بسم الله الرحمن الرحيم

**Sudan University of Science and Technology**
**College of Graduate Studies**

# Detecting Pulmonary Tuberculosis in Chest X-Ray Images Using Convolutional Neural Network

## الكشف عن السل الرئوي في صور الأشعة السينية للصدر بإستخدام الشبكات العصبية الإلتفافية

**A Thesis Submitted in partial fulfillment of the requirement for the M.Sc. Degree in Biomedical Engineering**

By:
- Aden Hassan Mergani Elsanosi.

Supervisor:
-Dr. Eltahir Mohammed Hussein.

February 2021

# الأيــة

**بسم الله الرحمن الرحيم**

(فَتَعَالَى اللَّهُ الْمَلِكُ الْحَقُّ ۗ وَلَا تَعْجَلْ بِالْقُرْآنِ مِن قَبْلِ أَن يُقْضَىٰ إِلَيْكَ وَحْيُهُ ۖ وَقُل رَّبِّ زِدْنِي عِلْمًا)

سورة طه- الأية (114)

صدق الله العظيم

# *DEDICATION*

*This dissertation is dedicated to my loving parents* ***Hassan*** *and* ***Ibtisam Elsanosi*** *for holding my hand since day one in life. Without their tender, guiding, and sometimes over protecting touch I would have never become who I am today. I will always appreciate all they have done, especially my mother for her great efforts to make me always on the top.*

*To my sisters and brothers* ***Wala, Khalid****,* ***Margani****,* ***Tamani****, and* ***Ahmed*** *have never left my side and are very special. Thank you for your everlasting love and warm encouragement throughout my research.*

*To my friends, who know me, but love me anyway. Without you I can never make do in this world.*

*To my fiancé* ***Anas Abdallah*** *who has been a constant source of support and encouragement.*

*Last but not least I am dedicating this to my cousin* ***Saba Abdulkarim*** *gone forever away from our loving eyes. May Allah grant you Jannah Firdaws.*

*Amen.*

# ACKNOWLEDGMENTS

First and foremost, I would like to thank Allah for giving me the strength and guidance to accomplish this thesis. Without His guidance, I will certainly not be able to complete this task successfully.

Would like to thank my thesis supervisor **Dr.Eltahir Mohamed** The door to his office was always open whenever I ran into a trouble spot or had a question about my research or writing, he steered me in the right direction whenever he thought I needed it. Many Thanks!

Finally, I must express my very profound gratitude to my parents and to my [sisters and brothers] for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

# TABLE of CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# ABSTRACT

The main objective of this study is to detect tuberculosis in chest x-ray images using a convolutional neural network. Tuberculosis (TB) is an infectious disease that generally attacks the lungs and causes death for millions of people annually. Chest radiography and deep-learning-based image segmentation techniques can be utilized for TB diagnostics. Convolutional Neural Networks (CNNs) have shown advantages in medical image recognition applications as powerful models to extract informative features from images. The application of CNNs for image classification has significantly increased prediction accuracy rates. Several convolutional neural networks (CNNs) such as VGG work by building a pre-trained model that is easy to set up with minimal preprocessing. It uses libraries with weights containing millions of images to train the model before application on the actual data. This process is also called transfer learning. This thesis presents a ConvNet model that uses VGG16 for classifying CXR images, the ConvNet model is applied to the chest X-ray (CXR) dataset to identify if the patient has Tuberculosis (TB), applying such a model bypasses the requirement of building sophisticated segmentation algorithms which could be time-consuming, require professional expertise, and are mostly specialized making them inadmissible for application to other similar problems, the model can achieve accuracy of 92%. The accuracy obtained is comparable to previous work done on the dataset.

## المستخلص

الهدف الرئيسي لهذه الدراسة هو الكشف عن مرض السل في صور الأشعة السينية للصدر بإستخدام الشبكات العصبية الالتفافية ,السل مرض معد يهاجم الرئتين بشكل عام ويتسبب في وفاة الملايين من الأشخاص سنوياً. ويمكن استخدام التصوير الشعاعي للصدر وتقنيات تجزئة الصور المستندة إلى التعلم العميق لتشخيص السل. أظهرت الشبكات العصبية الالتفافية (CNNs) مزايا في تطبيقات التعرف على الصور الطبية كنماذج قوية لاستخراج ميزات غنية بالمعلومات من الصور. وقد أدى تطبيق هذه الأجهزة على تصنيف الصور إلى زيادة كبيرة في معدلات دقة التنبؤ. تعمل العديد من الشبكات العصبية الالتفافية (CNNs) مثل VGG عن طريق بناء نموذج مدرب مسبقًا يسهل إعداده بأقل معالجة مسبقة. و يستخدم مكتبات ذات أوزان تحتوي على الملايين من الصور لتدريب النموذج قبل تطبيقها على البيانات الفعلية. وتسمى هذه العملية أيضاً بنقل التعلم. تقدم هذه الأطروحة نموذج ConvNet الذي يستخدم VGG16 لتصنيف صور CXR ، يتم تطبيق نموذج ConvNet على مجموعة بيانات الأشعة السينية للصدر (CXR) لتحديد ما إذا كان المريض مصابًا بالسل (TB) ، إن تطبيق هذا النموذج يتجاوز شرط إنشاء خوارزميات تجزئة متطورة يمكن أن تستغرق وقتا طويلا، وتتطلب خبرة مهنية، وهي متخصصة في معظمها تجعلها غير مقبولة لتطبيقها على مشاكل أخرى مماثلة. يمكن للنموذج تحقيق دقة بنسبة 92٪. وتماثل الدقة التي تم الحصول عليها في الأعمال السابقة التي تم إنجازها على مجموعة البيانات.

# Chapter One

# Introduction

## 1.1 General View

Tuberculosis (TB) is a global issue that seriously endangers public health. TB is an airborne ailment discovered by Robert Koch in 1822, and it is caused by Mycobacterium Tuberculosis (MTB), the causative organism of TB [1]. Depending on the site which MTB affects TB is categorized into two main types Pulmonary Tuberculosis (PTB) and Extra Pulmonary Tuberculosis (EPTB) [2].Tuberculosis disease is more prevalent in developing regions and can affect both males and females but more prominent in males. In Sudan the average number of admitted patient in Omdurman teaching hospital (Abu-Anja) is 900 patients per month.

Pathology is one of the most important means for diagnosing TB in clinical practice, to confirm TB as the diagnosis, finding specially stained TB bacilli under a microscope is critical. Because of the very small size and number of bacilli, it is a time-consuming and strenuous work even for experienced pathologists, and this strenuosity often leads to low detection rate and false diagnoses, beside pathology There are different conventional methods for diagnosing tuberculosis having several shortcomings; the need to find newer prompt methods for disease detection has been aided by the latest Artificial Intelligence [AI] tools. Convolutional neural networks (CNN) are one of the important tools that is being used in diagnosis and evaluation of medical conditions. In this thesis convolutional neural networks are used to detect pulmonary tuberculosis.

## 1.2 Problem Statement

Un proper diagnosis of tuberculosis on time is a prominent problem in medical field. Tuberculosis is curable but needs to be detected early for necessary treatment to be effective.

## 1.3 Objective

The main objective of this study is to detect tuberculosis in chest x-ray images using convolutional neural network.

The specific objectives are to:

1. develop a convolution neural network (CNN) model.
2. classify chest x-ray image into normal and abnormal.
3. asses the accuracy, sensitivity and specificity of the model.

## 1.4 Methodology

The main idea of this research is the designing of CNN model to detect pulmonary tuberculosis, firstly images are resized, and then divided into training, validation and testing, then it flows through the model and finally the images are classified into normal and abnormal TB.

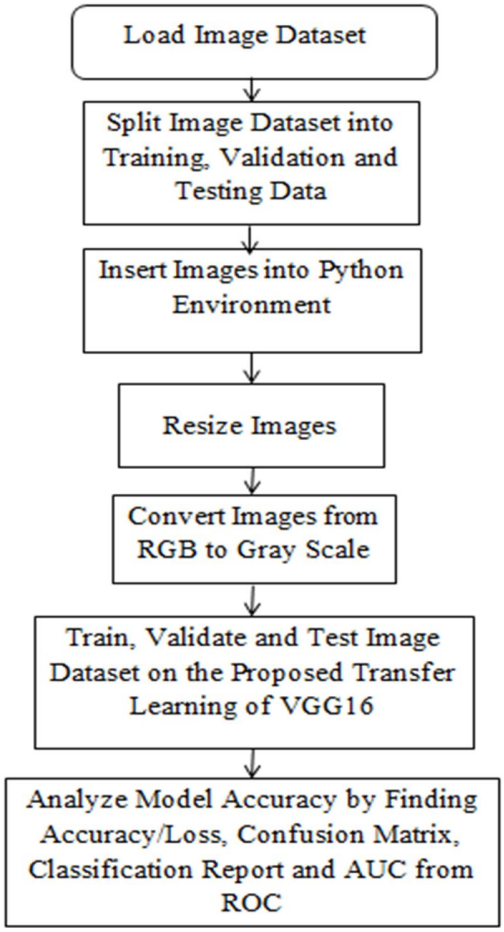


Figure 1.1: Block diagram of the proposed method

## 1.5 Thesis Layout

This thesis consists of six chapters, **Chapter one** is an introduction. The literature Reviews are presented in **chapter two**. Some theoretical background related to the

study was provided at **chapter three**, in **chapter four** the proposed model that was applied for classification of CXR images is discussed and described, **chapter five** introduce the results that were obtained from the applying of the proposed model, and **chapter six** provides the conclusions and recommendation of the thesis.

# Chapter Two

# Literature Reviews

**Hwang et al** [3] proposed for the first time the application of a CNN to tuberculosis detection the proposal includes the creation of a custom network, adapting an existing CNN to the specific problem of tuberculosis detection and recalculating the existing weights. The proposed architecture is a variant of the AlexNet network with a larger input layer (500x500 pixels), a new convolutional layer right after the input layer and a new max-pooling layer. The network was trained using a large private dataset consisting of approximately 10,000 images. In the case of the network trained with pre-learned weights, the accuracy reached 0.90 and the AUC reached 0.96. The trained model was also applied in the classification of the Montgomery and Shenzhen datasets where the obtained results are competitive in most cases. In the Montgomery dataset the accuracy was 0.674 and the AUC 0.884, whilst in the Shenzhen dataset the accuracy was 0.837 and the AUC was 0.926.

**Chang Liu et al** [4] proposed a novel method using Convolutional Neural Network (CNN) to deal with unbalanced, less-category X-ray images their method improved the accuracy for classifying multiple TB manifestations by a large margin. They explored the effectiveness and efficiency of shuffle sampling with cross-validation in training the network and found its outstanding effect in medical images classification. They achieved 85.68% classification accuracy in a large TB image dataset.

**Yan Xiong et al** [5] built a convolutional neural networks (CNN) model, named tuberculosis AI (TB-AI), specifically to recognize TB bacillus. The training set contains 45 samples, including 30 positive cases and 15 negative cases, where bacilli are labeled by human pathologists. Upon training the neural network model, 201 samples (108 positive cases and 93 negative cases) were collected as test set and used to examine TB-AI. They compared the diagnosis of TB-AI to the ground truth result provided by human pathologists, analyzed inconsistencies between AI and human, and adjusted the protocol accordingly. Trained TB-AI was run on the test data twice. Examined against the double confirmed diagnosis by pathologists

both via microscopes and digital slides, TB-AI achieved 97.94% sensitivity and 83.65% specificity.

In **Syeda Meraj et alv** [2] four simple Convolutional Neural Networks (CNN) models such as VGG-16, VGG-19, RestNet50, and GoogLenet were implemented in identification of TB manifested CXRs. Two public TB image datasets were utilized to conduct this research. This study was carried out to explore the limit of accuracies and AUCs acquired by simple and small-scale CNN with complex and large-scale CNN models. The results achieved from this work were compared with results of two previous studies. The results indicate that their proposed VGG-16 model has gained highest score overall compared to the models from other two previous studies.

**Pasa et al** [6] proposed a simple convolutional neural network optimized for the problem which is faster and more efficient than previous models but preserves their accuracy. The visualization capabilities of CNNs have not been fully investigated by testing saliency maps and gradient class activation maps (grad-CAM). These visualization techniques help us understand the network and may also be useful as an approximate visual diagnosis for presentation to radiologists.as tuberculosis visualization methods, and discussed them from a radiological perspective. Their model performed accuracy 79.0 % for Montgomery County (MC), 84.4% for Shenzhen (SH) and 86.2%for combined.

**Mustapha Oloko-Oba and Serestina Viriri** [7] proposed a Computer-Aided Detection model using Deep Convolutional Neural Networks to automatically detect TB from Montgomery County (MC) Tuberculosis radiographs. Their proposed model performed at 87.1% validation accuracy and evaluated using confusion matrix and accuracy as metrics.

**Ahmed T. Sahlol et al** [8] presented a novel hybrid method for efficient classification of chest X-ray images. First, the features are extracted from chest X-ray images using MobileNet, a CNN model, which was previously trained on the ImageNet dataset. Then, to determine which of these features are the most relevant, we apply the Artificial Ecosystem-based Optimization (AEO) algorithm as a feature selector. Their method is applied to two public benchmark datasets (Shenzhen and Dataset 2) and allows them to achieve high performance and

reduced computational time. It selected successfully only the best 25 and 19 (for Shenzhen and Dataset 2, respectively), while improving the classification accuracy (90.2% for Shenzen dataset and 94.1% for Dataset 2).

**Seelwan Sathitratanacheewin et al** [9] developed a Deep Convolutional Neural Network (DCNN) model using a Tuberculosis (TB)-specific chest x-ray (CXR) dataset of one population (National Library of Medicine Shenzhen No.3 Hospital) and tested it with non-TB-specific CXR dataset of another population (National Institute of Health Clinical Centers). In the training and intramural test sets using the Shenzhen hospital database, the DCCN model exhibited an AUC of 0.9845 and 0.8502 for detecting TB, respectively. However, the AUC of the supervised DCNN model in the ChestX-ray8 dataset was dramatically dropped to 0.7054. Using the cut points at 0.90, this suggested 72% sensitivity and 82% specificity in the Shenzhen dataset.

Table 2.1: Summary of literature reviews

| **Authors, Year** | **Dataset** | **Features / Parameter** | **Method** | **Classification Accuracy (%)** | **Area under the curve AUC** | **Sensitivity (%)** | **Specificity (%)** |
|---|---|---|---|---|---|---|---|
| Hwang et al,2016 | Private data , Shenzhen(SH) and Montgomery County (MC) | CXR | Transfer Learning of Modified Alexnet | 90<br>83.7<br>67.4 | 0.96<br>0.926<br>0.884 | - | - |
| Chang Liu et al,2017 | Private data set from Lima, Peru | CXR | CNN | 85.68 | - | - | - |
| Yan Xiong et al ,2018 | Data collected from the Department of Pathology, Peking University Hospital | tissue samples and treated by acid-fast stain | CNN | - | - | 97.94 | 83.65 |
| Syeda Meraj et al ,2019 | Shenzhen (SH and Montgomery County (MC) | CXR | VGG-16<br>VGG-19<br>ResNet50<br>GoogLenet | 86.74 ,77.14<br>84.33 , 77.14<br>81.92 , 71.42<br>80.72 , 71.42 | 92.0 , 75.0<br>91.0 , 90.0<br>91.2 ,76.0<br>88.0 ,75.0 | -<br>-<br>-<br>- | -<br>-<br>-<br>- |
| Pasa et al. 2019 | Montgomery County (MC), Shenzhen (SH) and Belarus Dataset | CXR | Optimized CNN | 79.0 for MC, 84.4 for SH and 86.2 for combined | 0.811 for MC, 0.9 for SH and 0.925 for combined | - | - |

| Authors, Year | Dataset | Features / Parameter | Method | Classification Accuracy (%) | Area under the curve AUC | Sensitivity (%) | Specificity (%) |
|---|---|---|---|---|---|---|---|
| Mustapha Oloko-Oba and Serestina Viriri,2020 | Montgomery County (MC) | CXR | DCNN | 87.1% | - | - | - |
| Ahmed T. Sahlol et al ,2020 | Shenzhen (SH), Dataset 2 | CXR | DCNN | 90.2%<br>94.1% | - | - | - |
| Seelwan Sathitratanacheewin et al ,2020 | Shenzhen Hospital Dataset and NIH ChestX-ray8 | CXR | DCNN | - | - | 72 | 82 |

Previous studies that have been carried out using CNN methodology for the detection of TB are discussed here; Table 2.1 shows several works of CNN on TB detection. Most of the authors have used Montgomery County (MC) and Shenzhen (SH) CXR sets since they are publicly available TB dataset, Based on Table 2.1, Syeda Meraj et al ,2019 have used both MC and SH datasets, executed on four CNN models such as VGG-16,VGG-19 ,ResNet50 and GoogLenet a greater accuracy values achieved when using Shenzhen dataset.

Private Datasets used by Hwang et al, 2016 and Chang Liu et al, 2017 achieved comparable accuracy results to other studies.

# Chapter Three

# Theoretical Background

## 3.1 Tuberculosis

Tuberculosis is an age-old disease. This ancient killer disease has survived for thousands of years. There are quite a few archaeological evidences, found in ancient Egypt, pointing to the existence of this disease. Such as pott's disease (spinal tuberculosis), found in Egyptian mummies [10], [11]. Furthermore, several literary documentations reveal the existence of TB, in ancient India, China and Greece [12], [13]. TB was known by various names, throughout the history. In ancient Greece, it was called "Phthisis" or "consumption" (because of extreme weight loss in patients). Tuberculosis (TB) is an airborne ailment discovered by Robert Koch in 1822, it is caused by Mycobacterium Tuberculosis (MTB), the causative organism of TB [1].

According to the (World Health Organization [WHO], 2020), About one-quarter of the world's population has a TB infection, which means people have been infected by TB bacteria but are not (yet) ill with the disease and cannot transmit it., indicating a risk of developing active TB during their lifetime. A total of 1.4 million people died from TB in 2019 (including 208000 people with HIV). Worldwide, TB is one of the top 10 causes of death and the leading cause from a single infectious agent (above HIV/AIDS).

In 2019, an estimated 10 million people fell ill with tuberculosis (TB) worldwide. 5.6 million Men, 3.2 million women and 1.2 million children. TB is present in all countries and age groups. (World Health Organization [WHO] , 2020). An estimated 60 million lives were saved through TB diagnosis and treatment between 2000 and 2019.

### 3.1.1 Types of TB

Broadly classified, there are two types of tuberculosis infections. **Active** and **latent** in active tuberculosis, the individual who is carrying the organism has active symptoms and can transmit the infection to other people.

In latent tuberculosis, the individual carries the bacteria but does not exhibit any symptoms whatsoever. This is because the immunity fights the infection and is

able to suppress it to an extent. Individuals suffering from latent tuberculosis cannot transmit the illness to others. However, at some point in their life, the bacteria can get reactivated and the infection can become active tuberculosis [14].

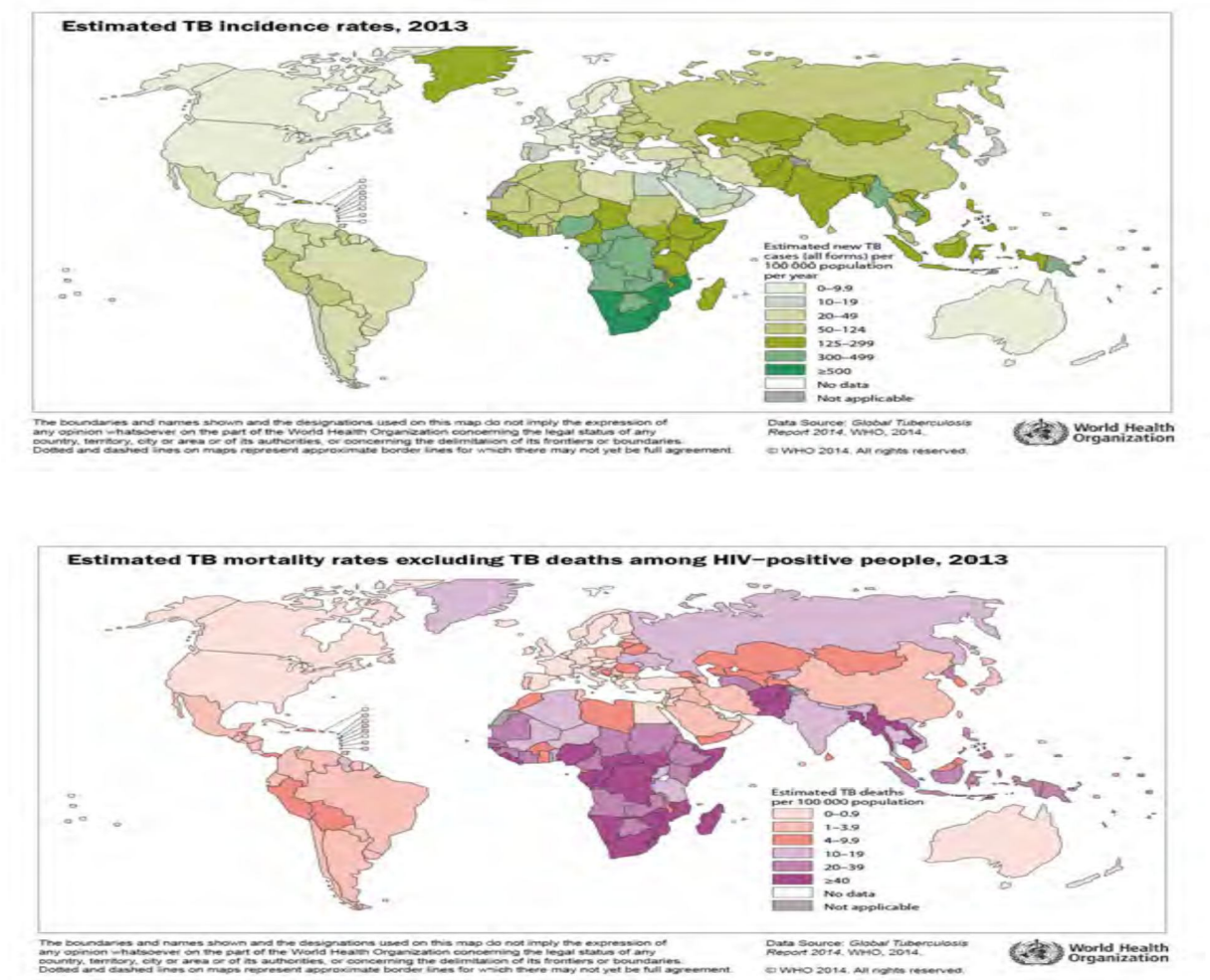


Figure 3.1: WHO maps (c. 2013) showing (Top) global TB incidence rates; (Bottom) TB mortality among HIV positive cases. [(WHO) Global Tuberculosis Report 2014]

If the bacteria affect lungs and its surrounding regions, then it is called as Pulmonary Tuberculosis (PTB).When the bacteria affects and spreads to other parts of the body then it is called Extra Pulmonary Tuberculosis (EPTB). Few types of PTB are primary Tuberculosis pneumonia, cavity Tuberculosis, Tuberculosis pleurisy, military TB and laryngeal TB [2]. On the other hand, types of EPTB include Pot's disease, lymph node disease, Tuberculosis meningitis,

adrenal Tuberculosis, Tuberculosis peritonitis, Tuberculosis pericarditis, Osteal tuberculosis [15, 16].

### 3.1.2 Risk Factors for Tuberculosis

Tuberculosis is usually spread from one person to another through droplet infection. This means that if an individual is carrying the tuberculosis bacteria in their lungs, by coughing up phlegm that carries the tuberculosis bacteria, they can spread it from one individual to another. In other words, tuberculosis spreads only on close contact.

There are a number of different risk factors that are responsible for the development of tuberculosis. People who are constantly exposed to those people who have the bacteria are more prone to picking up the infection. Tuberculosis generally affects people of the lower socio-economic classes because of lack of sanitary conditions and closed living spaces.

Individuals who are drug abusers can also pick up tuberculosis. The presence of the human immunodeficiency virus i.e. HIV is a strong risk factor for tuberculosis as well.

Children have poor immunity in their younger age. If they are exposed to someone to tuberculosis, their weak immune system is unable to fight the bacteria and they can get infected. Individuals who have cancer have low immunity as well and are prone to developing tuberculosis. Those who suffer from diabetes also have altered immunity and may have an inability to fight the tuberculosis bacteria. Finally, individuals who are taking certain medications that suppress their immunity are also prone to picking up the tuberculosis infection.

### 3.1.3 Symptoms of TB

Although human body may harbor the bacteria that cause tuberculosis (TB), the immune system usually can prevent the body from becoming sick. For this reason, doctors make a distinction between:

**a) Latent TB**

In this condition, patients have a TB infection, but the bacteria remain in their body in an inactive state and cause no symptoms. Latent TB, also called inactive TB or TB infection, isn't contagious. It can turn into active TB, so treatment is

important for the person with latent TB and to help control the spread of TB. An estimated 2 billion people have latent TB.

**b) Active TB.**

This condition makes body sick and in most cases can spread to others. It can occur in the first few weeks after infection with the TB bacteria, or it might occur years later.

**Signs and symptoms of active TB include:**

1. Coughing that lasts three or more weeks.
2. Coughing up blood.
3. Chest pain or pain with breathing or coughing.
4. Unintentional weight loss.
5. Fatigue.
6. Fever.
7. Night sweats.
8. Chills.
9. Loss of appetite

When TB occurs outside lungs, signs and symptoms vary according to the organs involved. For example, tuberculosis of the spine may give back pain, and tuberculosis in kidneys might cause blood in urine.

The symptoms of TB and few types of PTB and EPTB are shown in Figure 3.2

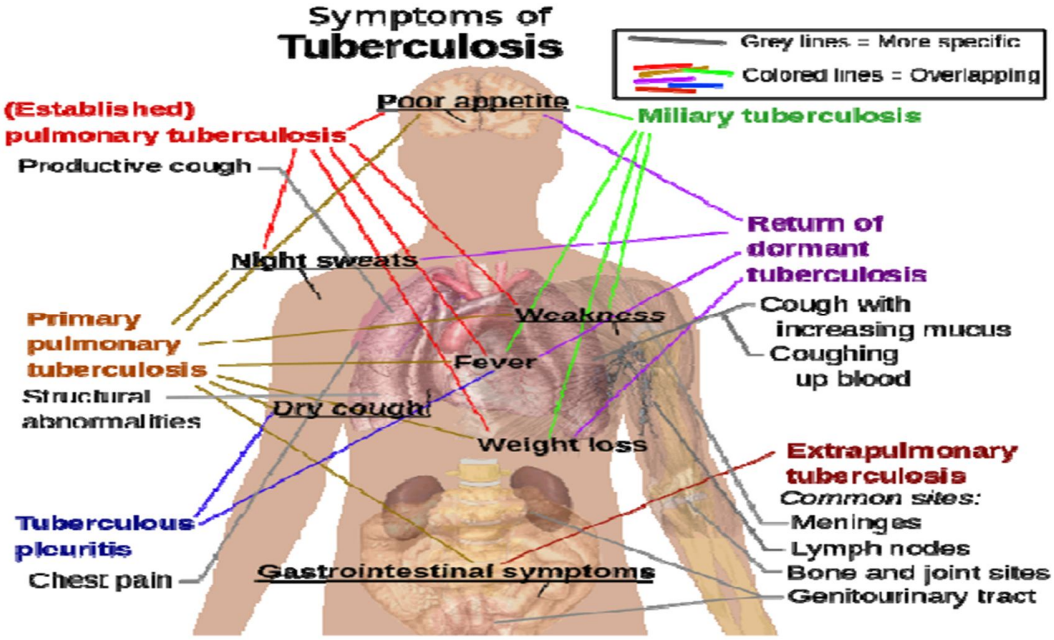


Figure 3.2: Symptoms and types of PTB and EPTB [17]

### 3.1.3 Existing methods for TB diagnosis

A number of tests are employed for TB detection like Polymerase chain reaction, nucleic acid magnification test and TB interferon gamma release assay. However all these tests have one of the major drawbacks that they are not specific, besides they require a lot of time for interpretation of results, and involve invasive techniques that are tedious to perform, some of the main TB diagnostic tests are listed below in Table 3.1.

Current diagnostic techniques such as microscopic sputum examination, TB chest X-ray, skin test and culture are not only time consuming but also have low efficacy rates. This has led to an extensive research for the development of new rapid and accurate diagnostic tools and techniques to achieve higher sensitivity and specificity to control the disease and reduce the death rates.

As mentioned in table, the existing processes of diagnosing tuberculosis are not only tedious but also take a long time for analysis.

Table 3.1: Main Shortcomings of TB diagnostic tests [17]

| Test | Methodology | Interpretations | Shortcomings |
|---|---|---|---|
| 1. Chest X-ray | X-ray of chest recorded to detect inflammation in the lungs. | Abnormal shadow visible on X-ray | Cannot exclude extra pulmonary TB |
| 2. TB skin test | Injecting small amount of Tuberculin in the lower arm and observing the swelling. | Bigger the raised area of the swelling, more are the chances of being infected by TB | May give false results if person was infected by some other bacteria. Cannot differentiate between latent TB and active TB. |
| 3. TB Interferon gamma release assays (IGRAs) | Mix blood sample with special substances to identify interferon gamma cytokine. | | Blood sample must be instantly examined, laboratory required, test is for detecting latent TB. |
| 4. Sputum smear test | A series of special stains are applied to a thin smear of patient's sputum and it is examined under microscope for signs of TB bacteria. | Morphological characteristics identification to detect presence of *M.tuberculosis* | In cases of HIV and TB co-infection, TB cannot be detected due to low levels of TB bacteria. |
| 5. Fluorescent microscopy | Illumination of patients sputum smear with quartz/high pressure mercury lamp | Morphological characteristics identification to detect presence of *M.tuberculosis* | Expensive and time consuming |
| 6. Culturing bacteria to test | Culture the bacteria from biological sample of patient on *M.tuberculosis* selective media | Detection of presence of bacteria by observing colony characteristics | Time consuming |
| 7. Polymerase chain reaction | The assay targets the KatG gene having unique sequence in TB bacterium. | Presence of the *Mycobacterium tuberculosis* complex will give the test positive. | Expensive |
| 8. GeneXpert test | Identification of DNA present in TB bacteria. | If DNA found, patient if TB positive. | Expensive. |
| 9. Nucleic acid amplification test | Amplification of nucleic acids from biological specimens of suspected patient. | If nucleic acids found, patient is TB positive. | Lower sensitivity for respiratory tract specimens. |

### 3.1.4 Radiographic impressions of TB

Major radiographic manifestations of active pulmonary TB include the following, some of which are shown in Figure 3.3:

a) Air space consolidation: lobar opacity, often reported as pneumonia or pneumonitis.
b) Miliary pattern: Fine granular sandy or seed-like appearance throughout the entirety of both lungs, reported as diffuse bilateral infiltrates, sometimes (correctly) referred to as micronodular .
c) Cavity formation: a finding with a detectible radio-dense rim, which may be continuous or discontinuous, is differentiated from a mass as it has some central complete or relative radiolucency.
d) Bronchiectasis or enlargement of airways can appear as tubular rings or cylinders of irregular diameter extending radially from the lung hila, with or without central radiolucency. [19]

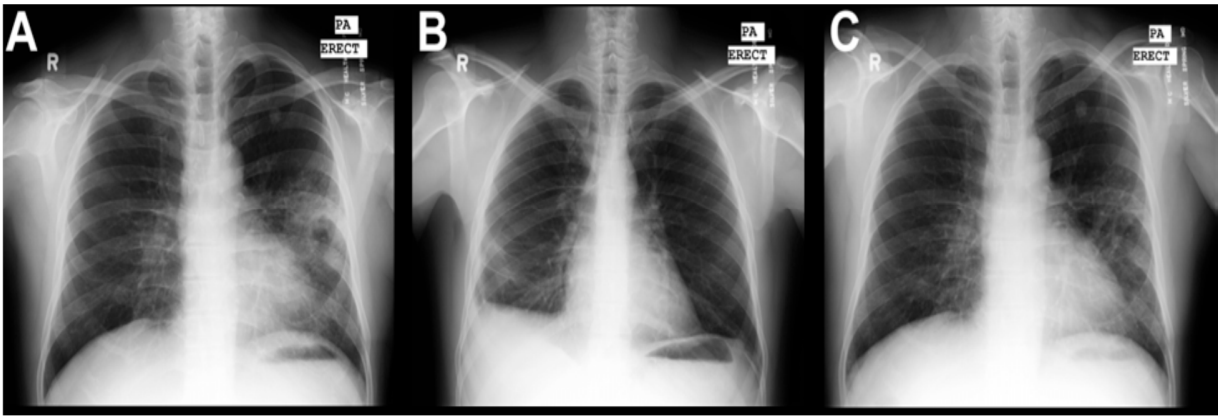


Figure 3.3: Example CXRs with manifestations of TB. CXR A and C in have infiltrates in both lungs. CXR B is a good example of pleural TB, which is indicated by the abnormal shape of the costophrenic angle of the right lung.

In addition to these findings of active disease, classic features of TB exposure include calcified granulomata and apical pleural thickening.

## 3.2 Artificial Intelligence in Medicine

The term “Artificial Intelligence” (AI) was coined by John McCarthy [20]. The word “AI,” was used for the first time in a workshop organized by McCarty, at Dartmouth College [21]. AI is an area of computer science. It helps in the development of computers that can mimic the human-like thought processing,

reasoning and self-correction ability [22]. The role of computer technologies is now increasing in the diagnostic procedures. Artificial intelligence (AI) algorithms assist in medical field for diagnosing the diseases through clinical signs and symptoms as well as radiological images of the patient. And it can be implemented for the diagnosis of TB. The term Artificial Intelligence [AI] is used for systems which execute certain tasks that would otherwise require human intervention. Tasks such as decision making, visual perception, speech recognition and translation of languages can be performed using AI systems. AI can be defined as, network that replicates the structure and function of the human intelligence.

### 3.2.1 Artificial Neural Network

Artificial Neural network is one such AI tool that has been extensively studied in the field of diagnosis of various diseases. Dr. Robert H. Nielsen, inventor of first neurocomputer, has defined a neural network as [23]: "… a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs." An Artificial neural network (ANN) draws inspiration from our brain or the biological neural networking system [24].

A neural network is formed by a series of "neurons" (or "nodes") that are organized in layers. Each neuron in a layer is connected with each neuron in the next layer through a weighted connection. The value of the weight $w_{ij}$ indicates the strength of the connection between the $i_{th}$ neuron in a layer and the $j_{th}$ neuron in the next one. The structure of a neural network is formed by an "input" layer, one or more "hidden" layers, and the "output" layer. The number of neurons in a layer and the number of layers depends strongly on the complexity of the system studied. Therefore, the optimal network architecture must be determined. The general scheme of a typical three-layered ANN architecture is given in Figure 3.4 The neurons in the input layer receive the data and transfer them to neurons in the first hidden layer through the weighted links. Here, the data are mathematically processed and the result is transferred to the neurons in the next layer. Ultimately, the neurons in the last layer provide the network's output [25]. The $j_{th}$ neuron in a hidden layer processes the incoming data () by: (i) Calculating the weighted sum and adding a "bias" term ($\theta_j$) according to Eq. 1:

$$net_j = \sum_{i=1}^{m} x_i \times w_{ij} + \theta_j \quad (j=1, 2, 3,\ldots) \qquad (1)$$

(ii) Transforming the $net_j$ through a suitable mathematical "transfer function", and (iii) Transferring the result to neurons in the next layer. Various transfer functions are available; however, the most commonly used is the sigmoid one:

$$F(x) = \frac{1}{1+e^{-x}} \quad (2)$$

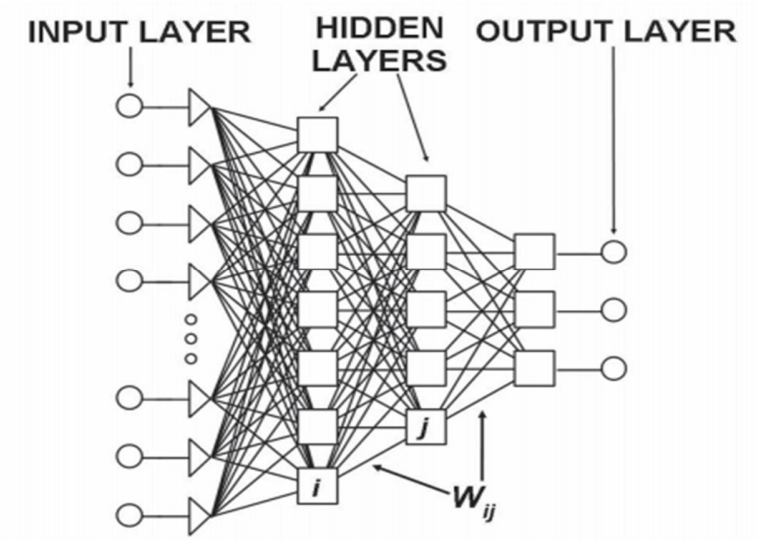


Figure 3.4: General structure of a neural network with two hidden layers. The $w_{ij}$ is the weight of the connection between the $i_{th}$ and the $j_{th}$ node

## 3.3 Convolutional Neural Network

Convolutional neural network (CNN), a class of artificial neural networks that has become dominant in various computer vision tasks, is attracting interest across a variety of domains, including radiology. CNN is designed to automatically and adaptively learn spatial hierarchies of features through back propagation by using multiple building blocks, such as convolution layers, pooling layers, and fully connected layers.

### 3.3.1 Building Blocks of CNN Architecture

The CNN architecture includes several building blocks, such as convolution layers, pooling layers, and fully connected layers. A typical architecture consists of repetitions of a stack of several convolution layers and a pooling layer, followed by one or more fully connected layers. The step where input data are transformed into output through these layers is called forward propagation (Figure 3.5). Although

convolution and pooling operations described in this section are for 2D-CNN, similar operations can also be performed for three-dimensional (3D)-CNN.

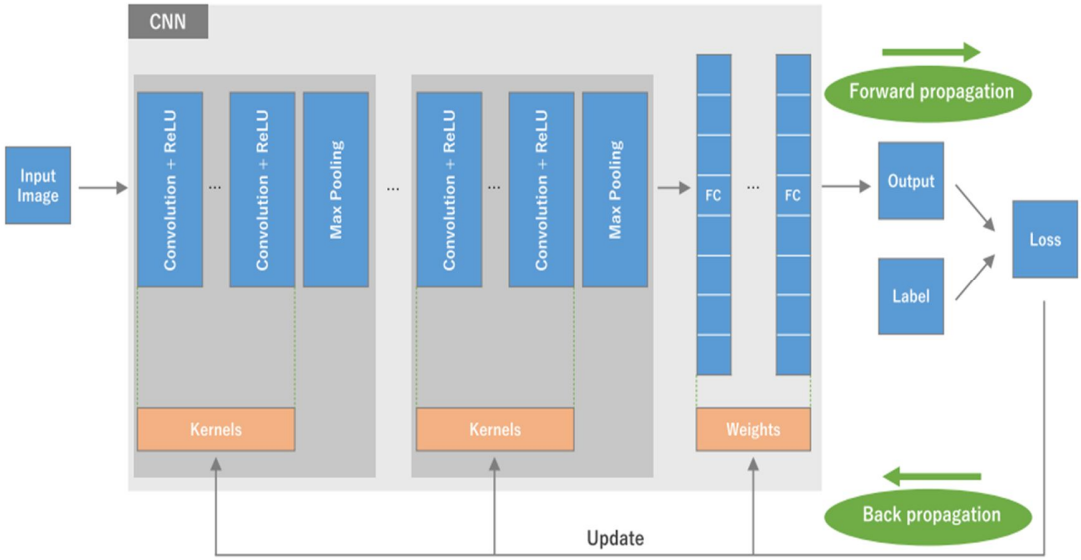


Figure 3.5: An overview of a convolutional neural network (CNN) architecture and the training process.

**Convolution layer**

A convolution layer is a fundamental component of the CNN architecture that performs feature extraction, which typically Convolution, Convolution is a specialized type of linear operation used for feature extraction, where a small array of numbers, called a kernel, is applied across the input, which is an array of numbers, called a tensor. An element-wise product between each element of the kernel and the input tensor is calculated at each location of the tensor and summed to obtain the output value in the corresponding position of the output tensor, called a feature map (Figure 3.6,a–c). This procedure is repeated applying multiple kernels to form an arbitrary number of feature maps. Also, each feature map can be formulated with respect to several input maps. This can be mathematically written as

$$y_j^i = \mathrm{f}\left(\sum_{i \in N_j} y_i^{l-1} \mathrm{x} M_{ij}^l + a_j^l\right) \qquad (3)$$

Where $y_j^i$ depicts the $j_{th}$ output feature of the $l_{th}$ layer, f (.) is a nonlinear function, $N_j$ is the input map selection, yl−1i refer to the $j_{th}$ input map of $l-1_{th}$

layer, $M_{ij}^{l}$ is the kernel for the input i and output map M in the $l_{th}$ layer, and $a_j^l$ is the addictive bias associated with the $j_{th}$ map output[7].

The key feature of a convolution operation is weight sharing: kernels are shared across all the image positions. Weight sharing creates the following characteristics of convolution operations: (1) letting the local feature patterns extracted by kernels translation b invariant as kernels travel across all the image positions and detect learned local patterns, (2) learning spatial hierarchies of feature patterns by downsampling in conjunction with a pooling operation, resulting in capturing an increasingly larger field of view, and (3) increasing model efficiency by reducing the number of parameters to learn in comparison with fully connected neural networks.

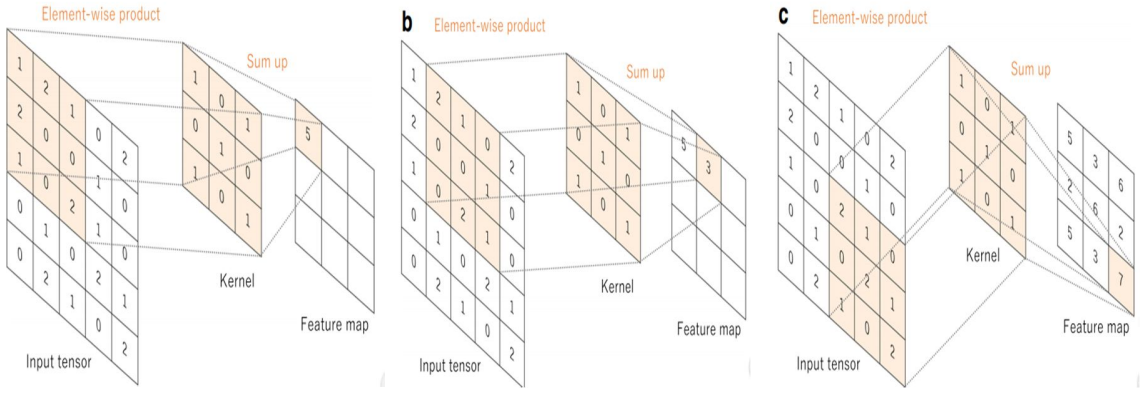


Figure 3.6, a–c: An example of convolution operation with a kernel size of 3 × 3.

**Nonlinear Activation Function**

The outputs of a linear operation such as convolution are then passed through a nonlinear activation function. Although smooth nonlinear functions, such as sigmoid or hyperbolic tangent (tanh) function, were used previously because they are mathematical representations of a biological neuron behavior, the most common nonlinear activation function used presently is the rectified linear unit (ReLU), which simply computes the function : f(x) = max(0, x) (Figure 3.7) [26, 27, 28–30].

**Pooling layer**

A pooling layer provides a typical downsampling operation which reduces the in-plane dimensionality of the feature maps in order to introduce a translation invariance to small shifts and distortions, and decrease the number of subsequent

learnable parameters. It is of note that there is no learnable parameter in any of the pooling layers, whereas filter size, stride, and padding are hyperparameters in pooling operations, similar to convolution operations.

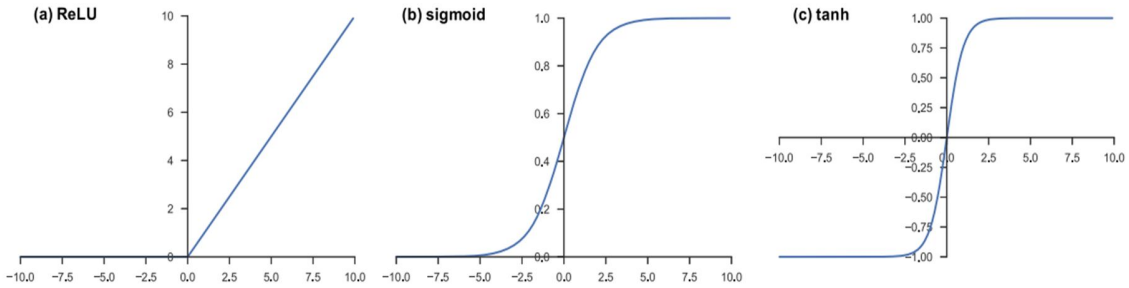


Figure 3.7: Activation functions commonly applied to neural networks: a rectified linear unit (ReLU), b sigmoid, and c hyperbolic tangent (tanh)

**a) Max pooling**

The most popular form of pooling operation is max pooling, which extracts patches from the input feature maps, outputs the maximum value in each patch, and discards all the other values (Figure 3.7). A max pooling with a filter of size 2 × 2 with a stride of 2 is commonly used in practice. This downsamples the in-plane dimension of feature maps by a factor of 2. Unlike height and width, the depth dimension of feature maps remains unchanged.

The downsampling of MaxPooling is formally written as:

$$y_j^i = f(a_j^l \text{down}(y_i^{l-1} - M_{ij}^l) + a_j^l) \qquad (4)$$

Where $a_j^l$ represent the multiplicative bias of every feature output map j that scale the output back to its initial range, down (.) can be substituted for either avg (.) or max (.) over an n × n window effectively scaling the input map by n times in every dimension.

Global average pooling, another pooling operation worth noting is a global average pooling [31].

**b) A global average pooling**

Performs an extreme type of downsampling, where a feature map with size of height × width is downsampled into a 1 × 1 array by simply taking the average of all the elements in each feature map, whereas the depth of feature maps is retained.

This operation is typically applied only once before the fully connected layers. The advantages of applying global average pooling are as follows: (1) reduces the number of learnable parameters and (2) enables the CNN to accept inputs of variable size.

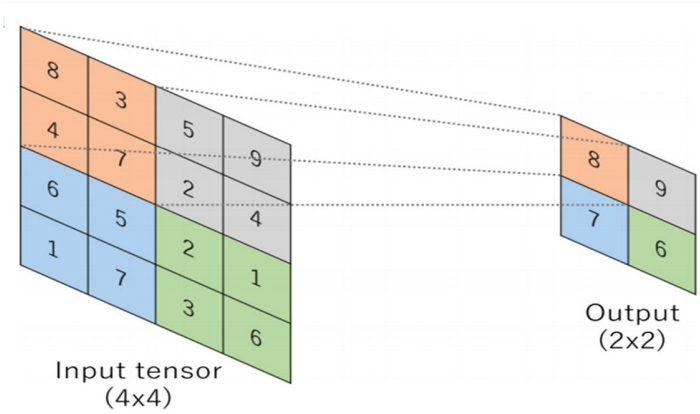


Figure 3.8: An example of max pooling operation with a filter size of 2 × 2.

**Fully Connected Layer**

The output feature maps of the final convolution or pooling layer is typically flattened, i.e., transformed into a one-dimensional (1D) array of numbers (or vector), and connected to one or more fully connected layers, also known as dense layers, in which every input is connected to every output by a learnable weight. Once the features extracted by the convolution layers and downsampled by the pooling layers are created, they are mapped by a subset of fully connected layers to the final outputs of the network, such as the probabilities for each class in classification tasks. The final fully connected layer typically has the same number of output nodes as the number of classes. Each fully connected layer is followed by a nonlinear function, such as ReLU.

## 3.3.2 Training a Network

It is a process of finding kernels in convolution layers and weights in fully connected layers which minimize differences between output predictions and given ground truth labels on a training dataset. Various training algorithms are available. However, the most commonly used is back propagation; Backpropagation algorithm is the method commonly used for training neural networks where loss function and gradient descent optimization algorithm play essential roles.

It is requires the use of two training parameters:

(i) Learning rate.

(ii) Momentum.

Usually, high values of such parameters lead to unstable learning, and therefore poor generalization ability of the network. The optimal values of the training parameters depend upon the complexity of the studied system. In general, the value of momentum is lower than that of learning rate. In addition, the sum of their values should be approximately equal to one. [25]

A model performance under particular kernels and weights is calculated by a loss function through forward propagation on a training dataset, and learnable parameters, namely kernels and weights are updated according to the loss value through an optimization algorithm called backpropagation and gradient descent, among others (Figure 3.5).

**Loss Function**

A loss function, also referred to as a cost function, measures the compatibility between output predictions of the network through forward propagation and given ground truth labels. Commonly used loss function for multiclass classification is cross entropy, whereas mean squared error is typically applied to regression to continuous values. A type of loss function is one of the hyperparameters and needs to be determined according to the given tasks.

**Overfitting**

It is refers to a situation where a model learns statistical regularities specific to the training set, i.e., ends up memorizing the irrelevant noise instead of learning the signal, and, therefore, performs less well on a subsequent new dataset. This is one of the main challenges in machine learning, as an overfitted model is not generalizable to never-seen-before data. In that sense, a test set plays a pivotal role in the proper performance evaluation of machine learning models, as discussed in the previous section. A model trained on a larger dataset typically generalizes better, though that is not always attainable in medical imaging. The other solutions include regularization with dropout or weight decay, batch normalization, and data augmentation, as well as reducing architectural complexity. Dropout is a recently introduced regularization technique where randomly selected activations are set to 0 during the training, so that the model becomes less sensitive to specific weights in the network [32].

**Dropout**

It is a regularization method that approximates training a large number of neural networks with different architectures in parallel.

During training, some number of layer outputs are randomly ignored or "dropped out." This has the effect of making the layer look-like and be treated-like a layer with a different number of nodes and connectivity to the prior layer. In effect, each update to a layer during training is performed with a different "view" of the configured layer.

## 3.4 K-fold Cross Validation

A more expensive and less naïve approach would be to perform K-fold Cross Validation; there is a parameter '*k*'. This parameter decides how many folds the dataset is going to be divided. Every fold gets chance to appear in the training set (*k-1*) times, which in turn ensures that every observation in the dataset appears in the dataset, thus enabling the model to learn the underlying data distribution better.

The value of '*k*' used is generally between 5 or 10. The value of '*k*' should not be too low or too high. If the value of '*k*' is too low (say $k = 2$), we will have a highly biased model. This case is similar to that of splitting the dataset into training and validation sets, hence the bias will be high and variance low. If the value of '*k*' is large (say $k = n$ (the number of observations)), then this approach is called Leave One out CV (LOOCV). ). LOOCV means K=N, where N is the number of samples in dataset. As the number of models trained is maximized, the precision of the model performance average is maximized too, but so is the cost of training due to the sheer amount of models that must be trained In this case, bias will be low but the variance will be high and the model will overfit, resulting in the model to fail in generalizing over the test set.

In a binary classification problem, a look at Stratified Cross Validation might be taken . It extends K-fold Cross Validation by ensuring an equal distribution of the target classes over the splits. This ensures that classification problem is balanced. It doesn't work for multiclass classification due to the way that samples are distributed.

The dataset is split into K partitions of equal size. K–1 are used for training, while one is used for testing. This process is repeated K times, with a different partition used for testing each time.

## 3.5 Transfer Learning

In fact, transfer learning is not a concept which just cropped up in the 2010s. The Neural Information Processing Systems (NIPS) 1995 workshop learning to Learn: Knowledge Consolidation and Transfer in Inductive Systems are believed to have provided the initial motivation for research in this field. Since then, terms such as Learning to Learn, Knowledge Consolidation, and Inductive Transfer have been used interchangeably with transfer learning. Invariably, different researchers and academic texts provide definitions from different contexts [33]. In their famous book, Deep Learning, Goodfellow et al refer to transfer learning in the context of generalization.

Their definition is as follows:

"Situation where what has been learned in one setting is exploited to improve generalization in another setting."

Thus, the key motivation, especially considering the context of deep learning is the fact that most models which solve complex problems need a whole lot of data, and getting vast amounts of labeled data for supervised models can be really difficult, considering the time and effort it takes to label data points.

However, getting such a dataset for every domain is tough. Besides, most deep learning models are very specialized to a particular domain or even a specific task. While these might be state-of-the-art models, with really high accuracy and beating all benchmarks, it would be only on very specific datasets and end up suffering a significant loss in performance when used in a new task which might still be similar to the one it was trained on.

This forms the motivation for transfer learning, which goes beyond specific tasks and domains, and tries to see how to leverage knowledge from pre-trained models and use it to solve new problems.

**Understanding Transfer Learning**

The first thing to remember here is that, transfer learning, is not a new concept which is very specific to deep learning. There is a stark difference between the traditional approach of building and training machine learning models, and using a methodology following transfer learning principles.

Traditional learning is isolated and occurs purely based on specific tasks, datasets and training separate isolated models on them. No knowledge is retained which can be transferred from one model to another. In transfer learning, knowledge

(features, weights etc) can be leveraged from previously trained models for training newer models and even tackle problems like having less data for the newer task.

Transfer learning enables to utilize knowledge from previously learned tasks and apply them to newer, related ones. If there significantly more data for task *T1*, learning must be utilized, and generalized this knowledge (features, weights) for task *T2* (which has significantly less data). In the case of problems in the computer vision domain, certain low-level features, such as edges, shapes, corners and intensity, can be shared across tasks, and thus enable knowledge transfer among tasks, Also , knowledge from an existing task acts as an additional input when learning a new target task[34].

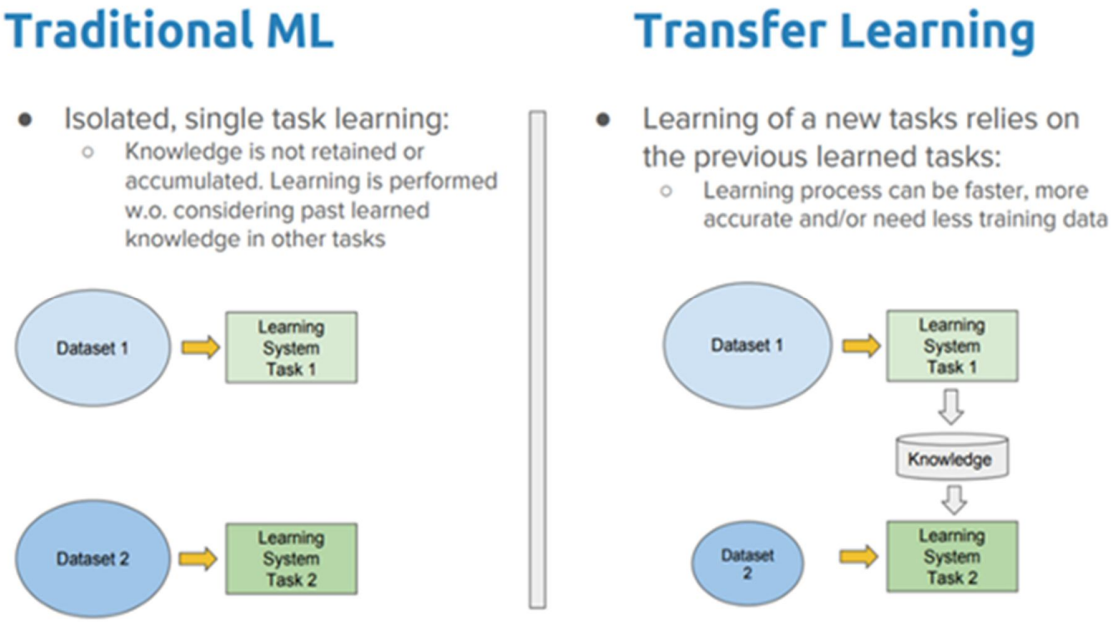


Figure 3.9: Traditional Learning vs Transfer Learning

Generally, Transfer learning is a common and effective strategy to train a network on a small dataset, where a network is pretrained on an extremely large dataset, such as ImageNet, which contains 1.4 million images with 1000 classes, then reused and applied to the given task of interest. The underlying assumption of transfer learning is that generic features learned on a large enough dataset can be shared among seemingly disparate datasets. This portability of learned generic features is a unique advantage of deep learning that makes itself useful in various domain tasks with small datasets. At present, many models pretrained on the

ImageNet challenge dataset are open to the public and readily accessible, along with their learned kernels and weights, such as AlexNet [35], VGG [36], ResNet [37], Inception [38], and DenseNet [39]. In practice, there are two ways to utilize a pretrained network: fixed feature extraction and fine-tuning (Figure 3.10).
In general, there are two types of transfer learning in the context of deep learning:

1. Transfer learning via feature extraction.
2. Transfer learning via fine-tuning.

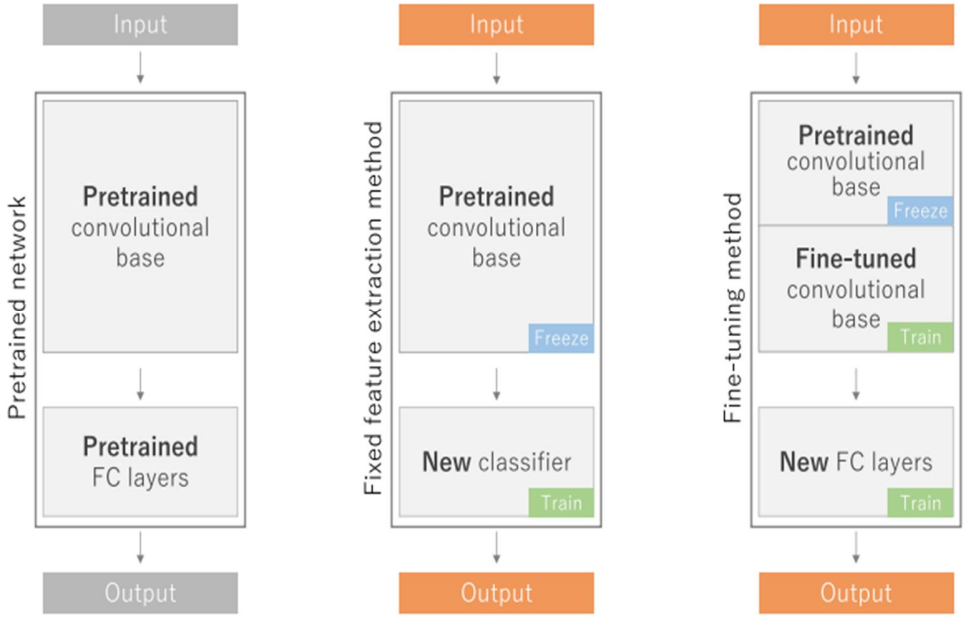


Figure 3.10: Transfer learning and its common ways used in pretrained network

A fixed feature extraction method is a process to remove fully connected layers from a network pretrained on ImageNet and while maintaining the remaining network, which consists of a series of convolution and pooling layers, referred to as the convolutional base, as a fixed feature extractor. In this scenario, any machine learning classifier, such as random forests and support vector machines, as well as the usual fully connected layers in CNNs, can be added on top of the fixed feature extractor, resulting in training limited to the added classifier on a given dataset of interest.

A fine-tuning method, is to not only replace fully connected layers of the pretrained model with a new set of fully connected layers to retrain on a given dataset, but to fine-tune all or part of the kernels in the pretrained convolutional

base by means of backpropagation. All the layers in the convolutional base can be fine-tuned or, alternatively, some earlier layers can be fixed while fine-tuning the rest of the deeper layers. This is motivated by the observation that the early-layer features appear more generic, including features such as edges applicable to a variety of datasets and tasks, whereas later features progressively become more specific to a particular dataset or task [40, 41].

## 3.6 The Pretrained VGG

VGG is an acronym for the Visual Geometric Group from Oxford University and VGG-16 is a network with 16 layers proposed by the Visual Geometric Group. These 16 layers contain the trainable parameters and there are other layers also like the Max pool layer but those do not contain any trainable parameters. This architecture was the 1st runner up of the Visual Recognition Challenge of 2014 i.e. ILSVRC-2014 and was developed by Simonyan and Zisserman.

The VGG research group released a series of the convolution network model starting from VGG11 to VGG19. The main intention of the VGG group on depth was to understand how the depth of convolutional networks affects the accuracy of the models of large-scale image classification and recognition. The minimum VGG11 has 8 convolutional layers and 3 fully connected layers as compared to the maximum VGG19 which has 16 convolutional layers and the 3 fully connected layers. The different variations of VGGs are exactly the same in the last three fully connected layers. The overall structure includes 5 sets of convolutional layers, followed by a MaxPool. But the difference is that as the depth increases VGG11 to VGG19 more and more cascaded convolutional layers are added in the five sets of convolutional layers.

The below-shown table (Table3.2) is the overall network configuration of different models created by VGG that uses the same principle but only varies in depth [42].

Table 3.2: VGG ConvNet configuration [42].

| ConvNet Configuration | | | | | |
|---|---|---|---|---|---|
| A | A-LRN | B | C | D | E |
| 11 weight layers | 11 weight layers | 13 weight layers | 16 weight layers | 16 weight layers | 19 weight layers |
| input (224 × 224 RGB image) | | | | | |
| conv3-64 | conv3-64<br>**LRN** | conv3-64<br>**conv3-64** | conv3-64<br>conv3-64 | conv3-64<br>conv3-64 | conv3-64<br>conv3-64 |
| maxpool | | | | | |
| conv3-128 | conv3-128 | conv3-128<br>**conv3-128** | conv3-128<br>conv3-128 | conv3-128<br>conv3-128 | conv3-128<br>conv3-128 |
| maxpool | | | | | |
| conv3-256<br>conv3-256 | conv3-256<br>conv3-256 | conv3-256<br>conv3-256 | conv3-256<br>conv3-256<br>**conv1-256** | conv3-256<br>conv3-256<br>**conv3-256** | conv3-256<br>conv3-256<br>conv3-256<br>**conv3-256** |
| maxpool | | | | | |
| conv3-512<br>conv3-512 | conv3-512<br>conv3-512 | conv3-512<br>conv3-512 | conv3-512<br>conv3-512<br>**conv1-512** | conv3-512<br>conv3-512<br>**conv3-512** | conv3-512<br>conv3-512<br>conv3-512<br>**conv3-512** |
| maxpool | | | | | |
| conv3-512<br>conv3-512 | conv3-512<br>conv3-512 | conv3-512<br>conv3-512 | conv3-512<br>conv3-512<br>**conv1-512** | conv3-512<br>conv3-512<br>**conv3-512** | conv3-512<br>conv3-512<br>conv3-512<br>**conv3-512** |
| maxpool | | | | | |
| FC-4096 | | | | | |
| FC-4096 | | | | | |
| FC-1000 | | | | | |
| soft-max | | | | | |

Features of VGG-16 network

1. **Input Layer:** It accepts color images as an input with the size 224 x 224 and 3 channels i.e. Red, Green, and Blue.
2. **Convolution Layer:** The images pass through a stack of convolution layers where every convolution filter has a very small receptive field of 3 x 3 and stride of 1. Every convolution kernel uses row and column padding so that the size of input as well as the output feature maps remains the same or in other words, the resolution after the convolution is performed remains the same.
3. **Max pooling:** It is performed over a max-pool window of size 2 x 2 with stride equals to 2, which means here max pool windows are non-overlapping windows.

4. Not every convolution layer is followed by a max pool layer as at some places a convolution layer is following another convolution layer without the max-pool layer in between.
5. The first two fully connected layers have 4096 channels each and the third fully connected layer which is also the output layer have 1000 channels, one for each category of images in the imagenet database.
6. The hidden layers have ReLU as their activation function.

# Chapter Four

# Proposed Model

## 4.1 Overview of the proposed method

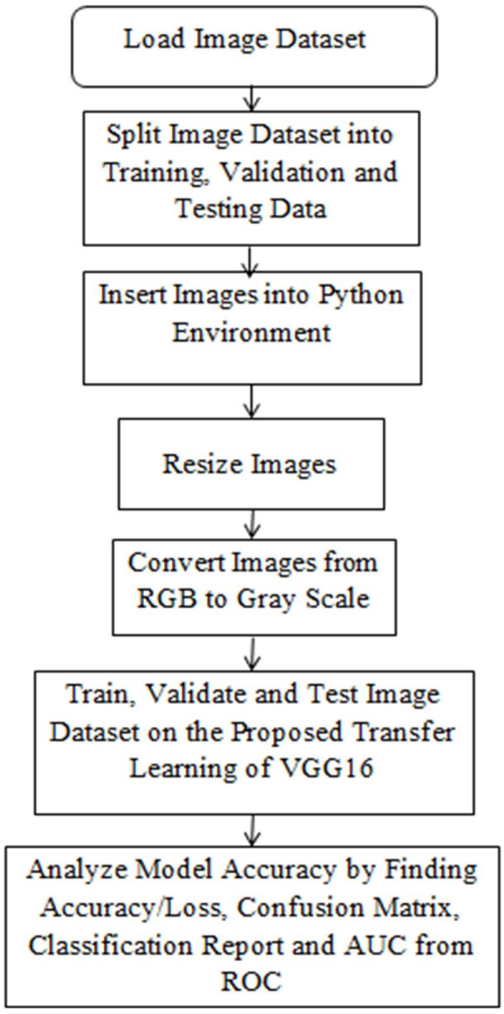


Figure 4.1: Block diagram of the proposed method

The process block diagram for the proposed method is shown above in Figure 4.1. chest X-rays images used are taken from Shenzhen dataset; Firstly images are resized, converted to gray scale, and then they are divided into training, validation and testing data, images are flowed through the transfer learned pre-trained model (Figure 4.3), to be classified into normal and abnormal TB, finally the accuracy and performance of classification is analyzed.

## 4.2 Dataset Description

The dataset we used is called Shenzhen dataset. The Shenzhen dataset was collected in collaboration with Shenzhen No.3 People's Hospital, Guangdong Medical College, Shenzhen, China. The chest X-rays are from outpatient clinics and were captured as part of the daily hospital routine within a 1-month period, mostly in September 2012, using a Philips DR Digital Diagnostic system.

The set contains **662** frontal chest X-rays, of which:

**326** are normal cases.

**336** are cases with manifestations of TB,

Including pediatric X-rays AP, the X-rays are provided in PNG format. Their size can vary but is approximately 3K × 3K pixels. All image file names follow the same template: CHNCXR_####_X.png, where #### represents a 4-digit numerical identifier, and X is either 0 for a normal X-ray or 1 for an abnormal X-ray. The clinical reading for each X-ray is saved in a text file following the same format, except that the ending ".png" is replaced with ".txt". Each reading contains the patient's age, gender, and abnormality seen in the lung, if any.

Shenzhen dataset free available on internet and it is downloaded from https: //lhncbc.nlm.nih.gov/LHC-publications/pubs/TuberculosisChestXrayImageData Sets .html

## 4.3 Image preprocessing

Images of Shenzhen dataset were resized to 96 x 96 and it were converted into gray scale image then it were loaded into python environment using CV2 libraries , each image defined by x and y where x is an array of image itself while y is an indication of its label (either 0 or 1) .

Figure 4.2: An example of preprocessed image from image dataset

## 4.4 Dataset Splitting

As dataset classified by CNN need to be trained, validated and tested, chest x-ray images in dataset will be split into three categories by the percent 70%, 10% and 20 % respectively as follows:

**464** chest X-rays images for training.
**66** chest X-rays images for validation.
**132** chest X-rays images for testing.
Training and validation data are combined since cross validation is applied.

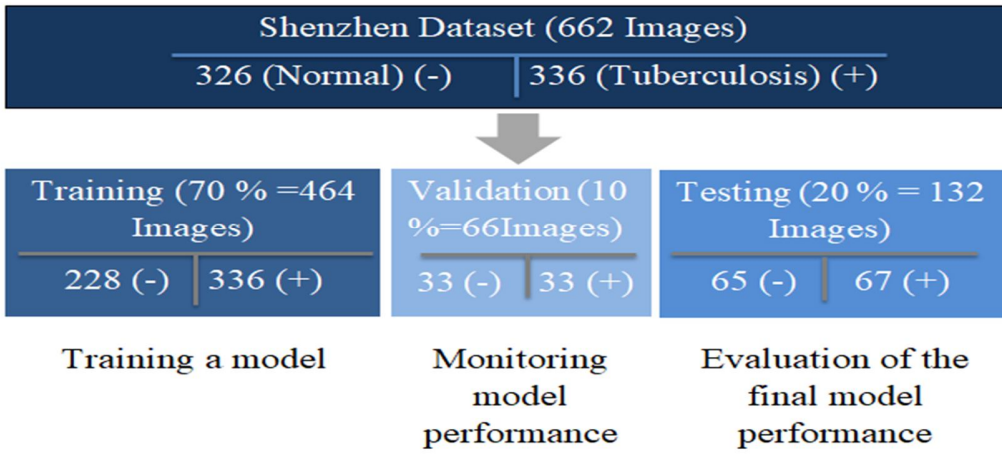


Figure 4.3: Dataset splitting

## 4.5 Proposed Pre-trained Model


Figure 4.4: The proposed pre-trained model

## Dataset

Dataset used for pre-training of vgg16 is ImageNet , ImageNet is a dataset of over 15 million labeled high-resolution images belonging to roughly 22,000 categories. The images were collected from the web and labeled by human labelers using Amazon's Mechanical Turk crowd-sourcing tool. Starting in 2010, as part of the Pascal Visual Object Challenge, an annual competition called the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) has been held. ILSVRC uses a subset of ImageNet with roughly 1000 images in each of 1000 categories. In all, there are roughly 1.2 million training images, 50,000 validation images, and 150,000 testing images. ImageNet consists of variable-resolution images.

Therefore, the images have been down-sampled to a fixed resolution of 256×256. Given a rectangular image, the image is rescaled and cropped out the central 256×256 patch from the resulting image.

**The Architecture**

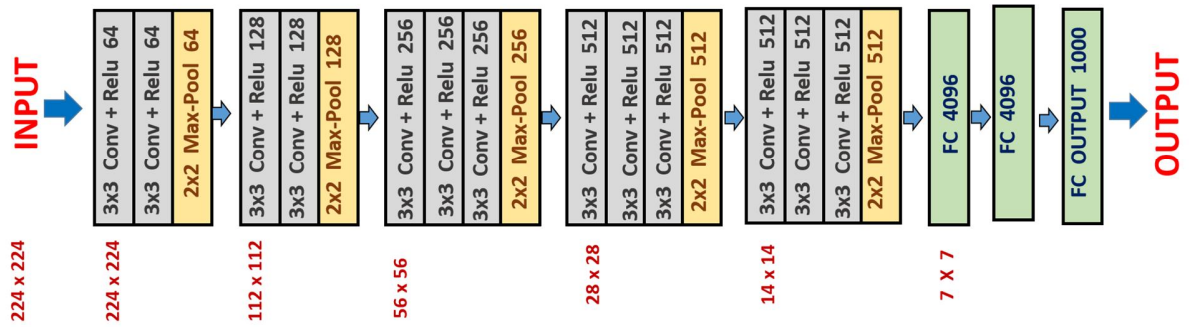


Figure 4.5: VGG16 Architecture

Three Fully-Connected (FC) layers follow a stack of convolutional layers (which has a different depth in different architectures): the first two have 4096 channels each, the third performs 1000-way ILSVRC classification and thus contains 1000 channels (one for each class). The final layer is the soft-max layer. The configuration of the fully connected layers is the same in all networks.

All hidden layers are equipped with the rectification (ReLU) non-linearity. It is also noted that none of the networks (except for one) contain Local Response Normalisation (LRN), such normalization does not improve the performance on the ILSVRC dataset, but leads to increased memory consumption and computation time.

## 4.6 Proposed VGG16 implementation

As mentioned above VGG16 have 13 conv layers and 5 pooling layers. Under normal circumstances the model usually has 3 fully connected dense layers followed by a soft-max layer (Figure 4.2). Soft-max layer is used as the final layer for multiclass prediction. It applies exponential function on all the input coming from the ConvLayer to obtain the final prediction. The results are normalized by dividing the output by the sum of all exponentials.

However, in the proposed model this layer is modified to use sigmoid as there are only had two outputs. In this study weights from ImageNet are used which in turn switches off the dense layers as the weights are provided from ImageNet by

default. ImageNet [43] is a library containing large scale ontology of images. It is built on the structure provided by WordNet and contains more than 14 million images that could be used for classification purposes. This library is provided by Keras. Activation functions are present between each Conv layer which defines the output of a neuron given set of inputs. Each Conv layer in VGG16 has Rectified Linear Unit activation (ReLU).

ReLu is sometimes chosen over sigmoid for activation as it trains much faster.in this model there is no normalization done in any of the layers as normalization does not have any significant impact on the accuracy and only manages to increase processing time. The Conv layers are labeled conv2d. The image starts with 96 × 96 with 3 dimensions RGB which then undergoes convolution in 2 hidden layers having 64 weights. Max-pooling reduces sample size to 48 from 96. This is followed by two other conv layers having 128 weights. The weights or filters keep increasing all the way up to 512. Here there are 4 max-pooling layers in between the conv layers. The total numbers of parameters obtained are 14,714,688. Since there is no normalization, all parameters are used for training the model.


Figure 4.6: Proposed VGG16 implementation

**Calculations involved in getting output size from each layer**

The formula involved in calculating the output size from each convolution layer is given as- [(N-f+ 2P)/S] + 1

Where

N = Input size , f = Filter size

S = Strides , P= padding

**Input Layer:**

- The size of the input image is 96 x 96.

**Convolution Layer - 1:**

Input size = N = 96

Filter size = f = 3 x 3

No. of filters = 64

Strides = S = 1

Padding = P = 1

Output feature map size = [(96–3+2)/1] + 1 = 96

Output with channels = 96 x 96 x 64

**Convolution Layer - 2:**

Input size = N = 96

Filter size = f = 3 x 3

No. of filters = 64

Strides = S = 1

Padding = P = 1

Output feature map size = [(96–3+2)/1] + 1 = 96

Output with channels = 96 x 96 x 64

**Max-Pooling Layer - 1:**

Input size = N = 224

Filter size = f = 2 x 2

Strides = S = 2

Padding = P = 0

Output feature map size = [(96–2+0)/2] + 1 = 48

Output with channels = 48x 48 x 64

**Convolution Layer - 3:**

Input size = N = 48

Filter size = f = 3 x 3

No. of filters = 128

Strides = S = 1

Padding = P = 1

Output feature map size = [(48–3+2)/1] + 1 = 48

Output with channels = 48x 48 x 64

**Convolution Layer - 4:**

Input size = N = 48

Filter size = f = 3 x 3

No. of filters = 128

Strides = S = 1
Padding = P = 1
Output feature map size = [(48–3+2)/1] + 1 = 48
Output with channels = 48 x 48 x 128

**Max-Pooling Layer - 2:**

Input size = N = 48
Filter size = f = 2 x 2
Strides = S = 2
Padding = P = 0
Output feature map size = [(48–2+0)/2] + 1 = 24
Output with channels = 24 x 24 x 128

**Convolution Layer - 5:**

Input size = N = 24
Filter size = f = 3 x 3
No. of filters = 256
Strides = S = 1
Padding = P = 1
Output feature map size = [(24–3+2)/1] + 1 = 24
Output with channels = 24 x 24 x 128

**Convolution Layer - 6:**

Input size = N = 24
Filter size = f = 3 x 3
No. of filters = 256
Strides = S = 1
Padding = P = 1
Output feature map size = [(24–3+2)/1] + 1 = 24
Output with channels = 24 x 24 x 128

**Convolution Layer - 7:**

Input size = N = 24
Filter size = f = 3 x 3
No. of filters = 256
Strides = S = 1
Padding = P = 1
Output feature map size = [(24–3+2)/1] + 1 = 24
Output with channels = 24 x 24 x 256

**Max-Pooling Layer - 3:**

Input size = N = 24
Filter size = f = 2 x 2
Strides = S = 2
Padding = P = 0
Output feature map size = [(24–2+0)/2] + 1 = 12
Output with channels = 12 x 12 x 256

Similar calculations will be performed for the rest of the network.

The results obtained from VGG16 are fitted to another separate conv layer obtained from conv-2d in Keras. This layer uses a sigmoid as activation function. Since there are only two outputs in our dataset, sigmoid is the perfect activation function where values only oscillate between −1 and 1 which reflect the class labels on shenzhen dataset. A positive value indicates that the patient suffers from tuberculosis while a negative value indicates that the person does not. This is followed by the application of Adaptive Moment Estimation (Adam) optimization to update network weights of the training and also reduce overfitting. Adam optimizes the results faster than other stochastic gradient descent optimizers because it is capable of calculating learning rate of each parameter in the model and also stores the momentum changes. Learning rate and decay of Adam optimizer were 0.0001, 1e-6 respectively.

## 4.7 The typical transfer-learning workflow

This leads to how a typical transfer learning workflow can be implemented in keras:

1. A vgg16 base model is instantiated and pre-trained weights are loaded into it.
2. All layers are frozen in the base model by setting trainable = False.
3. A new model on top of the output of one (or several) layers are Created from the base model.
4. New model is trained on new dataset (Shenzhen dataset).

On the process of training loss function used is Binary Cross-Entropy; it is the default loss function to use for binary classification problems. It is intended for use with binary classification where the target values are in the set {0, 1}.

Mathematically, it is the preferred loss function under the inference framework of maximum likelihood. It is the loss function to be evaluated first and only changed if there is a good reason. Cross-entropy will calculate a score that summarizes the average difference between the actual and predicted probability distributions for predicting class 1. The score is minimized and a perfect cross-entropy value is 0. Cross-entropy can be specified as the loss function in Keras by specifying '*binary_crossentropy*' when compiling the model.
For validation, k folds cross validation were used, k=5 were used to perform cross validation by using the algorithm of k-Fold technique:

1. A number of folds – k is picked. Which is 5 in the proposed.
2. The dataset is split into 5 equal (if possible) parts (they are called folds)
3. k – 1 folds will be chosen which will be the training set. The remaining fold will be the validation set.
4. The model is trained on the training set. On each iteration of cross-validation, a new model must be trained independently of the model trained on the previous iteration
5. Validation will be performed on the validation set.
6. The result of the validation is saved.
7. Steps 3 – 6 are repeated 5 times. Each time use the remaining fold as the test set. In the end, the model should have validated on every fold.
8. The final score will be the average of the results saved on step 6.

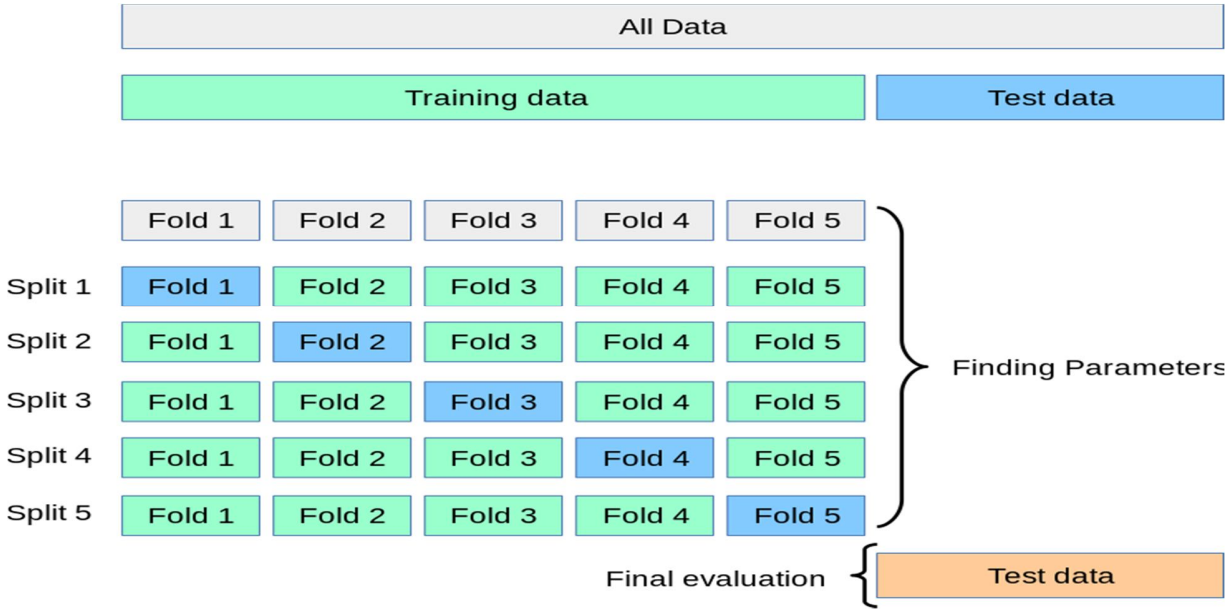


Figure 4.7: 5 Folds cross validation

## 4.8 Material and Software

The proposed model were executed on windows 10 operating system, with 500 GB hard disk, 4GB RAM, its developed in Python version: 3.6.12 64-bit, Spyder version: 4.1.5 was used as an integrated development environment (IDE), libraries used in this model were:

1. Keras.
2. Os.
3. Numpy.
4. Pandas.
5. Random.
6. Cv2.
7. Matplotlib.
8. Scikit-learn.

These libraries were downloaded by anaconda navigator and anaconda prompt.

***Note** one of the limitations in application of proposed model is hardware specification

# Chapter Five

# Results and Discussion

## 5. 1 Performance Evaluation of Classification

**A true positive (TP)** is one that detects the condition when the condition is present, label which was predicted positive and is actually positive.

**A true negative (TN**) result is one that does not detect the condition when the condition is absent, label which was predicted negative and is actually negative.

**A false positive (FP)** result is one that detects the condition when the condition is absent, Label which was predicted as positive but is actually negative.

**A false negative (FN)** result is one that does not detect the condition when the condition is present, labels which was predicted as negative, but is actually positive.

**Sensitivity, specificity and accuracy are described in terms of TP, TN, FN and FP:**

**Sensitivity (%)**(**Recall**) **(**True positive rate) It is the 'Completeness', ability of the model to identify all relevant instances measures the ability of a test to detect the condition when the condition is present,. Thus,

$$\text{Sensitivity} = \frac{TP}{(TP+FN)} \text{ x } 100$$

**Specificity (%)** (True negative rate) measures the ability of a test to correctly exclude the condition (not detect the condition) when the condition is absent. Thus,

$$\text{Specificity} = \frac{TN}{(TN+FP)} \text{x}100$$

**Accuracy (%):** is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations.

$$\text{Accuracy} = \frac{(TP+TN)}{(TP+FN)+(FP+TN)} \text{X}100$$

**Precision:** It is the 'Exactness', ability of the model to return only relevant instances ,Precision is the ratio of correctly predicted positive observations to the total predicted positive observations.

$$\text{Precision} = \frac{TP}{(TP+FP)} \text{ x100}$$

**F1-Score:** F1 Score is the weighted average of Precision and Recall. Used to indicate a balance between Precision & Recall providing each equal weightage, it ranges from 0 to 1. F1 Score reaches its best value at 1 (perfect precision & recall) and worst at 0.

$$\text{F1-Score} = \frac{2*(\text{Precision} * \text{Sensitivity})}{(\text{Precision} + \text{Sensitivity})} \text{ x100}$$

**Confusion Matrix:** Confusion matrix is a performance measurement technique for machine learning classification, it is a kind of table which helps to know the performance of the classification model on a set of test data for that true values are known, in the proposed model confusion matrix is 2x2 since classification is binary classification.

**AUC of ROC curve:** Another metric used is the AUC - The area under the Receiver Operating Characteristic Curve (ROC) rate). A Receiver Operating Characteristic (ROC) curve plots the true positive rate versus the false positive rate.

Area under the ROC curve (AUC) is used to measure performance of proposed classifier model. A larger AUC value generally means a larger sensitivity and/or specificity for the same operating point The ROC curves show different possible operating points depending on the confidence threshold set for the classifier. The Y-axis indicates the sensitivity (or recall) of a system, and the X-axis indicates the corresponding false positive rate, which is (1 – specificity).

## 5.2 Performance Evaluation of image Classification without cross validation

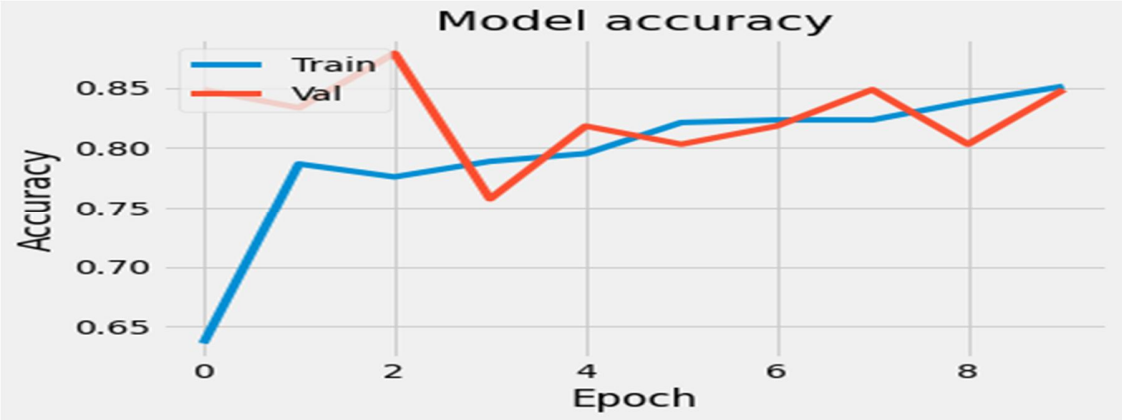

Figure 5.1: Plot shows accuracy for training and validation for 10 epochs without cross validation

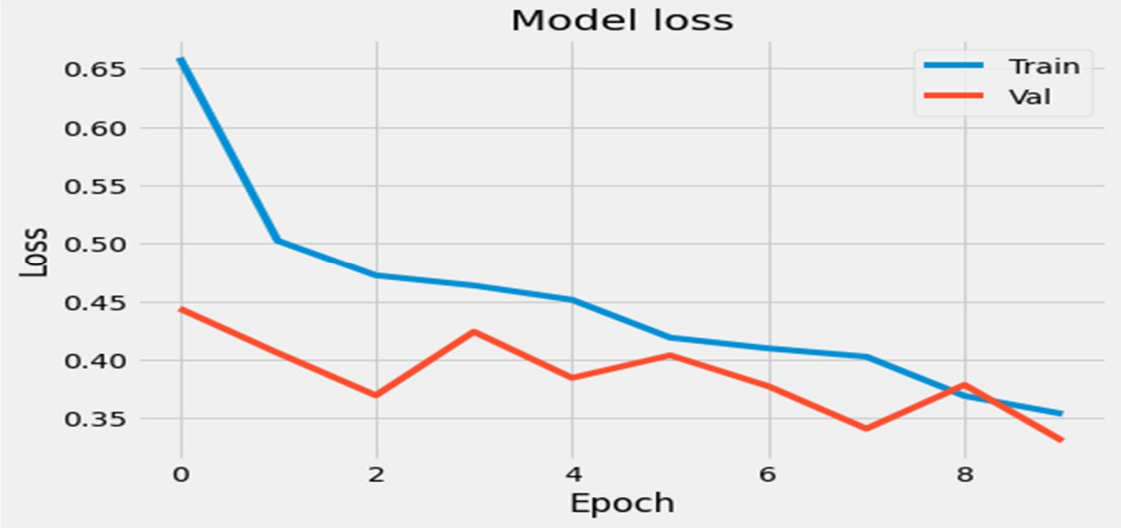


Figure 5.2: Plot shows loss for training and validation for 10 epochs without cross validation

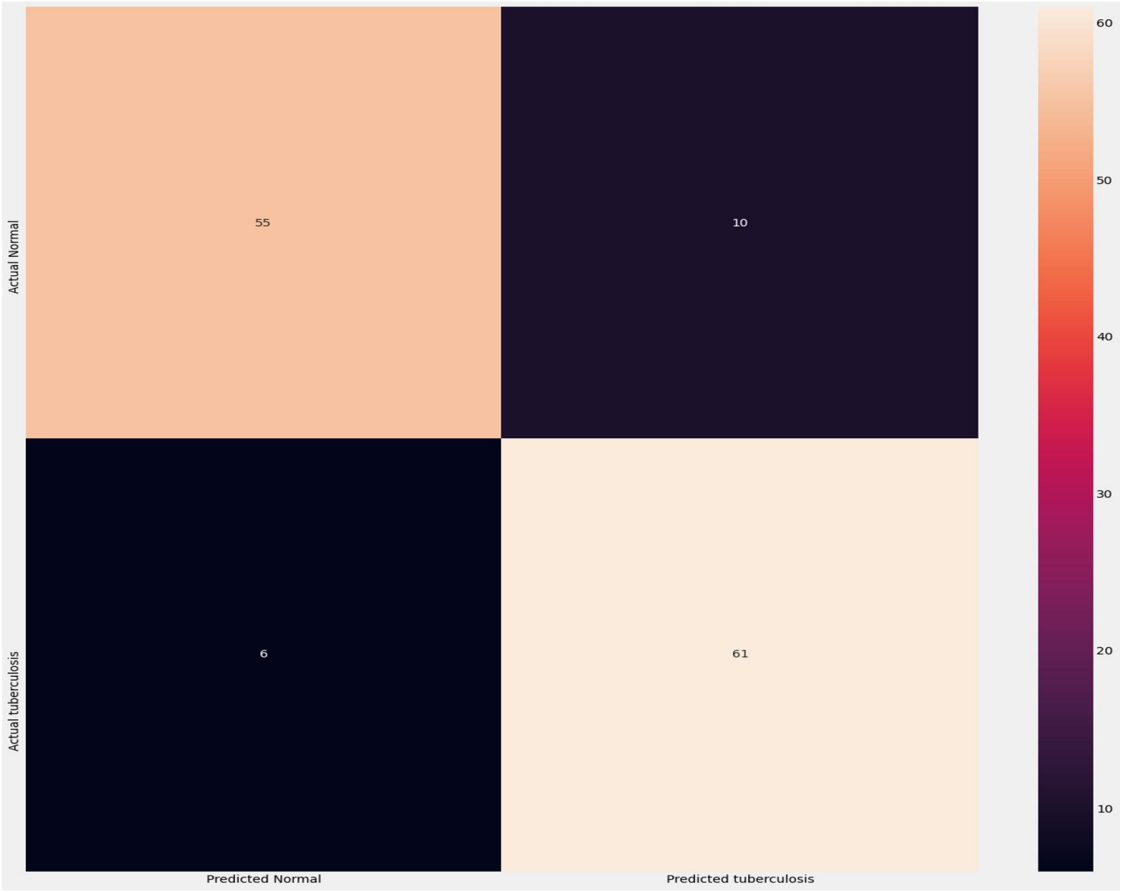

Figure 5.3: Confusion matrix for image classification without cross validation (10 epochs)

From confusion matrix, **TP**=61, **TN**=55, **FP**=10, **FN**=6.

Sensitivity (Recall) = $\frac{\text{TP}}{(\text{TP+FN})}$ x 100 =$\frac{61}{(61+6)}$ x 100= **91.04%**

Specificity = $\frac{\text{TN}}{(\text{TN+FP})}$ x100 =$\frac{55}{(55+10)}$ x 100 =**84.61%**

Accuracy = $\frac{(TP+TN)}{(TP+FN)+(FP+TN)}$ x100 = $\frac{(61+55)}{(61+6)+(10+55)}$ x 100

= **87.87%**

Precision = $\frac{\text{TP}}{(\text{TP+FP})}$ x100 = $\frac{61}{(61+10)}$ x100 = **85.91%**

F1-Score = $\frac{2*(\text{Precision} * \text{Sensitivity})}{(\text{Precision} + \text{Sensitivity})}$ = $\frac{2*(.8591 * .9104)}{(.8591 + .9104)}$

= **0.88**

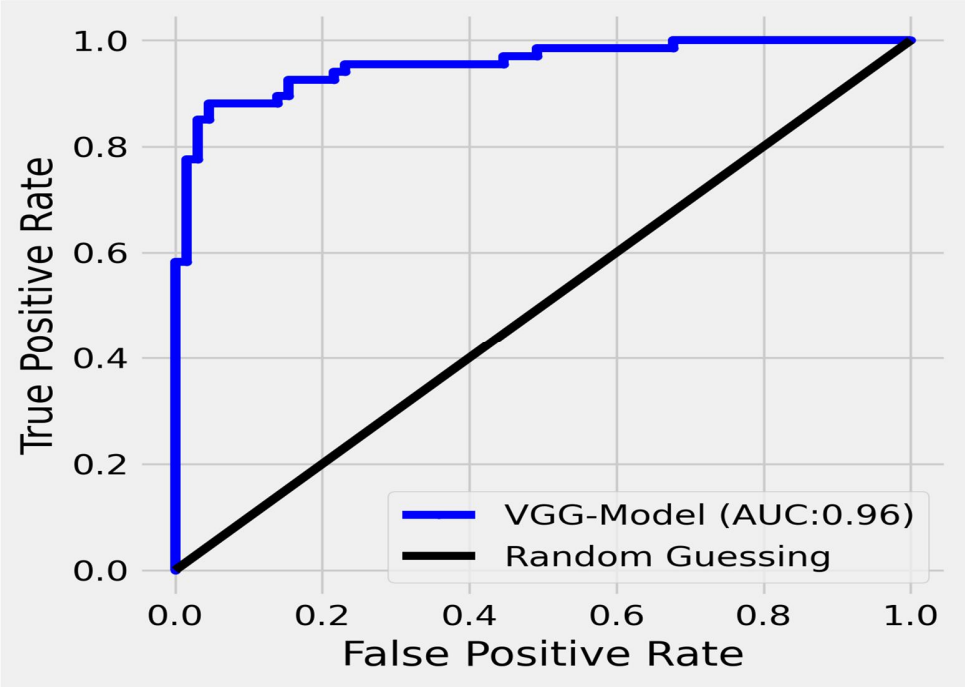


Figure 5.4: AUC of ROC for image classification without cross validation (10 epochs)

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| normal | 0.90 | 0.85 | 0.87 | 65 |
| tuberculosis | 0.86 | 0.91 | 0.88 | 67 |
| accuracy | | | 0.88 | 132 |
| macro avg | 0.88 | 0.88 | 0.88 | 132 |
| weighted avg | 0.88 | 0.88 | 0.88 | 132 |

Figure 5.5: Classification report for image classification without cross validation (10 epochs)

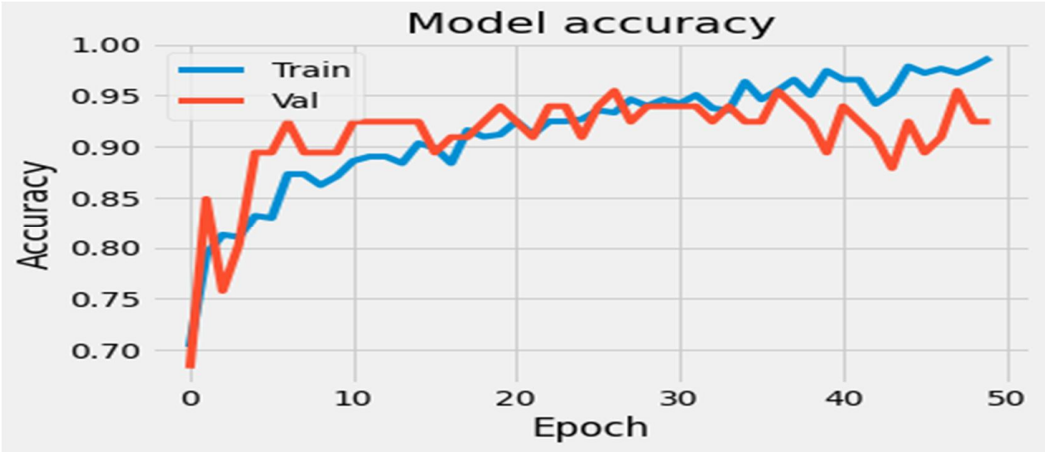


Figure 5.6: Plot shows accuracy for training and validation for 50 epochs without use of cross validation


Figure 5.7: plot shows loss for training and validation for 50 epochs without use of cross validation


Figure 5.8: Confusion matrix for image classification without cross validation (50 epochs)

From confusion matrix, **TP**=63, **TN**=59, **FP**=6, **FN**=4.

Sensitivity (Recall) = $\frac{\text{TP}}{(\text{TP+FN})}$ x 100 =$\frac{63}{(63+4)}$ x 100= **94.02%**

Specificity = $\frac{\text{TN}}{(\text{TN+FP})}$ x100 =$\frac{59}{(59+6)}$ x 100 =**90.76%**

Accuracy = $\frac{(TP+TN)}{(TP+FN)+(FP+TN)}$ x100 = $\frac{(63+59)}{(63+4)+(6+59)}$ x 100

= **92.42%**

Precision = $\frac{\text{TP}}{(\text{TP+FP})}$ x100 = $\frac{63}{(63+6)}$ x100 = **91.30%**

F1-Score = $\frac{2*(\text{Precision} * \text{Sensitivity})}{(\text{Precision} + \text{Sensitivity})} = \frac{2*(.9130 * .9402)}{(.9130 + .9402)}$

= **0.9264 ≈ 0.93**

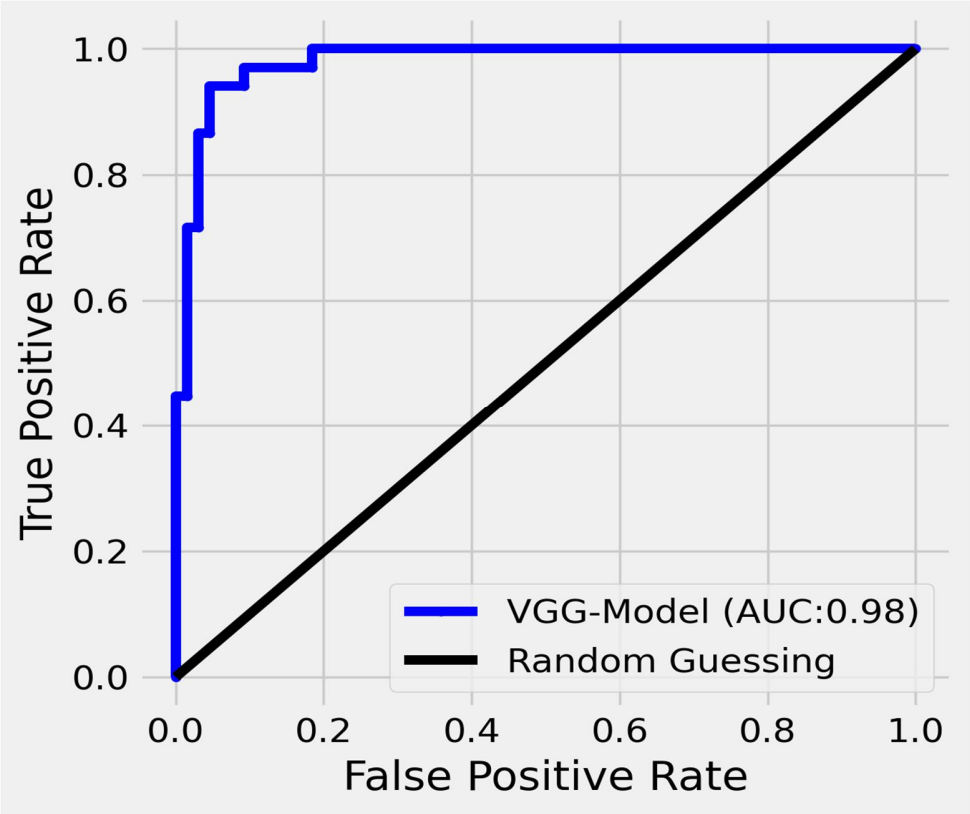


Figure 5.9: AUC of ROC for image classification without cross validation (50 epochs)

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| normal | 0.94 | 0.91 | 0.92 | 65 |
| tuberculosis | 0.91 | 0.94 | 0.93 | 67 |
| accuracy | | | 0.92 | 132 |
| macro avg | 0.92 | 0.92 | 0.92 | 132 |
| weighted avg | 0.92 | 0.92 | 0.92 | 132 |

Figure 5.10: Classification report for image classification without cross validation (50 epochs)

## 5.3 Performance Evaluation of image Classification with 5 folds cross validation

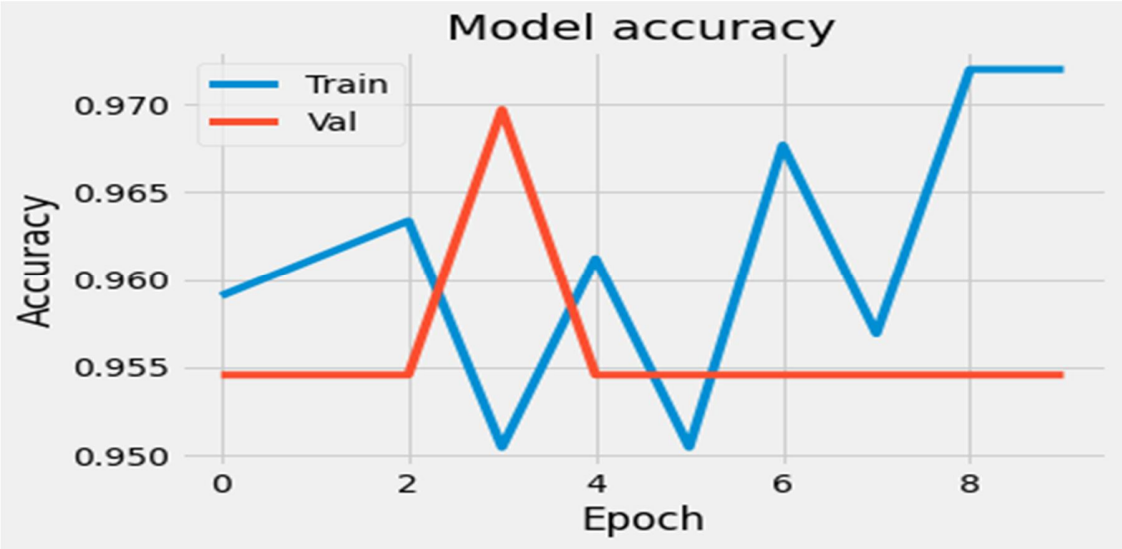


Figure 5.11: Plot shows accuracy for training and validation for last 10 epochs with use of 5 fold cross validation


Figure 5.12: Plot shows loss for training and validation for last 10 epochs with use of 5 fold cross validation

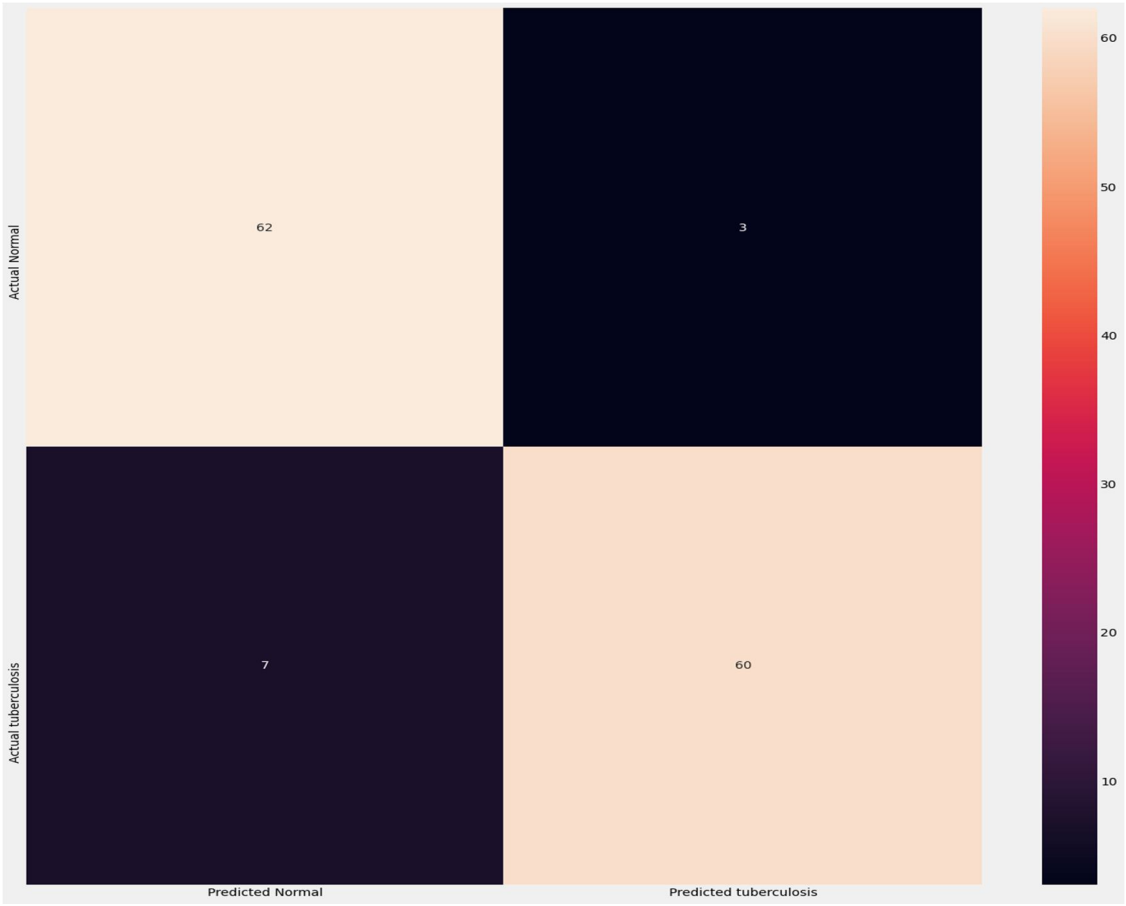


Figure 5.13: Confusion matrix of image classification for 5 folds cross validation

From confusion matrix, **TP**=60, **TN**=62, **FP**=3, **FN**=7.

Sensitivity (Recall) = $\frac{\text{TP}}{(\text{TP+FN})}$ x 100 =$\frac{60}{(60+7)}$ x 100= **89.55%**

Specificity = $\frac{\text{TN}}{(\text{TN+FP})}$ x100 =$\frac{62}{(62+3)}$ x 100 =**95.38%**

Accuracy = $\frac{(TP+TN)}{(TP+FN)+(FP+TN)}$ x100 = $\frac{(60+62)}{(60+7)+(3+62)}$ x 100

= **92.42%**

Precision = $\frac{\text{TP}}{(\text{TP+FP})}$ x100 = $\frac{60}{(60+4)}$ x100 = **95.23%**

F1-Score = $\frac{2*(\text{Precision} * \text{Sensitivity})}{(\text{Precision} + \text{Sensitivity})} = \frac{2*(.9523 * .8955)}{(.9523 + .8955)}$

= **0.9230**

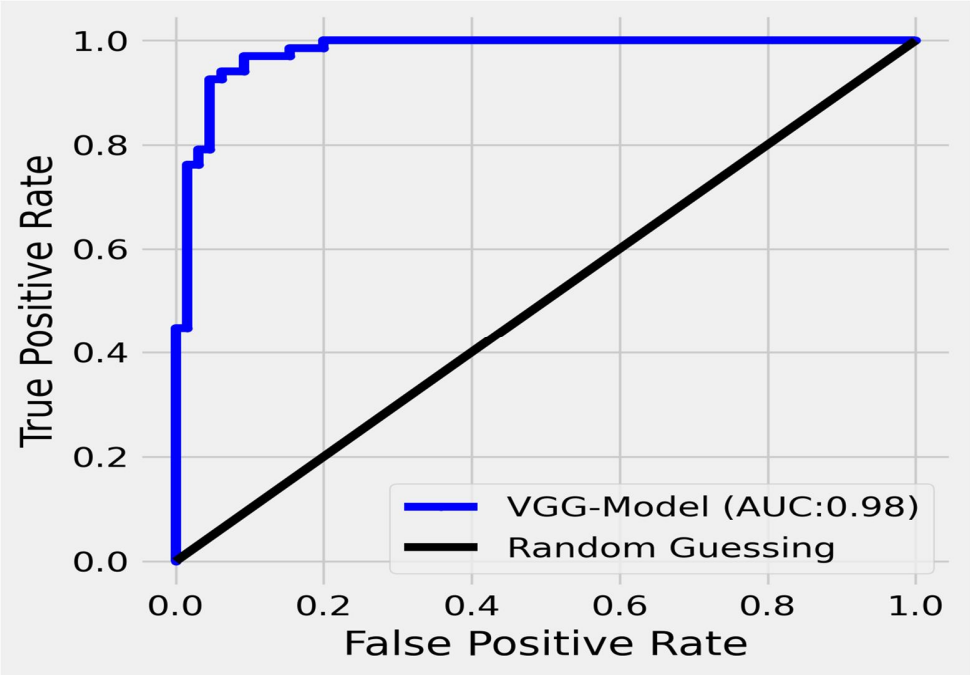


Figure 5.14: AUC of ROC for image Classification for 5 folds cross validation

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| normal | 0.90 | 0.95 | 0.93 | 65 |
| tuberculosis | 0.95 | 0.90 | 0.92 | 67 |
| accuracy | | | 0.92 | 132 |
| macro avg | 0.93 | 0.92 | 0.92 | 132 |
| weighted avg | 0.93 | 0.92 | 0.92 | 132 |

Figure 5.15: Classification reports of image classification for 5 folds cross validation

The classification report visualizer displays the precision, recall, F1, and support scores for the model. All of these parameters are defined above except support which is the number of actual occurrences of the class in the specified dataset.

Although the proposed model gives the same result for accuracy when using of 50 epochs with or without using of cross validation, it gives better result of sensitivity and specificity without using of cross validation ; therefor the final results that will be considered is the results of training the proposed model for 50 epochs without using of cross validation

Table 5.1: Comparison between the proposed model and some of literature reviews

| Authors, Year | Dataset | Features / Parameter | Method | Classification Accuracy (%) | Area under the curve AUC | Sensitivity (%) | Specificity (%) |
|---|---|---|---|---|---|---|---|
| Hwang et al,2016 | Private data , Shenzhen(SH) and Montgomery County (MC) | CXR | Transfer learning of Modified Alexnet | 90<br>83.7<br>67.4 | 0.96<br>0.926<br>0.884 | - | - |
| Syeda Meraj et al ,2019 | Montgomery County (MC), Shenzhen (SH) and Indiana University chest X-Ray dataset | CXR | VGG-16<br>VGG-19<br>ResNet50<br>GoogLenet | 86.74<br>84.33<br>81.92<br>80.72 | - | -<br>-<br>-<br>- | -<br>-<br>-<br>- |
| Pasa et al. 2019 | Montgomery County (MC), Shenzhen (SH) and Belarus Dataset | CXR | Optimized CNN | 79.0 for MC, 84.4 for SH and 86.2 for combined | 0.811 for MC, 0.9 for SH and 0.925 for combined | - | - |
| Seelwan Sathitratanacheewin et al ,2020 | Shenzhen Hospital Dataset and NIH ChestX-ray8 | CXR | CNN | - | - | 72 | 82 |
| The proposed model ,2021 | Shenzhen (SH) Dataset | CXR | Transfer Learning (VGG 16 model) | 92.42 | 0.98 | 94.02 | 90.76 |

Regarding the Shenzhen dataset the proposed model performed an accuracy reached to 92% compared to Hwang et al with an accuracy of 83.7% and 84.4% reached by Pasa et al. and when considering the application of VGG-16 Syeda Meraj et al suggested an accuracy of 86.74% applied in a combined dataset of Montgomery County (MC), Shenzhen (SH) and Indiana University chest X-Ray dataset; and by comparing Seelwan Sathitratanacheewin et al Sensitivity 72% and specificity 82% the proposed model reached to 94.02% and 90.76%, while The proposed model AUC was 0.98 compared to 0.96,0.926,0.884, 0.811 for MC, 0.9 for SH and 0.925 for combined mentioned in the table above.

# Chapter Six

# Conclusion and Recommendation

## 6.1 Conclusion

For decades, tuberculosis, a potentially serious infectious lung disease, continues to be a leading cause of worldwide death. Proven to be conveniently efficient and cost-effective, chest X-ray (CXR) has become the preliminary medical imaging tool for detecting TB. Detection model was proposed for tuberculosis, the proposed model a transfer learning approach with deep Convolutional Neural Networks, and it presented a ConvNet model that uses VGG16 for classifying CXR images to identify patients suffering from TB. Previous research on CXR classification applied complex models for lung segmentation before training the model. The proposed model shows that VGG16 can use the raw data to classify the results with comparable accuracy reached to 92%.

## 6.2 Recommendation

.

Recommendation would include running the model in:

1. Hardware with higher specification to enable increasing of input image size, decrease the proposed model running time and train the proposed model with dataset having greater number of images.
2. Website application and establishing graphical user interface (GUI) link the model to CXR images entered by users.

# References

[1] J. F. Murray, “Mycobacterium tuberculosis and the cause of consumption: from discovery to the fact,” American journal of respiratory and critical care medicine, 2004.

[2] Syeda Shaizadi Meraj, Razali Yaakob, Azreen Azman, Siti Nuralain Mohd Rum, Azree Shahrel Ahmad Nazri, Artificial Intelligence in Diagnosing Tuberculosis: A Review,2019.

[3] S. Hwang, H.-E. Kim, J. Jeong, H.-J. Kim, A novel approach for tuberculosis screening based on deep convolutional neural networks, in: SPIE Medical Imaging, International Society for Optics and Photonics, 2016.

[4] Chang Liu, Yu Cao, Marlon Alcantara, Benyuan Liu, Maria Brunette, Jesus Peinado ,Walter Curioso, TX-CNN: Detecting Tuberculosis In Chest X-Ray Images Using Convolutional Neural Network,2017.

[5] Yan Xiong, Xiaojun Ba, Ao Hou, Kaiwen Zhang, Longsen Chen, Ting Li **,**Automatic detection of mycobacterium tuberculosis using artificial intelligence,2018.

[6] F. Pasa1, V. Golkov, F. Pfeiffer, D. Cremers & D. Pfeiffer, Efficient Deep Network Architectures for Fast Chest X-Ray Tuberculosis Screening and Visualization, 2019.

[7] Mustapha Oloko-Oba and Serestina Viriri, Diagnosing Tuberculosis Using Deep Convovgflutional Neural Network, 2019.

[8] Ahmed T. Sahlol ,Mohamed Abd Elaziz, Amani Tariq Jamal, Robertas Damaševi˘cius , and Osama Farouk Hassan ,A Novel Method for Detection of Tuberculosis in Chest Radiographs Using Artificial Ecosystem-Based Optimisation of Deep Neural Network Features, 2020.

[9] Seelwan Sathitratanacheewin, Panasun Sunanta, Krit Pongpirul, Deep learning for automated classification of tuberculosis-related chest X-Ray: dataset distribution shift limits diagnostic performance generalizability, 2020.

[10] B. Herzog *et al* , “History of tuberculosis,” Respiration, 1998.

[11] T. M. Daniel, “The history of tuberculosis,” Respiratory medicine, 2006.

[12] S. Grzybowski and E. A. Allen, “Tuberculosis: 2. History of the disease in Canada,” CMAJ: Canadian Medical Association Journal, 1999.

[13] K. Kumar, “Spinal tuberculosis, the natural history of the disease, classifications, and principles of management with a historical perspective,” European Journal of Orthopaedic Surgery & Traumatology , 2016.
[14] Attaway, S. (2013). Matlab: A Practical Introduction to Programming and Problem Solving. Amsterdam: Elsevier, CDC Tuberculosis (TB). Disease: Symptoms and Risk Factors Features CDC, 2019.
[15] M. Purohit and T. Mustafa, “Laboratory diagnosis of extrapulmonary tuberculosis (EPTB) in a resource-constrained setting: state of the art, challenges and the need,” Journal of clinical and diagnostic research: JCDR, vol. 9, 2015.
[16] L. A. Dalvin and W. M. Smith, “Intraocular Manifestations of Mycobacterium tuberculosis: A Review of the Literature,” Journal of Clinical Tuberculosis and Other Mycobacterial Diseases, 2017.
[17] M. Häggström, “Medical gallery of Mikael Häggström 2014.Wikiversity Journal of Medicine 1, 2014.
[18] Payal Dande, Purva Samant , Acquaintance to Artificial Neural Networks and use of artificial intelligence as a diagnostic tool for tuberculosis: A review,2017
[19] Daley C, Gotway M, Jasmer R. Radiographic manifestations of tuberculosis. A Primer for Clinicians. San Francisco, CA: Curry International Tuberculosis Center, 2009.
[20] P. J. Hayes and L. Morgenstern, “On John McCarthy’s 80th birthday, in honor of his contributions,” AI Magazine, vol. 28, 2007.
[21] B. G. Buchanan, “A (very) brief history of artificial intelligence,” AI Magazine, vol. 26, 2005.
[22] J. N. Kok, E. Boers, W. A. Kosters, P. Van der Putten, and M. Poel, “Artificial intelligence: definition, trends, techniques, and cases,” Artif Intell, vol. 1, 2009.
[23] Artificial intelligence - neural networks, Available at: www.tutorialspoint .com/artificial_intelligence/artificial_intelligence_neural_networks.html.
[24] Al-Shayea QeetharaKadhim, Artificial neural networks in medical diagnosis. Int J Comp Sci Appl ,2011.
[25]Alberto López-Rodríguez , Eladia M Peña-Méndez , Petr Vaňhara Masaryk Josef Havel Masaryk Artificial neural networks in medical diagnosis Article in Journal of Applied Biomedicine, 2013 .
[26] Er O, Temurtas F, Tanrıkulu A., Tuberculosis Disease Diagnosis Using Artificial Neural Networks, 2008.
[27] LeCun Y, Bengio Y, Hinton G () Deep learning Nature 521: 436–444, 2015.

Krizhevsky A, Sutskever I, Hinton GE, ImageNet classification with deep convolutional neural networks, 2018.
[28] Nair V, Hinton GE, Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning, 2010.
[29] Ramachandran P, Zoph B, Le QV ,Searching for activation functions, 2017.
[30] Glorot X, Bordes A, Bengio Y, Deep sparse rectifier neural networks. In: Proceedings of the 14th International Conference on Artificial Intelligence and Statistics, 2011.
[31] Lin M, Chen Q, Yan S , Network in network, 2013.
[32] Sinno Jialin Pan and Qiang Yang Fellow IEEE, A Survey on Transfer Learning, 2011.
[33] W. Dai, Q. Yang, G. Xue, and Y. Yu, "Boosting for transfer learning," in Proceedings of the 24th International Conference on Machine Learning, Corvalis, Oregon, USA, 2007.
[34] Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov RR Improving neural networks by preventing co-adaptation of feature detectors, 2012.
[35] Krizhevsky A, Sutskever I, Hinton GE ImageNet classification with deep convolutional neural networks, 2012.
[36] Simonyan K, Zisserman A Very deep convolutional networks for large-scale image recognition, 2015.
[37] He K, Zhang X, Ren S, Sun J Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
[38] Szegedy C, Liu W, Jia Y et al, Going deeper with convolutions. In: IEEE Conference on Computer Vision and Pattern Recognition, 2015.
[39] Huang G, Liu Z, van der Maaten L, Weinberger KQ Densely connected convolutional networks. In: IEEE Conference on Computer Vision and Pattern Recognition, 2017.
[40] Zeiler MD, Fergus R, Visualizing and understanding convolutional networks. In: Proceedings of Computer Vision – ECCV, 2014.
[41] Yosinski J, Clune J, Bengio Y, Lipson H, How transferable are features in deep neural networks, 2014.
[42] Karen Simonyan , Andrew Zisserman, Very Deep Convolutional Networks For Large-Scale Image Recognition, Conference paper at ICLR 2015.
[43] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. 2009.

# Appendix: Model Code

```
# -*- coding: utf-8 -*-
"""
Created on Sat Nov 21 22:37:18 2020

@author: toshiba
"""

# General libraries
import os
import numpy as np
import pandas as pd
import random
import cv2
import matplotlib.pyplot as plt

# Deep learning libraries
from keras import layers
from keras.models import Model
from keras.optimizers import Adam
from keras.callbacks import ModelCheckpoint, ReduceLROnPlateau,
EarlyStopping
plt.style.use('fivethirtyeight')
```

```
img_height = 96
img_width = 96
batch_size = 4
no_of_epochs  = 10



from keras.applications.vgg16 import VGG16

base_model = VGG16(input_shape = (96, 96, 3), # Shape of our images
include_top = False, # Leave out the last fully connected layer
weights = 'imagenet')
for layer in base_model.layers:
    layer.trainable = False
# Flatten the output layer to 1 dimension
x = layers.Flatten()(base_model.output)
# Add a fully connected layer with 512 hidden units and ReLU activation
x = layers.Dense(512, activation='relu')(x)
# Add a dropout rate of 0.5
x = layers.Dropout(0.5)(x)
# Add a final sigmoid layer for classification
x = layers.Dense(1, activation='sigmoid')(x)
```

```
cnn = Model(inputs=base_model.input, outputs=x)

# Creating model and compiling

opt = Adam(lr=0.0001, decay=1e-6)

cnn.compile(optimizer = opt, loss = 'binary_crossentropy', metrics = ['accuracy'])


# Callbacks

checkpoint = ModelCheckpoint(filepath='best_weights.hdf5',
save_best_only=True, save_weights_only=True)

lr_reduce= ReduceLROnPlateau(monitor='val_loss', factor=0.3, patience=2,
verbose=2, mode='max')

early_stop = EarlyStopping(monitor='val_loss', min_delta=0.1, patience=1,
mode='min')


from tqdm import tqdm

import matplotlib.pyplot as plt


DATADIR = 'F:/shenzhen splitted/train and val'

data=[]

for img in tqdm(os.listdir(DATADIR)):

    try:

        img_array = cv2.imread(os.path.join(DATADIR,img))

        img_array = cv2.resize(img_array, (img_height, img_width))

        img_array = cv2.cvtColor(img_array, cv2.COLOR_BGR2RGB)

        img_array = img_array.astype(np.float32)/255.
```

```
        if img[-5]=='0':
            data.append([img_array, 0])
        else:
            data.append([img_array, 1])
    except Exception as e:
            pass
print(len(data))



random.shuffle(data)
for sample in data[:10]:
    print(sample[1])

X = []
y = []

for features,label in data:
    X.append(features)
    y.append(label)

X = np.array(X).reshape(-1, img_width, img_height, 3)
print(X.shape)
#5 Fold Cross Validation
```

```
k=5

num_validation_samples=66

validation_scores=[]

for fold in range(k):


validation_data=X[num_validation_samples*fold:num_validation_samples*(fold+
1)]


validation_labels=y[num_validation_samples*fold:num_validation_samples*(fold
+1)]
   if fold==0:

      training_data=X[num_validation_samples*(fold+1):]

      training_labels=y[num_validation_samples*(fold+1):]

   else:

      training_data=np.append(X[:num_validation_samples*fold],
X[num_validation_samples*(fold+1):],axis=0)

      training_labels=np.append(y[:num_validation_samples*fold],
y[num_validation_samples*(fold+1):],axis=0)


   cnn.history=
cnn.fit(training_data,training_labels,batch_size=batch_size,epochs=no_of_epochs
,validation_data=(validation_data,validation_labels)) # 50 epochs per model

   validation_score=cnn.evaluate(validation_data,validation_labels)

   validation_scores.append(validation_score[1])
```

```
#Average Validation Score
print('Average Validation Score: ', np.average(validation_scores)*100,'%')
#Visualize the models accuracy
plt.plot(cnn.history.history['accuracy'])
plt.plot(cnn.history.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='upper left')
plt.show()
#Visualize the models loss
plt.figure(figsize=(7.3,5))
plt.plot(cnn.history.history['loss'])
plt.plot(cnn.history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='upper right')
plt.show()
from tqdm import tqdm
import matplotlib.pyplot as plt
import cv2
DATADIR1 = 'F:/shenzhen splitted/test'
```

```
data1=[]
for img in tqdm(os.listdir(DATADIR1)):
    try:
        img_array1 = cv2.imread(os.path.join(DATADIR1,img))
        img_array1 = cv2.resize(img_array1, (img_height, img_width))
        img_array1 = cv2.cvtColor(img_array1, cv2.COLOR_BGR2RGB)
        img_array1 = img_array1.astype(np.float32)/255.
        if img[-5]=='0':
            data1.append([img_array1, 0])
        else:
            data1.append([img_array1, 1])
    except Exception as e:
            pass
print(len(data1))

import random

random.shuffle(data1)
for sample in data1[:10]:
    print(sample[1])

X1 = []
y1 = []
```

```
for features,label in data1:
    X1.append(features)
    y1.append(label)

X1 = np.array(X1).reshape(-1, img_width, img_height, 3)
print(X1.shape)
preds = cnn.predict(X1)
predictions = preds.copy()
predictions[predictions <= 0.5] = 0
predictions[predictions > 0.5] = 1

from sklearn.metrics import classification_report,confusion_matrix
cm = pd.DataFrame(data=confusion_matrix(y1, predictions, labels=[0,
1]),index=["Actual Normal", "Actual tuberculosis"],
columns=["Predicted Normal", "Predicted tuberculosis"])
import seaborn as sns
plt.figure(figsize=(20,20))
sns.heatmap(cm,annot=True,fmt="d")
print(classification_report(y_true=y1,y_pred=predictions,target_names
=['normal','tuberculosis']))
from sklearn.metrics import roc_curve, roc_auc_score
fpr, tpr, _ = roc_curve(y1, preds)
```

```
fig, ax1 = plt.subplots(1,1, figsize = (5, 5), dpi = 250)

ax1.plot(fpr, tpr, 'b.-', label = 'VGG-Model (AUC:%2.2f)' % roc_auc_score(y1, preds))

ax1.plot(fpr, fpr, 'k-', label = 'Random Guessing')

ax1.legend(loc = 4)

ax1.set_xlabel('False Positive Rate')

ax1.set_ylabel('True Positive Rate');

fig.savefig('roc.pdf')
```