REPUBLIC OF TURKEY
ALTINBAŞ UNIVERSITY
Institute of Graduate Studies
Graduate school of Science and Engineering
Electrical and Computer Engineering

WEIGHT OPTIMIZED CNN+MLP USING MODIFIED ENHANCED ARTIFICIAL ECOSYSTEM FOR COVID-19 DETECTION

Munaf Arab

A project submitted for the Master's Degree

Supervisor
Asst. Prof. Dr. Hakan Koyuncu

Istanbul, 2021

WEIGHT OPTIMIZED CNN+MLP USING MODIFIED ENHANCED ARTIFICIAL ECOSYSTEM FOR COVID-19 DETECTION

Arab,Munaf

Electrical and computer engineering

A project submitted for the Master's Degree

ALTINBAŞ UNIVERSITY

2021

I hereby declare that all information/data presented in this graduation project has been obtained in full accordance with academic rules and ethical conduct. I also declare all unoriginal materials and conclusions have been cited in the text and all references mentioned in the Reference List have been cited in the text, and vice versa as required by the abovementioned rules and conduct.

Munaf Arab

Signature

DEDICATION

This project is especially dedicated to Asst. Prof. Dr. Hakan Koyuncu who helped and guided me to successfully complete this project work. Also, I would like to dedicate this project to my dear mother, who has been a wonderful supporter until my research was completed.

WEIGHT OPTIMIZED CNN+MLP USING MODIFIED ENHANCED ARTIFICIAL ECOSYSTEM FOR COVID-19 DETECTION

Munaf Arab
A project submitted for the master's degree, Department, Altınbaş University,
Supervisor: Asst. Prof. Dr. Hakan KOYUNCU
Date: July/2021

ABSTRACT

Machine learning has been at the forefront of medical research efforts. It serves as a tool to assist researchers and practitioners in making an informed decision in the face of COVID-19 to forecast the spread of such diseases and identify the infected with much higher confidence. This study outlines the process of designing and refining an AI framework to classify COVID-19 from Pneumonia using X-ray scans combined with textual clinical data. The primary objective of this study is to merge multiple types of neural networks and study the effect of using metaheuristic algorithms in optimizing the weights of the neural network. The proposed framework also utilizes a lung segmentation process using a pre-trained ResNet34 model to generate a mask for each lung to eliminate the unnecessary features that might affect the result. The training data consists of 579 segmented X-ray (AP, PA views) images of COVID-19 and Pneumonia with each patient's textual medical data that include age and gender. The proposed framework achieved an accuracy of 97.85% compared to 94.32% without weight optimization. Furthermore, an extensive comparison with several other architectures from the literature was used to evaluate the model viability in detecting COVID-19.

Keywords: COVID-19, Multilayer perceptron, Convolutional Neural Network, Metaheuristics.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

At the beginning of the 21st century, severe acute respiratory syndrome (SARS) [1]became the first severe, readily transmissible virus to be identified at the end of February 2003 by the world health organization (WHO). SARS-COV-2, also known as COVID-19, was first observed in Wuhan, China, and shares 80% of its genome with SARS. COVID-19 already contributed to over 160 million confirmed cases and more than 4.8 million deaths as of October 2021[2]. Furthermore, the disease tends to spread droplets and the high mutation probability combined with the lack of appropriate testing kits significantly disturbed the efforts in containing the viral infection. The World Health Organization declared COVID-19 as a pandemic on 11 March 2020. The high Infectability of the disease resulted in more than 20% of infected cases being admitted to the intensive care units that imposed significant pressure on healthcare systems. The global health system was unprepared to handle such an increase in admissions rate, due to the lack of qualified personal and protective equipment. Furthermore, the high false-positive rate of test kits and lack of understanding regarding COVID-19 led to an extreme and justifiable measure taken by most governments to curb the escalating transmission rate of COVID-19; lockdowns were the most effective measure to prevent the spread of COVID-19, while this measure was effective in areas where it could be enforceable, unfortunately, the fact that COVID-19 has not been eliminated from the global scene, after the deployment of vaccines, indicated to the lack of awareness regarding COVID-19. To elevate some pressure from the health care system, the deployment of vaccines needs to be on a much larger scale and herd immunity is achieved, or COVID-19 will continue to be a threat to individuals and the healthcare systems.

The testing approach for COVID-19 using reverse transcription-polymerase chain reaction (RT-PCR) test kits has high accuracy in detecting COVID-19. However, there are several cases where the RT-PCR test is not feasible, especially in cases where the patient is physically unable to perform such test, e.g., if the patient is diagnosed with throat cancer [3], in addition, the lack of training and deployment of test kits in rural communities contributed significantly to the difficulty of utilizing such tests. Furthermore, several research papers pointed to the long-term effects of COVID-19 that cannot be detected using RT-PCR. Han et al. [4], in their results,

pointed to the long-term effect of COVID-19, one-third of participants, 40 of 114 that have previously discharged on recovery, showed abnormal lung changes, with higher findings in older age. RT-PCR works on a time window when the virus is active in the host body and the post-COVID symptoms cannot be detected post-recover[5], thus creating the need for an alternative testing method such as medical image analysis using X-ray scans as diagnostic method for testing suspected COVID-19 patient.

Artificial intelligence (AI) systems made a giant leap in the last decade, the employment of such systems in several fields has shown an advantage over traditional computing systems [6].AI systems have been utilized in analyzing medical images and can provide the ability to automate tasks that were previously reserved for specialists, while implementing them with far lower incidence of failure. Motivated by this, most of the research effort has focused on public health concerns after 2020. As a result, many AI researchers started to employ non-invasive methods like X-rays scans to detect COVID-19 and the post-COVID symptoms in suspected cases. Several recent studies have already utilized their AI implementations to classify skin cancer cells based only on medical images, proving the validity and accuracy of such methods. Medical scans, such as X-rays, Computer Tomography (CT) scans and MRIs, can effectively detect COVID-19. X-rays are the most common and by far the most cost-effective option and widely available in more countries, it is the ideal choice for automatic image analysis. Nevertheless, one must not discard the ability to investigate the other types of screening techniques; CT scans play an essential role in diagnosing COVID-19. Grillet et al. [7], in their work, investigated the ability of CT scans to diagnose the severity of COVID-19 patients. In their findings, the authors concluded that the patients are more likely to be admitted to the ICU with a high prevalence of acute pulmonary embolism; as a result, CT scans must be considered. The healthcare sector is full of cutting-edge, forward-thinking technologies, when combined with other techniques, that aim to revolutionize the way patients are treated.

Convolutional Neural Networks (CNN) are the de facto standard for image processing in several commercial and academic fields, most research done on this topic has focused primarily on the visual component of the problem. In this study, a deep learning model is developed that will serve as a predictive function to classify COVID-19 from Pneumonia infected patients, by

analyzing patient X-rays and textual clinical data such as age and gender. Furthermore, the deep learning model is based on a hybrid network that employs multiple CNNs combined with a multi-layer perceptron network (MLP). The data used to train the network is obtained from two different public sources, one for COVID-19 and the other for Pneumonia, by taking data points that have a single pathogen, to try and limit variables that may affect the results. In addition, data augmentation is utilized in the augmentation pipeline such as rotation, cropping and elastic transformation to address the class imbalance that is present in the dataset by increasing the number of samples artificially. Image segmentation is essential for medical image analysis, it forces the network to learn from the region of interest and eliminates the annotation that may be present in raw medical data. The segmentation model is based on a pre-trained ResNet34 model for improved quality segmentation of the lungs from an X-ray scan, Figure 1.1 shows the segmented lungs. Furthermore, Contrast Limited Adaptive Histogram Equalization (CLAHE) is also utilized to increase the quality of the contrast in X-ray images while minimizing the noise that is present in the raw images.

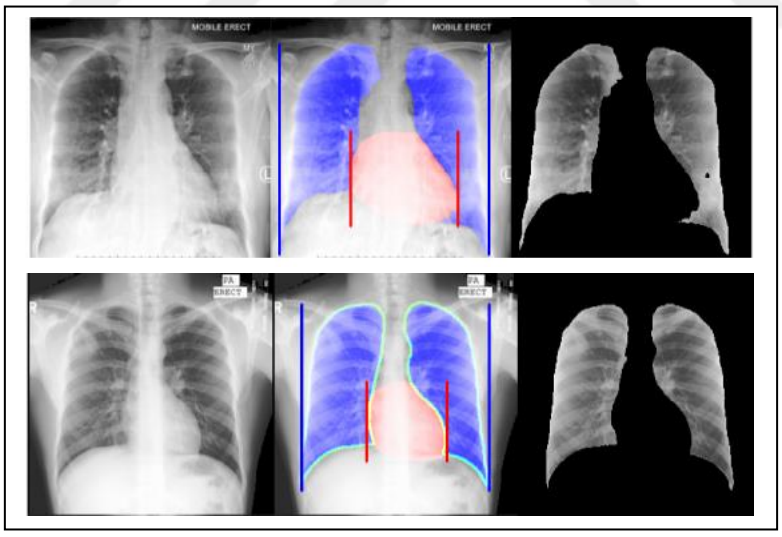


Figure 1.1 Pre-processed images, considering each one of the classes separately Top: COVID-19. Bottom: Pneumonia.

Typically, network weights are adjusted using a variant of Gradient Descent (GD) algorithms and Backpropagation (BP) [8], [9]; in their pursuit of achieving global optima, gradient descent often gets trapped in local optima [8], a solution to this inherited drawback is performing

significant update steps or using other searching methods such as Metaheuristics. Metaheuristic algorithms are less prone to get trapped in local optima. in this study. A metaheuristic search algorithm is employed as an optimization function to optimize the weights of a neural network. The hybrid network weights are further optimized using a modified version (MEAEO) of the Enhanced Artificial Ecosystem optimization [10]. Furthermore, the hybrid network hyperparameters are selected by using grid search for improved performance. The main goal of this study will be to develop and validate a novel weight-optimized Deep Learning model and incorporate several image augmentation techniques for better classification. The framework utilizes medical images and textual clinical data as input data to distinguish COVID-19 from pneumonia in infected individuals.

The remaining sections of this study can be outlined as follows; in Chapter 2, a detailed review of the most recent and related work that has similar functionality to the proposed framework based on COVID-19 detection and the optimization aspect. Chapter 3, the basics of machine learning and the fundamentals of neural networks Chapter 4 gives a detailed explanation of the hybrid neural network model, the weight optimization aspect and discusses the dataset with the limitation of using medical data. Chapter 5 will discuss the experiments and the performance evaluation of the proposed framework compared to similar frameworks in the literature and the reasoning behind employing the lung segmentation. Finally, Future work and the conclusion will be presented in Chapter 6.

1.1 PROBLEM STATEMENT AND CONTEXT

Health and Safety of humans are of utmost importance. the employment of X-ray images in detecting COVID-19 is useful in cases where RT-PCR is not a viable choice, RT-PCR test kits utilization depends on several factors that cannot be met in most situation.

This study uses several techniques such as image segmentation, machine learning and metaheuristics to analyze X-ray images for the purpose of detecting COVID-19 and pneumonia.

1.2 HYPOTHESIS

The use of computer aided diagnostics can help in detecting several health problems including COVID-19, while image segmentation is a required step to achieve optimal results that are not affected by bias, incorporating several data types as input for neural network systems can increase detection reliability and weight optimization is a valid technique for neural network by using a metaheuristic algorithm.

1.3 OBJECTIVES

The purpose of this study is to create a hybrid convolutional neural network and multi-layer perceptron model capable of predicting COVID-19 based on medical images and textual data. Additionally, an investigation for the impact in performance while using a metaheuristic algorithm for weight optimization. Furthermore, a comparison between the result obtained from several other models that share, to some extent, the same aim as this study is made in Chapter 5. To the best of our knowledge, no previous study has investigated the weight optimization aspect of convolutional neural network (CNN) and Multilayer perceptron (MLP) using meta-hubristic algorithms and COVID-19 data.

1.4 SCOPE

In this study, a range of objectives is outlined to demonstrate the validity of this research methodology in order to ensure that the study's experimentation remains rational, reasonable, and valuable but still exact.

i) Images and textual medical data: This research will use medical images obtained from [11] and [12], limiting the experiments to X-ray images with PA and AP views. Each image will be transferred to grayscale with the size of 256×256 pixels, further discussion regarding data processing will be detailed in later Chapters.

ii) Pathogens: COVID-19 and Pneumonia pathogens were selected for the experiments; each data point contains a single pathogen.

2. LITERATURE REVIEW

COVID-19 classification has been an appealing research topic for many researchers. Several research studies have been conducted to determine the potential of AI classification systems to identify and analyze COVID-19 scans. The employment of AI systems in the medical sector is undoubtedly beneficial for detecting, diagnosis of COVID-19. Furthermore, Cave et al. [13] discussed the benefits and drawbacks of using AI systems in the medical field from an ethical standpoint. Nevertheless, the fact that millions are affected is enough reason to explore this problem. DeGrave et al. [14], in their report, questioned the validity of AI systems created by researchers regarding the approaches taken to classify COVID-19; while the argument is up to debate, this section will survey several studies published in the literature similar to some extent, the proposed framework. This section also covers classification systems using CT and X-ray scans and image augmentation. Also, it highlights the benefits of using medical imaging techniques to increase system robustness and discusses metaheuristics techniques used for weight optimization.

Several studies have been published in the literature that uses various deep learning algorithms on X-ray images and demonstrated good performance [15], [16]. Ahsan et al. [17] proposed a CNN-MLP architecture that employs a merged CNN and an MLP network system to increase the model robustness. The proposed model employs COVID-19 X-ray images and textual clinical data as inputs, according to their findings, the model achieved an overall average accuracy of 94.6% ± 3.42% for binary classification. Later in Chapter 5, an implementation of this method is compared with the findings of this proposed study. Afifi et al. [18], the authors compared several pre-trained networks (Resnet18, densenet161, and inceptionv4) to detect COVID-19 from X-ray s images. Global and local attention mechanism was also utilized based on several pre-defined architectures for multi-label classification. In their approach, the authors employed lung segmentation to exclude undesirable features. They concluded that Deep Learning Ensembles coupled with Transfer Learning and lung segmentation is a viable technique; the proposed method achieved an accuracy of 91.2% for the three-class problem. Mukherjee et al. [19]concluded that a single CNN architecture could be trained on X-ray and

CT scans. In their report, the proposed tailored, simple architecture achieved an accuracy of 96.28% on a dataset that has balanced classes (number of samples) of 672 samples for X-ray and CT scans. Two X-ray and CT scan datasets are evaluated separately using the proposed architecture. They achieved an accuracy of 96.13% and 95.83%, respectively. Apostolopoulos et al. [20], the authors utilized pre-trained CNN models to classify COVID-19 from X-ray images; due to the small dataset of 224 COVID-19 positive images, the authors employed transfer learning trained on the ImageNet dataset to increase the robustness of the model. In their work, the authors compared VGG19, Mobile Net, Inception, Xception, and Inception-ResNet v2. They concluded that VGG19 outperforms other pre-trained models in classifying the COVID-19 class with an accuracy of 98.75%; the overall accuracy of the proposed system was 93.48%.

Furthermore, Wang et al. [21] proposed a deep learning model based on the design pattern of PEPX. The model was first pre-trained on the ImageNet dataset, and then the COVIDx dataset was used to retrain their model. Also, image augmentation was utilized. The authors reported an accuracy of 92.4% with 80% sensitivity for the COVID-19 class. A recent work [22] analyzed a trained DenseNet201 model using Layer-wise Relevance Propagation; the authors proposed an intermediate module between segmentation module U-Net and classification module using DenseNet201. The intermediate module applies the SoftMax function to the U-Net module. It generates an image based on the last channel of the SoftMax function and the original X-ray image using an elementwise multiplication, thus creating a segmented image with only the valuable regions included. They reported that the model successfully excluded areas outside the lungs and achieved a classification rate of 0.917.

Several other studies highlighted the importance of different data types in detecting COVID-19 infected individuals. For instance, Lassau et al. [23] used a deep learning-based CT scan model with textual clinical information. In their report, the authors found 12 variables that are significantly associated with severity like age, gender, and oxygen saturation, to name a few highlighting the importance of combining several data types to achieve a lower misclassification rate. Likewise, in [24], the authors proposed a multi-modal with the late fusion of textual clinical data to classify and assess the severity of COVID-19 infection. In their proposal, the framework

employed CNN based on the ResNet architecture for feature extraction. In addition, machine learning algorithms (random forest, SVM, KNN) have been used as feature classifiers. In their work, the authors performed feature extraction from CT images using the CNN model in addition to a fully connected layer to accommodate the proposed late fusion technique of the textual clinical data, thus creating a single row vector. The resulted output from the fusion is classified using the mentioned machine learning algorithms. The authors concluded that merging textual clinical data with the features extracted from CT images provides better classification accuracy; the proposed system accuracy ranges 95.4%97.7%. In [25], the authors evaluated the performance of three AI models against two radiologists. The AI models consist of CNN, MLP, and a joint model CNN+MLP that takes a CT slice and textual clinical data as an input. In their study, the authors employed a model that has been previously trained on pulmonary tuberculosis for the abnormal slice selection from the CT scan. The authors evaluated the proposed system performance with the result obtained from the radiologist and concluded that the proposed system is comparable in terms of classification accuracy.

Image segmentation is crucial in creating a model that learns from unbiased characteristics present in an unsegmented image. Lung segmentation forces the model to extract features that are relevant to the problem given. Most state-of-the-art segmentation architectures employ Convolutional Neural Networks for their operation. Ronneberger et al. in [26] proposed a model for segmented biomedical images titled U-Net. The objective is to isolate and extract the infected region from the medical image. U-Net architecture can be described as a decoder encoder network. Another study [27] employed VNET-IR-RPN to extract regions of interest in CT images; the proposed method was used to detect COVID-19, influenza-A, viral pneumonia, and healthy cases from CT images. The model achieved a good accuracy of 86.7% while testing CT images for the disease mentioned earlier.

Weight optimization is a challenging task that is crucial in supervised learning. Multiple MHAs have been proposed to address various challenges, such as neural network weight optimization [28] and image enhancement [29]. Metaheuristic algorithms have been used effectively to solve many optimization problems. Neural network training is one of the many

functions of metaheuristic algorithms and can be effective due to the advantages like simplicity, independence from the gradient and local optima avoidance.

Particle Swarm Optimization was used In [30] to optimize the weights and architecture of the MLP network. The authors proposed the use of two separate PSOs (PSO and PSO with weight decay). The first PSO is used to search the number of units in a hidden layer, while the PSO with weight decay is responsible for optimizing the weights. The authors tested this technique against several medical problems and concluded that PSO with weight decay is valid in optimizing network weights. In [31], the authors applied an enhanced version of the black hole algorithm [32] to search for optimal weights and biases, this method achieved a better result than other MHAs mentioned in the study. The experiments were performed on several datasets such as blood, liver disorders, seeds, statlog (Heart), which achieved a mean squared error of 0.1528 with the blood dataset.

In [33], the authors proposed a method to enhance the performance of artificial neural networks using PSO. It was used as an alternative for GD and Backpropagation to train the proposed system by returning the best weights obtained from the PSO. The experiments were carried out against ten publicly available datasets. The proposed approach achieved a lower training time averaging 76.91 seconds which is lower than training with a backpropagation that averaged 93.32 seconds. Furthermore, PSO achieved a higher classification accuracy of 97.20% compared with Backpropagation accuracy of 94.32%.

3. NEURAL NETWORKS, AND MACHINE LEARNING FUNDAMENTALS

This study is concerned with artificial neural networks, specifically convolutional neural networks. As a result, obtaining a good foundation in this topic is important. It is not recommended to go any deeper into the issue. There are countless books dedicated to this topic. As a result, this chapter can only provide a high-level overview of these topics. The chapter begins with a basic introduction to machine learning before moving on to artificial neural networks and convolutional neural networks.

3.1 MACHINE LEARNING

This section provides a summary of machine learning fundamentals. It explains the fundamental ideas of machine learning and briefly outlines some common machine learning terminology. This section is for readers who have no prior expertise with machine learning.

3.1.1 What is Machine Learning

Machine Learning is a branch of Artificial Intelligence that enables computers to automatically learn and improve based on their experiences without being explicitly programmed. Machine learning (ML) encompasses probabilistic and statistical modeling, machine learning, and deep learning approaches. It may be used to perform various tasks, including categorization, prediction, and decision making. As depicted in Figure 3.1,

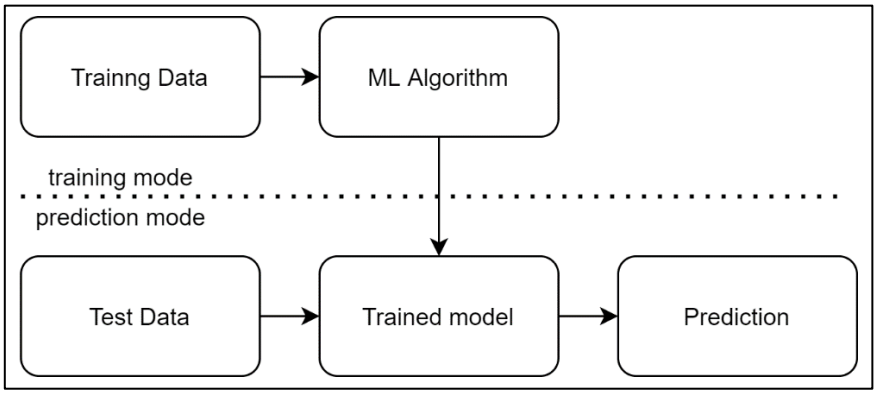


Figure 3.1: General workflow of a Machine learning Model.

the machine-learning process has two significant phases: training and prediction. In the training phase, the machine learning algorithm that has been selected for this task is fed a continuous stream of labeled data, while the prediction phase is when the trained machine learning algorithm predict a value for unknown data based on the trained weights. In this study, the data consist of medical and textual clinical data.

3.1.2 Learning Types

Machine learning algorithms can be trained on different learning types; the learning problems can be divided into three main subsets.

Supervised learning: in supervised learning, the Machine learning algorithm is trained on a count of labeled data that consist of datapoint and expected output for the data point. This type of learning problem can be further divided into classification and regression. For classification, the data is divided -sometimes equally- into two or more classes, and the objective is to draw some conclusions from the labelled input data. In contrast, regression means learning of continuous data, i.e., age. For example, the person's age, the input data could be height and facial features while the out could be the age of that exact person as exact value, not age groups. Supervised learning can be multi-class and multi-label problems.

Unsupervised learning: in this type of learning, the selected Machine learning algorithm Is trained on unlabeled data that contains only data without the expected output or label, and the objective is to find patterns or correlations in the data that can be used for identifying learned patterns in a new unseen data.

Reinforcement learning: as the name implies, the selected machine learning algorithm is positively or negatively rewarded based on the interaction with a pre-defined environment. An example can be given of how good an AI driving agent (machine learning algorithm) is with the defined environment.

The learning problem in this study is supervised learning with a multi-class problem. The dataset employed by this study consists of several data points with a label that defines the expected outcome in the training and prediction phases. The datasets will be explained in detail in Chapter 4.

3.1.3 Under- and Overfitting

A model that has been constructed based on a machine learning algorithm can be inefficient or insufficient for a particular task; these behaviors can be observed in the evaluation (testing) process of the said model. The performance observation is called generalization (low test error) can be divided into Underfitting or Overfitting to the data; either the model has insufficient capacity to achieve the generalization goal (Underfitting) or too much capacity to memorize the data (Overfitting), and lousy generalization is obtained. If a model is underfitting, then the training and testing errors are equally high, and if the model is Overfitting, then the training error is low while the test error is high. A model is said to have good performance is when the training and testing errors are equal and a good generalization is achieved. Figure 3.2 shows sinus functions that depict different generalizations of three models.

It is important to note that a higher capacity model can be fitted and have a suitable generalization of the data if a regularization technique is used, these techniques will be discussed in section 3.2.5.

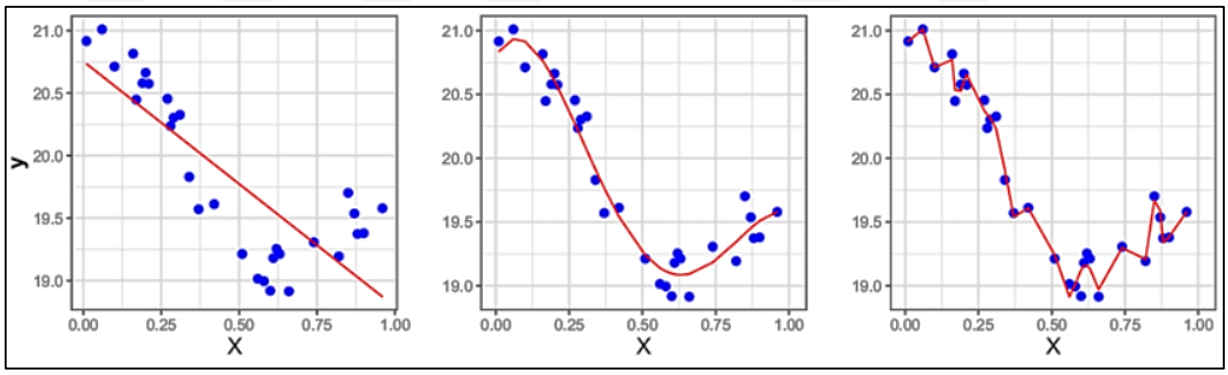


Figure 3.2: Depiction of the three fitting types. Left underfit center good fit, right overfit

3.1.4 Hyperparameter Selection

Hyperparameters are parameters of machine learning algorithms that are defined pre-training. These parameters can alter the algorithm behaviour; a good selection of these parameters can significantly impact the capacity of a model. Typically, hyperparameter selection (tuning) is based on a trial or error process, but there are automated processes of

selection that can be less tedious, for example, grid search. If the selected by the grid search parameters are performing as expected, then the selection process is terminated.

3.2 ARTIFICIAL NEURAL NETWORKS

Artificial Neural Networks (ANN) are inspired by living organisms' biological neural networks (BNN); an attempt to mimic the problem-solving capability of said networks artificially and is simply an aggregation of multiple Perceptrons (neuron) working together towards the same goal. Artificial Neural Networks consist of links that accept input values and apply a function to create an output value. Neural networks are one of the many types that fall under machine learning algorithms.

Artificial Neural Networks (ANN) are trained or fitted using a training dataset; each neuron has its weight adjusted gradually for the best prediction using optimization algorithms such as GD. In CNN, each weight is updated after each batch. The recent advancement in data collection techniques and the increased computational power in data processing permitted the design of complex networks that can solve complex problems promptly with more accuracy. ANNs typically have one node with at least one input, and one output referred to as the *input layer* and *output layer*. Additional layers with more inputs and outputs may be used for ANNs to perform non-linear classification; these extra layers are called *hidden layers*.

3.2.1 Perceptron

Perceptrons are the building blocks of the Artificial neural network, introduced by Rosenblatt [34]. and considered the first significant breakthrough in neural network models, The main component of neural networks is the neurons (Perceptron). In other terms, Neural networks are multi-layer networks of neurons that can learn complex patterns. Neurons are the basic building block of each neural network. Perceptrons enabled computers to learn from example data without relying on hand-coded information about which attributes are significant or how they interact with one another. Figure 3.3 depicts a simple Perceptron with multiple inputs and one output. In addition, the Perceptron also employs bias input w_0 :

$$y = w_0 + \sum_{i=1}^{n} x_i w_i \tag{3.1}$$

Where w_i is the weight value.

Furthermore, weights in perceptrons are numerical values that affect the output and consider the most crucial factor in affecting the output value of a perceptron. Weights are adjusted during the training phase to weaken or reinforce the input to get the desired output. The linear output of a Perceptrons is fed into an activation pre-defined functions (activation functions discussed in 2.2.4) and is responsible for activating the neuron if a certain threshold is fulfilled.

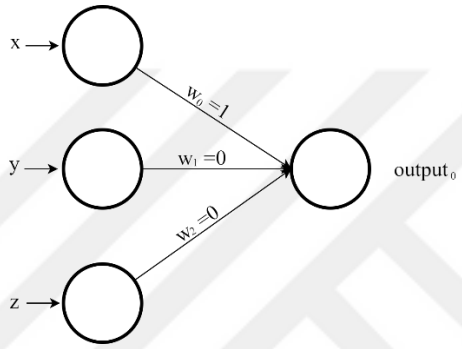


Figure 3.3: Perceptron with *n* inputs and one output.

3.2.2 Training of Artificial Neural Networks

This section explains how a pre-defined network is trained—considering the aim of this study using supervised learning.

After the hyperparameters and data pre-processing are completed, training a network for a specific task occurs. A network's weights are assigned at random or by employing a weight initializing function, such as HE-normal, which draws a sample within a pre-defined range. As previously established, input and output are supplied in supervised learning. The network analyses the input and compares the anticipated output to the given actual output; based on the similarity, the error is computed. The error is returned to the network for weight adjustment. This procedure is iterated according to a predefined iteration parameter.

Backpropagation

Backward propagation of errors (Backpropagation) is a technique used to adjust the neural network parameters based on the gradient of the error function. As the name implies, backpropagation performs a backward pass after each forward pass for adjusting the weights.

Loss Function

A loss function in mathematics is a function that describes from a probability distribution to an actual number and quantifies the average or anticipated loss of an estimator. Many statistical models, including linear regression and binary classification, rely on the notion of a loss function in their design and analysis. The type of loss function used in these settings is generally decided by the theoretical constraints needed for parameter estimation. The employment of loss function will be further expanded on in Chapter 4.

Gradient Descent

As discussed earlier, gradient descent is an optimization algorithm. The gradient is a numeric calculation that adjusts a network's parameters to minimize its output deviation. Gradient descent is used in finding a local minimum of a function.

Stochastic Gradient Descent

Stochastic gradient descent SGD [35] is a simple gradient descent implementation that updates the weights at each stage of the training process using *mini-batches*; in contrast, gradient descent requires the iteration of a complete training set before calculating the gradient. As a result, SGD is beneficial for describing the model's performance and monitor if the model is underperforming. The noise associated with the updating process is one of the benefits of SGD; it might allow for faster convergence. Mini-batches, iteration and epochs are essential concepts, and it is appropriate to discuss next.

Mini batches are, by definition, a small number of samples from a large dataset. The number of samples of each minibatch is pre-defined

Iteration or steps is the steps required for batches to complete one epoch.

Epoch is when the complete learning data are passed through the feedforward and backpropagation once.

Adaptive Gradient Descent

Adaptive gradient (Adagrad)[36] is a gradient-based optimization technique that retains the gradient square for each iteration. It adjusts the learning rate to the parameters; if a parameter

has a high value, the learning rate is lowered, and more minor updates are performed. Adagrad has the benefit of being less computationally costly.

Adam

Adaptive Moment Estimation (Adam)[37]. The main advantage of Adam is that it can handle noisy data more effectively than SGD. It combines the best of Adagrad and Root Mean Squared Propagation, which in turn is an extension of Adagrad and gradient descent.

Nesterov-accelerated Adaptive Moment Estimation

Nesterov-accelerated Adaptive Moment Estimation (Nadam)[38] is an extension to the Adaptive Moment Estimation (Adam) by adding Nesterov's Accelerated Gradient (NAG). Furthermore, it is proposed to enhance the optimization of the gradient in case of a flat curve is encountered.

3.2.3 Activation Functions

Activation functions are essential for building a neural network because they determine whether or not a neuron should be activated based on a rule or threshold. The primary role of activations is to induce nonlinearity into the neuron's output. The network without activation function can be considered linear regression.

ReLu

Rectified Linear Unit (ReLU) is the most widely employed activation function in neural networks [39]. It is a linear component that directly emits the data if it is positive; in other words, it sets all negative activations to 0. The rectifier provides just enough nonlinearity to facilitate highly complicated representations.

$$relu(x) = x^{+} = \max(0, x) \tag{3.2}$$

Where x is the neuron output

Hyperbolic tangent activation function

Hyperbolic tangent activation function (tanh) [40]can be defined as the ratio between the hyperbolic sine and the cosine functions. It is similar to the sigmoid function; tanh function has

a range of [1 to 1]. It has a drawback in mitigating the vanishing/exploding gradient problem, especially when the network has many layers.

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{3.3}$$

Where *x* is the output of a neuron

Swish

Proposed by the Google brain team [41], it's a smoothing function, and it's the multiplication of sigmoid function with an *x* input, and behave as a smoothing function. The main difference from ReLu is that it does not change the output values abruptly

$$swish(x) = x \times sigmoid\,(\beta x) = \frac{x}{1 + e^{-\beta x}} \tag{3.4}$$

Where *x* is the neuron output and β is a trainable parameter.

Scaled Exponential Linear Unit

Scaled Exponential Linear Unit (Selu)[42]is an activation function based on ELU that induce self-normalizing properties by using two fixed parameters α, λ to preserve the mean of 0 and stander deviation of 1, SELU can solve vanishing and exploding gradients.

$$\begin{aligned} selu(x) &= \lambda x \ if \ \ x \geq 0 \\ selu(x) &= \lambda\alpha(e^{x} - 1) \ if \ \ x < 0 \end{aligned} \tag{3.5}$$

Where *x* is the neuron output and $\alpha \approx 1.6733$, and $\lambda \approx 1.0507$.

SoftMax

SoftMax activation is a form of the artificial neural network activation function. The function is generally used to describe the probability distribution of classification problem results. It is a mathematical function that transforms the output of numbers into probabilities, in which the likelihoods of a probability distribution over *K* classes, SoftMax can be represented as :

$$\begin{bmatrix} 1.3 \\ 5.1 \\ 2.2 \\ 0.5 \\ 4.2 \end{bmatrix} \rightarrow \frac{e^{zi}}{\sum_{j=1}^{k} e^{zj}} \rightarrow \begin{bmatrix} 0.05 \\ 0.90 \\ 0.03 \\ 0.01 \\ 0.01 \end{bmatrix} \tag{3.6}$$

SoftMax scales the output of each class in a range between 0 and 1. The sum of outputs of all classes must be amount to 1, and the highest probability in a specific class is highlighted.

3.2.4 Regularization

Regularization refers to the techniques used in conjunction with each other to minimize the memorizing of the training data. As discussed earlier, a model can underperform in certain conditions, and this can be avoided if suitable regularization techniques are used.

Batch Normalization

It is necessary to discuss normalization to fully understand batch normalization. Normalization is a pre-processing technique for standardizing data. Data normalizations aim to transform the numeric values range to a common scale, it significantly improves the training stability and generalization performance of networks with diverse types of inputs. The most straightforward method is to scale it to a range from 0 to 1:

$$x_{normalized} = \frac{x - m}{x_{\max} - x_{\min}} \tag{3.7}$$

x is the data to normalize, m is the mean of the data, $x_{\max}$ and $x_{\min}$ are the highest and lowest values in a dataset. It is important to note that normalization across instances must be performed after separating the data into training and testing sets and using only data from the training set. That's because the testing set represents new, unseen data, and it is not meant to be available during the training stage. Using any data from the testing dataset before or during training may introduce bias into the performance evaluation.

Batch normalization, introduced by Ioffe et.al., [43] is a technique that mitigates the problem of internal covariate shift by normalizing data across network layers. Batch normalization improves network stability, performance and preserving the representation power of the units via normalizing the layers' inputs by re-shifting and re-scaling, which counteracts the change in distribution during learning and enables the layer to train more independently. The formula for Batch normalization is the following:

$$\mu_{\mathrm{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \tag{3.8}$$

$$\sigma_{\mathrm{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathrm{B}})^2 \tag{3.9}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathrm{B}}}{\sqrt{\sigma_{\mathrm{B}}^2 + \varepsilon}} \tag{3.10}$$

Where μ and σ is the mean and standard deviation, x is the value of x over a mini-batch $B = \{x_{i\ldots m}\}$, m describe the size of the mini-batch, $\tilde{x}_i$ is the normalized value. Rescaling and re-shifting can be mathematically presented as follows:

$$y_i \leftarrow \gamma \hat{x}_i + \beta = BN_{\gamma,\beta}(x_i) \tag{3.11}$$

γ and β are parameters to be learned during the training. So, the value $\tilde{x}_i$ could have any mean and standard of 0 and deviation of 1.

Dropout

Dropout is a regularization method that randomly ignores neurons (drop) for every iteration during training. Dropout is a simple technique with a significant impact on the performance of a network, with the right threshold, dropout can prevent neural networks from overfitting. Usually, dropout is placed on the fully connected layers only. It is important to note that dropout layers have no trainable parameters and are activated only in the training phase.

3.3 CONVOLUTIONAL NEURAL NETWORKS

Convolution Neural Networks (CNN) are specialized Artificial Neural Networks that excels in features extraction from images, and they are also applicable to other forms of data beyond

images. Convolution Neural Networks was inspired by neocognitron “neocognitron is a hierarchical, multilayered artificial neural network” [44], CNN was first introduced by Atlas et.al., 1987. The distinction between ANN and it is that it considers the multiple dimensions of an image rather than just one dimension and has the ability for weight sharing. The architecture of Convolutional Neural Networks is mainly composed of several convolutional layers and pooling layers to achieve a specific task. Each convolution layer is often followed by a regularization layer and a pooling layer. However, the ordering and inclusion of the different layers can be changed or reordered differently based on the needs of such networks. The use of convolutional networks permitted a breakthrough in image processing.

3.3.1 Role of Convolution Layer

Convolutional layers are the primary building block of convolutional neural networks. It is based on the advancement in machine learning techniques. There are different types of convolutional layers such as 1D, 2D and 3D convolutional layers. In this study, only a 2D convolution layer will be considered and can be abbreviated as conv2D.

Conv2D has several components manly filter sizes the stride. Filters are responsible for detecting spital features such as patterns; a filter or a kernel in a conv2D layer has a height and a width. They are generally smaller than the input image. Filters are moved or *convoluted* through the image for its operation. At the same time, strides are responsible for the number of pixels that a filter is shifted over the layer's input. Conv2D has several advantages over fully connected layers and can be summarized as the weight sharing ability and local connectivity. weight sharing, as the name implies weights are shared between multiple output units of a layer to find the optimum value for a specific task. Sharing weight has the benefit of faster feature extraction. Local connectivity of Conv2D means that not all input units are connected to the output of the network. Each neuron receives input from a local group of pixels in a feature map.

3.3.2 Pooling Layer

Pooling layers operate by progressively reducing the dimension of the feature map and summarizing the features contained in a region of the feature map generated by a convolution layer using various types of pooling, i.e., max- and average-pooling; the pooling technique also

decreases the parameters needed to train, leading to a reduction in computation. Figure 3.4 demonstrates how a pooling layer functions by taking a sample with a specified filter size and stride. The max-pooling algorithm computes the maximum in the filter's receptive field, whereas the average-pooling algorithm computes the average.

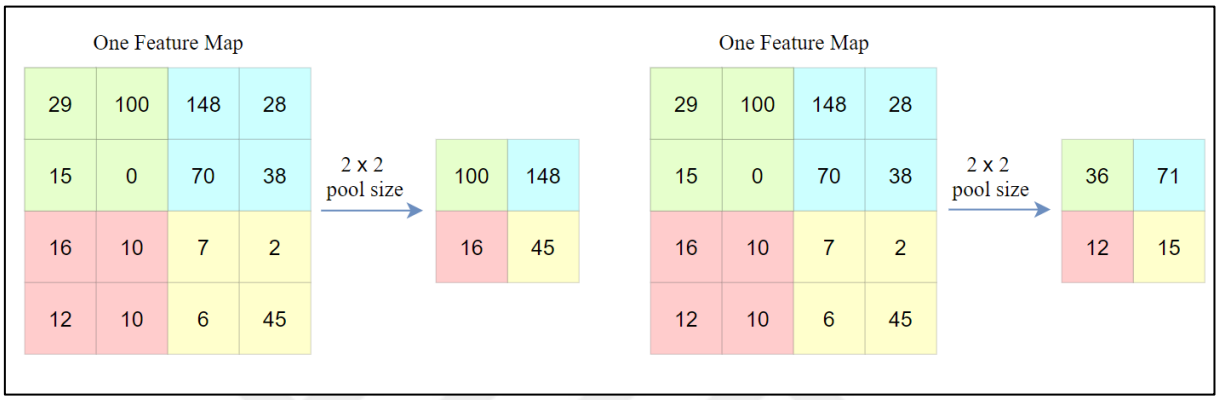


Figure 3.4: Example of max-and average poling. Left max pooling, right average pooling.

The most common pooling layers are the max- and average-pooling layers. In both pooling layers, the filter size and stride are employed as hyperparameters. Max-pooling is used to compute the highest in the filter's receptive field, whereas average-pooling is used to compute the median. Figure 3.4 also depicts filter size for both pooling methods.

3.4 MULTILAYER PERCEPTRONS

Multilayer Perceptrons, or MLPs for short they are the typical neural networks that comprise one or more layers of neurons. The structure can be divided into three layers: input layer, hidden layers, and output layer. The input layer contains artificial input neurons [45], which is the first layer of each neural network that specializes in providing the basic information of the initial data for subsequent layers like how many samples and the dimension of the data. A hidden layer in a neural network, located between the input and output layers and called hidden because it cannot be observed immediately from the system's inputs and outputs. The hidden layer usually contains zero or more neurons. Output layer, whether classification or regression, each neural network must have at least one output layer with a particular function to decide how to process the output for prediction or regression.

3.5 METAHEURISTICS

A metaheuristic is a generalized solving method A metaheuristic is a high-level problem-independent algorithmic framework that provides a set of guidelines or strategies to develop heuristic optimization algorithms. The term is also used to refer to a problem-specific implementation of a heuristic optimization algorithm according to the guidelines expressed in such a framework. metaheuristics algorithms can be classified into the following categories.

i) The evolution of species in nature inspires evolutionary algorithms. One of the most famous in this category is the Genetic Algorithm (GA)[46].

ii) Swarm based algorithms are inspired by the behaviour of living organisms such as bees and butterflies. Butterfly optimization [47] is a unique algorithm in this category.

iii) Human-based optimization, this category of algorithms, is designed based on humans' behaviour in tasks or activities. The Battle royale optimization (BRO) algorithm [48] is an example of this category.

iv) Physics inspired optimization Simulated Annealing (SA) is a well-known optimization algorithm that has been used to solve many multi-objective problems [28]. The physics present in nature is the inspiration for this category.

v) System based optimization, not many algorithms fall under this category. This category is used in this study and employed an Enhanced Artificial Ecosystem optimization (EAEO) [10]. The inspiration is drawn from the flow of energy in an ecosystem.

vi) Biobased optimization, as the name implies, this biology-based algorithm is inspired by nature biology; the Earthworm optimization algorithm [49] is an example in this category.

In this study, an enhanced artificial ecosystem is used as the main optimization algorithm. EAEO falls under the System-based optimization category. A detailed explanation will be presented in section 4.5.1. While genetic algorithm, particle swarm and Artificial ecosystem are employed as comparison functions to evaluate the algorithm performance regarding weight optimization.

4. METHODOLOGY

This section discusses the methods used in the proposed framework. First, an outline of the data used in this study and the collection and processing methods employed with the consideration to the guidelines of medical data processing outlined in [50]. Then a detailed explanation of the proposed hybrid neural network in Figure 4.1. and the different components used to achieve the desired output. Then, the weights optimization aspect is presented, and the incorporation of such technique in the framework is also presented. Furthermore, in the last section, a comprehensive explanation of the framework parameters used in the hybrid model, the programming language and the machine learning frameworks applied in these experiments.

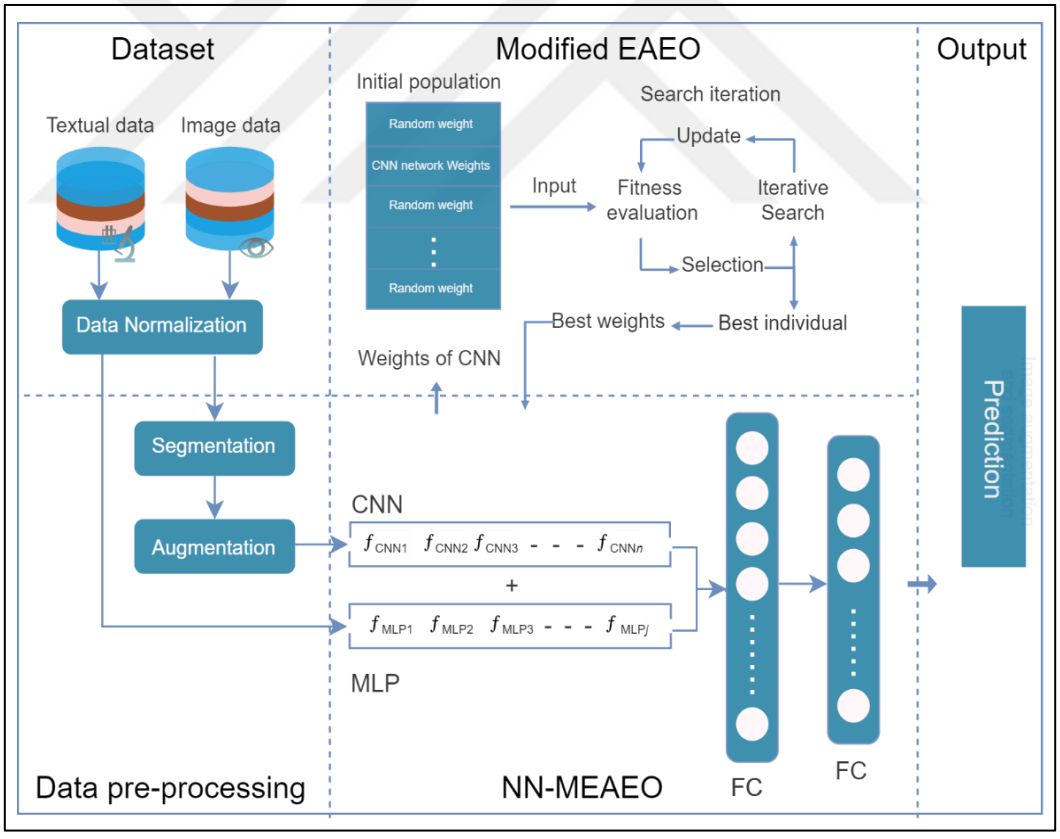


Figure 4.1: Illustration of the Proposed CNN-MLP framework used to generate a prediction based on the input data as shown in the figure.

The architecture of this proposed framework can be divided into three components, datasets collection and processing, a deep learning model, and weight optimization using a modified EAEO. A detailed explanation of each component is presented in this section. In the first phase, the raw data is supplied to the image processing pipeline which in turn generate a segmented image, then the segmented images are supplied into the proposed framework. Image pre-processing is discussed in section 4.2. In the second phase, the processed image is supplied to the proposed hybrid neural network MLP for feature extraction, learning, and recognition. Finally, the weights of the hybrid network are extracted and tuned using a modified EAEO to maximize the proposed network performance.

4.1 THE DATASET

One of the important steps to achieve a good generalization is the use of a suitable dataset. To achieve high generalization and minimize the false positive rate, a biomedical dataset with strict preprocessing methods is needed. For the classification task in this study, the dataset must contain X-ray images with the textual clinical information i.e., age, gender and so on. Furthermore, the images should contain an X-ray image with AP and PA views.

There are several datasets available in the research community that have a significant amount of images, but a suitable dataset for this study must contain textual clinical information which most of them does not provide such data. Therefore, two datasets from different sources are employed to achieve the required data count for a CNN to learn from.

The datasets used in this study are compiled from two different public sources. Table 4.1 shows the sample size in the dataset, whereas Figure. 4.2, provides an insight into the age and gender distribution in the dataset. While both datasets include over 122k samples, the utilization of only X-ray images with a single pathogen drastically limited the number of samples that could be utilized to train the neural network.

Table 4.1. The number of samples per class.

Class	Number of images	Age (years)
COVID-19	257	53.29 ± 16.72
Pneumonia	322	49.73 ± 19.66

Cohen/IEEE 8023 dataset [11] is a collection of X-ray images that have been extracted from online publications, websites, as well as through indirect collection from hospitals and physicians. The dataset contains several pathogen types, including, but not limited to, COVID-19, Pneumonia, SARS. Furthermore, textual clinical data such as age and gender are also present in the dataset. Only COVID-19 was used in this study from the previously mentioned dataset due to the small number of samples in other classes such as pneumonia (less than 30 images).

The NIH Chest X-ray dataset [12] is used as a source for pneumonia images and comprises 112,120 images from 30,805 unique patients. The dataset contains X-ray images of 14 different pathogen types; the dataset also contains textual clinical data such as age and gender for each image. The employment of this dataset mitigates the lack of specific pneumonia data in Cohen/IEEE 8023 dataset.

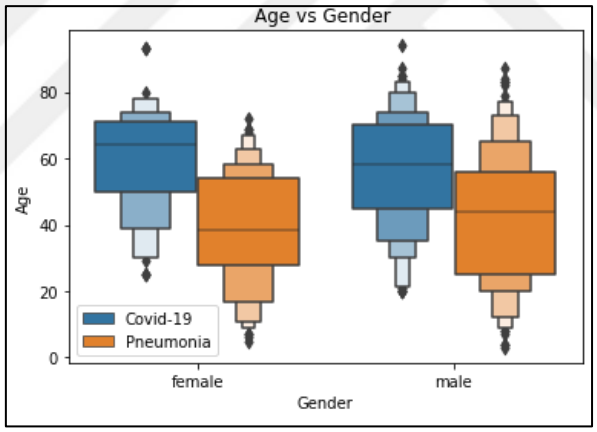


Figure 4.2: COVID-19 and Pneumonia distribution in the dataset, considering age and gender.

For a more straightforward explanation, the data can be split into image and textual clinical data (Clinical and lab testing data). For Image data, Only X-ray images with PA and AP views are employed. Two images of each patient are processed if available; the difference in images is the time between each scan. The training set is rescaled while the augmentation is randomly applied to achieve higher generalization.

The dataset includes patient textual clinical information such as age, gender, and several other variables for numerical and categorical data. Due to the scarcity of data, this work opted to use only age and gender from the textual data. Data augmentation has not been used on the textual data. Table 4.1. summarizes the data used. Each data point used in training and testing this model

contains two images with clinical data for each patient; if the patient has a different image, it will be regarded as a new data point.

4.2 DATA PRE-PROCESSING

Before the image and textual data are fed to the hybrid network. the dataset must be processed in a way that enables the network to learn from it effectively. First, the data is split into two subsets training and validation(testing) sets, while it is ideal to split the dataset into three sets, the small number of samples eliminated the use of this option. For the image preprocessing pipeline as depicted in Figure 4.3. The pipeline consists of six steps, initial transformation is first, in this step the X-ray image is scaled and cropped while maintaining the aspect ratio of the original image. After the transformation, the X-ray image is fed into the segmentation model and the lungs are extracted, this step will be discussed in detail in later sections. After segmentation of the lungs, the color space transformation is carried out, which transform the images to grayscale, if the source image is not in the required color space (grayscale), thus standardizing the images in the dataset regarding the color space. Then a final scaling of the X-ray images is carried out to scale the image to the network requirement. Histogram equalization is applied next, which enhance the contrast of the X-ray image. Finally, in the last step an augmentation transforms the image randomly and generate images that differ from the original images, this step is only applied for the training set.

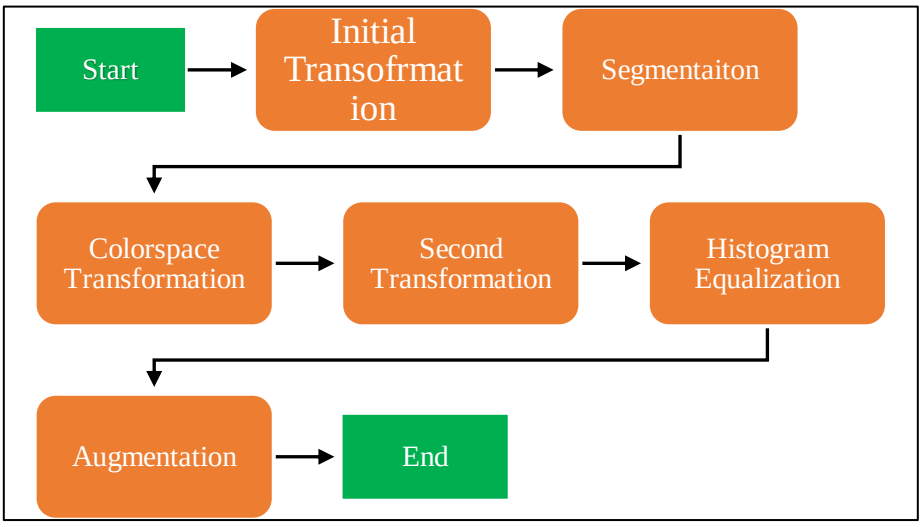


Figure 4.3: Flowchart of the image preprocessing pipeline.

4.2.1 Image Transformation

The images in both datasets are not standardized in terms of size and scale. Some of the X-ray images in both have different sizes, this attributed to the different standards that were flowed during the creation of said datasets. Image transformation step is required for the segmentation model to extract the lungs effectively. The X-ray images are scaled to the size 1024 x 1024 in the effort of preserving most of the details, while the aspect ratio is preserved, the preserving of the aspect ratio of an image so the final output is true to the original image and to not introduce an uncontrolled variable that may affect the reliability of the classification. X-ray images were transformed to uncompressed grayscale png files; as discussed earlier, only PA and AP views were considered in the experiments. In the second transformation, the images were merely resized to 256 x256.

4.2.2 Segmentation

For lung image analysis, automated segmentation of lung X-ray s is a vital first step; it is essential for accurate lung X-ray image analysis, such as lung cancer diagnosis and COVID-19 detection. Lung segmentation algorithms' performance and reliability depend mainly on the diversity of the training data. This study employs a pre-trained model based on ResNet34 to extract lungs from X-ray images; the pre-trained model achieved a Dice coefficient of 96.57%. Figure.6 shows the lungs after segmentation, which is fully evident after transformation.

In The segmentation step, the X-ray images are fed to a segmentation model that is based on a ResNet34 architecture. As discussed earlier, segmentation is a necessary initial step for lung image analysis, it is required to achieve classification accuracy that is not affected by superficial features(bias). The transformed X-ray images are fed into the segmentation model that takes the X-ray images as an input and outputs a binary mask. The generated mask is subtracted from the transformed image to exclude the regions of the lungs that the mask does not cover. For this study, the mask and newly segmented lungs are saved as png files.

Initially, images were mainly cropped to minimize and remove undesirable annotations evident in the image data but after studying the images more in detail, image segmentation was used 'and is necessary for medical image analysis to avoid bias from unwanted features. Figure 4.4 shows the process of segmentation.

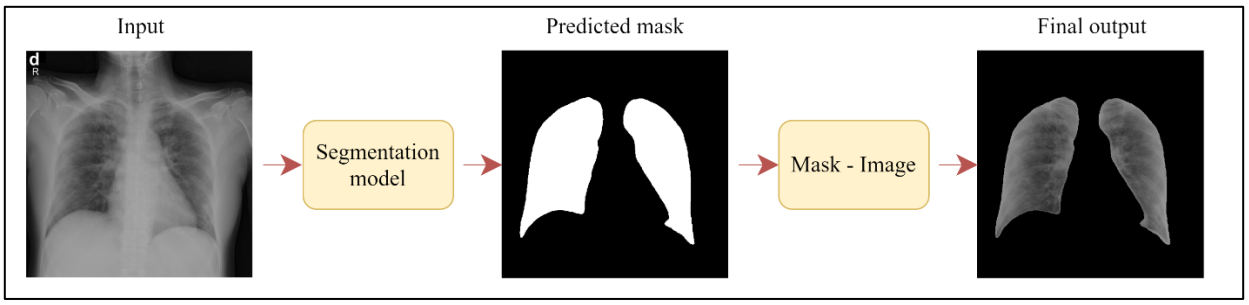


Figure 4.4: The process of X-ray segmentation**.**

4.2.3 Histogram Equalization

This technique is essential for the standardization of the image data, due to the difference in data collection and images sources. In other words, the intensity contrast across nearby areas on a plain radiograph of an X-ray image is determined by various factors, notably patient contrast, receptor contrast, and scatter radiations.

Furthermore, different calibrations might result in a significant difference in the dynamic range with the risk of bias. Before it can be utilized as a data source, the raw image data is processed using Contrast limited adaptive histogram equalization (CLAHE) which is a variant of Adaptive histogram equalization (AHE).

CLAHE is employed to enhance the contrast and improve the visibility level of the foggy image of the X-ray image in the dataset. Typical histogram equalization techniques amplify noise and introduce unwanted artefacts that may affect the classification reliability. CLAHE mitigate this problem that arises from using the mentioned technique. CLAHE is a straightforward technique to ensure that the data has a constant image dynamic. Figure 4.5 shows the difference between the CLAHE and histogram equalization, it is evident that the CLAHE performs better and shows details that histogram equalization overexposed.

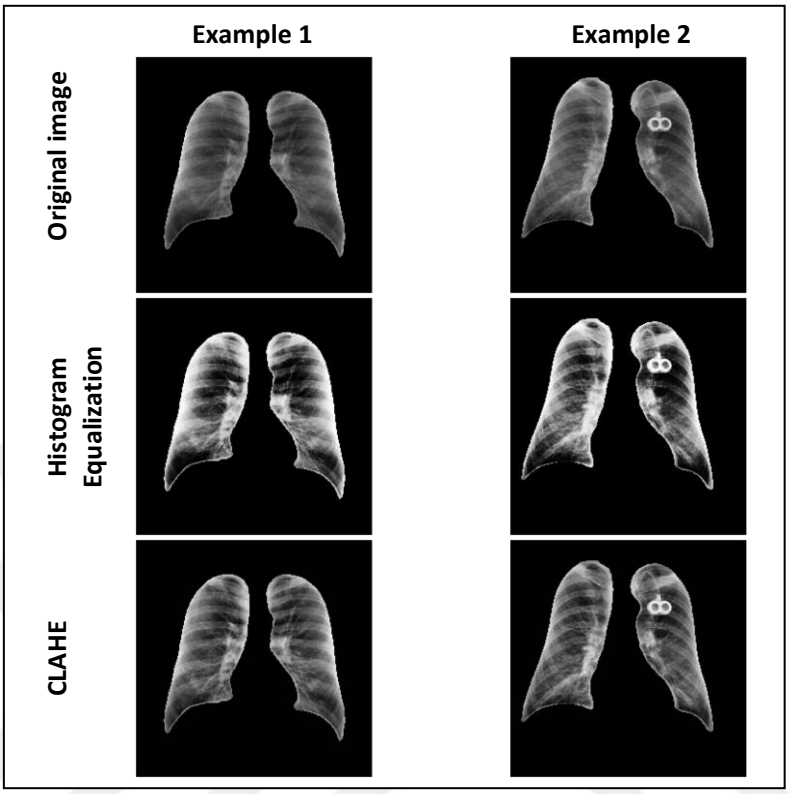


Figure 4.5: comparison between the grayscale equalization and CLAHE used**.**

4.2.4 Image Augmentation

Augmentation is a technique for artificially altering existing image data to produce additional data for training, with the goal of reducing the likelihood of the same image being utilized multiple times. This approach aids the network in addressing the barriers that limited datasets in this study present, which in turn result in improved classification performance in unseen data. In this study, and due to the different data types, only image data is augmented during training.

As discussed in section 4.2, modest augmentation is applied on X-ray images to train the neural network. The following augmentations techniques are used: random horizontal flip and rotation, elastic deformation and zooming. Each image is square cropped to maintain aspect ratio and resized to 256 × 256 pixels. Figure 4.6, shows the different augmentations used in this study.

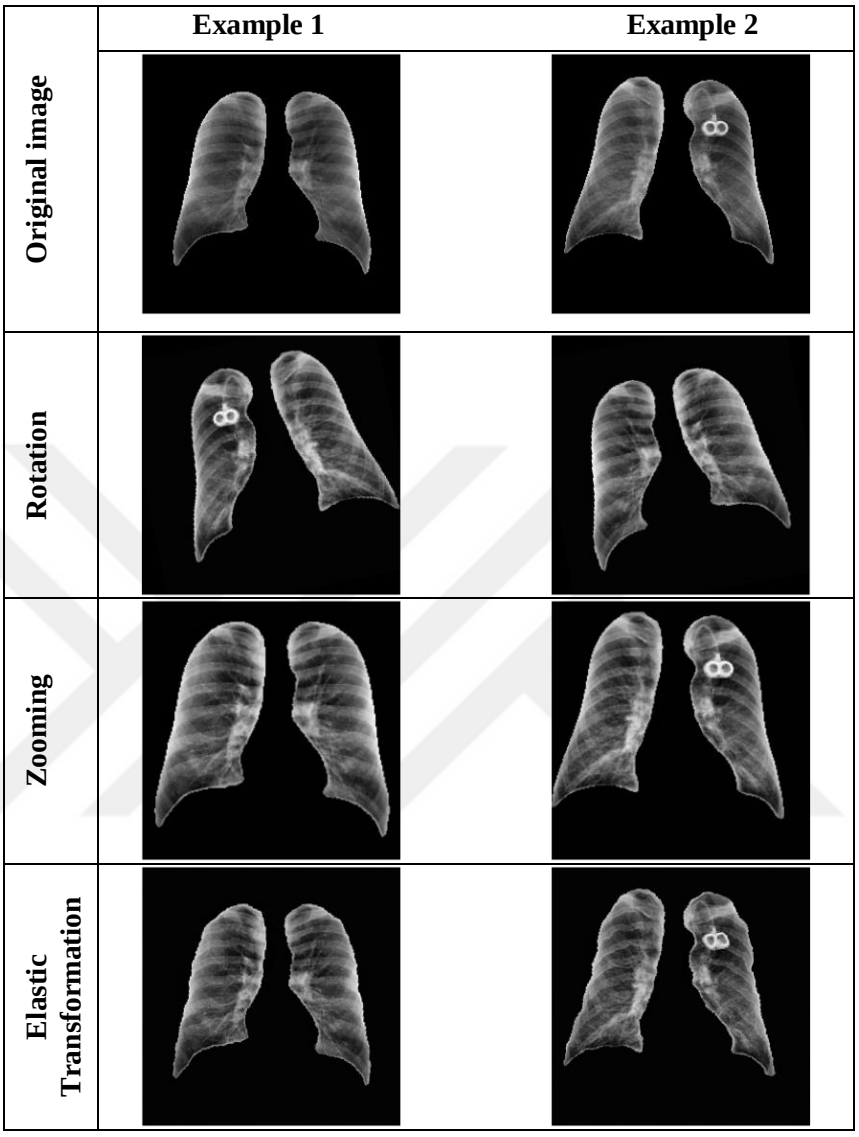


Figure 4.6: Examples of different augmentation used.

4.2.5 Textual Data Preprocessing

For textual clinical data in the (CSV) file, the data were normalized between 0 and 1, while the series data (i.e., age) was categorized based on groups.

4.3 THE PROPOSED HYBRID NEURAL NETWORK

Due to the high similarity among X-ray images, a large sample size is needed to train the network in the classification task. However, as discussed in section 4.1, a limited amount of

COVID-19 datasets containing X-ray images with textual clinical data. To mitigate this problem multibranch neural network is proposed to increase the reliability of the classification.

In this study, the proposed hybrid neural network architecture, as shown in Figure 4, consists of two CNNs and an MLP network. CNN network contains two branches and two sub-branches connected to each branch. The input layer contains artificial input neurons, which is the first layer of each neural network that specializes in providing the basic information of the initial data for subsequent layers such as the data size and the dimension of the data. For this study, three input layers were employed, two for the image data and one for the textual clinical data. A hidden layer in a neural network, located between the input and output layers and called hidden because it cannot be observed immediately from the system's inputs and outputs; the hidden layer usually contains zero or more neurons. In this study, convolutional layers are used in conjunction with regularization and dense layers to achieve the desired output. For convolutional layers, each layer is coupled with a regularization layer, i.e., batch normalization and a reduction layer; in this case, a Maxpooling2D was used to subsample the images, while the dense layers are used after each branch as shown in Figure. 4.7 with the selected activation function. Classification or regression, each neural network must have an output layer with a specific function to decide how to process the output for prediction or regression. In this study. The last dense layer (output) has a special activation function in this case SoftMax activation function is employed for the classification task.

The input for the CNN accepts image data of size 256 × 256 pixels with channels last ordering. Sub-branches contain three CNN blocks, and each is built using a 2D Convolutional layer, batch normalization and Maxpooling: all of the 2D Convolutional layers employees ReLu activation function. The features extracted from the X-ray images using CNN are concatenated with the MLP network's output, the resulted data is flattened and fed to the fully connected layers for prediction. The neural network was built using the Keras library and Tensorflow.

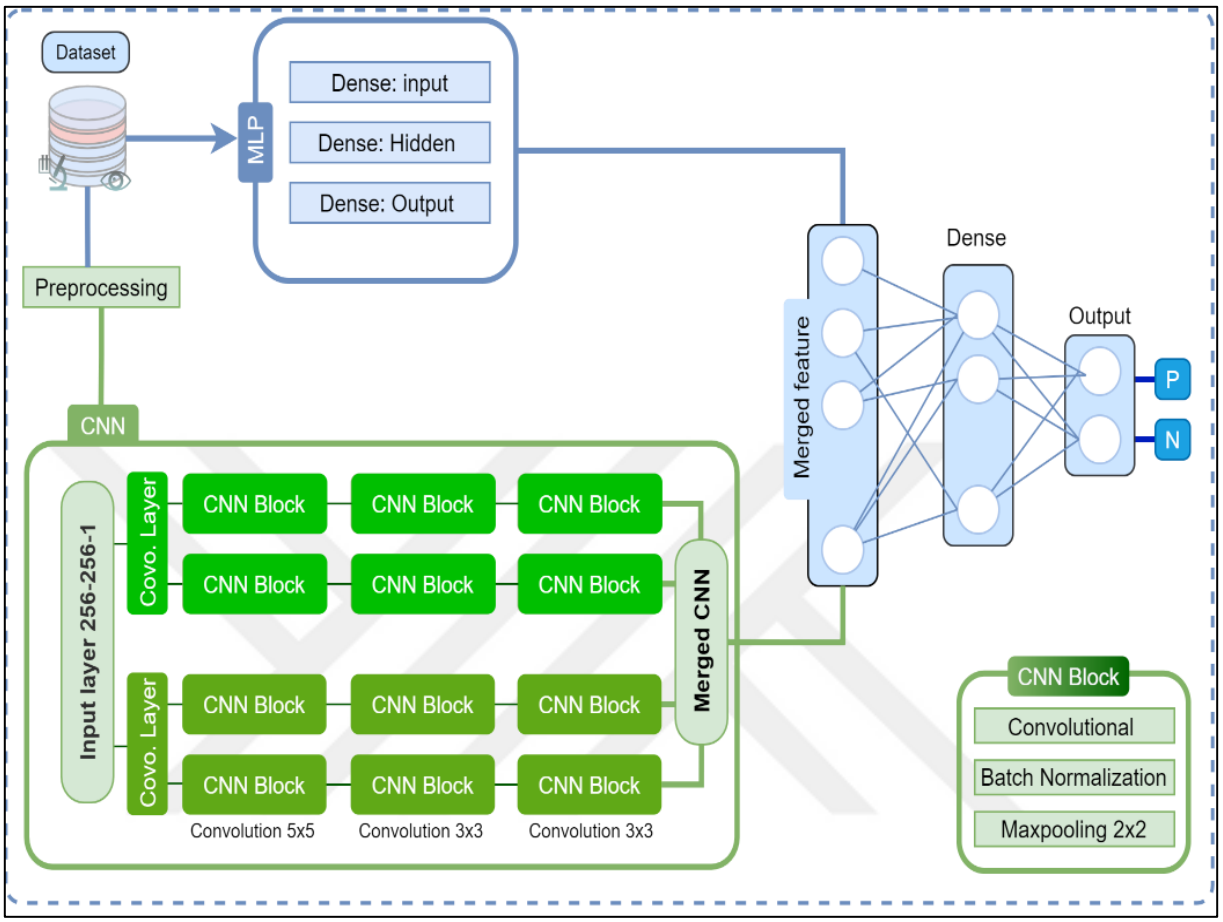


Figure 4.7. Proposed architecture for CNN-MLP network, illustrating both CNN network and MLP network.

The hybrid network weights are initialized using the modified EAEO algorithm to serve as a good starting point for training. The network is fitted to the data using Gradient Descent with Backpropagation. Adam optimizer is employed with learning decay: the learning rate decays if a monitored variable stagnates during training for a specified number of epochs (i.e.,' patience'). The following are the network hyperparameters and are selected using grid search: epochs = 100, batch size = 16, initial learning rate = 0.001, decay factor = 0.96 and patience = 5.

4.3.1 Focal Loss

A brief introduction to loss function is required before moving on to the focused loss. The loss function is a technique of evaluating how effectively the algorithm models the data by

computing the distance between the actual output of the algorithm and the predicted output. It translates decisions to their associated costs.

Focal loss for binary classification[51] is employed to address the class imbalance that is present in the dataset. binary focal loss is an improved version of Cross-Entropy Loss (CE), which introduce a parameter, called focusing parameter (γ), that allows instances that are difficult to classify to be penalized more severely than instances that are easy to classify. The focal loss is defined as

$$L(y, \hat{p}) = -\alpha y(1-\hat{p})^{\gamma} log(\hat{p}) - (1-y)\hat{p}^{\gamma} log(1-\hat{p}) \quad (4.1)$$

where y $\in\{0,1\}$ is a binary class label, $\hat{p} \in[0,1]$ is an estimate of the probability of the positive class, γ is the focusing parameter, α is a hyperparameter that governs the trade-off between precision and recall.

4.4 WEIGHT OPTIMIZATION

The use of metaheuristics algorithms to explore the search space for a better value based on a set of pre-defined steps; modified EAEO in Algorithm1 shows the pseudo-code for the proposed algorithm operations, that systemically improves parts that are deemed to have poor performance based on calculating the error value.

4.4.1 Enhanced Artificial Ecosystem Optimization

Enhanced Artificial Ecosystem Optimization (EAEO) is a nature-inspired metaheuristic based on Artificial Ecosystem Optimization (AEO) [52]. EAEO is a population-based optimizer that mimics the flow of energy in an ecosystem in three phases (production, consumption, decomposition) which are mimicked from the unique behaviour of the living organism in an ecosystem. A modified EAEO Algorithm is employed to optimize the weights of the proposed CNN network to increase the performance, as shown below.

CNN weight extraction is carried out when the trained weights are taken as an individual then vectorized for the initial population of MEAEO. Consequently, a super-individual Is created, and the search for optimal value may be hindered; to solve this, EAEO employs

Algorithm 1: Modified EAEO

input : Population size N, and the maximum number of generations $MaxGeneration$
output: the global best pop_i solution
Initialize population of solution from layer weights
Calculate the fitness values and sort the population
Find the best solution from the population $GlobalBest$
for $i \leftarrow 0$ **to** $MaxGeneration$ **do**
  Production:
  Calculate X_1 based on Eq. (4.4)
  Consumption:
  for $i \leftarrow 2$ **to** $MaxPopulation$ **do**
    Calculate $solution_i$ position using on Eq. (4.7) , (4.8) and (4.9).
    $LocalBest_i = solution_i$ position
  end
  for $i \leftarrow 0$ **to** $MaxPopulation$ **do**
    Calculate $LocalBest_i$ fitness using Eq. (4.2).
    if $LocalBest_i > GlobalBest$ **then**
      $GlobalBest_i \leftarrow LocalBest_i$
    end
  end
  Find $GlobalBest$ based on Max fitness value for Decomposition
  Decomposition:
  for $i \leftarrow 0$ **to** $MaxPopulation$ **do**
    Calculate $solution_i$ position using Eq. (4.11).
    $LocalBest_i = solution_i$ position
  end
  for $i \leftarrow 0$ **to** $MaxPopulation$ **do**
    Calculate sloution fitness $LocalBest_i \leftarrow i$ using Eq.(4.2).
    if $LocalBest_i > GlobalBest$ **then**
      $GlobalBest_i \leftarrow LocalBest_i$
    end
  end
end
Return $GlobalBest_i$

randomness in the mutation of individual position. In addition, to improve the optimization efficiency of MEAEO, after each generation, a random shuffle of data is employed to avoid misdirection of the search when given orderly data. At each iteration, the fitness function calculates the best individual using (2).

$$Fitness = 1 - Accuracy \tag{4.2}$$

The update stage consists of three phases: production, consumption, and decomposition. In the production phase, the worst individual is replaced with a newly created individual based on the values of the best individual and another random individual. This phase can be described mathematically as [10]:

$$G = 2 \times \left(1 - It / MaxIt\right) \tag{4.3}$$

where *MaxIt* is the maximum number of iterations, *It* is the current iteration, The operator *G* is added in the production phase [10]

$$x_2(t+1) = x_2(t) + G \times C \times [x_2(t) - x_i(t)] \tag{4.4}$$

where, *t* is the position of the current individual, *C* operator is Levy flight, which is used to enhance the exploration level. The Levy flight can be described as [10]:

$$C = \frac{0.5 \times v_1}{|v_2|} \tag{4.5}$$

where,

$$v_1 \sim N(0,1), \quad v_2 = N(0,1) \tag{4.6}$$

where *N(0, 1)* represents a normal distribution.

In the consumption phase. The new individual is updated, which can improve the search for a better solution. There are three types of consumers:

i) Herbivore type
ii) Carnivore type
iii) Omnivore type

The three consumer types can be represented using the sine-cosine function as follows [10]respectively.

$$\left\{ \begin{array}{l} for\ r \le 1/3, r_4 \le 0.5 \\ \quad x_i(t+1) = x_i(t) + sin(r_3) \times C \times [x_i(t_3) - x_1(t)] \\ for\ r_4 > 0.5 \\ \quad x_i(t+1) = x_i(t) + cos(r_3) \times C \times [x_i(t3) - x_1(t)] \end{array} \right\} \tag{4.7}$$

$$\left\{\begin{array}{l} for\, 1/3 < r \le 2/3, \quad r_4 \le 0.5 \\ \quad x_i(t+1) = x_i(t) + sin(r_3) \times C \times X_{ij}(t) \\ for\, r_4 > 0.5 \\ \quad x_i(t+1) = x_i(t) + cos(r_3) \times C \times X_{ij}(t) \\ j = randi([2\ \ i-1]), \\ X_{ij}(t) = x_i(t) - x_j(t) \end{array}\right\} \tag{4.8}$$

$$\left\{\begin{array}{l} for\, 2/3 < r \le 1, \quad r_4 \le 0.5 \\ \quad x_i(t+1) = x_i(t) + sin(r_5) \times C\ [r_5\, X_{i1}(t) + (1-r_5) X_{ij}(t)] \\ for\, r_4 > 0.5 \\ \quad x_i(t+1) = x_i(t) + cos(r_5) \times C\ [r_5\, X_{i1}(t) + (1-r_5) X_{ij}(t)] \\ j = randi([2_i - 1]), \\ X_{i1}(t) = x_i(t) - x_1(t) \end{array}\right\} \tag{4.9}$$

$$r_3 = 2\pi \times rand,\ \ r_5 = rand \tag{4.10}$$

In the Decomposition phase, decomposition coefficients are utilized for a faster decay of the weights; as a result, the decomposition improves the exploitation of the EAEO algorithm. "Based on the decomposer x_n, the position of the individual x_i in the population can be updated" [10] and can be represented as follows [10], [52]:

$$x - i(t+1) = x_n(t) + D[ex_n(t) - hx_i(t)] \tag{4.11}$$

$$D = 3u,\ \ u \sim N(0,1) \tag{4.12}$$

$$e = r_3 \times randi([1\ 2] - 1), \qquad h = 2r_3 - 1 \tag{4.13}$$

where, *D* is the factor for the decomposition coefficients, *e* and *h* are the weights and r_3 is a random float between [0, 1]

The network weights will be replaced in the fitness evaluation stage if the optimized weights are better than the original network weights. A flowchart of the optimization is depicted in Figure 4.8., which illustrates the weight of the CNN+MLP networks that are concatenated and fed to the optimization algorithm.

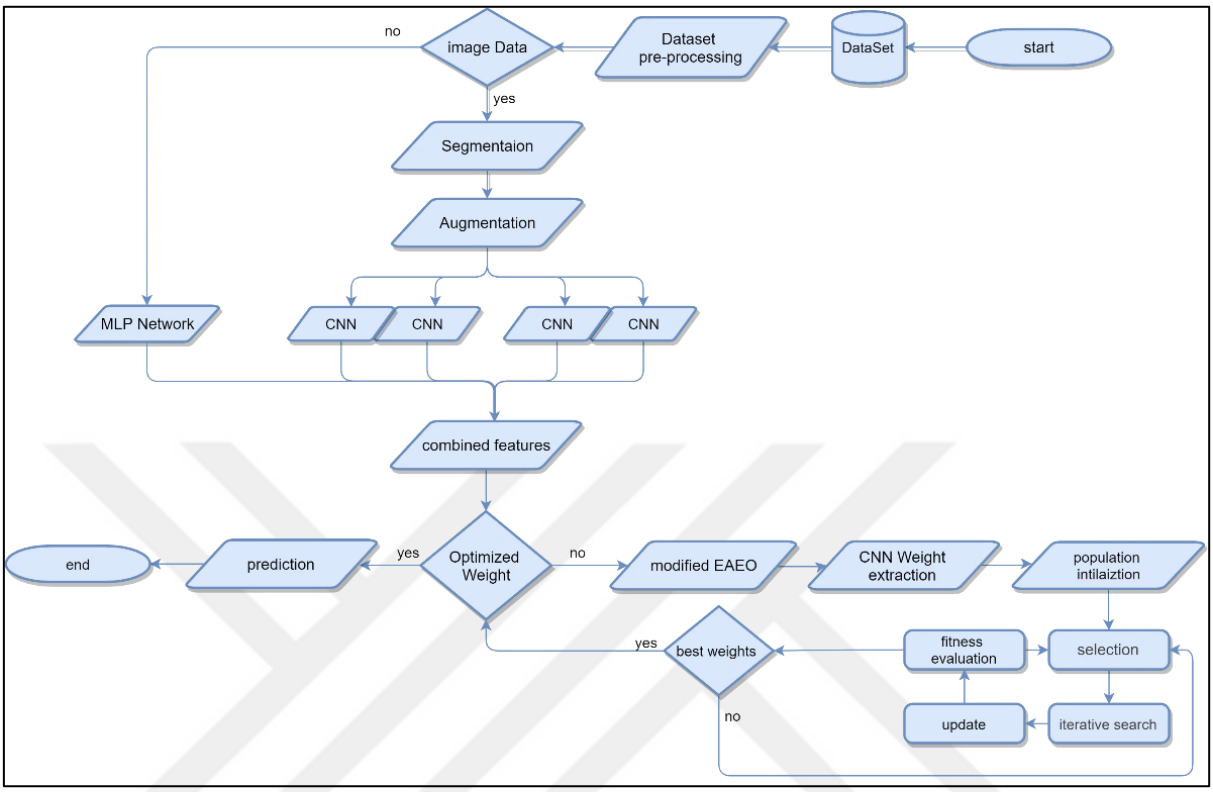


Figure 4.8: Flowchart of the proposed framework, in which the modified EAEO is incorporated.

4.5 TRAINING OF THE HYBRID NEURAL NETWORK

For the training phase. The model hyperparameters are initialized with values shown in Table 4.2. It is essential to note that multiple Graphical processing units (GPUs) are employed in this phase.

The process of training is significantly faster while training using GPUs, but an unavoidable issue while using GPUs is the reproducibility of the results and it may affect the result to some extent, Due to the random nature of the methods used.

A solution implemented in this study to help mitigate this phenomenon is the use of seeds in the random generators thus the randomness is controlled.

The seeds for the random generators are set pre-training and the values are also shown in Table 4.2.

Table 4.2: Hyperparameters values used for training.

	Hyperparameter	**Value**
Model	CNN - Input size	256×256
	MLP – Input size	2
	Random Seed	1337
Learning rate	Initial learning rate	1e-3
	Decay factor	0.5
	Decay steps	900
Training function	Batch size	16
	Epochs	100

4.6 PERFORMANCE METRICS

performance metrics are used to provide an overview of the model performance in a task based on the model prediction values and true data values. This study employs a variety of performance measures, including Accuracy, F1-Score, Precision, Recall, and confusion matrices. Accuracy will be the primary metric for this study, and graphs will be used to show the results whenever feasible.

Accuracy metric

This metric Calculates how often predictions equal labels. This metric creates two local variables, total and counts that are used to compute the frequency with which the output value of a model matches the supplied actual value. This frequency is returned as binary accuracy.

Confusion Matrix

Confusion Matrix is an important metric that describes the performance of the classifier on the test dataset. In other words, Confusion Matrix shows the values of correct and incorrect

prediction for each class(COVID-19 class – pneumonia class), where each row represents the number of samples of a class while columns represent the predicted samples in a class.

F1-Score

F1-Score is another measure that has been used in this study, F1-Score combines the precision and recall performance of the classes, where the score reaches its best value at 1 and worst at 0. The combination of recall and precision makes the F1-score sensitive to outliers in single classes. Due to the nature of the dataset used this metric represents the true performance of the classifier.

Precision is defined as the ability of a classifier to return the relevant data point. Precision measures the quality of the classifier in a task.

Recall is defined as the number of correctly classified samples by the classifier. Recall indicates the false-positive prediction that a classifier returned in the course of the evaluation.

These metrics are used in unison to calculate the F1-score, and may be expressed numerically as follows:

$$precision = \frac{\sum TP}{\sum TP + FP} \tag{4.14}$$

$$recall = \frac{\sum TP}{\sum TP + FN} \tag{4.15}$$

$$F1score = \frac{2 \times precision \times recall}{precision \times recall} \tag{4.16}$$

4.7 PROGRAMMING LANGUAGES AND FRAMEWORKS USED

The system proposed consists of several architectures, CNN, MLP and genetic algorithms. Taking advantage of the increase in the computational power provided by the parallel computing units (Cuda) in the GPU, by using Tensorflow as a backend for Keras are the main components to build the neural network part like CNN and MLP. all of the experiments in this work are done by employing google collaborator. At the same time, the visualization of the hyperparameters tuning and the final results by using “weghtandbisess”.

4.7.1 Python

several programming languages have been considered to implement this system, like Python, R and C#. Nevertheless, the python programming language provided outstanding characteristics. One of the most considerable advantages of python is the support of different libraries that assist in developing complex interconnected systems with little effort.

4.7.2 Pandas data frame

As discussed earlier, this system uses numerical and categorical data as input. As a result. Pandas' data frame has been chosen to process the data associated with clinical findings. Also, pandas' data frames have been used to obtain the different sets like training and validating sets.

4.7.3 Keras

Keras is a high-level Tensorflow API. It provides simple tools that, when used with Tensorflow as a backend, may yield unrivalled results. Keras also allows the creation of neural networks in sequential mode or Functional API. Functional API can and will be used to build the hybrid model utilized in this work. It also allows to link each layer to a different layer regardless of the layers' sequence; this enabled the construction of a neural network with different connections for each layer. All of this makes Keras suitable for the creation of various neural networks. The MIT license is used for Keras distribution. It is also compatible with various Python versions.

5. RESULTS AND DISCUSSION

The performance of the proposed hybrid network and the result from the different experiments are presented in this chapter, in addition, a comparison between the performance of the proposed hybrid network and several other networks in the literature in terms of classification accuracy is also presented.

The result in this chapter is obtained from the evaluation of the proposed network against the test dataset, in other words, the result represents the performance of unseen data. A five-fold cross-validation technique has been utilized to average the model's performance using an independent testing set. Table 5.1, in section 5.2, summarizes the averaged results obtained from the experiments that have been performed in this study. Likewise, the confusion matrices are presented in Figure 5.2, in section 5.2, which compares the actual target values with those predicted by the proposed model.

5.1 THE IMPROVEMENTS OF MERGING SEVERAL DATA TYPES

In this study, different data types were used to train the proposed hybrid network. This section discusses the impact of combining multiple data types, such as images and textual data. this approach provides advantages over the typical learning setup, such as decreased overfitting, and rapid learning due to the use of auxiliary information i.e., age, gender. Table 5.1 compare the results obtained from merging data types and the result of this test. While using only images the accuracy dropped significantly. In conclusion, the importance of merging several data types is undoubtedly beneficial in getting better performance.

Table 5.1: Performance of the different types of data used.

Data type	Class	Precision	Recall (sensitivity)	F1	Accuracy
Images only	COVID-19 Pneumonia	80 89	86 83	83 86	84
Images + Textual data	COVID-19 Pneumonia	94 94	95 93	94 94	94

5.2 HYPERPARAMETERS EVALUATION

This section investigates the influence of different hyperparameters on the performance of the proposed hybrid network, tests were carried out with different learning rates, and the activation functions that were discussed in Chapter 3. Figure 5.1 shows a general depiction of parrel coordinates of the Hyperparameters selection process.

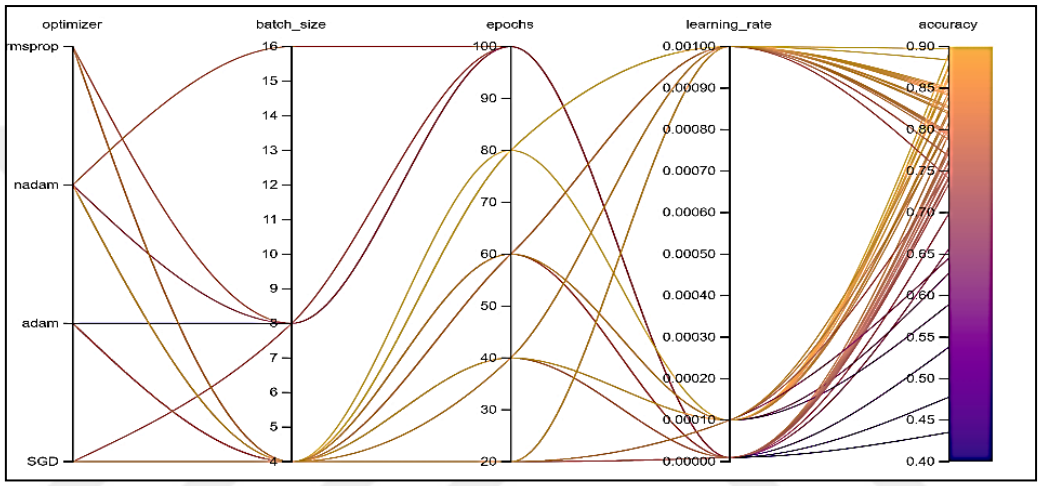


Figure 5.1. Parallel coordinates of Hyperparameters selection

5.2.1 Initial Learning Rate

As discussed in Chapter 4, the learning rate decay is employed, which decreases the rate of learning over time allowing the model to arrive at the best solution optimally. The initializing of the learning rate for the function of decay is an important task. If the learning rate was set to a high value, the model may overfit, while if a low value was set the training will progress slowly and increasing the training time exponentially. the best learning rate to initialize the network, in this case, was 1e-3.

5.2.2 Activation Function Selection

While ReLu is the defacto standard and for a good reason, the use of other activation functions must also be investigated, in this study, as discussed in Chapter 4, four activations functions were tested ReLu, swish, tanh and, SeLu. Figure 5.1 shows the performance of the

activation functions. While SeLu has a comparable result to ReLu, ReLu has a slight advantage and the choice of using this activation function was made.

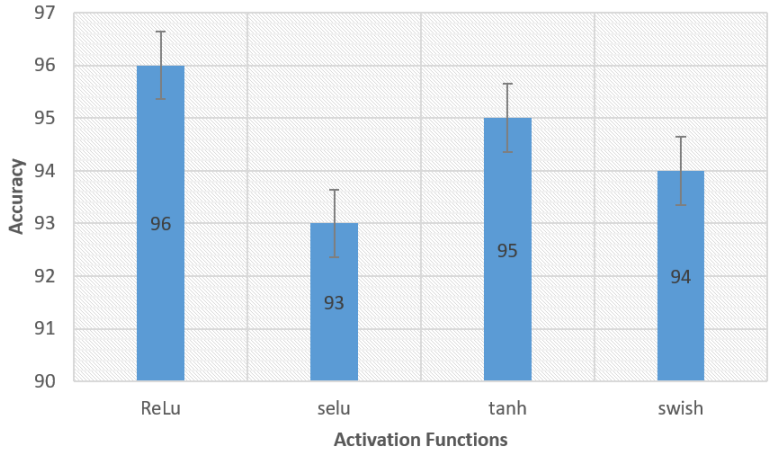


Figure 5.1: Performance of different activation functions in terms of accuracy.

5.3 OPTIMIZATION AND SEGMENTATION EXPERIMENTS

The result of weight optimization and image segmentation results will be discussed in this section. All of the experiments will employ weight optimization as a secondary stage. This section experiment employs performance metrics that have previously been discussed in section 4.7 to evaluate the performance for each class in the testing set. Due to the imbalanced classes (number of samples in each class) in the dataset, the focal loss has been employed to mitigate the bias from the inherited issue of the problem. Table.5.1. shows that optimizing the weights using a metaheuristic algorithm is beneficial. As an outcome, the experiments that employ weight optimization have higher scores; utilizing weight optimization is advantageous in accurately predicting the classes. However, the gap between the optimized and non-optimized experiments is not significant in terms of accuracy. The best accuracy obtained from the experiments while using the MEAEO for weight optimization is 97.85% compared to 94.23%; The result pointed out an even prediction distribution of the classes compared without optimizing the weights; the remaining metrics follow the same trend. It is important to note that each experiment uses the same hybrid neural network discussed in Chapter 4 and only differs in the pre-processing steps, which addresses the issues of superficial features that the network learns from.

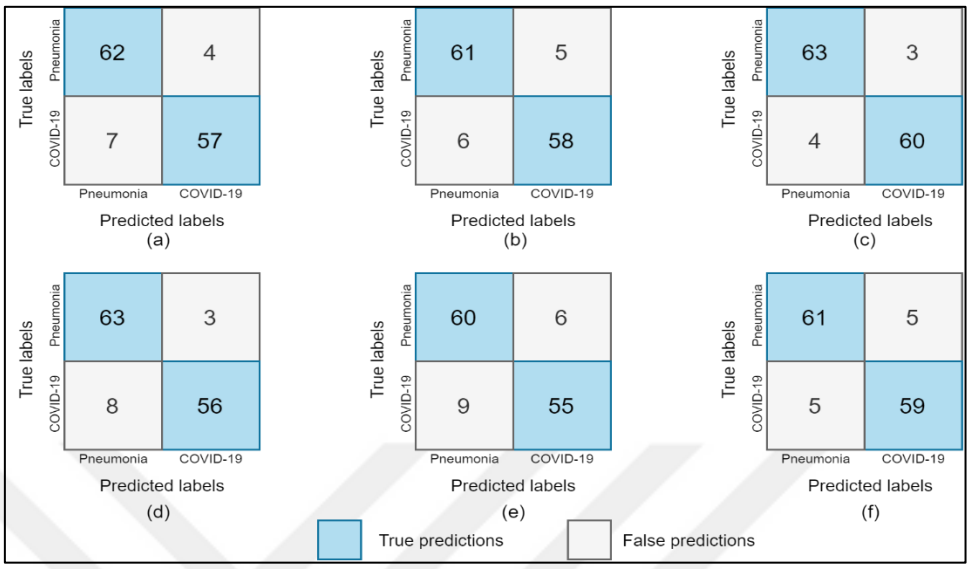


Figure 5.2. Confusion matrices for each of the experiments, considering each of the classes separately Top: with Weight optimization. Bottom. Without Weight optimization

Table 5.2. The general performance of all experiments.

Exp.	Class	Measures | CNN+MLP EAEO (NN-MEAEO)				Measures | CNN+MLP (Base)			
		Precision	Recall (sensitivity)	F1	Accuracy	Precision	Recall (sensitivity)	F1	Accuracy
Exp. 1	COVID-19	92.15 ± 1.58	92.33 ± 0.18	92.22 ± 1.32	91.21± 2.98	93.21 ± 1.02	86.52 ± 0.11	89.68 ± 0.28	91.37 ± 3.68
	Pneumonia	91.34 ± 0.43	91.25 ± 1.28	91.41 ± 0.73		88.88 ± 0.73	94.45 ± 1.00	91.61 ± 0.67	
Exp. 2	COVID-19	86.73 ± 1.20	91.75 ± 0.35	91.41 ± 0.26	91.24± 4.52	87.87 ± 0.00	92.42 ± 1.31	89.63 ± 0.48	89.50 ± 5.73
	Pneumonia	94.32 ± 0.75	91.11 ± 1.75	91.92 ± 0.96		91.91 ± 1.05	85.75 ± 0.28	88.88 ± 0.73	
Exp. 3	COVID-19	95.42 ± 0.88	**99.05 ± 0.50**	97.07 ± 0.65	**97.85 ± 3.86**	94.62 ± 1.23	95.65 ± 0.25	94.85 ± 0.82	94.23 ± 2.25
	Pneumonia	**99.50 ± 0.50**	93.37 ± 0.75	**97.17** ± 0.80		94.44 ± 0.50	93.13 ± 0.74	94.44 ± 0.18	

Results And Observations from Experiment 1

For this experiment, the model is trained using image augmentation without segmentation or cropping, while CLAHE is employed regardless. Image and clinical information are provided as input to the neural network. Without weight optimization, the detection performance varies for all classes (the Precision ranges from 91-93%), as shown in Table 5.1 while using the modified EAEO for weight optimization, an increment is observed in sensitivity for detecting COVID-19 (the Precision ranges from 88-93%). Sensitivity in Table 5.1 for all classes reaches a value of more than 90%, which can be considered outstanding. At the same time, the unoptimized weights show lower accuracy, as shown in Table 5.1 and Figure 5.2. Regarding

general performance, sensitivity values reached 92% with the weight optimization and 86% without weight optimization, also supported by the sensitivity values. In the confusion matrix, Figure 5.2 shows that a small number of samples are misclassified in the COVID-19 class and can be attributed to weight optimization. Figure 5.3 compares the result of the proposed method with other evolutionary algorithms, shows that PSO achieved the lowest in all tests. While the genetic algorithm for weight optimization resulted in better accuracy than PSO, an improvement is observed between AEO and EAEO in terms of accuracy and sensitivity.

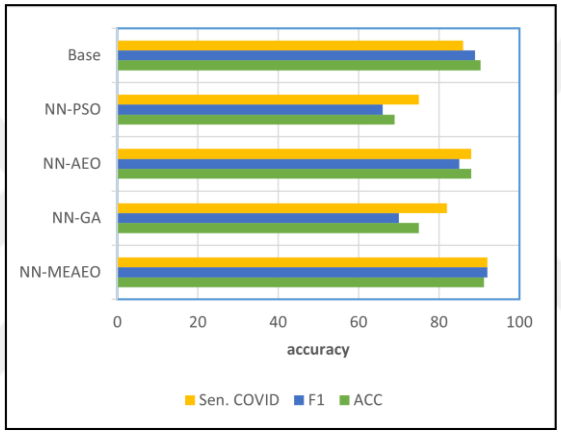
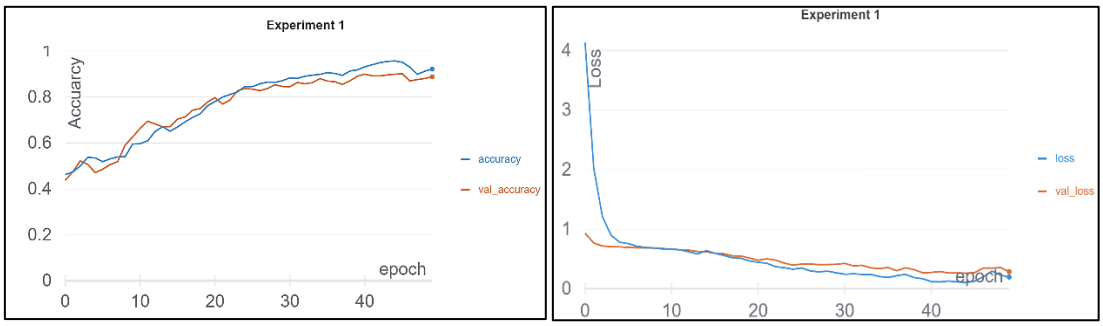


Figure 5.3. Performance of the optimization algorithms used in experiment 1.

The development of accuracy and loss during training and evaluation is depicted in Figure 5.4.while the F1-score increment is shown in Figure 5.5.



(a) **(b)**

Figure 5.4: (a) Accuracy and validation Accuracy during training for experiment 1, (b) loss and validation loss for experiment 1.

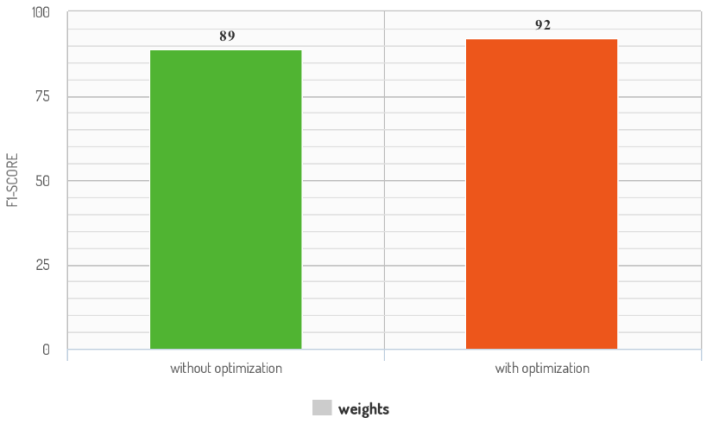


Figure 5.5: F1-score for experiment 1 considering the weight optimization.

Results And Observations from Experiment 2

Experiment 2 applies image cropping and CLAHE, image augmentation was also used, which are employed as discussed in 4.2. The result from experiment 2 shows that experiment 1 outperformed this experiment in terms of accuracy; this indicates that the system learns artificial features outside the interest region, i.e., the lungs and can be attributed to the varied equipment types that are used to generate the X-ray image of the patient, and perhaps even the different annotations added by the medical staff to the image. The detection performance increment while using weight optimization is in line with experiment 1.

Using the proposed optimization approach, the result in Table 5.1, shows the Precision values ranging from 86% to 91%, compared to 85% to 87% without utilizing the proposed approach for weights optimization. While Sensitivity values in Table 5.1 ranging from 85% to 92%. The overall performance while using the modified EAEO for weight optimization reached an accuracy of 91.24% and 89.50% without optimizing the weights. Figure. 5.6 shows the performance of the different optimization algorithms used in this experiment.

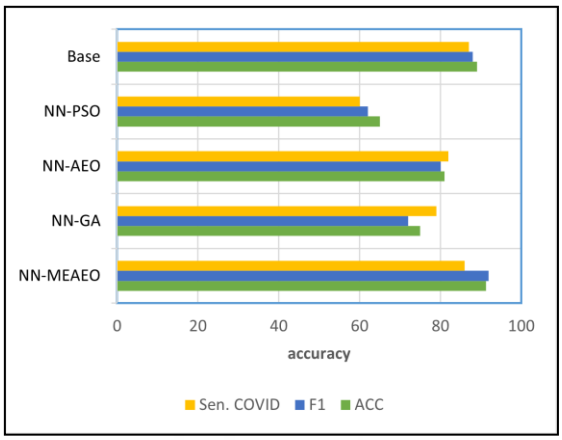
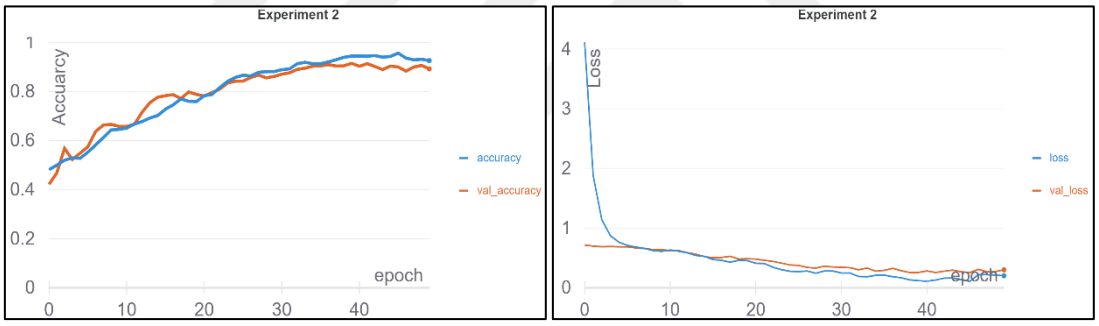


Figure 5.6: Performance of the optimization algorithms used in experiment 2.

The development of accuracy and loss during training and evaluation is depicted in Figure 5.7.while the F1-score increment is shown in Figure 5.8.



(a) **(b)**

Figure 5.7: (a) Accuracy and validation Accuracy during training for experiment 2, (b) loss and validation loss for experiment 2.

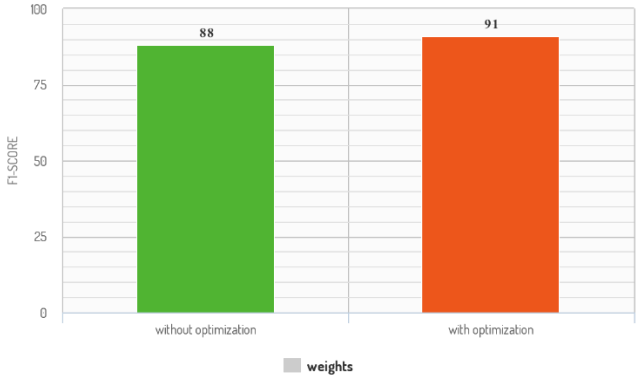


Figure 5.8: F1-score for experiment 2 considering the weight optimization.

Results And Observations from Experiment 3

Finally, for Experiment 3, this experiment employed lung segmentation, image augmentation, and CLAHE, as discussed in section 4.2. The network was forced to focus on the lungs by using lung segmentation. The result shows excellent sensitivity in the COVID-19 class compared to the other experiments. Experiment 3 showed the best result in sensitivity values in the range 95% to 97% while using the modified EAEO for weight optimization and 95%-93% without the optimization approach, as shown in Table 5.1 and Figure 5.2. The system's overall performance yielded an accuracy of 97.23%, with weight optimization using modified EAEO and 91.5% without optimization. The other optimization algorithms Figure 5.9. performance is in line with the previous experiments. The development of accuracy and loss during training and evaluation is depicted in Figure 5.10.while the F1-score increment is shown in Figure 5.11.

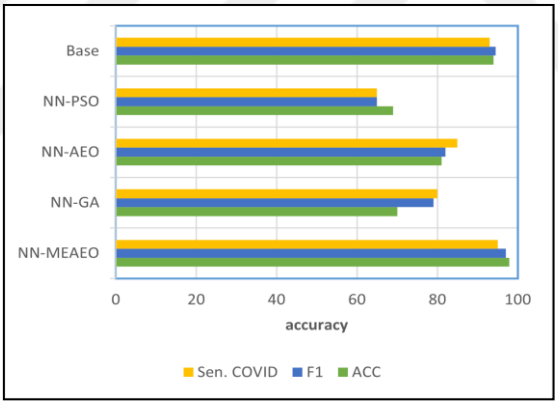
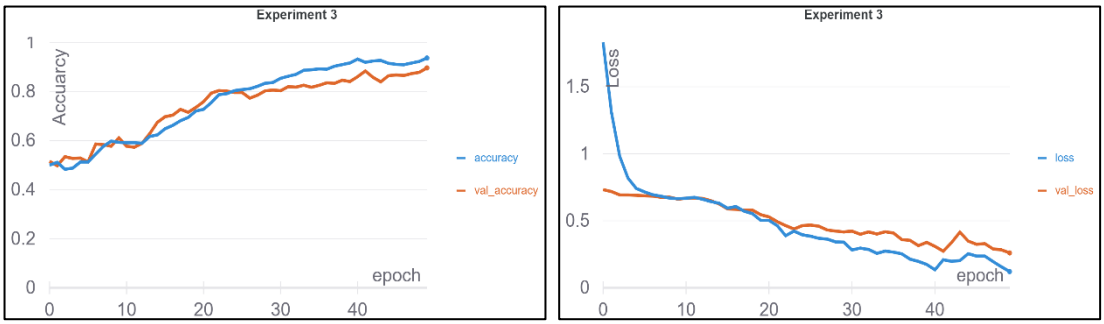


Figure 5.9. Performance of the optimization algorithms used in experiment 3.



(a) **(b)**

Figure 5.10: (a) Accuracy and validation Accuracy during training for experiment 3, (b) loss and validation loss for experiment 3.

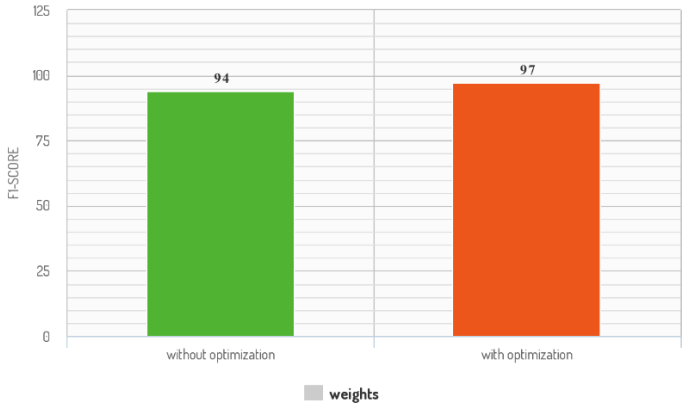


Figure 5.11: F1-score for experiment 3 considering the weight optimization..

5.4 INTERPRETING THE IMPROVEMENTS GAINED FROM APPLYING MODIFIED EAEO

As shown earlier in Table 5.1, experiments using the proposed optimization technique improved prediction performance compared to those that did not employ weight optimization, this indicates a possible performance loss if exclusively trained with traditional training algorithms like GD; however, when applied to the network, the suggested technique would gain an accuracy increase of 2-5% compared to utilizing typical training method. This increase is attributed to improved search methods, which can escape from local optima more effectively.

Furthermore, the modified EAEO's capability to avoid local optima is attributed to utilizing the trained weights as an individual in the EAEO population. This approach assisted the modified EAEO to obtain better results in the optimization process. Figure 5.12 depicts the development of the fitness value. The fitness function value(loss) continues to converge over the transformation process and seems relatively consistent in the end.

It should be noted that the cost function (loss) curve appears to have an inconsistency post-convergence since batches are employed with a relatively small number of samples while performing the optimization, in other words, the test set is altered in each cycle, resulting in oscillations in the fitness function output; however, the overall trend of the curve is positive.

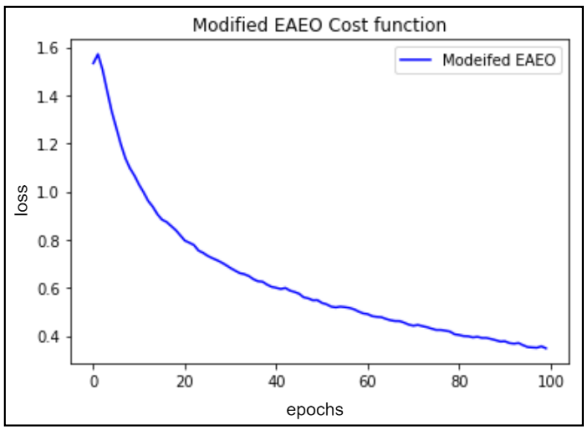


Figure 5.12. loss function error curve in experiment 3

5.5. COMPARING WITH OTHER MODELS

To determine how well the COVID-19 classification tests performed and to evaluate the effectiveness of the proposed method (which used medical data 579 samples). Table 5.3 contains a list of recent work in literature that has been referenced as a comparison to the performance of the proposed framework.

It is important to note that the current dataset and the problem formulation primarily differ from the mentioned studies, making a fair comparison with existing methods difficult. However, an attempt to provide an overview of these studies to determine the efficiency of the proposed approach in detecting COVID-19 has been conducted.

Table 5.3: Comparison of the performance of the proposed framework with current studies in the literature.

Reference	Data Type	Approach	Accuracy
Shambhu *et al.* [53]	X-ray	CNN	86.9%
Khan *et al.* [54]	X-ray	Transfer learning - Xception	89.6%
Arias-Londoño *et al.* [55]	X-ray	CNN+ lung segmentation	91.53%
Sakib *et al.* [56]	X-ray	CNN + GAN	93.94%
Xu *et al.* [24]	CT scans + Textual data	CNN + SVM	95.40%-97.70%
Ahsan *et al.* [17]	X- ray scans + Textual data	CNN + MLP	96.3%.
Goel *et al.* [57]	X-ray	CNN	97.78%
Proposed Framework	X-ray+ Textual data	CNN + MLP	97.85 %

The proposed novel framework in this study performed better regarding accuracy than the recent studies Table 5.2. In [53], CNN model-based binary classifier is employed on 746 CT

images containing COVID-19 and healthy patients, which achieved an accuracy of 86.9%. In [54], the overall accuracy of 89.6% is achieved with 93% precision in the COVID-19 class while using transfer learning based on the ImageNet dataset. However, the use of transfer learning based on the ImageNet dataset for biomedical image classification is not recommended. In [55], the experiments with image segmentation done by the authors achieved an accuracy of 91.5% when analyzing three classes with an F-score of 78.57 ±1.15% in the COVID-19 class. In[56], the proposed network achieved an accuracy of 93.94% in classifying COVID-19. Furthermore, the authors used a Generative adversarial network to increase the training data. In [24], the authors proposed a late fusion(merging) of CT scans and textual clinical data to assess the severity of COVID-19 in patients. They reported a classification accuracy between (95.4%-97.7%) and high class-specific accuracy (90.6%-99.9%). Ahsan et al. used multiple data types to classify COVID-19 and achieved an accuracy of 95.38%; however, the dataset has few samples. To the best of our ability, a model reconstruction from [17], which is comparable to the proposed model in this study, achieved an accuracy of 80.5% using the dataset in section 4.1. In [57], a binary classification based on the decoder encoder model achieved 97.78%. The dataset used in training the network consists of 2482 CT scans, of which 1230 images for patients not infected with coronavirus (but infected with other pulmonary diseases).

5.6. POTENTIAL FACTORS AFFECTING THE RESULTS

Several bias factors might affect the result. The lack of well-documented images with a different detector that might use a different method in capturing the scan could affect the consistency of the results. In COVID-19 classification, it is necessary to use chest X-ray images that are adequate for this purpose. Most studies in the literature employed various techniques to handle biological data in their approaches. These proposed techniques are accepted in conventional imaging classifications. Biomedical images should be addressed with caution, and specific rules must be followed while processing them so that the modified images do not differ from the original image. Due to the newly developing knowledge in this area, it is impossible to recommend a method or a set of techniques that is more efficient in identifying COVID-19 from a chest X-ray image. Most studies have an accuracy rate of more than 85%, which is

statistically a very high degree of accuracy. Aiming for 100% accuracy would be idealistic, as even a tiny percentage of misdiagnoses is problematic.

AI systems may have some utility in a supervised therapeutic context. We argue that putting these techniques into clinical practice requires more than just classification. As we show in experiment 1, it should be approached with caution. In contrast to what we saw in experiments 1 and 2, the result showed that the system recognizes areas to non-diagnostic regions. As a result, these findings are not in line with a clinical viewpoint. Lung segmenting is required to direct the neural network's attention to the lung organ, as shown in experiment 3.

6. CONCLUSIONS

In this study, a COVID-19 framework was proposed, which incorporated several techniques such as, lung segmentation and augmentation, merging neural networks, and an optimization algorithm (modified EAEO). These techniques assist in creating a robust framework to classify COVID-19 and pneumonia using a different dataset for each pathogen. A hybrid network is a viable approach to analyze X-ray images that correspond to pneumonia, COVID-19 patients, and biomedical images in general. The results demonstrated that merging several CNNs with a multi-layer perceptron and optimizing the weights using the modified EAEO can be beneficial to increase the system robustness. Furthermore, the employment of lung segmentation technique aided in achieving a reliable classification result that is not affected by unnecessary features. The impact of merging several data types such as images and textual clinical data is beneficial in improving classification accuracy. The proposed framework obtained an accuracy of 97.85% using the data in section 5.1, which is considerably better than most systems in the literature that predict COVID-19 from pneumonia.

A modified version of EAEO was proposed to *optimize* the neural network's weights and mitigate the performance loss from using typical training methods to train the neural network. In addition, weights of the hybrid network are incorporated in the initial population of the modified EAEO to guide the search to optimal values. The results obtained from all of the experiments support the idea that an optimization technique can further enhance the performance of a neural network. Moreover, merging different data types such as images and textual clinical data improves classification accuracy.

REFERENCES

[1] V. C. C. Cheng, S. K. P. Lau, P. C. Y. Woo, and K. Y. Yuen, "Severe Acute Respiratory Syndrome Coronavirus as an Agent of Emerging and Reemerging Infection," *Clin. Microbiol. Rev.*, vol. 20, no. 4, pp. 660–694, Oct. 2007, doi: 10.1128/CMR.00023-07.

[2] WHO, "WHO Coronavirus (COVID-19) Dashboard,." https://covid19.who.int/ (accessed Jul. 04, 2021).

[3] G. Chen, Q. Wu, H. Jiang, and Y. Zhong, "The impact of the COVID-19 pandemic on head and neck cancer patients," *Oral Oncol.*, vol. 110, p. 104881, Nov. 2020, doi: 10.1016/j.oraloncology.2020.104881.

[4] D. Al-Karawi, S. Al-Zaidi, N. Polus, and S. Jassim, "Machine Learning Analysis of Chest CT Scan Images as a Complementary Digital Test of Coronavirus (COVID-19) Patients," *medRxiv*, p. 2020.04.13.20063479, Apr. 2020, doi: 10.1101/2020.04.13.20063479.

[5] S. Mallett *et al.*, "At what times during infection is SARS-CoV-2 detectable and no longer detectable using RT-PCR-based tests? A systematic review of individual participant data," *BMC Med.*, vol. 18, no. 1, p. 346, Nov. 2020, doi: 10.1186/s12916-020-01810-8.

[6] A. Kordon, "Applied Artificial Intelligence-Based Systems as Competitive Advantage," in *2020 IEEE 10th International Conference on Intelligent Systems (IS)*, Aug. 2020, pp. 6–18. doi: 10.1109/IS48319.2020.9200097.

[7] F. Grillet, J. Behr, P. Calame, S. Aubry, and E. Delabrousse, "Acute Pulmonary Embolism Associated with COVID-19 Pneumonia Detected with Pulmonary CT Angiography," *Radiology*, vol. 296, no. 3, pp. E186–E188, Sep. 2020, doi: 10.1148/radiol.2020201544.

[8] J. Deussen, J. Hüser, and U. Naumann, "Toward Global Search for Local Optima," in *Operations Research Proceedings 2019*, Cham, 2020, pp. 97–104. doi: 10.1007/978-3-030-48439-2_12.

[9] M. Buscema, "Back Propagation Neural Networks," *Subst. Use Misuse*, vol. 33, pp. 233–70, Feb. 1998, doi: 10.3109/10826089809115863.

[10] "An Enhanced Artificial Ecosystem-Based Optimization for Optimal Allocation of Multiple Distributed Generations | IEEE Journals & Magazine | IEEE Xplore." https://ieeexplore.ieee.org/document/9208695 (accessed Jul. 29, 2021).

[11] J. P. Cohen, P. Morrison, and L. Dao, "COVID-19 Image Data Collection," *ArXiv200311597 Cs Eess Q-Bio*, Mar. 2020, Accessed: Jul. 04, 2021. [Online]. Available: http://arxiv.org/abs/2003.11597

[12] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers, "ChestX-ray8: Hospital-Scale Chest X-Ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases," p. 10.

[13] S. Cave, J. Whittlestone, R. Nyrup, S. O. hEigeartaigh, and R. A. Calvo, "Using AI ethically to tackle covid-19," *BMJ*, vol. 372, p. n364, Mar. 2021, doi: 10.1136/bmj.n364.

[14] A. J. DeGrave, J. D. Janizek, and S.-I. Lee, "AI for radiographic COVID-19 detection selects shortcuts over signal," *Nat. Mach. Intell.*, vol. 3, no. 7, pp. 610–619, Jul. 2021, doi: 10.1038/s42256-021-00338-7.

[15] M. Toğaçar, B. Ergen, and Z. Cömert, "COVID-19 detection using deep learning models to exploit Social Mimic Optimization and structured chest X-ray images using fuzzy color and stacking approaches," *Comput. Biol. Med.*, vol. 121, p. 103805, Jun. 2020, doi: 10.1016/j.compbiomed.2020.103805.

[16] G. Jain, D. Mittal, D. Thakur, and M. K. Mittal, "A deep learning approach to detect Covid-19 coronavirus with X-Ray images," *Biocybern. Biomed. Eng.*, vol. 40, no. 4, pp. 1391–1405, Oct. 2020, doi: 10.1016/j.bbe.2020.08.008.

[17] M. M. Ahsan, T. E. Alam, T. Trafalis, and P. Huebner, "Deep MLP-CNN Model Using Mixed-Data to Distinguish between COVID-19 and Non-COVID-19 Patients," *Symmetry*, vol. 12, no. 9, Art. no. 9, Sep. 2020, doi: 10.3390/sym12091526.

[18] A. Afifi, N. E. Hafsa, M. A. S. Ali, A. Alhumam, and S. Alsalman, “An Ensemble of Global and Local-Attention Based Convolutional Neural Networks for COVID-19 Diagnosis on Chest X-ray Images,” *Symmetry*, vol. 13, no. 1, p. 113, Jan. 2021, doi: 10.3390/sym13010113.
[19] H. Mukherjee, S. Ghosh, A. Dhar, S. M. Obaidullah, K. C. Santosh, and K. Roy, “Deep neural network to detect COVID-19: one architecture for both CT Scans and Chest X-rays,” *Appl. Intell.*, vol. 51, no. 5, pp. 2777–2789, May 2021, doi: 10.1007/s10489-020-01943-6.
[20] I. D. Apostolopoulos and T. A. Mpesiana, “Covid-19: automatic detection from X-ray images utilizing transfer learning with convolutional neural networks,” *Phys. Eng. Sci. Med.*, vol. 43, no. 2, pp. 635–640, Jun. 2020, doi: 10.1007/s13246-020-00865-4.
[21] L. Wang, Z. Q. Lin, and A. Wong, “COVID-Net: a tailored deep convolutional neural network design for detection of COVID-19 cases from chest X-ray images,” *Sci. Rep.*, vol. 10, no. 1, p. 19549, Nov. 2020, doi: 10.1038/s41598-020-76550-z.
[22] P. R. A. S. Bassi and R. Attux, “COVID-19 detection using chest X-rays: is lung segmentation important for generalization?,” *ArXiv210406176 Cs Eess*, Apr. 2021, Accessed: Jul. 28, 2021. [Online]. Available: http://arxiv.org/abs/2104.06176
[23] N. Lassau *et al.*, “Integrating deep learning CT-scan model, biological and clinical variables to predict severity of COVID-19 patients,” *Nat. Commun.*, vol. 12, no. 1, p. 634, Jan. 2021, doi: 10.1038/s41467-020-20657-4.
[24] M. Xu *et al.*, “Accurately Differentiating Between Patients With COVID-19, Patients With Other Viral Infections, and Healthy Individuals: Multimodal Late Fusion Learning Approach,” *J. Med. Internet Res.*, vol. 23, no. 1, p. e25535, Jan. 2021, doi: 10.2196/25535.
[25] X. Mei *et al.*, “Artificial intelligence–enabled rapid diagnosis of patients with COVID-19,” *Nat. Med.*, vol. 26, no. 8, pp. 1224–1228, Aug. 2020, doi: 10.1038/s41591-020-0931-3.
[26] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation,” in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Cham, 2015, pp. 234–241. doi: 10.1007/978-3-319-24574-4_28.
[27] X. Li, Y. Zhou, P. Du, G. Lang, M. Xu, and W. Wu, “A deep learning system that generates quantitative CT reports for diagnosing pulmonary Tuberculosis,” *Appl. Intell.*, vol. 51, no. 6, pp. 4082–4093, Jun. 2021, doi: 10.1007/s10489-020-02051-1.
[28] L. M. R. Rere, M. I. Fanany, and A. M. Arymurthy, “Simulated Annealing Algorithm for Deep Learning,” *Procedia Comput. Sci.*, vol. 72, pp. 137–144, Jan. 2015, doi: 10.1016/j.procs.2015.12.114.
[29] S. Chakraborty, A. Raman, S. Sen, K. Mali, S. Chatterjee, and H. Hachimi, “Contrast Optimization using Elitist Metaheuristic Optimization and Gradient Approximation for Biomedical Image Enhancement,” in *2019 Amity International Conference on Artificial Intelligence (AICAI)*, Feb. 2019, pp. 712–717. doi: 10.1109/AICAI.2019.8701367.
[30] M. Carvalho and T. B. Ludermir, “Particle Swarm Optimization of Neural Network Architectures andWeights,” in *7th International Conference on Hybrid Intelligent Systems (HIS 2007)*, Sep. 2007, pp. 336–339. doi: 10.1109/HIS.2007.45.
[31] H. Abdulwahab, N. Ahmad, A. Alsewari, and S. Salih, “An Enhanced Version of Black Hole Algorithm Via Levy Flight for Optimization and Data Clustering Problems,” *IEEE Access*, vol. PP, pp. 1–1, Aug. 2019, doi: 10.1109/ACCESS.2019.2937021.
[32] A. Hatamlou, “Black hole: A new heuristic optimization approach for data clustering,” *Inf. Sci.*, vol. 222, pp. 175–184, Feb. 2013, doi: 10.1016/j.ins.2012.08.023.
[33] A. H. Alsaeedi, A. H. Aljanabi, M. E. Manna, and A. L. Albukhnefis, “A proactive metaheuristic model for optimizing weights of artificial neural network,” *Indones. J. Electr. Eng. Comput. Sci.*, vol. 20, no. 2, Art. no. 2, Nov. 2020, doi: 10.11591/ijeecs.v20.i2.pp976-984.
[34] I. T. Brain and F. Rosenblatt, “The Perceptron: A Probabilistic Model for Information Storage and Organization.”

[35] S. Ruder, “An overview of gradient descent optimization algorithms,” *ArXiv160904747 Cs*, Jun. 2017, Accessed: Sep. 12, 2021. [Online]. Available: http://arxiv.org/abs/1609.04747
[36] J. Duchi, E. Hazan, and Y. Singer, “Adaptive Subgradient Methods for Online Learning and Stochastic Optimization,” p. 39.
[37] “[1412.6980] Adam: A Method for Stochastic Optimization.” https://arxiv.org/abs/1412.6980 (accessed Sep. 12, 2021).
[38] T. Dozat, “Incorporating Nesterov Momentum into Adam,” Feb. 2016, Accessed: Sep. 12, 2021. [Online]. Available: https://openreview.net/forum?id=OM0jvwB8jIp57ZJjtNEZ
[39] A. F. Agarap, “Deep Learning using Rectified Linear Units (ReLU),” *ArXiv180308375 Cs Stat*, Feb. 2019, Accessed: Sep. 12, 2021. [Online]. Available: http://arxiv.org/abs/1803.08375
[40] B. Zamanlooy and M. Mirhassani, “Efficient VLSI Implementation of Neural Networks With Hyperbolic Tangent Activation Function,” *IEEE Trans. Very Large Scale Integr. VLSI Syst.*, vol. 22, no. 1, pp. 39–48, Jan. 2014, doi: 10.1109/TVLSI.2012.2232321.
[41] P. Ramachandran, B. Zoph, and Q. V. Le, “Searching for Activation Functions,” *ArXiv171005941 Cs*, Oct. 2017, Accessed: Sep. 12, 2021. [Online]. Available: http://arxiv.org/abs/1710.05941
[42] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, “Self-Normalizing Neural Networks,” *ArXiv170602515 Cs Stat*, Sep. 2017, Accessed: Sep. 12, 2021. [Online]. Available: http://arxiv.org/abs/1706.02515
[43] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” *ArXiv150203167 Cs*, Mar. 2015, Accessed: Sep. 25, 2021. [Online]. Available: http://arxiv.org/abs/1502.03167
[44] K. Fukushima, “Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position,” *Biol. Cybern.*, vol. 36, no. 4, pp. 193–202, Apr. 1980, doi: 10.1007/BF00344251.
[45] S.-C. Wang, “Artificial Neural Network,” in *Interdisciplinary Computing in Java Programming*, S.-C. Wang, Ed. Boston, MA: Springer US, 2003, pp. 81–100. doi: 10.1007/978-1-4615-0377-4_5.
[46] D. E. Goldberg and J. H. Holland, “Genetic Algorithms and Machine Learning,” *Mach. Learn.*, vol. 3, no. 2, pp. 95–99, Oct. 1988, doi: 10.1023/A:1022602019183.
[47] “Butterfly optimization algorithm: a novel approach for global optimization | SpringerLink.” https://link.springer.com/article/10.1007/s00500-018-3102-4 (accessed Sep. 03, 2021).
[48] T. Rahkar Farshi, “Battle royale optimization algorithm,” *Neural Comput. Appl.*, vol. 33, no. 4, pp. 1139–1157, Feb. 2021, doi: 10.1007/s00521-020-05004-4.
[49] G. G. Wang, S. Deb, and L. D. S. Coelho, “Earthworm optimisation algorithm: a bio-inspired metaheuristic algorithm for global optimisation problems,” *Int. J. Bio-Inspired Comput.*, vol. 12, no. 1, p. 1, 2018, doi: 10.1504/IJBIC.2018.093328.
[50] W. Hryniewska, P. Bombiński, P. Szatkowski, P. Tomaszewska, A. Przelaskowski, and P. Biecek, “Checklist for responsible deep learning modeling of medical images based on COVID-19 detection studies,” *Pattern Recognit.*, vol. 118, p. 108035, Oct. 2021, doi: 10.1016/j.patcog.2021.108035.
[51] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal Loss for Dense Object Detection,” *ArXiv170802002 Cs*, Feb. 2018, Accessed: Sep. 30, 2021. [Online]. Available: http://arxiv.org/abs/1708.02002
[52] W. Zhao, L. Wang, and Z. Zhang, “Artificial ecosystem-based optimization: a novel nature-inspired meta-heuristic algorithm,” *Neural Comput. Appl.*, vol. 32, no. 13, pp. 9383–9425, Jul. 2020, doi: 10.1007/s00521-019-04452-x.
[53] S. Shambhu, D. Koundal, P. Das, and C. Sharma, “Binary Classification of COVID-19 CT Images Using CNN: COVID Diagnosis Using CT,” *Int. J. E-Health Med. Commun. IJEHMC*, vol. 13, no. 2, pp. 1–13, Jul. 2021, doi: 10.4018/IJEHMC.20220701.oa4.

[54] A. I. Khan, J. L. Shah, and M. M. Bhat, “CoroNet: A deep neural network for detection and diagnosis of COVID-19 from chest x-ray images,” *Comput. Methods Programs Biomed.*, vol. 196, p. 105581, Nov. 2020, doi: 10.1016/j.cmpb.2020.105581.

[55] J. D. Arias-Londoño, J. A. Gómez-García, L. Moro-Velázquez, and J. I. Godino-Llorente, “Artificial Intelligence Applied to Chest X-Ray Images for the Automatic Detection of COVID-19. A Thoughtful Evaluation Approach,” *IEEE Access*, vol. 8, pp. 226811–226827, 2020, doi: 10.1109/ACCESS.2020.3044858.

[56] S. Sakib, T. Tazrin, M. M. Fouda, Z. Md. Fadlullah, and M. Guizani, “DL-CRC: Deep Learning-Based Chest Radiograph Classification for COVID-19 Detection: A Novel Approach,” *IEEE Access*, vol. 8, pp. 171575–171589, 2020, doi: 10.1109/ACCESS.2020.3025010.

[57] C. Goel, A. Kumar, S. K. Dubey, and V. Srivastava, “Efficient Deep Network Architecture for COVID-19 Detection Using Computed Tomography Images,” Aug. 2020. doi: 10.1101/2020.08.14.20170290.

APPENDIX A

SUPPLEMENTAL MATERIAL

This section is designed for the important information that could not be included in the main text.

1.1. PYTHON LIBRARIES

several python libraries required and was used in this study, these libraries helped in the design of the framework process, in the framework. Keras and Tenserflow code will be appended to this appendix

i. Kaggle
ii. Pandas
iii. CV2 library for histogram equalization and CLAHE
iv. Imageaug for image augmentation
v. Mealpy for the base design of the evolutionary algorithms
vi. Resnet34 for image segmentation
vii. NumPy for matrices and random generator

1.1.2 Acquiring the Dataset

For the covid dataset and integration with GitHub is require, while the pneumonia dataset Kaggle integration is key to download the dataset. It's important to have a large disk space, over 40GB, for pneumonia dataset.

1.1.3 Dataset preprocessing

The dataset for pneumonia and covid-19 contains several pathogen and image types. It important to remove the data that does not benefits this study. The diseases are listed below:

i. 'Pneumonia/Viral/COVID-19'

ii. 'Pneumonia/Viral/SARS'

iii. 'Pneumonia/Fungal/Pneumocystis'

iv. 'Pneumonia/Bacterial/Streptococcus'

v. 'No Finding'

vi. 'Pneumonia/Bacterial/Chlamydophila'

vii. 'Pneumonia/Bacterial/Klebsiella'

viii. 'Pneumonia/Bacterial/Legionella'

ix. 'Pneumonia/Lipoid'

x. 'Pneumonia/Viral/Varicella'

xi. 'Pneumonia/Bacterial'

xii. 'Pneumonia/Bacterial/Mycoplasma'

xiii. 'Pneumonia/Viral/Influenza'

xiv. 'Tuberculosis'

xv. 'Pneumonia/Viral/Influenza/H1N1'

xvi. 'Pneumonia/Viral/Herpes '

xvii. 'COVID-19/Lipoid'

xviii. 'Pneumonia/Fungal/Aspergillosis'

xix. 'Pneumonia/Aspiration'

xx. 'Pneumonia/Bacterial/Nocardia'

xxi. 'Pneumonia/Bacterial/Staphylococcus/MRSA'

1.1.3 Libraries versions

i. Python 3.6 or higher.

ii. Keras >=2.1.0 or higher.

iii. Tensorflow >= 2.1 or higher.

1.1.5 Code for Network Design

The code require python 3.8, the latest Keras and Tensorflow. And can be found at GitHub(https://github.com/PFHaMu/NN-MEAEO.). The design and implementation all done in google colab for the sole reason that Google colab provide an outstanding graphical processing units with the latest libraries.