REPUBLIC OF TÜRKİYE

ALTINBAŞ UNIVERSITY

Institute of Graduate Studies

Electrical and Computer Engineering

AN ASSESSMENT OF THE EFFICACY OF DEEP LEARNING AND MACHINE LEARNING METHODS IN AUTOMATICALLY DETECTING PAEDIATRIC PNEUMONIA IN CHEST X-RAY IMAGES

Hussein Abd Ali Hatif ALSAADI

Master's Thesis

Supervisor

Asst. Prof. Dr. Mesut ÇEVIK

İstanbul, 2024

AN ASSESSMENT OF THE EFFICACY OF DEEP LEARNING AND MACHINE LEARNING METHODS IN AUTOMATICALLY DETECTING PAEDIATRIC PNEUMONIA IN CHEST X-RAY IMAGES

Hussein Abd Ali Hatif ALSAADI

Electrical and Computer Engineering

Master's Thesis

ALTINBAŞ UNIVERSITY

2024

The thesis titled **AN ASSESSMENT OF THE EFFICACY OF DEEP LEARNING AND MACHINE LEARNING METHODS IN AUTOMATICALLY DETECTING PAEDIATRIC PNEUMONIA IN CHEST X-RAY IMAGES** by HUSSEİN ABD ALİ HATİF ALSAADİ and submitted on 00/01/2024 has been **accepted unanimously** for the degree of Master of Science in Electrical and Computer Engineering.

Asst. Prof. Dr. Mesut ÇEVIK

Supervisor

Thesis Defense Committee Members:

Asst. Prof. Dr. Mesut ÇEVIK — Department of Computer Engineering, Altınbaş university

Asst. prof. Dr. — Altınbaş university

Asst. prof. Dr

I hereby declare that this thesis meets all format and submission requirements of a Master's thesis.

I affirm that all the information and data presented in this graduation project have been acquired in strict adherence to academic regulations and ethical standards. I thus affirm that I have properly cited all sources of unoriginal materials and conclusions in the text, and that all references listed in the Reference List have been referenced in the text, as per the norms and guidelines.

Hussein Abd Ali Hatif ALSAADI

Signature

DEDICATION

I would like to start by thanking God Almighty for giving me the knowledge, health, and stamina to finish my research, and I thank my supervisor Asst. Prof. Dr. Mesut ÇEVIK for all the support during my academic career. I dedicate this letter to my parents. There are not enough words to express, and I cannot repay what they have done for me. I will continue to work hard to meet their expectations, and finally, I dedicate this to my close family members and friends who have supported me through this study.

ABSTRACT

AN ASSESSMENT OF THE EFFICACY OF DEEP LEARNING AND MACHINE LEARNING METHODS IN AUTOMATICALLY DETECTING PAEDIATRIC PNEUMONIA IN CHEST X-RAY IMAGES

ALSAADI, Hussein Abd Ali Hatif

M.Sc., Electrical and Computer Engineering, Altınbaş University,

Supervisor: Asst. Prof. Dr. Mesut ÇEVIK

Date: January / 2024

Pages: 66

Accurate identification and classification of images of patients with pneumonia is vital for effective diagnosis and treatment. Advanced learning techniques such as CNN LSTM (Convolutional Neural Network Long Short-Term Memory) have proven 97% accuracy on this task. A labeled dataset of 5,856 images of pediatric pneumonia patients from Kaggle was used to evaluate these models. The dataset was divided into training (70%), testing (15%), and validation (15%) sets. It is worth noting that the ResNet 50 -CNN model and the CNN-ExtraTrees model achieved an accuracy rate of 98% and 94%, respectively. These models provide resources for clinicians and radiologists in their decision-making processes. Extracting features from images plays a role in pattern prediction using the optimizer as part of the modeling process. To improve performance, modifications were made to the RMSprop parameters. Model performance evaluation included metrics such as recall, precision, F1 score, and overall accuracy evaluation. These measures provide insight into the effectiveness of models while also serving as reference points for investigations. The exceptional accuracy rates achieved by these models confirm their ability to accurately classify patients with pneumonia, which has implications for improving patient outcomes.

The evaluation criteria used in this research provide an examination of the advantages and limitations of models that highlights the importance of incorporating these metrics to ensure accurate performance in scenarios.

Keywords: Medical Image, Pediatric Pneumonia, Resnet50-CNN, CNN-Extratrees, Convolutional Neural Network, Long Short-Term Memory.

TABLE OF CONTENT

Pages

LIST OF TABELS

Pages

LIST OF FIGURES

Pages

ABBREVIATIONS

ML	:	Machine Learning
KNN	:	K-Nearest Neighbours
RF	:	Random Forest
DT	:	Decision Trees
SVM	:	Support Vector Machine
ANN	:	Artificial Neural Network
IoT	:	Internet of Things
TP	:	True Positive
FP	:	False Positive
TN	:	True Negative
FN	:	False Negative
DOS	:	Denial of Service
D.P	:	Data Type Probing
M.C	:	Malicious Control
SC	:	Scan
M.O	:	Malicious Operation
W.S	:	Wrong Setup

1. INTRODUCTION

The apparently harmless term "pneumonia" hides a terrible truth. Millions of lives are lost annually as a result of this sneaky danger to world health, which disproportionately affects weaker groups including infants, the elderly, and people with weakened immune systems [1]. Pneumonia, in all of its manifestations, is a widespread and complicated health issue that requires our attention, comprehension, and coordinated efforts to lessen its grave effects [2].

Pneumonia is a contagious illness that mostly affects the airway system, producing inflammation in the lungs' air sacs. These infections, which may result in the buildup of liquid and pus in the lungs and decrease the lungs' capacity to exchange oxygen and carbon dioxide, often cause this inflammation. The effects of this disturbance are severe since pneumonia may become life-threatening if not treated [3.].

Pneumonia has a staggering global impact and a high incidence. It is still one of the most common causes of fatalities across all age groups and a major contributor to morbidity and mortality globally. Pneumonia kills a total of 740 180 kids in 2019 [4], accounting for 14% of all fatalities among children under the age of five. Pneumonia still poses a serious threat to adults, particularly those over 70, and is often made worse by underlying medical issues and compromised immune systems [5].

Pneumonia has a considerable negative economic and social effect in addition to its fatality rates. Hospitalization due to pneumonia puts a load on the healthcare system and uses a significant number of medical resources. It commonly results in lost days of work and school as well, which reduces production and puts a financial strain on families and communities [6]. Pneumonia's effects go beyond only what it does to a person's physical health; they also affect their psychological and emotional health as they cause families to deal with the death of loved ones and the worry that it will come again [7].

The pneumonia danger is exacerbated by the adaptable and varied character of the organism. Pathogens that cause pneumonia may take on several shapes, such as fungus, viruses, and bacteria as shown in figure 1.1 and 1.2. A few of the microbes that cause this illness include Streptococcus pneumonia's, Homophiles influenzal, and Mycoplasma pneumonia's. Effective treatment and prevention are complicated by these viruses' ongoing evolution [8].

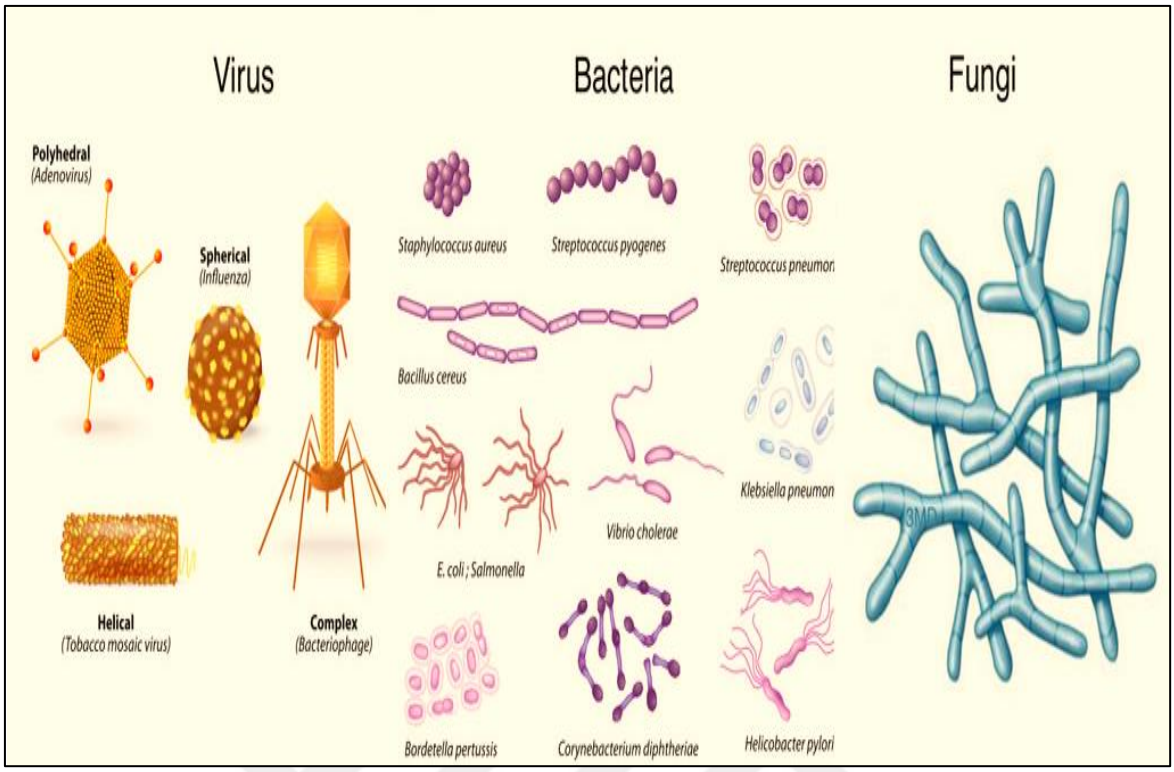

Figure 1.1: Causes of Pneumonia.

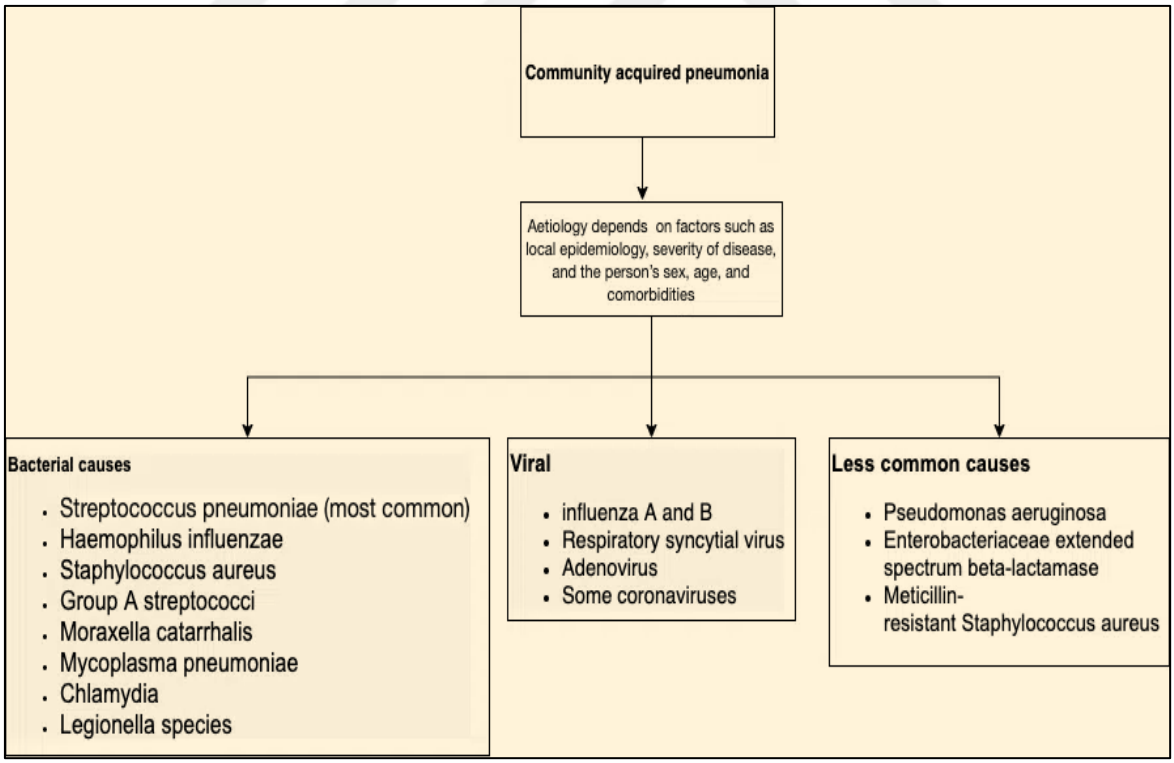

Figure 1.2: Types of Pneumonia and Their Causes.

Especially during flu seasons, viral pneumonia, which is often brought on by respiratory syncytial virus (RSV) or influenza viruses, offers significant hazards. In addition, the advent of new pathogens, such as COVID-19's causative agent, the severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2), has brought respiratory illnesses' potential for pandemic spread to light. The COVID-19 pandemic serves as a sobering reminder of how pneumonia may overwhelm healthcare systems and have far-reaching effects on society [9,10].

The quest to comprehend and treat pneumonia is an international, cross-disciplinary effort. Governments, medical professionals, researchers, campaigners, and communities are all involved. Every year that goes by, we learn more about the illness, which helps us to improve our methods and our capacity to shield people from it [11].

For prompt treatment and efficient management of pediatric pneumonia, an early and precise diagnosis is essential. In this setting, medical imaging, in particular a chest X-ray (CXR) scan, has become a vital diagnostic instrument for the identification and evaluation of pediatric pneumonia. Healthcare experts must manually analyze these photos, which takes time and is inconsistent amongst observers. This has led to an increase in interest in the creation and use of automated systems that use deep learning and machine learning to identify pneumonia as shown in figure 1.3 [12].

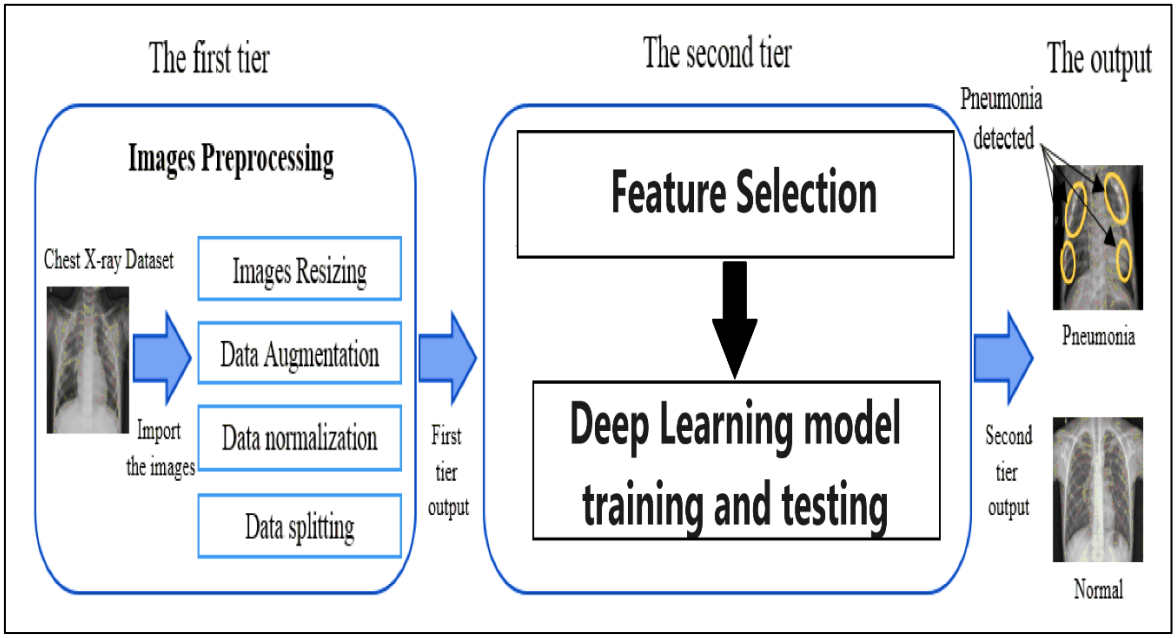

Figure 1.3: Steps Takin to Detect Pneumonia Using Chest X-Ray.

A field of artificial intelligence called "deep learning," which is a subset of machine learning, is motivated by the structure and operation of the human brain. It is characterized by deep neural networks, which have numerous layers and are capable of learning on their own how to extract intricate patterns and characteristics from input. Deep learning is especially appropriate for image analysis because to this capability, which includes the analysis of medical pictures like chest X-rays [13,14].

Deep learning has a wide range of possible applications, including the diagnosis of pneumonia. These algorithms may help physicians prioritize situations and enable quick intervention for patients who are most in need. By automating the initial examination procedure, they may lessen the pressure on radiologists and free them up to concentrate on situations that are more complicated and nuanced. Deep learning algorithms may be very helpful tools in environments with limited resources, increasing access to healthcare for marginalized people [15].

1.1 THESIS OBJECTIVES

We will dig into the methodology, structures, and datasets that support these advancements as we start out on a trip to investigate the significant influence of deep learning algorithms in the area of pneumonia identification using chest X-ray analysis. We will assess the effectiveness of several deep learning models objectively, highlighting their advantages and disadvantages. We will also go into the ethical and legal issues that surround the use of these technologies in healthcare.

Beyond the technical elements of deep learning and machine learning, this study is significant because it addresses the urgent clinical need for rapid and precise pediatric pneumonia diagnosis. We may be able to shorten diagnostic delays, better resource allocation, and improve the overall standard of care for pediatric pneumonia patients by using artificial intelligence in healthcare. This research emphasizes the multidisciplinary character of contemporary healthcare, where cooperation between computer scientists, healthcare administrators, and medical practitioners may result in innovative ideas that are beneficial to both patients and healthcare providers alike.

2. RELATED WORK

Various literary efforts have been made to employ Artificial Intelligence (AI) and Deep Learning methods to determine the presence or absence of pneumonia, which is a binary classification problem [16-20]. A restricted number of research have utilised machine learning methods to diagnose pneumonia [21-23]. Deep Learning (DL), specifically Convolutional Neural Networks (CNNs), have been employed to categorise pneumonia according to its etiological source, differentiating between bacterial and viral origins.

Rajaraman et al. (2018) [24] assessed the efficacy of various tailored convolutional neural network (CNN) architectures in diagnosing pneumonia and differentiating between viral and bacterial kinds using 5232 chest radiographs of paediatric patients. The data description is presented in Table 2.1. The researchers assessed the effectiveness of Sequential CNN, Inception CNN, Residual CNN, and VGG16 models, and determined the Region of Interest (ROI) using a separate visualisation technique. The tailored VGG16 model surpassed the assessed models, with an accuracy of 96.2% in identifying pneumonia and 91.8% in distinguishing between viral and bacterial classifications.

Table 2.1: The Data Used in Rajaraman Et Al. Research.

Category	Training Samples	Test Samples	File Type
Normal	1349	234	JPG
Bacterial	2538	242	JPG
Viral	1345	148	JPG

In 2020, Rahman et al. [25] attempted to automatically categorise multiple types of pneumonia by employing 5247 CXR images from the Kaggle pneumonia dataset, as described in table 2.2. An analysis was conducted on the performance of four commonly utilised pretrained models, specifically AlexNet, ResNet18, DenseNet201, and

SqueezeNet, utilising transfer learning. Based on their research, DenseNet201 demonstrated greater performance in comparison to other models. The diagnostic accuracy for pneumonia was 98%, while the accuracy for differentiating between the two primary causes of the disease was 93.3%. With the same objective.

Table 2.2: Data Used in Rahman Et Al. Work.

Category	Number of X-ray Images
Normal	1341
Bacterial Pneumonia	2561
Viral Pneumonia	1345
Total	5247

In 2020, Rahman et al. [25] attempted to automatically categorise multiple types of pneumonia by employing 5247 CXR images from the Kaggle pneumonia dataset, as described in table 2.2. An analysis was conducted on the performance of four commonly utilised pretrained models, specifically AlexNet, ResNet18, DenseNet201, and SqueezeNet, utilising transfer learning. Based on their research, DenseNet201 demonstrated greater performance in comparison to other models. The diagnostic accuracy for pneumonia was 98%, while the accuracy for differentiating between the two primary causes of the disease was 93.3%. With the same objective.

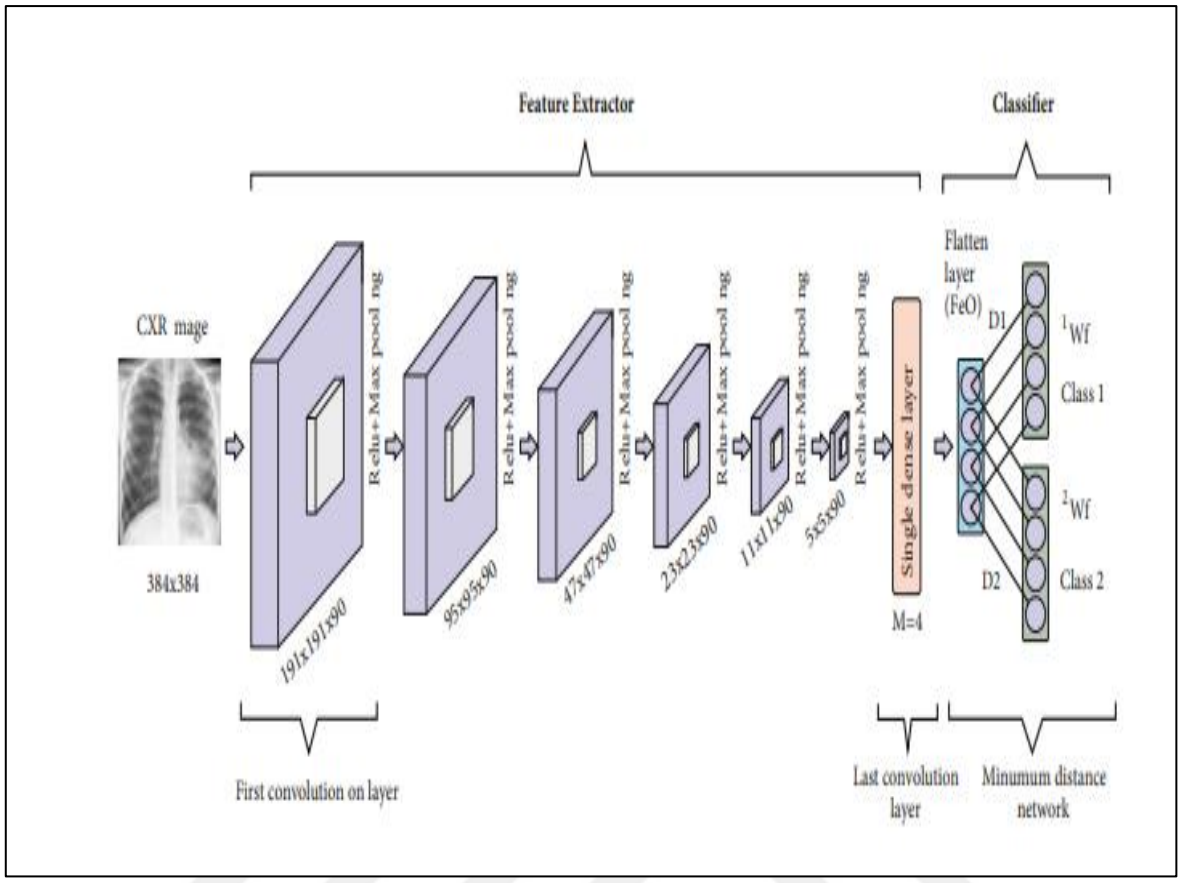

Figure 2.1: The Proposed System Used in [26].

Furthermore..., In 2021, Alqudah et al. [27] utilised a modified Convolutional Neural Network (CNN) structure to differentiate between bacterial and viral pneumonia as well as normal Chest X-rays (CXRs). This was achieved by analysing a dataset consisting of 5852 pictures. The framework had two discrete phases. At first, a Convolutional Neural Network (CNN) was employed to extract features. The architecture of the CNN used is illustrated in figure 2.2. Following that, in the second phase, K-Nearest Neighbour (KNN) and Support Vector Machine (SVM) classifiers were utilised. The researchers created two hybrid models, specifically CNN-KNN and CNN-SVM, and utilised a 10-fold cross validation methodology. The original hybrid model attained a precision rate of 94.03%, whereas the second hybrid model earned a precision rate of 93.9%.

#	Layer	Information		#	Layer	Information	
1	Input Layer	Size	64*64	9	Maxpol_2	Kernel Size	2*2
						Stride	2*2
2	Conv_1	Number of Filters	32	10	Conv_3	Number of Filters	32
		Kernel Size	3*3			Kernel Size	3*3
		Activation	RELU			Activation	RELU
3	Batch_Norm_1	Number of Channels	32	11	Batch_Norm_3	Number of Channels	32
5	Maxpol_1	Kernel Size	2*2	13	Maxpol_3	Kernel Size	2*2
		Stride	2*2			Stride	2*2
6	Conv_2	Number of Filters	16	14	Conv_4	Number of Filters	32
		Kernel Size	3*3			Kernel Size	3*3
		Activation	RELU			Activation	RELU
7	Batch_Norm_2	Number of Channels	16	15	Batch_Norm_4	Number of Channels	32

Figure 2.2: The Structure of the CNN Used in [27].

To address the task of multiclass classification, a separate research conducted in 2021 [28] employed a pre-trained Xception model along with data augmentation. The models accurately classified 5840 photographs from the Mendeley website, with an accuracy rate of 82.69%. Additionally, certain studies employed deep learning techniques to accurately identify pneumonia by analysing Chest X-ray (CXR) images. Harsh Agrawal utilised pre-processing techniques prior to implementing the ResNet50 v2 deep learning framework. This approach yielded a 96% accuracy in identifying pneumonia in CXR images [29].

On the other hand, Alquran et al. utilised Texture cues and conventional machine learning techniques to classify Chest-X-rays (CXR) into three categories: Pneumonia, regardless of its origin (viral or bacterial), COVID-19, and normal chest images. They reached an

accuracy of 93.1% for all classes [30]. Rajasenbagam et al. present a Deep learning model for diagnosing pneumonia infection using Chest X-ray (CXR) images. [31]'s proposed Convolutional Neural Network (CNN) was trained using a dataset of 12,000 chest X-ray (CXR) images which was manually generated using augmenting techniques on the original dataset figure 2.3 shows the effect of augmentation on the achieved accuracy and achieved a test accuracy of 99.34% the used methodology is shown in figure 2.4. Moreover, the suggested convolutional neural network (CNN) surpasses existing CNN models such as AlexNet, VGG16Net, and InceptionNet [31].

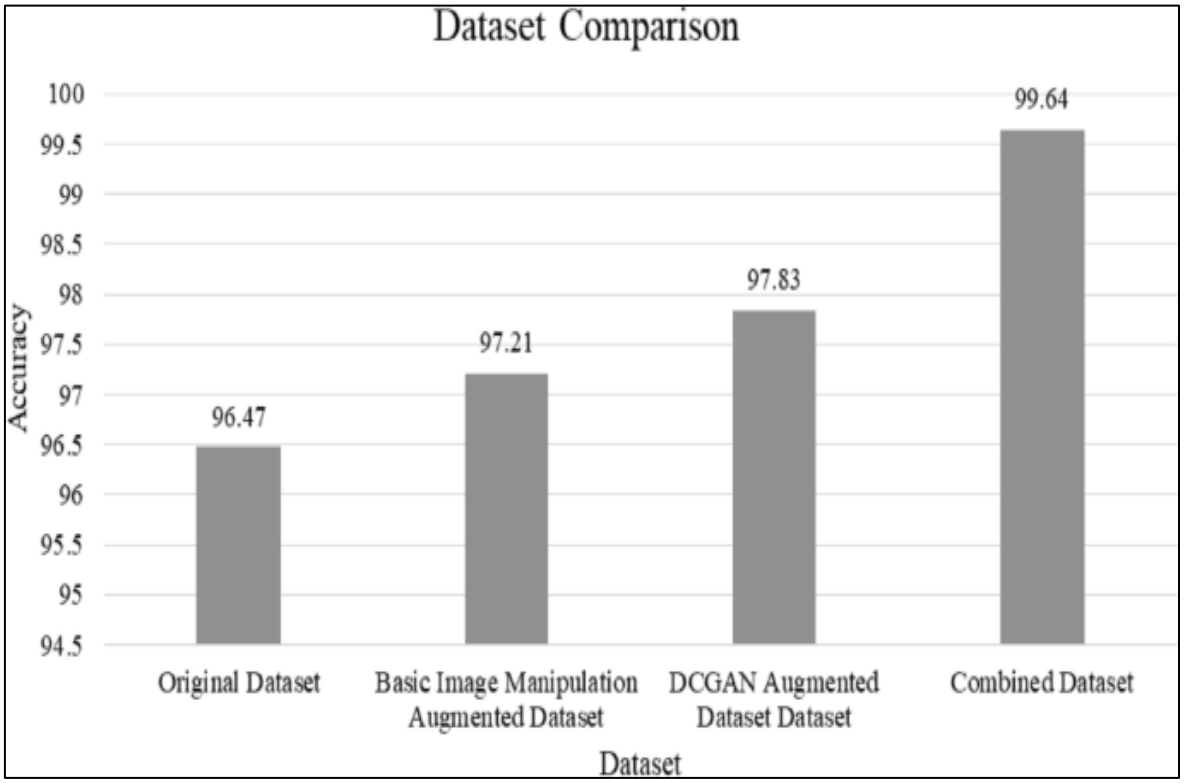

Figure 2.3: Augmentation Effect on the Achieved Accuracy [31].

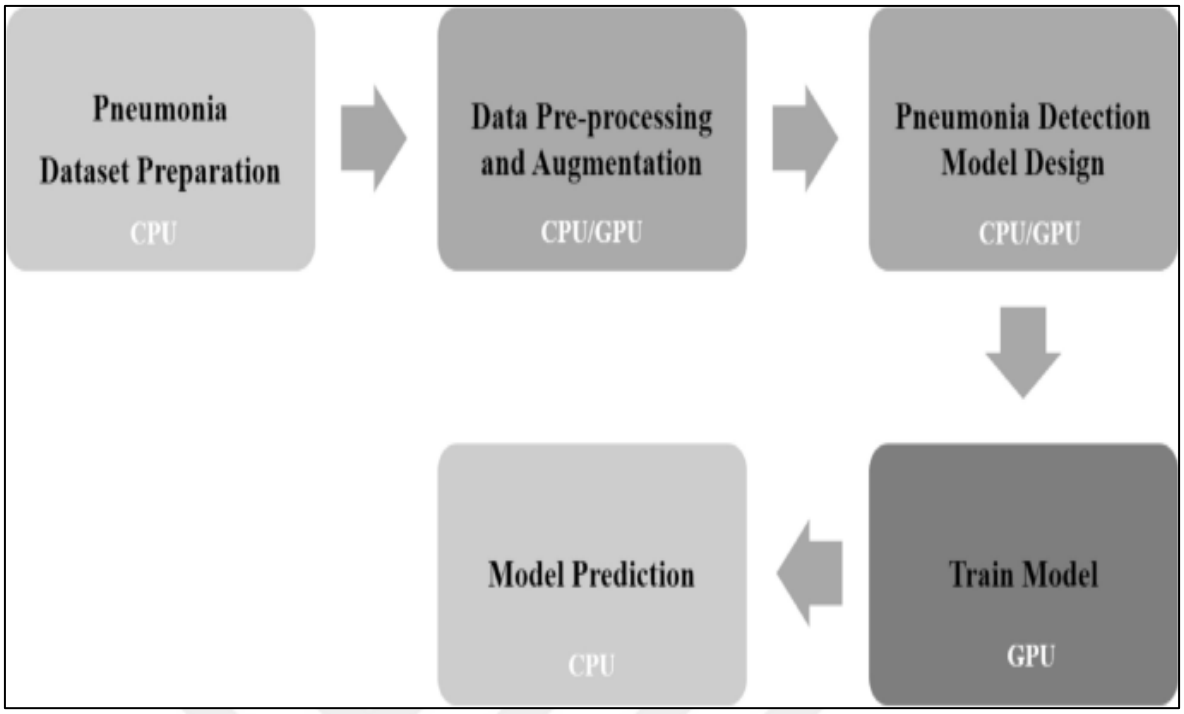

Figure 2.4: Methodology Used by [31].

In their 2022 study, Shah A, et al. [32] conducted an assessment of various deep learning models for the detection of pneumonia in chest X-ray (CXR) images. The performance outcomes, limits, and efficacy of the existing models have been assessed as well. While this work is a valuable and useful resource for applying DL to classify and locate pneumonia caused by covid-19, it is important to note that the study's focus is restricted to a small number of effective DL models specifically for pneumonia identification.

In 2022, Kareem et al. [33] proposed a meta-analysis of machine learning (ML) and deep learning (DL) models to predict the presence of pneumonia in chest X-ray (CXR) images. The research is centred around a security-focused method that may be utilised and implemented by several institutions to enhance the current system in order to tackle diverse health issues. This research examines various machine learning (ML) and deep learning (DL) models, including K-nearest neighbours (KNN), artificial neural networks (ANN), convolutional neural networks (CNN), and others, for the purpose of predicting pneumonia. It also provides details about the datasets used, the advantages, and disadvantages of applying these approaches. However, this SLR lacks a thorough evaluation of the quality of research on pneumonia detection. Moreover, it is essential to incorporate a comprehensive assessment of the performance measures of all models

documented in the existing literature. However, this review restricts the ability to make meaningful comparisons of pneumonia detection strategies.

Ultimately

The authors of [34] provided a comprehensive analysis of several pneumonia datasets that are available to the public. They also presented an overview of deep learning models that can be applied to these datasets. Nevertheless, this paper only examined a limited number of studies on pneumonia detection models and does not offer a comprehensive evaluation of their effectiveness or identify any research deficiencies in the current work.

3. METHODOLOGY

3.1 MEDICAL IMAGING TECHNIQUES

The integration of intelligence (AI), in the field of imaging has the potential to bring about a revolutionary change in healthcare. It can enhance accuracy, treatment planning. Ultimately improve patient outcomes. AI has been applied to medical imaging modalities such as X ray CT scan, MRI, ultrasound and PET scan. Let's explore some applications and advancements:

a. Automated Image Analysis:

AI algorithms are trained to analyze images and accurately quantify features like lesion size, volume and texture. This aids in disease identification and treatment planning [35].

b. Computer Aided Diagnosis:

AI algorithms can assist radiologists in identifying and diagnosing a range of conditions, including cancer. They act as opinions that potentially enhance diagnostic accuracy [36].

c. Image Segmentation:

AI algorithms designed for image segmentation help identify regions of interest within images such as tumors or organs. This enables analysis and diagnosis of those areas [37].

d. Image Alignment and Registration:

AI algorithms can. Register images obtained from different modalities like CT scans or MRIs. This assists, in treatment planning and monitoring patients [38].

e. Predictive Modeling:

AI algorithms have the ability to forecast outcomes by analyzing medical imaging data. They can predict disease progression. How patients will respond to treatment [39].

f. Cancer Detection:

Researchers are developing AI algorithms to assist in the detection of types of cancer such, as breast and lung cancer. These algorithms have shown accuracy levels to those of expert radiologists [40].

g. Neurological Disorders:

AI is being used in the diagnosis and treatment of disorders including Alzheimers and Parkinsons disease. Using MRI scans AI algorithms can accurately predict how these diseases will progress [40].

h. Cardiovascular Ailments:

AI algorithms are assisting in the identification and management of heart diseases. They can diagnose heart conditions with a level of accuracy based on CT scans.

i. LungDiseases:

Ongoing research is focused on developing AI algorithms for identifying and managing lung diseases like fibrosis and COPD (Chronic Obstructive Pulmonary Disease). These algorithms demonstrate accuracy when diagnosing these conditions using CT scans.

j. Medical Image Analysis:

AI algorithms have been specifically developed to automate measurements of features found in images, including X rays and MRI scans. By providing these automated measurements they improve the accuracy of diagnosis and help, with treatment planning [41].

The use of AI, in imaging is a developing area that has the potential to enhance patient care lower healthcare expenses and deliver more precise diagnoses efficiently. Nevertheless, it is essential to validate and incorporate AI algorithms into processes appropriately to guarantee their effective implementation in actual medical practice. With advancements in AI technology, it presents an opportunity, for revolutionizing the healthcare sector.

3.2 DATASET

The dataset I'm discussing here is a collection of chest X ray images that are specifically used for diagnosing pneumonia in patients. we share some details, about this dataset;

a. Data Components:

The dataset is made up of two components; Optical Coherence Tomography (OCT) images and chest X Ray images. However, for the purpose of this analysis we will be focusing solely on the chest X Ray images.

b. Pneumonia as a Health Issue:

Pneumonia poses a threat to the lives of children. It accounts for around 2 million deaths among children under the age of 5 every year [42]. It's actually one of the leading causes of death in children surpassing the mortality rates caused by HIV/AIDS, malaria and measles.

c. Global Impact:

Developing nations bear the brunt when it comes to childhood pneumonia cases with 95% occurring in countries across Southeast Asia and Africa. It's worth noting that pneumonia can be caused by both viral pathogens and managing these conditions requires approaches.

d. Importance of Radiographic Data:

Radiographic data plays a role in diagnosing pneumonia particularly through chest X rays. Timely and accurate diagnosis is absolutely vital, in order to provide treatment.

Chest X rays are commonly used to differentiate between types of pneumonia. However, it can be challenging to obtain interpretations, in environments with limited resources.

a. Dataset Purpose:

The purpose of this dataset is to assist in the classification of chest X rays from patients. Specifically, it aims to identify cases of pneumonia and distinguish between bacterial causes. This classification can help expedite the process for children who require medical attention[43].

b. Dataset Details:

The dataset consists of a total of 5,856 chest X ray images that have been categorized into two groups; "NORMAL" and "PNEUMONIA." Among the cases classified as "PNEUMONIA " 2,780 are caused by bacteria while 1,493 are attributed to viruses [44].

c. Labeling Convention:

The labeling process involves a combination of disease identification randomized identification and unique image numbering, for each patient. The diseases are categorized into three labels; "NORMAL," "VIRUS," and "BACTERIA."

d. Data Source:

This dataset includes chest X ray images obtained from pediatric patients aged one to five years old.

The pictures used in this study were obtained from Guangzhou Women and Children’s Medical Center, in Guangzhou as part of the patient’s routine care.

e. Quality Control and Labeling:

A careful assessment was performed to exclude any scans that were of quality or unreadable. The medical images were carefully evaluated by two doctors before being deemed suitable for training intelligence systems.

This dataset is an asset for developing and training AI models to diagnose pneumonia in patients. It has the potential to aid in the accurate detection of pneumonia cases in areas where access, to radiologists or medical experts is limited.

3.3 MACHINE LEARNING WITH IMAGE CLASSIFICATION

Image classification is a used task, in the field of machine learning. Its objective is to teach computers how to identify and categorize objects or patterns within images. This problem is crucial in the realm of computer vision [45]. Finds applications in domains, including facial recognition, medical image analysis and autonomous vehicles. Let’s take an overview of how machine learning techniques employed for image classification;

a. Data Collection; The initial step involved in any machine learning endeavor is. Preparing the required data. In the context of image classification this entails curating a dataset comprising images. Each image in the dataset is associated with a label or category (“cat," "dog," "car," etc.).

b. Data Preprocessing; Raw image data necessitates preprocessing steps for utilization. This may involve resizing images to a size pixel value and augmenting the dataset by applying transformations (such as rotation or flipping) to enhance training data diversity.

c. Feature Extraction; Before inputting images, into a machine learning model it's customary to extract features from them.

This can be achieved by using techniques, like Convolutional Neural Networks (CNNs) which have the ability to automatically learn features from the data. Alternatively, if you are working with machine learning algorithms you can use handcrafted methods to extract features.

d. Selecting the machine learning model for image classification is a step. Convolutional Neural Networks (CNNs) are widely preferred for this task because they can learn features from images effectively [46]. There are several popular CNN architectures available including LeNet, AlexNet, VGGNet, ResNet and Inception.

e. The chosen model is then trained using preprocessed image data. During training the model learns how to associate input images with their labels by optimizing its parameters through a loss function and an optimization algorithm like stochastic gradient descent (SGD).

f. Validation and hyperparameter tuning come next. After training the model it is crucial to evaluate its performance on a validation dataset to check for overfitting and determine the hyperparameters such as learning rate and batch size.

g. Finally once the model is trained and fine-tuned it undergoes testing and evaluation, on a test dataset to assess its real-world performance. Common evaluation metrics include accuracy, precision, recall and F1 score.

h. Once the model demonstrates performance, on the test data it can be put into action to make predictions on unseen images. This process of deployment may involve integrating the model into an application or service.

i. Continuous monitoring and upkeep of the deployed model are essential to ensure its performance over time. It is also advisable to update the model with data as part of the maintenance process.

j. To enhance performance and reduce training time leveraging trained models and fine tuning them specifically for your image classification task can be beneficial. This technique is commonly referred to as transfer learning.

Tools and libraries commonly utilized for machine learning based image classification include TensorFlow, Keras, PyTorch and scikit learn. Additionally, if computational resources are limited cloud-based services can be. Pre trained models, from these libraries can be utilized for image classification tasks. The following imaginative figure 3.1 illustrates the previous points

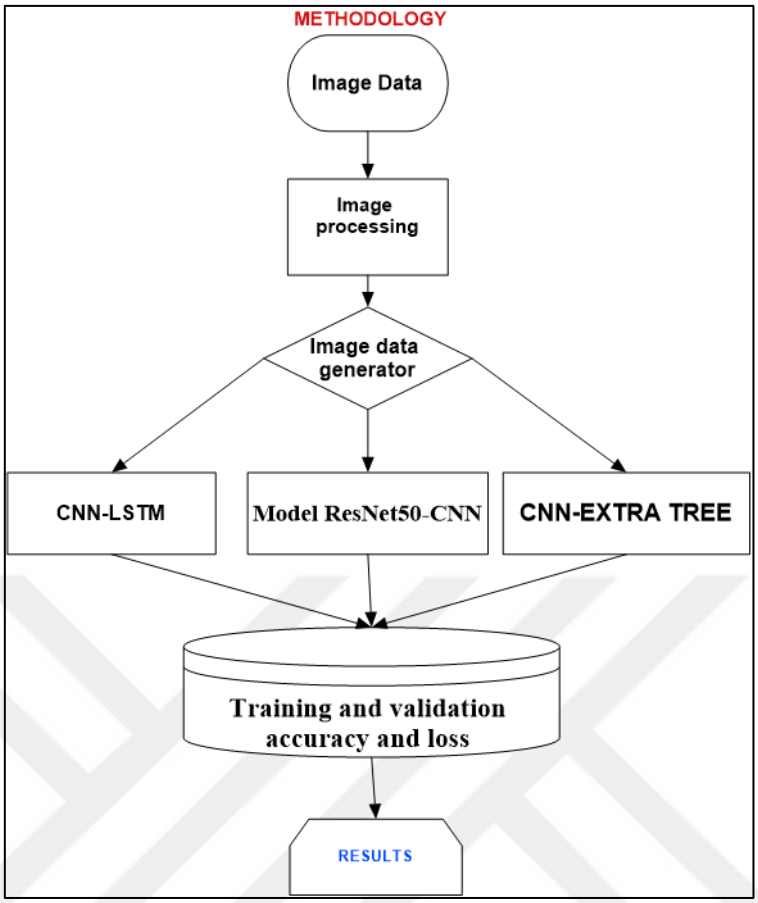

Figure 3.1: Illustrates the Steps Taken in Order to Classify Chest X-Ray.

3.4 CONVOLUTIONAL NEURAL NETWORKS (CNNS)

Convolutional Neural Networks (CNNs) are a type of networks that are specifically designed for processing structured grid data, with a primary focus on images and videos. They have brought advancements to the field of computer vision. Have found applications in various other domains as well [47]. CNNs are renowned for their ability to automatically learn and extract features from images, which makes them highly effective for tasks like image classification, object detection, facial recognition and more. The following figure 3.2 shows the process of image segmentation and feature extraction.

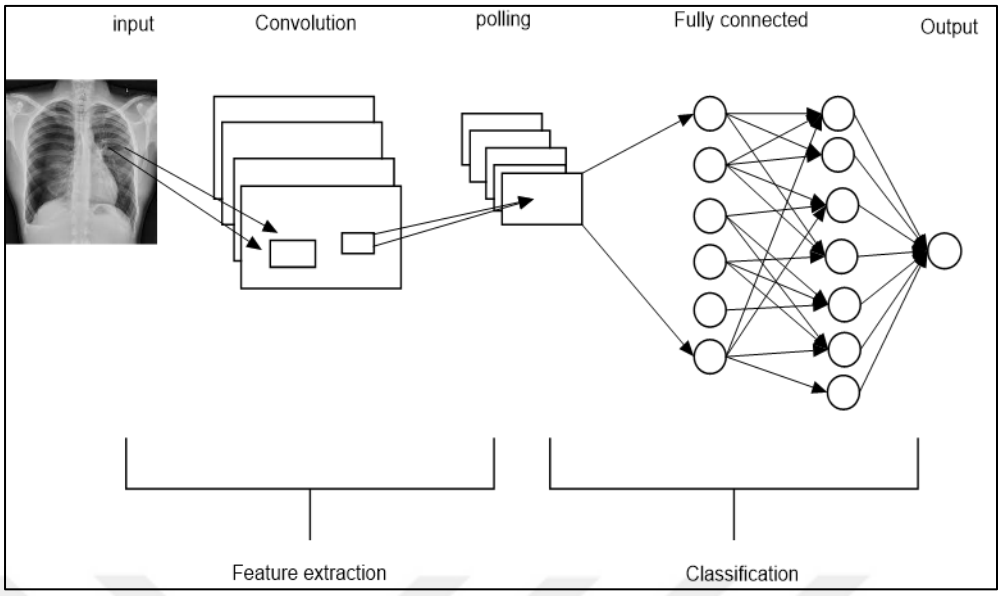

Figure 3.2: The Process of Image Segmentation and Feature Extraction.

Now let’s delve into an explanation of the components and operations within CNNs;

a. Convolution Layer:

The central operation in a CNN is called convolution. It involves moving a filter (also called a kernel) across the input image in order to identify patterns and features. These filters capture details. Extract features such as edges, textures or even more intricate patterns [48].

The outcome of performing convolution is known as a feature map. Each feature map represents the presence of a feature, within the input.

b. Stride:

Convolutional layers commonly utilize parameters that determine how much the filter moves across the input at each step. A larger stride reduces the size of the resulting feature map [49].

c. Padding:

Padding is employed to control the dimensions of the resulting feature maps.

"Valid" padding, which involves not adding any padding decreases the dimensions of the data. On the hand "same" padding adds zero padding to maintain both the input and output dimensions [50].

d. An activation function, ReLU (Rectified Linear Unit) 's applied element wise, after a convolution operation. This step introduces non linearity. Enables the network to learn patterns [].

e. Pooling layers Max Pooling or Average Pooling are used to reduce the dimensions of feature maps. This down sampling process enhances the networks' ability to handle variations in object position. Makes it more robust.

f. Connected layers come into play after convolutional and pooling layers. These layers are typically used for aggregating high level features and performing classification. Ultimately, they generate the predictions.

g. In CNNs weight sharing is a technique where the same set of weights (filters) is utilized across parts of an image. This approach helps capture features in regions of the image.

h. Deep CNNs consist of pooling layers stacked together. By stacking these layers, the network can learn representations of features ranging from simple, to complex.

i. To prevent overfitting and improve the model's ability to generalize it is common to incorporate dropout layers and employ L1/L2 regularization [51].

j. Various tasks often benefit from utilizing trained CNN models, like VGG, ResNet, Inception and MobileNet. Transfer learning involves leveraging a trained model as a feature extractor and fine tuning it to suit specific tasks.

k. In order to enhance the diversity of the training dataset and promote generalization of the model data augmentation techniques such as cropping, flipping and rotation are frequently employed.

i. The selection of a loss function depends on the task at hand; for instance, categorical cross entropy is commonly used for classification purposes. During training optimizers like Adam or Stochastic Gradient Descent (SGD) are utilized to update the weights of the model [52].

m. When evaluating classification tasks, accuracy, precision, recall, F1 score and confusion matrices are, among the employed metrics.

CNNs have revolutionized applications significantly. Have played a pivotal role in advancing the field of computer vision.

"They play a role, in cutting edge AI applications and remain a thriving field of ongoing research and advancement."

3.5 CNN- EXTRA TREE

Combining Convolutional Neural Networks (CNNs) with Extra Trees, an ensemble learning algorithm is not an approach. However, in scenarios where you aim to leverage

the strengths of both techniques it can be an option. It is crucial to understand when and why one would choose to combine these methods.

Here's how you can effectively utilize CNNs and Extra Trees together

a. Feature Extraction using CNN;

To start employ a CNN for feature extraction. CNNs excel at capturing spatial features from images. You can train the CNN for image classification by removing the connected layers at the end [53].

Once trained or utilizing a trained model like VGG, ResNet or Inception extract feature vectors from the last convolutional layer of the CNN for each image in your dataset. These feature vectors encapsulate the information about the images.

b. Integration of Features:

After obtaining the feature vectors from the CNN employ them as input for an Extra Trees classifier. Extra Trees is a learning algorithm that falls under forests and proves effective, in handling high dimensional data while making predictions based on these features [54].

Step 3; Utilize a Convolutional Neural Network (CNN) to extract features and then employ an Extra Trees classifier to make predictions using those extracted features.

Step 4; Additionally, you can incorporate the output of the Extra Trees model as a feature or input, in conjunction with another method (such as SVM, k NN or gradient boosting) to further enhance the accuracy of classification [55].

The reason behind this combination is to leverage the CNNs feature extraction capabilities along with the robustness of Extra Trees, in handling dimensional data and providing a reliable ensemble-based classification.

However please note that this approach may increase model complexity and necessitate additional computational resources and experimentation to strike the balance and combination of models. It's also important to consider your dataset and problem context when determining whether this amalgamation of techniques is appropriate. In some cases, an optimized CNN alone might suffice for image classification without requiring an ensemble method.

3.6 DEFINITION OF THE CNN MODEL

Firstly, you have created a model using Kera’s. This model consists of layers, such, as Conv2D (Convolutional 2D) MaxPooling2D and Dense (Connected) layers.

The CNN architecture utilizes Conv2D and MaxPooling2D layers, which're responsible for extracting features and reducing spatial dimensions.

To convert the output from the layers into a 1D vector we incorporate a Flatten layer.

Subsequently two Dense layers are added for classification purposes. Additionally, we include a dropout layer to prevent overfitting.

Model Compilation.

Next, we compile the CNN model using the 'adam' optimizer and 'sparse_categorical_crossentropy' loss function. The 'adam' optimizer is widely used in training networks. In this case our loss function is suitable for class classification problems.

During training we monitor accuracy as one of our metrics.

Feature Extraction;

In this example we utilize the CNN model to extract features from our data. It's important to note that we're not training the CNN itself; instead, we use it as a feature extractor. By employing the model. Predict() function, on our datasets (x_train x_test and x_val) we obtain these extracted features. [56]

3.7 EXTRA TREES CLASSIFIER

To begin you will need to import the from scikit learn which is a used machine learning library.

Next create an instance of the called et_classifier.

Now let’s focus on training the Extra Trees Model;

Train the et_classifier using the features extracted from the CNN model (features_train) and their corresponding labels (y_train). The Extra Trees classifier utilizes these extracted features, for classification.

This approach allows you to take advantage of the feature extraction capabilities of a trained CNN model and utilize those features as input for an ensemble classifier like Extra Trees. It's a method to improve the performance of your image classification task particularly if the CNN features provide information for your specific problem.

Keep in mind that achieving performance may require tuning hyperparameters for both the CNN model and the Extra Trees classifier. Additionally ensure that your dataset is well

preprocessed and balanced. Evaluating your model's performance on a validation set and tuning hyperparameters can play a role, in obtaining results.[57]

3.8 DEEP LEARNING WITH IMAGE CLASSIFICATION

Deep learning for image classification is a powerful approach that leverages neural networks, particularly Convolutional Neural Networks (CNNs), to automatically learn and extract relevant features from images and classify them into different categories. Here's a step-by-step guide on how to implement deep learning for image classification:

a. Data Collection and Preprocessing: Gather a labeled dataset of images, with each image associated with a specific category or label.
b. Preprocessing the data entails several steps, including scaling photos to a uniform size, normalising pixel values, and enriching the data by transformations such as rotation, flipping, and cropping. These measures aim to enhance the variety of training instances.
c. Model Architecture: Select an appropriate deep learning framework for the task of picture classification. Convolutional neural networks (CNNs) are often preferred for this particular purpose. You have the option to create your own Convolutional Neural Network (CNN) structure or utilise pre-existing models such as VGG, ResNet, Inception, or MobileNet. A conventional CNN structure for image classification comprises convolutional layers for extracting features, pooling layers for reducing resolution, and fully connected layers for categorization.
d. Model Training: Partition your dataset into training, validation, and test sets. The training set is utilised for model training, the validation set aids in hyperparameter tuning and overfitting monitoring, and the test set is employed for evaluating the model's performance.
e. Utilise the training data to train the deep learning model. This is the optimisation of model parameters, namely weights and biases, by the utilisation of an optimisation technique such as stochastic gradient descent (SGD). The objective is to minimise a loss function, such as categorical cross-entropy.
f. Hyperparameter Tuning:Conduct trials with hyperparameters such as learning rate, batch size, layer count, and neuron count per layer to determine the optimal model performance on the validation set.Regularization and Dropout: Use techniques like dropout and batch normalization to prevent overfitting.

g. Transfer Learning (Optional):If you possess a restricted dataset, it would be advisable to employ transfer learning. One can employ pre-trained models and refine them to suit their particular picture categorization objective.
h. Model Evaluation:Assess the real-world performance of your trained model by evaluating it on the test set. Standard evaluation measures comprise accuracy, precision, recall, F1-score, and confusion matrices.
i. Deployment:If your model performs well, you can deploy it for making predictions on new, unseen images. This might involve integrating the model into an application, website, or service.
j. Monitoring and Maintenance: Regularly supervise and uphold your implemented model to guarantee its consistent performance over an extended period. Regular retraining with updated data may be required.
k. Data Augmentation (Optional): Data augmentation techniques can be used to artificially increase the size of your training dataset by applying random transformations to the images during training, which can improve model generalization.
i. Ensemble Methods (Optional): Consider using ensemble methods like stacking or bagging to combine multiple deep learning models for improved accuracy.

Deep learning has been highly successful in image classification tasks, especially when applied to large and diverse datasets. TensorFlow, Keras, PyTorch, and other deep learning frameworks provide tools and libraries for implementing deep learning models for image classification. The choice of architecture and fine-tuning depends on the specific problem and data characteristics.

3.9 COMBINING CONVOLUTIONAL NEURAL NETWORKS (CNNS) WITH LONG SHORT-TERM MEMORY (LSTM)

Networks are a common approach in deep learning for tasks that involve both spatial and temporal information, such as video analysis, action recognition, and sequential image data. This combination allows you to leverage the feature extraction capabilities of CNNs for spatial information and the sequence modeling capabilities of LSTMs for temporal information. Here's a high-level overview of how to use CNNs and LSTMs together shown in figure 3.3 [24].

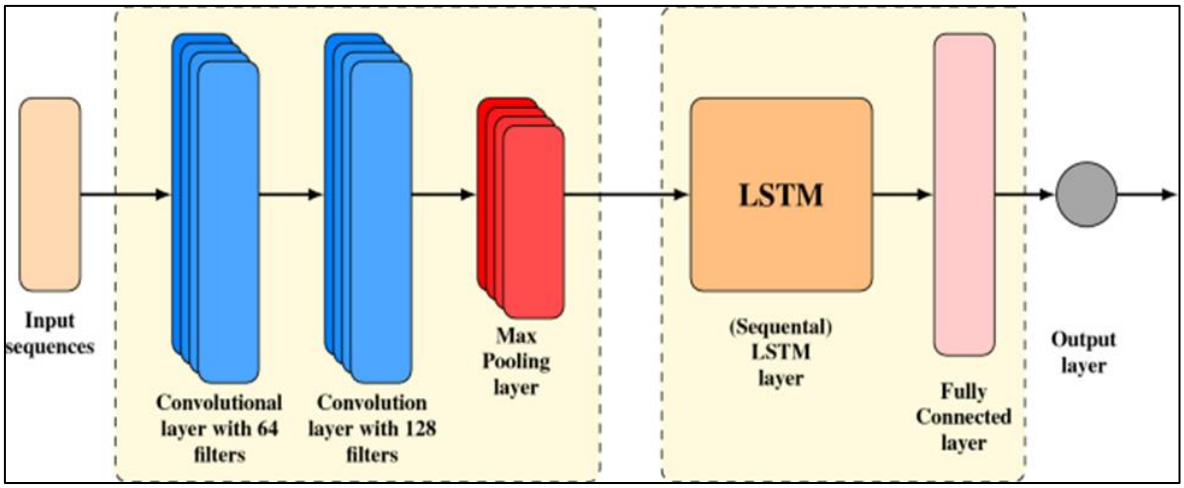

Figure 3.3 CNNs and LSTMs Layer [29].

a. Data Preparation:

Collect a dataset that consists of sets of images or video frames, with corresponding labels or annotations. Each set can be considered as a series of images over time.

Preprocess the images, which includes resizing, normalization and applying any data augmentation techniques.

b. CNN for Feature Extraction:

Utilize a CNN as the component of your model to extract features from each frame in the input sequence. The CNN will take each frame as input. Generate a fixed size feature vector for every frame.[58]

d. Sequence Formation:

Arrange the CNNs feature vectors along the axis to create a 3D input tensor. The dimensions would be (batch_size, sequence_length, feature_vector_size).

The sequence_length represents the number of frames in your sequence.

e. LSTM for Sequence Modeling:

Incorporate an LSTM layer (or multiple LSTM layers), on top of the CNN features. These LSTM layers will process the sequence of feature vectors.

The LSTM network has the capability to capture dependencies within the sequence and produce a series of states [59].

f. Output Layer:

Add a connected) layer or a softmax layer, to the LSTMs output. This will generate predictions or classifications based on the features.

g. Training:

Train the model from start to finish by using a loss function (like categorical cross entropy for classification) and an optimization algorithm such as Adam or RMSprop. You'll need labeled sequences for learning.

h. Hyperparameter Tuning:

Experiment with hyperparameters like the number of LSTM units, layers, learning rate, batch size and dropout rate. This will help optimize your model's performance.

i. Evaluation and Testing:

Assess the model's performance by evaluating it on a validation dataset. Common evaluation metrics, for data include accuracy, precision, recall and F1 score [60].

Estimate the real-world performance of your model by testing it on data.

j. Deployment:

Implement the trained model to make predictions, on sequences or videos.

k. Monitoring and Maintenance:

Consistently keep an eye on the performance of the deployed model. Make updates particularly if there are any changes in the data distribution, over time.

The use of Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks is highly efficient when it comes to tasks such, as recognizing actions, in videos identifying gestures and any other application that requires capturing both temporal information. The selection of CNN architecture LSTM architecture and the overall model design depends on the problem you are addressing and the unique of your characteristics.

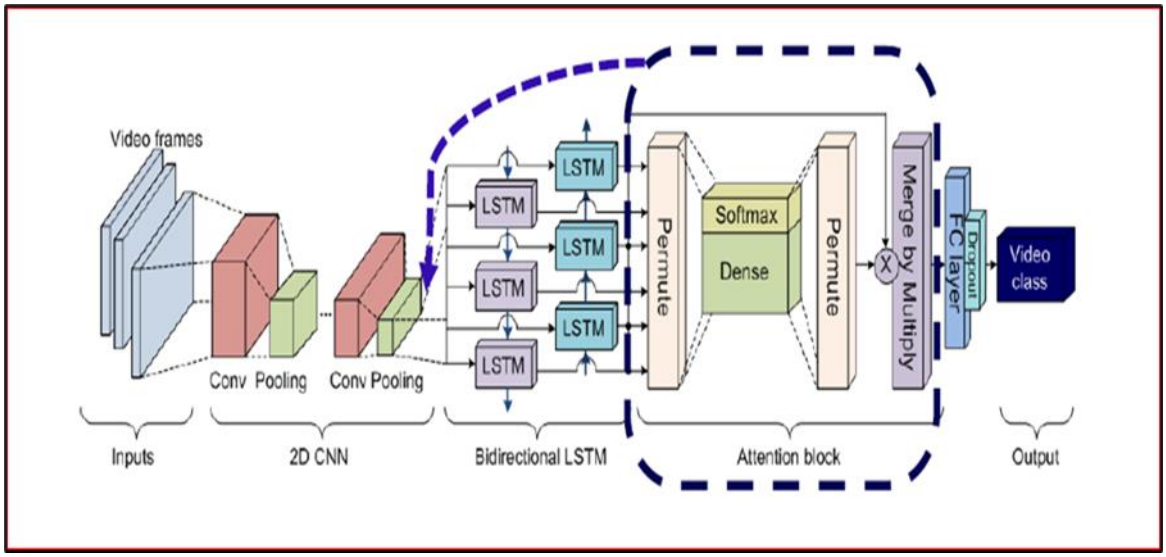

Figure 3.4: Long Short-Term Memory (LSTM) Layer When Combined with CNN [63].

3.10 TRANSFER LEARNING RESNET-50

Transfer learning using ResNet 50 is a technique, in learning that involves utilizing a trained ResNet 50 model, [64]which has been trained on a vast dataset for image classification and adjusting it to suit a different but related task. This approach is highly effective when you have limited data for your task or when you want to benefit from the knowledge acquired by the pre trained model.

Here is a step-by-step breakdown of how transfer learning with ResNet 50 operates;

a. Pre trained ResNet 50 Model:

Begin with a trained ResNet 50 model that has been extensively trained on a dataset like ImageNet. The model has already acquired the ability to identify features and patterns in images.

b. Eliminate the Top Classification Layer:

In order to adapt the model to your task it is customary to remove the classification layer of ResNet 50, which is designed specifically for its original classification purpose.

c. Customize the Classification Layer:

Integrate a classification layer onto the existing model that aligns with the number of classes relevant, to your specific task.

For instance, if you're engaged in classification, you might include a neuron, with a sigmoid activation function. In the case of class classification, you would add multiple neurons employing softmax activation as shown in figure 3.5.

d. Optional; Freezing trained Layers:

Depending on the size of your dataset you have the option to freeze the weights of pre trained layers (those, from the original ResNet 50) to prevent them from being updated during training. Freezing can be helpful when dealing with data as it allows you to retain the knowledge acquired by the pre trained model.

e. Training:

Train your model using your dataset. This process involves inputting your training data into the network calculating loss and updating the models' weights using an optimization algorithm. Throughout this process the network will learn how to adapt itself to your task.

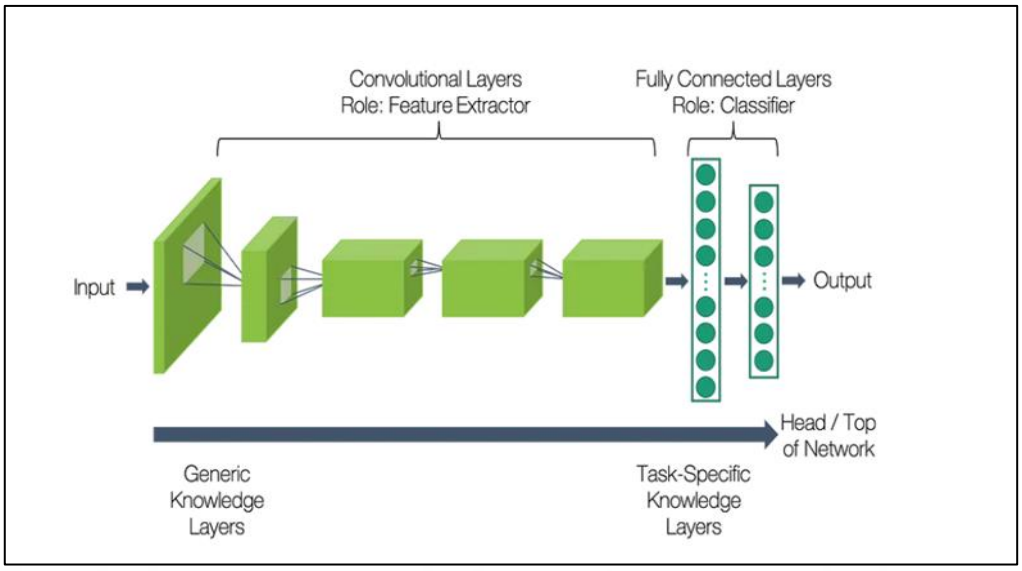

Figure 3.5: Customize the Classification Layer [65].

f. Optional Step; If desired you have the option to refine and enhance the model further by unfreezing certain or all of the trained layers and continuing the training process. This fine-tuning step helps optimize the model, for tasks leading to performance.

g. Validation and Testing; Once you have completed the training and potential fine-tuning stages it is crucial to assess your models performance using a validation dataset. Based on these evaluation results make any adjustments to improve its effectiveness. Finally put your model to the test by evaluating its real-world performance on a test dataset.

One of the advantages of utilizing ResNet 50 for transfer learning is that it saves both time and resources. By leveraging a existing model that has already learned an extensive range of features from a large dataset you can adapt it for various tasks such as image classification, object detection or even domain specific applications like medical image analysis. This approach proves beneficial in cases where creating a new massive dataset from scratch would be impractical[66].

It's important to note that the success of transfer learning relies heavily on how related the training task is, to your target task. The more similarity exists between them the greater potential benefits transfer learning can offer.

Here are some additional considerations and tips when employing transfer learning with ResNet 50;

Here are some steps to follow when preparing your data for model:

a. Preprocess the data; Make sure to preprocess the data in a way that matches the training process of ResNet 50. This typically involves resizing images to fit the models input size values, within a range of [0, 1] or [1, 1] and applying any desired data augmentation techniques.

b. Use data augmentation; Implementing techniques like cropping, flipping, rotation and color adjustments can be beneficial in increasing the effective size of your training dataset and improving the overall generalization of your model.

c. Select layers for tuning; When performing fine tuning on your model you have the flexibility to choose which layers to unfreeze. Generally deeper layers capture features while shallower layers focus on task specific features. Experiment with strategies for tuning to find what works best for your specific scenario.

d. Employ learning rate scheduling; Consider using learning rate schedules or adaptive learning rate techniques such as learning rate annealing during training. These approaches can help optimize convergence during the tuning phase.

e. Apply regularization techniques; To prevent overfitting especially when working with datasets apply regularization methods such as dropout and L2 regularization.

By following these steps you can effectively prepare your data for training, with ResNet 50. Enhance its performance.

f. Adjust the size of the batches to match the memory and hardware resources. Smaller batch sizes may result in gradients but they might be necessary when resources are limited.

g. Keep an eye, on your models performance during training by using validation datasets. If you notice a decline in performance consider implementing stopping.

h. Gain insights into what your model has learned by visualizing and interpreting the features it has acquired. Techniques like feature visualization and occlusion analysis can be helpful in this regard.

i. Save checkpoints of your model as you train it. This will allow you to resume training from a point if needed and retain the performing weights for evaluation purposes.

j. Tune your models hyperparameters, such as the number of epochs, learning rate and layer count through experimentation. Hyperparameter tuning can have an impact on your models performance.

k. Remember that transfer learning is not limited to image classification ! You can also utilize trained ResNet 50 models for related tasks in different domains, like object detection or image segmentation.

l. Deploying the Model;

Once you have a trained model it's time to deploy it for making predictions, on data. Ensure that you adapt the model to fit the inference environment and optimize it for performance.

Utilizing transfer learning techniques with models like ResNet 50 can be highly advantageous. This approach can save both time and resources while enhancing the performance of your learning models. It is widely adopted in real world scenarios. Has consistently shown effectiveness across tasks and domains.

Here are some additional factors to consider and best practices for successful transfer learning with ResNet 50;

m. Adapting to Domains;

If your target domain significantly differs from the source domain on which ResNet 50 was originally trained it becomes important to consider techniques for domain adaptation. This may involve gathering domain data or utilizing domain adaptation methods to fine tune the model ensuring better alignment with your data distribution.

n. Enhancing Model Generalization through Label Smoothing;

In cases incorporating an amount of label smoothing into your loss function can enhance the generalization capabilities of your model. Label smoothing involves using a target label distribution during training, which helps prevent overfitting issues.

o. Handling Class Imbalance;

It is essential to address any class imbalance, in your dataset particularly if certain classes are heavily underrepresented compared to others.

Here are some strategies to ensure a dataset;

p. Data Quality;

It's crucial to have high quality training data as the performance of the model greatly depends on it. Noisy or mislabeled data can have an impact, on the transfer learning process.

q. Feature Extraction vs. Fine Tuning;

You need to decide whether you want to extract features (using the trained ResNet 50 as a fixed feature extractor) [34]or fine tune the model. Feature extraction is useful when you have data while fine tuning allows the model to adapt closely to your specific task.

r. Multi Modal Transfer Learning;

Transfer learning can also be applied to tasks that involve modes of data. For example, you can use a trained ResNet 50 for image classification and combine it with another model for text analysis when working on tasks like image text matching.

s. Continuous Learning;

If your application requires adapting to changing data distributions over time consider implementing techniques for learning and updating your model accordingly. This ensures that your model remains effective, in environments.

t. Interpretable Features;

Think about techniques that can make the features learned by the trained ResNet 50 more interpretable.

Feature visualization and saliency maps are tools, for gaining insights into what the model focuses on when making predictions.

Step 28; Take advantage of learning frameworks like TensorFlow, PyTorch and Keras which provide trained models and tools for transfer learning. These frameworks simplify the process for you.

Step 29; Speed up the training and tuning process by utilizing hardware acceleration such as GPUs or TPUs. Faster hardware can significantly reduce the time required for training.

Step 30; Consider employing learning techniques by combining the predictions from models, including different variations of ResNet 50 or other architectures. Ensemble models often yield better performance results.

When applying transfer learning with ResNet 50 you'll find it to be a technique that can be adapted to tasks. Its success relies on data preparation experimenting with hyperparameters and understanding the specific requirements of your target task. Continuously. Evaluate your models performance to make improvements.

3.11 RESNET 50 ARCHITECTURE

ResNet 50 is one variant among several in the ResNet (Residual Network) architecture family. The term "50" indicates the number of layers, in this network. Here's a breakdown of its structure;[67]

Step 1; Initial Convolution Layer;

The network begins with a layer of convolution that has a size of 7x7 and includes 64 filters. It also has a value of 2. After that batch normalization and ReLU activation are applied.

Next we have a max pooling layer which uses a kernel size of 3x3 and a value of 2. This helps to reduce the dimensions.

The key building blocks, in ResNet 50 are called blocks. These blocks consist of layers along with shortcut connections. There are four types of blocks in ResNet 50;

a. Basic Blocks; These blocks consist of two layers with a size of 3x3 along with a connection.

b. Bottleneck Blocks; These blocks have three layers (1x1, 3x3 1x1) combined with a connection. They are used in layers to decrease complexity.

c. Convolutional Blocks; These blocks use a 1x1 convolution, for downsampling.

d. Identity Blocks; These blocks do not contain any layers. Are used to maintain the spatial dimensions.

ResNet 50 stacks these blocks together to increase the depth of the network.

The number of blocks, in each stage is as follows;

Stage 1; There are 3 blocks.

Stage 2; There are 4 blocks.

Stage 3; There are 6 blocks.

Stage 4; There are 3 blocks.

After the stage of blocks ResNet 50 applies global average pooling. This process converts the information in the feature maps into a one vector.

The final classification layer consists of a connected layer with 1,000 units. Uses the softmax activation function. It's important to note that this layer is specifically designed for the ImageNet classification task and is typically replaced when using transfer learning.

ResNet 50 is renowned for its depth. Was instrumental in introducing the concept of connections. These connections enable the training of neural networks by addressing the

vanishing gradient problem. As a result, ResNet 50 and its variations have become architectures in computer vision. Find extensive use in various image related tasks.

In addition to image classification ResNet 50 has also shown success in object detection semantic segmentation and other related applications. Researchers have even developed variants, like ResNet 101 and ResNet 152 to achieve higher performance levels.

3.12 CLASSIFICATION_REPORT

The classification_report function is widely used in the field of machine learning. Can often be found in libraries, like scikit learn for Python. Its main purpose is to generate a report that includes performance metrics for classification tasks. This report is quite helpful in evaluating the performance of a classification model. Some of the metrics included in the report are precision, recall, F1 score and support for each class. Lets break down what each of these metrics signifies;[68]

a. Precision (also known as Positive Predictive Value);

Precision measures the accuracy of predictions by calculating the ratio between positives and the sum of true positives and false positives. A high precision value suggests that the model correctly identifies instances while minimizing positives.

b. Recall (also known as Positive Rate or Sensitivity);

Recall evaluates the models ability to correctly identify all instances by calculating the ratio between true positives and the sum of true positives and false negatives. A high recall value indicates that the model successfully identifies instances.

c. F1 Score:

The F1 score provides a measure between precision and recall by calculating their mean. It is particularly useful when you want to strike a balance, between minimizing positives and false negatives.

d. Explanation:

Support refers to the count of instances belonging to each category in the dataset. It provides information, about the number of samples in each class.

The classification_report function is commonly used in class classification problems to present these metrics for every class. In classification you can expect to see these metrics for both the negative categories.

When utilizing the classification_report function you need to provide the labels (y_true). Predicted labels (y_pred) as inputs. This function then generates a report that outlines the performance of your classifier for each class in the classification task. It serves as a tool, for evaluating how well certain classes are predicted and identifying areas where improvements can be made in your classification model.

3.13 OPTIMIZERS RMSPROP

RMSprop, short, for Root Mean Square Propagation is an algorithm used in the training of networks especially in the deep learning field. It serves as an optimization technique that addresses some of the limitations of descent. Here is a detailed explanation and description of RMSprop;[69]

a. Adaptive Learning Rate;

One notable aspect of RMSprop is its utilization of a learning rate. Unlike descent where the learning rate remains fixed RMSprop dynamically adjusts the learning rate for each model parameter during training. This adaptability allows for efficient convergence.

b. Squared Gradients;

RMSprop keeps track of the gradients for each model parameter through a moving process. The squared gradient is obtained by squaring the gradients at each iteration.

c. Exponential Moving Average;

To compute and update the squared gradients RMSprop employs a weighted moving average method. This moving average takes into account gradients with a decay factor that determines their influence, on the calculation. In this case we denote it as E[g^2] where g represents the gradient.

d. Adjusting the learning rate;

To tune the learning rate, for each parameter we utilize a moving average of gradients. This adjustment is based on the notion that parameters with gradients in the past should have their learning rates decreased, while parameters with gradients should have their learning rates increased for more efficient convergence.
The formula used for this adjustment is;

updated_learning_rate = learning_rate / sqrt(E[g^2] + epsilon)

Here E[g^2] represents the moving average of gradients "learning_rate" refers to the learning rate and "epsilon" is a small constant added to ensure numerical stability. The division by the root scales the learning rate based on gradient magnitudes. It effectively reduces the rate for parameters with gradients. Increases it for those with smaller ones

Updating model parameters;

The actual update of each model parameter occurs by utilizing the adjusted learning rate and current gradient. The update rule can be summarized as follows;

new_parameter = old_parameter. (updated_learning_rate) * gradient

To find a value for a parameter we scale its corresponding gradient by the adjusted learning rate and subtract it from its current value. This process is repeated individually for each parameter during training.

RMSprop is widely employed as an optimization algorithm in learning due to its ability to adapt to variations, in characteristics.

It is highly effective, in addressing challenges such as the vanishing or exploding gradients, which're issues when training deep neural networks. Moreover it tends to result in improved stability and faster convergence in tasks, with optimization landscapes

4. RESULTS ANALYSIS

4.1 DATA PREPROCESSING

Data preprocessing plays a role, in the analysis of data and machine learning. It encompasses the process of cleaning and transforming data to ensure its suitability, for analysis and training models. Considering the dataset you have described it is important to give consideration to this aspect.

4.2 IMAGE RESIZING

The dataset comprises photographs, with resolutions ranging from 800 x 400 to 1200 x 800. Utilizing high resolution images in a program or model can add processing time and overhead. To tackle this the images have been resized to a resolution of 150 pixels on each side.

Resizing images to a resolution is a practice in both image analysis and machine learning. It helps standardize the data making it easier to work with and reducing complexity. This step is crucial for ensuring that all images share the dimensions, which's often necessary when training machine learning models.

By resizing the images to 150 pixels on each side it is highly likely that the datasets size has significantly decreased. This reduction makes it more manageable for analysis and model training purposes. Additionally, having a consistent image size simplifies the process of inputting these images into machine learning models that expect fixed input sizes.

Data preprocessing steps might also involve tasks like data cleaning, normalization, feature engineering or other actions, to the analysis or machine learning task at hand. These steps aim to enhance data quality and facilitate accurate and effective analysis or modeling.

4.3 MODEL DESIGN

In this study we utilized three models; ResNet 50 CNN, CNN LSTM and CNN Extra Trees.

4.3.1 ResNet-50 -CNN

The provided code showcases how to create a custom learning model, for image classification using the ResNet50 architecture. Here's a breakdown of the code;

a. Importing Essential Libraries;

The code begins by importing libraries including the RMSprop optimizer from TensorFlow/Keras. This optimizer will be utilized to train the model.

b. Loading Pre trained ResNet50 Model;

In this section the code loads a trained ResNet50 model using the ResNet50 function from Keras. It specifies parameters;

Weights='imagenet'; This indicates that the model should be initialized with trained weights that were trained on the ImageNet dataset.

Include_top=False; This signifies that the final classification layer of ResNet50 is excluded, as you will be adding your own custom classification layers.

Input_shape=(150, 150, 3); This parameter defines the expected input shape, for your data. For this scenario it assumes input images of size 150x150 pixels with three color channels (RGB).

c. Building Custom Layers on Top of the Base Model;

In this step additional layers are added on top of the trained ResNet50 model to enhance its performance.

The BatchNormalization layer is introduced to normalize the output, from the layer promoting training stability and convergence.

To reduce the spatial dimensions of the feature map before fully connected layers we incorporate a GlobalAveragePooling2D layer.

Multiple Dense layers are included to introduce non linearity into the model.

To prevent overfitting during training a Dropout layer is applied, randomly setting a fraction of input units to zero.

The last Dense layer serves as the output layer with a number of units to num_classes (the number of classes in your classification task). It utilizes softmax activation, for class classification.

d. Creating the Model;

This step involves using Keras Model function to construct the model. It specifies both the input (based on the base models input) and output (defined as earlier) for this custom learning model.

The resulting model object represents your tailored deep learning architecture.

You can use this code to create your custom image classification model. It begins by utilizing the feature extraction capabilities of the trained ResNet50 model and then incorporates additional layers to tailor it for your specific classification needs. Before training it on your dataset you have the option to compile the model with a loss function, optimizer and metrics.

```
[ ] num_classes=2
    from tensorflow.keras.optimizers import RMSprop
    # Load the pre-trained ResNet50 model
    base_model = ResNet50(weights='imagenet', include_top=False, input_shape=(150, 150, 3))

    # Add custom layers on top of the base model
    x = base_model.output
    x = BatchNormalization(axis=-1, momentum=0.99, epsilon=0.01)(x)
    x = GlobalAveragePooling2D()(x)  # Global average pooling layer
    x = Dense(64, activation='relu')(x)
    x = Dense(32, activation='relu')(x)
    x = Dense(16, activation='relu')(x)
    x = Dropout(rate=0.2, seed=42)(x)
    output = Dense(num_classes, activation='softmax')(x)

    # Create the model
    model = Model(inputs=base_model.input, outputs=output)
```

Figure 4.1 The Code Used for Creating Custom Model.

In the code shown in figure 4.1 you're putting together the custom learning model that you've created using TensorFlow/Keras. Compiling the model involves setting up the optimizer, loss function and evaluation metrics for training. Lets break down the code;

model.compile`; This is a method used to compile the model.

optimizer=tf.keras.optimizers.RMSprop()`; Here you specify the optimizer to train your model. You're using the RMSprop optimizer provided by TensorFlow/Keras. RMSprop adjusts the learning rate as training progresses. Is commonly used in learning tasks.

loss='sparse_categorical_crossentropy`; This parameter defines the loss function used during training. The 'sparse_categorical_crossentropy' loss is typically employed for class classification tasks when your target values are integers representing class indices.

metrics=['accuracy']`; You indicate which evaluation metric to use for your model. In this case 'accuracy' is chosen to monitor training progress. It measures how accurately samples in the training data are classified.

```
# Compile the model
model.compile(optimizer=tf.keras.optimizers.RMSprop(),
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

Figure 4.2 The Code Used to Compile the Model.

Currently we have employed 100 epochs. The outcomes indicated measures of accuracy, validation accuracy, loss and validation loss.

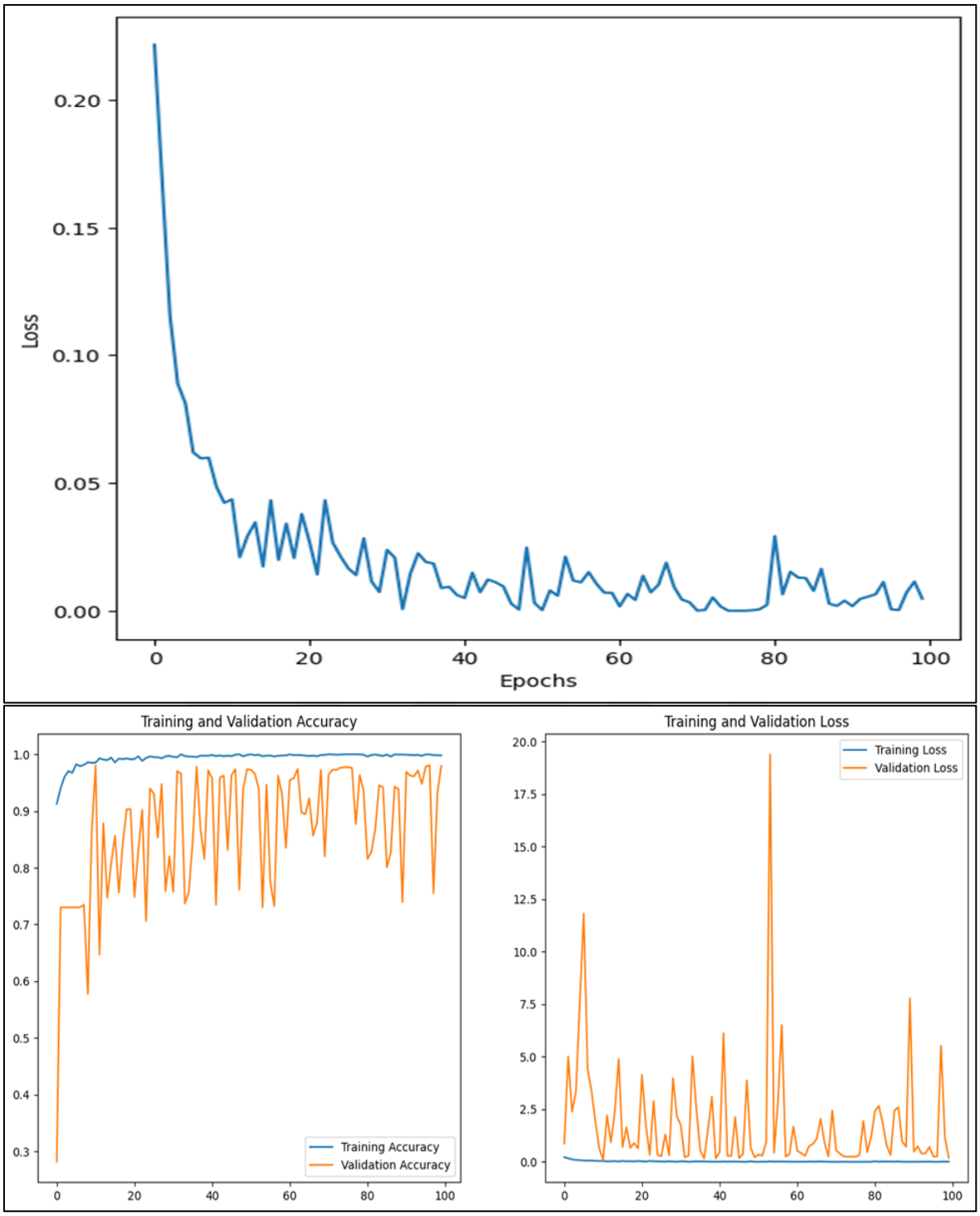

Figure 4.3: Training and Validation Performance.

The findings indicated that the CNN model achieved a level of accuracy in tracking the loss progression for activations during the 100 epochs. According to Table 5.1 the CNN model consistently predicted the loss accurately across activations, throughout the duration of 100 epochs.

Table 4.1: Performance of Our Results.

Best Result	Validation Loss	Validation Accuracy	Accuracy	Loss	Time
Epoch 11/100 1 29/129	0.0976	0.9806	0.9854	0.0436	25s 195ms/step

The classification report yields metrics for each class in the target data. These metrics, such, as precision, recall and F1 score help analyze the efficiency of the classification model.

Here's an example of how we utilize these indicators in the categorization report.

Firstly, let's see how the results turned out when we ran them through the CNN algorithm.

```
[ ] predictions = model.predict(x_val)
    pred=np.argmax(predictions,axis=1)
    pred = pred.reshape(1,-1)[0]
    print(classification_report(y_val,pred, target_names = ['NORMAL(Class 0)','PNEUMONIA (Class 1)']))

    28/28 [==============================] - 3s 53ms/step
                        precision    recall  f1-score   support

        NORMAL(Class 0)       0.98      0.95      0.96       237
    PNEUMONIA (Class 1)       0.98      0.99      0.99       641

                  accuracy                           0.98       878
                 macro avg       0.98      0.97      0.97       878
              weighted avg       0.98      0.98      0.98       878
```

Figure 4.4: Cnn Code with Performance.

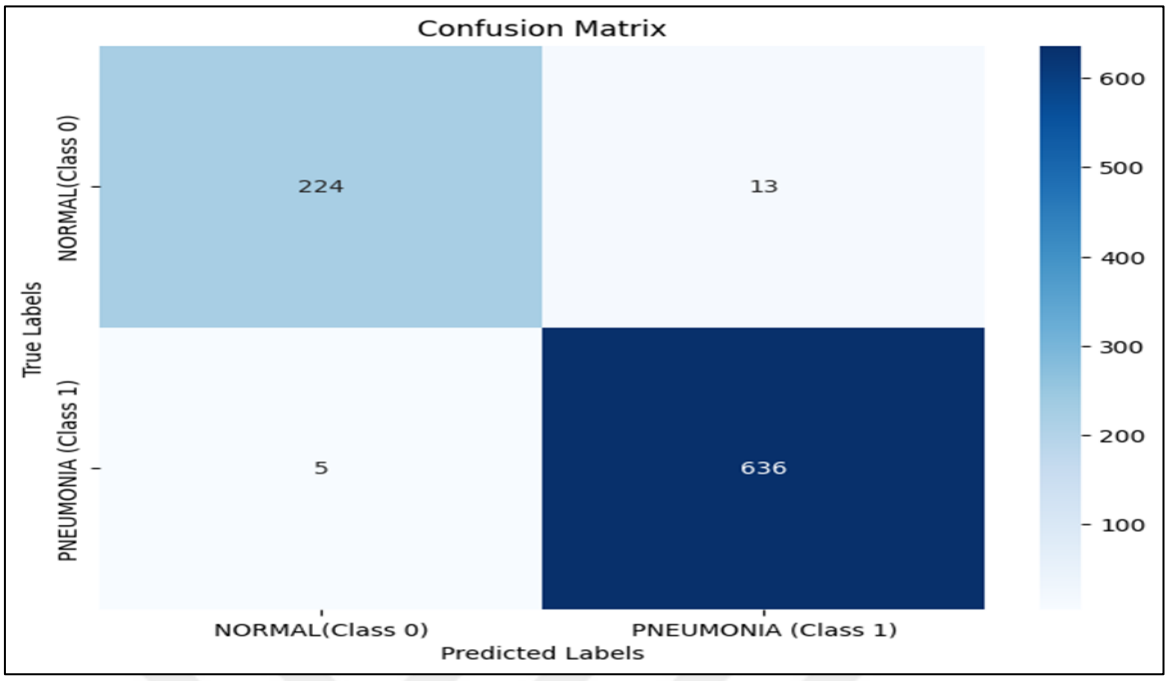

Figure 4.5: CM of CNN Performance.

4.3.2 Cnn-Lstm

The provided code defines a Keras Sequential model that combines CNN) and LSTM) layers. Lets break down the architecture of the model;

a. Input Layer:

The models input shape is specified as (1, 150 150 3) which means it expects images of size 150x150 pixels with 3 color channels (RGB).

b. TimeDistributed Convolutional Layers:

These layers apply 2D operations to extract features from the input images. The architecture includes TimeDistributed layers, with increasing numbers of filters; 512, 128 128 64 and 32.

Each convolutional layer uses a kernel size of (3,3). Applies the ReLU activation function.

After each layer there is a TimeDistributed layer that reduces the spatial dimensions of the extracted feature maps using a pool size of (2,2).

c. TimeDistributed Flatten Layer:

Following the layers there is a TimeDistributed Flatten layer. This layer reshapes the extracted feature maps from a two form into a one vector.

d. LSTM Layer:

The model incorporates an LSTM layer with 80 units, for processing.

The sequential data, which is obtained from the images using layers is processed by the LSTM layer. By setting the parameter return_sequences=False the LSTM layer provides the sequence state as output.

Next there is a layer, with 2 units and a sigmoid activation function following the LSTM layer. This layer handles the classification task. The models output is a 2D vector that represents probabilities for classification (“yes" or "no" for binary classification tasks).

To summarize the models architecture we use the) function. It displays information about the types of layers their output shapes and the total number of parameters in the model.

This architecture combines CNNs for extracting features from images and an LSTM for processing data. It is commonly used when working with time series data or sequences of images where both spatial and temporal information're crucial, for accomplishing a task. The model focuses on classification. Provides probability outputs for two classes.

To utilize and train this model effectively you will need to compile it with an optimizer, loss function and metrics. Afterward fitting the model, to your data can be achieved by utilizing the functions model.compile() and model.fit(). Additionally it is crucial to ensure that your training data and labels are compatible, with this architecture.

```
model = Sequential()
model.add(TimeDistributed(Conv2D(512,(3,3),activation='relu'),input_shape=(1,150,150,3)))
model.add(TimeDistributed(MaxPooling2D(pool_size=(2,2))))

model.add(TimeDistributed(Conv2D(128,(3,3),activation='relu')))

model.add(TimeDistributed(MaxPooling2D(pool_size=(2,2))))
model.add(TimeDistributed(Conv2D(128,(3,3),activation='relu')))

model.add(TimeDistributed(MaxPooling2D(pool_size=(2,2))))
model.add(TimeDistributed(Conv2D(64,(3,3),activation='relu')))
model.add(TimeDistributed(MaxPooling2D(pool_size=(2,2))))
model.add(TimeDistributed(Conv2D(32,(3,3),activation='relu')))
model.add(TimeDistributed(MaxPooling2D(pool_size=(2,2))))

model.add(TimeDistributed(Flatten()))

#RNN
model.add(LSTM(80,return_sequences=False))

model.add(Dense(2,activation='sigmoid'))

model.summary()
```

Figure 4.6: Implementation Code of Proposed Algorithm.

Compiling your Keras model using the RMSprop optimizer with specific hyperparameters. Here's a breakdown of the code:

a. Import RMSprop Optimizer:

i. The code starts by importing the RMSprop optimizer from TensorFlow/Keras using from tensorflow.keras.optimizers import RMSprop.

b. Specify RMSprop Optimizer with Learning Rate:

i. You create an optimizer object by specifying the RMSprop optimizer with a learning rate of 0.001. The learning rate controls the step size used during optimization, affecting the speed and quality of model training.

c. Compile the Model:

Finally, you compile the model using the model.compile method. The parameters you specify are as follows:

optimizer=optimizer: Here, you pass the RMSprop optimizer with the specified learning rate as the optimizer for training your model.

The loss option 'sparse_categorical_crossentropy' specifies the loss function to be utilized throughout the training process. The 'sparse categorical cross-entropy' is commonly employed in multi-class classification problems where the target values are represented by integers, specifically class indices.

The metric 'accuracy' is specified as the evaluation metric for the model. The metric quantifies the proportion of accurately categorized samples within the training set.

Upon configuring these parameters, your model is now assembled and prepared for the training process. To train your model on your dataset, you can utilise the model.fit method. This involves specifying the training data, labels, batch size, number of epochs, and validation data if needed. The provided optimizer, in conjunction with the loss and accuracy metric, will direct the training process.

```
[ ] from tensorflow.keras.optimizers import RMSprop
    # Compile the model with Adam optimizer and hyperparameters
    optimizer = tf.keras.optimizers.RMSprop(learning_rate=0.001)
    model.compile(optimizer=optimizer, loss = 'sparse_categorical_crossentropy', metrics = ['accuracy'])
```

Figure 4.7: Tensor Flow Implementation.

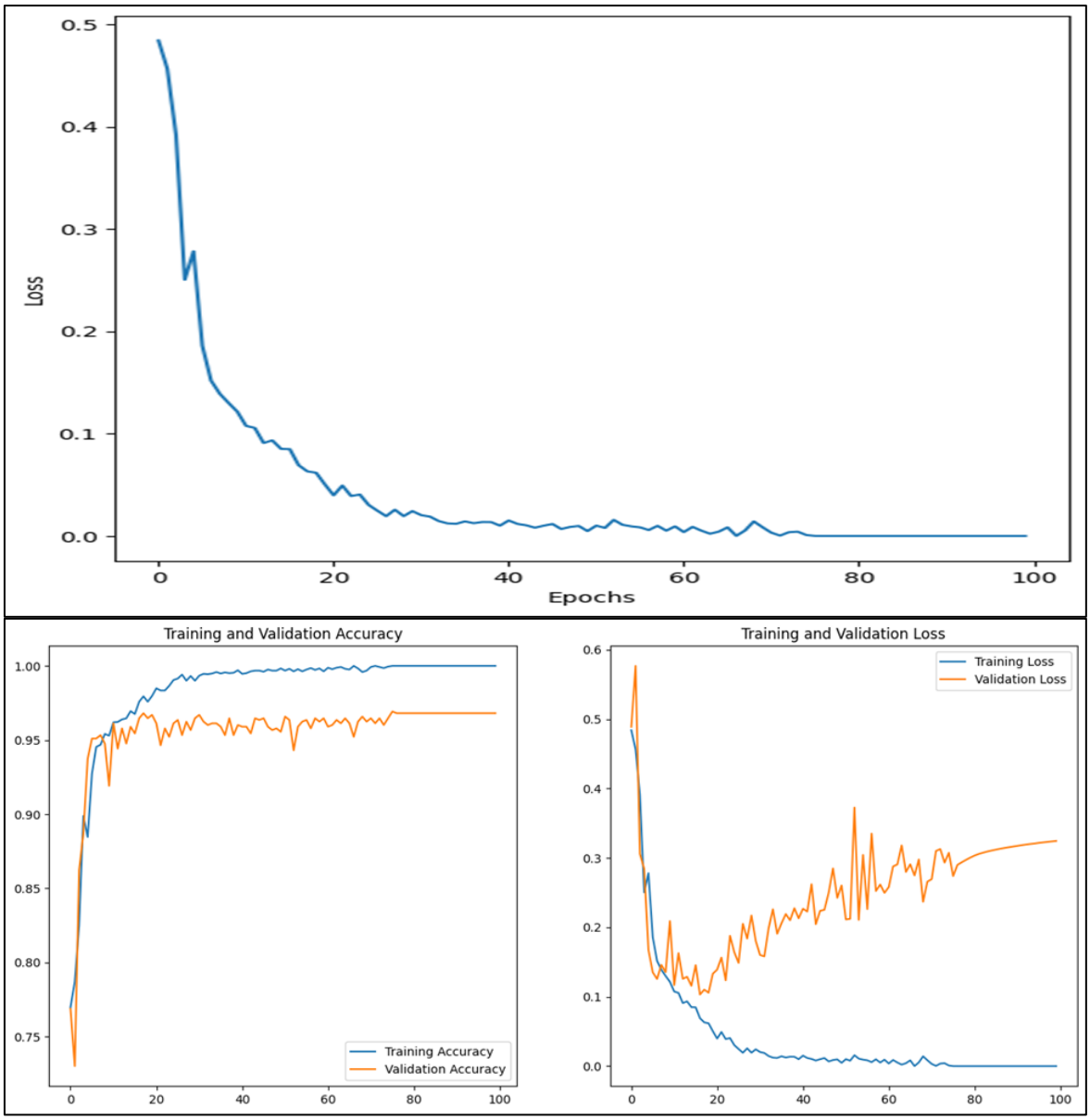

Figure 4.8: LSTM and CNN Implementation Code.

The findings of the study indicate that the CNN LSTM model showed a level of accuracy when it comes to tracking loss progression for activations, throughout 100 epochs. Table 5.2 presents a comparison of the model's accuracy, with research.

Table 4.2: LSTM and CNN Performence.

Best Result	Validation Loss	Validation Accuracy	Accuracy	Loss	Time
Epoch 11/100 1 29/129	0.1168	0.9613	0.9620	0.1075	2s 168ms/step

The classification report yields metrics for each class in the target data, such, as precision, recall and F1 score. These indicators are valuable, for evaluating the effectiveness of the classification model. Here is an example of how we utilize these metrics in the categorization report;

Firstly, let's examine the outcomes when we applied the CNN LSTM algorithm;

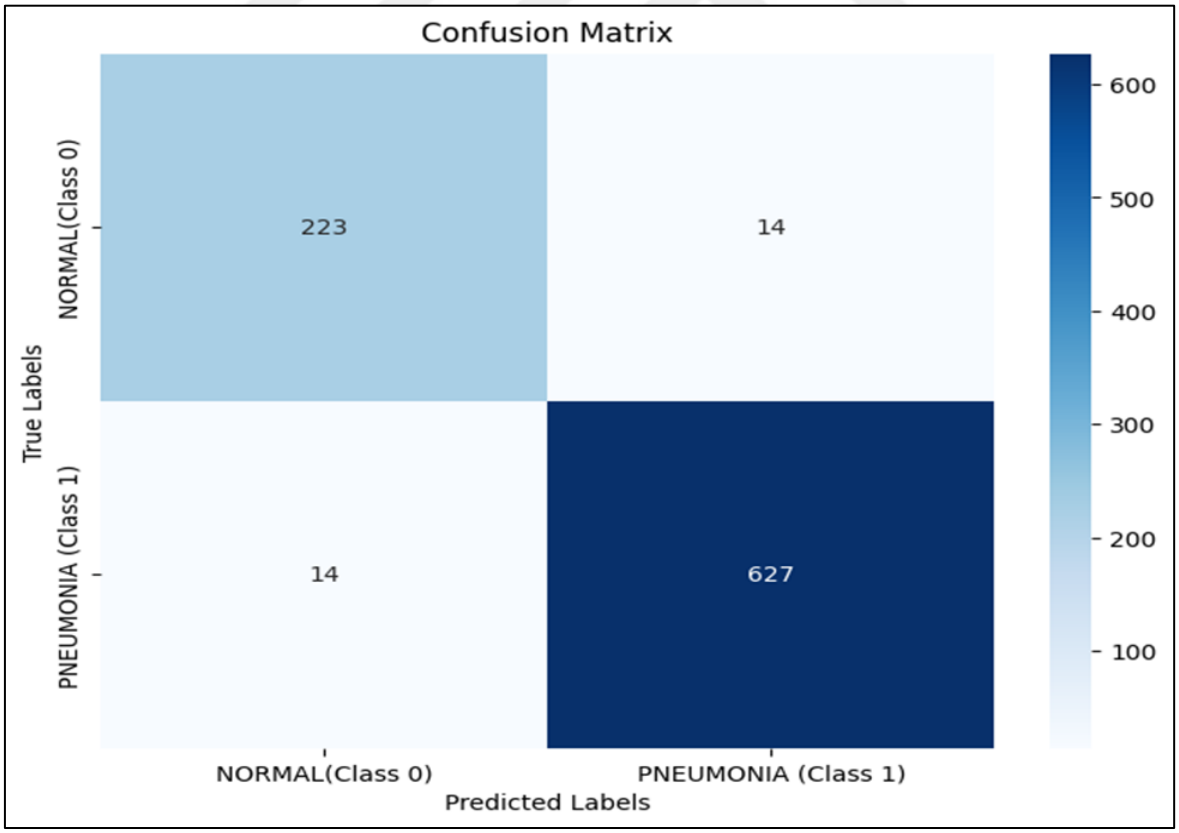

Figure 4.9: CM of CNN and LSTM Algorithm Implementation.

```
[ ] predictions = model.predict(x_val)
    pred=np.argmax(predictions,axis=1)
    pred = pred.reshape(1,-1)[0]
    print(classification_report(y_val,pred, target_names = ['NORMAL(Class 0)','PNEUMONIA (Class 1)']))

    28/28 [==============================] - 2s 40ms/step
                        precision    recall  f1-score   support

        NORMAL(Class 0)       0.94      0.94      0.94       237
    PNEUMONIA (Class 1)       0.98      0.98      0.98       641

                  accuracy                           0.97       878
                 macro avg       0.96      0.96      0.96       878
              weighted avg       0.97      0.97      0.97       878
```

Figure 4.10: CM of CNN and LSTM Algorithm Implementation Code.

4.3.3 CNN-Extra Trees

The provided code includes the definition of a Keras Sequential model, for image classification. It then proceeds to train a scikit learn Extra Trees Classifier using features extracted from the model. Lets go through the code step by step;

Keras Sequential Model: In Keras, a model is constructed as a linear sequence of layers. The model comprises the subsequent layers:

Conv2D layers are utilised for the purpose of extracting features from images through convolutional operations. The model consists of three Conv2D layers with ascending filter sizes (32, 64, 128) and ReLU activation functions. The initial layer additionally specifies that the input shape is (150, 150, 3), indicating images with size of 150x150 pixels and three colour channels (red, green, and blue).

MaxPooling2D layers are utilised to conduct max pooling, which effectively decreases the dimensions of the feature maps.

The flatten layer transforms the 2D feature maps into a 1D vector, so preparing them for subsequent layers.

The model consists of three Dense layers, each with varying numbers of units and ReLU activation functions. The last Dense layer consists of 700 units.

Dropout layer:

After the layer a Dropout layer is added with a rate of 0.5.

Dropout is a technique used to prevent overfitting by randomly deactivating a portion of input units during training.

Extra Trees Classifier:

You import the ExtraTreesClassifier, from scikit learn which is a method commonly used for classification tasks.

Training the Extra Trees Classifier:

To train the Extra Trees Classifier start by creating an instance of it called "et_classifier".

Based on the code comments it seems that you have already extracted features from a Keras model using model.predict. These features are stored in "features_train".

Now you can train the et_classifier using the extracted features (features_train) along with their labels (y_train).

The workflow here involves extracting features from your Keras model and then training a classifier (Extra Trees) on these extracted features. The actual training process is carried out using scikit learn library, which's highly useful for machine learning tasks including classification.

It's important to note that in this code snippet we are not training or making predictions, with the Keras model itself. Instead we are utilizing the extracted features from this model as input to train our Extra Trees Classifier.

```
[ ]  model = tf.keras.Sequential([
        Conv2D(32, (3, 3), activation='relu', input_shape=(150, 150, 3)),
        MaxPooling2D((2, 2)),
        Conv2D(64, (3, 3), activation='relu'),
        MaxPooling2D((2, 2)),
        Conv2D(128, (3, 3), activation='relu'),
        MaxPooling2D((2, 2)),
        Conv2D(128, (3, 3), activation='relu'),
        MaxPooling2D((2, 2)),
        Flatten(),
        Dense(512, activation='relu'),
        Dropout(0.5),  # Dropout layer to prevent overfitting
        Dense(256, activation='relu'),
        Dense(700, activation='relu')
    ])
```

Figure 4.11: Implementation Code of ET Activation Function.

```
[ ]  from sklearn.ensemble import ExtraTreesClassifier

     et_classifier = ExtraTreesClassifier()

     # Train the Extra Trees model on your data
     et_classifier.fit(features_train, y_train)
```

```
▾ ExtraTreesClassifier
ExtraTreesClassifier()
```

Figure 4.12: Implementation Code of ET.

The classification report presents metrics, for each class in the target data. These metrics, such as precision, recall and F1 score help analyze the effectiveness of the classification model. Here's how we utilize these indicators, in the categorization report. Lets start by examining the results obtained when applying the CNN ExtraTrees algorithm.

```
[ ]  from sklearn.metrics import classification_report
     # Display the classification report
     labels = ['NORMAL ','PNEUMONIA ']  # Replace with actual class labels
     print(classification_report(y_val, predictions, target_names=labels))
```

```
              precision    recall  f1-score   support

      NORMAL       0.91      0.85      0.88       237
   PNEUMONIA       0.95      0.97      0.96       641

    accuracy                           0.94       878
   macro avg       0.93      0.91      0.92       878
weighted avg       0.94      0.94      0.94       878
```

Figure 4.13: Implementation Code of ET and it's Performed.

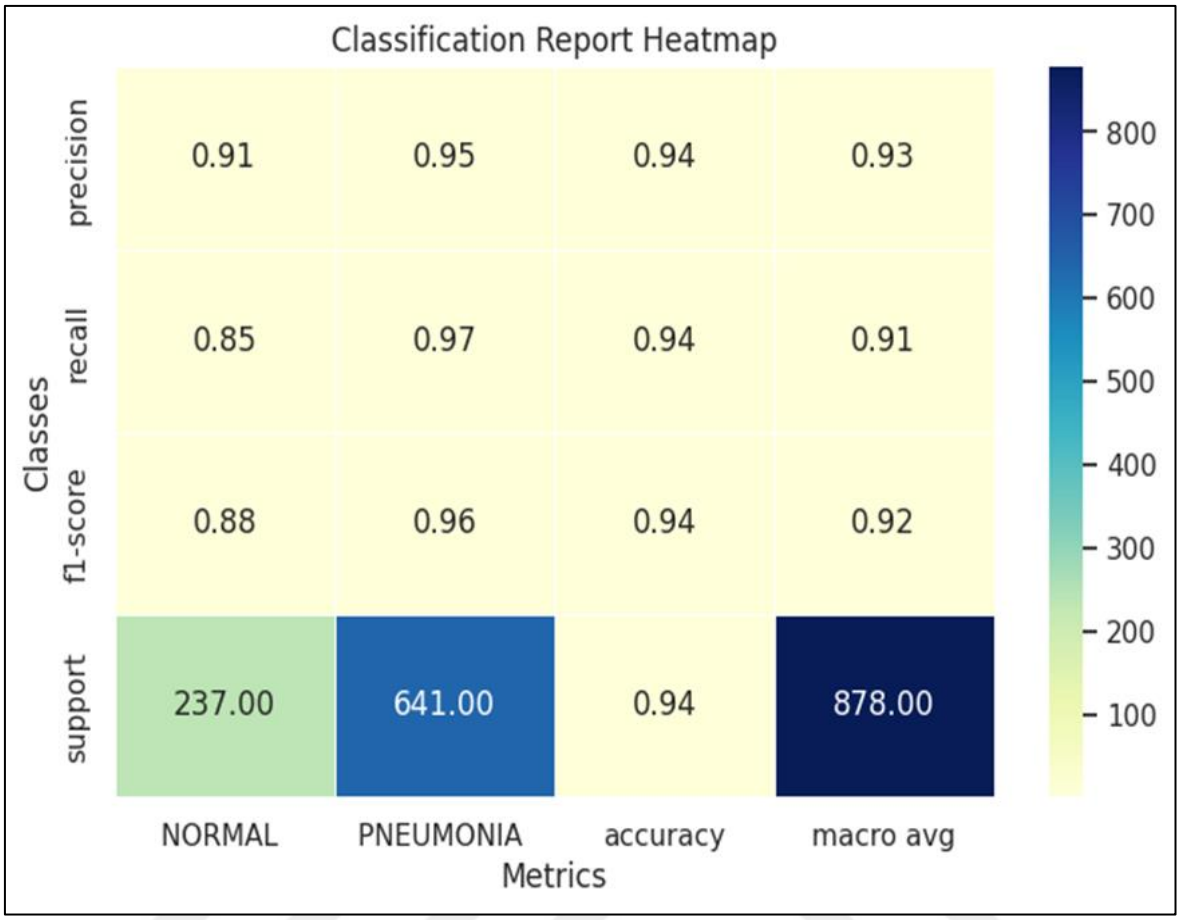

Figure 4.14: Metric implemention of ET.

4.4 DISCUSSION

Based on the table provided it is evident that the CNN model exhibited accuracy compared to models, including the CNN LSTM model during evaluation, with both test and validation datasets. This superiority was maintained across all models. The validation accuracy is depicted in Table 5.3.

Table 4.3: Our Models Implementation.

model	Accuracy with validation data
CNN-LSTM	0.97
ResNet-50 - CNN	0.98
CNN-ExtraTrees	0.94

4.5 COMPARISON OF ACCURACY WITH PREVIOUS WORK

Based on the table it's evident that both the CNN model and the CNN LSTM model outperform studies, in terms of accuracy. Both models achieved accuracy rates compared to research.

Table 4.4: Presents a Comparison of Accuracy, with Work.

Works	Classifier	Accuracy
Qiu et al. (2019)	CNN and RNN	92%
Guendel et al. (2020)	CNN	89.7%
Wang et al. (2018)	CNN	92.2%
This study	CNN-LSTM ResNet-50 -CNN CNN- ExtraTrees	97% 98% 94%

In our research, on this topic we found that by adding layers to the algorithms and adjusting the parameters in the optimizers along, with increasing the size of the dataset we were able to improve the accuracy rate of our model. This approach helped in training the algorithm.

5. CONCLUSIONS

This research employed five approaches to predict and manage chest diseases in children specifically focusing on patients diagnosed with pneumonia. The models were trained using data in its form. The images used were divided into three groups; 70%, for training 15% for testing and another 15% for validation. To ensure consistency the images were resized to a size of 150 pixels in both height and width. Subsequently various models were developed.

One of the models the CNN model, exhibited promising outcomes when the RMSprop optimizer was applied along with parameter tuning. Similarly encouraging results were achieved when combining layers in the CNN LSTM model and optimizing parameters with RMSprop. Additional models like ResNet 50 CNN and CNN Extra Trees also demonstrated findings. A comprehensive evaluation of these results was conducted alongside a classification report. Furthermore, a detailed comparative analysis was carried out to establish connections between this study's outcomes and previous research findings. This comparative study highlighted the performance of our developed models.

The resulting model emphasizes its proficiency in classifying chest images of patients offering a cost-effective solution, for medical practitioners.

I highly recommend that both researchers and medical device manufacturers invest in incorporating intelligence into the prediction process, for disease. It would be valuable to explore the models we have proposed and incorporate insights from researchers who have used approaches. The encouraging outcomes of our study demonstrate the effectiveness and potential of these models. In settings they hold promise for improving healthcare outcomes for pediatric chest disorders like pneumonia. This model serves as a tool for professionals empowering them to enhance their diagnostic capabilities and streamline decision making. Its accuracy and reliability make it an essential asset in treating chest diseases. By leveraging these models' healthcare providers can improve their skills leading to accurate and timely treatments for chest conditions, in children.

REFERENCES

[1] Rehman MU, Shafque A, Khan KH et al (2022) Novel privacy preserving non-invasive sensingbased diagnoses of pneumonia disease leveraging deep network model. Sensors 22. https://doi.org/ 10.3390/s22020461

[2] Pound, M.W., Drew, R.H. and Perfect, J.R., 2002. Recent advances in the epidemiology, prevention, diagnosis, and treatment of fungal pneumonia. Current opinion in infectious diseases, 15(2), pp.183-194.

[3] Obaro, S.K. and Madhi, S.A., 2006. Bacterial pneumonia vaccines and childhood pneumonia: are we winning, refining, or redefining?. The Lancet infectious diseases, 6(3), pp.150-161

[4] https://www.who.int/news-room/fact-sheets/detail/pneumonia

[5] Torres, A., Cilloniz, C., Niederman, M.S., Menendez, R., Chalmers, J.D., Wunderink, R.G. and van der Poll, T., 2021. Pneumonia. Nature Reviews Disease Primers, 7(1), pp.1-28.

[6] Santana, C.F., Alexandre, R.F. and Squiassi, H.B., 2018. Economic Impact of Absenteeism and Deaths Due to Pneumonia in Brazilian Economy. Value inHealth, 21, pp.S233-S234.

[7] Goveas, J.S. and Shear, M.K., 2020. Grief and the COVID-19 pandemic in older adults. The American Journal of Geriatric Psychiatry, 28(10), pp.1119-1125.

[8] irazitdinov I, Kholiavchenko M, Mustafaev T et al (2019) Deep neural network ensemble for pneumonia localization from a large-scale chest x-ray database. Comput Electr Eng 78:388–399. https://doi.org/10.1016/j.compeleceng.2019.08.004

[9] Ibrahim, A.U., Ozsoz, M., Serte, S., Al-Turjman, F. and Yakoi, P.S., 2021. Pneumonia classification using deep learning from chest X-ray images during COVID-
19. Cognitive Computation, pp.1-13.

[10] World Health Organization, 2014. Revised WHO classification and treatment of childhood pneumonia at health facilities: evidence summaries. Geneva: World Health Organization.

[11] Fedson, D.S., 2016. Treating the host response to emerging virus diseases: lessons learned from sepsis, pneumonia, influenza and Ebola. Annals of translational medicine, 4(21).

[12] Elshennawy, N.M. and Ibrahim, D.M., 2020. Deep-pneumonia framework using deep learning models based on chest X-ray images. Diagnostics, 10(9), p.649.

[13] Sharma, S. and Guleria, K., 2023. A systematic literature review on deep learning approaches for pneumonia detection using chest X-ray images. Multimedia Tools and Applications, pp.1-51.

[14] Sharifani, K. and Amini, M., 2023. Machine Learning and Deep Learning: A Review of Methods and Applications. World Information Technology and Engineering Journal, 10(07), pp.3897-3904.

[15] Celik, G., 2023. Detection of Covid-19 and other pneumonia cases from CT and X-ray chest images using deep learning based on feature reuse residual block and depthwise dilated convolutions neural network. Applied Soft Computing, 133, p.109906.

[16] Masad, I.S.; Alqudah, A.; Alqudah, A.M.; Almashaqbeh, S. A hybrid deep learning approach towards building an intelligent system for pneumonia detection in chest X-ray images. Int. J. Electr. Comput. Eng. (2088-8708) 2021, 11, 5530–5540. [Google Scholar] [CrossRef]

[17] Ayan, E.; Ünver, H.M. Diagnosis of pneumonia from chest X-ray images using deep learning. In Proceedings of the 2019 Scientific Meeting on Electrical-Electronics & Biomedical Engineering and Computer Science (EBBT), Istanbul, Turkey, 24–26 April 2019; pp. 1–5. [Google Scholar] [CrossRef]

[18] Chouhan, V.; Singh, S.K.; Khamparia, A.; Gupta, D.; Tiwari, P.; Moreira, C.; Damaševičius, R.; De Albuquerque, V.H.C. A novel transfer learning based approach for pneumonia detection in chest X-ray images. Appl. Sci. 2020, 10, 559. [Google Scholar] [CrossRef][Green Version]

[19] Luján-García, J.E.; Yáñez-Márquez, C.; Villuendas-Rey, Y.; Camacho-Nieto, O. A transfer learning method for pneumonia classification and visualization. Appl. Sci. 2020, 10, 2908. [Google Scholar] [CrossRef][Green Version]

[20] Elshennawy, N.M.; Ibrahim, D.M. Deep-pneumonia framework using deep learning models based on chest x-ray images. Diagnostics 2020, 10, 649. [Google Scholar] [CrossRef]

[21] Al Mamlook, R.E.; Chen, S.; Bzizi, H.F. Investigation of the performance of Machine Learning Classifiers for Pneumonia Detection in Chest X-ray Images. In Proceedings

of the 2020 IEEE International Conference on Electro Information Technology (EIT), Chicago, IL, USA, 31 July–1 August 2020. [Google Scholar] [CrossRef]

[22] Yee, S.L.K.; Raymond, W.J.K. Pneumonia Diagnosis Using Chest X-ray Images and Machine Learning. In Proceedings of the 2020 10th International Conference on Biomedical Engineering and Technology, Tokyo, Japan, 15–18 September 2020. [Google Scholar] [CrossRef]

[23] Toğaçar, M.; Ergen, B.; Cömert, Z. A deep feature learning model for pneumonia detection applying a combination of mRMR feature selection and machine learning models. Irbm 2020, 41, 212–222. [Google Scholar] [CrossRef]

[24] Rajaraman, S.; Candemir, S.; Kim, I.; Thoma, G.; Antani, S. Visualization and interpretation of convolutional neural network predictions in detecting pneumonia in pediatric chest radiographs. Appl. Sci. 2018, 8, 1715. [Google Scholar] [CrossRef][Green Version]

[25] Rahman, T.; Chowdhury, M.E.; Khandakar, A.; Islam, K.R.; Islam, K.F.; Mahbub, Z.B.; Kadir, M.A.; Kashem, S. Transfer learning with deep convolutional neural network (CNN) for pneumonia detection using chest X-ray. Appl. Sci. 2020, 10, 3233. [Google Scholar] [CrossRef]

[26] Polat, Ö.; Dokur, Z.; Ölmez, T. Determination of Pneumonia in X-ray Chest Images by Using Convolutional Neural Network. Turk. J. Electr. Eng. Comput. Sci. 2021, 29, 1615–1627. [Google Scholar] [CrossRef]

[27] Alqudah, A.M.; Qazan, S.; Masad, I.S. Artificial Intelligence Framework for Efficient Detection and Classification of Pneumonia Using Chest Radiography Images. J. Med. Biol. Eng. 2021, 41, 599–609. [Google Scholar]

[28] Madhubala, B.; Sarathambekai, S.; Vairam, T.; Sathya Seelan, K.; Sri Sathya, R.; Swathy, A.R. Pre-Trained Convolutional Neural Network Model Based Pneumonia Classification from Chest X-ray Images. 2021. Available online: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3852043 (accessed on 6 August 2021).

[29] Agrawal, H. Pneumonia Detection Using Image Processing and Deep Learning. In Proceedings of the 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS), Coimbatore, India, 25–27 March 2021; pp. 67–73. [Google Scholar]

[30] Alquran, H.; Alsleti, M.; Alsharif, R.; Qasmieh, I.A.; Alqudah, A.M.; Harun, N.H.B. Employing Texture Features of Chest X-Ray Images and Machine Learning in COVID-19 Detection and Classification. Mendel 2021, 27, 9–17. [Google Scholar] [CrossRef]

[31] Rajasenbagam, T.; Jeyanthi, S.; Pandian, J.A. Detection of pneumonia infection in lungs from chest X-ray images using deep convolutional neural network and content-based image retrieval techniques. J. Ambient Intell. Humaniz. Comput. 2021, 1–8. [Google Scholar] [CrossRef]

[32] Shah A, Shah M (2022) Advancement of deep learning in pneumonia/Covid-19 classifcation and localization: A systematic review with qualitative and quantitative analysis. Chronic Dis Transl Med 8:154–171. https://doi.org/10.1002/cdt3.17

[33] Kareem A, Liu H, Sant P (2022) Review on Pneumonia Image Detection: A Machine Learning Approach. Human-Centric Intelligent Systems 2:31–43. https://doi.org/10.1007/s44230-022-00002-2

[34] Padash S, Mohebbian MR, Adams SJ et al (2022) Pediatric chest radiograph interpretation: how far has artifcial intelligence come? A systematic literature review. Pediatr Radiol 52:1568–1580. https://doi.org/10.1007/s00247-022-05368-w

[35] Ghaffar Nia, N., Kaplanoglu, E., & Nasab, A. (2023). Evaluation of artificial intelligence techniques in disease diagnosis and prediction. Discover Artificial Intelligence, 3(1), 5.

[36] Kaur, S., Singla, J., Nkenyereye, L., Jha, S., Prashar, D., Joshi, G. P., ... & Islam, S. R. (2020). Medical diagnostic systems using artificial intelligence (ai) algorithms: Principles and perspectives. IEEE Access, 8, 228049-228069.

[37] Mitra, S., & Shankar, B. U. (2015). Medical image analysis for cancer management in natural computing framework. Information Sciences, 306, 111-131.

[38] Paudyal, R., Shah, A. D., Akin, O., Do, R. K., Konar, A. S., Hatzoglou, V., ... & Shukla-Dave, A. (2023). Artificial Intelligence in CT and MR Imaging for Oncological Applications. Cancers, 15(9), 2573.

[39] Ahmed, Z., Mohamed, K., Zeeshan, S., & Dong, X. (2020). Artificial intelligence with multi-functional machine learning platform development for better healthcare and precision medicine. Database, 2020, baaa010.

[40] Agarwal, S., Yadav, A. S., Dinesh, V., Vatsav, K. S. S., Prakash, K. S. S., & Jaiswal, S. (2023). By artificial intelligence algorithms and machine learning models to diagnosis cancer. Materials Today: Proceedings, 80, 2969-2975.

[41] Myszczynska, M. A., Ojamies, P. N., Lacoste, A. M., Neil, D., Saffari, A., Mead, R., ... & Ferraiuolo, L. (2020). Applications of machine learning to diagnosis and treatment of neurodegenerative diseases. Nature Reviews Neurology, 16(8), 440-456.

[42] Zhou, S. K., Greenspan, H., & Shen, D. (Eds.). (2017). Deep learning for medical image analysis. Academic Press.

[43] Keleb, A., Sisay, T., Alemu, K., Ademas, A., Lingerew, M., Kloos, H., ... & Adane, M. (2020). Pneumonia remains a leading public health problem among under-five children in peri-urban areas of north-eastern Ethiopia. PLoS One, 15(9), e0235818.

[44] Zhu, Z. (2020). Detecting Coronavirus Disease 2019 Pneumonia in Chest X-Ray Images Using Deep Learning. University of California, Los Angeles.

[45] Ikeuchi, K. (Ed.). (2021). Computer vision: A reference guide. Cham: Springer International Publishing.

[46] Mall, P. K., Singh, P. K., Srivastav, S., Narayan, V., Paprzycki, M., Jaworska, T., & Ganzha, M. (2023). A comprehensive review of deep neural networks for medical image processing: Recent developments and future opportunities. Healthcare Analytics, 100216.

[47] Kumar, A., Gandhi, C. P., Zhou, Y., Kumar, R., & Xiang, J. (2020). Improved deep convolution neural network (CNN) for the identification of defects in the centrifugal pump using acoustic images. Applied Acoustics, 167, 107399.

[48] Zaniolo, L., & Marques, O. (2020). On the use of variable stride in convolutional neural networks. Multimedia Tools and Applications, 79(19-20), 13581-13598.

[49] Rocha, A. P. (2021). Roll Padding and WaveNet for Multivariate Time Series in Human Activity Recognition. Trends and Applications in Information Systems and Technologies: Volume 1, 1365, 238.

[50] Zheng, Q., Yang, M., Tian, X., Wang, X., & Wang, D. (2020). Rethinking the Role of Activation Functions in Deep Convolutional Neural Networks for Image Classification. Engineering Letters, 28(1).

[51] Plusch, G., Arsenyev-Obraztsov, S., & Kochueva, O. (2023). The Weights Reset Technique for Deep Neural Networks Implicit Regularization. Computation, 11(8), 148.

[52] Haji, S. H., & Abdulazeez, A. M. (2021). Comparison of optimization techniques based on gradient descent algorithm: A review. PalArch's Journal of Archaeology of Egypt/Egyptology, 18(4), 2715-2743.

[53] Sharma, H., Jain, J. S., Bansal, P., & Gupta, S. (2020, January). Feature extraction and classification of chest x-ray images using cnn to detect pneumonia. In 2020 10th international conference on cloud computing, data science & engineering (Confluence) (pp. 227-231). IEEE.

[54] Dai, Y., Gieseke, F., Oehmcke, S., Wu, Y., & Barnard, K. (2021). Attentional feature fusion. In Proceedings of the IEEE/CVF winter conference on applications of computer vision (pp. 3560-3569).

[55] Singh, U., Rizwan, M., Alaraj, M., & Alsaidan, I. (2021). A machine learning-based gradient boosting regression approach for wind power production forecasting: A step towards smart grid environments. Energies, 14(16), 5196.

[56] Loy, J. (2019). Neural Network Projects with Python: The ultimate guide to using Python to explore the true power of neural networks through six projects. Packt Publishing Ltd.

[57] Sahu, R., Dash, S. R., Cacha, L. A., Poznanski, R. R., & Parida, S. (2020). Epileptic seizure detection: a comparative study between deep and traditional machine learning techniques. Journal of integrative neuroscience, 19(1), 1-9.

[58] Zha, W., Liu, Y., Wan, Y., Luo, R., Li, D., Yang, S., & Xu, Y. (2022). Forecasting monthly gas field production based on the CNN-LSTM model. Energy, 124889.

[59] Yousefi, M., & Hansen, J. H. (2020). Block-based high performance CNN architectures for frame-level overlapping speech detection. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 29, 28-40.

[60] Yin, H., Zhang, X., Wang, F., Zhang, Y., Xia, R., & Jin, J. (2021). Rainfall-runoff modeling using LSTM-based multi-state-vector sequence-to-sequence model. Journal of Hydrology, 598, 126378.

[61] Yacouby, R., & Axman, D. (2020, November). Probabilistic extension of precision, recall, and f1 score for more thorough evaluation of classification models. In Proceedings of the first workshop on evaluation and comparison of NLP systems (pp. 79-91).

[62] Udayakumar, K., & Subiramaniyam, N. P. (2020). Deep learning-based production assists water quality warning system for reverse osmosis plants. H2Open Journal, 3(1), 538-553.

[63] Chen, Y., Wang, L., Liu, X., & Wang, H. (2023). Artificial Intelligence-Empowered Art Education: A Cycle-Consistency Network-Based Model for Creating the Fusion Works of Tibetan Painting Styles. Sustainability, 15(8), 6692.

[64] Mukti, I. Z., & Biswas, D. (2019, December). Transfer learning based plant diseases detection using ResNet50. In 2019 4th International conference on electrical information and communication technology (EICT) (pp. 1-6). IEEE.

[65] Kongmanee, J., & Thanapattheerakul, T. (2020, November). Fine-tuning a lightweight convolutional neural networks for COVID-19 diagnosis. In CSBio'20: Proceedings of the Eleventh International Conference on Computational Systems-Biology and Bioinformatics (pp. 101-103).

[66] Ansari, N., Faizabadi, A., Motakabber, S., & Ibrahimy, M. (2020). Effective Pneumonia Detection using Res Net based Transfer Learning. Test Engineering and Management, 82, 15146-15153.

[67] Mandal, B., Okeukwu, A., & Theis, Y. (2021). Masked face recognition using resnet-50. arXiv preprint arXiv:2104.08997.

[68] Amelio, A., Bonifazi, G., Cauteruccio, F., Corradini, E., Marchetti, M., Ursino, D., & Virgili, L. (2023). Representation and compression of Residual Neural Networks through a multilayer network based approach. Expert Systems with Applications, 215, 119391.

[69] Elavarasi, S. A., & Kannan Venkatesan, M. V. (2022). Classification Model Based on Pathological Data for Kidney Diseases Prediction using Machine Learning Approach. Journal of Algebraic Statistics, 13(1), 169-177.

[70] Desai, C. (2020). Comparative analysis of optimizers in deep neural networks. International Journal of Innovative Science and Research Technology, 5(10), 959-962.