*Republic of Iraq*

*Ministry of Higher Education and Scientific Research*

*University of Babylon*

*College of Science for Women*

*Department of Computer Sciences*

# Covid-19 Classification Using Machine Learning and Metaheuristics Optimization Algorithms

***A thesis***

*Submitted to the Council of College of Science for Women, the University of Babylon in a Partial Fulfilment of the Requirements for the Degree of Master in Science\ Computer Sciences*

***Submitted By***

**Asraa Mohsin Mohammad**

***Supervised By***

**Prof. Dr. Hussain Attia**

**Prof. Dr. Yossra Hussain Ali**

**2023 A. D.** **1445 A. H**

بسم الله الرحمن الرحيم


صَدَقَ اللهُ العَظِيمْ

# Supervisors Certification

*I certify that this thesis entitled "Covid-19 Classification Using Machine Learning and Metaheuristics Optimization Algorithms" Written by "Asraa Mohsin Mohammad" Was prepared under my supervision at College of Sciences for Women As a partial fulfillment of the requirements for the degree of a Master's in Computer Sciences.*

***Signature***:

***Name: Prof. Dr. Hussain Attia***

***Date:  /  / 2023***

***Address: University of Babylon/College of Sciences for Women***

***Signature***:

***Name: Prof. Dr. Yossra Hussain Ali***

***Date:  /  / 2023***

***Address: University of Technology, Baghdad / Computer sciences department***

# Head of the Department Certification

***In view of the available recommendations, I forward the thesis entitled "Covid-19 Classification Using Machine Learning and Metaheuristics Optimization Algorithms" for debate by the examining committee.***

***Signature:***

***Name: Dr. Saif M. Kh. Al-Alak***

***Date: / / 2023***

***Address: University of Babylon/College of Science for Women***

# Dedications

*To my great creator, dear God*

*To the first teacher of mankind, the prophet Mohammed, may God bless him and his family and grant them peace.*

*To my intercession with God in this world and the hereafter, the pure imams, peace be upon them.*

*To the example of dedication and devotion.... Beloved father, may he rest in peace.*

*To whom offered me happiness and comfort over her happiness ... My honorable mother.*

*To those who wish happiness and success for me from the bottom of their hearts without any compensation ... my dear brothers and sisters.*

*To those who supported me and encouraged me with all love and patience... my husband, my daughter, and my son.*

*To all my dear loyal friends*

*I offer you that humble work...*

*Asraa Mohsin… (2023)*

# Acknowledgements

*First and foremost, I express my gratitude to God, who bestowed upon me the patience and strength required to successfully complete this study. I firmly believe that nothing is achievable without His divine assistance.*

*I would also like to extend my sincere appreciation and thanks to my esteemed supervisors*

***Prof. Dr. Hussain Attia and Prof. Dr. Yossra Hussain Ali***

*For their unwavering support and invaluable guidance throughout this research endeavor. Their guidance, as well as their valuable tips and suggestions, significantly contributed to the enhancement of this study. Without their dedication and mentorship, I would not have been able to bring this thesis to fruition. I am profoundly indebted to them for the invaluable knowledge and experiences I gained during this research journey.*

***Asraa Mohsin… (2023)***

# Abstract

Medical image processing is a crucial field within medical science that greatly impacts visualization applications, It enhances the accuracy and efficiency of diagnosis, facilitates treatment planning, and enables monitoring of disease progression. By leveraging the power of digital image analysis.

The outbreak of the Coronavirus (COVID-19), caused by the SARS-CoV-2 virus, results in severe acute respiratory syndrome. A viral infection initially emerged in Wuhan at the end of 2019. Due to this outbreak, COVID-19 has evolved into a pandemic, posing a significant threat to human lives and causing disruption to the economy. Chest X-ray images have proven to be valuable in monitoring the impact of COVID-19 on lung tissue.

In this thesis, a system is proposed to classify COVID-19 based on chest X-ray images. for expanding the diversity of data and enhancing the system's performance in recognizing various cases, The system utilizes two datasets: chest x-ray images (pneumonia), consisting of 800 chest X-ray images and is named in the thesis DS1.and extensive covid-19 X-ray, consisting of 9000 chest X-ray images, has been called DS2.Preprocessing techniques such as resizing, denoising, and contrast enhancement are applied.

For each of the four orientations (0, 45, 90, and 135), six first-order statistical features are extracted using the Gray Level Co-occurrence Matrix (GLCM) method. Additionally, Local Binary Patterns (LBP) are used to extract ten first-order statistical features. The combination of LBP and GLCM methods has the potential to enhance the classification accuracy of medical images, particularly in texture classification.

This thesis proposed the feature selection approach in two cases. Feature selection is using The Extra Tree algorithm and Swarm algorithms, including The Moth-Flame

Optimization algorithm (MFO), The Grey Wolf Optimizer algorithm (GWO), The Glowworm Swarm Optimization algorithm (GSO), and Hybrid Moth Flame Optimization (MFO) and Glowworm Swarm Optimization (GSO) algorithms (HMFGWS). After feature selection, three classifiers, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Extreme Gradient Boosting XGBoost, are employed to assess the classification performance .

The results demonstrate that the proposed algorithms achieves superior performance in feature selection and classification. When using DS1, the accuracy rates are 99% with KNN, 72% with SVM, and 99% with XGBoost. When using DS2, the accuracy rates are 92% with KNN, 75% with SVM, and 97% with XGBoost.

# List of Contents

# List of Figures

# List of Tables

# List of Algorithm

# List of Abbreviations

| Abbreviations | Meaning |
|---|---|
| **ACO-SU** | Ant Colony Optimization with Symmetric Uncertainty |
| **ANA** | Ant Nesting Algorithm |
| **ANFC** | Adaptive Neuro-Fuzzy Classifier |
| **ASM** | Angular Second Moment |
| **BGA** | Binary Genetic Algorithm |
| **BGWO** | Binary Grey Wolf Optimizer |
| **BHHO** | Binary Harris Hawks Optimizer |
| **BPSO** | Binary Particle Swarm Optimization |
| **CAD** | Computer Aided Diagnosis |
| **CLAHE** | Contrast Limited Adaptive Histogram Equalization |
| **CNN** | Convolutional Neural Networks |
| **COR** | Correlation |
| **CT** | Computed Tomography |
| **CXR** | Chest X-Ray |
| **FN** | False Negatives |
| **FP** | False Positives |
| **GLCM** | Gray-Level Co-occurrence Matrix |
| **GSO** | Glowworm Swarm Optimization |
| **GWO** | Gray Wolf Optimizer |
| **HBGWOHHO** | Hybrid Binary Grey Wolf and Harris Hawks Algorithm |
| **HHO** | Harris Hawks Optimization Algorithm |
| **HMFGWS** | Hybrid Moth Flame Glowworm Swarm |
| **HOM** | Homogeneity |
| **IDCNN** | Inception-based Deep Convolutional Neural Network |
| **IDM** | Inverse Difference Moment |
| **JPEG** | Joint Photographic Experts Group |
| **JPG** | Joint Photographic Group |
| **KNN** | K-Nearest Neighbors' Algorithm |
| **LBP** | Local Binary Patterns |
| **MFO** | Moth-Flame Optimization |
| **MOPSO** | Multi-Objective PSO |

| | |
|---|---|
| **PNG** | Portable Network Graphics |
| **PSO** | Particle Swarm Optimization |
| **SSA** | Salp Swarm Optimization Algorithm |
| **SVM** | Support-Vector Machine |
| **TN** | True Negatives |
| **TP** | True Positives |
| **TS** | Tabu Search |
| **TSP** | Traveling Salesman Problem |
| **UCI** | University Of California, Irvine |
| **WOA** | Whale Optimization Algorithm |
| **WSN** | Wireless Sensor Network |
| **XGBoost** | Extreme Gradient Boosting |

# *Chapter One*

## *General Introduction*

## 1.1 Introduction

Medical imaging plays a vital role in diagnosing and treating a wide range of diseases. It offers valuable insights into the internal structures of the human body, empowering healthcare professionals to detect anomalies and make well-informed decisions. In recent years, advancements in machine learning and optimization techniques have revolutionized the field of medical image analysis, enhancing the accuracy and efficiency of diagnostic processes[1].

One of the most significant healthcare challenges in recent times has been the COVID-19 pandemic. As the virus spread rapidly across the globe, there is an urgent need for rapid and accurate detection methods to help healthcare systems manage the overwhelming number of cases. Medical imaging, such as chest X-rays and computed tomography (CT) scans, became indispensable tools for diagnosing COVID-19 and assessing its severity[2].

Machine learning, a subfield of artificial intelligence, has shown a remarkable potential in the classification and analysis of medical images. By leveraging large datasets of labeled images, machine learning algorithms can learn to identify patterns and features indicative of specific diseases or conditions[3]. This capability has been particularly valuable in the case of COVID-19, where machine learning models have been developed to classify medical images as either COVID-19 positive or negative. In addition to machine learning, metaheuristic optimization techniques have emerged as powerful tools in the field of medical image analysis. Metaheuristic algorithms are inspired by natural phenomena or problem-solving strategies and are capable of efficiently exploring large solution spaces to find optimal or near-optimal solutions. These techniques have been applied to various aspects of medical image analysis, including feature selection, image segmentation, and classification[4].

When it comes to COVID-19 classification using machine learning and metaheuristic optimization, The main objective is to create precise and dependable models that can assist in promptly and accurately diagnosing the illness[5]. By training machine learning models on large datasets of COVID-19-related images, these models can learn to identify specific patterns of COVID-19 infection. The models can then be used to classify new, unseen images as COVID-19 positive or negative.

Metaheuristic optimization techniques can further enhance the performance of these machine learning models by optimizing their hyper parameters or improving feature selection. These techniques enable fine-tuning of the models to improve their accuracy and generalization capabilities, thereby increasing their diagnostic value in real-world scenarios[6].

The combination of machine learning and metaheuristic optimization has brought significant advancements to the domain of medical image analysis, specifically within the context of COVID-19 classification. By leveraging these technologies, healthcare professionals can expedite the diagnosis of COVID-19, leading to faster and more targeted treatments. However, ongoing research and development efforts are necessary to refine these techniques further and maximize their potential for the benefit of patients worldwide[7].

## 1.2 Literature Review

This section discusses prior and correlated research on classification systems that utilize different algorithms to select optimal features between (2020 and 2023):

1. Sagban , et al.[8] , in 2020, Hybrid bat-ant colony algorithm was utilized to improve feature selection by leveraging the algorithm's frequency tuning and automatic zoom capabilities. Subsequently, researchers utilized the Ant-Miner classifier with a five-fold cross-validation approach, where each fold

employed 20% of the test data. The dataset used in the study consisted of 858 patients with cervical cancer, downloaded from the UCI repository. The study yielded accuracy rates of 95.93% for the Hinselmann test, 90.91% for Schiller's test, 94.88% for the Citology test, and 95.8% for the Biopsy test.

2. Fahad, et al.[9], in 2020, an approach called ACO-SU was introduced, which combines Ant Colony Optimization (ACO) with symmetric uncertainty. This novel method assessed the utility of incoming features and removes unnecessary ones. Whenever a new feature is detected, the algorithm updates the existing feature set. The researchers assessed the effectiveness of this new technique by utilizing 14 medical image datasets obtained from the UCI repository. They employed three classifiers (JRip, J48, decision table) to evaluate its performance, resulting in an average classification accuracy of 72.69%.
3. Dharmalingam, et al.[10], in 2021, a new method was presented, combining multi objective optimization with the Tabu search algorithm to overcome the limitations of the conventional single-objective Particle Swarm Optimization. This approach generates a set of optimal solutions, which are then used to select the most appropriate features extracted from (CT) images of the lungs. During the classification phase, the K-Nearest Neighbors (KNN) method was utilized, incorporating normal distribution and class probability. The dataset employed in this study encompasses lung CT images obtained from diverse sources, including Stanley Medical Hospital. The resulting overall classification accuracy reached 90.588%.
4. Al-Wajih, et al. [11], in 2021, a hybrid method known as HBGWOHHO was introduced, which merges the Harris Hawks and Binary Grey Wolf algorithms. This approach underwent evaluation using 18 standard UCI benchmark

datasets, and the quality of the selected features was assessed through a wrapper-based K-Nearest Neighbors (KNN) approach. The performance of the proposed hybrid technique was compared to that of Binary Grey Wolf Optimizer (BGWO), Binary Harris Hawks Optimizer (BHHO), Binary Particle Swarm Optimization (BPSO), Binary Genetic Algorithm (BGA), and Binary Hybrid BWOPSO. The average accuracy achieved with the suggested approach reached 92%.

5. Abu-Bakr, et al.[12], in 2022, a comparative analysis was conducted involving (KNN), (SVM) and (XGBoost) classification techniques, with a focus on their performance when coupled with optimization algorithms, including Harris Hawks Optimization (HHO), Salp Swarm Optimization (SSA), Whale Optimization Algorithm (WOA), and Gray Wolf Optimizer (GWO). These algorithms were employed for feature selection in the context of the research presented. The dataset used in this study comprised 9,000 2D X-ray images in the anterior chest view. The results indicated that the XGBoost and KNN techniques achieved an accuracy of 96%, while SVM attained an accuracy of 89%.
6. Alfian, et al.[13], in 2022, this research explored the use of (SVM) in conjunction with an Extremely Randomized Trees classifier (Extra-Trees) for early-stage breast cancer diagnosis based on risk factors. The Extra-Trees classifier was employed to eliminate irrelevant features, while the SVM was utilized for breast cancer diagnosis. A dataset comprising information from 116 subjects with breast cancer risk factors was utilized to train machine-learning models for predicting breast cancer, conducted model assessment through a stratified 10-fold cross-validation process. The combined SVM and

Extra-Trees model introduced in our research attained the highest level of accuracy, reaching up to 80.23%.

7. Abunadi, et al.[14], in 2022, this paper presents an innovative approach for diagnosing and categorizing COVID-19, known as the GSO-IDCNN model. This model combines automated Glowworm Swarm Optimization (GSO) with an Inception-based Deep Convolutional Neural Network (IDCNN). The proposed model incorporates a Gaussian Smoothing Filter (GSF) to remove noise from radiological images. Additionally, it utilizes the IDCNN-based feature extractor, taking advantage of the Inception v4 model. To further boost IDCNN's performance, hyperparameters are meticulously tuned using the GSO algorithm. Lastly, an Adaptive Neuro-Fuzzy Classifier (ANFC) is employed to detect the presence of COVID-19. The integration of the GSO algorithm with the ANFC model for COVID-19 diagnosis underscores the innovative nature of this research. To validate its effectiveness, extensive simulations were conducted on benchmark radiological imaging databases, resulting in an impressive accuracy of 0.9429.
8. Abu-Bakr, et al.[15], in 2022, a novel approach was introduced, presenting a hybrid binary version of (HHO) and (SSA), referred to as HHOSSA, for Covid-19 classification. The HHOSSA method aims to enhance the performance of the basic HHO by harnessing the Salp algorithm's capability to select the most optimal fitness values. To evaluate the effectiveness of HHOSSA, it was compared against two well-established optimization algorithms, the Whale Optimization Algorithm (WOA) and the Grey Wolf Optimizer (GWO), using a dataset comprising a total of 800 chest X-ray images. The evaluation involved four performance metrics (Accuracy, Recall, Precision, and F1) and utilized three classifiers SVMs, KNN, and XGBoost.

The HHOSSA algorithm achieved an accuracy of 96% with the SVM classifier and an impressive accuracy of 98% with the XGBoost and KNN classifiers.

9. Riana, et al.[16], in 2023, this paper conducts a comparison among three approaches: the KNN classification technique, Information Gain feature selection combined with KNN algorithm, and Information Gain feature selection in conjunction with PSO applied to the KNN classification algorithm. According to the research findings, the KNN classification, when coupled with Information Gain feature selection and PSO, outperforms several other models. Specifically, it achieved an accuracy rate of 87.33% with a value of K equal to five, surpassing the performance of the KNN model without feature selection or optimization and even surpassing the KNN model with Information Gain feature selection.

## Table 1.1 Summary of the Related works

| NO. | Ref & Year | Proposed method | Algorithm | Dataset | Classifier used | Accuracy |
|---|---|---|---|---|---|---|
| 1 | [8] 2020 | Hybrid bat-ant colony | Bat algorithm and ant colony algorithm | cervical cancer downloading from the UCI repository | Ant Miner classifier | Hinselmann achieved an accuracy of 95.93%, Schiller's achieved 90.91%, Cytology achieved 94.88%, and Biopsy achieved 95.8% |
| 2 | [9] 2020 | ACO-SU | Ant colony optimization and symmetric uncertainty | 14 medical image datasets from the UCI repository | J48, JRip, decision table | 72.69% |
| 3 | [10] 2021 | MOPSO + TS | multi-objective PSO and Tabu search | Stanley Medical Hospital and various other scanning facilities. | KNN Classifier | 90.588 % |
| 4 | [11] 2021 | HBGWOHHO | BGWO,BHHO,BPSO, BGA and BWOPSO | 18 standard UCI benchmark datasets | KNN Classifier | 92% |
| 5 | [12] 2022 | Comparative HHO, SSA, WOA, GWO | HHO, SSA, WOA, GWO | 9000 2D X-ray images: Two groups: 5500 pictures of healthy lungs and 4044 images of individuals with COVID-19. | XGBoost , KNN , SVM | 96%,96%,89 %. |
| 6 | [13] 2022 | The Extra-Trees classifier was employed to eliminate unimportant attributes, while SVM was used to determine the breast cancer status. | The extra-trees classifier | The dataset was collected from the Gynecology Department of Coimbra's University Hospital Centre (CHUC). Referred to as the Coimbra Breast Cancer Dataset (CBCD), it | SVM | 80.23% |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | includes information from 64 women diagnosed with breast cancer and 52 individuals who are in good health. | | |
| 7 | [14] 2022 | (GSO) with an inception-based (IDCNN) | (GSO) with (IDCNN) | A collection consisting of 220 COVID-19 images, 27 images displaying normal conditions, 15 pneumocystis images, and 11 images of SARS. | (IDCNN) | 94% |
| 8 | [15] 2022 | hybrid (HHO) (SSA) (HHOSSA) | (HHOSSA) | There are a total of 800 chest X-ray images, with 400 of them being chest X-ray images of COVID-19 cases, and the other 400 being chest X-ray images of normal conditions. | SVM , KNN , XGboost | 96%,98% and 98% |
| 9 | [16] 2023 | It examines the KNN classification method, Information Gain-based feature selection in combination with the KNN classification algorithm, and Information Gain feature selection along with PSO applied to the KNN classification algorithm. | PSO | The study utilized Twitter data concerning public opinions on the government's COVID-19 treatment measures in Indonesia. The collected data was sourced from the following link: https://www.kaggle.com/dionisiusdh/covid19indonesi an-twitter-sentiment, It was then selected, and a total of 1500 tweets were collected. | KNN | 87,33% |

## 1.3 Problem Statement

Medical imaging plays a vital role in the field of healthcare, aiding in the diagnosis and treatment of various medical conditions. These images provide valuable insights into the internal structures of the body, allowing healthcare professionals to make informed decisions about patient care.

The main challenge lies in obtaining huge and diverse collections of medical images accompanied by accurate diagnoses of cases. Proper and false classifications can be of paramount importance for training machine learning models.

The quality of data and the validity of diagnoses accompanying medical images must be ensured, as incorrect classification may lead to the circulation of inaccurate information and a serious impact on medical decisions.

The emergence of the COVID-19 pandemic, caused by the SARS-CoV-2 virus, has posed a substantial global challenge to public health. Early and accurate detection of COVID-19 cases is paramount for effective disease control and management.

Traditional diagnostic methods for COVID-19, such as PCR testing, have their limitations, including time-consuming procedures and limited testing capacity. Moreover, the interpretation of medical images can be affected by deteriorating image quality, and the fatigue of doctors and diagnostic specialists may impact the accuracy of visual diagnoses.

Despite advancements in machine learning, there is an urgent need to enhance the accuracy of classifying COVID-19 cases and examining medical images. Ongoing research is necessary to improve model performance and address challenges specific to disease-related medical images. Machine learning has shown great potential in classifying COVID-19 cases using medical imaging data, but there is room for improvement in accuracy. Optimizing machine learning algorithms for COVID-19

classification is crucial to enhance diagnostic capabilities in the fight against the pandemic.

## 1.4 The Aim of Thesis

The primary goal of the presented work is to enhance the system's efficiency in detecting cases of covid-19. The suggested method seeks to develop a best framework that combines using machine learning and metaheuristic optimization to selects the most effective features.

**The main objectives of this thesis can be summarized as follows:**

1- Developing a computer-aided detection (CAD) system as an assistant, which provides automated analysis of medical images taken for COVID-19 patients".

2- Utilizing multiple metaheuristic algorithms to select the most optimal features.

3-To evaluate the work, several metrics are used, including Precision, Recall, F1-score and accuracy.

## 1.5 Thesis Layout

In addition to the previously mentioned chapter, the other chapters of this study are structured as follows:

- Chapter Two, "Theoretical Background": This chapter offers an in-depth explanation of the theoretical concepts relevant to the present research, including feature extraction, feature selection, swarm intelligence, algorithms, and classifiers.
- Chapter Three "Proposed System": This chapter, shows a detailed description of the design and implementation of the proposed system. It also includes illustrative diagrams and algorithms to provide a clear understanding.

- Chapter Four, "Experimental Results and Evaluation": This chapter presents the experimental findings of the proposed system, supported by figures and tables, and includes a comparative analysis.
- Chapter Five, "Conclusions and Future Work": This final chapter encapsulates the research's conclusions drawn from the conducted experiments. It also offers suggestions for future research directions.

# *Chapter Two*

## *Theoretical Background*

## 2.1 Introduction

The domain of Computer-Aided Diagnosis (CAD) is continuously advancing in the medical field, providing potential support to physicians from diverse specialties in confidently interpreting medical images and enabling radiologists to conduct a more comprehensive examination of the image[17].

Medical images often contain noise, making them challenging to diagnose. To address this, medical image preprocessing helps to identify suspicious areas that are not visible to the naked eye but can be detected through the CAD system, providing a second diagnostic perspective to assist physicians and diagnostic experts in reaching a final decision[18].

Optimization-based techniques, known as metaheuristic algorithms, have emerged as a powerful tool for solving optimization problems, offering simplicity, flexibility, and the ability to overcome the problem of local optima[19].

These algorithms typically comprise two main stages: exploration and exploitation, where the former involves a thorough search of the promising search space and the latter involves local searches in the identified promising area(s). In recent times, swarm intelligence algorithms have become a fresh and innovative method within computer-aided design (CAD) for the specific goal of selecting features [20].

In this chapter, explore the methods utilized for the preprocessing of medical images, extracting features from them, selecting relevant features, and performing classification. Additionally, the thesis will cover the fundamental algorithmic principles and terminology underlying these methods.

## 2.2 Image Preprocessing

Preprocessing can play a vital role in improving the quality of images by eliminating irrelevant details or enhancing specific visual features necessary for further processing, such as resizing, denoising, or improving the image's histogram. Geometric image alterations like rotation, scaling, and translation are categorized as preprocessing techniques because they utilize analogous approaches [21]. Image preprocessing encompasses a wide array of methods, encompassing tasks like image resizing, image filtering, and enhancing contrast.

1. Image resizing: involves changing the dimensions of an image, either to make it smaller or larger. This technique is used to adjust the size of an image to fit a particular display or storage format, or to reduce the processing time for large images. Image resizing can be done using various algorithms, such as nearest neighbor, bilinear, bicubic, and Lanczos interpolation[22].

2. Image filtering: involves applying a mathematical function or kernel to an image to remove noise, enhance details, or blur the image. The most commonly used filters are the Gaussian filter, median filter, and edge detection filter. Image filtering is used to improve the quality of an image and make it easier to analyze for specific applications[23].

3. Contrast enhancement: involves increasing the dynamic range of an image, which means increasing the contrast between the light and dark areas of the image. This method is employed to enhance the visibility of subtle image details and enhance the overall image quality. There are several approaches available for contrast enhancement, including histogram equalization, contrast stretching, and adaptive histogram equalization, among others [24].

## 2.2.1 Gaussian Smoothing Filter

Gaussian smoothing, also known as Gaussian blur, is a widely used image processing technique for reducing noise and blurring an image. It applies a Gaussian filter to the image, which convolves the image with a Gaussian kernel, resulting in a blurred version of the original image[25]. The Gaussian filter is a low-pass filter that attenuates high-frequency components, effectively smoothing out the image.

The Gaussian-smoothing filter is based on the Gaussian function, which is a continuous probability distribution defined by the equation:

$$G(x, y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2 + y^2}{2\sigma^2}} \quad (2.1)$$

In this equation, G( x , y ) represents the Gaussian function evaluated at the position (x, y), π is the mathematical constant pi, and σ represents the standard deviation of the Gaussian distribution. The standard deviation controls the amount of blurring applied to the image. A higher value of σ results in a stronger blur[26].

## 2.2.2 Contrast Limited Adaptive Histogram Equalization (CLAHE)

It is an image processing method that has been applied in diverse domains, including medical imaging. In the context of COVID-19 images, CLAHE can play a significant role in enhancing the visualization of relevant features and improving the diagnostic accuracy[27].

CLAHE is a modification of the conventional histogram equalization method that aims to overcome its limitations, such as the amplification of noise and the loss of local contrast. It accomplishes this by segmenting the image into small sections known as tiles and independently applying histogram equalization to each tile. This approach allows CLAHE to dynamically improve contrast within specific regions of the image, maintaining both overall and localized information Can be represented as an equation [28] :

ApplyCLAHE(I,tile_size,clip_limit)=EnhanceHistogram($I_{tiles}$,clip_limit) (2.2)

ApplyCLAHE represents the application of the CLAHE (Contrast Limited Adaptive Histogram Equalization) technique to an input image I. $I_{tiles}$denotes the image I that has been divided into small tiles. Each tile is a localized region of the image. ,tile_size is the parameter specifying the size of each tile. clip_limit is another parameter representing the clip limit, which determines the maximum limit for the amplification of histogram equalization within each tile. EnhanceHistogram is a function that performs histogram equalization on each tile independently, enhancing the contrast within specific regions of the image.

When applied to COVID-19 images, CLAHE can help reveal subtle details and variations in the lung tissue affected by the infection. This is particularly useful in chest X-rays and computed tomography (CT) scans, which are commonly used for diagnosing COVID-19 pneumonia. By improving contrast and highlighting relevant features, CLAHE enables radiologists and healthcare professionals to identify and analyze abnormalities more effectively[29].

## 2.3 Feature Extraction

Feature extraction for X-ray images involves the process of identifying and extracting relevant information or characteristics from the images to facilitate analysis and interpretation[30]. The goal is to capture discriminative features that can aid in tasks such as disease diagnosis, anomaly detection, or image classification. Here are some feature extraction techniques commonly used in X-ray images:

1- ***Intensity-based Features****:* focus on the pixel intensities in the image. They can capture information about the brightness, contrast, or texture of the X-ray image. Examples of intensity-based features include mean intensity, standard

deviation, histogram-based features, and texture features like gray-level co-occurrence matrix (GLCM) features or local binary patterns (LBP)[31].

2- ***Texture-based Features:*** focus on the patterns and spatial arrangements of pixel intensities within the X-ray image. They can provide information about the roughness, coarseness, or regularity of the texture. Statistical features like (GLCM) features, (LBP), or Gabor wavelet features are commonly used for texture characterization in X-ray images[32].

3- ***Shape-based Features*:** focus on the geometric characteristics of objects within the X-ray image. These features can be useful for detecting and characterizing specific anatomical structures or abnormalities. Examples of shape-based features include area, perimeter, aspect ratio, circularity, and eccentricity of objects[33].

4- ***Edge-based Features*:** are derived from the edges or boundaries of objects in the X-ray image. These features can capture information about the contours and shapes of structures. Common edge-based features include edge density, edge length, and curvature[34].

5- edge-based features include edge density, edge length, and curvature[34].

### 2.3.1 Gray Level Co-Occurrences Matrix Features

The GLCM (Grey-Level Co-Occurrence Matrix) is a basic and straightforward technique for extracting texture features. It has been widely employed across diverse texture analysis applications and remains a valuable approach for feature extraction in this domain. The GLCM generates eight statistical measures, including angular second moment (ASM), variance, correlation (COR), inverse difference moment (IDM), contrast, entropy, mean, homogeneity (HOM), and which together define the texture features. By establishing connections between grayscale values of adjacent

pixels at different angles, The GLCM encodes extensive image details into a matrix format based on pixel distance and orientation. It is a second-order statistical technique tailored for texture analysis in grayscale images. Nevertheless, it involves intricate computations. When employing all 256 gray levels to construct the GLCM, each resulting matrix measures 256x256. Extracting textural attributes from this matrix entails the computation of every element, thereby larger matrix sizes demanding more computations[35][36]. The ensuing features unveiled by the histogram analysis are listed in table (2.1)[37]:

**Table (2.1) GLCM Features' Formula**

| Feature | Formula |
|---|---|
| **Contrast** | $\sum_{1;j=0}^{n-1} p_{i,j\,(i-j)^2}$ |
| **Correlation** | $\sum_{1;j=0}^{n-1} p_{i,j}\left[\frac{(i-\mu_i)(j-\mu_j)}{\sqrt{(\sigma_i^2)}(\sigma_j^2)}\right]$ |
| **Energy** | $\sum_{1;j=0}^{n-1} p_{i,j\,\lvert i-j\rvert}$ |
| **Entropy** | $\sum_{1;j=0}^{n-1} p_{i,j(-ln\,p_{i,j})}$ |
| **Homogeneity (inverse difference moment)** | $\sum_{1;j=0}^{n-1} \frac{p_{i,j}}{1+(i-j)^2}$ |
| **Mean** | $\mu_{i=\sum_{i,j=0}^{n-1} i(P_{i,j}),\mu_{ij=}\sum_{i,j=0}^{n-1} j(P_{i,j})}$ |

| | |
|---|---|
| **Standard Deviation** | $\sqrt[2]{\sum_{g=0}^{L-1} P(g-\mu)^2 P(g)}$ |
| **Uniformity (Angular Second Moment)** | $\sum_i \sum_j \{P(i-j)^2\}$ |

## 2.3.2 Local Binary Pattern Features (LBP)

Is a widely used feature extraction technique in medical image classification tasks. It captures the texture information of images by comparing the intensities of a central pixel with its surrounding neighbors. LBP has shown effectiveness in various medical image analysis applications, including tumor classification, lesion detection, and disease diagnosis[38]. Here's an overview of feature extraction using LBP in medical image classification:

***1. LBP Calculation:*** The calculation of LBP involves comparing the intensity values of a center pixel with its neighboring pixels. Here's the step-by-step process with equations[39]:

**a**. specify a center pixel in the image, denoted as $I_C$.

**b**. specify a neighborhood around the center pixel, typically represented by a circular region.

**c**. Compare the intensity value of each neighboring pixel, denoted as $I_n$, with the center pixel's intensity value $I_C$.

**d**. Assign a binary value of '1' if $I_n \geq I_C$, and '0' otherwise

$$b\_i = \begin{cases} 1 & \text{if } I_n \geq I_C \\ 0 & \text{if } I_n < I_C \end{cases} \qquad (2.3)$$

Where (b_i) represents the binary value of the (i-th) neighboring pixel.

**e**. Traverse the neighborhood in a circular order, and concatenate the binary values to form a binary code. Let's denote the binary code as B.

For a neighborhood with P neighboring pixels, the resulting binary code B will be a concatenation of P binary values. Mathematically, B can be represented as:

B = b_ {P-1} b_ {P-2} ... b_1 b_0 (2.4)

***2. LBP Histogram:*** Once the LBP values are calculated for all pixels in the image, a histogram is constructed to represent the distribution of different LBP patterns. The histogram captures the frequency of occurrence of each LBP pattern[40].

H= (H1, H2, ... , HL)

Let's denote the number of different LBP patterns as L. The histogram, denoted as H, can be represented as an L-dimensional vector, where each element (H_i) represents the frequency of the (i-th) LBP pattern in the image.

***3. Feature Representation:*** The LBP histogram serves as the feature representation of the image. It summarizes the frequency of different LBP patterns present in the image[41].

***4. Classification:*** Once the LBP features are extracted, they can be used as input for various classification algorithms, such as support vector machines (SVM), random forests, or convolutional neural networks (CNN). These algorithms learn from the LBP feature representations and classify the medical images into different categories or classes[42].

The LBP method utilizes specific equations (from Eq (2.5) to Eq (2.11)) to create the set of features [43]:

$$LBP = \sum_{n=0}^{7} s(I_n - I_c) \times (2^2) \quad (2.5)$$

Eq (2.4) computes the LBP value for a pixel by summing the differences between its neighboring pixels and the central pixel.

$$A_{i,j}^{1} = S(I_1, I_C) \times (2^7) + S(I_2, I_C) \times (2^6) \quad (2.6)$$

$$A_{i,j}^{2} = S(I_3, I_C) \times (2^5) + S(I_4, I_C) \times (2^4) \quad (2.7)$$

$$A_{i,j}^{3} = S(I_5, I_C) \times (2^3) + S(I_6, I_C) \times (2^2) \quad (2.8)$$

$$A_{i,j}^{4} = S(I_7, I_C) \times (2^1) + S(I_8, I_C) \times (2^0) \quad (2.9)$$

Eq (2.6) to Eq (2.9) determine four distinct texture patterns ($A_{i,j}^{1}$ to$A_{i,j}^{4}$) by evaluating the central pixel in relation to its eight neighboring pixels arranged in a circular manner.

$$A = A_{i,j}^{1} + A_{i,j}^{2} + A_{i,j}^{3} + A_{i,j}^{4} \quad (2.10)$$

Eq (2.10) aggregates the four texture patterns ($A_{i,j}^{1}$ to$A_{i,j}^{4}$)) into a single value, A.

$$LBP = \text{Histogram}_A \quad (2.11)$$

Eq (2.11) Forms the histogram of the combined texture patterns (A) for every cell within the image

$$S(z) = \begin{cases} 0, z < 0 \\ 1, z \geq 0 \end{cases} \quad (2.12)$$

Eq (2.12) is a binary function that outputs 1 when its argument is non-negative and 0 otherwise.

These equations hold significant importance as they define the role of *I*c, which represents the pixel value in the center, and *I*n, which represents the pixel values of the neighboring pixels. The LBP operator, grounded in these equations, generates binary values (z) based on the difference between the central pixel and its adjacent pixels. This resultant LBP value, in conjunction with the texture patterns ($A_{i,j}^{1}$ to$A_{i,j}^{4}$), is employed to capture distinctive texture characteristics within the CXR images. Figure 2.1 illustrates the pixel labeling technique used by the LBP operator[41].

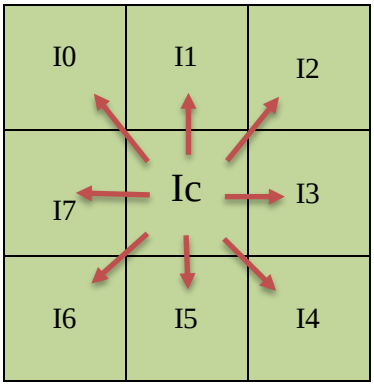



**Figure (2.1) shows the LBP operator labeling pixels**

Figure (2.2) shows an example of computing the original LBP method for a micro pattern[44].

The LBP for, 10001011‟ is getting for 3 by 3 micro patterns with the center pixel value6. The binary representation is converted into its decimal counterpart, resulting in 139 duplicate values. LBP histograms are then generated based on all micro patterns associated with these decimal values.

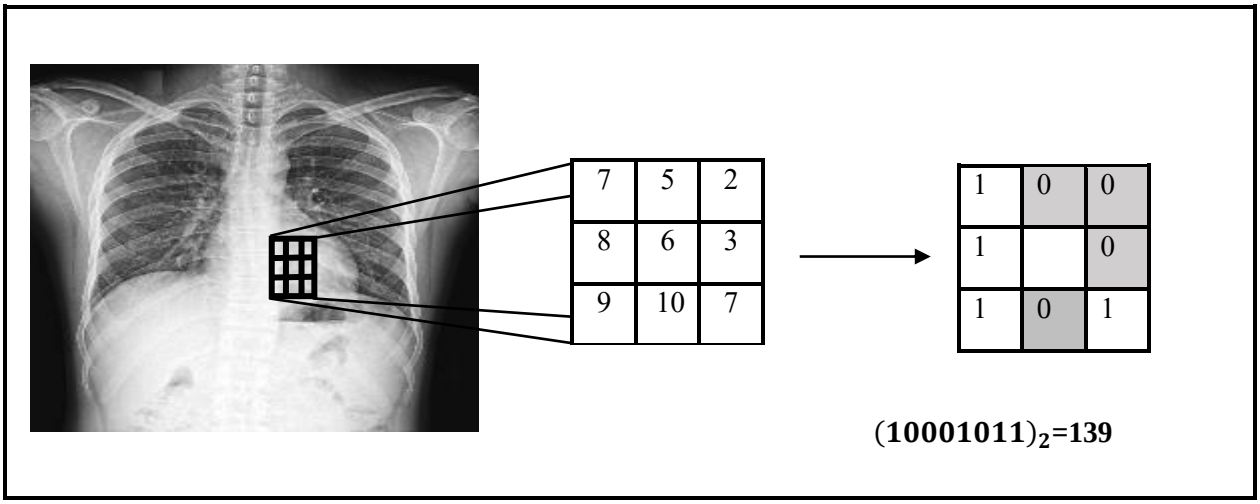



**Figure (2.2) shows an example of computing the original LBP**

## 2.4 Feature Selection

The procedure of feature selection, in machine learning and data analysis, involves the identification and selection of a subset of significant features from the initial set of input variables. The primary objective of feature selection is to enhance model performance, decrease overfitting, improve interpretability, and reduce computational complexity by concentrating on the most informative and distinguishing features[45].

Feature selection techniques can be broadly categorized into three main approaches: embedded methods, wrapper methods, and filter methods[46]. Embedded methods include feature selection within the actual process of training the model. These methods aim to find the optimal subset of features that maximize the model's performance or minimize a specific loss function. Embedded methods have the advantage of being computationally efficient since feature selection is integrated into the model training process. However, they may be limited to the specific model they are embedded in and may not generalize well to other models. Wrapper methods assess the effectiveness of various sets of features by employing a particular learning algorithm as a black box. These methods create a loop where different feature subsets are selected, and the learning algorithm is trained and evaluated on each subset[47]. The evaluation criterion (e.g., accuracy, cross-validation score) is used to determine the best subset of features. Wrapper methods can consider the interactions and dependencies between features but tend to be computationally expensive since they require training and evaluating the model multiple times for different feature subsets. Filter methods assess the relevance of features based on their statistical properties or their relationship with the target variable, independent of any specific learning algorithm. These methods rank or score the features using various statistical

measures and select the top-ranked features. Filter methods are computationally efficient and can be applied before model training[48].

### 2.4.1 Extra Trees Classifier Feature selection

Feature selection is an important step in machine learning that involves identifying the most relevant features from a given set of input features (also known as predictors or independent variables). The goal of feature selection is to improve model performance, reduce overfitting, and enhance interpretability by focusing on the most informative features[49].

ExtraTreesClassifier is a machine learning algorithm that belongs to the family of ensemble methods, specifically the Random Forest algorithm. It creates an ensemble of decision trees and combines their predictions to make the final prediction. ExtraTreesClassifier introduces additional randomness in the tree construction process compared to traditional decision trees, which can lead to improved performance[50].

Can use ExtraTreesClassifier as a tool for feature selection indirectly. The algorithm assigns importance scores to each feature based on their contribution to the prediction. By analyzing these importance scores, you can identify the most informative features and exclude less relevant ones[51].

feature selection can be performed using ExtraTreesClassifier[52][51]:

1- Training the ExtraTreesClassifier: First, you need to train an ExtraTreesClassifier model on your dataset. This involves providing your feature matrix X and target vector y to the classifier and fitting the model to the data.
2- Extracting feature importances: After training the ExtraTreesClassifier, you can access the feature importances assigned by the model to each input feature.

Feature importances indicate the relative importance of each feature in making accurate predictions. These importances are calculated based on how much each feature contributes to reducing the impurity (e.g., Gini impurity) or improving the separation of classes within the decision trees.

3- Selecting features: Once you have obtained the feature importances, you can proceed with selecting the most relevant features. There are different approaches you can take:
   A- Top-k feature selection: You can choose a specific number k and select the top k features with the highest importances. This approach is suitable when you have a predetermined number of features to select.
   B- Threshold-based selection: Alternatively, you can set a threshold value for the importance score and select features whose importance exceeds that threshold. This approach allows you to select a variable number of features based on their importance.
4- Creating a new feature matrix: Finally, you create a new feature matrix by selecting only the chosen features. This reduced feature matrix contains the subset of features that are considered most important for the model.

By employing ExtraTreesClassifier for feature selection, you can leverage the inherent feature importances provided by the algorithm to identify and select the most informative features. This can lead to improved model performance, reduced overfitting, and enhanced interpretability.

Extra-Trees comprise a multitude of individual decision trees, with each decision tree grown using the entire training dataset [53]. A decision tree is structured with a root node, child nodes, and leaf nodes, as illustrated in Figure (2.3)

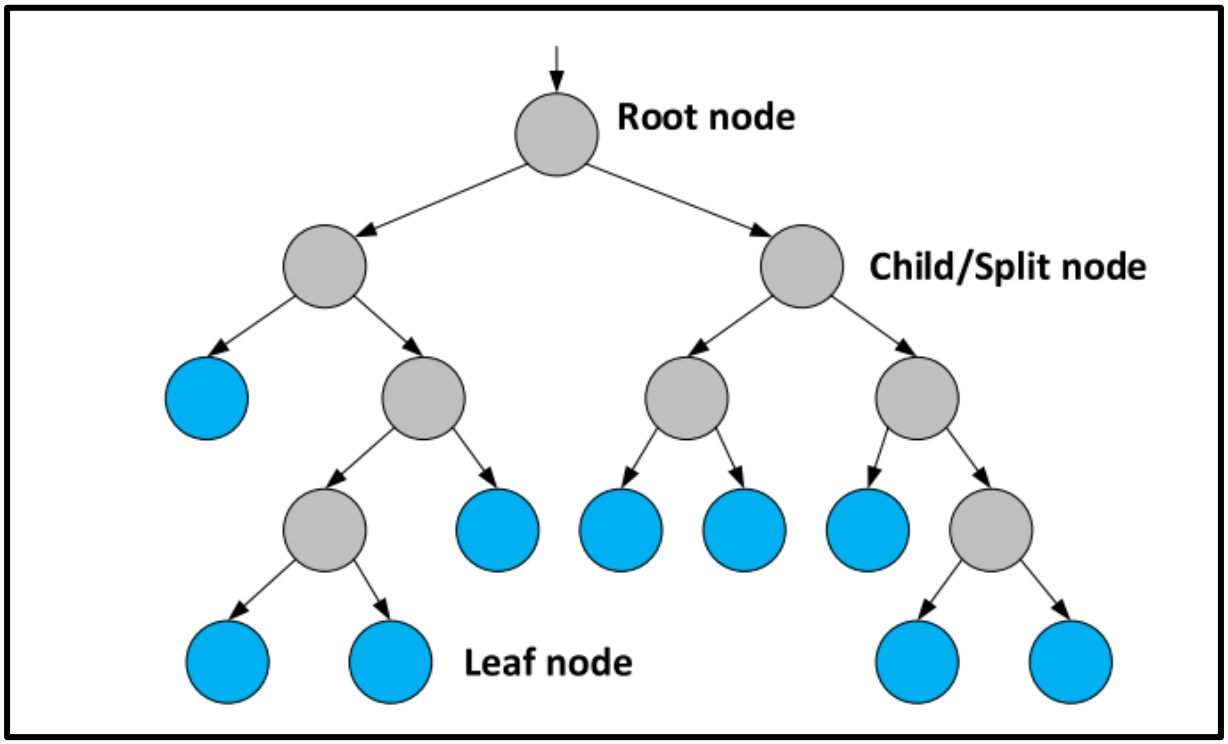



**Figure (2.3): Illustration of a decision tree.**

Given a training dataset, denoted as X = {x1, x2, ..., $x_N$}, where each sample xi = {f1, f2, ..., $f_D$} is a D-dimensional vector with $f_j$ representing the feature, where j ∈ {1, 2, ..., D}, Extra-Trees generates M distinct decision trees. In each decision tree, $S_p$ represents the subset of the training dataset X at a specific child node p. Then, at each node p, the Extra-Trees algorithm selects the optimal split based on $S_p$ and a random subset of features. the subset $S_p$ at child node p is divided into two sets: $S_p^{right}$ containing those samples satisfying the condition of the split rule, and $S_p^{left}$ containing the rest of the training samples. The stages of Extra-Trees algorithm are detailed in Algorithm (2.1).

**Algorithm 2.1: Extra-Trees algorithm**

**inputs**: Training subset $S_p$ is defined as{ s1,s2, ..., $S_{Qp}$ , where each sample $S_i$ is represented as a D-dimensional vector{f1, f2, ..., $f_D$} is a D-dimensional vector, the number of attributes to select randomly, K, and the minimum number of samples required to split a node, $n_{min}$.

**Output**: best split [$f_* < f_*^c$ ] at the child node p.

**Begin**

**If** $Q_p <$ $n_{min}$ or all observations within the node have an identical label.

Stop splitting and define the node as a leaf node.

**Else**

Select a random subgroup of K features {f1, f2, ..., $f_k$} among the original D features.

**For** each feature k in the subgroup do :

Find $f_k^{max}$ and $f_k^{min}$ as the maximal and minimal values of the feature k in subset$S_p$.

Obtain a random cut-point, $f_k^c$, uniformly in the range $\left[f_k^{min}, f_k^{max}\right]$ .

Set[$f_k < f_k^c$ ] as a candidate split

**End for**

Select a split [$f_* < f_*^c$ ]such that S core($f_*^c$) = $\min_{k=1,\ldots k}$ S core ($f_k^c$)

**End**

## 2.4.2 Swarm Intelligence

Swarm Intelligence is a collective behavior observed in groups of decentralized, self-organized systems, inspired by the behavior of social insect colonies or animal herds. It is a field of study that explores how simple individuals, following local rules and interacting with each other, can collectively solve complex problems or exhibit intelligent behavior as a group[54].

The key idea behind Swarm Intelligence is that the collective behavior emerges from the interactions and coordination among the individuals, rather than being dictated by a central authority or control. Each individual, often referred to as an agent or a particle, typically has limited knowledge and capabilities but can

communicate, share information, and adapt their behavior based on local interactions with their neighbors or the environment[55].

Swarm Intelligence offers valuable insights, methodologies, and algorithms for solving complex problems, especially in dynamic and uncertain environments. Its applications span diverse domains[56].

### 2.4.2.1 Glowworm Swarm Optimization algorithm (GSO)

Is a metaheuristic optimization algorithm inspired by nature, drawing its principles from the coordinated actions of glowworms. It was proposed by Krishnan and Ghose in 2005, is a recently introduced in the field of swarm intelligence, inspired by the behavior of real glowworms. In this algorithm, each glowworm represents a potential solution to an optimization problem within a search space and carries a certain amount of "luciferin," which is a luminescent substance[57].

The movement of the glowworms is controlled by their proximity to luciferin, where the brighter the glowworm (higher luciferin intensity), the more favorable its position in the search space. Thus, the level of luciferin attached to each glowworm serves as an indicator of its fitness or quality of solution. Glowworms employ a probabilistic technique to navigate towards neighboring glowworms within their local-decision area. They are attracted to those neighbors with higher luciferin intensity than their own, aiming to move towards better solutions in the optimization problem [58].

The decision radius and the size of the local decision domain for each glowworm is influenced by the number of its neighbors. If the density of neighbors is low, the local-decision domain expands to search for more potential solutions. On the other hand, if the density of neighbors is high, the domain shrinks, enabling the swarm to divide into smaller groups and explore different regions of the search space[59].

The algorithm progresses through iterations, and in each iteration, glowworms adjust their brightness and update their positions. The update is guided by the attraction and repulsion between glowworms. Glowworms are attracted to brighter neighbors and tend to move towards them, while they are repelled by other glowworms within their sensing range[60].

The GSO algorithm also incorporates a local search mechanism, which allows the glowworms to refine their positions in the search space and explore the local neighborhoods more effectively.

Here is equation used in the Glowworm Swarm Optimization (GSO) algorithm, The movement equation determines how each glowworm updates its position in the search space based on the attractiveness and repulsion factors.

It is typically represented as in equation (2.13)[61]:

$$\boldsymbol{x_i^{(t+1)} = x_i^{(t)} + \delta_i^{(t)} . StepSize . \frac{x_{best}^{(t)} - x_i^{(t)}}{||x_{best}^{(t)} - x_i^{(t)}||}} \quad (2.13)$$

Where:

- $x_i^{(t)}$ is the position of glowworm i at iteration t,
- $\delta_i^{(t)}$ is a random number in the range [0, 1],
- StepSize is a parameter controlling the step size,
- $x_{best}^{(t)}$ is the position of the best glowworm within the sensing range of glowworm i at iteration t,
- ||·|| denotes the Euclidean distance.

This equation calculates the new position of glowworm i by moving towards the best glowworm in its neighborhood, with a step size controlled by StepSize and a random perturbation factor $\delta_i^{(t)}$. The stages of GSO are detailed in Algorithm (2.2)[61].

**Algorithm 2.2 : The Glowworm Swarm Optimization**

**Input**:
n: Number of glowworms.
I: Initial luciferin value for each glowworm.
D: The target destination node.
γ: A parameter for updating luciferin.
p: another parameter for updating luciferin.
t : Time step

**Output**:
Data message sent from node S to the destination node D.

**Begin**
Generate n glowworms.
Set the initial luciferin value for each glowworm to I.
Direct the glowworms toward D.
**For** each glowworm i ($1 \leq i \leq n$) when reach a vehicle
Compute Di (t) and pi (t).
Update luciferin using Equation
li (t +1) = (1-p )*1i (t) + γ*J (xi (t +1 )).
Select nexthop according to the probability Equation
Pij(t) = (1 - li(t)) / ∑k∈Ni(t) (1 - lk(t))
Send glowworm i to the next-hop.
**End For**
Node S sends data to neighbor with maximum luciferin
**If** (receiving node is D) then
Save data message
**Else**
Send data to a neighbor with maximum luciferin.
**End**

### 2.4.2.2 Moth-Flame Optimization algorithm (MFO)

Is a nature-inspired metaheuristic optimization algorithm introduced by Mirjalili in 2015. It draws inspiration from the behavior of moths attracted to flames and aims to find optimal or near-optimal solutions. Moths are naturally attracted to bright light sources, exhibiting positive phototactic behavior, which forms the basis for the algorithm[62].

The MFO algorithm simulates this natural behavior by modeling moths' attraction to flames. It iteratively adjusts the positions of moths to emulate their flight towards the brightest moth, which represents the flame. This process involves exploring the search space to seek better solutions [63].

Updating the position of a moth *i* at iteration *t* is represented as in equation (2.14):

$$\boldsymbol{x_i^{(t+1)} = x_i^{(t)} + \beta . e^{-\gamma . d_i^{(t)}} . x_{best}^{(t)} - x_i^{(t)}} + \epsilon . rand() \qquad (2.14)$$

Where:

- $x_i^{(t)}$ is the position of moth i at iteration t,
- $x_{best}^{(t)}$ is the position of the currently best moth (the brightest) at iteration t,
- $d_i^{(t)}$ is the Euclidean distance between moth i and the flame (i.e., the distance to the currently best moth),
- β controls the moth's attraction towards the flame,
- γ controls the effect of the distance on the movement,
- ϵ is a random perturbation factor,
- rand () generates a random number between 0 and 1.

The MFO algorithm's stages are detailed in Algorithm 2.3[64].

**Algorithm 2.3: The Moth-Flame Optimization**

**Input**: Objective function f(X), X = (X1 X2… Xd), Number of moths in the population (N), dimension (d), Maximum iteration (Maximum*iter*), Flame number (N.FM), b and other related parameters are determined؛

**Output**: The best solution with the minimum fitness function value in the ecosystem

**Begin**

Initialize the parameters for Moth-flame

Initialize Moth position $M_2$ randomly

**for** i = 1 to n do

Calculate the fitness function fi

**end for**

**while** iteration Max_iterations do

//Update the position of M Calculate the number of flames using

N.FM=round(N×rand(0,1))

Evaluate the fitness function fi

**if** iteration ==1 then

F=sort(M) and OF=sort (OM)

**else**

Fsort(M-1, Mt) and OF=sort(M-1, Me)

**End if**

**for** i = 1 to n do

**for** j = 1 to d do

//Update the values of r and t

//Calculate the value of D respect to its corresponding moth using

D(i,j)=‖X(i,j)−Flame(j)‖

//Update M(i,j)respect to its corresponding moth using

X(i,j)=X(i,j)×exp(−b×D(i,j)×t)+rand(0,1)×(F(j)−OF(j))

**end for**

**end for**

**end while**

Print the best solution

**End**

*//Where N is the number of moths in the population, d is the dimension, t is the time (number of iterations), b is the control parameter for the rate of descent, X(i,j) is the position of the i-th moth in dimension j, Flame(j) is the position of the flame in dimension j, rand(0,1) generates a random value between 0 and 1, ‖·‖ is the shortest distance between two points in space.*

MFO leverages the transverse orientation navigational strategy of moths, allowing them to maintain their orientation with respect to the moon or a source of light while flying at night. In the algorithm, moths represent candidate solutions, and their flight towards the light source symbolizes the search for an optimal solution. The intensity of the light source mirrors the objective function to be optimized. During each iteration, moths update their positions based on their orientation towards the light source. Moths with better fitness attract others to move towards them, mimicking the natural behavior of moths seeking brighter regions. Additionally, random movements are introduced to ensure diversity and exploration within the search space [65].

The primary goal of the Moth-Flame Optimization algorithm is to iteratively refine the positions of moths until a satisfactory or near-optimal solution is achieved. By emulating moths' navigation behavior and their attraction to light sources, the algorithm effectively explores the search space and converges towards promising regions, making it suitable for optimizing a wide range of problems[62].

### 2.4.2.3 Grey wolf Optimizer Algorithm (GWO)

Is a cutting-edge metaheuristic technique inspired by the cooperative behaviors and characteristics of grey wolves. It leverages a population-based approach, drawing inspiration from the collaborative hunting strategies employed by this specific subspecies of wolves. Grey wolves are renowned for their high intelligence, adaptability, and well-structured social hierarchy, which enables them to efficiently hunt prey and overcome challenges in their environment. The GWO algorithm emulates these traits to effectively address complex optimization problems[66].

Within the GWO algorithm, the population represents a pack of candidate solutions, much like a wolf pack. Each candidate solution serves as a potential answer to the optimization problem. Similar to the wolves, the algorithm fosters collaboration and information sharing among individuals in the population. During the optimization process, the wolves simulate hunting and exploration behaviors to seek the optimal solution. At each iteration, the GWO algorithm updates the position of each "wolf" (candidate solution) based on its fitness, reflecting its performance in solving the optimization problem. The wolves dynamically adjust their positions using three primary actions: encircling prey, attacking, and following the leader. These actions replicate the hunting strategies of real wolves[67].

"Encircling prey" imitates wolves surrounding solutions with better fitness to enhance exploration of the search space. "Attacking" focuses on the most promising solutions to intensify the exploitation of the search space. "Following the leader" mirrors the wolves' inclination to be guided by the alpha wolf, the one with the highest fitness, to converge toward a better solution[68]. By utilizing these behaviors and a population-based approach, the GWO algorithm efficiently explores the solution space while striking a balance between exploration and exploitation. The stages of GWO are detailed in Algorithm (2.4)[69].

| **Algorithm 2.4: The Grey Wolf Optimization** |
|---|
| **Inputs**: the values for N and T.<br>**Outputs**: "Xα" |
| **Begin**<br>Initialize the population of the grey wolves Xi (I = 1, 2, ..., n)<br>Initialize a $Aw$, and $Cw$, 3.<br>Calculate the fitness values of each wolf<br>Xα = the best fitness value as the alpha wolf,<br>Xβ = the second-best fitness value as the beta,<br>Xδ = the third-best fitness value as the delta wolf<br>while not (stopping criteria)<br>Update the positions of the Omega wolves accordingly<br>Update a $Aw$, and $Cw$,<br>Calculate the fitness values of all wolves and sort them<br>Update the positions of Xα, Xβ, and Xδ<br>t=t+1<br>end while<br>Return Xα.<br>**End** |

## 2.5 Machine Learning Image Classifiers

Image classifiers in machine learning refer to algorithms or models that are trained to recognize and categorize images into different predefined classes or categories. These classifiers use various machine learning techniques to learn patterns and features from image data, enabling them to make accurate predictions on unseen images[70]. There are different approaches and algorithms used for image classification in machine learning like SVM, KNN, and XGBoost.

### 2.5.1 Support Vector Machine (SVM)

Is a popular supervised machine-learning algorithm used for classification and regression tasks. SVM is particularly effective in scenarios where the data is non-linearly separable or has a high dimensionality[71].

In classification tasks, SVM aims to find an optimal hyperplane that separates the data points belonging to different classes with the maximum margin. This hyperplane is determined by support vectors, which are the data points closest to the decision boundary. The margin is the distance between the hyperplane and the support vectors[72].

SVM can handle both linearly separable and non-linearly separable data by employing a kernel function. The kernel function maps the input data into a higher-dimensional feature space, where the data points may become linearly separable. This transformation allows SVM to find non-linear decision boundaries in the original feature space. The stages of SVM are detailed in Algorithm (2.5)[73].

**Algorithm (2.5): SVM algorithm**

**Input:** Training dataset: $(X_1, y_1)$, $(X_2, y_2)$, …, $(X_n, y_n)$, where ($X_i$, is the feature vector and $y_i$ is the corresponding class label ($y_i \in \{-1,1\}$

**Output**: A hyperplane represented by the weights (w) and bias (b).

**Begin**

Normalize or scale the feature vectors if needed.

Initialize weights w and bias b to zeros.

Choose a regularization parameter C.

Use an optimization algorithm to minimize the following objective function:.

$$\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \max(0, 1 - y_i(w.x_i + b))$$

//(*The optimization process involves updating w and b iteratively to find the optimal hyperplane that maximally separates the classes while minimizing classification errors*).

$$y_{new} = sign(w.x_{new} + b)$$

*// (Given a new input $x_{new}$, predict the class label $y_{new}$ using the learned Hyperplane)*

**End**

### 2.5.2 K-Nearest Neighbor (KNN)

Is a simple yet powerful supervised machine-learning algorithm used for classification and regression tasks, and it provides a simple yet effective approach for making predictions based on the nearest neighbors in the feature space. KNN is a non-parametric algorithm, meaning it does not make any assumptions about the underlying data distribution[74].

In the KNN algorithm, the "K" refers to the number of nearest neighbors considered for making predictions.

The choice of K is important and can impact the performance of the KNN algorithm. A smaller K value makes the model more sensitive to noise, while a larger K value may lead to over smoothing and less local sensitivity. Thus, selecting an optimal K value is often determined through cross-validation or other model evaluation techniques[75].

The KNN algorithm is straightforward to implement and does not involve explicit model training. However, it can be computationally expensive, especially for large datasets, as it requires calculating distances between the test instance and all training instances. Additionally, proper data preprocessing, feature scaling, and handling missing values are important steps to ensure the algorithm performs effectively[76]. The stages of KNN are detailed in Algorithm (2.6)[77].

**Algorithm (2.6): KNN algorithm**

**Input:**

- Training dataset X with features
- Corresponding class labels y
- Value of K (number of neighbors)

**Output:** Predicted class labels for the test instances

**Begin**

**Pre-processing**

Perform any necessary data preprocessing steps such as feature scaling or normalization.

**Training** Store the training dataset

**Prediction** For each test instance in the dataset

Calculate the distance metric (e.g., Euclidean distance) between the test instance and all training instances.

Select the K instances with the shortest distances (nearest neighbors).

Determine the majority class among the K neighbors.

Assign the test instance to the majority class as the predicted class label.

**End**

## 2.5.3 Extreme Gradient Boosting (XGboost)

Is a powerful machine learning algorithm that is widely used for both regression and classification problems. It is an optimized implementation of the gradient boosting framework, which combines the predictions of multiple weak models (typically decision trees) to create a strong predictive model[78].

XGBoost excels in several key aspects, making it a preferred choice for various tasks. It incorporates regularization techniques to mitigate overfitting, pruning methods to optimize the model's structure, and gradient-based optimization algorithms to enhance training efficiency. Additionally, XGBoost supports parallel processing, enabling faster computations on large datasets[79].

XGBoost stands out for its powerful predictive capabilities, efficient computation, robustness against overfitting, handling of missing values, feature

importance analysis, and cross-validation support. These features make XGBoost an invaluable tool in machine learning for achieving accurate predictions and extracting insights from complex datasets[80].

Due to its superior performance, XGBoost has gained significant popularity in machine learning competitions and various real-world applications. Multiple programming languages, including Python, R, and Java, support it. The stages of XGBoost are detailed in Algorithm (2.7)[81]. Figure (2.4) The illustrates the progression of tree-based algorithms over time[82].

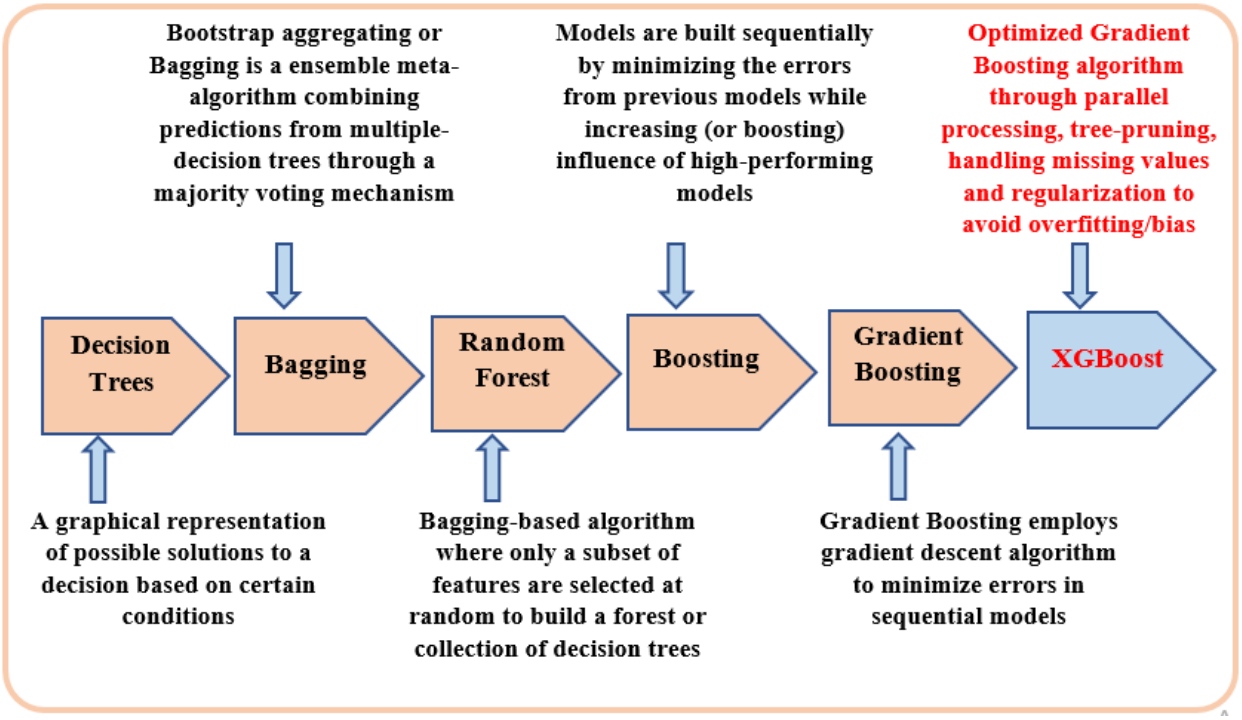



**Figure (2.4). Evolution of XGBoost Algorithm from Decision Trees**

| **Algorithm (2.7): XGBoost Algorithm** |
| --- |
| **Input:** Dataset<br>**Output:** Classified dataset |
| **Begin**<br>Generate an initial prediction and compute the residuals.<br>Construct an XGBoost tree.<br>Prune the tree.<br>Determine the output values for the tree's leaves.<br>Make new predictions using the updated model.<br>Calculate residuals based on the new predictions.<br>**End** |

## 2.6 Performance Evaluation

In this study, the classification performance was assessed using several evaluation metrics, including precision, recall, accuracy, and F1 score. These metrics are defined as follows[83]:

1. **Accuracy**: - is a metric that measures the proportion of correct predictions compared to the total number of instances evaluated. The following equation is used to calculate accuracy.

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (2.15)$$

2. **Precision**: is a metric that calculates the ratio of true positive predictions to the total number of positive predictions. It is computed using the following equation:

$$Precision = \frac{TP}{TP+FP} \quad (2.16)$$

3. **Recall**: is the ratio of correctly predicted positive instances to the total number of positive instances. The following equation is used to calculate recall:

$$Recall = \frac{TP}{TP+FN} \quad (2.17)$$

4. **F1-score**: The weighted harmonic mean, which takes into account both recall and precision, is used to calculate a single performance metric.

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (2.18)$$

"TP" (true positives) represents the number of positive images correctly classified by the classifier, while "TN" (true negatives) represents the number of negative images correctly classified. False negatives (FN) are negative images mistakenly classified as positive, whereas false positives (FP) are positive images mistakenly classified as negative.

# *Chapter Three*

## *The Proposed System*

## 3.1 Introduction

A machine learning and metaheuristic optimization approach for the identification of COVID-19 illness is presented. Two distinct chest X-ray datasets were utilized to ensure the generalization of the proposed methodology.

To enhance the discriminatory power of the extracted features, two well-established techniques, namely the Gray Level Co-occurrence Matrix (GLCM) and Local Binary Pattern (LBP), were employed. These techniques contribute to a more nuanced representation of image patterns, facilitating a more effective feature extraction process.

Feature selection, a critical step in optimizing classification performance and minimizing the number of features, was carried out using the Extra Tree and Swarm Optimization algorithms. These algorithms were chosen for their ability to efficiently navigate the feature space and identify the most relevant features for the task of COVID-19 illness identification.

In the final phase of the study, three diverse classifiers were employed to evaluate the classification performance of the proposed approach. Support Vector Machine (SVM), k-Nearest Neighbors (KNN), and eXtreme Gradient Boosting (XGBoost) were selected as they represent a range of classification methodologies, allowing for a comprehensive assessment of the model's robustness and adaptability.

The holistic system plan and workflow are visually represented in Figure (3.1), providing a clear overview of the integration of image processing, feature extraction, optimization, and classification stages in the proposed methodology for identifying COVID-19 illness. This comprehensive approach aims to contribute to the development of accurate and efficient diagnostic tools in the ongoing battle against the global pandemic.

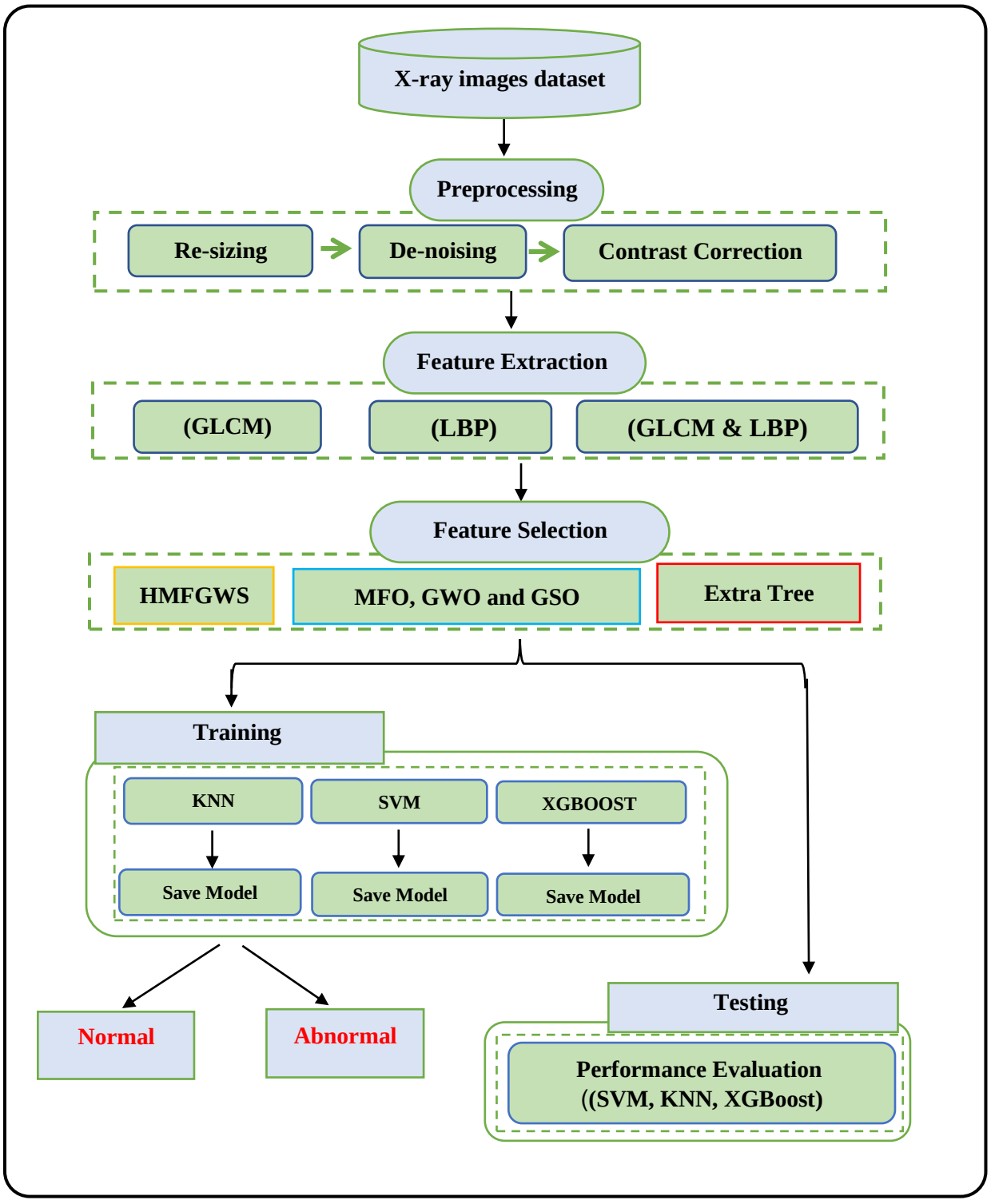



**Figure 3.1: General diagram of the proposed system**

## 3.2 Preprocessing

In the X-ray image, enhancement process is crucial for preparing the data and improving the classification accuracy ratio. Chest X-ray images frequently encounter challenges such as noise, suboptimal quality, and considerable size, primarily stemming from equipment limitations or photographic techniques. To tackle these issues, the suggested system incorporates three preprocessing steps, as detailed hereafter:

### 3.2.1 Re-sizing :

To reduce processing time, all X-ray images will be resized to a uniform size of (200 × 200). This resizing step is essential because larger images contain a higher number of pixels, which results in longer preprocessing times.

### 3.2.2 De-noising:

After resizing, the images undergo de-noising using a Gaussian blur filter. This filter effectively removes noise and unwanted details. Similar to the mean filter, the Gaussian filter is adept at eliminating both Gaussian and salt and pepper noise. It significantly contributes to detecting false edges caused by noise.

### 3.2.3 Contrast Correction:

Following the application of the Gaussian filter, the images are subjected to CLAHE to enhance contrast. CLAHE is a technique that adjusts image contrast based on image histograms. This step leads to obtaining resized images with improved clarity and higher contrast. As a result of these preprocessing steps, the X-ray images are transformed into clear, high-contrast versions. This enhancement positively impacts the feature extraction process, consequently increasing the overall classification accuracy. The preprocessing steps are visually represented in Figure (3.2) and further detailed in the algorithm (3.1).

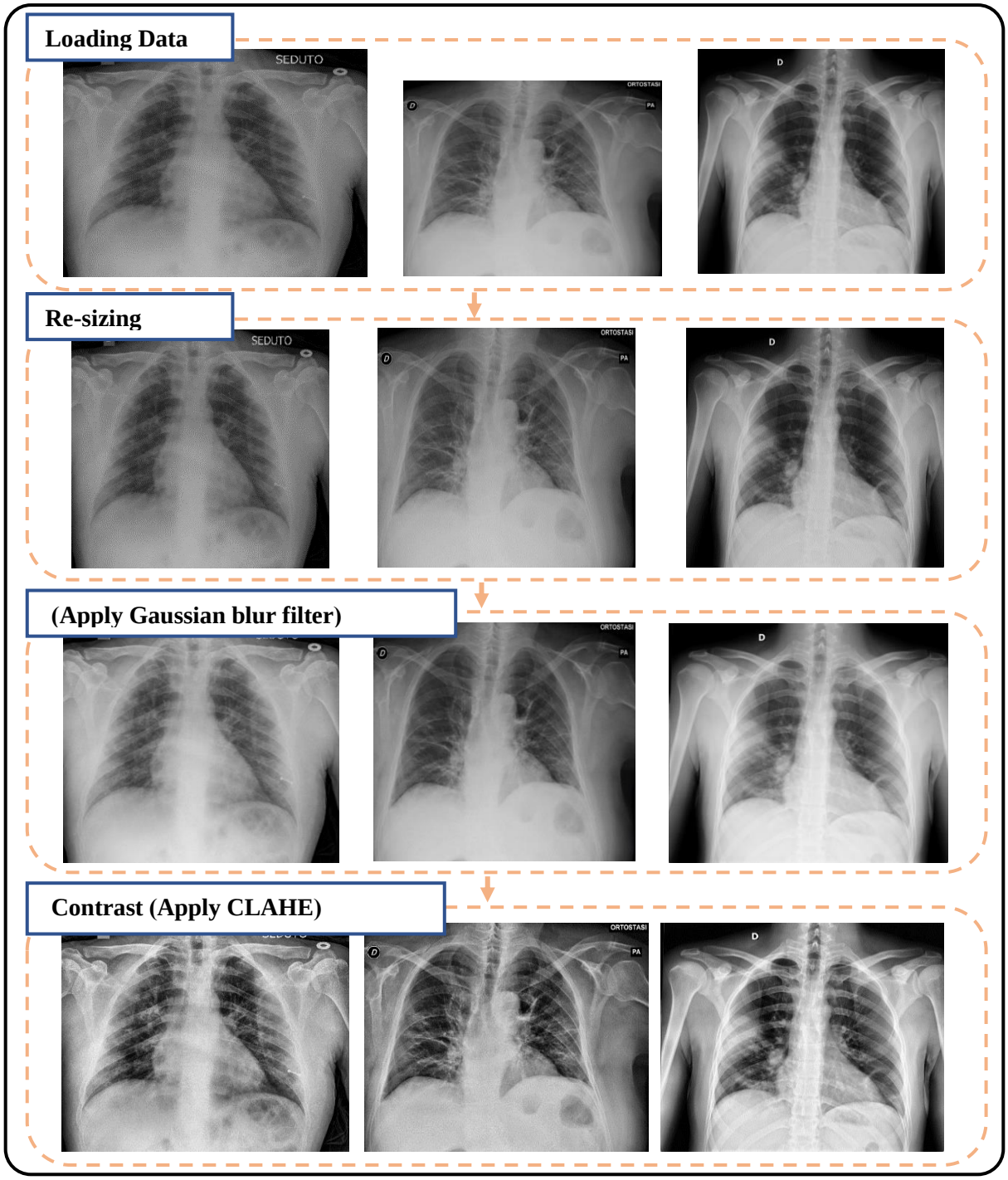



**Figure 3.2: Steps in the preprocessing stage.**

| **Algorithm (3.1): X-ray Image Preprocessing** |
|---|
| **Input**: chest X-ray image dataset with a size of (200, 200) pixels<br>**Output**: Enhanced images |
| **Begin**<br>**For** each image in the dataset<br>Resize the image to (200, 200) pixels and store it as Gimg[i].<br>Apply the Gaussian blur filter to Gimg[i].<br>Apply the CLAHE to Gimg[i] With<br>clipLimit=2.0 and tileGridSize=(8, 8).<br>Normalize Clahe[i] to enhance the image quality.<br>**End**<br>Return the set of enhanced images.<br>**End** |

## 3.3 Feature Extraction

Is a crucial step in image processing and computer vision tasks, where the goal is to capture relevant information from the raw image data to facilitate further analysis and decision-making.

Feature extraction is done by sequential feature extraction, Two commonly used techniques for feature extraction in image processing are (LBP) and (GLCM).

### 3.3.1 Gray Level Co-occurrence Matrix (GLCM):

The features of this technique and how it is calculated are discussed in depth in section (2.3.1). The GLCM algorithm is used on the original grayscale image. GLCM characterizes the spatial distribution of pixel intensity pairs within the image by considering a specific displacement between neighboring pixels.

The GLCM is a square matrix, and its elements represent the number of occurrences of each pixel intensity pair at the given displacement and orientation. From the GLCM, various statistical measures such as (ASM), variance, (COR), contrast, homogeneity, and energy can be computed. These measures capture different aspects of texture patterns, providing additional information about the texture characteristics of the image. The normalized image will be employed to

extract six GLCM features at four different orientation angles (0, 45, 90, and 135) from the X-ray images, resulting in 24 features.

### 3.3.2 Local Binary Pattern (LBP):

LBP operates on grayscale images and captures the local structure of an image. For each pixel in the image, LBP compares its intensity value with its neighboring pixels and generates a binary code based on the comparison results. This binary code is then converted to a decimal value, which represents the LBP value for the center pixel. The technique and its discussion are covered in depth in section (2.3.2).

The LBP algorithm is applied to the entire image, resulting in a new image called the LBP image, where each pixel contains its corresponding LBP value. This LBP image effectively encodes texture patterns, such as edges, corners, and spots.

The stages of Feature extraction are detailed in Algorithm (3.2).

**Algorithm (3.2): Feature extraction algorithm**

**Input**: Improved X-ray image
**Output**: Feature Vectors

```
Begin
    Image Reading
    Image Preprocessing with Algorithm (3.1)
   For i=1 to 6
        Calculate first-order feature[i] (image)
        Calculate GLCM [i] (image)
        For j=1 to 10
            Compute LBP [j] (image)
        End For
   End For
    Return (Features Vectors)
End
```

### 3.3.3 Feature Combination and Utilization:

After obtaining both the LBP image and the GLCM-based texture features, these two sets of features can be combined to create a more comprehensive feature vector for the image. The combination can be as simple as concatenating the LBP histogram with the GLCM-based statistical measures.

By sequentially applying LBP and GLCM, the resulting feature vector captures both local and global texture information, making it more robust and discriminative compared to using either LBP or GLCM alone. This approach leverages the complementary strengths of both techniques, enhancing the overall performance of texture-based image analysis tasks. Leading to improved representation and better performance in various computer vision applications. The feature extraction steps are visually represented in Figure (3.3) and further detailed in the algorithm (3.3) .

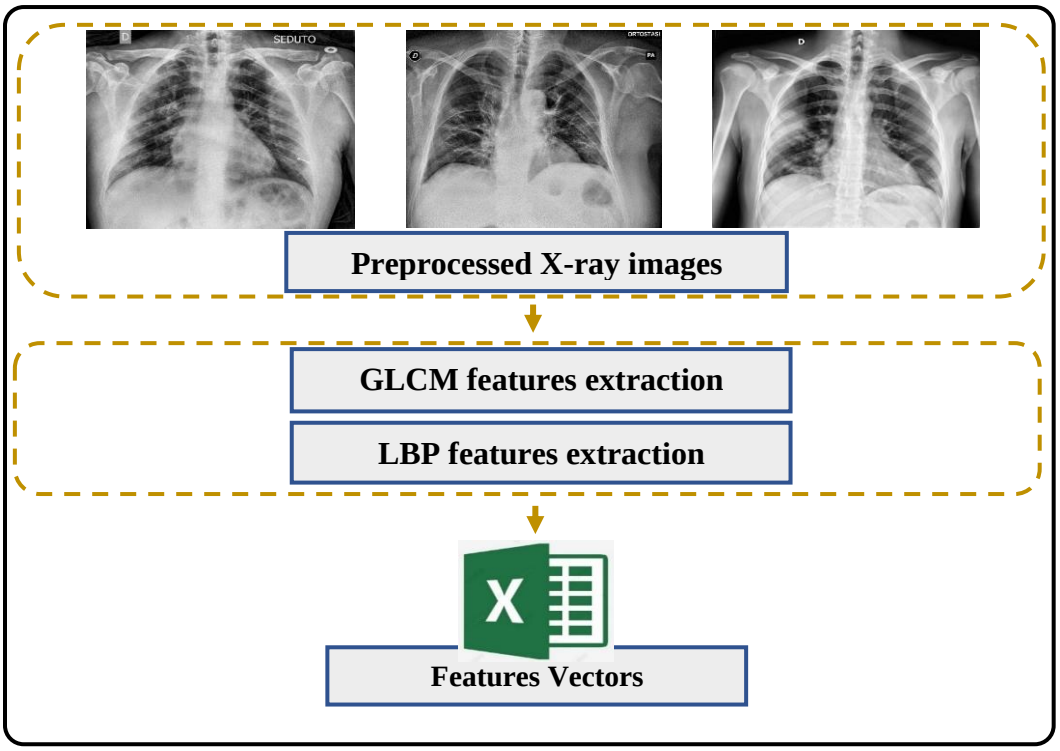



**Figure 3.3: Block diagram of feature extraction**

| **Algorithm (3.3): Proposed Feature Extraction Combination Algorithm** |
|---|
| **Input**: Enhanced images<br>**Output**: Features Vectors |
| **Begin**<br>Initialize an empty list to store the extracted features: features Vectors = []<br>**While** not at the end of the dataset<br>**{**<br>Execute feature extraction in parallel using algorithm (3.2).<br>}<br>**End while**<br>Combine the extracted features from all the images and store them in feature vectors<br>**Return** (features Vectors)<br>**End** |

## 3.4 Feature Selection

This thesis proposes two methods for feature selection. The first is based on the Extra Tree Feature Selection algorithm. The second is Swarm Optimization algorithms) based on Glowworm Swarm Optimization (GSO) algorithm, Moth Flame Optimization (MFO) algorithm, Gray Wolf Optimization (GWO) algorithm, Hybrid (Moth Flame Optimization (MFO) and Glowworm Swarm Optimization (GSO) algorithms) (HMFGWS). Figure (3.4) describes the general structure of the feature selection framework.

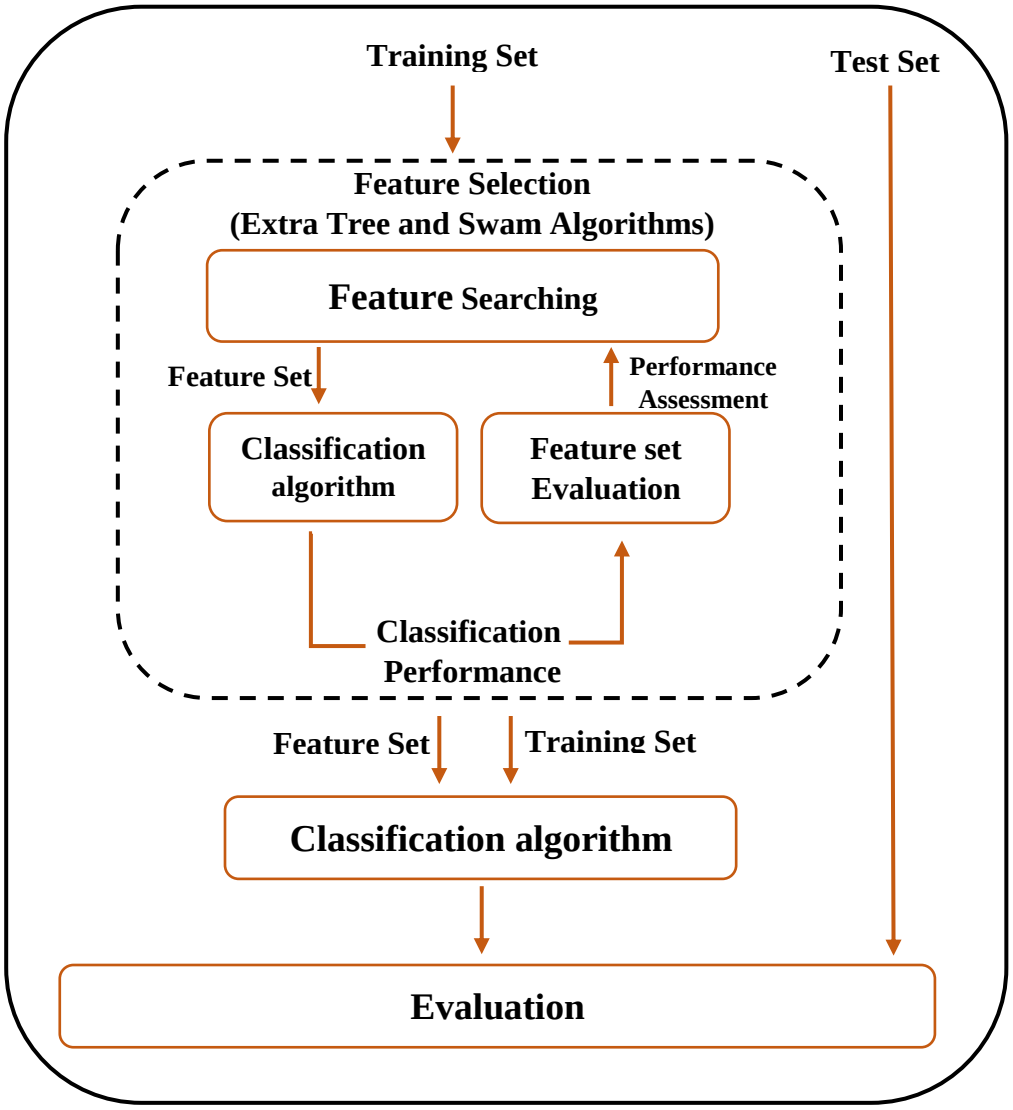



**Figure 3.4 Block diagram of feature selection.**

### 3.4.1 Modified Extra Trees Classifier Feature Selection

The Extra Trees are a useful ensemble learning technique for classification issues that may be used for the task of selecting relevant features. It is superior to both classic decision trees and Random Forests due to its focus on fixing the issues of overfitting in decision trees and the random nature of feature selection in the latter. Extra Trees builds numerous decision trees and uses an average feature importance score to identify which features are more significant. The technique and its discussion are covered in depth in section (2.4.1).

Overfitting is prevented and the variability of feature significance scores is minimized by randomly picking subsets of features at each split throughout the tree-building process. Finally, the average feature significance scores over all trees are used to get the final feature importance scores, with the most essential features being those that regularly receive high scores. The stages of The Extra Tree are detailed in Algorithm (3.4).

**Algorithm 3.4: Extra Tree Feature Selection**

**Input**:
- X: Training dataset features
- Y: Target variable
- n_trees: Number of trees to be used in the Extra Trees model
- m_features: Number of randomly selected features at each split
- scoring_metric: Scikit-learn metric to evaluate feature importance

**Output**: Sorted list of feature importances

```
Begin
   feature_importances = []
   For i in 1 to n_trees:
     - random_features = random_select_features(X.shape[1], m_features)
     -model = ExtraTreesClassifier()
     -model.fit(X[:,random_features], y)
     -importance_scores = calculate_feature_importances(model, scoring_metric)
     -Append importance_scores to feature_importances
   End For
   mean_importances  = calculate_mean_importances(feature_importances)
   std_importances = calculate_std_importances(feature_importances)
   feature_importances = [(mean_importances[i], std_importances[i])
       For i in range(X.shape[1])]
         sorted_features = sort_features_by_importance(feature_importances)
         Return sorted_features
       End For
End
```

*//The functions random_select_features, calculate_feature_importances, calculate_mean_importances, and calculate_std_importances . These functions are responsible for randomly selecting features, computing feature importances, calculating mean and standard deviation of importances, and sorting features, respectively.*

## 3.4.2 Modified Moth-Flame Optimization algorithm (MFO)

The Moth-Flame Optimization (MFO) algorithm, inspired by moths' behavior when attracted to flames and their unique navigation strategy, is of significant importance in the field of optimization. This algorithm offers a nature-inspired approach to solving complex problems and has practical applications in various domains. The algorithm and its discussion are covered in depth in section (2.4.2.2). MFO's key value lies in its ability to imitate the behavior of moths drawn to flames, allowing it to efficiently explore solution spaces and converge towards optimal or near-optimal solutions. By considering moths as candidate solutions and their orientation towards a source of light as a symbol of the optimal solution, MFO effectively tackles a wide range of optimization challenges.

Furthermore, the incorporation of attraction to brighter regions, resembling moths' natural behavior, ensures that the algorithm converges towards promising regions within the search space. This adaptability and the ability to mimic nature's optimization strategies make MFO a valuable tool for addressing real-world optimization problems across diverse fields. Algorithm (3.5) outlines the stages of MFO.

**Algorithm(3.5) : Moth-Flame Optimization**

**Input**: The objective function f(X) that the Moth-Flame Optimization algorithm aims to optimize is a fitness function that evaluates the quality of a given solution represented by a set of variables X = (X1, X2, ..., Xd), Number of moths in the population (N), dimension (d), Maximum iteration (Maximumiter), Flame number (N.FM), and b are the inputs;
**Output**: optimum ecological choice with the lowest fitness function value

```
Begin
  For  i = 1: N
    For  j = 1:  d
        Produce solutions for N-organisms using equation,
        Xi, j (i = 1, 2... N).
          //This step initializes the positions of moths in the search space by assigning random
          values within the lower bound (LB(i)) and upper bound (UB(i)) for each dimension).
```

```
            X ( i, j) = LB (i) + (UB  (i) -LB(i))* rand();
            //(X (i, j) represents the position of the ith moth in the jth dimension. rand ()
            generates a random number between 0 and 1).
      End  For
    End  For
   //fitness value f(X)  calculated for each moth in the population:
    While Currentiter < Maximumiter +1
         if Iteration== 1
                Enter N. FM = N in initial population
         else
                Employ using
                N.FM=round(N.FMlastiter-Currentiter (N.FMlastiter-1)/Maxiter)
         end if
          FM = Fitness Function f(x)؛
              if Iteration == 1
                     arrange the moths according to FM
                     Update Fmi
                     Iteration = 0؛
              else
                      Moths according to FM from last iteration.
                      Update Fmi
              end if
                    for j = 1: N
                        for  k = 1 : d
                           Find  r and  t using
                           r =-1+ Current, iter (-1/Maxiter) & t = ( r - 1 )x k + 1
                           Update Moths' location in relation to their specific flame.
                         end for
                    end for
                    Current iter = Current iter + 1؛
     end while
End
```

### 3.4.3 Modified Grey Wolf Optimizer Algorithm (GWO)

Inspired by natural processes, GWO is an innovative metaheuristic algorithm that uses a population-based approach. It has been used recently with great success in a number of difficult optimization problems. This method is simple and effective, it models its behavior after that of a certain subspecies of wolves and is hence named for them. The algorithm and its discussion are covered in depth in section (2.4.2.3). Wolves combine their collective intelligence and physical prowess to hunt prey animals and defeat their enemies through cooperative attacks. Several studies have verified that this algorithm can outperform and reveal improvements. Through these behaviors and the population-based approach, the GWO algorithm efficiently explores the solution space, effectively balancing between exploration and exploitation. The stages of GWO are detailed in Algorithm (3.6).

**Algorithm(3.6): The Grey Wolf Optimization**

**Inputs**: the values for N and T.
**Outputs**: "Xα"

**Begin**
Initialize the population of search agents (grey wolves) [Xi: i = 1, 2... N].
Set the maximum number of iterations as K. Initialize t = 1.
Calculate the fitness value for each search agent [g(Xi): i = 1, 2, ..., N].
Determine the best, second-best, and third-best search agents:
Xα = Arg(min) [g(Xi): i = 1, 2, ..., N]
Xβ = Arg(min) [g(Xi): i = 1, 2, ..., N, i ≠ α]
Xδ = Arg(min) [g(Xi): i = 1, 2, ..., N, i ≠ α, β]
Set i = 1 and let a = 2(1 - t/K).
Repeat for j = 1, 2, 3:
Generate random vectors r1j and r2j.
Calculate: Aj = a(2 * r1j - 1), Cj = 2 * r2j.
Calculate distance measures:
Dα = |C1 * Xα - Xi|
Dβ = |C2 * Xβ - Xi|
Dδ = |C3 * Xδ - Xi|
Update the search agents:
X1 = Xα - A(1) * Dα
X2 = Xβ - A(2) * Dβ

X3 = Xδ - A(3) * Dδ
Xi = (X1 + X2 + X3) / 3 for i = 1 to N.
If i < N, increment i by 1, and go to the step where we repeat for j.
If t < K, increment t by 1, and go to the step where we determine the best agents.
End the procedure using Xα as the optimal solution.
**End**

The objective function being minimized is denoted as g (Xi). This function represents the fitness value or performance measure associated with each individual search agent (grey wolf) represented by Xi in the population. Xα (X Alpha) This variable represents the search agent (grey wolf) with the best fitness value (lowest g (Xi)) among all the individuals in the population. In other words, Xα corresponds to the alpha (α) wolf, which is the global best solution found so far in the optimization process. Xβ (X Beta) represents the search agent (grey wolf) with the second-best fitness value (second lowest g (Xi)) in the population. It is the individual with the second-highest fitness level after Xα. The Xβ corresponds to the beta (β) wolf, which helps in exploration and is used to diversify the search space. Xδ (X Delta) is the search agent (grey wolf) with the third-best fitness value (third lowest g (Xi)) among the population, excluding Xα and Xβ. It is the individual with the third-highest fitness level and is used to further diversify the search space. Xδ corresponds to the delta (δ) wolf.

### 3.4.4 Modified Glowworm Swarm Optimization algorithm (GSO)

The algorithm takes inspiration from the behavior of glowworms, which emit light to attract mates and establish spatial awareness in their environment. In the GSO algorithm, a swarm of virtual glowworms represents the candidate solutions to an optimization problem. Each glowworm corresponds to a potential solution and has a position in the search space. The algorithm and its discussion are covered in depth in section (2.4.2.1).

The GSO algorithm follows a population-based approach where each glowworm iteratively adjusts its position based on its own behavior and the behavior of its neighboring glowworms. The behavior of the glowworms is governed by two main factors: their brightness and their movement. The brightness of a glowworm represents the quality of its current solution, and it affects the visibility of the glowworm to its neighbors. The movement of a glowworm determines its exploration or exploitation behavior.

The stages of GSO are detailed in Algorithm (3.7).

**Algorithm(3.7) : The Glowworm Swarm Optimization**

**Input**:
- Number of Glowworms (n)
- Initial Luciferin Values (I) for each glowworm
- Destination (D)
- Decay Factor (p)
- Gain Factor (γ)
- Maximum Iterations (max_iterations)
- Termination Criteria (e.g., a threshold for luciferin or maximum iterations)

**Output**:
- Updated Luciferin Values (li(t+1)) for all glowworms
- Probability Matrix (Pij(t)) for next-hop selection
- Movement of Glowworms
- Data Transmission to neighbors with highest luciferin (if D is reached)

**Begin**

Generate a certain number of glowworms.
Make I the starting value for each glowworm's luciferin.
Direct the glowworm towards destination D.
**For** each glowworm ($1 \leq i \leq n$) upon reaching a vehicle :
Calculate Di (t), pi (t).
*//Use the equation: to update the luciferin.*
li ( t +1 ) is equal to ( 1-p )1i ( t ) + γJ ( xi ( t +1 ) ) .
// *Choose the next-hop based on the probability equation:*
Pij ( t ) is equal to 1j ( t ) - 1i ( t ) / ∑ k ∈ Ni ( t ) 1k ( t ) - 1i ( t ) .
Move the glowworm i to the next hop.
**Repeat** steps 4 to 8 for all glowworms.
Node S transmits data to the neighbor with the highest luciferin.
**If** D be the receiving node:

```
                The message to save the data.
                Exit.
            Else
                Send the data to your neighbor who has the most luciferin.
            End if
        End for
End
```

Di (t) the distance measure from glowworm i to the destination D at iteration t. This measure helps the glowworms move towards the destination.pi (t) a measure of the glow intensity of glowworm i at iteration t. It indicates how bright the glowworm appears to its neighbors, influencing their movement.

(li (t+1)The updated luciferin value for glowworm i at iteration t+1. It is calculated based on the previous luciferin value (li (t)), a decay factor (p), and a gain factor (γ) multiplied by the local neighborhood measure (J (xi (t+1))).

J (xi (t+1)) the light intensity of the location xi (t+1) where glowworm i intends to move to. It indicates how favorable that location is in terms of finding a better solution).

(Pij (t): The probability that glowworm i chooses glowworm j as its next-hop neighbor at iteration t. It is determined based on the difference between their luciferin levels (1j (t) - 1i (t)) and the sum of such differences within the neighborhood of glowworm i).

### 3.4.5 Hybrid Moth Flame Optimization (MFO) and Glowworm Swarm Optimization (GSO) algorithms) (HMFGWS)

The HMFGWS algorithm is a hybrid approach that combines elements from MFO, GSO. This combination aims to leverage the strengths of each individual algorithm to enhance the optimization process. Hybrid algorithms often aim to improve convergence speed, solution quality, or both by exploiting the complementary characteristics of different algorithms. The stages of HMFGWS are detailed in Algorithm (3.8).

***Benefits of Hybrid Moth Flame Glowworm Swarm (HMFGWS) Algorithm:***

1. Combined Strengths: By integrating the MFO and GSO components, the HMFGWS algorithm can leverage the strengths of both algorithms. MFO's attraction to light sources and GSO's neighbor-based attraction can lead to efficient exploration and exploitation of the solution space.
2. Enhanced Exploration and Exploitation: The hybrid nature of HMFGWS can potentially strike a balance between global exploration (covering the entire search space) and local exploitation (refining solutions around promising regions).
3. Improved Convergence Speed: Combining multiple optimization techniques can lead to faster convergence towards optimal or near-optimal solutions compared to using each algorithm individually.
4. Robustness: The HMFGWS algorithm may exhibit increased robustness, as it can mitigate the weaknesses of one algorithm with the strengths of the other. This adaptability can make the algorithm more suitable for a wider range of optimization problems.

***Challenges of Hybrid Moth Flame Glowworm Swarm (HMFGWS) Algorithm:***

1. Complexity and Tuning: Combining different optimization techniques introduces additional complexity, including parameter tuning. Finding the right combination of parameters for both MFO and GSO components can be challenging and time-consuming.
2. Algorithm Interaction: Ensuring that the MFO and GSO components interact effectively can be tricky. Poorly integrated components might lead to suboptimal performance.
3. Computational Overhead: The hybrid approach may require more computational resources compared to standalone algorithms due to the combination of components and the interactions between them.
4. Algorithm Dependency: The success of the HMFGWS algorithm can be influenced by the performance of MFO and GSO individually. If one component performs poorly, it could impact the overall optimization process.
5. Problem Specificity: The performance of the HMFGWS algorithm might vary based on the characteristics of the optimization problem. The algorithm could excel on some problems while being less effective on others.

| **Algorithm (3.8) :Hybrid Moth Flame Glowworm Swarm Optimization algorithms (HMFGWS)** |
|---|
| **Input**: max_generations_HMFGWS = 100<br>population_size_HMFGWS = 50<br>**Output**:best_solution = selected_individuals[0]<br>best_fitness = best_solution.fitness |
| **Begin**<br>Define objective function<br>function objective function(x):<br>Implementation of the objective function<br>return f(x)<br>Initialize population for HMFGWS<br>population_HMFGWS = random_initialization(population_size_HMFGWS) |

```
    For generation in range(max_generations_HMFGWS)
        For individual in population_HMFGWS:
            Update position using MFO-inspired step
            update_position_using_mfo(individual)
            Update position using GSO-inspired step
            update_position_using_gso(individual)
            Update position using hybrid step
            update_position_using_hybrid(individual)
            Evaluate fitness
            Individual. Fitness = objective_function (individual. Position)
            Select top individuals for next generation
            population_HMFGWS. Sort(key=lambda ind: ind.fitness)
           selected_individuals =
population_HMFGWS[:population_size_HMFGWS]
            Hybrid Moth Flame Glowworm Swarm (HMFGWS) Algorithm
        End for
    End for
    Print ("Best Solution:", best_solution. Position)
    print("Best Fitness:", best_fitness)
End
```

Max_generations_HMFGWS represents the maximum number of generations the algorithm will run for, and population_size_HMFGWS represents the number of individuals (solutions) in each generation's population.

In the output section, the best solution from the final generation's selected individuals is identified, and its fitness value is stored. Objective function objective_function(x) that takes an input x (which represents a solution's position) and returns the value of the objective function f(x) to be minimized or maximized.

The initial population for the Hybrid Moth Flame Glowworm Swarm algorithm is initialized using a function called random_initialization. The size of the population is determined by population_size_HMFGWS.

# *Chapter Four*

## *Experimental Results And Discussion*

## 4.1 Introduction

The results of the proposed Covid-19 categorization system presented in the previous part will be explained and investigated in this chapter.

Two groups of chest X-ray images were used in the evaluation of the proposed method. To prevent overfitting during the training process, 80% of the samples were allocated for system training, while the remaining 20% were used for testing. The effectiveness of the system was assessed using four widely employed metrics in classification tasks: precision, recall, F1 score, and accuracy. Furthermore, this chapter will present a comparative analysis that includes some swarm algorithms with machine learning algorithms.

## 4.2 Software and Hardware Requirements

The suggested system functions through the utilization of an HP personal computer with specific features, including an Intel(R) Core(TM) i5-5200U processor clocked at 2.20GHz (with 4CPUs), 8GB of RAM running Windows 10, and a 64 bit OS. This system was operated using the Python programming language within the Anaconda platform.

## 4.3 Dataset

The training classification model utilizes two different datasets, each with its own description:

***1- The chest x-ray images pneumonia (DS1)***

The dataset contains a total of 800 CXR images, consisting of 400 images showing confirmed COVID-19 infections and 400 images representing the normal condition. The images are in PNG format and grayscale, with dimensions standardized to 200 × 200 pixels[84].

### *2- The extensive covid-19 X-ray (DS2)*

The dataset contains 4,044 CXR images showing confirmed cases of COVID-19 infections and 5,500 CXR representing normal conditions. These images are available in PNG, JPG, and JPEG formats and are grayscale. All the images have been resized to a uniform size of 200 × 200 pixels[85].

Figure (4.1) shows samples of the dataset.

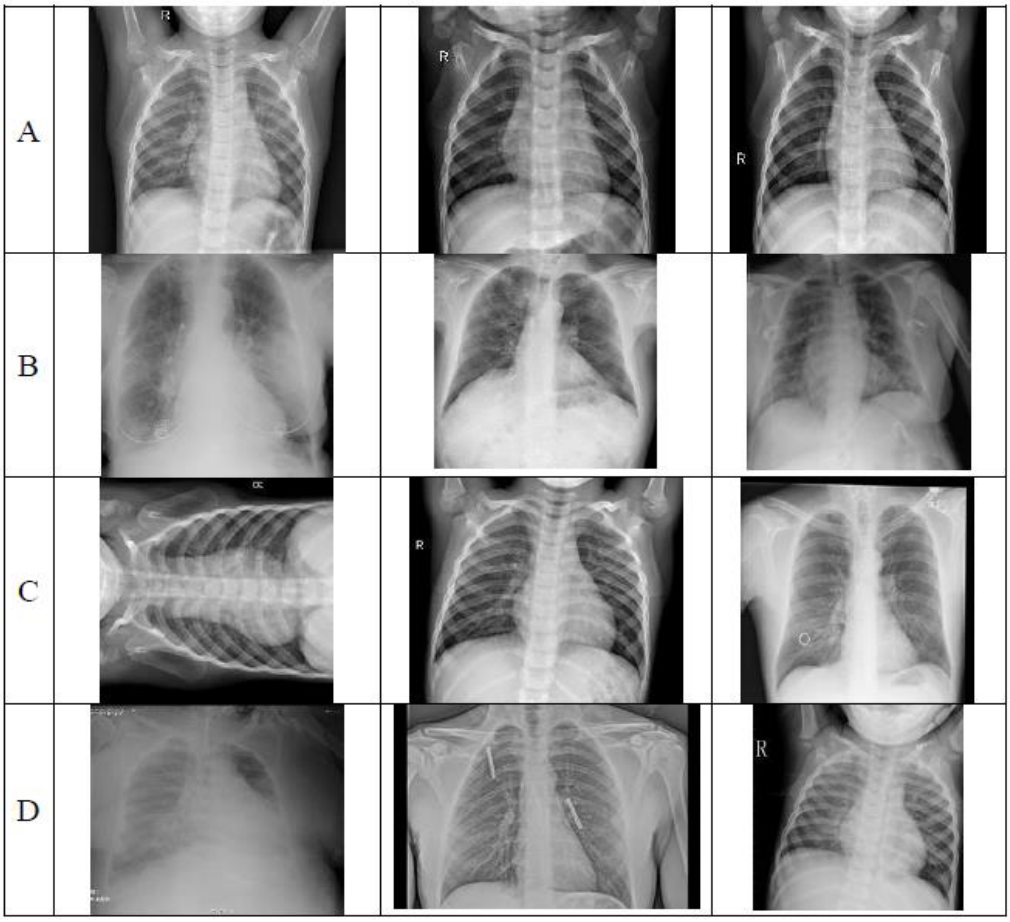


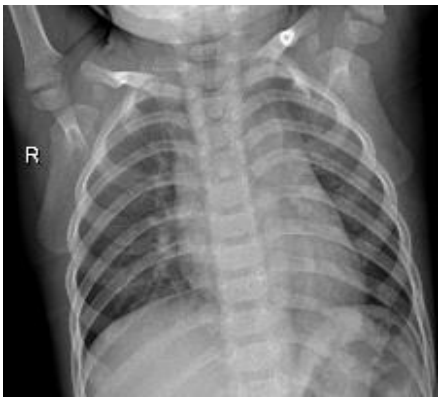

**Figure 4.1 illustrates samples from the dataset as follows:**
**(A) Normal CXR images from DS 1,**
**(B) Confirmed COVID-19 infections CXR images from DS 1,**
**(C) Normal CXR images from DS 2, and**
**(D) Confirmed COVID-19 case CXR images from DS 2.**

## 4.4 Preprocessing

The preprocessing operation consists of three sub-operations, which are outlined in the following steps:

### *1. Re-sizing:*

To reduce the processing time for the CXR images in the dataset, the images were re-sized to a dimension of 200 × 200 pixels. Figure (4.2) demonstrates the distinction between the original image and the re-sized version.

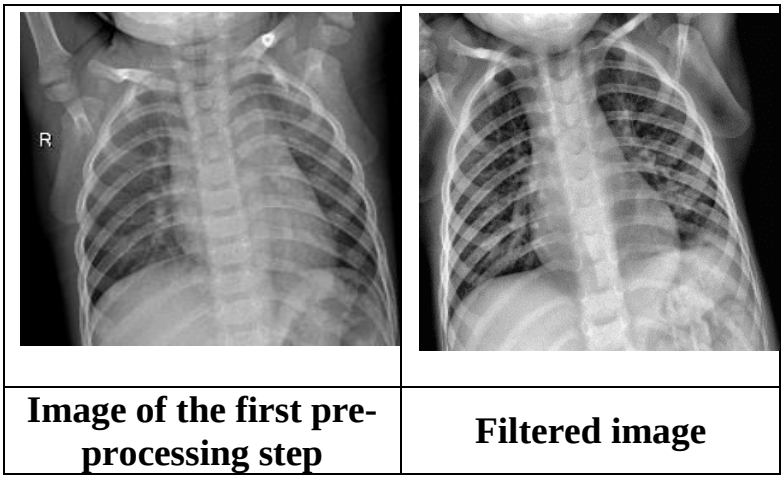


| Original image | Re-sized image |
|---|---|

**Figure 4.2: The resizing processing of dataset images.**

### *2. De-noising:*

The objective of this operator is to produce a visually smoother and blurrier image. To remove noise from the images, the Gaussian blur filter was utilized. Through experimentation, it was determined that this filter outperformed other uniform low-pass filters in terms of performance. Figure (4.3) illustrates the image both before and after the application of the filter.

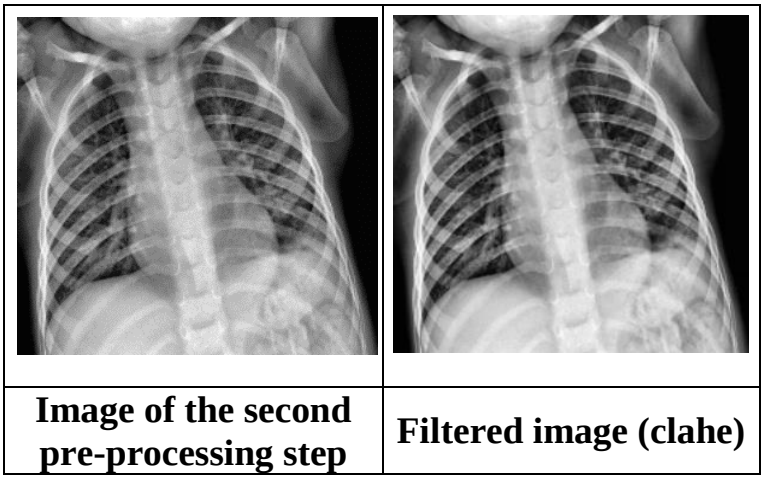


| Image of the first pre-processing step | Filtered image |
|---|---|

**Figure 4.3: Image before and after Filter image**

### *3. Contrast Correction*

For contrast correction, the CLAHE filter was utilized to enhance the image's contrast prior to the feature extraction stage. The image before and after the enhancing technique are shown in Figure (4.4).



| Image of the second pre-processing step | Filtered image (clahe) |
|---|---|

**Figure 4.4: Before and after CLAHE enhancement of the image**

## 4.5 Feature Extraction

The extraction of features is a crucial stage in any image classification system. It plays a vital role as the subsequent processes rely on its outcome. Consequently, determining the significant features to extract is a challenging task.

In the features extraction process, which was previously explained 24 features were extracted from the CXR images. Six first-order statistical GLCM features were used to extract the features for each of the four different orientations (0, 45, 90, and 135). The features were extracted using LBP's 10 first-order statistical features. Combining Local Binary Patterns (LBP) and Gray Level Co-occurrence Matrix (GLCM) methods can lead to more accurate texture classification, improve the classification of medical images.

The Gray Level Co-occurrence Matrix (GLCM) feature extraction approach has been extensively utilized in a number of texture research applications and is still efficient today. Texture properties were measured using six GLCM statics: angular second moment (ASM), variance, contrast, correlation (COR), homogeneity (HOM), and energy. A matrix that connects the distance and angle between pixels in an image is referred to as GLCM, using an internal connection between two image pixels that is determined by the values of grayscale that display the entire image information at every angle. In the Local Binary Pattern (LBP) feature extraction technique when representing an image for classification purposes, features are extracted in the form of numerical values. Chest X-ray images have texture because they contain recurring patterns that show how the chest images were created using the absorption of various spectrums dependent on the density of the tissue. To extract the necessary characteristics.

## 4.6 Feature Selection and Classification Results

In this section, an analysis and discussion will be conducted on the outcomes of evaluating the feature selection optimization problem using the extra tree algorithm and a set of swarm algorithms, which include the Moth-Flame Optimization algorithm (MFO), Grey Wolf Optimizer algorithm (GWO), Glowworm Swarm Optimization algorithm (GSO), and Hybrid Moth Flame Glowworm Swarm (HMFGWS) Algorithm.

To assess the effectiveness of algorithms employed for feature selection, three different classifiers were employed for classification purposes. The tables below showcase the outcomes of the feature selection algorithms when applied to three classifiers: KNN, SVM, and XGBoost.

Figures (4.5) (4.6) (4.7) shows Comparison between the proposed algorithms for Extra Tree, MFO, GWO, GSO, HMFGWS using SVM, KNN and XGBoost Classifier.

### 4.6.1 Basic Extra Tree Features Selection

In this section, tables (4.1), (4.2), and (4.3) show the performance of the extra tree used for feature selection. Three methods were used to extract the features: the first method (GLCM), the second method (LBP), and the third method (merging GLCM with LBP). The results indicated that, in DS1, the merge method outperformed three classifiers (KNN, SVM, and XGBoost) with accuracy rates of 98%, 95%, and 99%. In DS2, it achieved accuracy rates of 92%, 93%, and 91% when evaluated against the same three classifiers KNN, SVM, and XGBoost**.**

**Table 4.1: The Performance of Extra Tree after using just (GLCM), across three classifiers SVM, KNN, and XGBoost, using DS1, DS2**

| | classifier | Precision | Recall | F1-score | ACC |
|---|---|---|---|---|---|
| **DS1** | SVM | 0.95 | 0.94 | 0.95 | 0.9458 |
| | K-NN | 0.98 | 0.97 | 0.97 | 0.9750 |
| | XGBoost | 0.95 | 0.94 | 0.95 | 0.9841 |
| **DS2** | SVM | 0.92 | 0.93 | 0.92 | 0.9252 |
| | K-NN | 0.94 | 0.93 | 0.93 | 0.9328 |
| | XGBoost | 0.92 | 0.96 | 0.95 | 0.9554 |

Table 4.1 summarizes the Extra Tree algorithm's efficiency utilizing only GLCM features with three different classifiers (SVM, KNN, and XGBoost) on two datasets. The best accuracy was achieved, with a rate of 98.41% on DS1 using the XGBoost classifier and 95.54% on DS2 using the same classifier.

**Table 4.2: The Performance of Extra Tree after using just (LBP), across three classifiers SVM, KNN and XGBoost, using DS1, DS2**

| | classifier | Precision | Recall | F1-score | ACC |
|---|---|---|---|---|---|
| **DS1** | SVM | 0.96 | 0.96 | 0.96 | 0.9625 |
| | K-NN | 0.96 | 0.95 | 0.95 | 0.9562 |
| | XGBoost | 0.97 | 0.97 | 0.97 | 0.9791 |
| **DS2** | SVM | 0.83 | 0.85 | 0.85 | 0.8531 |
| | K-NN | 0.83 | 0.78 | 0.80 | 0.8300 |
| | XGBoost | 0.95 | 0.87 | 0.87 | 0.9144 |

Table 4.2 summarizes the Extra Tree algorithm's efficiency utilizing only LBP features with three different classifiers (SVM, KNN, and XGBoost) on two datasets. The best accuracy was achieved, with a rate of 0.9791% on DS1 using the XGBoost classifier and 0.9144% on DS2 using the same classifier.

**Table 4.3: The Performance of Extra Tree after using (GLCM with LBP), across three classifiers SVM, KNN and XGBoost, using DS1, DS2**

| | classifier | Precision | Recall | F1-score | ACC |
|---|---|---|---|---|---|
| **DS1** | SVM | 0.93 | 0.96 | 0.95 | 0.95 |
| | K-NN | 0.98 | 0.99 | 0.98 | 0.987 |
| | XGBoost | 0.99 | 0.99 | 1.00 | **0.9958** |
| **DS2** | SVM | 0.91 | 0.89 | 0.94 | 0.93 |
| | K-NN | 0.92 | 0.91 | 0.91 | 0.9171 |
| | XGBoost | 0.96 | 0.96 | 0.94 | **0.9601** |

Table 4.3 summarizes the Extra Tree algorithm's efficiency utilizing (GLCM with LBP), features with three different classifiers (SVM, KNN, and XGBoost) on two datasets. The best accuracy was achieved, with a rate of 0.9958% on DS1 using the XGBoost classifier and 0.9601% on DS2 using the same classifier.

### 4.6.2 MFO, GWO, GSO and HMFGWS Features Selection

When selecting the features in Swarm Optimization algorithms, three methods were used to extract the features, the first method (GLCM), the second (LBP) and the third (merging GLCM with LBP). The results showed that the merge method is the best. As shown in Tables (4.4), (4.5) (4.6), (4.7) (4.8), (4.9) (4.10), (4.11) (4.12), (4.13) (4.14), (4.15).

**Table 4.4: The Performance of GWO after using just (GLCM), across three classifiers SVM, KNN and XGBoost, using DS1, DS2**

| | classifier | Precision | Recall | F1-score | ACC |
|---|---|---|---|---|---|
| **DS1** | SVM | 0.98 | 0.86 | 0.92 | 0.9166 |
| | K-NN | 0.96 | 0.93 | 0.96 | 0.9562 |
| | XGBoost | 0.93 | 0.98 | 0.96 | 0.9870 |
| **DS2** | SVM | 0.80 | 0.80 | 0.83 | 0.8087 |
| | K-NN | 0.84 | 0.76 | 0.84 | 0.8100 |
| | XGBoost | 0.83 | 0.86 | 0.85 | 0.8977 |

Table 4.4 summarizes the GWO algorithm's efficiency utilizing only GLCM features with three different classifiers (SVM, KNN, and XGBoost) on two datasets. The best accuracy was achieved, with a rate of 0.9870% on DS1 using the XGBoost classifier and 0.8977% on DS2 using the same classifier.

**Table 4.5: The Performance of GWO after using just (LBP), across three classifiers SVM, KNN and XGBoost, using DS1, DS2**

| | classifier | Precision | Recall | F1-score | ACC |
|---|---|---|---|---|---|
| **DS1** | SVM | 0.93 | 0.96 | 0.95 | 0.95 |
| | K-NN | 0.98 | 0.98 | 0.98 | 0.981 |
| | XGBoost | 0.99 | 0.99 | 0.99 | 0.9937 |
| **DS2** | SVM | 0.90 | 0.86 | 0.91 | 0.8916 |
| | K-NN | 0.88 | 0.87 | 0.89 | 0.8767 |
| | XGBoost | 0.95 | 0.93 | 0.94 | 0.9465 |

Table 4.5 summarizes the GWO algorithm's efficiency utilizing only LBP features with three different classifiers (SVM, KNN, and XGBoost) on two datasets. The best accuracy was achieved, with a rate of 0.9937% on DS1 using the XGBoost classifier and 0.9465% on DS2 using the same classifier.

**Table 4.6: The Performance of GWO after using just (GLCM with LBP), across three classifiers SVM, KNN and XGBoost, using DS1, DS2**

| | classifier | Precision | Recall | F1-score | ACC |
|---|---|---|---|---|---|
| **DS1** | SVM | 0.84 | 0.98 | 0.91 | 0.9083 |
| | K-NN | 0.98 | 0.99 | 0.99 | 0.9900 |
| | XGBoost | 0.99 | 0.98 | 0.99 | **0.9984** |
| **DS2** | SVM | 0.90 | 0.89 | 0.91 | 0.8990 |
| | K-NN | 0.89 | 0.90 | 0.89 | 0.8965 |
| | XGBoost | 0.91 | 0.90 | 0.90 | **0.9593** |

Table 4.6 summarizes the GWO algorithm's efficiency utilizing (GLCM with LBP), features with three different classifiers (SVM, KNN, and XGBoost) on two datasets. The best accuracy was achieved, with a rate of 0.9984% on DS1 using the XGBoost classifier and 0.9593% on DS2 using the same classifier.

**Table 4.7: The Performance of GSO after using just (GLCM), across three classifiers SVM, KNN and XGBoost, using DS1, DS2**

| | classifier | Precision | Recall | F1-score | ACC |
|---|---|---|---|---|---|
| **DS1** | SVM | 0.85 | 0.79 | 0.78 | 0.7875 |
| | K-NN | 0.95 | 0.95 | 0.96 | 0.9500 |
| | XGBoost | 0.98 | 0.97 | 0.98 | 0.9800 |
| **DS2** | SVM | 0.84 | 0.78 | 0.85 | 0.8227 |
| | K-NN | 0.85 | 0.78 | 0.85 | 0.8227 |
| | XGBoost | 0.84 | 0.79 | 0.84 | 0.8964 |

Table 4.7 summarizes the GSO algorithm's efficiency utilizing only GLCM features with three different classifiers (SVM, KNN, and XGBoost) on two datasets. The best accuracy was achieved, with a rate of 0.9800% on DS1 using the XGBoost classifier and 0.8964% on DS2 using the same classifier.

**Table 4.8: The Performance of GSO after using just (LBP), across three classifiers SVM, KNN and XGBoost, using DS1, DS2**

| | classifier | Precision | Recall | F1-score | ACC |
|---|---|---|---|---|---|
| **DS1** | SVM | 0.92 | 0.96 | 0.94 | 0.9375 |
| | K-NN | 0.95 | 0.94 | 0.96 | 0.9562 |
| | XGBoost | 0.96 | 0.96 | 0.94 | 0.9743 |
| **DS2** | SVM | 0.91 | 0.87 | 0.92 | 0.9055 |
| | K-NN | 0.90 | 0.87 | 0.91 | 0.8930 |
| | XGBoost | 0.95 | 0.92 | 0.90 | 0.9539 |

Table 4.8 summarizes the GSO algorithm's efficiency utilizing only LBP features with three different classifiers (SVM, KNN, and XGBoost) on two datasets. The best accuracy was achieved, with a rate of 0.9743% on DS1 using the XGBoost classifier and 0.9539% on DS2 using the same classifier.

**Table 4.9: The Performance of GSO after using (GLCM with LBP), across three classifiers SVM, KNN and XGBoost, using DS1, DS2**

| | classifier | Precision | Recall | F1-score | ACC |
|---|---|---|---|---|---|
| **DS1** | SVM | 0.81 | 0.76 | 0.74 | 0.7625 |
| | K-NN | 0.97 | 0.98 | 0.99 | 0.9900 |
| | XGBoost | 0.97 | 0.98 | 0.97 | **0.9977** |
| **DS2** | SVM | 0.82 | 0.71 | 0.71 | 0.7555 |
| | K-NN | 0.92 | 0.94 | 0.93 | 0.9229 |
| | XGBoost | 0.96 | 0.93 | 0.95 | **0.9712** |

Table 4.9 summarizes the GSO algorithm's efficiency utilizing (GLCM with LBP), features with three different classifiers (SVM, KNN, and XGBoost) on two datasets. The best accuracy was achieved, with a rate of 0.9977% on DS1 using the XGBoost classifier and 0.9712% on DS2 using the same classifier.

**Table 4.10: The Performance of MFO after using just (GLCM), across three classifiers SVM, KNN and XGBoost, using DS1, DS2**

| | classifier | Precision | Recall | F1-score | ACC |
|---|---|---|---|---|---|
| **DS1** | SVM | 0.96 | 0.91 | 0.94 | 0.9375 |
| | K-NN | 0.96 | 0.97 | 0.96 | 0.9625 |
| | XGBoost | 0.99 | 0.96 | 0.95 | 0.9937 |
| **DS2** | SVM | 0.81 | 0.81 | 0.83 | 0.8129 |
| | K-NN | 0.80 | 0.79 | 0.80 | 0.8012 |
| | XGBoost | 0.84 | 0.85 | 0.84 | 0.8886 |

Table 4.10 summarizes the MFO algorithm's efficiency utilizing only GLCM features with three different classifiers (SVM, KNN, and XGBoost) on two datasets. The best accuracy was achieved, with a rate of 0.9937% on DS1 using the XGBoost classifier and 0.8886% on DS2 using the same classifier.

**Table 4.11: The Performance of MFO after using just (LBP), across three classifiers SVM, KNN and XGBoost, using DS1, DS2**

| | classifier | Precision | Recall | F1-score | ACC |
|---|---|---|---|---|---|
| **DS1** | SVM | 0.92 | 0.91 | 0.92 | 0.9166 |
| | K-NN | 0.93 | 0.93 | 0.94 | 0.9312 |
| | XGBoost | 0.97 | 0.97 | 0.98 | 0.9781 |
| **DS2** | SVM | 0.90 | 0.89 | 0.89 | 0.90 |
| | K-NN | 0.88 | 0.92 | 0.90 | 0.8851 |
| | XGBoost | 0.95 | 0.92 | 0.91 | 0.9526 |

Table 4.11 summarizes the MFO algorithm's efficiency utilizing only LBP features with three different classifiers (SVM, KNN, and XGBoost) on two datasets. The best accuracy was achieved, with a rate of 0.9781% on DS1 using the XGBoost classifier and 0.9526% on DS2 using the same classifier.

**Table 4.12: The Performance of MFO after using (GLCM with LBP), across three classifiers SVM, KNN and XGBoost, using DS1, DS2**

| | classifier | Precision | Recall | F1-score | ACC |
|---|---|---|---|---|---|
| **DS1** | SVM | 0.90 | 0.85 | 0.86 | 0.8666 |
| | K-NN | 0.97 | 0.99 | 0.98 | 0.9800 |
| | XGBoost | 0.96 | 0.94 | 0.96 | **0.9985** |
| **DS2** | SVM | 0.81 | 0.83 | 0.84 | 0.8200 |
| | K-NN | 0.93 | 0.92 | 0.92 | 0.9265 |
| | XGBoost | 0.93 | 0.94 | 0.91 | **0.9687** |

Table 4.12 summarizes the MFO algorithm's efficiency utilizing (GLCM with LBP), features with three different classifiers (SVM, KNN, and XGBoost) on two datasets. The best accuracy was achieved, with a rate of 0.9985% on DS1 using the XGBoost classifier and 0.9687% on DS2 using the same classifier**.**

**Table 4.13: The Performance of HMFGWS after using just (GLCM), across three classifiers SVM, KNN and XGBoost, using DS1, DS2**

| | classifier | Precision | Recall | F1-score | ACC |
|---|---|---|---|---|---|
| **DS1** | SVM | 0.96 | 0.91 | 0.94 | 0.9375 |
| | K-NN | 0.96 | 0.90 | 0.93 | 0.9312 |
| | XGBoost | 0.96 | 0.96 | 0.96 | 0.9625 |
| **DS2** | SVM | 0.80 | 0.80 | 0.83 | 0.8023 |
| | K-NN | 0.80 | 0.85 | 0.85 | 0.8300 |
| | XGBoost | 0.82 | 0.87 | 0.86 | 0.8426 |

Table 4.13 summarizes the HMFGWS algorithm's efficiency utilizing only GLCM features with three different classifiers (SVM, KNN, and XGBoost) on two datasets. The best accuracy was achieved, with a rate of 0.9625% on DS1 using the XGBoost classifier and 0.8426% on DS2 using the same classifier.

**Table 4.14: The Performance of HMFGWS after using just (LBP), across three classifiers SVM, KNN and XGBoost, using DS1, DS2**

| | classifier | Precision | Recall | F1-score | ACC |
|---|---|---|---|---|---|
| **DS1** | SVM | 0.99 | 0.97 | 0.98 | 0.9812 |
| | K-NN | 0.98 | 0.97 | 0.97 | 0.975 |
| | XGBoost | 0.99 | 0.97 | 0.98 | 0.9812 |
| **DS2** | SVM | 0.89 | 0.93 | 0.91 | 0.8935 |
| | K-NN | 0.89 | 0.88 | 0.88 | 0.8867 |
| | XGBoost | 0.90 | 0.94 | 0.92 | 0.9029 |

Table 4.14 summarizes the HMFGWS algorithm's efficiency utilizing only LBP features with three different classifiers (SVM, KNN, and XGBoost) on two datasets. The best accuracy was achieved, with a rate of 0.9812% on DS1 using the XGBoost classifier and 0.9029% on DS2 using the same classifier.

**Table 4.15: The Performance of HMFGWS after using (GLCM with LBP), across three classifiers SVM, KNN and XGBoost, using DS1, DS2**

| | classifier | Precision | Recall | F1-score | ACC |
|---|---|---|---|---|---|
| **DS1** | SVM | 0.99 | 0.99 | 0.99 | 0.9801 |
| | K-NN | 0.98 | 0.98 | 0.97 | 0.9750 |
| | XGBoost | 0.98 | 0.98 | 0.98 | **0.99** |
| **DS2** | SVM | 0.92 | 0.91 | 0.92 | 0.9176 |
| | K-NN | 0.92 | 0.91 | 0.92 | 0.9202 |
| | XGBoost | 0.93 | 0.92 | 0.93 | **0.9725** |

Table 4.15 summarizes the HMFGWS algorithm's efficiency utilizing (GLCM with LBP), features with three different classifiers (SVM, KNN, and XGBoost) on two datasets. The best accuracy was achieved, with a rate of 0.99% on DS1 using the XGBoost classifier and 0.9725% on DS2 using the same classifier**.**

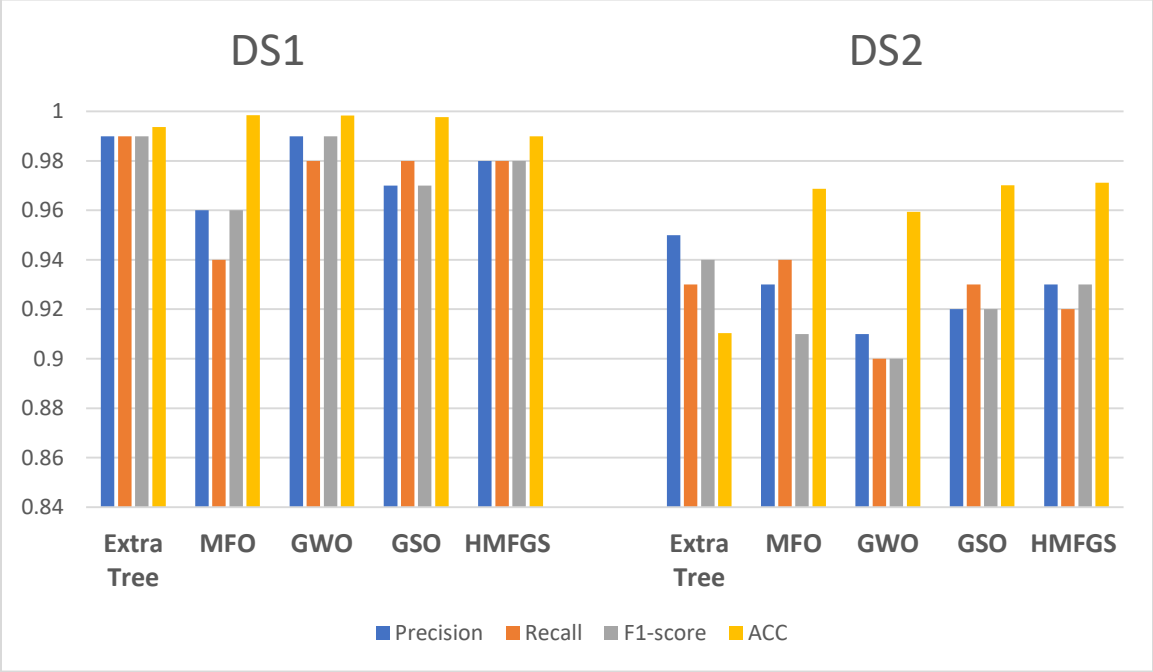



**Figure 4.5: Comparison between Extra tree, MFO, GWO, GSO and HMFGWS proposed algorithms using XGboost Classifier.**

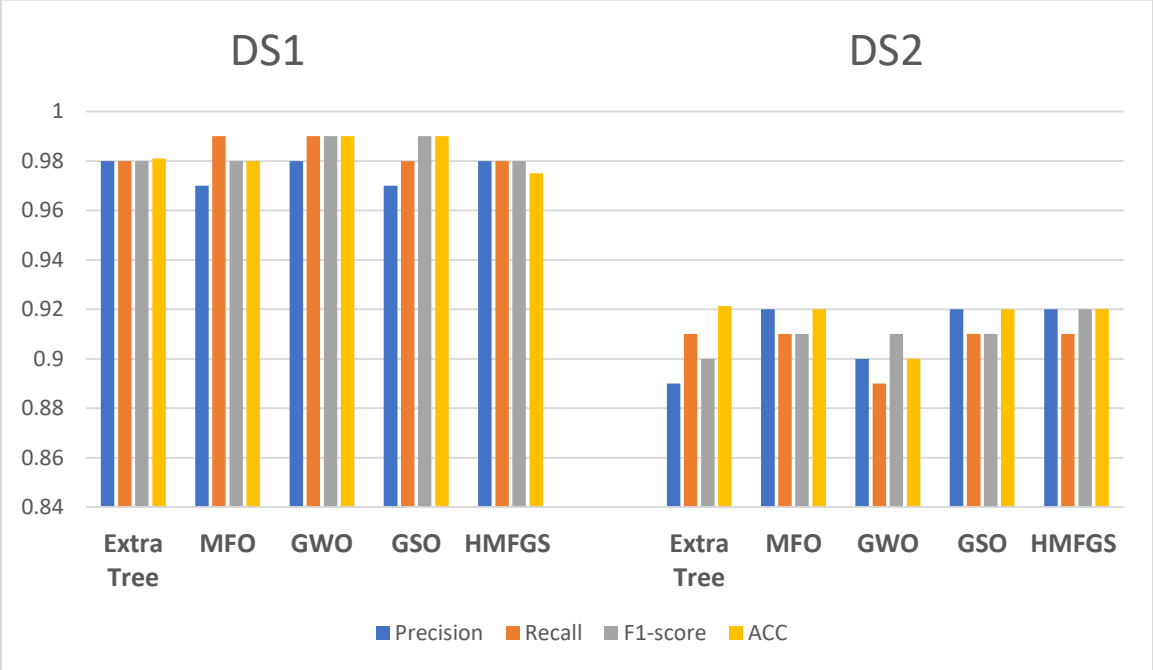



**Figure 4.6: Comparison between Extra tree, MFO, GWO, GSO and HMFGWS proposed algorithms using KNN Classifier**

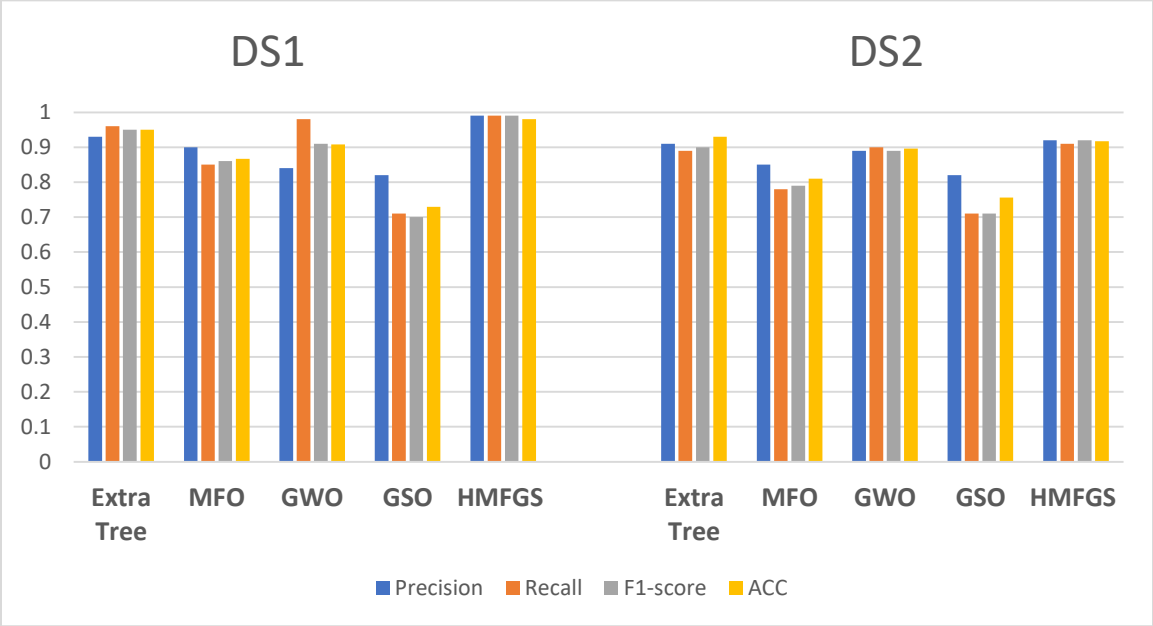



**Figure 4.7: Comparison between Extra tree, MFO, GWO, GSO and HMFGWS proposed algorithms using SVM Classifier**

**Table 4.16: the total results over three classifiers SVM, KNN and XGBoost, using DS1, DS2**

| | Algorithms | Dataset | XGBoost | KNN | SVM |
|---|---|---|---|---|---|
| 1 | Extra Tree | DS1 | 0.9958% | 0.987% | 0.95% |
| | | DS2 | 0.9601% | 0.9171% | 0.93% |
| 2 | GWO | DS1 | 0.9984% | 0.99% | 0.9083% |
| | | DS2 | 0.9593% | 0.8965% | 0.8990% |
| 3 | GSO | DS1 | 0.9977% | 0.99% | 0.7625% |
| | | DS2 | 0.9712% | 0.9229% | 0.7555% |
| 4 | MFO | DS1 | 0.9985% | 0.9800% | 0.8666% |
| | | DS2 | 0.9687% | 0.9265% | 0.8200% |
| 5 | HMFGWS | DS1 | 0.99% | 0.9750% | 0.9801% |
| | | DS2 | 0.9725% | 0.9202% | 0.9176% |

## 4.7 Comparison with previous works

The table below indicates that the suggested system outperforms a group of earlier studies that created the classification system using the same techniques and data set.

**Table 4.17: Comparison with previous works using the same datasets and algorithms**

| Comparison with previous works using the same datasets | | | | | |
|---|---|---|---|---|---|
| **No.** | **Ref & Year** | **Algorithms** | **Dataset** | **Classifiers** | **Accuracy** |
| 1 | [86] 2022 | Application of CycleGAN and Inception V3 | 5500 normal and 4044 COVID-19 X-ray chest scans. | Deep learning | 92% |
| 2 | [12] 2022 | (HHO), (SSA), (WOA), and (GWO) | 9,000 2D X-ray images, comprising two distinct categories: 5,500 images depicting healthy lungs and 4,044 images showing the X-rays of patients diagnosed with COVID-19. | KNN, SVM, XGBoost | HHO (94%,89%,94%) SSA (96%, 80%94%) WOA (96%, 82%92%) GWO (94%, 89%,96%) |
| 3 | [15] 2022 | Hybrid-binary version of the Harris-Hawks algorithm Optimization (HHO) and Salp-Swarm Optimization (SSA) (HHOSSA) | 800 CXR images(400 CXR images covid-19 and 400 CXR images of normal) | SVM, KNN and XGBoost | 96%,98%and 98% |
| **Comparison with previous works using the same algorithms** | | | | | |
| 4 | [47] 2020 | moth flame optimization (MFO) algorithm | improve the classification tasks in medical applications (Twenty-three medical data sets from UCI, Keel, Kaggle) | KNN Classifier | 83% ,75% and 70% |
| 5 | [13] 2022 | The extra-trees classifier was used to remove irrelevant features, while SVM was utilized to diagnose the breast cancer status | The dataset was collected from the Gynecology Department of the University Hospital Centre of Coimbra (CHUC). The Coimbra Breast Cancer Dataset (CBCD) comprises 64 female individuals diagnosed with breast cancer and 52 individuals who are in good .health | SVM Classifier | 80.23% |

| | | | | | |
|---|---|---|---|---|---|
| 6 | [14] 2022 | glowworm swarm optimization (GSO) with an inception-based (IDCNN) ,called the GSO-IDCNN model. | A collection of 220 images related to COVID-19, along with 27 images representing normal cases, 15 images depicting pneumocystis cases, and 11 images representing SARS cases | Deep learning | 94% |
| **Proposed** | **2023** | **Extra Tree & Swarm Optimization algorithms** | **DS1** | **SVM** | **0.9801%** |
| | | | | **KNN** | **0.99%** |
| | | | | **XGBoost** | **0.9985%** |
| | | | **DS2** | **SVM** | **0.93%** |
| | | | | **KNN** | **0.9265%** |
| | | | | **XGBoost** | **0.9725%** |

The table (4.17) shows that the proposed work achieves higher accuracy compared to previous works for both DS1 and DS2 datasets, using Extra Tree and Swarm Optimization algorithms with different classifiers.

# *Chapter Five*

## *Conclusions and Future works*

## 5.1 Conclusions

In this thesis, the extra tree method and swarm algorithms like MFO, GWO, GSO, and HMFGWS were introduced to address the challenge of feature selection. The following points outline the key discoveries and conclusions drawn from the experimental outcomes.

1. The performance of classification system was enhanced through choosing significant and accurate features in the medical images, and the results were clear from the tables reported previously.
2. Using multiple technique features vector is based on two technologies. GLCM in four different directions along with LBP in the feature vector contributes to the model's robustness, as it encompasses a wide range of image variations, ensuring comprehensive discrimination capabilities for image type classification.
3. Using the Extra Tree algorithm, for feature selection, indicate a good performance, as shows in Table (4.3).
4. Using the MFO, GWO, GSO algorithm, which are the Swarm optimization algorithms, for feature selection, and tall provided accurate results with slight differences, as shows in Table (4.6), (4.9) and (4.12).
5. The findings of the experiment proved that the HMFGWS algorithm for feature selection and classification have outperformed performance than MFO and GSO, as shown in Table (4.15).
6. Three classifiers were used, namely SVM, KNN, and XGBoost. The best classification accuracy was achieved by XGBoost with medical images, as shown in Table (4.16).
7. The findings showed that the extensive covid-19 X-ray data group's highest level of accuracy came from the XGBoost of 97%, SVM techniques of 93%, whereas the KNN techniques recorded the lowest accuracy of 92%. In the chest x-ray images (pneumonia) data group's, XGBoost and KNN classifiers achieved the highest

accuracy rates up to 99%, while SVM technologies revealed the lowest accuracy rates at 98%.

## 5.2 Future Works

To enhance the performance of the suggested system for future works, the following strategies can be implemented:

1- Creating a real-time application for the suggested technology to be utilized in e-medical applications.

2- Using deep learning algorithms with huge dataset to improve feature extraction or classification step.

3- The system has the capability to categorize various lung diseases by extracting distinct features relevant to the specific ailment, such as Pneumonia.

4- It is possible in the future to use different swarm algorithms like fox optimization Algorithm, Ant Nesting Algorithm (ANA), and Donkey and Smuggler optimization Algorithm to enhance the process of feature selection.

# *References*

## *References*


[1] R. Najjar, "Redefining Radiology : A Review of Artificial Intelligence Integration in Medical Imaging," *diagnostics*, vol. 13, no. 2760, p. 25, 2023, doi: https://doi.org/10.3390/diagnostics13172760.

[2] R. Filip, R. G. Puscaselu, L. Anchidin-norocel, M. Dimian, and W. K. Savage, "Global Challenges to Public Health Care Systems during the COVID-19 Pandemic : A Review of Pandemic Measures and Problems," *J. Pers. Med. Rev.*, vol. 12, no. 1295, p. 22, 2022, doi: https://doi.org/ 10.3390/jpm12081295.

[3] A. S. Ahuja, "The impact of artificial intelligence in medicine on the future role of the physician," *PeerJ*, vol. 7, no. 7702, 2019, doi: 10.7717/peerj.7702.

[4] A. Heidari and N. J. Navimipour, *Machine learning applications for COVID-19 outbreak management*, vol. 34, no. 18. Springer London, 2022. doi: 10.1007/s00521-022-07424-w.

[5] E. Jordan, E. S. DELIA, L. SURBHI, and A. SHAPOUR, "Optimization in the Context of COVID-19 Prediction and Control : A Literature Review," *IEEE Access*, vol. 9, pp. 130072–130093, 2021, doi: 10.1109/ACCESS.2021.3113812.

[6] M. A. Eshraghi, A. Ayatollahi, and S. B. Shokouhi, "COV-MobNets : a mobile networks ensemble model for diagnosis of COVID-19 based on chest X-ray images," *BMC Med. Imaging*, vol. 23, no. 83, pp. 1–11, 2023, doi: https://doi.org/10.1186/s12880-023-01039-w.

[7] D. Kuzinkovas and S. Clement, "The Detection of COVID-19 in Chest X-rays Using Ensemble CNN Techniques," *Inf. is a Sci.*, vol. 14, no. 370, pp. 1–18, 2023, doi: https://doi.org/10.3390/info14070370.

[8] R. Sagban, H. A. Marhoon, and R. Alubady, "Hybrid bat-ant colony optimization algorithm for rule-based feature selection in health care," *Int. J. Electr. Comput. Eng.*, vol. 10, no. 6, pp. 6655–6663, 2020, doi: 10.11591/ijece.v10i6.pp6655-6663.

[9] L. G. Fahad, S. F. Tahir, W. Shahzad, M. Hassan, H. Alquhayz, and R. Hassan, "Ant colony optimization-based streaming feature selection: An application to the medical image diagnosis," *Sci. Program.*, vol. 2020, no. 1064934, p. 10, 2020, doi: 10.1155/2020/1064934.

[10] V. Dharmalingam and D. Kumar, "Hybrid feature selection model for classification of lung disorders," *J. Ambient Intell. Humaniz. Comput.*, vol.

13, no. 12, pp. 5609–5625, 2022, doi: 10.1007/s12652-021-03224-7.

[11] R. Al-Wajih, S. J. Abdulkadir, N. Aziz, Q. Al-Tashi, and N. Talpur, "Hybrid binary grey Wolf with Harris hawks optimizer for feature selection," *IEEE Access*, vol. 9, pp. 31662–31677, 2021, doi: 10.1109/ACCESS.2021.3060096.

[12] A. S. Issa, Y. H. Ali, and A. R. Tarik, "Comparative Analysis of Swarm Algorithms to Classification of covid19 on X-Rays," *2022 Int. Conf. Data Sci. Intell. Comput.*, vol. 22883321, no. Icdsic, pp. 164–169, 2022, doi: 10.1109/ICDSIC56987.2022.10075733.

[13] G. Alfian, S. Muhammad, F. Imam, L. F. 3 Norma, and T. Fransiskus, "Predicting Breast Cancer from Risk Factors Using SVM and Extra-Trees-Based Feature Selection Method," *Computers*, vol. 11, no. 9, p. 14, 2022, doi: 10.3390/computers11090136.

[14] I. Abunadi, A. A. Albraikan, J. S. Alzahrani, and M. M. Eltahir, "An Automated Glowworm Swarm Optimization with an Inception-Based Deep Convolutional Neural Network for COVID-19 Diagnosis and Classification," *Healthc.*, vol. 10, no. 4, 2022, doi: 10.3390/healthcare10040697.

[15] A. S. Issa, Y. H. Ali, and T. A. Rashid, "An Efficient Hybrid Classification Approach for COVID-19 Based on Harris Hawks Optimization and Salp Swarm Optimization," *Int. J. online Biomed. Eng.*, vol. 18, no. 13, pp. 113–130, 2022, doi: 10.3991/ijoe.v18i13.33195.

[16] Riana, M. Mazdadi, I. Budiman, and M. Herteno, "IMPLEMENTATION OF INFORMATION GAIN AND PARTICLE SWARM OPTIMIZATION ON SENTIMENT ANALYSIS OF COVID-19 HANDLING USING K-NN," *(Jurnal Inform. dan Komput.*, vol. 6, no. 1, pp. 7–12, 2023, doi: 10.33387/jiko.v6i1.5260.

[17] F. A. Eltawil, M. Atalla, E. Boulos, A. Amirabadi, and P. N. Tyrrell, "Analyzing Barriers and Enablers for the Acceptance of Artificial Intelligence Innovations into Radiology Practice : A Scoping Review," *Tomogr. is an Int. Sci.*, vol. 9, pp. 1443–1455, 2023.

[18] B. Chen, H. Chen, and M. Li, "Feature Selection Based on BP Neural Network and Adaptive Particle Swarm Algorithm," *Mob. Inf. Syst.*, vol. 6715564, p. 11, 2021, doi: 10.1155/2021/6715564.

[19] I. M. EL-Hasnony, M. Elhoseny, and Z. Tarek, "A hybrid feature selection model based on butterfly optimization algorithm: COVID-19 as a case study,"

*Expert Syst.*, vol. 39, no. 3, p. e12786, 2022, doi: https://doi.org/10.1111/exsy.12786.

[20] B. Goyal, A. Dogra, S. Agrawal, and B. S. Sohi, "Noise issues prevailing in various types of medical images," *Biomed. Pharmacol. J.*, vol. 11, no. 3, pp. 1227–1237, 2018, doi: 10.13005/bpj/1484.

[21] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, 2014, doi: 10.1016/j.advengsoft.2013.12.007.

[22] N. H. Barnouti, "Improve Face Recognition Rate Using Different Image Pre-Processing Techniques," *Am. J. Eng. Res.*, vol. 5, no. 4, pp. 46–53, 2016.

[23] M. Basu, "Gaussian-based edge-detection methods - A survey," *IEEE Trans. Syst. Man, Cybern. Part C (Applications Rev.*, vol. 32, no. 3, pp. 252–260, Sep. 2002, doi: 10.1109/TSMCC.2002.804448.

[24] K. C. Prabu Shankar and S. Prayla Shyry, "A Survey of image pre-processing techniques for medical images," *J. Phys. Conf. Ser.*, vol. 1911, no. 1, pp. 1–6, 2021, doi: 10.1088/1742-6596/1911/1/012003.

[25] Y. Zhang and F. Ma, "Application and Optimization of Image Fuzzy Control Algorithm based on Gaussian Blur in TensorFlow Training," in *3rd International Conference on Mechatronics Engineering and Information Technology*, 2019, pp. 854–860.

[26] P. Characterization, M. Using, and O. Coherence, *Plaque characterization methods using optical coherence tomography*. 2017. doi: 10.1016/B978-0-12-804734-7.00005-1.

[27] T. Rahman *et al.*, "Exploring the effect of image enhancement techniques on COVID-19 detection using chest X-ray images," *Comput. Biol. Med.*, vol. 132, no. November 2020, p. 104319, 2021, doi: 10.1016/j.compbiomed.2021.104319.

[28] W. Nural, M. Mustafa, R. Umar, A. Zulkeflee, E. Awalludin, and A. Nazhatulshima, "Enhancing Moon Crescent Visibility Using Contrast-Limited Adaptive Histogram Equalization and Bilateral Filtering Techniques," *J. Telecommun. Inf. Technol.*, no. 1, pp. 3–13, 2022.

[29] P. Singh, "Feature Enhancement in Medical Ultrasound Videos Using Contrast-Limited Adaptive Histogram Equalization," *J. Digit. Imaging*, vol. 33, pp. 273–285, 2020.

[30] A. Subasi, *Feature Extraction and Dimension Reduction*. 2019. doi:

10.1016/b978-0-12-817444-9.00004-0.

[31] R. Hassan, Z. Sultani, and B. Dhannoon, “Content-Based Image Retrieval System using Color Moment and Bag of Visual Words with Local Binary Pattern,” *Karbala Int. J. Mod. Sci.*, vol. 9, no. 1, p. 14, 2023.

[32] J. Zhang, G. Li, and S. He, “Texture-Based Image Retrieval by Edge Detection Matching GLCM,” in *2008 10th IEEE International Conference on High Performance Computing and Communications*, Sep. 2008, pp. 782–786. doi: 10.1109/HPCC.2008.55.

[33] M. Yang *et al.*, “A Survey of Shape Feature Extraction Techniques,” in *Pattern Recognition*, 2010, pp. 43–90.

[34] Y. Wu, “Edge Detection of Depth Image Based on Contour Curve,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 563, no. 5, 2019, doi: 10.1088/1757-899X/563/5/052037.

[35] K. Nyein Nyein Hlaing, “First Order Statistics and Glcm Based Feature Extraction for Recognition of Myanmar Paper Currency,” in *The IIER International Conference*, 2015, pp. 1–6.

[36] W. Zhong, J. Chen, B. Tian, and Y. Xie, “The research of color sorting algorithm based on gray level co - Occurrence matrix,” in *Proceedings of 2013 2nd International Conference on Measurement, Information and Control, ICMIC 2013*, 2013, pp. 926–930. doi: 10.1109/MIC.2013.6758111.

[37] YU Jian, “Texture Image Segmentation Based on Gaussian Mixture Models and Gray Level Co-occurrence Matrix,” in *2010 Third International Symposium on Information Science and Engineering*, 2010, p. 4. doi: 10.1109/ISISE.2010.9.

[38] B. Kaddar, H. Fizazi, and A. Boudraa, “Texture features based on an efficient local binary pattern descriptor,” *Comput. Electr. Eng.*, vol. 70, no. August 2018, pp. 1–13, 2017, doi: 10.1016/j.compeleceng.2017.08.009.

[39] A. K. Bedi and R. K. Sunkaria, “Mean distance local binary pattern: a novel technique for color and texture image retrieval for liver ultrasound images,” *Multimed. Tools Appl.*, vol. 3, no. 80, p. 30, 2021, doi: 10.1007/s11042-021-10758-7.

[40] L. S. López, “Local Binary Patterns applied to Face Detection and Recognition,” Signal Theory & Communication Department, 2010.

[41] D. Huang, C. Shan, M. Ardabilian, Y. Wang, and L. Chen, “Local Binary

Patterns and Its Application to Facial Image Analysis : A Survey," *IEEE Trans. Syst.*, vol. 41, no. 6, p. 18, 2017.

[42] M. A. Berbar, "Features extraction using encoded local binary pattern for detection and grading diabetic retinopathy," *Heal. Inf. Sci. Syst.*, vol. 10, no. 1, pp. 1–13, 2022, doi: 10.1007/s13755-022-00181-z.

[43] A. A. Suleiman, "Image Classification Based on Enhancement of Local Binary Pattern," Middle East University, 2018.

[44] R. Nosaka, Y. Ohkawa, and K. Fukui, "Feature Extraction Based on Co-occurrence of Adjacent Local Binary Patterns," in *SpringerLink*, 2011, pp. 82–91.

[45] E. Odhiambo Omuya, G. Onyango Okeyo, and M. Waema Kimwele, "Feature Selection for Classification using Principal Component Analysis and Information Gain," *Expert Syst. Appl.*, vol. 174, no. February, p. 114765, 2021, doi: 10.1016/j.eswa.2021.114765.

[46] M. Ghosh, R. Guha, R. Sarkar, and A. Abraham, "A wrapper-filter feature selection technique based on ant colony optimization," *Neural Comput. Appl.*, vol. 32, no. 12, pp. 7839–7857, 2020, doi: 10.1007/s00521-019-04171-3.

[47] R. Abu Khurmaa, I. Aljarah, and A. Sharieh, *An intelligent feature selection approach based on moth flame optimization for medical diagnosis*, vol. 33, no. 12. Springer London, 2021. doi: 10.1007/s00521-020-05483-5.

[48] C. Chen, Y. Tsai, F. Chang, and W. Lin, "Ensemble feature selection in medical datasets : Combining filter , wrapper , and embedded feature selection results," *wiley*, vol. 37, no. 5, pp. 1–10, 2020, doi: 10.1111/exsy.12553.

[49] A. Gupta, "Feature Selection Techniques in Machine Learning," *Analytics Vidhya*, 2023. https://www.analyticsvidhya.com/blog/2020/10/feature-selection-techniques-in-machine-learning/

[50] A. T. B. Chandrika, M. T. Student, M. Mandal, M. Mandal, and M. Mandal, "Android malware detection using extra trees classifier based feature selection and machine learning," *Int. J. techo-engineering*, vol. XIII, no. Iii, pp. 435–443, 2021.

[51] K. Thankachan, "What? When? How?: ExtraTrees Classifier," *Towards Data Science*, 2022. https://towardsdatascience.com/what-when-how-extratrees-classifier-c939f905851c

[52] S. Gupta and N. Goe, “Enhancement of Performance of K-Nearest Neighbors Classifiers for the Prediction of Diabetes Using Feature Selection Method,” 2020, pp. 681–686.

[53] M. R. Camana, S. Ahmed, C. E. Garcia, and I. Koo, “Extremely Randomized Trees-Based Scheme for Stealthy Cyber-Attack Detection in Smart Grid Networks,” vol. 8, no. Ml, p. 13, 2020, doi: 10.1109/ACCESS.2020.2968934.

[54] G. BATTINENI1, N. CHINTALAPUDI2, G. SAGARO3, and F. AMENTA, “REVIEW ANALYSIS ON IMPORTANCE OF SWARM,” *Int. J. Comput. Sci. Mob. Comput.*, vol. 8, no. 5, pp. 182–186, 2019.

[55] N. Berente, B. Gu, and J. Recker, “M ANAGING A RTIFICIAL I NTELLIGENCE 1,” vol. 45, no. 3, pp. 1433–1450, 2021, doi: 10.25300/MISQ/2021/16274.

[56] Yaniv Altshuler, “Recent Developments in the Theory and Applicability of Swarm Search,” *Entropy*, vol. 25, no. 5, p. 12, 2023.

[57] T. Kalaiselvi, P. Nagaraja, and Z. A. Basith, “A Review on Glowworm Swarm Optimization A Review on Glowworm Swarm Optimization,” *Int. J. Inf. Technol.*, vol. 3, no. 2, p. 8, 2017.

[58] K. Kaipa and D. Ghose, “Glowworm Swarm Optimisation: A New Method for Optimising Multi-Modal Functions,” *Int. J. Comput. Intell. Stud.*, vol. 1, Jan. 2009, doi: 10.1504/IJCISTUDIES.2009.515637.

[59] Z. Tang and Y. Zhou, “A Glowworm Swarm Optimization Algorithm for Uninhabited Combat Air Vehicle Path Planning,” vol. 24, no. 1, pp. 69–83, 2015, doi: 10.1515/jisys-2013-0066.

[60] V. B. Meyer-rochow, “Glowworms: A review of Arachnocampa spp. and kin,” vol. 22, no. 3, p. 16, 2017, doi: https://doi.org/10.1002/bio.955.

[61] N. Zainal, A. Zain, N. Radzi, and A. Udin, “Glowworm Swarm Optimization (GSO) Algorithm for Optimization Problems: A State-of-the-Art Review,” *Appl. Mech. Mater.*, vol. 421, pp. 507–511, Apr. 2013, doi: 10.4028/www.scientific.net/AMM.421.507.

[62] S. Mirjalili, “Moth-Flame Optimization Algorithm : A Novel Nature-inspired Heuristic Paradigm,” *KNOWLEDGE-BASED Syst.*, no. July, 2015, doi: 10.1016/j.knosys.2015.07.006.

[63] S. H. Hashemi Mehne and S. Mirjalili, “Moth-Flame Optimization Algorithm: Theory, Literature Review, and Application in Optimal Nonlinear

Feedback Control Design: Methods and Applications," in *Nature-Inspired Optimizers*, 2020, pp. 143–166. doi: 10.1007/978-3-030-12127-3_9.

[64] S. K. Sahoo and A. K. Saha, "A Modernized Moth Flame Optimization Algorithm for Higher Dimensional Problems," in *INTERNATIONAL CONFERENCE ON SUSTAINABLE ENGINEERING AND TECHNOLOGY*, 2022, pp. 9–20.

[65] B. Kumar, Saroj.Apu, Sahoo.aha, Kumar.Ezugwu, Absalom E.Agushaka, Jeffrey O.Abuhaija, *Moth Flame Optimization : Theory , Modifications , Hybridizations , and Applications*, vol. 30, no. 1. Springer Netherlands, 2023. doi: 10.1007/s11831-022-09801-z.

[66] N. Mittal, U. Singh, and B. S. Sohi, "Modified Grey Wolf Optimizer for Global Engineering Optimization," *Appl. Comput. Intell. Soft Comput.*, vol. 2016, no. 7950348, p. 17, 2016.

[67] P. Niu, S. Niu, and L. Chang, "Knowledge-Based Systems The defect of the Grey Wolf optimization algorithm and its verification method," *Knowledge-Based Syst.*, vol. 171, pp. 37–43, 2019, doi: 10.1016/j.knosys.2019.01.018.

[68] Y. Kang, Z. Xu, H. Wang, Y. Yuan, X. Yang, and K. Pu, "An Improved Gray Wolf Optimization Algorithm with a Novel Initialization Method for Community Detection," *MDPI Journals*, vol. 10, no. 20, pp. 1–21, 2022.

[69] By and M. M. A. Abdulgader, "Bio Inspired Evolutionary Fuzzy System for Data Classification," 2019.

[70] S. S. Nath, J. Kar, G. Mishra, S. Chakraborty, and N. Dey, "A Survey of Image Classification Methods and Techniques," in *International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)in 2014*, 2014, pp. 554–557.

[71] George Henry Lewes and M. A. & R. Khanna, "Support Vector Machines for Classification," in *Efficient Learning Machines*, 2015, pp. 39–66.

[72] M. Peixeiro, "The Complete Guide to Support Vector Machine (SVM)," *Towards Data Science*, 2019. https://towardsdatascience.com/the-complete-guide-to-support-vector-machine-svm-f1a820d8af0b

[73] T. Do, "Parallel multiclass stochastic gradient descent algorithms for classifying million images with very-high-dimensional," *Vietnam J Comput Sci*, vol. 1, no. 2014, pp. 107–115, 2014, doi: 10.1007/s40595-013-0013-2.

[74] K. Taunk, S. De, S. Verma, and A. Swetapadma, "A brief review of nearest

neighbor algorithm for learning and classification," *2019 Int. Conf. Intell. Comput. Control Syst. ICCS 2019*, no. January, pp. 1255–1260, 2019.

[75] A. H. Khaleel, G. A. Al-suhail, and B. M. Hussan, "A Weighted Voting of K-Nearest Neighbor Algorithm for Diabetes Mellitus," vol. 6, no. 1, pp. 43–51, 2017.

[76] A. Gallego, J. Calvo-zaragoza, J. J. Valero-mas, and J. R. Rico-juan, "Clustering-based k -nearest neighbor classification for large-scale data with neural codes representation," vol. 74, pp. 531–543, 2018, doi: 10.1016/j.patcog.2017.09.038.

[77] M. E. Syed, "Attribute weighting in K-nearest neighbor classification," 2014.

[78] Z. A. Ali, Z. H. Abduljabbar, H. A. Taher, A. B. Sallow, and S. M. Almufti, "Exploring the Power of eXtreme Gradient Boosting Algorithm in Machine Learning : a Review," *Acad. J. Nawroz Univ.*, vol. 12, no. 2, pp. 320–15, 2023, doi: 10.25007/ajnu.v12n2a1612.

[79] A. V. Tokmak, A. Akbulut, and C. Catal, "Boosting the visibility of services in microservice architecture," *Cluster Comput.*, vol. 5, no. 18 September, pp. 1–13, 2023, doi: 10.1007/s10586-023-04132-5.

[80] Z. Ergul and Z. Kamisli Ozturk, "Performance Analysis of XGBoost Classifier with Missing Data," in *In 2021 The 1st International Conference on Computing and Machine Intelligence ,ICMI 2021*, 2021.

[81] N. Memon, S. Patel, and D. Patel, "Comparative Analysis of Artificial Neural Network and XGBoost Algorithm for PolSAR Image Classification," 2019, pp. 452–460. doi: 10.1007/978-3-030-34869-4_49.

[82] "https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d @ towardsdatascience.com." [Online]. Available: https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d

[83] P. K. Mall, P. K. Singh, and D. Yadav, "GLCM Based Feature Extraction and Medical X- RAY Image Classification using Machine Learning Techniques," *2019 IEEE Conf. Inf. Commun. Technol.*, vol. 19533775, no. April, pp. 1–6, 2019.

[84] Kaggle, "Chest X-Ray Images (Pneumonia) | Kaggle," *Kaggle's chest X-ray images (Pneumonia) dataset*, 2020. https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia

[85] W. El-Shafai and F. E. Abd El-Samie, “Extensive COVID-19 X-Ray and CT Chest Images Dataset,” *Mendeley Data*, 2020, [Online]. Available: https://data.mendeley.com/datasets/8h65ywd2jr/3

[86] G. Bargshady, X. Zhou, P. D. Barua, R. Gururajan, Y. Li, and U. R. Acharya, “Application of CycleGAN and transfer learning techniques for automated detection of COVID-19 using X-ray images,” *Pattern Recognit. Lett.*, vol. 153, pp. 67–74, 2022, doi: 10.1016/j.patrec.2021.11.020.

## Publications

1. Paper entitled “Comparative Analysis of MFO, GWO and GSO for Classification of Covid-19 Chest X-Ray Images” published in Baghdad Science Journal (BSJ) Publisher: College of Science for Women/ University of Baghdad, Scopus Q2, Clarivate, 2023.
   https://bsj.uobaghdad.edu.iq/index.php/BSJ/article/view/9236

2. Paper entitled “Classification of COVID-19 Disease Based on Extra Tree Features Selection “accepted in in Iraqi Journal of Science (IJS) (Publisher: University of Baghdad, Scopus Q3, 2023.

# الخلاصة


تعد معالجة الصور الطبية مجالًا ذا أهمية بالغة في العلوم الطبية، حيث يؤثر بشكل كبير على تطبيقات التصوير الطبي. يعمل هذا المجال على تعزيز دقة وفعالية التشخيص، وتسهيل تخطيط العلاج، وتمكين مراقبة تطور المرض، وذلك من خلال الاستفادة الكاملة من قوة تحليل الصور الرقمية.

يشير تفشي فيروس كورونا(COVID-19)، الناتج عن فيروسSARS-CoV-2 ، إلى الإصابة بالمتلازمة التنفسية الحادة الخطيرة. ظهرت العدوى في البداية في ووهان في نهاية عام 2019، وبسبب هذا التفشي، تطور مرض كوفيد-19 إلى جائحة فشكل تهديدًا كبيرًا للحياة البشرية وسبب اضطرابًا في الاقتصاد. أثبتت صور الأشعة السينية للصدر أهميتها في مراقبة تأثير كوفيد-19 على أنسجة الرئة.

في هذه الدراسة، تم اقتراح نظام لتصنيف فيروس كورونا (COVID-19) باستخدام صور الأشعة السينية للصدر. من أجل تعزيز أداء النظام وتوسيع تنوع البيانات، يتم استخدام مجموعتين من البيانات:
((Chest x-ray images (pneumonia)، تتألف من 800 صورة أشعة سينية للصدر وتسمى في الرسالة DS1. و( Extensive covid-19 X-ray) تتألف من 9000 صورة أشعة سينية للصدر، وقد تم تسميتها DS2. يتم تطبيق تقنيات معالجة مسبقة مثل تغيير الحجم، وتقليل الضوضاء، وتحسين التباين.

لكل من الاتجاهات الأربعة (0، 45، 90، و135)، يتم استخراج ست ميزات إحصائية باستخدام (GLCM). بالإضافة إلى ذلك، يتم استخدام (LBP) لاستخراج عشر ميزات إحصائية. الجمع بين أساليب LBP و GLCM لها القدرة على تعزيز دقة تصنيف الصور الطبية.

تقوم هذه الدراسة بتقديم نهج اختيار الميزات في حالتين باستخدام خوارزمية Extra Tree وخوارزميات Swarm، خوارزمية(MFO) ، خوارزمية(GWO) ، خوارزمية(GSO) ، والخوارزمية الهجينة (HMFGWS) وبعد اختيار الميزة، يتم استخدام ثلاثة مصنفات لتقييم الأداء: (SVM)،(KNN) ، و(XGBoost). تظهر النتائج أن الخوارزميات المقترحة تحقق أداءً متفوقًا في اختيار الميزات والتصنيف. عند استخدام DS1، تكون نسب الدقة 99٪ باستخدام KNN، و72٪ باستخدام SVM، و99٪ باستخدام XGBoost. عند استخدام DS2، تكون نسب الدقة 92٪ باستخدام KNN، و75٪ باستخدام SVM، و97٪ باستخدام XGBoost.

جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعة بابل
كلية العلوم للبنات
قسم علوم الحاسوب

# تصنيف كوفيد-19 باستخدام تعلم الآلة وخوارزميات تحسين ال Metaheuristics

رسالة مقدمة الى مجلس كلية العلوم للبنات في جامعة بابل وهي جزء من متطلبات الحصول على درجة الماجستير في علوم الحاسبات

مقدمة من قبل
اسراء محسن محمد

بأشراف
د. حسين عطية
د. يسرى حسين علي

1445 هـ

2023 م