupervised learning.

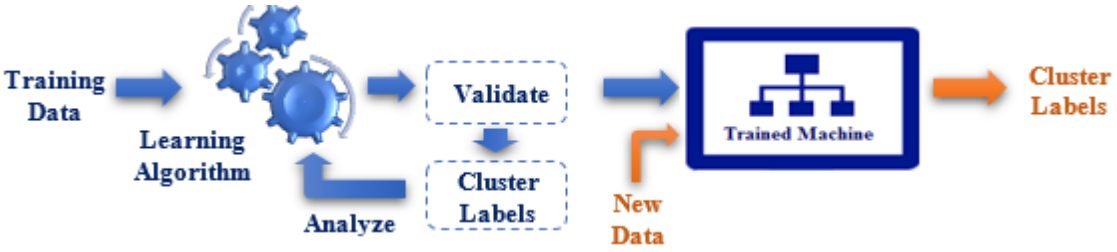


Figure 22. Different stages of the unsupervised learning [208].

2.3.3. Semi-supervised

Semi-supervised learning represents a middle ground between supervised and unsupervised learning approaches. It leverages a combination of both labeled and unlabeled data during training, effectively blending the strengths of each paradigm. This hybrid approach is particularly valuable in real-world scenarios where labeled data is scarce, expensive, or time-consuming to obtain, while large volumes of unlabeled data are readily available [209]. The primary objective of semi-supervised learning is to enhance predictive performance by utilizing the structural information from unlabeled data alongside the limited labeled data, achieving better results than would be possible using labeled data alone. Common application domains for semi-supervised learning include machine translation, fraud detection, automated data labeling, and text classification, among others. This approach is especially effective when unlabeled data can help the model learn the underlying distribution more accurately. Figure 23 illustrates the various stages involved in a semi-supervised machine learning process.

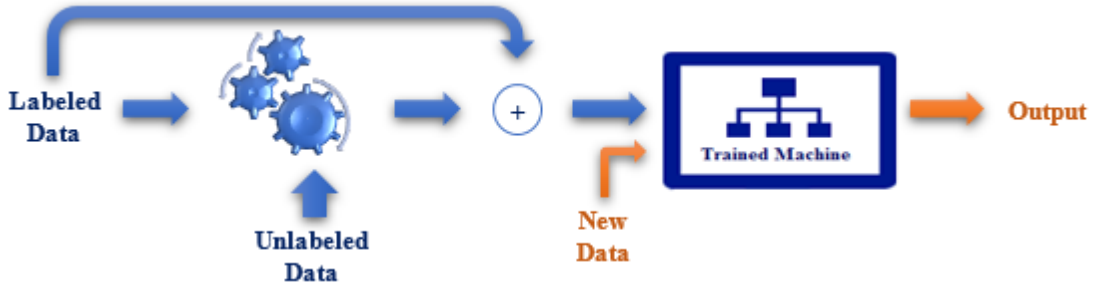


Figure 23. Different stages of the semi-supervised learning [208].

2.3.4. Reinforcement

Reinforcement learning (RL) is a machine learning paradigm in which software agents or machines learn to make optimal decisions through interaction with an environment [210]. Unlike supervised approaches, RL is environment-driven, relying on feedback in the form of rewards or penalties to guide learning. The agent's objective is to learn a strategy(or policy) that maximizes cumulative reward over time while minimizing potential risks [209]. This learning model is particularly effective in scenarios that require sequential decision-making, adaptation, and continuous improvement. RL is instrumental in powering advanced AI systems, especially in areas such as robotics, autonomous vehicles, intelligent manufacturing,

and logistics optimization. However, it is generally not suited for simple or well-defined problems, where traditional learning techniques may suffice. Ultimately, the effectiveness of any machine learning model (be it reinforcement-based or otherwise) depends on aligning the learning strategy with the nature of the data and the intended outcome. As such, choosing the appropriate learning technique is essential to building robust, intelligent systems across diverse application domains.

2.3.5. Deep learning

Since its resurgence in 2006, deep learning has rapidly gained momentum and become a cornerstone in hundreds of research efforts across diverse fields, from information processing to artificial intelligence. As a specialized branch of machine learning, deep learning relies on models that learn from multiple layers of abstraction, enabling them to capture intricate and non-linear relationships within data. At the heart of deep learning lies a hierarchical architecture, where high-level features are progressively built upon lower-level ones, allowing the system to automatically learn complex representations. This layered structure is what gives deep learning its "deep" designation. Notably, many deep learning frameworks are rooted in unsupervised learning representations [211]. Deep learning sits at the intersection of various disciplines, including neural networks, graphical models, optimization, artificial intelligence, pattern recognition, and signal processing. Its rise in popularity can be attributed to several key factors: the exponential growth in computational power (particularly through GPUs), the availability of vast amounts of training data, and its pivotal role in pushing the boundaries of modern machine learning especially in fields like computer vision, natural language processing, and speech recognition.

2.4. Machine learning workflow

Before diving into the detailed workflow of machine learning, it is important to understand the growing importance of intelligent systems in handling complex image processing tasks. With the rapid increase in visual data across fields such as healthcare, security, and remote sensing, there is a strong demand for automated methods capable of interpreting and analyzing images with high accuracy. Machine learning has emerged as a powerful tool in this regard, offering the ability to learn patterns from data and make informed predictions. This section describes steps involved in the machine learning workflow for image processing applications (Figure 24).

- **Data Collection:** The first step in any machine learning workflow is collecting relevant data, which forms the foundation upon which models are built. This data can come from a variety of sources, including internal databases, APIs, web scraping, IoT sensors, user interactions, or public datasets. The quality, quantity, and variety of data gathered have a direct impact on the model's ability to learn meaningful patterns. For instance, biased or incomplete data can lead to inaccurate or unfair models. It's also important to consider the format and structure of the data, whether it's structured (e.g., tables), semi-structured (e.g., JSON/XML), or unstructured (e.g., text, images). Ethical

considerations, such as user privacy and data consent, are especially critical during this phase, particularly when dealing with sensitive or personal information.

- **Data Preprocessing:** Raw data is often messy, inconsistent, and incomplete, which makes preprocessing a critical step. This stage involves cleaning the data by handling missing values, correcting inconsistencies, removing duplicates, and identifying outliers that may distort analysis. Next, the data is transformed into a format suitable for machine learning algorithms. Categorical variables are encoded into numerical values, numerical features are normalized or standardized, and irrelevant or redundant features may be dropped. Feature engineering is also key here, where new features are created from existing ones to enhance the model's predictive power. For high-dimensional data, techniques like Principal Component Analysis (PCA) may be used to reduce dimensionality, improving both computational efficiency and model performance.

- **Data Splitting:** Once preprocessing is complete, the dataset is divided into distinct subsets to enable objective model evaluation. The data is typically split into three parts: training, validation, and test sets. The training set is used to teach the model, allowing it to learn patterns from the data. The validation set helps fine-tune the model's hyperparameters and prevent overfitting by providing feedback on how well the model generalizes to unseen data during training. Finally, the test set acts as a final checkpoint to evaluate the model's real-world performance on completely unseen data. This separation ensures that the model's evaluation is fair, unbiased, and not influenced by data it has already encountered.

- **Model Selection:** With the data prepared, the next step is to choose an appropriate algorithm based on the nature of the problem, whether it's classification, regression, clustering, or reinforcement learning. For example, logistic regression, decision trees, support vector machines (SVM), and neural networks are commonly used for classification tasks, while linear regression is used for predicting continuous outcomes. In unsupervised settings, clustering algorithms like K-Means, DBSCAN, or hierarchical clustering help discover natural groupings in the data. The choice of algorithm depends on several factors, including the size of the dataset, the number of features, interpretability requirements, training time, and the expected accuracy. Often, multiple models are trained and compared before settling on the best-performing one.

- **Model Training:** Training a machine learning model involves feeding it the training data so it can learn to make predictions or decisions without being explicitly programmed. During this phase, the model adjusts its internal parameters to minimize the difference between its predictions and the actual outcomes, a process guided by a loss function. Optimization techniques such as gradient descent are used to iteratively improve the model's performance. Depending on the complexity of the model and the size of the dataset, training can range from seconds to hours or even days. It's essential

to monitor the training process to prevent issues like overfitting, where the model becomes too tailored to the training data and performs poorly on new data.

- **Model Evaluation:** After training, the model's effectiveness is assessed using performance metrics relevant to the task. For classification problems, metrics like accuracy, precision, recall, F1-score, and ROC-AUC are commonly used, while regression tasks may use mean squared error (MSE), mean absolute error (MAE), or R-squared. The evaluation is done on the test set, which contains data the model hasn't seen before, providing a true measure of how it might perform in real-world scenarios. To further validate the model's robustness, techniques like k-fold cross-validation are used, where the dataset is split into k parts, and the model is trained and tested k times on different combinations. This helps ensure the results are not biased by a specific data split.

- **Model Optimization:** Model optimization focuses on refining the model to achieve better performance. This includes tuning hyperparameters, predefined settings that influence the training process but are not learned from the data, such as learning rate, depth of trees, or number of neurons in a neural network. Techniques like grid search, random search, or Bayesian optimization are used to find the best hyperparameter combinations. Regularization methods like L1 (Lasso) and L2 (Ridge) are applied to reduce overfitting by penalizing overly complex models. Ensemble learning methods, such as bagging (Random Forests) or boosting (XGBoost), can be employed to combine multiple weak models into a stronger one. Effective optimization can significantly enhance both accuracy and generalization.

- **Model Deployment :** Once the model meets performance expectations, it is deployed into a real-world environment where it can provide predictions on new, incoming data. Deployment involves integrating the model into an application or service, often via REST APIs, microservices, or cloud platforms. The model must be scalable, secure, and able to respond in real time or batch mode depending on the use case. Monitoring infrastructure is also important at this stage to track system health, latency, and throughput. Additionally, considerations like version control, rollback mechanisms, and containerization (e.g., using Docker) ensure that the model operates reliably and can be updated or replaced as needed.

- **Model Maintenance:** The final step in the ML lifecycle is ongoing model maintenance. As the environment or input data evolves, the model's performance can degrade, a phenomenon known as concept drift. To combat this, models must be continuously monitored for accuracy, fairness, and reliability. If performance drops, retraining the model with recent data may be necessary. Maintenance also includes periodically re-evaluating features, retraining with additional data, updating dependencies, and testing against new edge cases. Tools for A/B testing can be used to compare new models with existing ones in live environments, ensuring only

improvements are deployed. A well-maintained model ensures long-term value and alignment with evolving user needs.

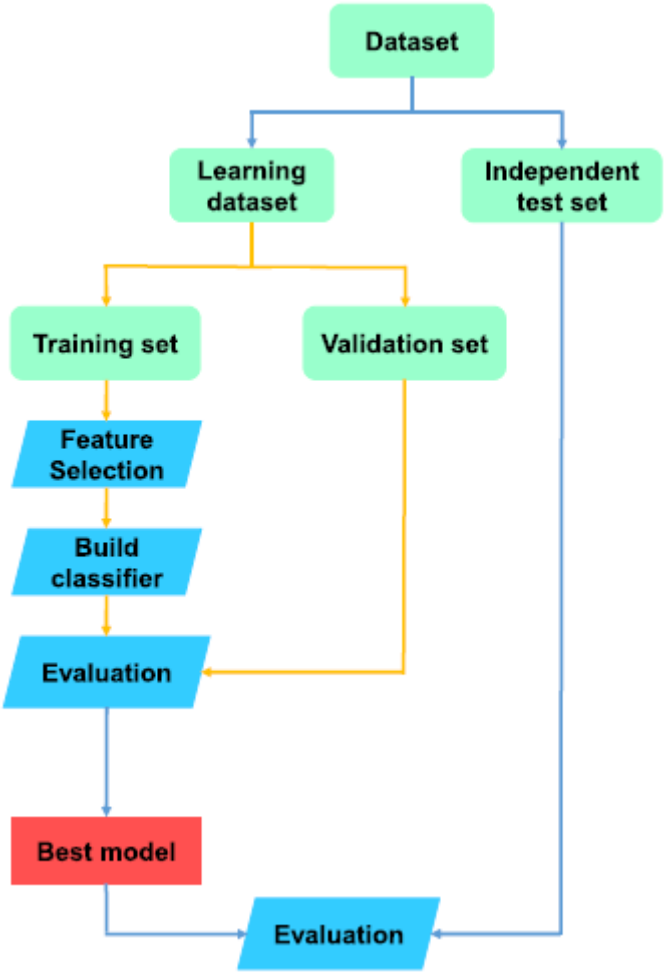


Figure 24. Flowchart of machine learning process.

2.5. Machine learning tasks

Machine learning (ML) encompasses a diverse range of tasks that allow models to learn from data and make intelligent decisions. These tasks can be broadly categorized into classification, regression, clustering, anomaly detection, dimensionality reduction, and reinforcement learning, each with unique applications across different domains.

- **Classification:** Classification is a supervised learning task where the model assigns input data to predefined categories. It is widely used in applications like spam detection (classifying emails as spam or not), medical diagnosis (predicting diseases based on symptoms), and image recognition (identifying objects in images). Algorithms such as Support Vector Machines (SVM), Random Forest (RF), and Neural Networks are commonly employed for classification tasks.

- **Regression:** Regression is another supervised learning task that predicts a continuous numerical value based on input features. It is commonly applied in house price prediction, stock market forecasting, and weather prediction. Techniques such as Linear Regression, Decision Trees, and Deep Learning-based regression models help establish relationships between variables to make accurate predictions.
- **Clustering:** Clustering is an unsupervised learning task that groups data points into clusters based on similarity. It is used in applications like customer segmentation (grouping customers by behavior), anomaly detection (identifying unusual patterns in network security), and gene expression analysis in bioinformatics. Popular clustering algorithms include K-Means, DBSCAN (Density-Based Spatial Clustering), and Hierarchical Clustering.
- **Anomaly Detection:** Anomaly detection identifies outliers or rare events that differ significantly from normal data patterns. It is widely used in fraud detection (identifying suspicious financial transactions), network security (detecting cyber-attacks), and industrial maintenance (predicting equipment failures). Methods such as One-Class SVM, Isolation Forests, and Autoencoders are commonly used for anomaly detection.
- **Dimensionality Reduction :** Dimensionality reduction is crucial for handling high-dimensional datasets by reducing the number of features while preserving essential information. It is commonly applied in imag**e** compression, text processing (reducing feature space in Natural Language Processing), and medical diagnostics (extracting key biomarkers from genomic data). Techniques such as Principal Component Analysis (PCA), t-Distributed Stochastic Neighbor Embedding (t-SNE), and Autoencoders help simplify complex datasets.
- **Reinforcement Learning:** Reinforcement learning (RL) is a task where agents learn optimal strategies by interacting with an environment and receiving rewards for desired actions. RL is extensively used in robotics (autonomous navigation), game playing (AlphaGo, OpenAI's Dota 2 bot), and recommendation systems (personalized content recommendations). Key RL algorithms include Q-Learning, Deep Q Networks (DQN), and Policy Gradient Methods.

Finally, machine learning tasks serve as the foundation for intelligent systems across multiple industries. The choice of ML task depends on the problem type, data characteristics, and desired outcomes. As ML research advances, hybrid approaches combining supervised, unsupervised, and reinforcement learning continue to enhance performance in real-world applications.

2.6. Classification analysis

Classification is a fundamental supervised learning approach in machine learning, often framed as a predictive modeling task wherein the objective is to assign predefined class labels to input instances [193]. Formally, it involves learning a function (F) that maps input features (X) to target outputs (Y), which may represent categories, labels, or classes. This process can be applied to both structured and unstructured datasets to determine the appropriate class of

new observations. A common example is email spam detection, where messages are categorized as either “spam” or “not spam”. In the following sections, we outline several prevalent classification tasks.

- **Binary Classification:** Binary classification involves predictive tasks where each instance is categorized into one of two distinct classes, such as “true/false” or “yes/no” [193]. Typically, one class represents the default or normal state, while the other denotes an abnormal or exceptional condition. For example, in medical diagnostics, “cancer not detected” may represent the normal condition, whereas “cancer detected” signifies an abnormal outcome. Similarly, the task of distinguishing between “spam” and “not spam” in email filtering exemplifies binary classification.
- **Multiclass Classification:** Multiclass classification extends the binary framework to problems involving more than two class labels [193]. Unlike binary classification, it does not rely on a dichotomy of normal versus abnormal outcomes. Instead, instances are assigned to one of several possible categories. A typical example is the classification of network intrusions in the NSL-KDD dataset [212], where attacks are categorized into four distinct classes: Denial of Service (DoS), User to Root (U2R), Remote to Local (R2L), and Probing.
- **Multi-label Classification:** Multi-label classification addresses scenarios where a single instance may simultaneously belong to multiple classes or categories, making it a generalization of multiclass classification. Here, labels are not mutually exclusive and may follow a hierarchical structure, such as in multi-level text classification. For example, a Google News article may be concurrently tagged under “technology,” “city name,” and “latest news.” This paradigm necessitates sophisticated learning algorithms capable of predicting multiple, potentially overlapping labels [213].

Numerous classification algorithms have been developed and extensively studied in the fields of machine learning and data science [214]. In the subsequent section, we provide an overview of the most widely adopted classification techniques across various application domains.

2.6.1. Support Vector Machine (SVM)

A Support Vector Machine (SVM) is a supervised learning algorithm widely utilized for both classification and regression tasks [2][215][216]. Although applicable to regression problems, SVM is particularly effective and predominantly used for classification. The core objective of the SVM algorithm is to determine the optimal hyperplane that best separates data points belonging to different classes within an N-dimensional feature space. This separation is achieved by maximizing the margin between the nearest data points of opposing classes, referred to as support vectors. The dimensionality of the hyperplane corresponds to the number of input features: for example, with two features, the hyperplane is a line with three features, it becomes a two-dimensional plane. As the number of features increases

beyond three, the geometric complexity of the hyperplane escalates, making it more challenging to visualize.

2.6.1.1. Support Vector Machine Terminology

In the context of Support Vector Machines (SVM), several key terminologies define the mechanics and theoretical foundation of the algorithm. At the core is the hyperplane, which is a decision boundary that separates the feature space into distinct classes. In an N-dimensional space, this hyperplane has N−1 dimensions and is mathematically represented by a linear equation. Support vectors are the data points that lie closest to the hyperplane, they are critical because they directly influence the position and orientation of the hyperplane. The margin refers to the distance between the hyperplane and the nearest support vectors from each class. The objective of the SVM algorithm is to maximize this margin, thereby improving the model's generalization ability and reducing overfitting. A hard margin SVM assumes that the data is linearly separable and aims for perfect classification with no misclassification, while a soft margin SVM allows for some misclassification in exchange for better performance on non-linearly separable data. The degree of tolerance to misclassification is controlled by a regularization parameter C, where a smaller value allows more misclassification and a larger value enforces stricter separation. In cases where data cannot be linearly separated, kernel functions are employed to implicitly map the original input space into a higher-dimensional space, where a linear separation is possible. Common kernel types include the linear kernel**,** polynomial kernel, and Radial Basis Function (RBF) kernel**.** The decision function is the model's output that determines which side of the hyperplane a new data point falls on, thereby assigning it to a specific class. Together, these components enable SVMs to be powerful and flexible tools for both linear and non-linear classification tasks.

2.6.1.2. Support vector machine algorithm working

In Support Vector Machines (SVM), an optimal hyperplane is typically defined as the one that maximizes the margin of separation between the two classes. This hyperplane, known as the maximum-margin hyperplane or hard margin, is constructed to achieve the greatest possible distance between itself and the closest data points from each class, these critical points are termed support vectors. The rationale behind maximizing the margin is to enhance the model's ability to generalize to unseen data by minimizing the risk of misclassification. A wider margin implies a more confident decision boundary, which in turn improves the robustness of the classifier, especially in linearly separable datasets.

The optimal hyperplane in a Support Vector Machine (SVM) is chosen such that the distance to the nearest data point from each class is maximized. This distance defines the margin, and the corresponding hyperplane is referred to as the maximum-margin hyperplane or hard margin, provided that the data is linearly separable. Among the candidate hyperplanes illustrated in Figure 25**,** the hyperplane labeled L2 satisfies this criterion by maintaining the greatest margin from the closest data points on either side. Therefore, L2 is selected as the optimal decision boundary.

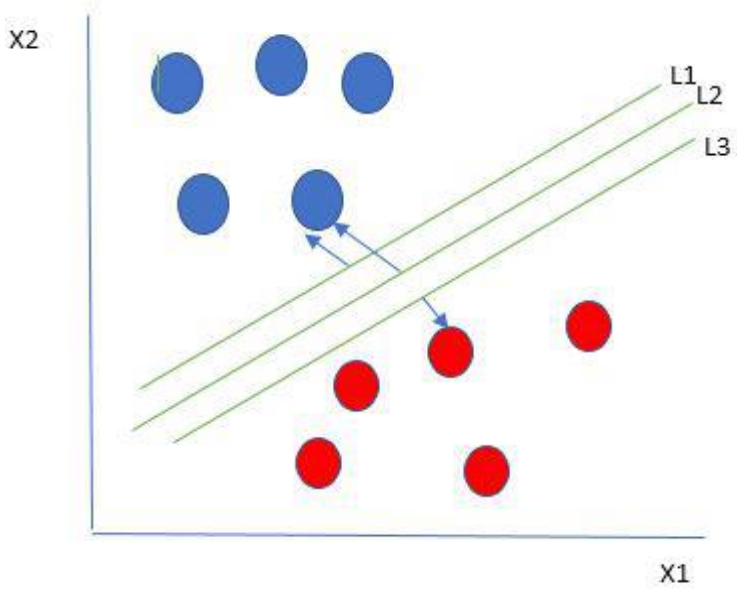


Figure 25. Multiple hyperplanes separate the data from two classes.

2.6.2. Decision Tree algorithm (DT)

A Decision Tree is a widely used supervised learning algorithm in machine learning, designed to predict outcomes based on input features [217]. Structurally, it resembles a tree, where internal nodes represent decision rules or tests on specific attributes, branches denote the outcomes of these tests (i.e., attribute values), and leaf nodes correspond to the final predictions or class labels. The algorithm operates by recursively partitioning the data into subsets based on the most significant feature at each step, thereby creating a model that is both interpretable and effective. Decision trees are versatile and can be applied to both classification and regression tasks, making them suitable for a wide range of applications in machine learning. Their ability to model non-linear relationships and handle both numerical and categorical data contributes to their popularity across various domains.

2.6.2.1. Decision tree terminologies

In the context of decision trees, several key terminologies define the structure and learning process of the model. At the top of the tree is the root node, which represents the initial feature or attribute upon which the first split is made. From this node, the data is recursively divided into subsets based on decision rules, which are derived from the feature values. The points where these splits occur are called internal nodes or decision nodes, and each one tests a specific attribute to determine the path the data should follow. The branches emerging from these nodes correspond to the possible outcomes or values of the tested attribute. The terminal points of the tree, known as leaf nodes or terminal nodes, represent the final output or prediction, which could be a class label in classification tasks or a numerical value in regression tasks. The path from the root node to a leaf node constitutes a decision path, encapsulating a series of rules that lead to a prediction. Another important concept is

information gain (or Gini impurity), which is used as a criterion to select the optimal attribute for splitting; it quantifies how well a given feature separates the data into distinct classes. Overfitting is a common challenge in decision trees, where the model becomes too complex and captures noise in the training data, leading to poor generalization. Techniques such as pruning (removing branches that add little predictive value) are employed to counteract this. Overall, understanding these terminologies is essential for interpreting, constructing, and optimizing decision tree models effectively.

2.6.2.2. Decision trees working

A Decision Tree is a popular machine learning model used for both classification and regression tasks. It constructs a tree-like hierarchical structure, where each internal node represents a decision based on a feature or attribute, each branch corresponds to the outcome of that decision, and each leaf node denotes the final prediction or result. The process of constructing a decision tree typically involves the following steps:

1. **Root Node Selection**:
    - Start with the entire dataset as the root.
    - Select the feature that best splits the dataset based on a specific metric (e.g., Gini Impurity, Information Gain, Mean Squared Error for regression).
2. **Splitting**:
    - Partition the dataset into subsets based on the selected feature and its threshold values.
    - Repeat the process for each subset, creating child nodes.
3. **Stopping Criteria**:
    - Stop splitting when:
        - All data points in a subset belong to the same class.
        - A predefined depth is reached.
        - Splitting no longer improves performance significantly.
4. **Prediction**:
    - For classification, the majority class in a leaf node is the predicted class.
    - For regression, the average of the target values in the leaf node is used.

2.6.3. Random forest algorithm

A Random Forest Algorithm [218][219] is a supervised machine learning algorithm that is extremely popular and is used for Classification and Regression problems in Machine Learning. We know that a forest comprises numerous trees, and the more trees more it will be robust. Similarly, the greater the number of trees in a Random Forest Algorithm [220], the higher its accuracy and problem-solving ability. Random Forest is a classifier that contains several decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset. It is based on the concept of ensemble learning which is a process of combining multiple classifiers to solve a complex problem and improve the performance of the model.

2.6.3.1. Random Forest Algorithm working

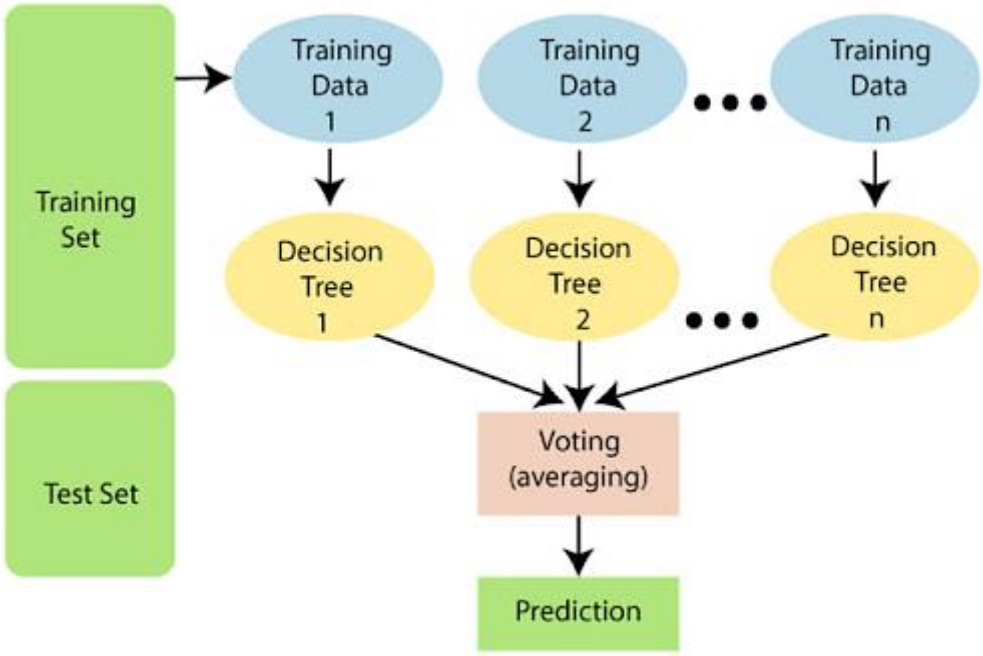


Figure 26. Random forest algorithm.

The following steps explain the working Random Forest Algorithm (Figure 26):

Step 1: Select random samples from a given data or training set.

Step 2: This algorithm will construct a decision tree for every training data.

Step 3: Voting will take place by averaging the decision tree.

Step 4: Finally, select the most voted prediction result as the final prediction result.

2.6.3.2. Difference between decision tree and random forest

Decision Trees	Random Forest
They usually suffer from the problem of overfitting if it's allowed to grow without any control.	Since they are created from subsets of data and the final output is based on average or majority ranking, the problem of overfitting doesn't happen here.
A single decision tree is comparatively faster in computation.	It is slower.
They use a particular set of rules when a data set with features are taken as input.	Random Forest randomly selects observations, builds a decision tree and then the result is obtained based on majority voting. No formulas are required here.

Table 2. Difference between decision tree and random forest algorithm.

2.6.3.3. Important hyperparameters

Hyperparameters are used in random forests to either enhance the performance and predictive power of models or to make the model faster. The following hyperparameters are used to enhance the predictive power:

- n_estimators: Number of trees built by the algorithm before averaging the products.
- max_features: Maximum number of features random forest uses before considering splitting a node.
- mini_sample_leaf: Determines the minimum number of leaves required to split an internal node.

The following hyperparameters are used to increase the speed of the model:

- n_jobs: Conveys to the engine how many processors are allowed to use. If the value is 1, it can use only one processor, but if the value is -1,, there is no limit.
- random_state: Controls randomness of the sample. The model will always produce the same results if it has a definite value of random state and if it has been given the same hyperparameters and the same training data.
- oob_score: OOB (Out Of the Bag) is a random forest cross-validation method. In this, one-third of the sample is not used to train the data but to evaluate its performance.

2.6.4. Naive bayes algorithm

The Naive Bayes classifier is a popular supervised machine learning algorithm used for classification tasks [221] [222]. It is a classification technique based on Bayes' Theorem with an independence assumption among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. It belongs to the family of generative learning algorithms, which means that it models the distribution of inputs for a given class or category. This approach is based on the assumption that the features of the input data are conditionally independent given the class, allowing the algorithm to make predictions quickly and accurately.

In statistics, naive Bayes are simple probabilistic classifiers that apply Bayes' theorem. This theorem is based on the probability of a hypothesis, given the data and some prior knowledge. The naive Bayes classifier assumes that all features in the input data are independent of each other, which is often not true in real-world scenarios. However, despite this simplifying assumption, the naive Bayes classifier is widely used because of its efficiency and good performance in many real-world applications.

Moreover, it is worth noting that naive Bayes classifiers are among the simplest Bayesian network models, yet they can achieve high accuracy levels when coupled with kernel density estimation. This technique involves using a kernel function to estimate the probability density function of the input data, allowing the classifier to improve its performance in complex scenarios where the data distribution is not well-defined. As a result, the naive Bayes classifier is a powerful tool in machine learning

3. Machine learning in image processing

Machine learning (ML) has revolutionized image processing by enabling computers to automatically analyze, interpret, and manipulate images with high accuracy and efficiency [223]. Traditional image processing techniques rely on manually designed filters and algorithms for tasks such as edge detection, noise reduction, and segmentation. In contrast, ML models learn patterns and features from large datasets, allowing them to perform complex image-related tasks with minimal human intervention.

One of the most powerful ML techniques in image processing is deep learning, particularly Convolutional Neural Networks (CNNs). CNNs are designed to recognize spatial hierarchies of features, making them highly effective for tasks like image classification, object detection, and segmentation. For example, CNNs are widely used in facial recognition, medical imaging (e.g., MRI and X-ray analysis), and autonomous driving, where they help identify objects in real-time. Other machine learning algorithms, such as Support Vector Machines (SVMs) and K-Means clustering, are used in applications like handwritten character recognition and image segmentation. Feature extraction techniques, such as Histogram of Oriented Gradients (HOG) and Scale-Invariant Feature Transform (SIFT), are often combined with machine learning models to enhance their ability to recognize objects and patterns in images. Another major application of ML in image processing is image segmentation, where an image is divided into meaningful regions. This is crucial in medical imaging, where tumor detection and tissue segmentation are required for diagnosis.

3.1. Machine learning for image segmentation

Machine learning has significantly advanced image segmentation, enabling precise and automated partitioning of images into meaningful regions, which is essential in fields such as medical imaging, autonomous driving, and satellite image analysis. Traditional machine learning techniques, such as K-means clustering, Support Vector Machines (SVMs), and Random Forests (RFs), rely on manually extracted features like color, texture, and edge information to classify pixels into different regions. For instance, K-means clustering has been widely used in remote sensing, where it groups satellite images into land cover types such as forests, water bodies, and urban areas based on spectral properties. Similarly, SVM-based segmentation has been applied in medical imaging, where it helps identify tumors in MRI and CT scans by classifying pixels based on texture and intensity features. Random Forest classifiers, which utilize an ensemble of decision trees, have also been used for histopathological image segmentation, distinguishing between different tissue structures for cancer diagnosis. However, these traditional approaches often struggle with high-dimensional, complex images because they depend on handcrafted feature extraction, which may not generalize well across diverse datasets. To overcome these limitations, deep learning has revolutionized image segmentation by enabling end-to-end learning without manual feature selection. CNNs and their advanced variants, such as Fully Convolutional Networks (FCNs) [22], are now widely used for pixel-wise classification, significantly improving segmentation accuracy. One of the most impactful architectures, U-Net [19], has shown exceptional

performance in medical image segmentation, such as detecting lung infections in COVID-19 CT scans or identifying brain tumors in MRI scans. Its encoder-decoder structure enables it to capture both low-level and high-level spatial information, making it highly effective for segmenting small and complex structures. More advanced models, such as Mask R-CNN, perform instance segmentation, where they not only classify pixels but also separate different objects of the same category, making them valuable in autonomous driving for detecting individual pedestrians and vehicles. DeepLabV3, another widely used segmentation model, employs atrous convolutions to capture multi-scale contextual information, which enhances its ability to segment complex scenes like urban landscapes and road environments. More recently, transformer-based models such as SETR (Segmenter Transformer) have been introduced, leveraging self-attention mechanisms to capture global dependencies in images, improving segmentation performance in applications like satellite imagery analysis and industrial defect detection. The integration of traditional machine learning approaches with deep learning-based techniques continues to improve segmentation precision, making it an essential tool in various domains, including biomedical imaging, agricultural monitoring, security surveillance, and robotics.

3.2. Machine learning for image classification

Machine learning techniques such as SVMs and RFs have been widely used for image classification due to their strong generalization capabilities and robustness [220]. SVM, a powerful supervised learning algorithm, is particularly effective for binary and multi-class classification tasks. It works by finding the optimal hyperplane that maximizes the margin between different classes in a high-dimensional feature space. For instance, SVMs have been successfully applied in medical imaging [8], such as classifying malignant and benign tumors based on texture and intensity features extracted from MRI scans. On the other hand, RF, an ensemble learning method, constructs multiple decision trees and aggregates their predictions to enhance classification accuracy and reduce overfitting. RF is particularly useful when handling high-dimensional datasets with irrelevant features. A common application is in remote sensing, where RF is used to classify land cover types from satellite imagery by analyzing spectral and spatial features [224]. While these traditional machine learning models require manual feature extraction, they remain highly effective in cases where deep learning may be computationally expensive or when labeled training data is limited. Their interpretability and ability to handle small datasets make them valuable tools for various classification tasks, including medical diagnostics, object recognition, and agricultural monitoring. In contrast, deep learning models like Convolutional Neural Networks (CNNs) have revolutionized image classification by automatically learning hierarchical features from raw images[225]. CNN architectures such as AlexNet, VGGNet, and ResNet have outperformed traditional models in benchmark datasets like ImageNet by recognizing intricate patterns and spatial hierarchies. For instance, CNN-based models have achieved state-of-the-art accuracy in medical diagnostics, such as detecting pneumonia from chest X-rays or classifying diabetic retinopathy from retinal images. More recent advancements, such as Vision Transformers (ViTs) and self-supervised learning, further enhance classification performance by capturing long-range dependencies and reducing reliance on large labeled

datasets. The shift from traditional machine learning models to deep learning approaches has significantly increased the accuracy, scalability, and automation of image classification across various fields, including healthcare, security, and autonomous systems.

3.3. Machine learning for feature selection

Feature selection in image processing using machine learning is a crucial step that enhances model performance by eliminating irrelevant or redundant features while preserving the most informative ones [5]-[10][38]. Unlike feature extraction, which transforms raw data into new feature representations, feature selection focuses on choosing the most significant features from the original dataset, reducing dimensionality and computational costs. Traditional filter-based methods, such as mutual information, chi-square tests, and correlation analysis, rank features based on their statistical importance before model training. These methods are commonly used in texture analysis and remote sensing, where spectral bands or pixel intensities are selected based on their correlation with classification labels. Wrapper methods, like Recursive Feature Elimination (RFE), iteratively remove less important features while training a classifier, ensuring that only the most relevant attributes are retained. A well-known example is gene expression analysis in medical imaging, where RFE is applied to select the most important biomarkers from high-dimensional MRI or CT scan data.

Embedded methods, such as LASSO (Least Absolute Shrinkage and Selection Operator) regression and tree-based algorithms like Random Forest (RF) and Gradient Boosting Machines (GBMs), perform feature selection during training, making them computationally efficient. In medical image segmentation, RF-based feature selection is used to identify the most relevant pixel intensity patterns for tumor detection. Deep learning models integrate feature selection through attention mechanisms, which allow neural networks to focus on the most informative regions of an image. For example, in retinal disease detection, attention-based CNNs highlight critical areas in fundus images, improving diagnostic accuracy. Additionally, auto-encoders, a type of neural network, perform unsupervised feature selection by learning compact, high-level representations of images, reducing noise and redundancy. In autonomous driving, feature selection techniques help identify key visual cues, such as lane markings and obstacles, while filtering out irrelevant background details. The integration of traditional and deep learning-based feature selection methods has led to advancements in biomedical imaging, satellite image classification, and industrial quality control, ensuring more accurate and efficient decision-making in complex image processing tasks.

4. Machine learning and metaheuristic for image processing

The integration of metaheuristic algorithms with machine learning has significantly advanced image processing by enhancing segmentation, classification, feature selection, and feature extraction. Metaheuristic algorithms, inspired by natural and biological processes, provide efficient global optimization techniques that improve the performance of machine learning models in handling complex, high-dimensional image data. These methods help

optimize hyperparameters, refine cluster selection, and improve accuracy in various computer vision tasks.

4.1. Machine learning and metaheuristics for image segmentation

Image segmentation is essential in medical imaging, remote sensing, and object detection, where precise boundary detection is required. Traditional clustering-based segmentation methods, such as FCM, can be improved by integrating metaheuristic optimization techniques. GWO, WOA, and FA have been widely applied to optimize segmentation by selecting the best cluster centroids. For instance, GWO-FCM hybrid models have been employed for MRI brain tumor segmentation, ensuring more precise region identification compared to traditional FCM. In deep learning, metaheuristic techniques like PSO and DE have been used to optimize Fully Convolutional Networks (FCNs) and U-Net architectures, leading to enhanced segmentation performance in medical image processing and autonomous driving applications(See chapter 2, section 4.3).

4.2. Machine learning and metaheuristics for image classification

Image classification is a fundamental task in image processing, where machine learning models categorize images into predefined classes. Traditional machine learning techniques, such as SVMs and RFs, rely on well-selected features, while deep learning models like CNNs automatically extract hierarchical features from images [225]. Metaheuristic algorithms play a crucial role in optimizing these models. For example, HHO and GA have been used to fine-tune hyperparameters in CNN architectures, improving classification accuracy in medical imaging and satellite image recognition. Additionally, ACO has been successfully applied to optimize the selection of relevant training samples, reducing computational complexity while maintaining high accuracy.

4.3. Machine learning and metaheuristics for feature selection

Feature selection is critical in reducing data dimensionality while retaining the most informative features for classification and segmentation tasks. Traditional feature selection methods, such as filter-based and wrapper-based approaches, often struggle with high-dimensional data. Metaheuristic algorithms, including GWO, PSO, and Artificial Bee Colony (ABC), have been extensively applied to improve feature selection in medical imaging and several field image classification [5][6][7][8]. For example, GWO-based feature selection has been utilized in breast cancer diagnosis, selecting the most relevant texture and intensity features from mammograms. Similarly, PSO-based feature selection has been applied to histopathological image classification, improving diagnostic accuracy by reducing redundant information.

4.4. Machine learning and metaheuristics for feature extraction

Feature extraction transforms raw image data into a set of meaningful representations, improving the performance of machine learning models. While deep learning architectures like CNNs can automatically extract features, metaheuristic algorithms can optimize this process by selecting the most relevant deep features. For instance, Hybrid CNN-GA models have been applied to remote sensing image analysis, where GA refines extracted CNN features to improve land cover classification. Additionally, ACO-based feature extraction has been employed in industrial defect detection, ensuring that only the most relevant image features are used for defect identification, reducing false positives and increasing reliability.

5. Conclusion

A wide range of applications in recent literature demonstrate the effectiveness of combining machine learning techniques with metaheuristic algorithms for optimization tasks. This hybrid approach is particularly valuable in scenarios where traditional machine learning methods struggle with high-dimensional data, local optima, or suboptimal parameter configurations. Metaheuristics Optimization algorithms have been extensively used to enhance various aspects of machine learning models, most notably for feature selection, hyperparameter tuning, and model structure optimization. In the field of medical diagnosis, this synergy has proven especially beneficial, enabling the development of more accurate and reliable predictive models. Among these applications, breast cancer classification has received significant attention due to its critical importance in early detection and treatment planning. In the next chapter, we focus on the integration of metaheuristic optimization with machine learning techniques to improve the classification performance in breast cancer diagnosis, highlighting relevant methods, experimental results, and their impact on clinical outcomes.

Chapter 4

Grey wolf optimizer for breast cancer classification

1. Introduction

The Grey Wolf Optimizer (GWO), inspired by the social hierarchy and hunting behavior of grey wolves, has proven to be an effective metaheuristic algorithm for solving complex optimization problems, In the context of breast cancer classification, GWO is particularly useful for feature selection, a critical step in reducing dimensionality and improving classifier performance. Medical datasets, such as those containing breast cancer features, often include redundant or irrelevant attributes that can negatively impact the accuracy and efficiency of classification models. By simulating the intelligent hunting strategy of grey wolves, GWO can explore the feature space effectively and identify the most relevant subset of features that contribute to accurate diagnosis. This not only reduces computational cost but also enhances the interpretability of the model. When integrated with machine learning classifiers such as Support Vector Machines (SVM) or neural networks, GWO-based feature selection has shown promising results in increasing classification accuracy and ensuring reliable detection of breast cancer at early stages.

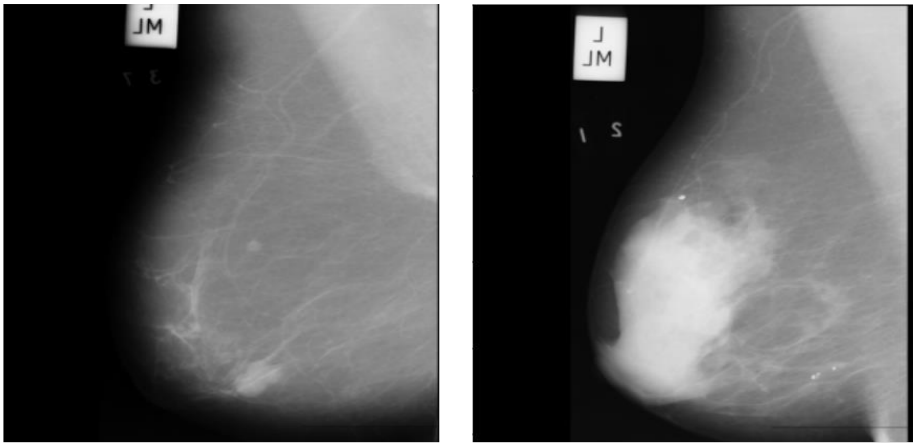
In order to develop an effective approach for precise breast cancer classification, we proposed two methods including a Modified Grey Wolf Optimizer combined with random forest (MGWO-RF) and Correlation technique combined with the Modified grey Wolf Optimizer (CMGWO) for feature selection step then the classification using SVm, RF and NB classifiers. We used the publicly available Wisconsin Breast Cancer Dataset (WBCD), and its features were computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe the characteristics of the cell nuclei present in the image.

2. Breast cancer disease

Breast cancer remains one of the leading causes of mortality among women worldwide [226][227]. Despite being largely preventable in its early stages, a significant number of cases are still diagnosed at advanced stages, reducing the effectiveness of treatment and survival outcomes. The development and implementation of accurate and efficient diagnostic techniques are therefore crucial for enabling personalized care and minimizing the risk of cancer recurrence. In clinical practice, healthcare professionals rely on diverse data sources, including electronic medical records, laboratory test results, and disease-specific studies to support accurate diagnosis and prognosis of breast cancer. Moreover, the integration of artificial intelligence (AI) technologies into the medical domain is increasingly gaining traction, offering promising potential to automate diagnostic processes and enhance the accuracy and efficiency of breast cancer detection.

Breast cancer develops in the cells of the breast tissue, typically originating in the fatty or fibrous connective tissues. It is a malignant tumor that can grow rapidly and progressively worsen, potentially leading to death if not detected and treated in time. While breast cancer predominantly affects women, it can also occur in men, though such cases are rare. Several risk factors contribute to the development of breast cancer, including age, genetic predisposition, and family history. Clinically, breast tumors are generally categorized into two primary types based on their origin and behavior [228]. Figure 27 depicts two sample images from the mammography image analysis society (MIAS) [229] dataset for cancer and normal cases.

- **Benign Tumors**: These are non-cancerous growths composed of cells that do not pose a serious threat to health. Benign tumors do not invade nearby tissues or spread to other parts of the body (a process known as metastasis). They are generally not harmful unless they exert pressure on surrounding tissues, nerves, or blood vessels, potentially causing discomfort or damage.
- **Malignant Tumors**: These tumors consist of cancerous cells capable of invading nearby tissues and spreading to other areas of the body through the bloodstream or lymphatic system, a process referred to as metastasis. Malignant tumors are more aggressive and potentially life-threatening if not treated promptly.



(a) (b)

Figure 27. Two sample images from the MIAS dataset for (a) cancerous, and (b) normal case [229].

Breast cancer can be classified into several subtypes based on the origin, cellular characteristics, and molecular markers:

- **Ductal Carcinoma In Situ (DCIS):** DCIS is a non-invasive or pre-invasive breast cancer where abnormal cells are confined within the milk ducts and have not invaded surrounding breast tissue. It is considered the earliest form of breast cancer (Stage 0) and is typically detected through mammography. While DCIS itself is not life-

threatening, it can increase the risk of developing invasive breast cancer if left untreated.

- **Invasive Ductal Carcinoma (IDC):** IDC is the most common type of breast cancer. It originates in the milk ducts and invades the surrounding breast tissue, with the potential to metastasize to other parts of the body. IDC may present as a lump in the breast and is typically diagnosed through imaging and biopsy.
- **Invasive Lobular Carcinoma (ILC):** Originates in the lobules and has a tendency to be more difficult to detect on mammograms [230]. ILC begins in the lobules, the glands responsible for milk production, and invades surrounding tissues. ILC can be more challenging to detect through imaging due to its growth pattern, which often results in a thickening or swelling rather than a distinct lump.
- **Inflammatory Breast Cancer (IBC):** A rare but aggressive form that blocks lymph vessels in the skin of the breast, causing swelling and redness [231]. IBC is a rare and aggressive form of breast cancer that blocks lymph vessels in the skin of the breast, leading to redness, swelling, and warmth. It often lacks a distinct lump and can be mistaken for an infection. IBC is typically diagnosed at a more advanced stage and requires a combination of chemotherapy, surgery, and radiation therapy.

However, medical imaging plays a critical role in breast cancer screening, diagnosis, and treatment planning [223]:

- **Mammography:** The gold standard for breast cancer screening; it uses low-dose X-rays to identify abnormal masses or calcifications (see figure 28) [232].

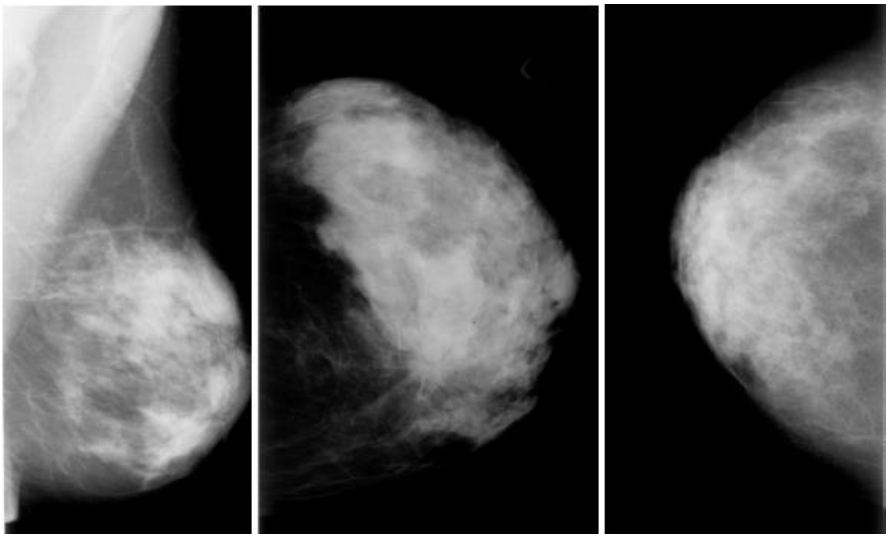


Figure 28. Mammography images from the digital database for screening mammography dataset from kaggle [228].

- **Ultrasound:** Useful for characterizing palpable lumps and distinguishing between solid and cystic masses; often used as a supplemental tool to mammography [233].

- **Magnetic Resonance Imaging (MRI):** Offers high sensitivity, especially in high-risk patients; it is useful for detecting multifocal and multicentric disease [234].
- **Digital Breast Tomosynthesis (DBT):** A 3D imaging technique that improves cancer detection rates and reduces false positives [235].
- **Positron Emission Tomography (PET)/CT:** Generally used for staging and detecting metastasis in advanced cases [236].

2.1. Publicly available datasets for breast cancer research

In the field of breast cancer detection and diagnosis, researchers extensively utilize several publicly available datasets to develop and evaluate machine learning and deep learning models. Among the most commonly used datasets is the Wisconsin Diagnostic Breast Cancer (WDBC) dataset, which contains features extracted from fine needle aspirate (FNA) images and supports binary classification of tumors into benign and malignant categories. Another widely used resource is the Digital Database for Screening Mammography (DDSM), which provides a large collection of mammographic images annotated by radiologists, including cases with verified pathology reports. In addition to these, datasets such as INbreast**,** CBIS-DDSM, and Breast Ultrasound Images Dataset offer rich image-based data for algorithm development across various imaging modalities like mammography, ultrasound, and MRI. The availability of these public datasets has been instrumental in accelerating progress in automated breast cancer detection, enabling reproducible research and benchmarking of algorithmic performance.

Breast cancer remains a major global health concern. The combination of early detection, advanced imaging techniques, and the use of publicly available datasets can significantly enhance the accuracy of diagnosis and treatment planning. These datasets also support the development of Computer Aided Diagnostic (CAD) based solutions to assist clinicians and researchers in fighting breast cancer more effectively.

2.2. Wisconsin diagnosis breast cancer dataset (WDBC)

The dataset used in this study is sourced from the UCI Machine Learning Repository and is known as the Wisconsin Diagnostic Breast Cancer (WDBC) dataset [237]. It comprises 569 instances, each representing a breast tumor diagnosis classified as either benign or malignant. Among these, 357 cases (62.74%) are labeled benign, while 212 cases (37.26%) are malignant. The class distribution is visually represented in Figure 29. The dataset includes 33 attributes in total. These consist of an ID number. The last feature, unnamed, which had the value null for all occurrences, and, a diagnosis label (where *M* denotes malignant and *B* denotes benign), and 30 real-valued features extracted from digitized images of fine needle aspirate (FNA) biopsies of breast masses. These features describe various morphological characteristics of the cell nuclei observed in the biopsy images.

Specifically, the 30 numeric attributes represent ten cell features, including radius, texture, perimeter, area, smoothness, compactness, concavity, concave points, symmetry, and fractal dimension. For each of these ten features, the dataset provides three computed values: the mean, standard error, and worst (for the largest value), leading to 30 numerical descriptors per

instance. The first column contains a unique identifier for each patient, while the second column holds the class label indicating the diagnosis. Columns 3 to 32 consist of the real-valued features, which are used to train and evaluate classification models that predict whether a tumor is benign (non-cancerous) or malignant (cancerous). A complete description of these features is provided in Table 3.

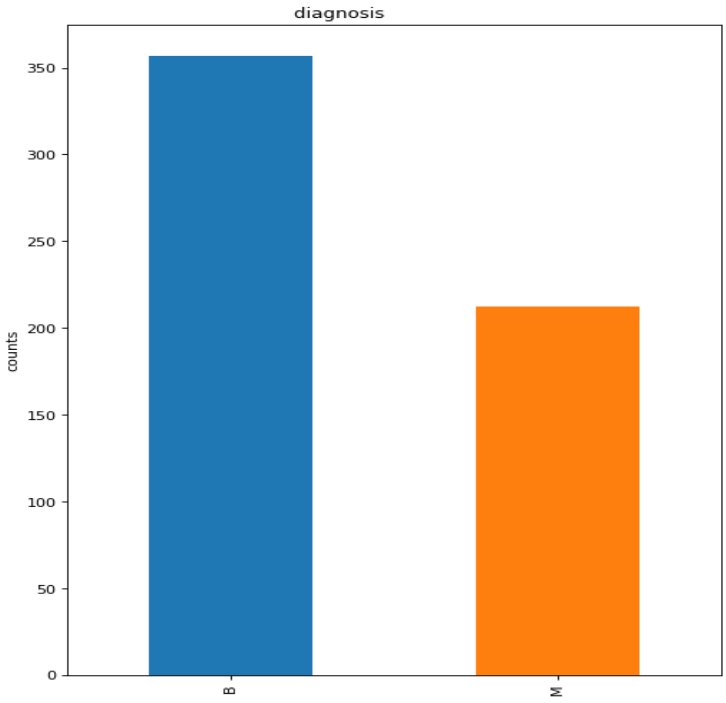


Figure 29. The distribution of the number of Benign and Malignant classes in the WDBC dataset.

Table 3. Wisconsin diagnosis breast cancer features description.

Number	Feature Group	Description
1	Radius	Distance from center to perimeter.
2	Texture	Standard deviation of gray-scale values.
3	Perimeter	Length of the cell boundary.
4	Area	Size of the cell nucleus.
5	Smoothness	Local variation in radius lengths.
6	Compactness	(Perimeter² / Area - 1.0).
7	Concavity	Severity of concave portions of the contour.
8	Concave Points	Number of concave portions of the contour.
9	Symmetry	Symmetry of the nucleus shape.
10	Fractal Dimension	Roughness of the contour (coastline approximation).

2.3. WDBC dataset pre-processing

Purification and modification of the dataset are required before applying ML algorithms to the dataset, it is a necessary step to pre-process the data. Performance and accuracy of the

predictive model are not only affected by the algorithms used, but also by the quality of the dataset and pre-processing. The phases of pre-processing used in this investigation are as follows:

- **Missing values checking:** The dataset contains 569 instances of 33 variables. However, it was discovered that the variable id had no effect on the dataset description or on disease prediction because it merely keeps a serial record of the instances. As a result, the dataset's id feature was removed. Additionally, while conducting additional preprocessing operations on the dataset, it was discovered that the last feature, unnamed: 32, had the value null for all occurrences. This might be a mistake in the data collection process, because of this the feature was also removed from the dataset.

- **Encoding data:** The performance of machine models depends on various aspects. One element that influences performance of the models are the methods used to analyze data and feed it to the model. As such, vital step in encoding data is turning data into categorical variables understood by ML models. Encoding data, elevates model quality and helps in feature engineering. The class label "diagnosis" was expressed as strings of (B= Benign, M= Malignant). This category characteristic must be converted to restricted numbers. This is done to transform data into a format that ML algorithms can understand. Label encoding was used to encode the diagnostic occurrences in this study, and the result was (M=1, B =0).

- **Outliers checking:** An outlier is a statistic or observation that deviates from a distribution's overall pattern. If few data are significantly different or not in range of main trend then those are termed outliers. There skewness results, affecting the mean and standard deviation of the distribution. In this work detects the existence of outliers in the dataset. As a result, outliers were identified and eliminated from their respective features.

- **Normalization :** Feature normalization or standardization is also performed to scale the values and eliminate bias caused by differing feature ranges. This step ensures that all features contribute equally to the learning process and helps algorithms converge more efficiently during training.

As a result, The Wisconsin Diagnostic Breast Cancer (WDBC) dataset is a widely used benchmark in medical machine learning tasks, particularly for classifying tumors as benign or malignant. Before training any model, data preprocessing is essential to ensure quality input. The dataset undergoes several preprocessing steps including handling missing values, normalization or standardization of the features to a common scale, and encoding the target variable into binary values (e.g., benign as 0 and malignant as 1). This helps to enhance model performance and prevent bias toward certain features due to scale differences.

3. Grey Wolf Optimizer Algorithm

The grey wolf (Canis lupus), a member of the canid family, is recognized as an apex predator due to its position at the top of the food chain. These wolves usually live in packs ranging from five to twelve members and exhibit strong social behavior, particularly in hunting and leadership dynamics. The structured social hierarchy within the pack, illustrated in Figure 30, inspired the development of the Grey Wolf Optimizer (GWO) algorithm [11].

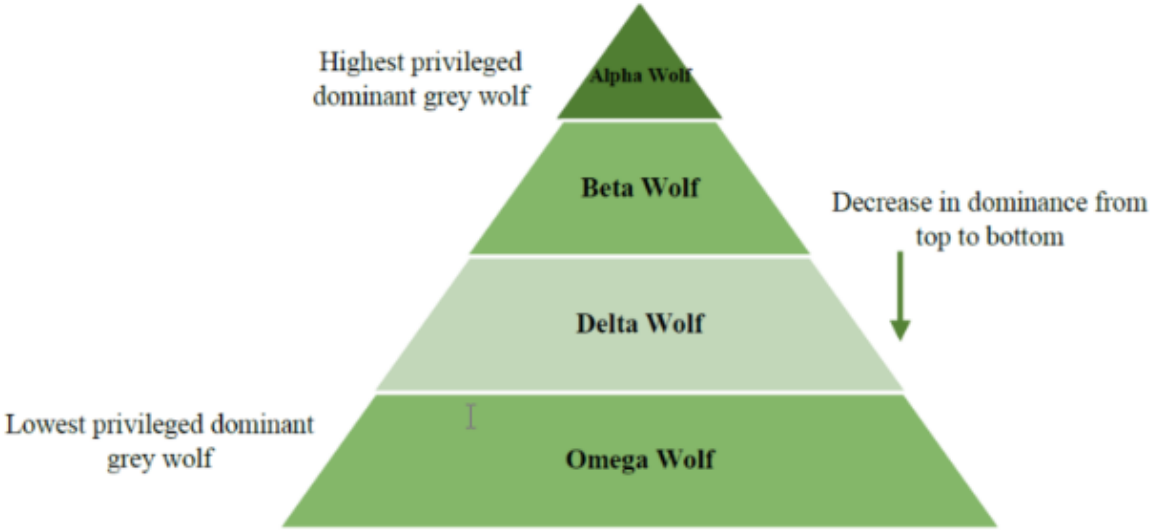


Figure 30. The hierarchy of Grey Wolf Optimizer.

The leader of the wolf pack, whether male or female, is referred to as the alpha. This individual is primarily responsible for making key decisions, such as when to hunt, where to rest, and when the pack should move. Interestingly, the alpha is not necessarily the strongest member physically, highlighting that leadership within the pack relies more on organization and discipline than on sheer strength.

In the grey wolf hierarchy, the beta occupies the second-highest rank, serving as the alpha's subordinate and trusted aide. Beta wolves assist the alpha in decision-making, managing pack responsibilities, and maintaining order among lower-ranking members. They also act as advisors to the alpha and reinforce its commands throughout the pack.

Below the betas are the deltas, who are subordinate to both alphas and betas but hold authority over the lowest-ranking wolves. This group includes roles such as scouts, sentinels, senior members, hunters, and caregivers.

At the bottom of the hierarchy is the omega, the least dominant member of the pack. Omegas must yield to all others, typically eat last, and often serve as the scapegoat or tension reliever within the group. Occasionally, they also take on the role of babysitters for the pack.

3.1. Mathematical model

The social hunting behavior of grey wolves is mathematically represented by utilizing the positions of the fittest wolves [11], namely the alpha, beta, and delta wolves, to guide the

search for the optimal solution. Meanwhile, omega wolves follow the dominant wolves during the hunting process.

3.1.1. Encircling the prey

For hunting, grey wolves chase and encircle the prey. Mathematically, it is modeled as given in Eq. (1) and (2):

$$\vec{X}(t+1) = \vec{X}_P(t) + \vec{A}.\vec{D} \quad (1)$$

$$\vec{D} = \left|\vec{C} \,.\, \vec{X}_P(t) - \vec{X}(t)\right| \quad (2)$$

here t is the indicated current iteration and t + 1 represents next iteration. $\vec{X}$ is the position vector of grey wolf and $\vec{X}_P$ is the position vector of prey. $\vec{A}$ and $\vec{C}$ are coefficient vectors where they are depicted as:

$$\vec{A} = 2\vec{a} \,.\, \vec{r_1} - \vec{a} \quad (3)$$

$$\vec{C} = 2 \,.\, \vec{r_2} \quad (4)$$

where $\vec{r_1}$ and $\vec{r_2}$ are the random vectors in the range of [0, 1] and components of $\vec{a}$ are linearly decreased from 2 to 0 over the courses of iterations [11]

3.1.2. Hunting

Once the prey's location is estimated, the grey wolves encircle it to begin the hunt. The hunting process is directed by the alpha, beta, and delta wolves. Although the exact location of the prey is unknown across the entire search space, it is assumed that the alpha, beta, and delta wolves have knowledge of the prey's position. As a result, the three best solutions are retained, and the remaining wolves (omega wolves) update their positions based on these optimal solutions. The hunting process is mathematically governed by Equation (7), which is derived from Equations (5) and (6).

$$\vec{D_\alpha} = \left|\vec{C_1}.\vec{X}_\alpha - \vec{X}\right|, \qquad \vec{D_\beta} = \left|\vec{C_2}.\vec{X}_\beta - \vec{X}\right| \;, \quad \vec{D_\delta} = \left|\vec{C_3}.\vec{X}_\delta - \vec{X}\right| \quad (5)$$

$$\vec{X_1} = \vec{X_\alpha} - A_1 \,.\, \left(\vec{D_\alpha}\right), \vec{X_2} = \vec{X_\beta} - A_2 \,.\, \left(\vec{D_\beta}\right), \vec{X_3} = \vec{X_\delta} - A_3 \,.\, \left(\vec{D_\delta}\right) \quad (6)$$

$$\vec{X}(t+1) = \frac{\vec{X_1} + \vec{X_2} + \vec{X_3}}{3} \quad (7)$$

Figure 31 demonstrates how a search agent updates its position based on the locations of the alpha, beta, and delta wolves within a 2D search space. As shown, the final position is randomly influenced by the positions of the alpha, beta, and delta agents. Therefore, these wolves collectively determine the prey's location, while the other wolves adjust their positions randomly around it.

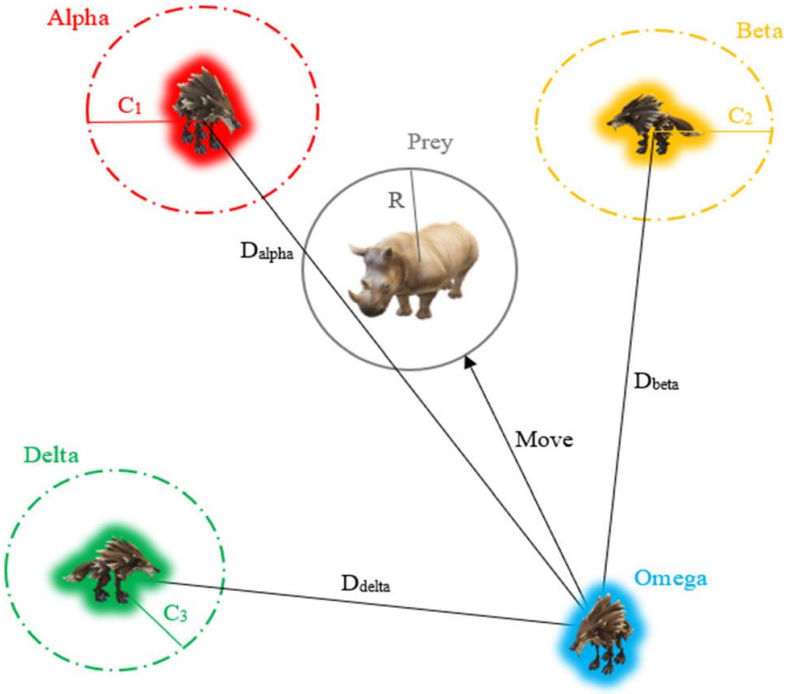


Figure 31. Position updating in the Grey Wolf Optimizer [11].

During the search and hunting process, exploration and exploitation are managed through the parameters $|\overrightarrow{A}|$ and $|\overrightarrow{C}|$. The parameter $|\overrightarrow{A}|$ gradually decreases from 2 to 0, ensuring a balance between exploration and exploitation. When $|\overrightarrow{A}|>1$, the grey wolves spread out to explore the search space for the prey, while when $|\overrightarrow{A}|<1$, they converge towards each other to attack the prey (See figure 32). The inherent randomness in this process helps prevent the wolves from becoming trapped in local minima, promoting a more effective global search.

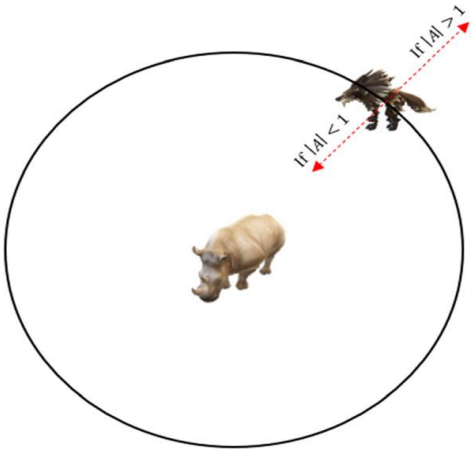


Figure 32.Attacking prey and searching prey [11].

Algorithm 1 : Pseudocode of GWO

Algorithm: Grey Wolf Optimizer (GWO)
Input: MaxIterations (T), Population Size (N), Search Space.
1. Initialize the population of grey wolves (X_i, i = 1, 2 ... N) randomly
2. Evaluate the fitness of each wolf
3. Identify the three best wolves:
- Alpha (X_α) -> Best solution
- Beta (X_β) -> Second best solution
- Delta (X_δ) -> Third best solution
4. For each iteration t = 1 to T:
For each wolf (X_i):
Update the control parameters a (linearly decreases from 2 to 0).
Compute coefficient vectors A and C using Equation 3 and 4.
Compute new positions relative to α, β, and δ Using Equation 5-7.
Evaluate new fitness value.
Update X_α, X_β, and X_δ if better solutions are found.
5. Return the best solution Xα

3.2. Literature review about Modified GWO

The Grey Wolf Optimizer (GWO) has emerged as a powerful and widely adopted metaheuristic algorithm inspired by the leadership hierarchy and hunting strategies of grey wolves in nature. Since its introduction, GWO has demonstrated impressive performance across various optimization tasks [238]. However, like many metaheuristics, it also faces challenges such as premature convergence, lack of population diversity, and difficulty in balancing exploration and exploitation. To address these limitations, numerous researchers have proposed enhanced and modified versions of GWO. These variations aim to improve the algorithm's adaptability, convergence speed, and overall performance by integrating concepts from reinforcement learning, random walks, evolutionary strategies, and swarm intelligence, among others. This literature review explores a range of such modified GWO approaches.

A study [239] introduced the Variable Weights-GWO, an enhanced version of the Grey Wolf Optimizer that incorporates variable weights to better preserve the social hierarchy within the wolf pack. In this approach, the weight assigned to the alpha position must always be equal to or greater than those of the beta and delta positions, with the beta's weight also required to be at least equal to that of the delta. Furthermore, a new formula was proposed to adjust the control parameter, aiming to reduce the chances of the algorithm getting stuck in local optima. The performance of VW-GWO was evaluated against ALO, PSO, BA, and the original GWO across 11 benchmark functions, and the results validated the effectiveness of the proposed method.

Another researcher [240] proposed an enhanced version of GWO called Improved Alpha-Guided GWO (IAgGWO), which incorporates a novel guidance mechanism along with a mutation operation to speed up convergence and prevent the algorithm from getting trapped in local optima. The use of scalar coefficients A and C simplifies the implementation of the

algorithm. The study demonstrated the superiority of IAgGWO by comparing it with four other algorithms across 35 benchmark functions and through its application to the engineering design problem of a two-stage operational amplifier.

A more precise model [241] was developed to mimic the hierarchy of authority and group hunting tactics used by grey wolves in the wild. According to this new model, each wolf moves straight in the direction of the prey's predicted location and the location of each wolf is dynamically evaluated by the leader wolves . Evaluations using the CEC2017 benchmark suite showed that the improved optimizer significantly outperforms the original GWO and its later variants in terms of both convergence speed and solution stability.

To enhance the wolf pack's ability to locate prey, a study [242] proposed a modified version of GWO known as Random-Walk-GWO, which incorporates random walks for the leading wolves. Experimental results based on the CEC 2014 benchmark revealed that RWGWO outperformed both the standard GWO and other metaheuristic algorithms.

Another proposed algorithm, RBGWO [243], aims to enhance the overall efficiency of the search process by effectively balancing exploration and exploitation. It introduces three consecutive improvement strategies, including a random walk guided by Student's t-distributed random values and a social hierarchy mechanism. The first strategy updates each grey wolf's position using weight-based variables. The second incorporates a random walk approach inspired by [242] to refine position updates. The third introduces a novel randomization technique to further boost the search efficiency and reinforce the random walk process. When tested on the CEC 2014 benchmark functions at various scales, RBGWO outperformed the standard GWO.

The Experienced GWO (EGWO) [244] integrates reinforcement learning techniques to determine the optimal actions to take during different phases of the optimization process and across various regions of the search space. A neural network is used to store and utilize this experiential knowledge. The proposed EGWO was evaluated against the original GWO, PSO, and GA in two key optimization tasks: feature selection and neural network weight adaptation. The results demonstrated that EGWO delivered significantly improved performance over the compared algorithms.

The Improved Grey Wolf Optimizer (I-GWO) [245] was designed to address global optimization and engineering design challenges. To tackle issues such as limited population diversity, imbalance between exploration and exploitation, and premature convergence in the original GWO, the I-GWO incorporates a novel mobility strategy known as the Dimension Learning-based Hunting (DLH) search method. Experimental results on various engineering design problems highlighted the algorithm's effectiveness and versatility.

An Enhanced Grey Wolf Optimizer (EGWO) was proposed in [246], incorporating Lévy flight and binomial crossover mechanisms to improve the grey wolves' hunting behavior. This enhanced strategy was also applied to optimize clustering processes. The EGWO's performance was evaluated using seven benchmark datasets from the UCI repository and compared against five other clustering algorithms. Empirical results demonstrated that EGWO is a robust and promising approach for efficient large-scale data clustering.

3.3. Modified GWO using weighted position update method

In the conventional GWO algorithm, the position of search agents (omega wolves) is updated by averaging the positions of the top three wolves including alpha, beta, and deltas wolves, as defined in Equation (7). While simple, this uniform averaging approach may lead to premature convergence and low-quality solutions, particularly in complex or high-dimensional search spaces. To overcome this limitation, a weighted position update mechanism is adopted, inspired by the work of S. Kumar and M. Singh [8]. This approach assigns varying weights to the contributions of the alpha, beta, and delta wolves based on their fitness, allowing the more optimal leaders to have a greater influence on the position updates. This modification enhances both the convergence speed and solution quality by dynamically adjusting the influence of each leading wolf.

The mathematical formulation for this technique is presented in Equations (8) and (9), which redefine the agents' movement in a more adaptive and fitness-aware manner.

$$W_1=A_1*C_1, \qquad W_2=A_2*C_2, \qquad W_3=A_3*C_3 \tag{8}$$

$$X(t+1) = (W_1*X_1 + W_2*X_2 + W_3*X_3) / (W_1+W_2+W_3) \tag{9}$$

3.4. Grey Wolf Optimizer for image processing

This section presents a review of relevant studies on image processing, with a particular focus on medical imaging applications. Various enhancements to the Grey Wolf Optimizer (GWO) have been proposed to improve segmentation, classification, and feature selection tasks. The following subsections provide an in-depth discussion of these approaches and their effectiveness in different image processing domains.

The rapid expansion of multimedia content, particularly images, on social media platforms has intensified interest in content-based image retrieval (CBIR) systems. Despite the emergence of various CBIR techniques, face recognition continues to present significant challenges. To address this, a study [247] introduced an enhanced version of the Grey Wolf Optimizer, called Varying Weight GWO (VW-GWO), for optimizing a Support Vector Machine (SVM)-based facial recognition model. Simulation results demonstrated that VW-GWO significantly improved classification accuracy and stability.

In another advancement, a Mixed GWO approach [248] was proposed to effectively handle optimization problems involving continuous, discrete, or mixed variables. Leveraging this bio-inspired technique, the study successfully performed simultaneous denoising and unmixing of multispectral images.

Furthermore, an Ensemble Grey Wolf Optimizer (EGWO) [249] was developed by integrating an elite-based search strategy and a modified position update equation. This method showed promising results when tested on 12 images from the USC-SIPI dataset.

In the realm of medical imaging, chest X-ray (CXR) images have become preferred over CT scans for COVID-19 detection, owing to their clearer representation of lung abnormalities. To enhance diagnostic accuracy and reduce reliance on manual interpretation, a

three-stage classification model CXGNet was introduced [250]. It combines an enhanced GWO with genetic algorithm (EGWO-GA) and deep learning-based convolutional neural networks (DLCNN) for optimal feature selection. This model outperformed traditional diagnostic methods such as RT-PCR, antigen, and serological tests in both speed and efficiency.

Segmentation, a critical stage in image processing, also benefits from GWO-based enhancements. Among the popular segmentation methods, histogram-based thresholding stands out for its simplicity and effectiveness. For multi-level thresholding tasks, researchers [251] proposed a Discrete Multi-Objective Shuffled GWO (D-MOSG) algorithm, which delivered superior segmentation performance across various benchmarks. Experimental results confirmed that the Discrete Multi-Objective Shuffled Grey Wolf Optimizer (D-MOSG) outperforms other algorithms in multi-level image thresholding tasks, delivering superior segmentation accuracy.

In a related study [252], a Modified Grey Wolf Optimizer (MGWO) was introduced to enhance the original GWO algorithm. This variant was applied to the segmentation of leaf spot diseases in maize using four distinct threshold levels. The results demonstrated that MGWO delivers competitive performance, highlighting its effectiveness as a robust optimizer for multi-threshold image segmentation applications.

4. Modified Grey Wolf Optimizer and Random Forest strategy

Using the Modified Grey Wolf Optimization (MGWO) algorithm for breast cancer classification enhances diagnostic accuracy by selecting the most relevant features from complex medical datasets. In this section we implement a effective method for breast cancer classification based on feature selection and classification by integrating GWO with Machine learning.

4.1. Modified GWO and random forest strategy

In this section, breast cancer classification was performed using the WDBC dataset, which includes various features extracted from digitized images of breast tissue samples. The process began with comprehensive data preprocessing, including handling missing values, normalizing features for consistency, and encoding target labels for binary classification (benign vs. malignant). Feature selection was then applied using a Modified Grey Wolf Optimization (MGWO) algorithm, which improves the standard GWO by enhancing its search capability to effectively identify the most relevant features while eliminating redundant or irrelevant ones. Classification was conducted using a Random Forest (RF) classifier, chosen for its robustness, ensemble learning approach, and efficiency in handling high-dimensional data. Two experiments were carried out: the first involved training the RF classifier with all original features, while the second used only the optimized feature subset selected by MGWO.

The final step involved evaluating the overall classification strategy using key performance metrics, including accuracy, precision, recall, and F1-score. The results demonstrated that the MGWO-based feature selection not only effectively reduced the dimensionality of the dataset

but also maintained or improved the classification performance. This highlights the advantage of integrating an intelligent feature selection method with a powerful classifier like Random Forest. Figure 33 illustrates the workflow of the two experimental scenarios proposed for breast cancer diagnosis.

In the first scenario, the process consists of three main phases. The first phase is data preprocessing, which is common to both scenarios. During this phase, data cleaning and filtering were carried out to eliminate noise and prevent the generation of ineffective rules or patterns. Specifically, the WDBC dataset was cleaned, and outliers were removed using the outer line (outlier detection) approach. The second phase involves classification using a Random Forest classifier trained on all the original features of the dataset. The third phase is the evaluation of classification performance using appropriate metrics.

The second scenario consists of four main phases. It begins with the same data preprocessing step as the first scenario. Next, a feature selection process is applied using the Modified Grey Wolf Optimization (MGWO) algorithm to identify the most significant features contributing to classification accuracy. In the third phase, a Random Forest classifier is again used, but this time trained only on the selected features. Finally, the fourth phase involves the evaluation of classification performance to compare the effectiveness of the reduced feature set against the full set used in the first scenario.

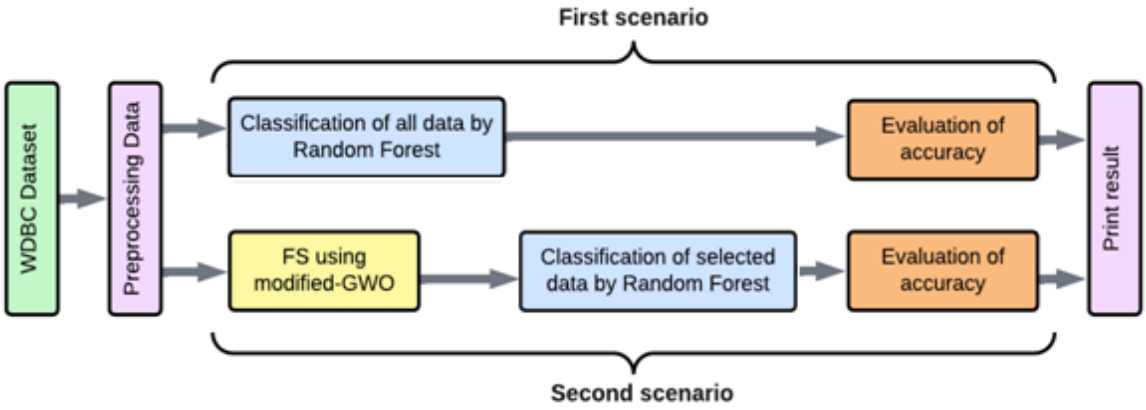


Figure 33. Proposed method for accurate breast cancer classification.

4.2. Experimental results

In this study, the primary objective was to enhance classification performance and improve diagnostic accuracy by reducing the feature dimensionality using the Modified Grey Wolf Optimization (MGWO) algorithm. During the experiments, the MGWO was configured to run for 20 iterations with 10 search agents. For the classification task, a Random Forest (RF) classifier was employed to distinguish between malignant and benign tumors, chosen for its high accuracy, robustness, and ability to handle complex datasets. The most promising results were achieved through a hybrid approach that combined MGWO for feature selection with the RF classifier for final prediction. As shown in Table 4, the proposed method demonstrated improved performance across all evaluation metrics, including sensitivity, specificity,

precision, F1-score, and accuracy, when dimensionality reduction was applied using MGWO prior to classification with Random Forest.

Table 4. Classification results of the proposed MGWO-RF approach using different performance measures.

Performance Measures	Classification results (%)	
	Without feature selection	Feature Selection using MGWO-RF
Sensitivity	96,3	98,1
Specificity	97,8	98,9
Precision	96,3	98,1
F1-score	96,3	98,1
Accuracy	97,2	98,6

4.2.1. Comparing the classification results between the modified GWO-RF and the base GWO-RF

Table 5 presents a performance comparison between the proposed Modified GWO-RF approach and the baseline GWO-RF method. This comparison aims to evaluate the impact of integrating a weighted position update mechanism into the original GWO algorithm. The results clearly demonstrate that the modified version significantly enhances classification performance. Specifically, the Modified GWO-RF approach achieved an accuracy of 98.6%, an F1-score of 98.1%, and a sensitivity of 98.1%, outperforming the baseline across these key metrics. These improvements highlight the effectiveness of the proposed enhancements in optimizing feature selection and boosting diagnostic accuracy.

Table 5. Comparing classification results between the modified GWO-RF approach and the base GWO-RF approache.

Performance Measures	Classification results (%)	
	Modified GWO with RF	Original GWO with RF
Sensitivity	98,1	96,3
Specificity	98,9	98,9
Precision	98,1	98,1
F1-score	98,1	97,2
Accuracy	98,6	97,9

4.2.2. Comparing classification results between the modified GWO-RF and existing feature selection approaches

Table 6 provides a comparative analysis of the classification performance between the proposed MGWO-RF approach and several existing feature selection-based methods for breast cancer detection. The comparison highlights how different techniques perform in terms of key evaluation metrics such as accuracy, sensitivity, and F1-score. From the results, it is

evident that the proposed MGWO-RF method consistently outperforms the other approaches, demonstrating superior effectiveness in selecting the most relevant features and enhancing classification performance. These findings validate the strength of integrating the Modified Grey Wolf Optimization with the Random Forest classifier for accurate and reliable breast cancer diagnosis.

Table 6. Comparing results of the modified GWO-RF approach with existing feature selection approaches.

Approaches	Authors	Years	Number of features	Accuracy %
FS-KNN	Sayed et al.[5]	2019	14	90,28
FS – GBDT	Rao et al. [6]	2019	14	92.80
FS-KNN	Abdel- Basset et al.[7]	2020	16	94,82
FS + EGWO-SVM	S. Kumar & M. Singh[8]	2021	6	98,24
Proposed approach	Proposed	2022	12	98,60

5. Hybrid algorithm using correlation and Modified GWO based feature selection

Feature selection is a crucial step in many machine learning tasks, including classification, where the goal is to identify a subset of relevant features that enhance model performance while reducing computational complexity. In high-dimensional datasets, such as the Wisconsin Diagnostic Breast Cancer (WDBC) dataset, feature selection becomes even more significant due to the risk of overfitting, computational inefficiencies, and the presence of noisy or irrelevant features.

Traditional feature selection methods often rely on either filter, wrapper, or embedded techniques. Filter methods evaluate the relevance of features based on their statistical properties, while wrapper methods assess subsets of features by evaluating model performance. However, these methods have their limitations, particularly when dealing with correlated features that can result in redundancy and hinder classification performance.

To address these challenges, a hybrid approach that combines correlation-based feature selection with the optimization power of the Modified Grey Wolf Optimizer (MGWO) is proposed. This hybrid algorithm aims to first remove highly correlated features, ensuring that only independent and non-redundant features are retained, and then further optimize this subset using MGWO to identify the most relevant features for classification. By combining correlation-based feature selection with the Modified Grey Wolf Optimizer, this hybrid algorithm aims to achieve a balance between reducing the feature space and maintaining or enhancing classification accuracy, providing an efficient and reliable method for breast cancer classification and other similar high-dimensional classification tasks.

The experimental results indicate that the proposed method successfully achieves a balance between feature relevance and diversity, resulting in improved performance across multiple evaluation metrics.

5.1. Pearson Correlation technique

The Pearson Correlation Coefficient is a statistical measure used to evaluate the linear relationship between two continuous variables [253], [254]. It helps determine the degree to which one variable can be predicted based on the behavior of another. In the context of feature selection, this method is commonly used to assess the correlation between input features and the target variable. Ideally, the selected features should exhibit a strong correlation with the target variable, while maintaining minimal correlation with one another to avoid redundancy. When two features are highly correlated with each other, they carry overlapping information, and retaining both may lead to unnecessary complexity in the model. In such cases, only one of the correlated features is typically retained, as the other does not contribute additional predictive value. The correlation coefficient (r) ranges from -1 to +1, a value of +1 indicates a perfect positive correlation, -1 indicates a perfect negative correlation, and 0 signifies no linear correlation. The closer the absolute value of the coefficient is to 1, the stronger the linear relationship between the two variables.

5.2. Feature selection and classification approach using Correlation and Modified Grey Wolf Optimizer (MGWO)

The proposed method aims to enhance the classification accuracy for the Wisconsin Diagnostic Breast Cancer (WDBC) dataset by performing feature selection in two level: first using correlation-based feature selection to remove redundant features, and then applying the Modified Grey Wolf Optimizer (MGWO) to optimize the remaining uncorrelated features. The resulting feature set is subsequently fed into classifiers such as Support Vector Machine (SVM)**,** Random Forest (RF)**,** and Naïve Bayes (NB) for classification (see Figure 34).

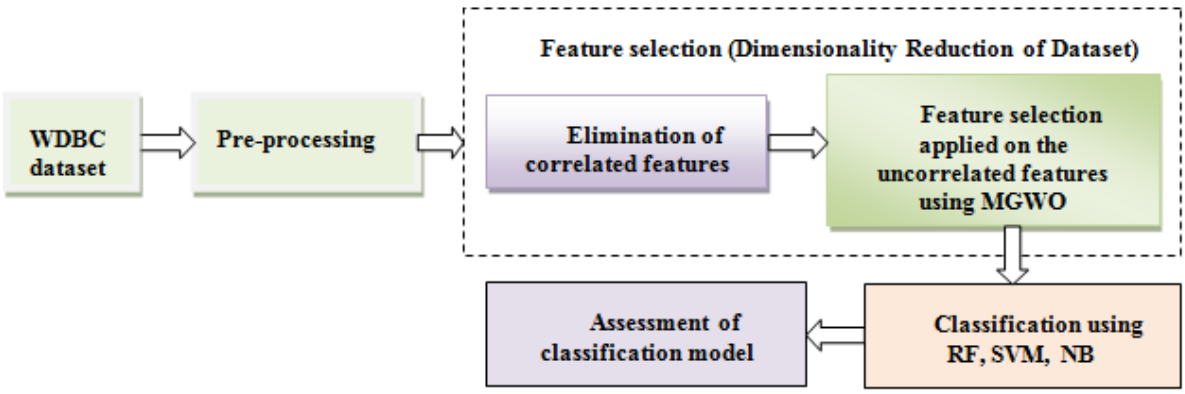


Figure 34. Flowchart of the suggested breast cancer classification method based feature selection.

In our proposed model, the WDBC dataset was utilized, and a preprocessing step was performed to clean the data by removing irrelevant or unused features. The feature selection (FS) process employed a two-step strategy that combined a correlation-based method with the Modified Grey Wolf Optimization (MGWO) algorithm to identify the most relevant attributes. For the classification task, we implemented multiple machine learning algorithms,

including Support Vector Machine (SVM), Random Forest (RF), and Naïve Bayes (NB), to evaluate the model's performance.

The proposed approach for breast cancer classification follows a two-stage process: feature selection and classification. The first stage focuses on dimensionality reduction using a combination of filter and wrapper methods. Initially, correlation-based feature selection is applied to the WDBC dataset to remove highly correlated features both among themselves and with the target variable (cancer tumor or not). This technique helps in reducing redundancy and improving the feature space by selecting only the most independent and relevant features. As a result, the number of features is reduced from 30 to 16, which enhances the efficiency and performance of subsequent steps (refer to Section 5.2.1 for further details).

Once the correlated features are eliminated, the Modified Grey Wolf Optimizer (MGWO) is applied to the remaining 16 non-correlated features. The MGWO algorithm is designed to select the most significant and relevant features, optimizing the feature set further. By applying the MGWO to 16 features instead of the original 30, the algorithm can operate more efficiently, providing better results with fewer variables. This step ensures that the selected features contribute maximally to the classification process and improve the overall accuracy of the model (as described in Section 5.2.2).

In the second stage of the process, the reduced and optimized feature set is used to classify the breast cancer data using three different machine learning classifiers: Support Vector Machine (SVM), Random Forest (RF), and Naïve Bayes (NB). These classifiers are chosen for their robustness in handling high-dimensional datasets and their ability to provide reliable predictions for breast cancer classification. The overall process is represented in Algorithm 2, which outlines the two main stages of the proposed method: the feature selection process, which combines correlation-based selection and MGWO, followed by the classification stage, where machine learning classifiers are used to perform the final classification task.

Algorithm 2. Pseudocode of the proposed method.
Phase 1: Input: upload WDBC Dataset Step1 : Preprocessing and removing unused features from Dataset. Step2 : Feature selection with correlation technique and removing correlated features from original dataset (see Section 5.2.1). Step3 : Feature selection applied on uncorrelated features using MGWO algorithm (see Section 5.2.2). Output : Selected features Phase 2: Classification of breast cancer using selected features based on the output of Phase1 and assessment the accuracy of classification.

5.2.1. Feature selection using correlation

As shown in Figure 35, a heat map was employed to analyze the correlations between the features of the dataset. The analysis revealed a strong correlation between the features "radius-mean", "parametric-mean", and "area-mean", as all these features provide similar information regarding the size of breast cancer cells. Given the redundancy in the information conveyed by these features, only the "area-mean" feature was selected to effectively represent the size of the breast cancer cells, streamlining the feature set while preserving essential information.

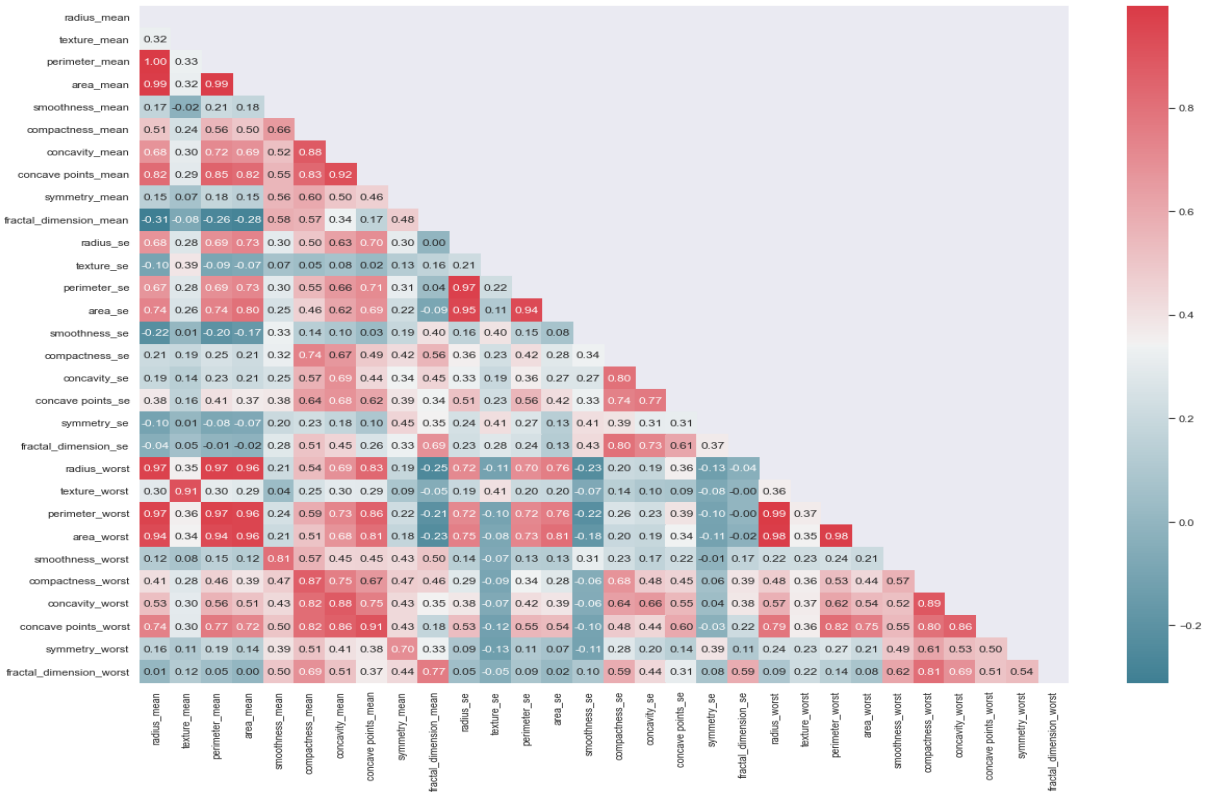


Figure 35. Heat-map plot showing the correlations among all features of WDBC.

A total of 14 features were removed, including 'perimeter-mean', 'radius-mean', 'compactness-mean', 'concave points-mean', 'radius-se', 'perimeter-se', 'radius-worst', 'perimeter-worst', 'compactness-worst', 'concave points-worst', 'compactness-se', 'concave points-se', 'texture-worst', and 'area-worst'. Following this feature elimination process, 16 features remained for further analysis. The relationships between these selected features are depicted in Figure 36.

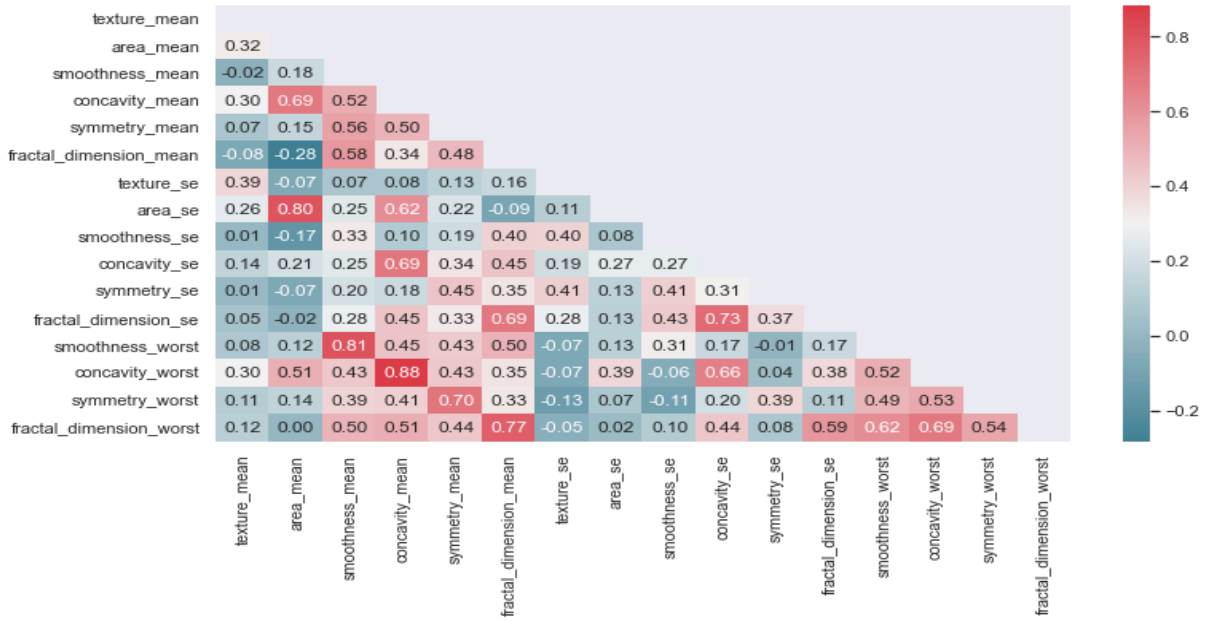


Figure 36. Heat-map plot showing the correlations among selected features of WDBC.

5.2.2. Feature Selection using Modified GWO

To effectively detect breast cancer tumors in our study, we leveraged the strengths of the Modified Grey Wolf Optimizer (MGWO) algorithm to identify the most relevant subset of features. Algorithm 3 presents the pseudo code of the MGWO algorithm. As previously mentioned, in the MGWO approach, Equation (8) is employed instead of Equation (7) to generate more accurate and relevant results. Various classifiers were then trained using the feature subset determined by the MGWO algorithm. An illustrative example of the position vector used by the alpha search agent in the MGWO algorithm for feature selection is depicted in Figure 37. The position vector consists of binary values (1 or 0) for each feature. For an n-dimensional problem, the position vector contains n bits. A feature is excluded from the subset if its corresponding position in the vector is 0, while it is selected if the value is 1. Therefore, the number of selected features corresponds to the number of 1s in the position vector, representing the optimal subset of features chosen by the algorithm. Algorithm 4 further details how the MGWO algorithm effectively identifies the optimal feature subset. The most important features selected through this method, after applying MGWO to the uncorrelated features, include texture-mean, area-mean, concavity-mean, symmetry-mean, fractal-dimension-mean, area-se, concavity-se, smoothness-worst, and fractal-dimension-worst. These nine features were found to be the most significant for efficiently identifying breast cancer and achieving optimal classification accuracy.

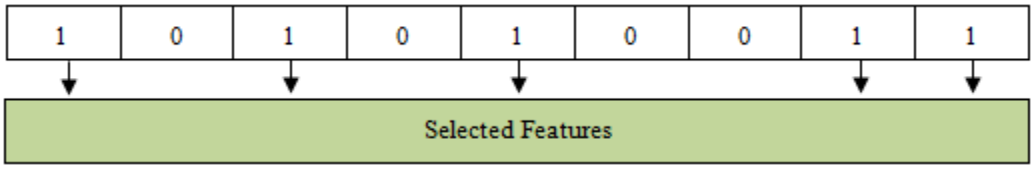


Figure 37. Representation of feature selection technique with MGWO.

Algorithm 3. Pseudocode of Modified GWO
Input: - Dataset, Number of features (Dim), Population Number of Iteration Output: Minimum number of selected features by MGWO initialize alpha, beta, and delta positions Initialize alpha pos, beta pos, and delta pos Initialize the positions of search agents For each Iteration For each Searchagent no - Calculate objective function for each search agent - Update Alpha pos, Beta pos, and Delta pos end For For each Searchagent no For each features Update the Position of search agents including omegas using Equations (1)-(6) and Equation (8) end For end For end For return Alpha pos.

Algorithm 4. The optimal subset of features using modified GWO.

For each feature in alpha pos[i] (i=1,2,. . . ,Dim)
if (alpha pos[i] > 0, 5)
alpha pos[i] =1
Else if (alpha pos[i] < 0, 5)
alpha pos[i] = 0
End if
End for

5.3. Breast cancer classification steps

Following the feature selection process, the classification step aims to accurately distinguish between malignant and benign breast cancer cases using the most relevant features. In this work, three popular supervised machine learning classifiers were applied:

Support Vector Machine (SVM), Random Forest (RF), and Naïve Bayes (NB). Each classifier was trained and evaluated using the features selected by the metaheuristic-based optimization process. SVM was used due to its strong performance in high-dimensional spaces and its ability to find optimal hyperplanes for classification. Naïve Bayes, despite its simplicity and assumptions of feature independence, was included for its fast computation and effectiveness on small datasets. Random Forest, an ensemble learning method based on decision trees, was chosen for its robustness against overfitting and its ability to model complex, non-linear relationships.

The performance of each classifier was assessed using several standard evaluation metrics, including accuracy, precision, sensitivity, specificity, F1-score, and the area under the ROC curve (AUC). The results showed that the Random Forest classifier consistently outperformed both SVM and NB across these metrics. Its ensemble nature allows it to better capture interactions among features, making it particularly effective when working with the optimized feature subset. These findings indicate that RF is a suitable and reliable choice for breast cancer classification when combined with a robust feature selection method, offering both high classification accuracy and generalization capability.

5.4. Experimental Results

In this study, feature selection (FS) was executed using a correlation-based technique integrated with a Modified Grey Wolf Optimization (GWO) algorithm. To evaluate the effectiveness of the selected features, multiple machine learning classifiers, including Support Vector Machine (SVM), Random Forest (RF), and Naïve Bayes (NB) were employed. The proposed hybrid approach was validated using the Wisconsin Diagnostic Breast Cancer (WDBC) dataset.

The experimental setup was developed in Python, with the Modified GWO configured to run for 20 iterations using 10 search agents. All simulations were conducted on a system equipped with an Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz and 8GB of RAM. This configuration was chosen to balance computational efficiency with the capability to process feature-rich biomedical data. The results demonstrate the robustness and classification accuracy improvements achieved by incorporating the enhanced feature selection strategy.

5.4.1. Comparison of different performance metrics between different classifiers

The table 7 presents a comparative evaluation of three machine learning classifiers including SVM, RF, and NB based on five standard performance metrics: precision, F1-score, sensitivity, specificity, and accuracy. This analysis specifically emphasizes the performance of the Correlation–Modified GWO in the context of breast cancer classification. Among the evaluated classifiers, combining Correlation-MGWO with Random Forest demonstrates the most balanced and reliable performance in breast cancer classification. It achieves the highest accuracy (99.12%), indicating superior overall predictive power. Furthermore, its high precision, sensitivity, and F1-score, along with its strong specificity, confirm its robustness and efficacy in distinguishing between malignant and benign tumors. These results suggest that Correlation-MGWO with Random Forest is a highly effective model for clinical decision

support systems aimed at breast cancer diagnosis, offering a compelling combination of accuracy, reliability, and interpretability.

Table 7. Comparison of different performance metris between different classifiers for breast cancer classification using the data of Confusion Matrix.

Evaluation measurement	SVM	RF	NB
Precision	**100%**	97,6%	100%
F1-score	**96,4%**	95,2%	92,5%
Sensitivity	93%	93%	86%
Specificity	**100%**	98,6%	100%
Accuracy	97,4%	**99,12%**	96,5%

5.4.2. Comparing the classification accuracy between CBGWO (Correlation + Base GWO) and CMGWO (Correlation + Modified GWO)

Table 8 presents a comparative analysis of classification accuracy for three machine learning algorithms including RF, SVM, and NB, the algorithm evaluated under three experimental conditions: (i) without feature selection, (ii) with feature selection using Correlation and Base Grey Wolf Optimizer (CBGWO), and (iii) with feature selection using Correlation and Modified Grey Wolf Optimizer (CMGWO). This analysis illustrates the effectiveness of the Modified Grey Wolf Optimizer-based feature selection techniques in enhancing classification performance, particularly for breast cancer diagnosis.

Overall, the results clearly demonstrate that feature selection plays a crucial role in improving classifier performance. Among the feature selection techniques, the Correlation + Modified GWO (CMGWO) consistently yields the highest accuracy across all classifiers, confirming its superiority in identifying informative and non-redundant features. Random Forest shows the best absolute performance across all scenarios, but the most substantial relative improvement is observed in SVM. These findings validate the effectiveness of the proposed CMGWO approach as a robust feature selection strategy for breast cancer classification tasks.

Table 8. Comparison of classification accuracy using proposed approach between CBGWO and CMGWO.

Classifiers	Without Feature selection	CBGWO	CMGWO
RF	97.07%	98.83%	**99.12%**
SVM	92.10%	92.98%	97.36%
NB	94.40%	93.85%	96.50%

5.4.3. Comparison of the classification accuracy between different classifiers using ROC curve (receiver operating characteristic curve)

ROC curve helps to better understand the power of a machine learning algorithm. We can easily observe in Figure 31 that RF is the perfect classifier. The Area Under the Curve (AUC) is the measure of the ability of a classifier to distinguish between classes, and it is used as a summary of the ROC curve. The higher the AUC, the better the performance among classifiers. From Figure 38, we see that RF gives good results compared with SVM and NB classifier in terms of ROC-AUC metric by achieving an AUC criterion equal to 99,3%.

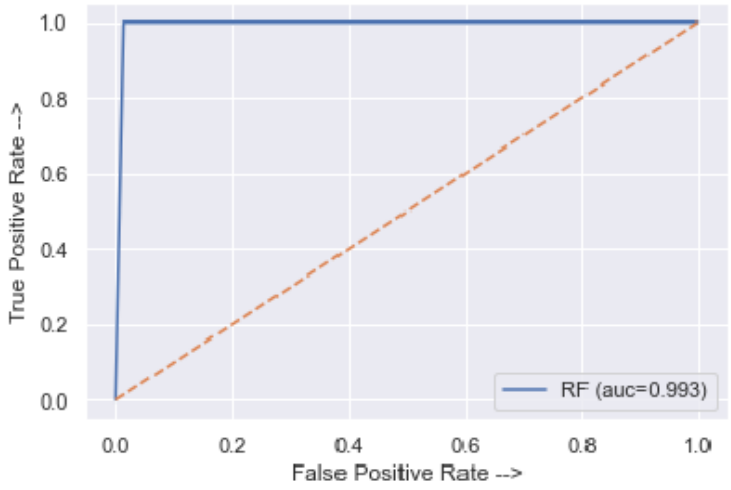


Figure 38. ROC curve metric of RF classifier.

The second best classifier was SVM by obtaining 97% as shown in Figure 39. Figure 40 represents the ROC-AUC metric obtaining by NB classifier and achieving 94,6%.

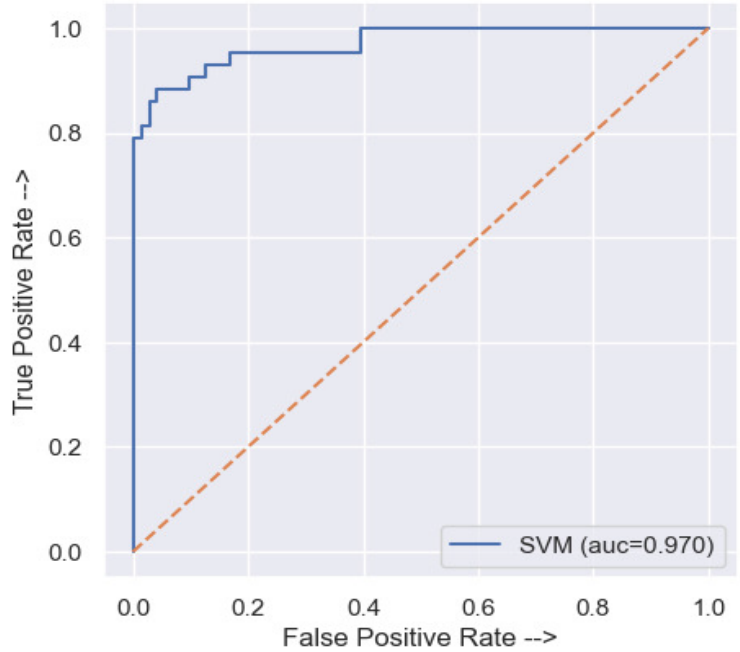


Figure 39. ROC curve metric of SVM classifier.

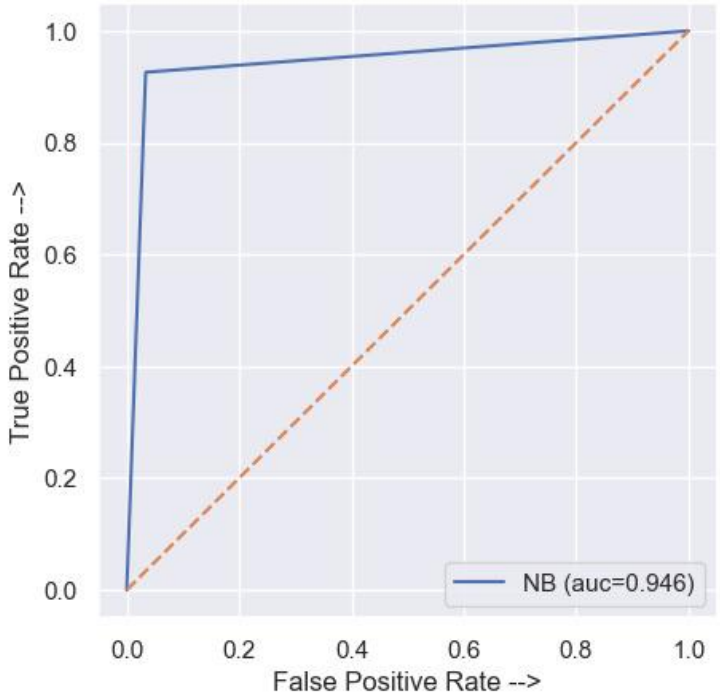


Figure 40. ROC curve metric of NB classifier.

5.4.4. Comparison of the suggested method with existing works

Table 9 presents a comparison of the classification accuracy achieved by various feature selection methods used in conjunction with different classifiers for the task of breast cancer classification. The table includes results from prior studies, as well as the performance of the proposed approach, which utilizes Correlation + Modified Grey Wolf Optimizer (CMGWO)**.**

The comparison in Table 9 shows that the proposed CMGWO method with Random Forest achieves the highest classification accuracy (99.12%**),** surpassing all other existing feature selection methods and classifiers. Although the performance of the proposed method with SVM and Naïve Bayes is slightly lower compared to other feature selection techniques like GOA, the overall results highlight the robust nature of the proposed CMGWO method, especially when used with Random Forest. This suggests that CMGWO is an effective feature selection technique that can significantly enhance classification performance, particularly with more complex classifiers like Random Forest.

Table 9. Evaluation of the proposed method by comparing results with existing feature selection methods.

Authors	Feature Selection Technique	Classifier	Accuracy (%)
Darzi et al. [9]	Genetic Algorithm	Case-based reasoning (CBR)	97.37
A. Rahmani et al. [10]	Feature Selection with GOA	SVM	98.83
S. Kumar and M. Singh [8]	Feature Selection with Enhanced GWO-SVM	SVM	98.24
Ibrahim et Nazir. [48]	Correlation + Principal Component Analysis	Ensemble machine learning	98.24
Proposed	Proposed-CMGWO	SVM	97.36
		NB	96.5
		RF	**99.12**

6. Conclusion

Integrating machine learning with metaheuristic algorithms has proven to be an effective strategy for solving a wide range of complex problems across various domains, particularly in image processing. This hybrid approach leverages the predictive power of machine learning models and the global optimization capabilities of metaheuristics to improve accuracy, robustness, and adaptability in tasks such as image segmentation, classification, and feature selection.

In this chapter, we presented an effective approach for breast cancer classification by integrating a feature selection method based on the Modified Grey Wolf Optimization (MGWO) algorithm with various machine learning classifiers. The MGWO was employed to identify the most relevant features, reducing data dimensionality while preserving critical diagnostic information. Several classifiers were tested, with random forest showing particularly strong performance when combined with MGWO. On the other hand, we also proposed a novel hybrid approach that combines correlation-based analysis with the MGWO for feature selection in breast cancer classification. In this method, correlation is first used to eliminate redundant or highly correlated features that may negatively impact the performance of the classifier. Subsequently, MGWO is employed to optimize the selection of the most relevant subset of features, enhancing the discriminative power of the model. This combination leverages the simplicity and effectiveness of correlation filtering with the robust exploration and exploitation capabilities of MGWO, resulting in improved classification accuracy and reduced computational complexity. The experimental results demonstrated that the proposed hybrid approaches improves classification accuracy and overall diagnostic reliability. These findings confirm the potential of intelligent feature selection in enhancing machine learning-based medical diagnosis systems.

However, one of the major challenges of this integration is the high computational cost, especially when dealing with large-scale image datasets or high-dimensional feature spaces. To address this issue, we explore the use of parallel metaheuristic algorithms deployed on distributed systems. By distributing the computation across multiple processing nodes, we aim to significantly reduce execution time while maintaining or even improving solution quality. This parallelization strategy allows for more efficient exploration of the search space, making it feasible to apply hybrid Metaheuristic-ML approaches in real-time or large-scale image processing applications.

Chapter 5

Parallel metaheuristic for image segmentation

1. Introduction

Image segmentation methods often struggle with the computational demands of high-resolution data and the complexity of extracting meaningful regions. To overcome these challenges, this section proposes two parallel metaheuristic-based segmentation approaches designed to improve both processing speed and segmentation quality. The first approach involves a Parallel Whale Optimization Algorithm (WOA) combined with K-Means clustering, implemented using Python's multiprocessing module. By distributing the optimization and clustering tasks across multiple CPU cores, this method accelerates the segmentation process while effectively exploring the solution space to enhance accuracy. The second approach utilizes a Parallel GWO integrated with Fuzzy C-Means (FCM) for MRI brain image segmentation, with key computations such as membership updates and centroid adjustments offloaded to the GPU. This GPU-based strategy leverages the parallel processing power of modern hardware to significantly reduce computation time and enable more iterations, resulting in improved convergence and segmentation quality. Overall, both parallel implementations demonstrate that harnessing multi-core CPUs and GPUs can minimize execution time and boost the effectiveness of metaheuristic-driven image segmentation techniques.

To accelerate the segmentation process, parallel computing techniques are employed using multiprocessing and GPU acceleration. Multiprocessing enables concurrent execution of segmentation tasks across multiple CPU cores, reducing processing time for large-scale images. Meanwhile, GPU-based parallelism significantly speeds up iterative optimization and clustering processes, making it feasible to handle terabyte-scale datasets efficiently. Frameworks such as pytorch, tensorFlow, and CUDA facilitate GPU acceleration, allowing deep learning models and optimization algorithms to execute in parallel. By leveraging machine learning, metaheuristic optimization, and parallel computing, this research aims to advance high-performance image segmentation for applications requiring large-scale data processing, such as medical imaging, remote sensing, and real-time object detection.

2. Parallel Whale Optimization Algorithm-Kmeans for image segmentation using Multiprocessing

To improve the quality and efficiency of image segmentation, the present work targets two core objectives including optimizing cluster centroids and accelerating the segmentation process. First, the Whale Optimization Algorithm (WOA) [14] is employed to determine the optimal centroids for each cluster, providing a strong initialization for the subsequent

segmentation step. These centroids are then utilized by the K-means algorithm, which performs the actual image segmentation based on the optimized positions identified by WOA.

To address the second objective which is computational acceleration, we introduce a parallelized implementation of this hybrid strategy using the multiprocessing framework. By distributing the WOA optimization across multiple processing units, we significantly reduce execution time while maintaining segmentation accuracy.

This section is structured as follows: we begin with a comprehensive overview of the WOA algorithm and its mathematical formulation, followed by a description of the K-means clustering technique. Next, we detail the integration of these methods into the hybridWOA-K-means framework, and finally, we elaborate on the parallelization strategy employed to enhance performance on multi-core systems.

2.1. Whale Optimization Algorithm (WOA)

To address numerical optimization problems, the Whale Optimization Algorithm (WOA) was introduced by Mirjalili and Lewis in 2016 [14]. This algorithm is inspired by the social behavior and bubble-net hunting strategy of humpback whales, humpback whales consider as one of the largest mammal species on Earth. The distinctive hunting technique used by humpback whales, known as bubble-net feeding, serves as the foundation for WOA's design. Algorithm 5 presents the pseudo-code for the whale optimization algorithm. The algorithm is based on three core phases: encircling prey, the bubble-net attacking method, and searching for prey. The mathematical models corresponding to these strategies are detailed in the following subsections.

2.1.1. Encircling Prey. At first, whales detect the position of prey and encircle it. this process is simulated by WOA. The global optimal solution is treated as the prey, while the other candidate solutions modify their places in reference to the global optimal solution, the location of the candidate solution, $\vec{X}\ (t+1)$ is calculated by the two following equations:

$$\vec{D} = |\vec{C}\ .\overrightarrow{X^*}(t) - \vec{X}(t)| \tag{11}$$

$$\vec{X}(t+1) = \overrightarrow{X^*}(t) - \vec{A}\ .\vec{D} \tag{12}$$

where $\overrightarrow{X^*}(t)$ is the global optimal solution (the best position recorded), $\vec{X}(t)$ denotes the position of candidate solution in the current generation (t) (denotes the best position recorded), t refers to the number of current iterations, and $\vec{A}$ and $\vec{D}$ are coefficient vectors, which are calculated as follows:

$$\vec{A} = 2\vec{a}\ .\vec{r} - \vec{a} \tag{13}$$

$$\vec{C} = 2\vec{r} \tag{14}$$

where $\vec{a}$ decreases linearly from 2 to 0 over iterations and $\vec{r}$ is randomly generated vector range in [0, 1]. the global optimal solution gained in the current generation can be used to

adjust the position of the candidate solution. The position of candidate solution could be updated via altering the values of $\vec{A}$ and $\vec{C}$.

2.1.2. Bubble-Net Attacking Method. Two methods are used in this step and they are designed as follows :

- **Shrinking Encircling Mechanism**: Equations (12) and (13) define the mathematical model of the shrinking encircling process. The value of $\vec{A}$ depends on the change of $\vec{a}$. In other words, by assigning a random number to $\vec{A}$ in [−1, 1], the new position of the candidate solution will be found between the current solution and global optimal solution.
- **Spiral Updating Position:** this technique calculates the new position of the candidate solution. The distance between the present candidate solution and global optimal solution is first calculated by Equation (15). Then the new position is generated by Equation (16).

$$\overrightarrow{D'} = |\overrightarrow{X^*}(t) - \vec{X}(t) \tag{15}$$

$$\vec{X}(t+1) = \overrightarrow{D'} . e^{bl} . \cos(2\pi l) + \overrightarrow{X^*}(t) \tag{16}$$

where b are constants that define the geometry of the logarithmic spiral and l is randomly generated range in [−1, 1], respectively.

there is a 50% probability of successful exploitation (attacking the prey) by using either a shrinking mechanism or a spiral model, and this is manipulated by a random number pro ∈ [0, 1]. The mathematical formula is :

$$\vec{X}(t+1) = \begin{cases} \overrightarrow{X^*}(t) - \vec{A} . \vec{D} & \text{pro} < 0.5 \\ \vec{X}(t+1) = \overrightarrow{D'} . e^{bl} . \cos(2\pi l) + \overrightarrow{X^*}(t) & \text{pro} > 0.5 \end{cases} \tag{17}$$

2.1.3. Search for Prey. Whales search at random based on their position. By using Equation (13), these processes can be utilized in WOA. $\vec{A}$ is designed as a random value less than 1 or higher than −1; this makes a chance for WOA to execute a random search. Thus, the purpose of this method is to enhance WOA's exploratory capabilities. When $\vec{A}$is greater than 1, it enables WOA to execute a wide search, the following formula is the model:

$$\vec{D} = |\vec{C} . \vec{X}_{rand}(t) - \vec{X}(t)| \tag{18}$$

$$\vec{X}(t+1) = \vec{X}_{rand}(t) - \vec{A} . \vec{D} \tag{19}$$

where $\vec{X}_{rand}$ denotes a random position of whale, $\vec{A}$ is less than 1, and the global optimal solution of the current iteration is selected for updating candidate solutions.

Algorithm 5 : Pseudo-code of the WOA algorithm

```
Initialize the whales population Xi(i=1,2, …,n)
Calculate the fitness of each solution
X*=The best search agent.
While i< maximum number of iteration do
        For every solution do
                Update a, A,C and p
                If (p<0.5) then
                        If (|A|<1) then
                                Update the postion of the current solution using Equation (12)
                          Else If (|A|>1) then
                                Random solution is generated
                                Update the position of the current solution using Equation (19)
                Else if (p>=0.5) then
                        Update the position of the current solution using Equation (16)
        End
        Check whether any solution exceeds the search space and adjust it
        Compute the fitness of every solution
        Update X* if there is a better solution
        t=t+1
End
Return X*
```

2.2. Hybrid Whale Optimization Algorithm -Kmeans

In this work, an enhanced image segmentation technique is introduced through a hybridization of the Whale Optimization Algorithm (WOA) with the K-means clustering method. This integrated approach leverages the global search capability of WOA, which is an evolutionary, nature-inspired metaheuristic, to identify optimal solution regions within the search space, while the K-means algorithm subsequently refines cluster centroids to achieve high quality segmentation. By combining these two strategies, the proposed method aims to enhance segmentation effectiveness across multiple evaluation metrics. The algorithm's structure is detailed in Algorithm 6. To steer the optimization process, the Sum of Squared Errors (SSE) is employed as the objective function, which focuses on minimizing intra-cluster variance to ensure pixel homogeneity within each cluster. The segmentation procedure unfolds in five systematically organized stages:

1. **Initialization**: Each whale is initialized with a randomly generated set of centroids, serving as candidate solutions for image segmentation.
2. **Fitness Evaluation**: The SSE is computed for each whale's current centroid configuration, providing a quantitative assessment of clustering effectiveness.

3. **WOA Global Search**: The whale population is updated using WOA's position-update strategies, including the encircling prey model and spiral movement, to explore the solution space broadly.
4. **K-means Local Refinement**: Following each global search phase, K-means is applied to refine the centroid positions of each whale, thereby enhancing local clustering precision.
5. **Termination**: The algorithm concludes either upon reaching the predefined maximum number of iterations or when the SSE exhibits negligible improvement across successive iterations.

2.3. Parallel Whale Optimization Algorithm -Kmeans strategy

In the present study, a Parallel Whale Optimization Algorithm–K-means (PWOA-Kmeans) framework was developed with the dual objectives of accelerating the image segmentation process and enhancing key performance metrics such as accuracy, Peak Signal-to-Noise Ratio (PSNR), Root Mean Square Error (RMSE), and Structural Similarity Index (SSIM). The core concept involves executing the Whale Optimization Algorithm (WOA) in a parallel computing environment to optimize the cluster centroids for the K-means algorithm more efficiently.

In this parallel design, each computational process is tasked with optimizing a single whale, where a whale is represented by a unique set of candidate centroids. Thus, for a population size of N whales, Nparallel processes are launched, each Computing Processing Unit core independently refining the centroid positions of one whale. For example, if the number of whales is set to eight, eight concurrent processes are executed, each dedicated to the optimization of one distinct whale. A schematic overview of this parallel structure is depicted in Figure 41, where whales (W) are mapped to their corresponding processes (P).

The procedure begins with image loading and preprocessing, which includes flattening and normalizing the input image to prepare it for segmentation. Subsequently, a pool of worker processes is instantiated using the multiprocessing library, typically matching the number of available CPU cores to maximize computational throughput.

Next, the parallelized WOA-Kmeans algorithm is executed. The WOA global position update and the K-means local refinement steps are performed concurrently for each whale across the distributed processes. Upon completion of all processes, the solution yielding the lowest Sum of Squared Errors (SSE) is identified as the optimal whale. Finally, a refined K-means algorithm is applied using the selected optimized centroids, producing the segmented output image.

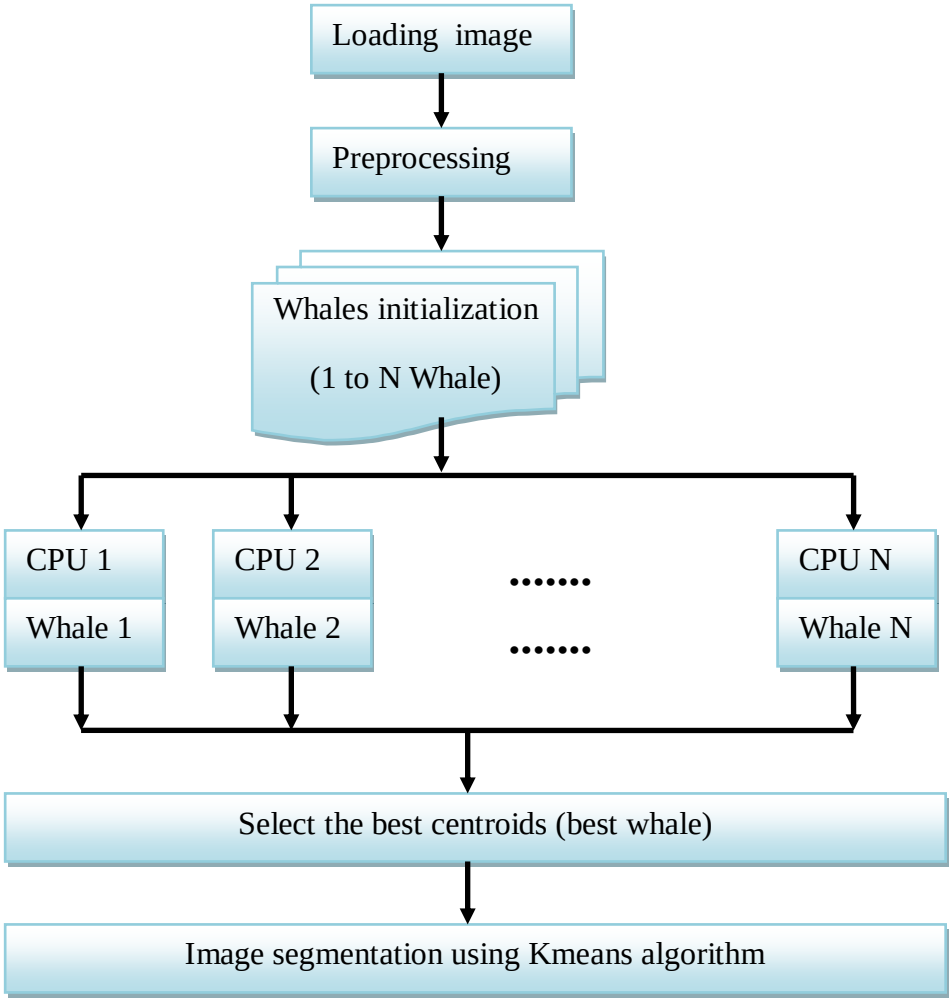


Figure 41. Flowchart of parallel WOA-Kmeans method.

2.4. Experimental results

To validate the robustness and efficiency of the proposed Parallel WOA–K-means (PWOA-Kmeans) approach, a comprehensive comparative analysis was conducted against its sequential counterpart under identical parameter settings. Both implementations were executed using 30 whales and 30 iterations. The parallel version leveraged Python's multiprocessing module and was deployed on an Intel(R) Core(TM) i7-1065G7 processor featuring 8 cores, each operating at 1.30 GHz.

The results underscore the advantages of parallelization, as the segmentation process significantly benefited from the integration of WOA with K-means in a parallel execution environment. To further demonstrate the method's generalizability and robustness, it was tested across a diverse set of grayscale images. These included three widely used benchmark images from the Berkeley Segmentation Dataset [255], including, Cameraman, Lena, and Baboon, as we see in figure 42 where images represented as Image 1, Image 2, and Image 3 respectively, as well as a Leukemia cell image (Image 4 from figure 42) obtained from the ALL-IDB medical imaging database[256].

Figure 42, Figure 43 and Figure 44 provide a visual representation of the segmentation results. Figure 42 displays the original input images, figure 43 presents the segmented outputs

using the sequential WOA–K-means approach, and figure 44 illustrates the segmented results obtained from the parallel PWOA–Kmeans method.

Initially, segmentation was performed without parallelization to establish a baseline. Subsequently, the same images were segmented using the multiprocessing-based parallel implementation. The final stage involved a comparative evaluation between the two models, focusing on various performance indicators such as segmentation accuracy, PSNR, RMSE, and SSIM to assess improvements in both computational efficiency and segmentation quality.

Image 1 Image 2 Image 3 Image 4

Figure 42.The original input images.

Image 1 Image 2 Image 3 Image 4

Figure 43.The segmented outputs using the sequential WOA–Kmeans approach.

Image 1 Image 2 Image 3 Image 4

Figure 44.The segmented results obtained from the parallel WOA–Kmeans method.

2.4.1. Comparing the time between sequential and parallel approachfor different images

As illustrated in Table 10, utilizing Parallel WOA to optimize the K-means clustering algorithm by identifying the optimal centroids for each cluster significantly reduces execution time across various test images. The results clearly demonstrate that the Parallel WOA-K-means consistently outperforms other approaches in terms of computational speed, delivering the fastest segmentation times for all tested images. In contrast, the sequential WOA-K-means exhibited the longest execution times, making it the least efficient method among those evaluated. These findings validate the effectiveness of parallelization in enhancing both performance and scalability of the hybrid segmentation framework.

Table 10. Comparing the computation time between sequential and parallel algorithm.

Images	Time (second)	
	Sequentiel approach	Parallel approach
Image 1	186,11	16.82
Image 2	131	16.13
Image 3	156	17
Image 4	120	11.5

2.4.2. Comparing several metric between sequential and parallel approach for different images

In this experiment, the effectiveness of the proposed approach is evaluated with a focus on its ability to determine optimal cluster centroids for improved image segmentation quality. As presented in Table 11, a comparative analysis between the Parallel WOA-Kmeansand itssequential counterpart is conducted using key performance metrics: Accuracy**,** RMSE**,** PSNR**,** and SSIM. The results clearly demonstrate that the Parallel WOA-Kmeans consistently outperforms the sequential version, delivering superior segmentation accuracy**,** lower reconstruction error**,** higher image fidelity, and enhanced structural preservation. This highlights the advantage of incorporating parallel optimization into the segmentation pipeline for both performance and quality enhancement.

Table 11. Comparing the performance metrics between parallel and sequential model.

	Parallel approach				Sequential approach			
	Accuracy	RMSE	PSNR	SSIM	Accuracy	RMSE	PSNR	SSIM
Image 1	98,37	7,52	30,60	0,86	98,52	7,53	30,59	0,85
Image 2	94,14	7,67	30,43	0,84	91,19	8,28	29,76	0,81
Image 3	96,97	8,22	29,82	0,88	95,09	8,19	29,85	0,86
Image 4	100	3,92	36,25	0,91	100	3,94	36,21	0,90

3. Parallelized Grey Wolf Optimizer-Based Fuzzy C-Means for MRI Segmentation on GPU

In the field of medical image analysis, accurate and efficient brain MRI segmentation plays a critical role in the diagnosis and treatment of neurological disorders. Traditional clustering-based segmentation methods often struggle with the complexity and noise inherent in MRI data. To address these challenges, we propose a parallel implementation of a hybrid Grey Wolf Optimizer–Fuzzy C-Means (GWO-FCM) algorithm for brain MRI segmentation using GPU acceleration. The Grey Wolf Optimizer (GWO), inspired by the leadership hierarchy and hunting behavior of grey wolves, is employed to optimize the initial cluster centers and improve the convergence of the FCM algorithm. By integrating GWO with FCM, the method achieves enhanced segmentation accuracy and robustness against intensity inhomogeneity and noise. Furthermore, to overcome the high computational cost typically associated with metaheuristic-based clustering, the proposed approach leverages the parallel processing capabilities of Graphics Processing Units (GPUs). GPU-based parallelization is used to accelerate both the GWO optimization process and the iterative updates of the FCM membership matrix and cluster centers. This parallel GWO-FCM framework significantly reduces execution time while maintaining high segmentation quality, making it well-suited for large-scale medical image analysis and real-time clinical applications.

3.1. Parallel MRI image segmentation using GPU

In medical literature, magnetic resonance imaging (MRI) is recognized as the most widely used modality for brain imaging, followed by computed tomography (CT), positron emission tomography (PET), and ultrasound [257, 258]. MRI is particularly favored due to its ability to provide detailed anatomical visualization of the entire brain, including the spinal cord and vascular structures, thanks to its superior contrast capabilities [259]. Unlike CT, MRI is non-ionizing, where MRI uses strong magnetic fields and radio waves to create detailed images of the body's internal structures. These radio waves do not carry enough energy to ionize atoms or molecules, making MRI a safer imaging option, especially for repeated use. as it does not expose patients to harmful radiation [260]. Among the most commonly utilized MRI sequences are T1-weighted, T2-weighted, and FLuid Attenuated Inversion Recovery (FLAIR) [261, 262].

Despite its advantages, MRI comes with certain challenges, where it requires expensive, high-performance machinery, and data acquisition and image reconstruction are often time-intensive, making efficient and accurate processing techniques essential for timely clinical decision-making [263–265]. Moreover, MRI brain images often suffer from artifacts, rendering segmentation a particularly complex and critical task in medical image analysis. Medical image segmentation [266] has garnered significant attention in recent years and remains a central focus in the field of biomedical imaging research [267]. It plays a pivotal role in delineating anatomical structures such as tumors, bones, organs, and critical brain regions. A wide array of algorithms has been developed to support this task, including thresholding, clustering, level set methods, active contours, and region-growing techniques [268], each offering unique advantages for specific imaging scenarios.

Even with these advancements, brain MRI segmentation continues to pose substantial challenges, largely due to the presence of imaging artifacts, intensity inhomogeneities, and anatomical variability. These complexities make it difficult to identify a universal segmentation strategy capable of consistently delivering optimal results across diverse datasets. Consequently, there is no one-size-fits-all solution that can comprehensively address the computational demands of brain image segmentation. To overcome these limitations, GPU-accelerated segmentation methods have emerged as a powerful alternative. These approaches aim to fulfill three main objectives: (1) enabling the comparative analysis of multiple segmentation algorithms, (2) facilitating the rapid and automated segmentation of large-scale medical image datasets, and (3) providing interactive visualization and segmentation tools that operate in real time. Leveraging the massively parallel architecture of modern GPUs, which can house hundreds of cores and support thousands of concurrent threads, technologies like CUDA-based parallel programming significantly enhance performance and scalability. As a result, GPU computing has become an indispensable tool for solving computationally intensive problems in medical imaging, particularly in the domain of segmentation.

In this section, we introduce a GPU-accelerated Parallel Grey Wolf Optimization-based Fuzzy C-Means (P-GWO-FCM) clustering framework tailored for efficient and accurate MRI image segmentation. The proposed method synergistically combines the exploratory strength of Grey Wolf Optimization (GWO) with the clustering precision of Fuzzy C-Means (FCM) to overcome the limitations associated with poor centroid initialization, a common drawback in traditional FCM. To further elevate segmentation quality, we incorporate Fuzzy Entropy as a fitness function, providing a more robust measure of uncertainty inherent in medical imaging data. This not only promotes the formation of well-defined and compact clusters but also enhances resilience to noise and intensity inhomogeneity commonly observed in MRI scans. On the other hand, the iterative nature of both GWO and FCM poses computational challenges, especially when dealing with large-scale, high-resolution images. To address this, the algorithm is parallelized using GPU architecture, enabling concurrent execution of key operations such as GWO position updates, fitness evaluations, and FCM membership calculations. This parallel strategy dramatically reduces computational time, accelerates convergence, and facilitates the practical deployment of the method in real-time or large-scale medical image analysis scenarios.

3.2. Fuzzy Entropy Clustering (FEC)

Fuzzy Entropy Clustering (FEC) is an advanced clustering technique that incorporates entropy-based regularization into fuzzy clustering frameworks to more effectively manage uncertainty and address the challenges of overlapping clusters [269, 270]. Unlike conventional fuzzy clustering approaches, FEC leverages an entropy measure to quantify the ambiguity in pixel-to-cluster assignments, thereby enhancing the algorithm’s sensitivity to vague boundaries and noise inherent in complex image data. At the core of FEC lies the principle of entropy minimization, where the fuzziness of the membership matrix is systematically reduced throughout the clustering process. This is achieved by iteratively computing the degree of membership for each data point relative to the cluster centroids,

while simultaneously minimizing the entropy to encourage more definitive assignments. The algorithm continues to refine the cluster centers and membership values until it converges to an optimal configuration. By embedding entropy as a guiding metric, FEC promotes clearer segmentation boundaries, robustly handles uncertainty, and yields well-separated, compact clusters. The mathematical formulation for fuzzy entropy, which plays a pivotal role in the optimization objective, is provided in Equation (20).

$$E = -\sum_{i=1}^{N}\sum_{j=1}^{C} u_{ij}\log(u_{ij}) \quad (20)$$

Where N represent the number of pixels, C the number of clusters and u_{ij} is the membership value of pixel i to cluster j.

3.3. Fuzzy-C Mean Grey Wolf Optimizer algorithm

We propose a hybrid segmentation framework that synergistically combines Grey Wolf Optimization (GWO) with Fuzzy Entropy (FE) as the objective function to refine the performance of the Fuzzy C-Means (FCM) clustering algorithm. By exploiting GWO's robust global search capabilities, this integration aims to optimize the selection of cluster centers, while the incorporation of fuzzy entropy significantly improves the algorithm's ability to handle uncertainty and overlapping regions which is a critical challenge in medical image segmentation.

Within this hybrid model, the GWO algorithm guides the optimization process using its biologically inspired mechanisms: encircling prey, hunting strategies, and the final attack phase, which together maintain a dynamic balance between global exploration and local exploitation. These iterative position updates ensure convergence toward more discriminative and stable cluster configurations. As a result, the enhanced FCM model benefits from sharper segmentation boundaries, increased resilience to noise, and improved accuracy. The conceptual structure of this hybrid GWO-FE-FCM method is visually depicted in Figure 45.

The segmentation process commences with the initialization of a wolf population, where each wolf encodes a candidate solution namely a potential set of cluster centroids. During the fitness evaluation phase, each solution is assessed using Fuzzy Entropy Clustering (FEC), which quantifies the uncertainty associated with cluster memberships. The objective is to minimize the fuzzy entropy, thereby promoting crisp cluster boundaries and improving the interpretability of segmentation results. Following fitness assessment, Grey Wolf Optimization (GWO) drives the position update phase, where the wolves' positions (i.e., centroids) are iteratively adjusted based on the hierarchy of the alpha, beta, and delta wolves. This biologically inspired mechanism guides the swarm toward promising regions of the search space through controlled exploration and exploitation. Each iteration refines the centroid configuration with the dual goal of minimizing entropy and improving segmentation fidelity. These two steps, the fitness computation and position adjustment are cyclically repeated until a termination condition is met, either upon reaching the predefined number of iterations or when convergence is achieved (i.e., negligible changes in fitness values across iterations). Once optimization concludes, the final refined centroids are fed into the Fuzzy C-Means (FCM) algorithm to perform the actual segmentation. Pixels are assigned to clusters

based on their fuzzy membership degrees, yielding a segmented image that delineates distinct regions according to intensity variations. This entire procedure is summarized in Algorithm 7.

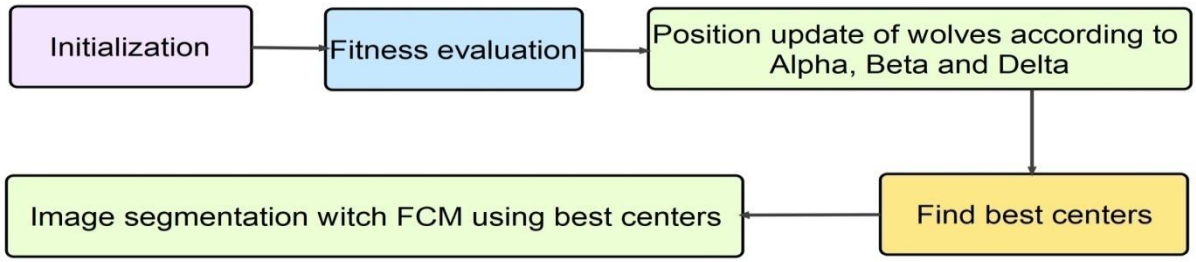


Figure 45. Diagram of hybrid GWO-FCM-FE.

Algorithm 7 : Pseudo-code of Hybrid GWO-FCM-FEC.

Input:
Grayscale image I of size M× N; Number of clusters C; Maximum number of iterations MaxIter

Step1:Initialization

- Randomly initialize the positions of N wolves (set of cluster) in the image intensity range.

Step 2: Iterative Optimization

For each iteration from t=1to MaxIter:

1. Evaluate Fitness:
   - For each wolf (cluster center set):
     - Compute fuzzy memberships μ_{ij} for each pixel based on the distance to the cluster centers.
     - Calculate the fuzzy entropy *E* using Equation 20.
     - Assign E as the fitness for the current wolf.
2. Update Leaders:
   - Identify α, β, δ: the three best wolves with the lowest entropy values.
3. Update Wolf Positions:
   - For each wolf :
     - Calculate the distance D to alpha wolf, Beta and delta wolf using Equation 11.
     - Update position using Equation 12.
     - Clip the updated position to stay within the search space(valid intensity range).

Step3:Segmentation

- After the final iteration, use the best wolf () cluster centers as best initial centroids for FCM algorithm.

3.4. Parallel GWO and Fuzzy C-Mean using Graphic Processing Unit

The proposed methodology integrates Parallel Grey Wolf Optimization (PGWO) and Parallel Fuzzy C-Means (PFCM) in a two-stage sequential framework to effectively optimize cluster centroids for MRI image segmentation, as illustrated in Figure 46. In the first phase, PGWO is employed to perform a global search for optimal centroids, using Fuzzy Entropy as the fitness function. Leveraging GPU acceleration, the algorithm parallelizes key components such as wolf position updates and fitness evaluations, enabling the simultaneous assessment of multiple candidate solutions across the search space.

Upon identifying the most promising set of centroids, the second phase executes the Parallel FCM algorithm using also GPU-accelerated. This stage benefits from fine-grained parallelism where membership matrix computations and cluster center updates are performed concurrently. Specifically, each pixel's membership degree is computed in parallel, while centroid updates utilize parallel reduction operations, substantially minimizing computational overhead and accelerating convergence.

The sequential deployment of PGWO and PFCM capitalizes on the strengths of both algorithms: PGWO provides a robust global search mechanism to initialize the clustering process effectively, while PFCM delivers precise local refinement to enhance segmentation quality. This hybrid approach strategically balances exploration and exploitation, resulting in improved segmentation performance and computational efficiency when compared to traditional, non-parallel clustering methods.

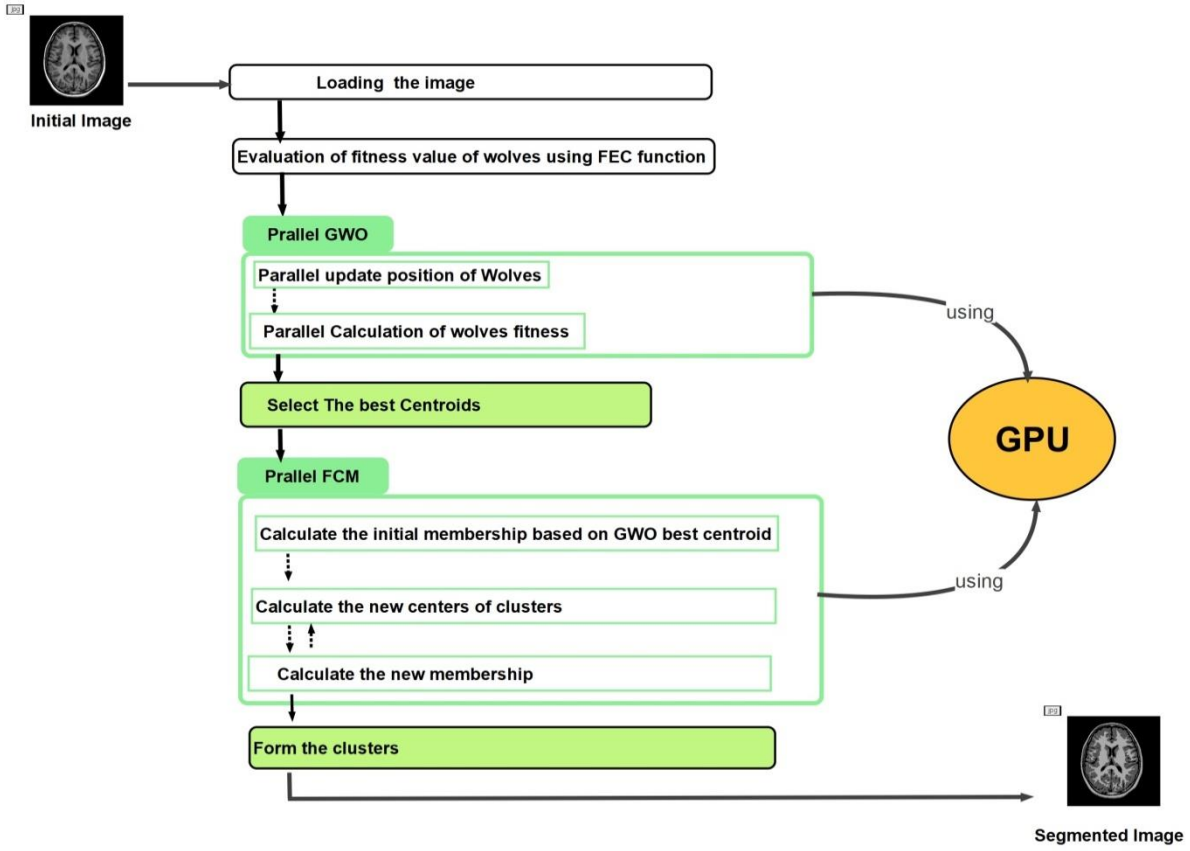


Figure 46. The process of proposed Parallel GWO and Parallel FCM.

Figure 47 presents a schematic diagram of the proposed hybrid approach, integrating Parallel PGWO and PFCM for MRI image segmentation. The framework begins with the execution of PGWO, guided by Fuzzy Entropy (FE) as the fitness function, to systematically explore the search space and determine optimal cluster centroids. This stage follows a well-defined sequence of operations designed to maximize search efficiency through GPU-parallelized computation.

Following the identification of candidate centroids by PGWO, the Parallel FCM algorithm is employed to perform refined segmentation. This phase further enhances the clustering precision by leveraging parallel computation for key operations such as membership updates and centroid recalculation.The comprehensive implementation details of both PGWO and PFCM are outlined in Algorithm 8 and Algorithm 9, respectively, offering a step-by-step procedural breakdown of the parallel mechanisms employed in each stage of the segmentation pipeline.

Algorithm 8 : Pseudo-code of PGWO.

For each iteration from t=1 to MaxIter:

Step 1: Parallel fuzzy membership calculation steps:

- Assign each pixel (or block of pixels) to a thread.
- Compute distance between the pixel intensity and all cluster centers.
- Update fuzzy membership μ_{ij} for the pixel *i* in each cluster *j* using :

$$\mu_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{d_{ij}}{d_{ik}}\right)^{2/(m-1)}}$$

Where *m* is the fuzziness factor.

- store μ_{ij} in global memory.

Step 2: Parallel fitness evaluation steps:

- Assign each wolf (set of centers) to a thread.
- Compute fuzzy entropy E using Equation (20).
- Store E in global memory for each wolf.

Step 3: Parallel GWO position update steps:

- Assign each wolf to a thread.
- Update positions using GWO algorithm Equations (11) and (12).
- Clip values to ensure valid intensity range.

End-for

Select Best wolf (Best Centers).

Algorithm 9 : Pseudo-code of PFCM.

Initialization using best centroid selected by P-GWO

For each iteration from t=1 to MaxIter:

Step1:

Initialize Membership Matrix in Parallel :

For Each pixel x_i is assigned an initial membership value for each cluster k.

Step2:

Compute Cluster Centers in Parallel:

For each cluster k:

- Compute weighted sum of all pixels based on membership.

Step3:

Compute New Membership Matrix in Parallel:

For each pixel x_i and cluster k:

- Compute distances d_{ki}
- Compute new membership values using the fuzzy rule.

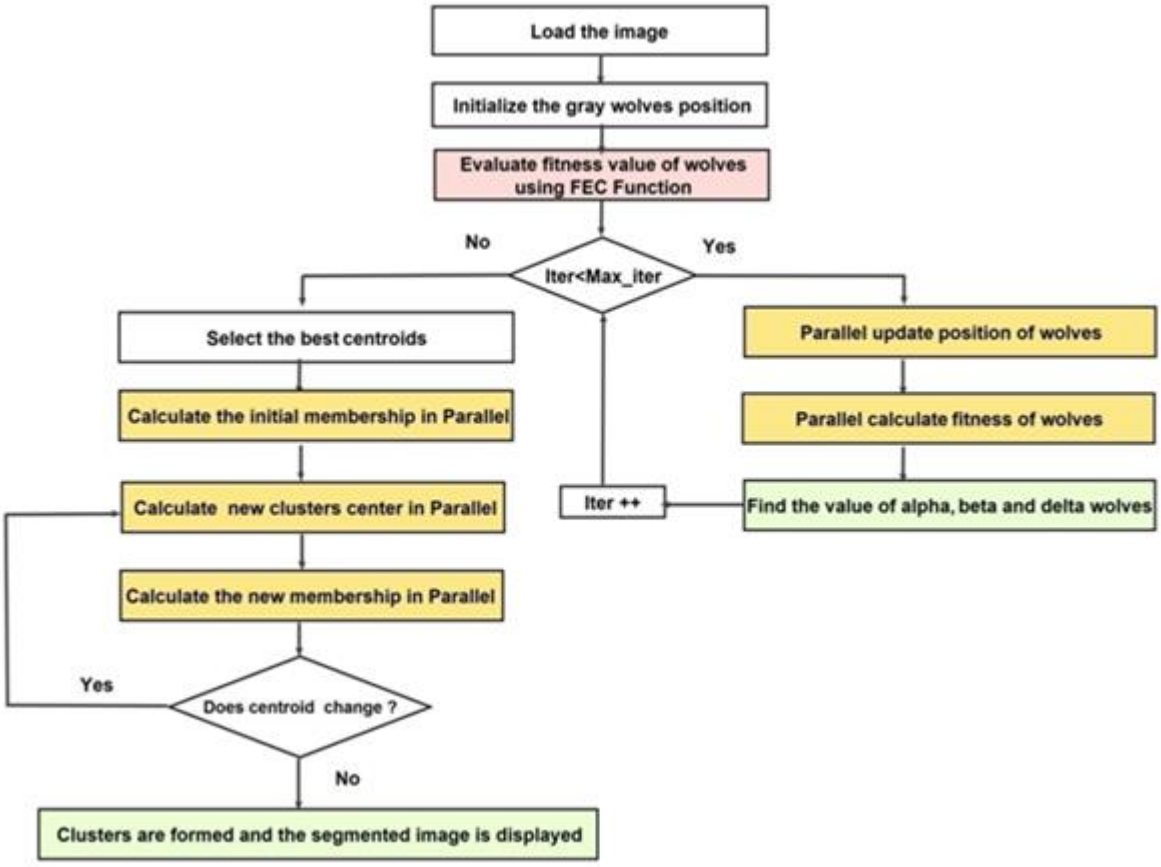


Figure 47. Diagram of the proposed P-GWO-FCM.

3.5. Experimental results

The experimental evaluation of the proposed Parallel Grey Wolf Optimization with Fuzzy C-Means (P-GWO-FCM) approach was conducted using three distinct datasets: a simulated brain tumor dataset [271], a real-world clinical MRI dataset sourced from Kaggle [272], and the dataset from the Radiological Society of NorthAmerica (RSNA), provided as part of a recent Kaggle competition [273]. These datasets were chosen to comprehensively assess the segmentation capability of the method across both synthetic and clinical imaging scenarios.

To quantitatively evaluate segmentation performance, several well-established metrics were employed: the Jaccard Index, Davies-Bouldin Index (DBI), Partition Coefficient Index (PCI), and Partition Entropy Index (PEI). These metrics collectively provide insight into segmentation accuracy, intra-cluster compactness, inter-cluster separation, and membership fuzziness. The proposed P-GWO-FCM technique was benchmarked against traditional Fuzzy C-Means (FCM), Sequential GWO-FCM, and other relevant state-of-the-art methods. This comparative analysis highlights the improvements achieved through parallelization and hybridization. The mathematical definitions and formulations of each evaluation metric are detailed below.

1. **Jaccard Index:** The Jaccard Index (JI) measures the similarity between two sets, commonly used for evaluating segmentation accuracy:

$$\mathrm{JI} = \frac{|A \cap B|}{|A \cup B|} \tag{21}$$

Where A is the ground truth and B is the segmented region.

2. **Partition Coefficient Index (PCI)**: PCI evaluates the compactness of clusters in fuzzy clustering, defined as:

$$\mathrm{PCI} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} u_{ij}^2 \tag{22}$$

where u_{ij} is the membership value of pixel iin cluster j, and N is the total number of pixels.

3. **Davies-Bouldin Index (DBI):** DBI assesses cluster compactness and separation, given by:

$$DBI = \frac{1}{C}\sum_{i=1}^{C} max_{i \neq j} \frac{s_i + s_j}{d_{ij}} \tag{23}$$

Where s_i is the dispersion of cluster i , and d_{ij} is the distance between cluster centroids.

4. **Partition Entropy Index (PEI):** PEI measures the fuzziness of cluster memberships:

$$PEI = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} u_{ij} \log u_{ij} \tag{24}$$

Whereu_{ij} is the membership value of pixel *i* in cluster *j*.

5. **Dice coefficient measure**: is a statistic measure used for comparing the similarity between two samples and ranges between 0 and 1:

$$Dice = \frac{2|A \cap B|}{|A| + |B|} \quad (25)$$

Where A and B are the segmented image and the ground truth image.

3.5.1. Evaluation on Simulated Brain Tumor Dataset

The effectiveness of the proposed P-GWO-FCM approach was rigorously evaluated using a simulated brain tumor dataset, with a focus on accurately segmenting key brain tissues including White Matter (WM), Gray Matter (GM), and Cerebrospinal Fluid (CSF). To quantitatively assess segmentation performance, the Jaccard Index was calculated for each tissue type, capturing the degree of spatial overlap between the segmented outputs and their corresponding ground truth regions. The evaluation was performed on a T1-weighted brain MRI scan with a resolution of 217 × 181 pixels. This setup was designed to test the robustness and precision of the segmentation under practical imaging conditions. Accurately delineating WM, GM, and CSF is of critical importance in clinical neuro imaging, particularly for diagnostic assessments in neurology and radiology.

Figure 48 offers a comprehensive visual comparison of the segmentation results across different methods. The original brain image is displayed in Figure 48*(a)*, while the manually annotated ground truth for WM, GM, and CSF is shown in Figure 48*(b)*. The segmentation outputs generated by traditional FCM, sequential GWO-FCM, and the proposed P-GWO-FCM method are illustrated in Figure 48(c), *48*(d), and *48*(e), respectively. This visual analysis underscores the performance enhancements introduced by incorporating GWO and parallel processing into the clustering framework.

Figure 48 effectively illustrates the superior performance of the P-GWO-FCM method in maintaining regional homogeneity, producing segmentation results that are both uniform and structurally coherent. Notably, the proposed approach demonstrates a remarkable ability to preserve fine anatomical details from the original MRI scans, an essential feature in medical image analysis, where subtle tissue variations can carry significant diagnostic implications. To quantitatively assess this performance, the Jaccard Similarity (JS) was computed for each of the evaluated methods: conventional FCM, Sequential GWO-FCM, and the proposed Parallel GWO-FCM. As summarized in Table 12, the average JS values obtained using the P-GWO-FCM method were consistently higher across all tissue classes (WM, GM, CSF), underscoring its enhanced segmentation accuracy.

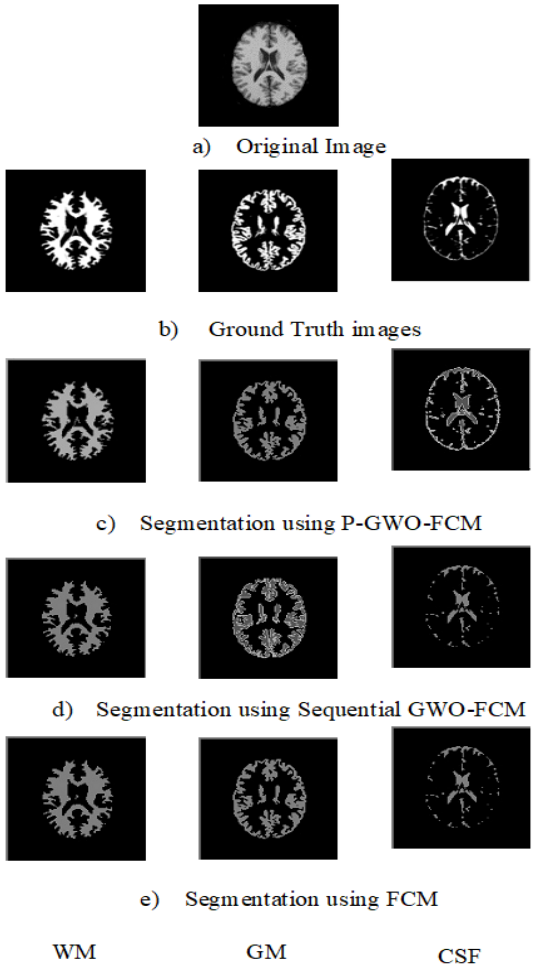


Figure 48. Segmentation results of WM, GM, CSF using FCM, sequential-GWO-FCM, and P-GWO-FCM.

Table 12 presents a comprehensive comparison of the segmentation outcomes based on the JS metric, which measures the degree of spatial overlap between the segmented outputs and the ground truth. A higher JS score directly correlates with improved accuracy, and the results clearly indicate that the proposed parallelized hybrid approach outperforms both the standalone FCM and its sequential hybrid variant.

Table 12. Jaccard similarity values for the three methods on simulated MR images.

Method	Jaccard Measure			Average
	WM	GW	CSF	
P-GWO-FCM	0,94	0,89	0,93	**0,92**
Sequential GWO-FCM	0,89	0,90	0,92	0,90
FCM	0,88	0,80	0,90	0,86

The P-GWO-FCM method clearly outperforms competing algorithms, attaining the highest Jaccard Similarity (JS) scores across all examined tissue classes: 0.94 for White Matter (WM), 0.89 for Gray Matter (GM), and 0.93 for Cerebrospinal Fluid (CSF). These individual results contribute to an impressive average JS value of 0.92, which exceeds the performance metrics of both traditional FCM and its sequential hybrid variant.

This superior performance highlights the efficacy of integrating Grey Wolf Optimization (GWO) with Fuzzy Entropy (FE) to enhance the FCM framework for brain MRI segmentation. The global search capabilities of GWO significantly mitigate the limitations of poor initialization and susceptibility to local optima, which are common challenges in conventional clustering approaches. Simultaneously, the incorporation of fuzzy entropy introduces greater robustness in uncertain or ambiguous regions, preserving fine-grained structural variations crucial for medical diagnosis.

Moreover, the parallel implementation of the algorithm delivers substantial computational speedups without compromising segmentation precision. This makes the proposed method particularly suitable for large-scale and real-time medical imaging applications, a benefit further demonstrated in the subsequent experimental results.

In addition to the internal comparisons, the proposed P-GWO-FCM method was benchmarked against established segmentation techniques, namely FCM-GENIUS [274] and Deep-JCR (Deep Joint Calibrationless Reconstruction) [275], both of which utilize the Dice Similarity Coefficient as the primary performance metric. As shown in Table 13, P-GWO-FCM consistently outperforms these state-of-the-art approaches, achieving superior Dice scores of 0.93 for White Matter (WM), 0.89 for Gray Matter (GM), and **0.95** for Cerebrospinal Fluid (CSF).

These results underscore the robustness and precision of the proposed hybrid method, which not only enhances segmentation accuracy but also maintains computational efficiency through parallel processing. The comparative advantage of P-GWO-FCM reaffirms the value of integrating metaheuristic global search and fuzzy entropy-based refinement into the FCM framework for high-resolution brain MRI segmentation tasks.

Table 13. Comparing the dice coefficient between the proposed method and the existing methods.

Methods	Dice coefficient		
	WM	GM	CSF
P-GWO-FCM	**0,93**	**0,89**	**0,95**
Sequential GWO-FCM	0,93	0,88	0,87
FCM	0,88	0,88	0,78
FCM-GENIUS [274]	0.73	0.76	0.19
Deep-JCR [275]	0.913	0.855	0.805

3.5.2. Evaluation on Clinical brain MRI Dataset

To further assess the algorithm's performance, experiments were conducted on a clinical MRI dataset [272] from Kaggle, which includes a variety of tumor patterns with different intensity distributions. The segmentation performance of the P-GWO-FCM method was compared to that of FCM and sequential GWO-FCM. The effectiveness of these three methods was evaluated using the DBI, PEI and PCI metrics, with the results presented in Table 14. Figure 49 show cases a selection of MRI images from the clinical dataset used in the experiments, displaying 10 images of varying sizes to demonstrate the robustness of the proposed approach. These images were segmented using all three algorithms. Figure 50 illustrates the segmentation results obtained for these 10 brain MRI images using the P-GWO-FCM method.

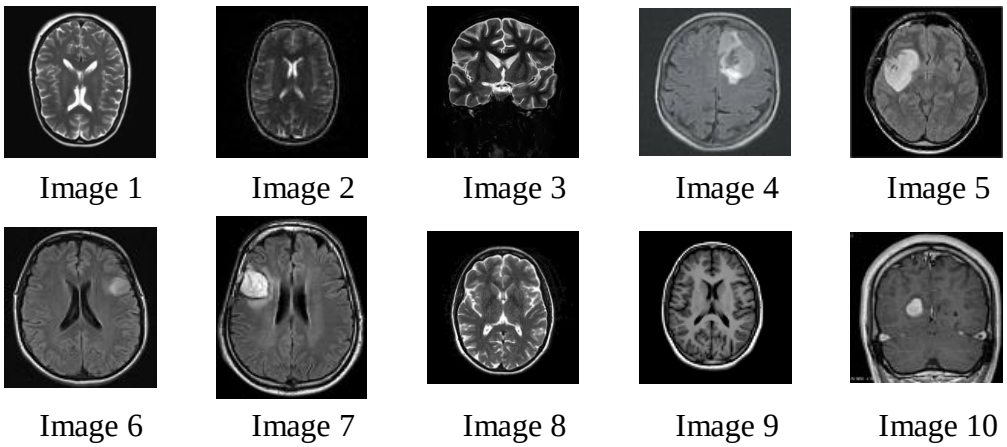


Figure 49. Samples of brain MR images from clinical dataset.

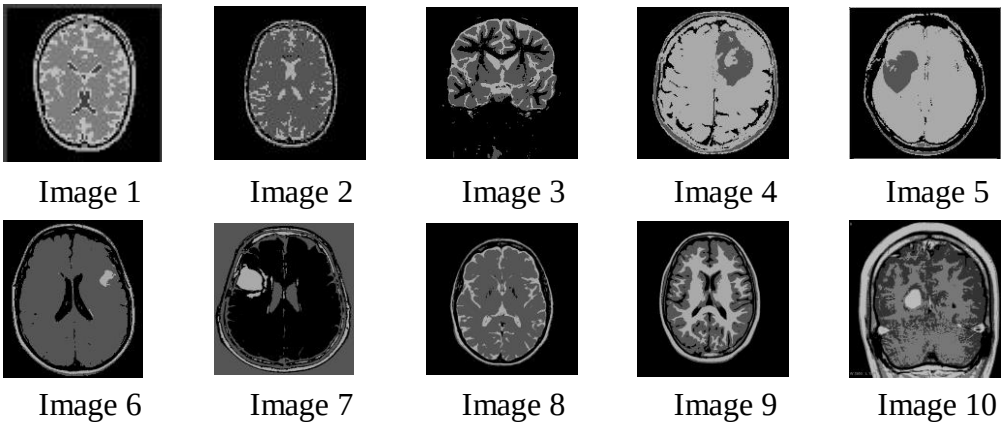


Figure 50.Segmentation results on the clinical brain MR Images with P-GWO-FCM.

A visual analysis of the segmented images reveals that the P-GWO-FCM method excels in clarity, detail preservation, and precise delineation of tissue boundaries. The results in Table 14 provide a comprehensive comparison of P-GWO-FCM, traditional FCM, and Sequential-GWO-FCM, demonstrating that P-GWO-FCM consistently outperforms the other methods across various evaluation metrics. These metrics assess clustering quality, particularly in

terms of cluster compactness, separation, and the clarity and reliability of cluster assignments. Regarding the DBI, which evaluates clustering quality by measuring compactness and separation, P-GWO-FCM achieved an average DBI value of 0.30, significantly lower than those of FCM and sequential GWO-FCM. This result indicates better cluster definition and separation, improving segmentation quality. The PEI, which quantifies uncertainty in membership assignments, further supports the robustness of our approach, as P-GWO-FCM attained an exceptionally low average PEI value of 0.25.This low PEI value reflects the minimal overlap between clusters, demonstrating the algorithm's ability to assign data points with higher confidence and precision. Additionally, the PCI measures clustering fuzziness and further demonstrates the superiority of P-GWO-FCM. The algorithm achieved an impressive average PCI value of 0.91, indicating clearer partitioning with reduced fuzziness. This high PCI value, consistent across all test images, confirms that cluster memberships are predominantly close to 0 or 1, leading to more definitive segmentation.

Table 14. Comparing of FCM, sequential GWO-FCM and P-GWO-FCM using Different metrics.

Images	FCM			Sequential-GWO-FCM			P-GWO-FCM		
	DBI	PEI	PCI	DBI	PEI	PCI	DBI	PEI	PCI
Image1	0.32	0.20	0.90	0.30	0.18	0 .88	0.30	0.18	0.92
Image2	0.35	0.22	0.87	0.33	0.20	0.89	0.24	0.17	0.93
Image3	0.29	0.31	0.89	0.19	0.31	0 .90	0.15	0.27	0.90
Image4	0. 39	0.31	0.86	0.42	0.31	0.86	0.43	0.30	0.91
Image5	0.25	0.28	0.91	0.25	0.30	0.93	0.20	0.28	0.92
Image6	0.35	0.26	0.88	0.33	0.26	0.90	0.31	0.26	0.90
Image7	0.36	0.24	0.90	0.36	0.24	0.91	0.30	0.19	0.91
Image8	0.40	0.30	0.87	0.37	0.30	0.87	0.37	0.35	0.92
Image9	0.39	0.32	0.87	0.40	0.32	0.89	0.35	0.24	0.89
Image10	0.42	0.30	0.86	0.39	0.33	0.87	0.36	0.33	0.90
Average	0.35	0.27	0.88	0.33	0.27	0.89	0.30	0.25	0.91

Table 15 presents a comparison between the sequential GWO-FCM and the P-GWO-FCM approach in terms of execution time. The results clearly show that P-GWO-FCM outperforms the sequential algorithm, achieving significantly faster execution. In contrast, with sequential GWO-FCM, larger images require more time for segmentation. However, the proposed parallel approach maintains a consistently low segmentation time regardless of image size, as illustrated in Figure 51.The execution time remains low even when using images with larger dimensions due to GPU acceleration, which efficiently processes parallel computations. The GPU's ability to handle multiple operations simultaneously distributes the workload across thousands of cores, significantly reducing processing time compared to the sequential approach.

Table 15. Comparing Time between sequential and P-GWO-FCM.

Images	Dimension x*y	Time (second)	
		Sequential-GWO-FCM	P-GWO-FCM
Image1	79*78	2,01	0,60
Image2	100*100	2,18	0,62
Image3	200*200	10,36	0,62
Image4	223*226	11,26	0,62
Image5	225*225	11,71	0,64
Image6	291*340	23,28	0,87
Image7	374*456	41,65	0,78
Image8	442*442	48,02	0,84
Image9	728*725	155,71	0,86
Image10	911*938	250,10	0,89

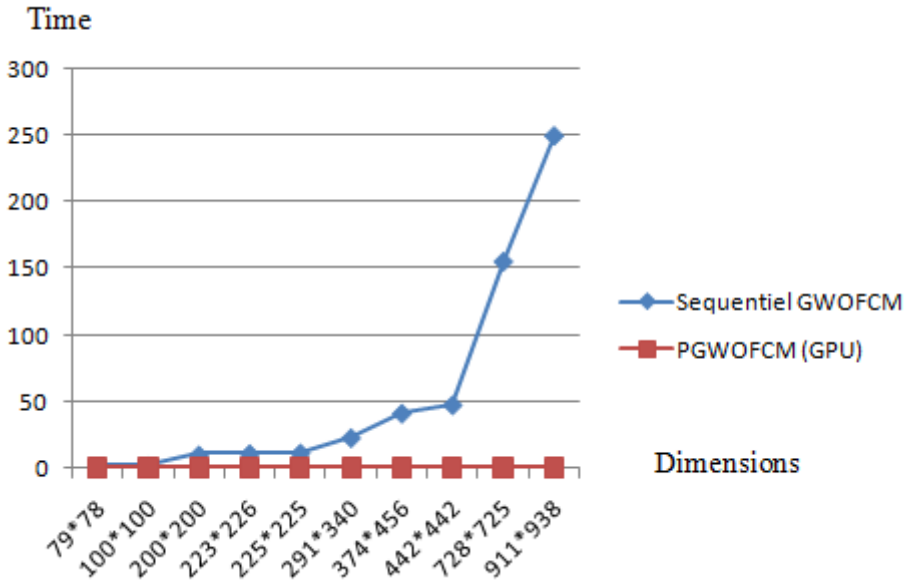


Figure 51. Comparing time between sequential-GWO-FCM and P-GWO-FCM on different sizes images.

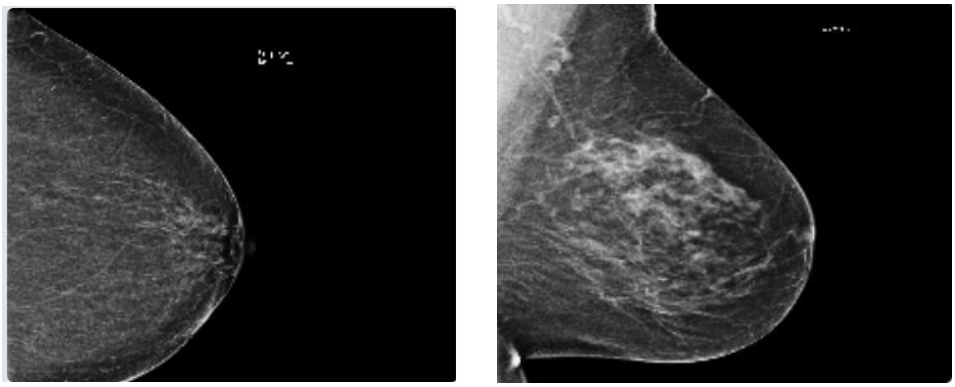
3.5.3. Evaluation on clinical breast cancer disease dataset

Another experimentation was conducted in this section, where the primary dataset utilized in this experiments is sourced from the Radiological Society of North America (RSNA), provided as part of a recent Kaggle competition [273]. This extensive dataset comprises 54,713 DICOM-format breast imaging studies, collected from approximately 11,000 patients. For each patient, a minimum of four images is included, captured from different breast laterality (left and right) and viewing angles, specifically craniocaudal (CC) and mediolateral oblique (MLO) views. The dataset is characterized by considerable diversity in both image

resolution and format, encompassing standard image formats such as JPEG and JPEG2000, and DICOM-specific pixel representations including monochrome1 and monochrome2. This heterogeneity presents a realistic challenge for preprocessing and model generalization. Figure 52 illustrates two representative samples from the RSNA dataset, showcasing one cancerous and one non-cancerous case, thereby highlighting the visual variation between healthy and pathological breast tissues.

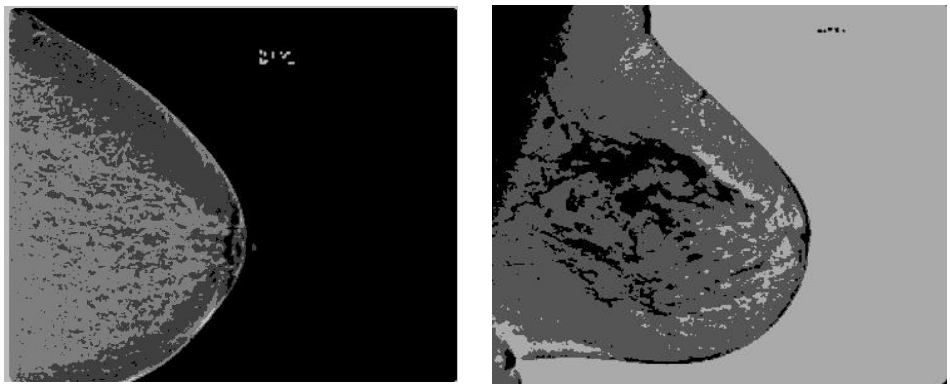
The visual results of the segmentation process are illustrated in Figures 52, 53, and 54. As shown in Figure 52, representative sample image from the dataset are presented to provide context for the segmentation task. Figure 53 displays the segmentation outcomes obtained using Sequential-GWO-FCM, highlighting its effectiveness in delineating key image regions. In contrast, Figure 54 presents the results produced by the P-GWO-FCM, allowing for a visual comparison between the two approaches in terms of segmentation quality and accuracy.

The results from Table 16 demonstrate that P-GWO-FCM method for image segmentation significantly outperforms the sequential method in terms of quality and computation time. Specifically, the parallel method achieved a Peak Signal-to-Noise Ratio (PSNR) of 31,97 and a Root Mean Square Error (RMSE) of 3.31, indicating a higher fidelity reconstruction and lower error compared to the sequential method. These performance metrics highlight the superiority of the parallel method, as a higher PSNR and lower RMSE generally reflect better segmentation quality. On the other, in terms of computational efficiency, P-GWO-FCM demonstrated superior performance with the shortest segmentation time of 1,74 second, outperforming Sequential GWO-FCM, which required 38,4 seconds. This noticeable difference in processing time highlights the efficiency of P-GWO-FCM, making it a more suitable choice for applications where rapid image segmentation is critical. The reduced execution time also suggests better scalability and responsiveness, especially in real-time or large-scale medical imaging scenarios.



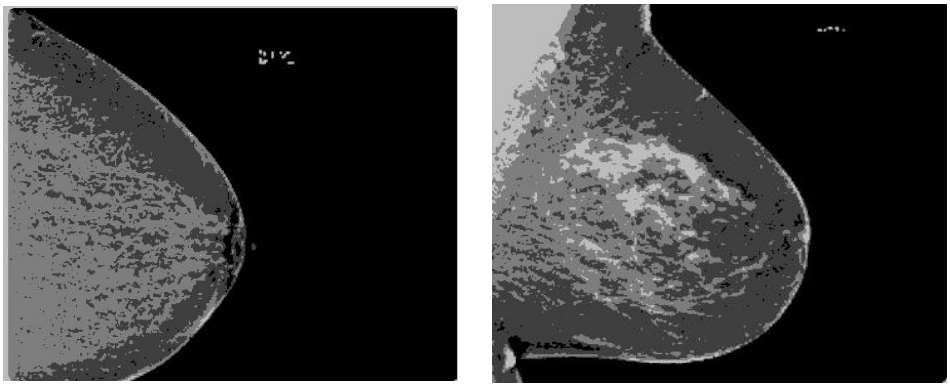
(a) (b)

Figure 52. Samples from the RSNA dataset, where (a) non-cancerous image, (b) cancerous image.



(a) (b)

Figure 53. The segmentation results obtained using Sequential-GWO-FCM where (a) non-cancerous image, (b) cancerous image.



(a) (b)

Figure 54. The segmentation results obtained using Parallel-GWO-FCMwhere (a) non-cancerous image, (b) cancerous image.

Table 16. Comparing segmentation results between the sequential-GWO-FCM and P-GWO-FCM on breast cancer images.

	Sequential-GWO-FCM	P-GWO-FCM
RMSE	6,42	3,31
PSNR	29,8	31,97
Time (second)	38,4	1,74

4. Conclusion

Parallel computing techniques have proven to be highly effective in addressing the computational challenges associated with large-scale image segmentation. By integrating machine learning, metaheuristic optimization, and parallel processing, significant improvements in segmentation accuracy and efficiency can be achieved. The use of multiprocessing on CPUs and GPU acceleration allows for the real-time processing of large datasets, making advanced segmentation techniques more accessible for practical applications. The combination of metaheuristic algorithms with ML-based segmentation offers a robust and adaptive approach to image segmentation. Methods such as parallel GWO-FCM on GPU and Parallel WOA-Kmeans using Multiprocessing demonstrate enhanced performance in terms of convergence speed and segmentation quality. Providing a scalable solution for diverse imaging domains. Future work can explore further optimizations in parallel computing frameworks, including distributed deep learning models, cloud-based parallel processing, and hybrid CPU-GPU architectures. Additionally, integrating explainable AI techniques with segmentation algorithms can improve interpretability and reliability in critical applications such as medical imaging.

In conclusion, parallel image segmentation methods offer a powerful approach for processing vast amounts of image data with improved speed, accuracy, and scalability. By continuing to advance parallel ML and metaheuristic techniques, researchers can further enhance image segmentation solutions across various domains, including healthcare, remote sensing, and real-time surveillance.

Image segmentation is a fundamental process in medical image analysis, enabling the delineation of anatomical structures such as tumors, organs, and tissues. Traditional segmentation methodslike thresholding, edge detection, and region growingoften struggle with complex structures, noise, and high-resolution data, limiting their effectiveness in clinical applications.

To enhance segmentation accuracy and adaptability, machine learning (ML) techniques have been widely adopted. Algorithms such as Support Vector Machines (SVM), Random Forests (RF), K-Means, and Fuzzy C-Means (FCM) have been applied successfully. However, their performance heavily depends on optimal parameter settings and relevant feature selection.

To address these optimization challenges, metaheuristic algorithmssuch as Grey Wolf Optimizer (GWO), Whale Optimization Algorithm (WOA), and Genetic Algorithms (GA)offer effective global search strategies. They have proven useful for enhancing segmentation accuracy, selecting key features, and optimizing ML model parameters.

Nevertheless, metaheuristics are computationally intensive. This has motivated the use of parallel computing techniques, including CPU multiprocessing and GPU acceleration, to expedite metaheuristic-based image processing. Parallel metaheuristics enable faster convergence and scalability, making them well-suited for large-scale and real-time medical imaging tasks.

This thesis was driven by the need to develop scalable and accurate image processing systems capable of handling large medical datasets. Specifically, the thesis addresses the limitations of sequential metaheuristic algorithms in feature selection and image segmentation, proposing parallel and hybrid solutions that integrate metaheuristics, machine learning, and high-performance computing.The research explores the intersection of these domains to tackle two primary challenges:

- The selection of relevant features for accurate classification of medical conditions such as breast cancer,
- the segmentation of complex medical images, particularly MRI scans, with high precision and computational efficiency.

The first contribution of this work lies in the development of two robust metaheuristic-based feature selection approaches for breast cancer classification. The first method combines Grey Wolf Optimizer (GWO) with Random Forest (RF), achieving high classification accuracy (up to 98.6% on the WDBC dataset). The second method integrates Correlation-based Feature Selection (CFS) with GWO, followed by classification using RF, Support Vector Machine (SVM), and Naïve Bayes (NB). This hybrid strategy, termed CMGWO-RF, further improves performance, reaching 99.12% accuracy, and proves effective in reducing dimensionality while enhancing model interpretability.

The second major contribution concerns the design and implementation of parallel image segmentation frameworks tailored for medical imaging. Twostrategies were proposed:

1. Parallel WOA-KMeans using multiprocessing, which leverages CPU cores to accelerate the segmentation process while improving clustering quality. The approach demonstrated superior performance on grayscale test images and medical datasets,

with reduced execution times (e.g., from 186s to 16s) and better segmentation metrics such as PSNR, SSIM, and accuracy.

2. P-GWO-FCM for MRI segmentation, which integrates GWO with Fuzzy C-Means (FCM) optimized by Fuzzy Entropy, and implements the entire pipeline on GPU using CUDA. This method significantly improves segmentation quality, achieving a Jaccard Similarity (JS) score of 0.92, and Dice coefficients of 0.93, 0.89, and 0.95 for WM, GM, and CSF tissues, respectively. Compared to traditional FCM, sequential GWO-FCM, FCM-GENIUS, and Deep-JCR, the P-GWO-FCM method outperformed all in both accuracy and execution time (from 250s down to 0.89s on large images).

Together, these contributions demonstrate the potential of combining metaheuristics, machine learning, and parallelism to develop scalable and intelligent image analysis systems, particularly in domains requiring high precision such as medical diagnostics. The results confirm that parallel and hybrid metaheuristic strategies not only improve the effectiveness of image segmentation and classification, but also address the computational bottlenecks associated with large-scale image processing.

Building upon the results of this thesis, several future research directions can be pursued:

- Extending the proposed approaches to multi-modal and 3D medical images, integrating modalities like PET, and CT
- Developing hybrid deep learning-metaheuristic systems, combining CNNs or Vision Transformers with GWO, PSO, or WOA.
- Implementing distributed computing and cloud deployment of segmentation pipelines for real-time analysis in clinical settings.
- Applying multi-objective optimization frameworks to balance segmentation accuracy, processing time, and memory usage.

- Ali Mezaghrani, Mohammed Debakla, Khalifa Djemal, (2024). Nove1 feature selection method for accurate breast cancer classification using Correlation coefficient and Modified GWO Algorithm, Computer science journal of moldova, v o1.32, no .2(9 5), 2024,

- A. Mezaghrani, M. Debakla and K. Djemal, "Parallel Whale Optimization Algorithm-Kmeans for Image Segmentation using Multiprocessing,*" 2025 International Symposium on iNnovative Informatics of Biskra (ISNIB)*, Biskra, Algeria, 2025, pp. 1-6, doi: 10.1109/ISNIB64820.2025.10983906.

- Ali Mezaghrani, Mohamed Debakla, and Khalifa Djemal « Robust Method for Breast cancer classification Based on Feature selection using RGWO Algorithm » First International conference, on Aftificiallntelligence: Theories and Applications ICÆTA 2022 Mascara, Algeria, November 7-8' 7022;

- Ali Mezaghrani, Mohamed Debakla, and Khalifa, « Feature selection using correlation method for breast cancer classification», First National Conference on Science and Technology, June 2022, Mascara, Algeria.

Bibliography

[1] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. « U-net : Convolutional networks for biomedical image segmentation.» International Conference on Medical image computing and computer-assisted intervention. Springer, Cham, 2015.

[2] Cortes, C., and Vapnik, V. «Support-vector networks. Machine Learning », 20 (3), 273–297,1995. DOI:10.1007/BF00994018.

[3] MacQueen, J. Some methods for classification and analysis of multivariate observations. Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, 1, 281–297,1967.

[4] Bezdek, J. C. Pattern Recognition with Fuzzy Objective Function Algorithms. Springer,1981.

[5] G. L. Sayed, A. E. Hassanien and A. T. Azar, Feature selection via a novel chaotic crow search algorithm, Neural Comput. Appl. (31) 171–188. 2019.

[6]. H. Rao, X. Shi, A. Rodrigue, J. Feng and Y. Xia, Feature selection based on artificial bee colony and gradient boosting decision tree, Appl. Soft Comput, (74) 634–642 . 2019.

[7]. M. Abdel-Basset, D. El-Shahat, I. El-henawy and S. Mirjalili, A new fusion of grey wolf optimizer algorithm with a two-phase mutation for feature selection, Exp. Syst. Appl, (139) 112824 . 2020.

[8]. S. Kumar and M. Singh, «Breast Cancer Detection Based on Feature Selection Using Enhanced Grey Wolf Optimizer and Support Vector Machine Algorithms»,Vietnam Journal ofComputer Science, vol. 8, no. 2, pp. 177–197, 2021, DOI: 10.1142/S219688882150007X.

[9] M. Darzi, A. AsgharLiaei, M. Hosseini, and H. Asghari, «Feature selection for breast cancer diagnosis: A case-based wrapper approach »,World Academy of Science, Engineering and Technology,International Journal of Medical, Health, Biomedical, Bioengineeringand Pharmaceutical Engineering, vol. 5, pp. 220–223, 2011. https://api.semanticscholar.org/CorpusID:51751976.

[10] A.E. Rahmani, M. Katouli, «Breast cancer detection improvement by grasshopper optimization algorithm and classification SVM», Revue d'Intelligence Artificielle, vol.34, no.2, pp. 195–202, 2020, DOI: https://doi.org/10.18280/ria.340210.

[11] Mirjalili S, Mirjalili SM, Lewis A,«Grey wolf optimizer». Adv Eng Softw 69:46–61, 2014.

[12] Mirjalili S,«The ant lion optimizer». Adv Eng Softw 83:80–98, 2015.

[13] wHolland J (1991) «Adaptation in natural and artificial systems», 1975

[14] Mirjalili S, Lewis A,«The whale optimization algorithm». Adv Eng Softw 95:51–67, 2016.

[15] Bao X, Jia H, Lang C, «A novel hybrid Harris Hawks optimization for color image multilevel thresholding segmentation ». IEEE Access 7:76529–76546. 2019. https://doi.org/10.1109/ACCESS.2019.2921545.

[16] X.-p. Wang, J.-S. Pan and S.-C. Chu, «A parallel multi-verse optimizer for application in multilevel image segmentation », Ieee Access, pp. 32018-32030, 2020.

[17] Xiao Sui, Shu-Chuan Chu, Jeng-Shyang Pan and HaoLuo. «Parallel Compact Differential Evolution for Optimization Applied to Image Segmentation». Appl. Sci. 10, 2195, 2020, doi:10.3390/app10062195.

[18] Ali Mezaghrani, Mohamed Debakla, and Khalifa Djemal « Robust Method for Breast cancer classification Based on Feature selection using RGWO Algorithm » First International conference, on Aftificial lntelligence: Theories and Applications ICÆTA 2022 Mascara, Algeria, November 7-8' 7022.

[19] Ali Mezaghrani, Mohammed Debakla, Khalifa Djemal. « Nove1 feature selection method for accurate breast cancer classification using Correlation coefficient and Modified GWO Algorithm ». Computer science journal of moldova, v o1.32, no .2(9 5), 2024.

[20] Gonzalez, R. C., & Woods, R. E. « Digital Image Processing (2nd ed.) ». Prentice Hall, 2002.

[21] Moustakidesa,G.,Briassoulisb,D.,E. Psarakisa,E.,Dimasb, «3D image acquisition and NURBS based geometry modelling of natural objects », Advances in Engineering Software, 955–969,2000.

[22] Haralick, R. M. & Shapiro, L. G. «Image Segmentation Techniques». Computer Vision, Graphics, and Image Processing. 1985.

[23] Canny, J. «A Computational Approach to Edge Detection ». IEEE Transactions on Pattern Analysis and Machine Intelligence ,1986.

[24] Marr, D., and Hildreth, E. «Theory of edge detection. Proceedings of the Royal Society of London». Series B. Biological Sciences, 207(1167), 187–217.1980.

[25] D. Sangeetha, and P. Deepa, «FPGA implementation of cost-effective robust Canny edge detection algorithm », Journal of Real-Time Image Processing, vol.16, no.4, pp.957-970, 2019.

[26] Sezgin, M., and Sankur, B. «Survey over Image Thresholding Techniques and Quantitative Performance Evaluation ». Journal of Electronic Imaging, 2004.

[27] Bradley, D., and Roth, G. « Adaptive thresholding using the integral image » . Journal of Graphics, GPU, and Game Tools, 12(2), 13–21, 2007.

[28] Otsu, N.« A Threshold Selection Method from Gray-Level Histograms » . IEEE Transactions on Systems, Man, and Cybernetics, 1979.

[29] Gurjeet Kaur, Kuldeep Singh, « A comparative study of image segmentation using thresholding techniques», Conference Paper , March 2013, Gurgaon College of Engineering, Gurgaon, 2013.

[30] Adams, R., and Bischof, L. « Seeded region growing ». IEEE Transactions on Pattern Analysis and Machine Intelligence, 16(6), 641–647, 1994.

[31] Vincent, L., and Soille, P. « Watersheds in digital spaces: An efficient algorithm based on immersion simulations ». IEEE Transactions on Pattern Analysis and Machine Intelligence, 13(6), 583–598, 1991.

[32] Horowitz, S. L., and Pavlidis, T. « Picture segmentation by a tree traversal algorithm ». Journal of the ACM (JACM), 23(2), 368–388, 1976.

[33] Geman, S., & Geman, D. « Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images » . IEEE Transactions on Pattern Analysis and Machine Intelligence, 6(6), 721–741, 1984.

[34] B. R. Lee, «An image segmentation approach for fruit defect detection using k-means clustering and graph-based algorithm », Vietnam Journal of Computer Science, vol. 2, no. 1, pp. 25-33, 2015.

[35] Comaniciu, D., and Meer, P. « Mean shift: A robust approach toward feature space analysis ». IEEE Transactions on Pattern Analysis and Machine Intelligence, 24(5), 603–619, 2002.

[36] He, K., Gkioxari, G., Dollár, P., andGirshick, R. « Mask R-CNN ». Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2961–2969, 2017.

[37] Wang, J., Sun, K., Cheng, T., et al.« Deep high-resolution representation learning for visual recognition ». IEEE Transactions on Pattern Analysis and Machine Intelligence, 43(10), 3349–3364, 2020.

[38] J. Long, E. Shelhamer, and T. Darrell, «Fully convolutional networksfor semantic. Segmentation »,in IEEE Conference on Computer Visionand Pattern Recognition, pp. 3431-3440, 2015.

[39] Goodfellow I , Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y, «Generative adversarial nets». In: Advances in neural information processing systems,2014.

[40] Hamed Mohammadi Azni, Mohsen Afsharchi, Armin Allahverdi, «Improving brain tumor segmentation performance using CycleGAN based feature extraction », Multimedia Tools and Applications, 82:18039–18058, 2023 ,https://doi.org/10.1007/s11042-022-14174-3.

[41] Merity S, « Single headed attention rnn: Stop thinking with your head». 2019, arXiv preprint.arXiv: 1911.11423

[42] Ning J, Jiaxian W, Xiang M, Ke Y, Yuchang M, « Multitask learning model based on multi-scale cnn and lstm for sentiment classification ». IEEE Access 8:77060–77072, 2020.

[43] Chong Z, Pin L, Kai Qin A, Kay Chen T, « Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics ». IEEE Trans Neural Networks Learn Syst 28(10):2306–2318, 2016.

[44] X. Zhang, J. Cui, W. Wang, and C. Lin, «A study for texture feature extraction of high-resolution satellite images based on a direction measure and gray level co-occurrence matrix fusion algorithm »,Sensors, vol. 17, no. 7, p. 1474, 2017.

[45] T. Ojala, M. Pietikäinen, and T. Maenpaa, «Multiresolution gray-scale and rotation invariant texture classification with local binary patterns »,IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 971–987, 2002.

[46] Dalal, N., & Triggs, B. « Histograms of Oriented Gradients for Human Detection ». IEEE CVPR, 2005.

[47] Herv ´e Abdi1∗ and Lynne J. Williams2, « Principal component analysis », Volume 2, July/August 2010.DOI: 10.1002/wics.101

[48] Sara Ibrahim, Saima Nazir and Sergio A. Velastin, «Feature Selection Using Correlation Analysis and Principal Component Analysis for Accurate Breast Cancer Diagnosis, »J.Imaging, vol. 7, no. 11, Article No. 225, 2021. [Online]. Available: https://doi.org/10.3390/jimaging7110225.

[49] D. Jaswal, S. V, and K. P. Soman, «Image Classification Using Convolutional Neural Networks,» Int. J. Sci. Eng. Res., vol. 5, no. 6, pp. 1661–1668, 2014, doi: 10.14299/ijser.2014.06.002.

[50] Tarik Saidani. Multi-level Optimization of an Image Processing Application on Parallel Machines. Other [cond-mat.other]. University of Paris Sud - Paris XI, 2012. French. NNT: 2012PA112268. tel-00776111

[51] Michael J. Flynn. Some computer organizations and their effectiveness. IEEE Transactions on Computers, 21(9) :948–960, September 1972.

[52] J.L. Hennessy and D.A. Patterson. Computer Architecture : A Quantitative Approach. The Morgan Kaufmann Series in Computer Architecture and Design. Elsevier Science, 2011.

[53] Michael J. Flynn and Kevin W. Rudd. Parallel architectures. ACM Comput. Surv., 28(1) :67–70, March 1996.

[54] H.T. Kung, C.E. Leiserson, and Carnegie-Mellon University. Dept. of Computer Science. Systolic Arrays for (VLSI). CMU-CS. Carnegie-Mellon University, Department of Computer Science, 1978.

[55] IEEE. IEEE Standard . 1003.1c-1995 thread extensions. Technical report, IEEE, 1995.

[56] L. Dagum and R. Menon. OpenMP : an industry standard API for shared-memory programming. IEEE Computational Science and Engineering, 5(1) :46–55, 1998.

[57] V. S. Sunderam. PVM: a framework for parallel distributed computing. Concurrency : Pract. Exper.,2(4) :315–339, November 1990.

[58] Message Passing Forum. MPI : A Message-Passing Interface Standard. Technical report, Message Passing Interface Forum, Knoxville, TN, USA, 1994.

[59] CORPORATE Rice University. High performance fortran language specification. SIGPLAN Fortran Forum, 12(4) :1–86, December 1993.

[60] Ken Kennedy, Charles Koelbel, and Hans Zima. The rise and fall of high performance fortran : an historical object lesson. In Proceedings of the third ACM SIGPLAN conference on History of programming languages, HOPL III, pages 7–1–7–22, New York, NY, USA, 2007. ACM.

[61] Ujval Kapasi, William J. Dally, Scott Rixner, John D. Owens, and Brucek Khailany. The Imagine stream processor. In Proceedings 2002 IEEE International Conference on Computer Design, pages 282–288, September 2002.

[62] W. J. Dally, P. Hanrahan, M. Erez, T. J. Knight, F. Labonte, J-H A., N. Jayasena, U. J. Kapasi, A. Das, J. Gummaraju, and I. Buck. Merrimac : Supercomputing with streams. In SC'03, Phoenix, Arizona, November 2003.

[63] William Thies, Michal Karczmarek, and Saman Amarasinghe. Streamit : A language for streaming applications. In International Conference on Compiler Construction, Grenoble, France, Apr 2002.

[64] Ian Buck, Tim Foley, Daniel Horn, Jeremy Sugerman, Kayvon Fatahalian, Mike Houston, and Pat Hanrahan. Brook for gpus : stream computing on graphics hardware. ACM Trans. Graph., 23(3) :777–786, August 2004.

[65] John Nickolls, Ian Buck, Michael Garland, and Kevin Skadron. Scalable parallel programming with cuda. Queue, 6 :40–53, March 2008.

[66] Khronos OpenCLWorking Group. The OpenCL Specification, version 1.0.29, 8 December 2008.

[67] M Vijayaraj, R.Malar Vizhi, P.Chandrakala, Laith H. Alzubaidi, Khasanov Muzaffar, R Senthilkumar,"Parallel and Distributed Computing for High- Performance Applications",E3S Web of Conferences 399, 04039 , 2023, https://doi.org/10.1051/e3sconf/202339904039.

[68]Alismaili, S. Z., Li, M., Shen, J., Huang, P., He, Q., and Zhan, W. Organisational-level assessment of cloud computing adoption: Evidence from the Australian SMEs. Journal of Global Information Management, 28(2), 73–89. 2020, doi:10.4018/JGIM.2020040104

[69] Mell, P., and Grance, T. Perspectives on cloud computing and standards. National Institute of Standards and Technology (NIST). Information Technology Laboratory,2009.

[70] Rimal, B. P., Jukan, A., Katsaros, D., and Goeleven, Y. Architectural requirements for cloud computing systems: An enterprise cloud approach. Journal of Grid Computing, 9(1), 3–26,2011. doi:10.1007/s10723-010-9171-y

[71] Collange, S., Dandass, Y.S., Daumas, M. and Defour, D. 'Using graphics processors for parallelizing hash-based data carving', in HICCS, Hawaii International Conference on System Sciences, pp.1–10. 2009.

[72]Kalaiselvi, T., Sriramakrishnan, P., and Somasundaram, K. Survey of using GPU CUDA programming model in medical image analysis. Informatics in Medicine Unlocked, 9, 133–144, 2017. doi:10.1016/j.imu.2017.08.001

[73] Paz-Gallardo, Abel and Plaza, Antonio. A New Morphological Anomaly Detection Algorithm for Hyperspectral Images and its GPU Implementation. Proceedings of SPIE - The International Society for Optical Engineering. 8157. 10.1117/12.892282. 2011.

[74] Glaskowsky, P.N. (2009) NVIDIA's Fermi: The First CompleteGPU Computing Architecture [online] http://www.nvidia.com/content/pdf/fermi_white_papers/p.glaskowsky_nvidia's_fermithe_first_complete_gpu_architecture.pdf

[75] ImaneZouaneb, M.Belarbi, A.Chouarfia. “Multi approach for realtimesystems specification: case study of GPU parallel systems,” in International Journal of Big Data Intelligence IJBDI, vol.3,pp.122-141. 2016

[76] Memeti, S., Li, L., Pllana, S., Kołodziej, J., & Kessler, C.Benchmarking OpenCL, OpenACC, OpenMP, and CUDA. Proceedings of the 2017 Workshop on Adaptive Resource Management and Scheduling for Cloud Computing - ARMS-CC ’17. 2017. doi:10.1145/3110355.3110356

[77] Liu, W., Schmidt, B., Voss, G., & Müller-Wittig, W.Accelerating molecular dynamics simulations using Graphics Processing Units with CUDA. Computer Physics Communications, 179(9), 634–641. 2008. doi:10.1016/j.cpc.2008.05.008

[78] Scholl, I.; Aach, T.; Deserno, T.M.; Kuhlen, T. Challenges of Medical Image Processing. Comput. Sci. Res. Dev, 26, 5–13. 2011.

[79] Sancho, J.; Sutradhar, P.; Rosa, G.; Chavarrías, M.; Perez-Nunez, A.; Salvador, R.; Lagares, A.; Juárez, E.; Sanz, C. GoRG: Towards a GPU-Accelerated Multiview Hyperspectral Depth Estimation Tool for Medical Applications. Sensors 2021, 21, 4091.

[80] Graca, C.; Falcao, G.; Figueiredo, I.N.; Kumar, S. Hybrid Multi-GPU Computing: Accelerated Kernels for Segmentation and Object Detection with Medical Image Processing Applications. J. Real-Time Image Process. 2017, 13, 227–244.

[81] De, A.; Zhang, Y.; Guo, C. A Parallel Adaptive Segmentation Method Based on SOM and GPU with Application to MRI Image Processing. Neurocomputing 2016, 198, 180–189.

[82] Celik, B.; Gul, S.; Celik, M. A Stochastic Programming Approach to Surgery Scheduling under Parallel Processing Principle. Omega 2023, 115, 102799.

[83] Kalaiselvi, T.; Sriramakrishnan, P.; Somasundaram, K. Survey of Using GPU CUDA Programming Model in Medical Image Analysis. Inform. Med. Unlocked 2017, 9, 133–144.

[84] Bin Yu, Quan Zhou, Li Yuan, Huageng Liang, Pavel Shcherbakov, and Xuming Zhang, “3D medical image segmentation using the serial–parallel convolutional neural network and transformer based on crosswindowselfattention”, CAAI Transactions on Intelligence Technology, 2024, DOI: 10.1049/cit2.12411

[85] Aniket Pramanik, Xiaodong Wu, and Mathews Jacob, “Joint Calibrationless Reconstruction and Segmentation of Parallel MRI” ,ieee transactions on medical imaging, vol. Xx, no. Xx, xxxx 2020.

[86] Dongsheng Wang, TiezhenXv, Jiehui Liu, Jianshen Li, Lijie Yang and Jinxi Guo, “Bidirectional Efficient Attention Parallel Network for Segmentation of 3D Medical Imaging”, Electronics 2024, 13, 3086. https://doi.org/10.3390/electronics13153086

[87] Fatma Chahbar, Medjeded Merati, and Said Mahmoudi,”MPB-UNet: Multi-Parallel Blocks UNet for MRI Automated Brain Tumor Segmentation”, Electronics 2025, 14, 40, https://doi.org/10.3390/electronics14010040

[88] Sanjay Saxena , Suraj Shama, “Brain Tumor Segmentation by Texture Feature Extraction with the Parallel Implementation of Fuzzy C-Means using CUDA on GPU”, 5th IEEE International Conference on Parallel, Distributed and Grid Computing(PDGC-2018), 20-22 Dec, 2018, Solan, India

[89] Noureddine Ait Ali, BouchaibCherradi, Ahmed El Abbassi, Omar Bouattane, and Mohamed Youssfi, “Parallel Implementation and Performance Evaluation of some Supervised Clustering Algorithms for MRI Images Segmentation”, Association for Computing Machinery(2019), BDIoT'19, October 23–24, 2019, Rabat, Morocco, 2019. https://doi.org/10.1145/3372938.3373007.

[90] Shadi AlZuÕbi, Mohammed Shehab, Mahmoud Al-Ayyoub, Yaser Jararweh, Brij Gupta, Parallel Implementation for 3D Medical Volume Fuzzy Segmentation, Pattern RecognitionLetters, 2018, doi: 10.1016/j.patrec.2018.07.026

[91] PrajoonaValsalan, P. Sriramakrishnan, S. Sridhar, G. Charlyn Pushpa Latha, A. Priya, S. Ramkumar, A. Robert Singh, and T. Rajendran, "Knowledge based fuzzy c‐means method for rapid brain tissues segmentation of magnetic resonance imaging scans with CUDA enabled GPU machine", Journal of Ambient Intelligence and Humanized Computing, May 2020, https://doi.org/10.1007/s12652-020-02132-6.

[92] Rakhimov, M., Mamadjanov, D., and Mukhiddinov, A. A High-Performance Parallel Approach to Image Processing in Distributed Computing. 2020 IEEE 14th International Conference on Application of Information and Communication Technologies (AICT), 2020. doi:10.1109/aict50176.2020.936884

[93]. Benchara, F. Z., and Youssfi, M. A new scalable distributed k-means algorithm based on Cloudmicro-services for High-performance computing. Parallel Computing, 101, 102736, 2021.

[94]. Enfedaque, P., Aulı-Llinas, F., and Moure, J. C.GPU implementation of bitplane coding with parallel coefficient processing for high performance image compression. IEEE Transactions onParallel and Distributed Systems, 28(8), 2272-2284. 2017.

[95] Elsayed Badr, Sultan Almotairi , Mustafa Abdul Salam , Hagar Ahmed, New Sequential and Parallel Support Vector Machine with Grey Wolf Optimizer for Breast Cancer Diagnosis, Alexandria Engineering Journal (2022) 61, 2520–2534.

[96] Sawsan M. Mahmoud, Rokaia Shalal Habeeb, "Analysis of Large Set of Images Using MapReduce Framework", I.J. Modern Education and Computer Science, 2019, 12, 47-52 Published Online December 2019 in MECS (http://www.mecs-press.org/) DOI: 10.5815/ijmecs.2019.12.05

[97]. Tan, X.; Di, L.; Zhong, Y.; Yao, Y.; Sun, Z.; Ali, Y. Spark-based adaptive Mapreduce data processing method for remote sensing imagery. Int. J. Remote Sens. 2021, 42, 191–207.

[98] S. Almufti, R. Asaad and B. Salim, "Review on Elephant Herding Optimization Algorithm Performance in Solving Optimization Problems", Sci-encepubco.com, 2019. [Online]. Available: https://www.sciencepubco.com/index.php/ijet/article/view/28473. [Accessed: 26- May- 2019].

[99] S. Almufti, "U-Turning Ant Colony Algorithm powered by Great Deluge Algorithm for the solution of TSP Problem", Hdl.handle.net, 2018.

[100] S. Almufti, "Using Swarm Intelligence for solving NPHard Problems," Academic Journal of Nawroz University, vol. 6, no. 3, pp. 46–50, 2017. https://doi.org/10.25007/ajnu.v6n3a78.

[101] S. Almufti and A. Shaban, "U-Turning Ant Colony Algorithm for Solving Symmetric Traveling Salesman Problem", Academic Journal of Nawroz University, vol. 7, no. 4, pp. 45-49, 2018. https://doi.org/10.25007/ajnu.v6n4a270.

[102] Fister, I., Yang, X.-S., Fister, I., Brest, J., and Fister, D. (2013a). A Brief Review of Nature-Inspired Algorithms for Optimization. . (2013a). http://arxiv.org/abs/1307.4186

[103] Almufti, S., Marqas, R., & Asaad, R. Comparative study between elephant herding optimization (EHO) and U-turning ant colony optimization (U-TACO) in solving symmetric traveling salesman problem (STSP). Journal Of Advanced Computer Science and Technology, 8(2), 32.2019.

[104] Rai, D., & Tyagi, K.Bio-inspired optimization techniques. ACM SIGSOFT Software Engineering Notes, 38(4), 1–7.2013.https://doi.org/10.1145/2492248.2492271

[105] Almufti, S.Using Swarm Intelligence for solving NPHard Problems. Academic Journal of Nawroz University, 6(3), 46–50. 2017.https://doi.org/10.25007/ajnu.v6n3a78.

[106] Ihsan, R. R., Almufti, S. M., Ormani, B. M., Asaad, R. R., and Marqas, R. B. (2021). A survey on Cat Swarm Optimization algorithm. Asian J. Res. Comput. Sci, 10, 22-32. 2021.

[107] Sadeeq, H. T., Abdulazeez, A. M., Kako, N. A., Zebari, D. A., and Zeebaree, D. Q. A New Hybrid Method for Global Optimization Based on the Bird Mating Optimizer and the Differential Evolution. 2021 7th

International Engineering Conference “Research and Innovation amid Global Pandemic" (IEC), 54–60. 2021.https://doi.org/10.1109/IEC52205.2021.9476147

[108] Agrawal, P., Abutarboush, H. F., Ganesh, T., andMohamed, A. W. Metaheuristic Algorithms on Feature Selection: A Survey of One Decade of Research (2009-2019). IEEE Access, 9, 26766–26791.2021. https://doi.org/10.1109/ACCESS.2021.3056407.

[109] Storn R, Price K, Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11(4):341–359. 1997.

[110] Yang XS A new metaheuristic bat-inspired algorithm. In: Nature Inspired Cooperative Strategies for Optimization. Springer, (2010), p 65–74

[111] Yang XS, Deb S Cuckoo search via levy flights. In: 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), (2009),IEEE, pp 210–214

[112] Saremi S, Mirjalili S, Lewis A Grasshopper optimisation algorithm: theory and application. Adv Eng Softw , (2017),105:30–47

[113] Yang XS. Firefly algorithms for multimodal optimization. In: International symposium on stochasticalgorithms, Springer, pp 169–178.2009.

[114] Mirjalili S. Dragonfly algorithm: a new meta-heuristic optimization technique for solving singleobjective, discrete, and multi-objective problems. Neural Comput Appl 27(4):1053–1073. 2016.

[115] Yang XS. Flower pollination algorithm for global optimization. In: International Conference on Unconventional Computing and Natural Computation, Springer, pp 240–249.2012.

[116] Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671–680

[117] Shareef H, Ibrahim AA, Mutlag AH. Lightning search algorithm. Appl Soft Comput 36:315–333. 2015.

[118] Rashedi E, Nezamabadi-Pour H, Saryazdi S. Gsa: a gravitational search algorithm. Inf Sci 179(13):2232–2248.2009.

[119] Abedinpourshotorban H, Shamsuddin SM, Beheshti Z et al. Electromagnetic field optimization: A physics-inspired metaheuristic optimization algorithm. Swarm Evol Comput 26:8–22. 2016.

[120] Shi Y. Brain storm optimization algorithm. In: International Conference in Swarm Intelligence, Springer, pp 303–309. 2011.

[121] Rao RV, Savsani VJ, Vakharia D, Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput Aided Des 43(3):303–315.2011.

[122] Mohamed AW, Hadi AA, Mohamed AK, Gaining-sharing knowledge based algorithm for solving optimization problems: a novel nature-inspired algorithm. Int J Mach Learn Cybern 11(7):1501–1529.2020.

[123] Amir Seyyedabbasi, Wadhah Zeyad Tareq Tareq, Nebojsa Bacanin, An Efective Hybrid Metaheuristic Algorithm for Solving Global Optimization Algorithms, Multimedia Tools and Applications (2024) 83:85103–85138 https://doi.org/10.1007/s11042-024-19437-9

[124] Lee, R.S., Dunnmon, J.A., He, A., Tang, S., Re, C., Rubin, D.L.: Comparison of segmentation-free and segmentation-dependent computer-aided diagnosis of breast masses on a public mammography dataset. J. Biomed. Inform. 113, 103656 (2021).

[125] Jiang, D., Li, G., Tan, C., Huang, L., Sun, Y., Kong, J.: Semantic segmentation for multiscale target based on object recognition using the improved faster-rcnn model. Futur. Gener. Comput. Syst. 123, 94–104 .2021.

[126] Menglin, W., Cai, X., Chen, Q., Ji, Z., Niu, S., Leng, T., Rubin, D.L., Park, H.: Geographic atrophy segmentation in sd-oct images using synthesized fundus autofluorescence imaging. Comput. Methods Programs Biomed. 182, 105101 .2019.

[127] Jia, W., Tian, Y., Luo, R., Zhang, Z., Lian, J., Zheng, Y.: Detection and segmentation of overlapped fruits based on optimized mask r-cnn application in apple harvesting robot. Comput. Electron. Agric. 172, 105380 .2020.

[128] Mandal, D., Chatterjee, A., Maitra, M.: Robust medical image segmentation using particle swarm optimization aided level set based global fitting energy active contour approach. Eng. Appl. Artif. Intell. 35, 199–214 .2014.

[129] Singh, V.: Sunflower leaf diseases detection using image segmentation based on particle swarm optimization. Artif. Intell. Agric. 3, 62–68 .2019.

[130] Singh, S., Mittal, N., Thakur, D., Singh, H., Oliva, D.: Nature and biologically inspired image segmentation techniques. Arch. Comput. Methods Eng. 1, 28 .2021.

[131] Farshi, T.R., Drake, J.H., O¨ zcan, E.: A multimodal particle swarm optimization-based approach for image segmentation. Expert Syst. Appl. 149, 113233 .2020.

[132] Essam H. Houssein, Gaber M. Mohamed, Youcef Djenouri,Yaser M. Wazery, Ibrahim A. Ibrahim. “Nature inspired optimization algorithms for medical image segmentation: a comprehensive review”, Cluster Computing , 27:14745–14766 .2014. https://doi.org/10.1007/s10586-024-04601-5(0123456789().

[133] Oliva, D., Elaziz, M.A., Hinojosa, S.: Multilevel thresholding for image segmentation based on metaheuristic algorithms. In: Metaheuristic Algorithms for Image Segmentation: Theory and Applications, pages 59–69. Springer, 2019.

[134] Sathya, P.D., Kalyani, R., Sakthivel, V.P.: Color image segmentation using Kapur, Otsu and minimum cross entropy functions based on exchange market algorithm. Expert Syst.
Appl. 172, 114636 ,2021.

[135] Elaziz, M.A., Heidari, A.A., Fujita, H., Moayedi, H.: A competitive chain-based Harris hawks optimizer for global optimization and multi-level image thresholding problems. Appl. Soft Comput. 95, 106347 , 2020.

[136] Aziz, M.A.E., Ewees, A.A., Hassanien, A.E.: Whale optimization algorithm and moth-flame optimization for multilevel thresholding image segmentation. Expert Syst. Appl. 83, 242–256 , 2017.

[137] Houssein, E.H., El-dinHelmy, B., Oliva, D., Pradeep, J., Premkumar, M., Elngar, A.A., Shaban, H.: An efficient multithresholding based covid-19 ct images segmentation approach using an improved equilibrium optimizer. Biomed. Signal Process. Control 73, 103401, 2022.

[138] Houssein, E.H., Emam, M.M., Ali, A.A.: An efficient multilevel thresholding segmentation method for thermography breast cancer imaging based on improved chimp optimization algorithm. Expert Syst. Appl. 185, 115651, 2021.

[139] Gao, H., Zheng, F., Pun, C.-M., Haidong, H., Lan, R.: A multilevel thresholding image segmentation based on an improved artificial bee colony algorithm. Comput. Electr. Eng. 70, 931–938, 2018.

[140] Sri Madhava Raja, N., Arockia Sukanya, S., Nikita, Y.: Improved pso based multi-level thresholding for cancer infected breast thermal images using Otsu. Proced. Comput. Sci. 48, 524–529, 2015

[141] Mohamad Amin Bakhshali and Mousa Shamsi: Segmentation of color lip images by optimal thresholding using bacterial foraging optimization (bfo). J. Comput. Sci. 5(2), 251–257, 2014.

[142] Agrawal, S., Panda, R., Bhuyan, S., Panigrahi, B.K.: Tsallis entropy based optimal multilevel thresholding using cuckoo search algorithm. Swarm Evol. Comput. 11, 16–30, 2013.

[143] Manikandan, S., Ramar, K., Willjuice Iruthayarajan, M., Srinivasagan, K.G.: Multilevel thresholding for segmentation of medical brain images using real coded genetic algorithm. Measurement 47, 558–568, 2014.

[144] Hang, S., Zhao, D., Fanhua, Yu., Heidari, A.A., Zhang, Yu., Chen, H., Li, C., Pan, J., Quan, S.: Horizontal and vertical search artificial bee colony for image segmentation of covid-19 x-ray images. Comput. Biol. Med. 142, 105181, 2022.

[145] Aljanabi, M., O¨ zok, Y.E., Rahebi, J., Abdullah, A.S.: Skin lesion segmentation method for dermoscopy images using artificial bee colony algorithm. Symmetry 10(8), 347, 2018.

[146] Abdel-Basset, M., Chang, V., Mohamed, R.: Hsma_woa: a hybrid novel slime Mould algorithm with whale optimization algorithm for tackling the image segmentation problem of chest x-ray images. Appl. Soft Comput. 95, 106642, 2020.

[147] Nameirakpam Dhanachandra and Yambem Jina Chanu: An image segmentation approach based on fuzzy c-means and dynamic particle swarm optimization algorithm. Multimed. Tools Appl. 79(25), 18839–18858 , 2020.

[148] Bandyopadhyay, R., Kundu, R., Oliva, D., Sarkar, R.: Segmentation of brain mri using an altruistic Harris hawks' optimization algorithm. Knowl.-Based Syst. 232, 107468 , 2021.

[149] Dorgham, O.M., Mohammed Alweshah, M.H., Ryalat, J.A., Khader, M., Alkhalaileh, S.: Monarch butterfly optimization algorithm for computed tomography image segmentation. Multimed. Tools Appl. 80(20), 30057–30090, 2021.

[150] Qi, A., Zhao, D., Fanhua, Yu., Heidari, A.A., Zongda, W., Cai, Z., Alenezi, F., Mansour, R.F., Chen, H., Chen, M.: Directional mutation and crossover boosted ant colony optimization with application to covid-19 x-ray image segmentation. Comput. Biol. Med. 148, 105810, 2022.

[151] Ryalat, M.H., Dorgham, O., Tedmori, S., Al-Rahamneh, Z., Al- Najdawi, N., Mirjalili, S.: Harris hawks optimization for covid- 19 diagnosis based on multi-threshold image segmentation. Neural Comput. Appl. 35(9), 6855–6873, 2023.

[152] Dongmei, W., Yuan, C.: Threshold image segmentation based on improved sparrow search algorithm. Multimed. Tools Appl. 81(23), 33513–33546, 2022.

[153] Dhanachandra, N., Manglem, K., Chanu, Y.J.: Image segmentation using k-means clustering algorithm and subtractive clustering algorithm. Proced. Comput. Sci. 54, 764–771, 2015.

[154] Rastgarpour, M., Shanbehzadeh, J., Soltanian-Zadeh, H.: A hybrid method based on fuzzy clustering and local region-based level set for segmentation of inhomogeneous medical images. J. Med. Syst. 38(8), 1–15, 2014.

[155] Anitha Vishnuvarthanan, M., Rajasekaran, P., Govindaraj, V., Zhang, Y., Thiyagarajan, A.: An automated hybrid approach using clustering and nature inspired optimization technique for improved tumor and tissue segmentation in magnetic resonance brain images. Appl. Soft Comput. 57, 399–426, 2017.

[156] Zexian, F., An, J., Yang, Q., Yuan, H., Sun, Y., Ebrahimian, H.: Skin cancer detection using kernel fuzzy c-means and developed red fox optimization algorithm. Biomed. Signal Process. Control71, 103160 , 2022.

[157] Jain, S., Indora, S., Atal, D.K.: Lung nodule segmentation using Salp shuffled shepherd optimization algorithm-based generative adversarial network. Comput. Biol. Med. 137, 104811, 2021.

[158] Ajai, A.K., Anitha, A.: Clustering based lung lobe segmentation and optimization based lung cancer classification using ct images. Biomed. Signal Process. Control 78, 103986, 2022.

[159] Benaichouche, A.N., Oulhadj, H., Siarry, P.: Improved spatial fuzzy c-means clustering for image segmentation using pso initialization, Mahalanobis distance and post-segmentation correction. Digital Signal Process. 23(5), 1390–1400, 2013.

[160] Ramana Kumari, A., Rao, S.N., Ramana Reddy, P.: Design of hybrid dental caries segmentation and caries detection with meta-heuristic-based resnext-rnn. Biomed. Signal Process. Control 78, 103961, 2022.

[161] Nguyen, U.T.V., Bhuiyan, A., Park, L.A.F., Ramamohanarao, K.: An effective retinal blood vessel segmentation method using multi-scale line detection. Pattern Recogn. 46(3), 703–715, 2013.

[162] Nadipally, M.: Optimization of methods for image-texture segmentation using ant colony optimization. In: Intelligent data analysis for biomedical applications, pages 21–47. Elsevier, 2019.

[163] Mesejo, P., Valsecchi, A., Marrakchi-Kacem, L., Cagnoni, S., Damas, S.: Biomedical image segmentation using geometric deformable models and metaheuristics. Comput. Med. Imaging Graph. 43, 167–178 , 2015.

[164] Sivakumar, V., Janakiraman, N.: A novel method for segmenting brain tumor using modified watershed algorithm in mri image with fpga. Biosystems 198, 104226, 2020.

[165] Li Zhang and Chee Peng Lim: Intelligent optic disc segmentation using improved particle swarm optimization and evolving ensemble models. Appl. Soft Comput. 92, 106328, 2020.

[166] Escorcia-Gutierrez, J., Torrents-Barrena, J., Gamarra, M., Romero-Aroca, P., Valls, A., Puig, D.: A color fusion model based on Markowitz portfolio optimization for optic disc segmentation in retinal images. Expert Syst. Appl. 174, 114697, 2021.

[167] Aghazadeh, N., Moradi, P., Castellano, G., Noras, P.: An automatic mri brain image segmentation technique using edge-region- based level set. J. Supercomput. 79(7), 7337–7359, 2023.

[168] Liu, S., Peng, Y.: A local region-based Chan-Vese model for image segmentation. Pattern Recogn. 45(7), 2769–2779, 2012.

[169] Shi, C., Cheng, Y., Liu, F., Wang, Y., Bai, J., Tamura, S.: A hierarchical local region-based sparse shape composition for liver segmentation in ct scans. Pattern Recogn. 50, 88–106, 2016.

[170] Pham, T.X., Siarry, P., Oulhadj, H.: A multi-objective optimization approach for brain mri segmentation using fuzzy entropy clustering and region-based active contour methods. Magn. Reson. Imaging 61, 41–65 ,2019.

[171] Huang, Q., Bai, X., Li, Y., Jin, L., Li, X.: Optimized graph based segmentation for ultrasound images. Neurocomputing 129, 216–224 ,2014.

[172] marwa obayya , muhammad kashif saeed , nuha alruwais , saud s. alotaibi , mohammed assiri , and ahmed s. salama. Hybrid Metaheuristics with Deep Learning based Fusion Model for Biomedical Image Analysis. Digital Object Identifier 10.1109/ACCESS.2023.3326369, VOLUME 11, 2023.

[173] Malik, S.; Akram, T.; Ashraf, I.; Rafiullah, M.; Ullah, M.; Tanveer, J. A Hybrid Preprocessor DE-ABC for Efficient Skin-Lesion Segmentation with Improved Contrast. Diagnostics12, 2625, 2022. https://doi.org/ 10.3390/diagnostics12112625

[174] Baldeon-Calisto, Maria, and Susana K. Lai-Yuen. ”AdaResU-Net: Multiobjective adaptive convolutional neural network for medical image segmentation.” Neurocomputing 392: 325-340, 2020.

[175] Targ, Sasha, Diogo Almeida, and Kevin Lyman. ”Resnet in resnet: Generalizing residual architectures.” arXiv preprint arXiv:1603.08029 , 2016.

[176] Hassanzadeh, Tahereh, Daryl Essam, and Ruhul Sarker. ”EvoU-Net: An evolutionary deep fully convolutional neural network for medical image segmentation.” Proceedings of the 35th Annual ACM Symposium on Applied Computing. 2020.

[177] Dawid Połap I , Karolina K˛esik, Marcin Wo´zniak, and Robertas Damaševi˘cius, Parallel Technique for the Metaheuristic Algorithms Using Devoted Local Search and Manipulating the Solutions Space, Appl. Sci.8, 293, 2018. doi:10.3390/app8020293

[178] T. Dokeroglu and A. Cosar, “Optimization of one-dimensional bin packing problem with island parallel grouping genetic algorithms,” Computers & Industrial Engineering, vol. 75, pp. 176–186, 2014.

[179] E. Alba and M. Tomassini, “Parallelism and evolutionary algorithms,” IEEE Transactions on Evolutionary Computation, vol. 6, no. 5, pp. 443–462, 2002.

[180] H.-C. Huang, “Fpga-based parallel metaheuristic pso algorithm and its application to global path planning for autonomous robot navigation,” Journal of Intelligent and Robotic Systems, vol. 76, no. 3,pp. 475–488, 2014.

[181] S, . G¨ulc¨u and H. Kodaz, “A novel parallel multi-swarm algorithm based on comprehensive learning particle swarm optimization,” Engineering Applications of Artificial Intelligence, vol. 45, pp. 33–45,2015.

[182] Luong, T. V. Métaheuristique parallèles sur GPU. Ph.D. thesis Ecole doctorale sciences Pour l’ingenieur Universite Lille Nord-de-France, 2011.

[183] Talbi, E. G. Metaheuristics : from design to implementation. John Wiley and Sons, 624 pages, 2009.

[184] Che-Lun Hung, Yuan-Huai Wu. Parallel genetic-based algorithm on multiple embedded graphic processing units for brain magnetic resonance imaging segmentation 2016. Computers and Electrical Engineering, 2016, 1–11, http://dx.doi.org/10.1016/j.compeleceng.2016.09.028.

[185] Laila Almutairi, Ahed Abugabah, Hesham Alhumyani, Ahmed A. Mohamed. Intelligent biomedical image classification in a big data architecture using metaheuristic optimization and gradient approximation. Wireless Networks ,2024, 30:7087–7108, https://doi.org/10.1007/s11276-023-03573-5(0123456789().,-volV)(0123456789,-().volV)

[186] Khalid M. Hosny, Asmaa M. Khalid, Hanaa M. Hamza, SeyedaliMirjalili. Multilevel segmentation of 2D and volumetric medical images using hybrid Coronavirus Optimization Algorithm. Computers in Biology and Medicine ,2022,150, https://doi.org/10.1016/j.compbiomed.2022.106003 .

[187] Keith B. History of Machine Learning. Keith D. Foote blog on March 26, 2019. Available on https://www.dataversity.net/a-brief-history-of-machine-learning. Accessed on December 20, 2020 .

[188]. Russell S., Norvig P. Artificial Intelligence: A Modern Approach (2nd ed.). Prentice Hall. ISBN 978-0137903955, 2003.

[189] Langley, P., Simon, H., Bradshaw, G. and Zytkow, J. Scientific Discovery: Computational Explorations of the Creative Processes. MIT Press, Cambridge, 1987.

[190]. Lee, H., and Choi, B. Knowledge management enablers, processes, and organizational performance: An integrative view and empirical examination. Journal of Management Information Systems, 20(1), 179– 228. 2003.

[191] Sarker IH, Hoque MM, MdK Uddin, Tawfeeq A. Mobile data science and intelligent apps: concepts, ai-based modeling and research directions. Mob Netw Appl, pages 1–19, 2020.

[192] Sarker IH, Kayes ASM, Badsha S, Alqahtani H, Watters P, Ng A. Cybersecurity data science: an overview from machine learning perspective. J Big Data.7(1):1–29, 2020.

[193] Han J, Pei J, Kamber M. Data mining: concepts and techniques. Amsterdam: Elsevier; 2011.

[194] McCallum A. Information extraction: distilling structured data from unstructured text. Queue. 3(9):48–57. 2005.

[195] Moustafa N, Slay J. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). Military communications and information systems conference (MilCIS), 2015;pages 1–6. IEEE, 2015.

[196] Canadian institute of cybersecurity, university of new brunswick, iscx dataset, http:// www. unb. ca/ cic/ datas ets/ index. html/ (Accessed on 20 October 2019).

[197] Cic-ddos2019 [online]. available: https:// www. unb. ca/ cic/ datas ets/ ddos- 2019. html/ (Accessed on 28 March 2020).

[198] Koroniotis N, Moustafa N, Sitnikova E, Turnbull B. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: bot-iot dataset. Fut Gen Comput Syst.100:779–96, 2019.

[199] Santi P, Ram D, Rob C, Nathan E. Behavior-based adaptive call predictor. ACM Trans Auton Adapt Syst. 6(3):21:1–21:28, 2011.

[200] Sarker IH, Colman A, Kabir MA, Han J. Phone call log as a context source to modeling individual user behavior. In: Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (Ubicomp): Adjunct, Germany, pages 630–634. ACM, 2016.

[201] Eagle N, Pentland AS. Reality mining: sensing complex social systems. Person Ubiquit Comput. 10(4):255–68, 2006.

[202] Balducci F, Impedovo D, Pirlo G. Machine learning applications on agricultural datasets for smart farm enhancement. Machines. 6(3):38, 2018.

[203] Khadse V, Mahalle PN, Biraris SV. An empirical comparison of ,supervised machine learning algorithms for internet of things data. In: 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), IEEE. 1–6, 2018.

[204] Zikang H, Yong Y, Guofeng Y, Xinyu Z. Sentiment analysis of agricultural product ecommerce review data based on deep learning. In: 2020 International Conference on Internet of Things and Intelligent Applications (ITIA), IEEE, 2020; pages 1–7

[205] Safdar S, Zafar S, Zafar N, Khan NF. Machine learning based decision support systems (dss) for heart disease diagnosis: a review. Artif Intell Rev.50(4):597–623, 2018.

[206] Perveen S, Shahbaz M, Keshavjee K, Guergachi A. Metabolic syndrome and development of diabetes mellitus: predictive modeling based on machine learning techniques. IEEE Access. 7:1365–75, 2018.

[207] Harmon SA, Sanford TH, Sheng X, Turkbey EB, Roth H, Ziyue X, Yang D, Myronenko A, Anderson V, Amalou A, et al. Artificial intelligence for the detection of covid-19 pneumonia on chest ct using multinational datasets. Nat Commun. 11(1):1–7, 2020.

[208] ALI BOU NASSIF, ISMAIL SHAHIN, IMTINAN ATTILI, MOHAMMAD AZZEH, AND KHALED SHAALAN, Speech Recognition Using Deep Neural Networks: A Systematic Review, Article in IEEE Access · February 2019 DOI: 10.1109/ACCESS.2019.2896880

[209] Mohammed M, Khan MB, Bashier Mohammed BE. Machine learning: algorithms and applications. CRC Press; 2016.

[210] Kaelbling LP, Littman ML, Moore AW. Reinforcement learning: a survey. J Artif Intell Res. 1996;4:237–85.

[211] Y. Cho and L. K. Saul, ``Kernel methods for deep learning," in Proc. Adv. Neural Inf. Process. Syst. (NIPS), vol. 22, 2009, pp. 342_350.

[212] Tavallaee M, Bagheri E, Lu W, Ghorbani AA. A detailed analysis of the kdd cup 99 data set. In. IEEE symposium on computational intelligence for security and defense applications. IEEE.2009:1–6. 2009.

[213] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, et al. Scikit-learn: machine learning in python. J Mach Learn Res. 12:2825–30, 2011.

[214] Witten IH, Frank E. Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, 2005.

[215] Hsu, C. W., Chang, C. C., & Lin, C. J. A practical guide to support vector classification. Technical Report, Department of Computer Science, National Taiwan University. 2010.

[216] Burges, C. J. C. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, 2, 121–167. 1998. DOI:10.1023/A:1009715923555

[217] Quinlan, J. R. "Induction of Decision Trees." Introduces ID3, one of the earliest algorithms for decision tree construction. 1986.

[218] "Random Forests" by Leo Breiman. This seminal paper introduces the Random Forest algorithm, detailing its construction, advantages, and performance metrics. 2001.

[219] "WildWood: A New Random Forest Algorithm" by Gaïffas et al. « Introduces an enhanced Random Forest variant that improves prediction accuracy and computational efficiency » (2021) .

[220] Jehad Ali , Rehanullah Khan , Nasir Ahmad , Imran Maqsood, « Random Forests and Decision Trees », IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 5, No 3, September 2012 ISSN (Online): 1694-0814 www.IJCSI.org

[221] I.Rich , « An empirical study of the naive Bayes classifier », T.J. Watson Research Center, 2001.

[222] Indika Wickramasinghe ,and Harsha Kalutarage, « Naive Bayes: Applications, Variations and Vulnerabilities - A Review of Literature with Code Snippets for Implementation », Methodologies and Application, Volume 25, pages 2277–2293, 2021.

[223] Jalloul, R.; Chethan, H.K.; Alkhatib, R. A Review of Machine Learning Techniques for the Classification and Detection of Breast Cancer from Medical Images. Diagnostics 2023, 13, 2460. https:// doi.org/10.3390/diagnostics13142460

[224] Yongguang Zhai , Zhongyi Qu, and Lei Hao ,Land Cover Classification Using Integrated Spectral, Temporal, and Spatial Features Derived from Remotely Sensed Images, Remote Sens. 2018, 10(3), 383; https://doi.org/10.3390/rs10030383

[225] Kurdi, S.Z.; Ali, M.H.; Jaber, M.M.; Saba, T.; Rehman, A.; Damaševi˘cius, R. Brain Tumor Classification Using Meta-Heuristic Optimized Convolutional Neural Networks. J. Pers. Med. 2023, 13, 181. https://doi.org/10.3390/ jpm13020181

[226]. UCI. Breast Cancer Wisconsin Dataset. Available online: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+ (Diagnostic) (accessed on 9 June 2019).

[227]. Coccia, M. The increasing risk of mortality in breast cancer: A socioeconomic analysis between countries. J. Soc. Adm. Sci. 2019, 6 218–230.

[228] Jalloul, R. Chethan, H.K., Alkhatib, R. A Review of Machine Learning Techniques for the Classification and Detection of Breast Cancer from Medical Images. Diagnostics 2023, 13, 2460. https:// doi.org/10.3390/diagnostics13142460

[229] Jafari, Z.; Karami, E. Breast Cancer Detection in Mammography Images: A CNN-Based Approach with Feature Selection. Information 2023, 14, 410. https://doi.org/ 10.3390/info14070410.

[230] Breastcancer.org. (2023). Types of Breast Cancer. https://www.breastcancer.org

[231] National Cancer Institute. (2022). Inflammatory Breast Cancer. https://www.cancer.gov/types/breast/ibc

[232] Pisano, E. D., et al. Diagnostic performance of digital versus film mammography for breast-cancer screening. NEJM, 353(17), 1773–1783, 2005.

[233] Berg, W. A., et al. Combined screening with ultrasound and mammography vs mammography alone in women at elevated risk of breast cancer. JAMA, 299(18), 2151–2163, 2008.

[234] Mann, R. M., et al. Breast MRI: state of the art. Radiology, 277(3), 520–536, 2015.

[235] Rafferty, E. A., et al. Breast cancer screening using tomosynthesis in combination with digital mammography. JAMA, 309(22), 2343–2349, 2013.

[236] Groheux, D., et al. 18F-FDG PET/CT in staging, restaging, and treatment response assessment of breast cancer. Journal of Nuclear Medicine, 54(9), 1309–1321, 2013.

[237] A. B. Yusuf, R. M. Dima, and S. K. Aina, "Optimized Breast Cancer Classification using Feature Selection and Outliers Detection," Journal of the Nigerian Society of Physical Sciences, vol. 3, no. 4, pp. 298–307, 2021, DOI: 10.46481/jnsps.2021.331.

[238] Yunyun Liu, Azizan As'arry, Mohd Khair Hassan, Abdul Aziz Hairuddin, Hesham Mohamad, Review of the grey wolf optimization algorithm: variants and applications, Neural Computing and Applications (2024) 36:2713–2735, 2024. https://doi.org/10.1007/s00521-023-09202

[239] Gao ZM, Zhao J. An improved grey Wolf optimization algorithm with variable weights. Comput Intell Neurosci. 2019. https:// doi.org/10.1155/2019/2981282

[240] Hu P, Chen S, Huang H, Zhang G, Liu L. Improved alpha-guided grey wolf optimizer. IEEE Access 7:5421–5437. 2019. https://doi.org/10.1109/ACCESS.2018.2889816

[241] Luo K. Enhanced grey wolf optimizer with a model for dynamically estimating the location of the prey. Appl Soft Comput J 77:225–235. 2019.https://doi.org/10.1016/j.asoc.2019.01.025

[242] Gupta S, Deep K. A novel random walk grey wolf optimizer. Swarm Evol Comput 44:101–112. 2019.https://doi.org/10. 1016/j.swevo.2018.01.001

[243] Adhikary J, Acharyya S. randomized balanced grey wolf optimizer (RBGWO) for solving real life optimization problems. Appl Soft Comput. 2022. https://doi.org/10.1016/j.asoc.2022.108429

[244] Emary E, Zawbaa HM, Grosan C. Experienced gray wolf optimization through reinforcement learning and neural networks. IEEE Trans Neural Netw Learn Syst 29(3):681–694. 2018.https://doi.org/10.1109/TNNLS.2016.2634548

[245] Nadimi-Shahraki MH, Taghian S, Mirjalili S. An improved grey wolf optimizer for solving engineering problems. Expert Syst Appl. 2021. https://doi.org/10.1016/j.eswa.2020.113917

[246] Tripathi AK, Sharma K, Bala M. A novel clustering method using enhanced grey wolf optimizer and mapreduce. Big Data Res 14:93–100. 2018.https://doi.org/10.1016/j.bdr.2018.05.002

[247] Barman B, Dewang RK, Mewada A. Facial recognition using grey wolf optimization. Mater Today Proc 58:273–285. 2022. https://doi.org/10.1016/j.matpr.2022.02.161

[248] Martin B, Marot J, Bourennane S. Mixed grey wolf optimizer for the joint denoising and unmixing of multispectral images. Appl Soft Comput J 74:385–410. 2019.https://doi.org/10.1016/j.asoc.2018.10.019

[249] Yu X, Wu X. Ensemble grey wolf optimizer and its application for image segmentation. Expert Syst Appl. 2022. https:// doi.org/10.1016/j.eswa.2022.118267

[250] Gopatoti A, Vijayalakshmi P, CXGNet: a tri-phase chest X-ray image classification for COVID-19 diagnosis using deep CNN with enhanced grey-wolf optimizer. Biomed Signal Process Control. 2022. https://doi.org/10.1016/j.bspc.2022.103860

[251] Karakoyun M, Gu¨lcu¨ S¸ , Kodaz H (2021) D-MOSG: discrete multi-objective shuffled gray wolf optimizer for multi-level image thresholding. Eng Sci Technol Int J 24(6):1455–1466. 2021, https://doi.org/10.1016/j.jestch.2021.03.011

[252] Yu H et al, Image segmentation of leaf spot diseases on maize using multi-stage Cauchy-enabled grey wolf algorithm. Eng Appl Artif Intell. 2022, https://doi.org/10.1016/j.engappai.2021. 104653

[253] Ali M. Alsaqr, "Remarks on the use of Pearson's and Spearman's correlation coefficients in assessing relationships in ophthalmic data," African Vision and Eye Health, vol. 80, no. 1, Article No. 612, 2021, DOI: 10.4102/aveh.v80i1.612.

[254] H. Akoglu, "User's guide to correlation coefficients," Turkish Journal of Emergency Medicine, vol. 18, no. 3, pp. 91–93, 2018, DOI:https://doi.org/10.1016/j.tjem.2018.08.001.

[255] 14 . Martin, D. , Fowlkes, C. , Tal, D. , and Malik, J. (2001). A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In Computer vision, 2001. ICCV 2001. Proceedings. Eighth IEEE international conference on: Vol. 2 (pp. 416–423). IEEE .

[256] Labati RD, Piuri V, Scotti F. All-Idb: the acute lymphoblastic leukemia image database For image processing Ruggero Donida Labati IEEE Member, Vincenzo Piuri IEEE Fellow, Fabio Scotti IEEE Member Universit ' a degli Studi di Milano, Department of Information Technologies,. IEEE Int Conf Image Process 2045–2048. 2011.

[257]. Laiton-Bonadiez, C.; Sanchez-Torres, G.; Branch-Bedoya, J. Deep 3D Neural Network for Brain Structures Segmentation Using Self-Attention Modules in MRI Images. Sensors 2022, 22, 2559.

[258]. Liu, Y.; Unsal, H.S.; Tao, Y.; Zhang, N. Automatic Brain Extraction for Rodent MRI Images. Neuroinformatics 2020, 18, 395–406.

[259]. Kontos, D.; Megalooikonomou, V.; Gee, J.C. Morphometric Analysis of Brain Images with Reduced Number of Statistical Tests: A Study on the Gender-Related Differentiation of the Corpus Callosum. Artif. Intell. Med. 2009, 47, 75–86.

[260]. Munir, K.; Frezza, F.; Rizzi, A. Deep Learning Hybrid Techniques for Brain Tumor Segmentation. Sensors 2022, 22, 8201.

[261]. Widmann, G.; Henninger, B.; Kremser, C.; Jaschke, W. MRI Sequences in Head & Neck Radiology–State of the Art. Fortschr. Rontgenstr. 2017, 189, 413–422.

[262]. Dong, Q.;Welsh, R.C.; Chenevert, T.L.; Carlos, R.C.; Maly-Sundgren, P.; Gomez-Hassan, D.M.; Mukherji, S.K. Clinical Applications of Diffusion Tensor Imaging. Magn. Reson. Imaging 2004, 19, 6–18

[263]. Sun, L.; Zu, C.; Shao, W.; Guang, J.; Zhang, D.; Liu, M. Reliability-Based Robust Multi-Atlas Label Fusion for Brain MRI Segmentation. Artif. Intell. Med. 2019, 96, 12–24.

[264]. Richard, N.; Dojat, M.; Garbay, C. Automated Segmentation of Human Brain MR Images Using a Multi-Agent Approach. Artif.Intell. Med. 2004, 30, 153–176.

[265]. González-Villà, S.; Oliver, A.; Valverde, S.; Wang, L.; Zwiggelaar, R.; Lladó, X. A Review on Brain Structures Segmentation in Magnetic Resonance Imaging. Artif. Intell. Med. 2016, 73, 45–69.

[266]. Cao, R.; Ning, L.; Zhou, C.; Wei, P.; Ding, Y.; Tan, D.; Zheng, C. CFANet: Context Feature Fusion and Attention Mechanism Based Network for Small Target Segmentation in Medical Images. Sensors 2023, 23, 8739.

[267]. Anusooya, G.; Bharathiraja, S.; Mahdal, M.; Sathyarajasekaran, K.; Elangovan, M. Self-Supervised Wavelet-Based AttentionNetwork for Semantic Segmentation of MRI Brain Tumor. Sensors 2023, 23, 2719.

[268]. Lu, F.; Tang, C.; Liu, T.; Zhang, Z.; Li, L. Multi-Attention Segmentation Networks Combined with the Sobel Operator for Medical Images. Sensors 2023, 23, 2546.

[269] FU Hai-Jun, WU Xiao-Hong, MAO Han-Ping, WU Bin,"Fuzzy Entropy Clustering Using Possibilistic Approach", Procedia Engineering 15 (2011) 1993 – 1997, doi:10.1016/j.proeng.2011.08.372

[270] Li R P, Mukaidono M. A Maximum-Entropy Approach to Fuzzy Clustering. International Joint Conference of the Fourth IEEE International Conference on Fuzzy Systems and The Second International Fuzzy Engineering Symposium., Proceedings of 1995 IEEE International Conference on Fuzzy Systems, 1995, Vol 4, pp.2227-2232.

[271] C.A. Cocosco, V. Kollokian, R.K.-S. Kwan, A.C. Evans : "BrainWeb: Online Interface to a 3D MRI Simulated Brain Database"NeuroImage, vol.5, no.4, part 2/4, S425, 1997 -- Proceedings of 3-rd International Conference on Functional Mapping of the Human Brain, Copenhagen, May 1997

[272] Praveen Kumar Ramtekkar, Anjana Pandey, Mahesh Kumar Pawar, "Accurate detection of brain tumor using optimized feature selection based on deep learning techniques", Vol.:(0123456789) Multimedia Tools and Applications, April 2023, https://doi.org/10.1007/s11042-023-15239-7.

[273] Carr, C.; Kitamura, F.; Partridge, G.; Kalpathy-Cramer, J.; Mongan, J.; Andriole, K.; Lavender Vazirabad, M.; Riopel, M.; Ball, R.; Dane, S.; et al. RSNA Screening Mammography Breast Cancer Detection, Kaggle 2022. Available online: https: //kaggle.com/competitions/rsna-breast-cancer-detection (accessed on 1 December 2022).

[274] PrajoonaValsalan, P. Sriramakrishnan, S. Sridhar, G. Charlyn Pushpa Latha, A. Priya, S. Ramkumar, A. Robert Singh, and T. Rajendran, "Knowledge based fuzzy c‐means method for rapid brain tissues segmentation of magnetic resonance imaging scans with CUDA enabled GPU machine", Journal of Ambient Intelligence and Humanized Computing, May 2020, https://doi.org/10.1007/s12652-020-02132-6.

[275] Aniket Pramanik, Xiaodong Wu, and Mathews Jacob.(2023). Joint Calibrationless Reconstruction and Segmentation of Parallel MRI. ieee transactions on medical imaging. ECCV 2022 Workshops, LNCS 13803, pp. 437–453. https://doi.org/10.1007/978-3-031-25066-8_24.