

M.Tech (Bioinformatics)

TITIKSHA SHARMA

2023

NEURAL NETWORK-ASSISTED DETECTION OF PRIMARY TUBERCULOSIS FOR IMPROVED DIAGNOSIS

A DISSERTATION

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE AWARD OF THE DEGREE

OF

MASTER OF TECHNOLOGY

In

BIOINFORMATICS

Submitted by

TITIKSHA SHARMA

[2K21/BIO/08]



Under the supervision of

Dr. Asmita Das

Department of Biotechnology

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of
Engineering)

Bawana road, Delhi-110042

MAY, 2023

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

CANDIDATE'S DECLARATION

I, Titiksha Sharma (2k21/BIO/08) of M.Tech (Bioinformatics), hereby certify that the work which is being submitted in this major project report entitled “**Neural Network-assisted detection of primary tuberculosis for improved diagnosis**” which is submitted by me to the Department of Biotechnology, Delhi Technological University, Delhi in the partial fulfillment of the requirement for the award of the degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Place: Delhi

Date : 31/05/2023


Titiksha Sharma
(2K21/BIO/08)

DEPARTMENT OF BIOTECHNOLOGY
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

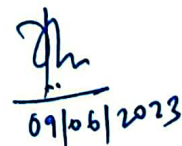
CERTIFICATE

I hereby certify that the project Dissertation titled “Convolutional Neural Network-assisted detection of Primary Tuberculosis for improved diagnosis” which is submitted by Titiksha Sharma, 2K21/BIO/08 to the Department of Biotechnology, Delhi Technological University, Delhi in the partial fulfillment of the requirement for the award of the degree of Master of Technology, is record of the project work carried out by the students under my supervision. To the best of my knowledge this work has not been submitted in part for any Degree or Diploma to this University or elsewhere.

Place: Delhi
Date: 9/06/23



Dr. Asmita Das
(SUPERVISOR)



09/06/2023

Prof. Pravir Kumar
(Head of the Department)

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to Dr. Asmita Das, Assistant Professor at DTU, Delhi, for granting me the opportunity to work on this project under her guidance. I am truly thankful for her mentorship, advice, and support throughout my project work.

I would also like to extend my deep appreciation to Prof. Pravir Kumar, the Head of the Department of Biotechnology at DTU, for providing me with the privilege to pursue my studies and work in this esteemed institution.

Furthermore, I would like to acknowledge and thank my seniors, Ph.D. scholars, classmates, faculty members, and staff for their invaluable guidance and support. Their assistance has played a crucial role in my achievement of positive outcomes in my work.

Titiksha

TITIKSHA SHARMA
(2K21/BIO/08)

ABSTRACT

The aim of the study is to develop a predictive model for Tuberculosis (Tb), a serious chronic infectious disease. Tb (caused by a bacterial *Mycobacterium tuberculosis*) causes enormous global health issues, and early detection is critical for initiating treatment on time. Leveraging the power of Convolutional Neural Networks (CNN), a deep learning model, a robust system was constructed to predict Tb status based on chest X-ray images. Traditional diagnostic methods often involve time-consuming laboratory tests, necessitating the need for more efficient and accessible approaches. The proposed CNN model was trained using a large dataset of annotated chest X-ray images, enabling it to learn relevant features indicative of Tb infection. Extensive evaluation and validation confirmed the model's high accuracy and reliability in diagnosing Tb, thereby providing a valuable tool for healthcare professionals. By revolutionizing medical image analysis, these models have the power to transform healthcare delivery, leading to better patient outcomes, optimized resource allocation, and significant advancements in the field. Continued research, collaboration, and implementation are crucial to fully harness the future potential of deep learning models in clinical practice.

Table of Content

Declaration	ii
Certificate	iii
Acknowledgement	iv
Abstract	v
List of Figures	viii-ix
List of Tables	x
List of Abbreviations	xi
CHAPTER 1 INTRODUCTION	1-3
CHAPTER 2 REVIEW OF LITERATURE	4
2.1 Tuberculosis	4
2.2 Mechanisms: Host Immune System Manipulation by <i>Mycobacterium tuberculosis</i>	6
2.3 Transmission of Tuberculosis	7
2.4 Role of X-ray in Tb detection	8
2.5 Machine Learning and steps of building a machine learning model	10
2.6 Image detection and analysis	15
2.7 Orange Data Mining tool	17
2.7.1 Orange Widgets	17
CHAPTER 3 METHODOLOGY	18
3.1 Data Collection	18
3.2 Data Validation and Cleaning	19

3.3 Orange data mining tool for the prediction of best algorithm	19
3.4 Uploading the data on Google Collab	19
3.5 Further data pre-processing & splitting of data into test and train dataset	20
3.6 Deep Learning Model Training	21
3.7 Performance Evaluation	22
3.8 Model deployment & Integration	23
3.9 Prediction & threshold setting for classification	24
CHAPTER 4 RESULT AND DISCUSSION	25
4.1 Data Collection, Pre-processing and Model building	25
4.2 Model Evaluation	26
4.3 Result and Validation of Model performance by testing it with help of the Data which was not used to train the model	29
CHAPTER 5 CONCLUSION	30
Appendix	31-35
References	36-39

LIST OF FIGURES

Figure	Description	Page no.
FIGURE 1	Basic steps involved in Machine Learning	2
FIGURE 2.2.1	The schematic diagram illustration of <i>Mycobacterium tuberculosis</i> (MTb) infection cycle	7
FIGURE 2.5.1	Schematic representation of Neural Network	14
FIGURE 2.6.1	Schematic representation of a CNN with two hidden layers	16
FIGURE 2.7.1	Orange workflow and widgets	17
FIGURE 3.1.1	An X-ray visualization of (a) Normal Patient (Tb negative) (b) Tuberculosis Patient (Tb Positive)	18
FIGURE 3.4.1	Uploaded X-Ray images from desktop for training the Model	19
FIGURE 3.4.2	Uploaded X-Ray images from Kaggle for training the Model	20
FIGURE 3.5.1	Data pre-processing and splitting the data into train and test	21
FIGURE 3.6.1	Code for building Convolutional layers	22
FIGURE 3.7.1	This section of code evaluates the built Model	23
FIGURE 3.8.1	The section of code shown in the picture is to save the model in tflite format and further testing it using a picture which was not used to train the model	24
FIGURE 3.9.1	The section of code shown in the picture is to predict if new the test image is Tb Positive or Tb negative	24
FIGURE 4.1.1	A schematic workflow built on Orange Data	25

mining tool for find the best model for this study

FIGURE 4.1.2	The confusion matrix of CN2 rule inducer, SVM, kNN and Neural Network	26
FIGURE 4.2.1	The picture shows that 662 files were uploaded	27
FIGURE 4.2.2	The accuracy of this model was 78.36%	27
FIGURE 4.2.3	The picture shows that 3168 files were uploaded	28
FIGURE 4.2.4	The accuracy of the model significantly improved to 96.11%.	<u>28</u>

LIST OF TABLES

Table no.	Description	Page no.
TABLE 1	Stages of Tb and symptoms in each stage	8
TABLE 2	Difference in Machine Learning and Deep	14
TABLE 3	The table displays the model predictions for a	31

LIST OF ABBREVIATION

Tb	Tuberculosis
MTb	Mycobacterium tuberculosis
ML	Machine Learning
DL	Deep Learning
CNN	Convolutional Neural Networks
MDR	Multidrug Resistent Tb
XDR	Extensively drug- resistant Tb

CHAPTER 1

INTRODUCTION

Tuberculosis (Tb) is an infectious respiratory disease caused by *Mycobacterium tuberculosis* (MTb) ^[1]. It is a global health concern, significantly impacting morbidity and mortality worldwide ^{[1][2]}. Tb is primarily transmitted through inhaling respiratory droplets containing MTb, allowing the bacteria to enter the lungs and initiate infection ^[2]. Once inside the host, MTb faces a complex interplay with the immune system, leading to diverse clinical outcomes and disease manifestations. MTb infection begins when the inhaled bacteria reach the alveolar spaces of the lungs. The bacteria are taken up by resident alveolar macrophages, the primary immune cells in the lungs, triggering the initial immune response ^[2]. Alveolar macrophages recognize MTb and attempt to eliminate the pathogen through various defense mechanisms. However, MTb has evolved several strategies to evade and manipulate the immune system, allowing it to establish infection and persist within host cells ^[3].

According to the WHO Global Tuberculosis Report 2022, there has been a concerning global increase in tuberculosis (Tb) cases in 2021, with 10.6 million new cases reported, marking a 4.5% rise compared to the previous year. This reversal of the long-standing decline in Tb cases is attributed to the disruptions caused by the COVID-19 pandemic. The India Tb Report 2022, released in mid-May 2023, reveals similar stats in India; the incidence rate of all forms of Tb in 2020 was 188 per 100,000 population, ranging from 129 to 257 per 100,000 population. Moreover, the report highlights a 19% increase in the total number of newly notified Tb cases in 2021 (1,933,381) compared to 2020 (1,628,161). Although drugs such as Bedaquiline and Delamanid have demonstrated effectiveness, there are various limitations associated with their use. These limitations include the need for prolonged treatment duration, potential drug toxicity, and the emergence of drug-resistant strains ^[5]. The existing Tb treatment regimens are characterized by their long duration and the requirement for

strict adherence, which can pose challenges for patients to sustain ^[6]. Despite a relatively low proportion of delay compared to previous studies, enhancing strategies for early detection, diagnosis, and treatment of tuberculosis (Tb) infection remains crucial. Strengthening these measures is necessary to minimize complications associated with Tb and prevent further spread of the disease ^[4].

Traditional methods of tuberculosis (Tb) diagnosis, such as sputum microscopy and culture, have limitations in accuracy, speed, and accessibility ^[7]. These methods often require specialized laboratories, skilled technicians, and time-consuming procedures, leading to delays in diagnosis and treatment initiation; due to their low sensitivity, false-negative results might also be generated, resulting in delayed response ^[8]. In recent years, machine learning techniques have shown promise in improving tuberculosis (Tb) detection accuracy and efficiency. By utilizing large datasets encompassing clinical data, laboratory findings, and imaging data like chest X-rays, researchers have successfully trained machine learning models to identify Tb cases with high precision.

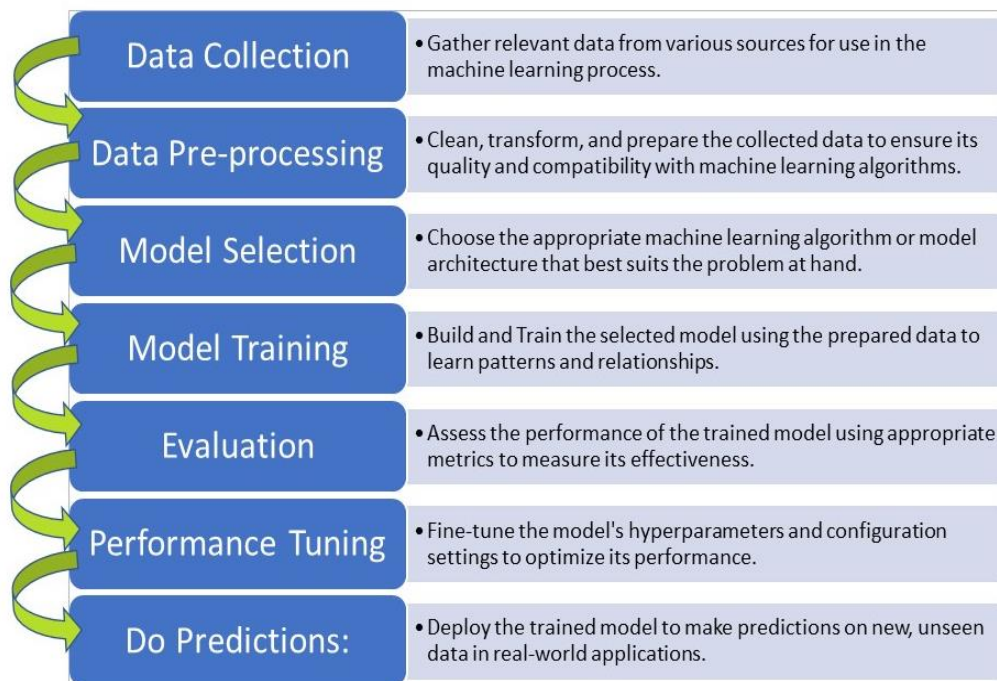


FIGURE 1: Basic steps involved in Machine Learning

The objective of this project is to utilize machine learning tool and techniques for the detection of Primary Tuberculosis (Tb) using X-ray images. The goal is to develop a predictive model that can accurately identify Tb cases, enabling healthcare professionals to efficiently predict Tb and deliver improved patient care. The project aims to find the most efficient and accurate Machine Learning algorithm with the help of Orange Data mining tool and then building the model using Python as coding. The predictive model build will enhance the efficiency and effectiveness of Tb detection, ultimately contributing to early diagnosis, timely treatment initiation, and better health outcomes for individuals affected by Tb.

CHAPTER 2

REVIEW OF LITERATURE

2.1 Tuberculosis

Tuberculosis (Tb) is an infectious disease caused by the bacteria *Mycobacterium tuberculosis* which primarily affects the lungs but can infect other regions of the body as well [\[1\]](#). It is a leading cause of illness and mortality worldwide, ranking among the most common ten leading causes of death. Tb is very contagious because it is transmitted by the inhalation of aerosols produced by people with pulmonary Tb [\[1\]\[2\]](#). The result of tuberculosis infection can range from quick clearance by innate immunity to the development of active disease or the development of latent infection that can later reactivate [\[3\]](#). The global impact of tuberculosis (Tb) is significant, with millions of new cases recorded each year. According to the World Health Organization's Global Tuberculosis Report 2022, there was a concerning increase in Tb infections worldwide in 2021, with 10.6 million new cases registered, a 4.5% increase compared to the previous year. The disruptions produced by the COVID-19 epidemic have had a negative impact on Tb control efforts. Africa and Asia, in particular, face a tremendous burden of tuberculosis, with a high concentration of patients. India, Indonesia, and China all contribute considerably to the worldwide Tb caseload, highlighting the critical need for tailored interventions in these areas. Additionally, the emergence of drug-resistant variants of tuberculosis (Tb), particularly multidrug-resistant (MDR) and extensively drug-resistant (XDR) Tb, offers a major challenge to tuberculosis control. These drug-resistant pathogens complicate treatment regimens and demand specialised management approaches. It is critical for worldwide Tb control efforts to ensure robust solutions to address the surge in Tb cases and treatment resistance.

With a large number of cases and a high prevalence of the disease, tuberculosis is a major public health concern in India. According to the India Tb Report 2022, by Central Tb division, published in mid-May 2023, the incidence rate of all kinds of Tb in 2020 was 188 per 100,000 population, with rates ranging by location. India is expected to have one-third of the global population with latent tuberculosis infection. However, the country has a number of problems in the fight against tuberculosis, including the growth of drug-resistant strains, the complexity and length of treatment regimens, and the co-occurrence of Tb and HIV. The direct observed treatment short-course (DOTS) technique has produced encouraging results, with high percentages of treatment success reported. Nonetheless, prolonged therapy, adherence to treatment protocols, and fair access to healthcare services are essential barriers to effectively combating tuberculosis in India. Because of its distinct properties that set it apart from other bacterial infections, *Mycobacterium tuberculosis* (MTb) infection is a prominent cause of sickness. One notable distinction is its ability to develop long-term infections within host cells [\[9\]](#). Unlike many other bacteria that are eliminated by the immune system or respond effectively to typical antibiotic treatments, MTb has evolved sophisticated methods to avoid detection by the immune system and persist within the host for extended periods [\[10\]](#). This propensity to endure contributes to the chronic nature of tuberculosis (Tb) and the difficulties in completely eradicating the infection. Another distinctive feature of MTb is its complex cell wall composition, characterized by lipid-rich acids known as mycolic acids [\[11\]](#). This distinct cell wall structure protects against host immune defences and prevents antibiotic penetration. As a result, typical antibiotic medications against other bacteria may not be effective against MTb, involving longer treatment durations and numerous drug combinations [\[12\]](#).

Furthermore, MTb has high genetic variability, resulting in strains with varying levels of virulence and drug resistance. This genetic heterogeneity helps MTb adapt to varied conditions and improves its ability to survive and spread within populations [\[13\]](#). Drug-resistant strains hinder treatment efforts and demand the development of novel therapeutic techniques. The immune response to MTb infection differs from the immunological response to other bacterial infections. MTb can disrupt phagocytic signaling pathways, block phagosome-lysosome fusion, and disrupt immunological responses within the phagosome. These mechanisms allow MTb to elude immune surveillance and develop a niche for survival within host cells [\[9\]\[12\]](#).

2.2 Mechanisms of Host Immune System Manipulation by *Mycobacterium tuberculosis*

Mycobacterium tuberculosis which belongs to the "Actinomycetales" order, is a slow-going bacteria; it has complicated strategies for manipulating the host immune system and delaying the immune response, which adds to the difficulties in treating this disease. A complete understanding of MTb's different strategies for evading immune responses is essential for creating successful treatments [13]. MTb uses a variety of strategies to ensure its survival and resistance to medicines. It inhibits phagosome-lysosome fusion, preventing acidification and destruction of the bacterium [14]. MTb can establish multi-drug resistant colonies and remain within the host in this manner. MTb enters the body predominantly by inhaling aerosolized droplets containing the bacteria [15]. MTb enters the lungs and comes into contact with the alveolar lining fluid, which contains surfactant proteins and hydrolases that can aid in the pathogen's uptake and degradation by immune cells. When MTb enters the alveoli, it is quickly engulfed by alveolar macrophages, the principal phagocytic cells. In most situations, these macrophages can eliminate germs via the innate immune response [15]. If MTb survives this first defense, it begins to multiply within the macrophages. It can migrate to neighboring epithelial and endothelial cells, resulting in exponential growth and a large bacterial burden. MTb can spread to other organs via lymphatic and circulatory pathways during the early stages of illness. MTb can establish long-term persistence in extrapulmonary locations such as lymph nodes, adipose tissue, and bone marrow by modifying the local tissue environment [16].

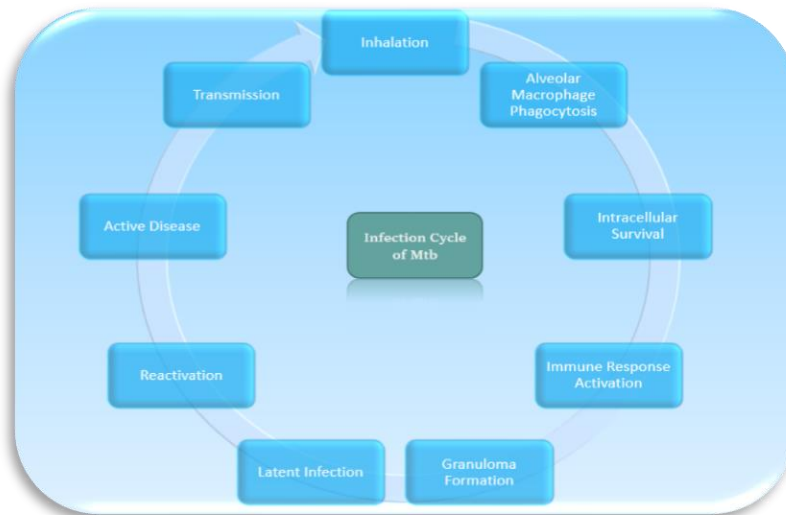


FIGURE 2.2.1: The schematic diagram illustration of *Mycobacterium tuberculosis* (MTb) infection cycle

2.3 Transmission of Tuberculosis

Transmission takes place when an infected individual coughs or sneezes and releases Tb bacteria into the air ^[17]. In some situations, individuals exposed to Tb may develop an active infection later in life. It is critical to distinguish between being infected with tuberculosis germs and having active tuberculosis disease because they reflect different stages of the infection.

The first stage is exposure, which occurs when someone comes into touch with someone who has active tuberculosis. During this stage, the exposed person often exhibits no signs or symptoms of the disease, and testing such as skin tests or chest X-rays may appear normal ^[18]. Latent tuberculosis infection is the second stage, at this stage, the person has tuberculosis germs in their body but is asymptomatic. The germs are effectively contained by the immune system, resulting in inactivity. Skin or blood tests for tuberculosis infection may be positive, although chest X-rays are usually normal or reveal merely prior scarring. Other regions of the body usually show no evidence of current illness. The third stage is tuberculosis, which shows aggressive symptoms and infection indicators. At this stage, the individual may have positive Tb infection tests, such as skin or blood testing, and imaging examinations, such as chest X-rays, may indicate abnormalities indicative of an active infection ^[19]. Additional diagnostic techniques, such as biopsies, may be used to establish the presence of active

tuberculosis illness [20].

Healthcare practitioners can accurately identify and manage tuberculosis by recognising the various stages of the infection [21]. Early detection and treatment are critical for limiting the spread of tuberculosis and reducing consequences.

Tb Stage	Signs and Symptoms	Key Differences and Tests
Exposure	Typically, no signs or symptoms	Negative skin test, normal chest X-ray
Latent Tb Infection	No symptoms	Positive skin or blood test for Tb, normal chest X-ray
Tb Disease	Cough, fatigue, weight loss, night sweats, fever, Hemoptysis (coughing up blood), chest pain, Shortness of breath, chest tightness	Positive or negative skin/blood test, positive chest X-ray, Sputum culture, nucleic acid amplification test (NAAT), Chest CT scan, bronchoscopy, biopsy

Table 1: The table summarises the stages of tuberculosis, signs and symptoms, and essential diagnostic tests.

2.4 Role of X-ray in Tb detection

The chest X-ray is critical in the diagnosis and therapy of tuberculosis (Tb) [22]. It is one of the most common imaging modalities used to detect lung abnormalities associated with tuberculosis infection [23]. Here are some essential elements showing the importance of X-ray in the detection of tuberculosis:

- a) Chest X-rays are routinely used for initial screening and diagnosis of tuberculosis [22]. They aid in the identification of pulmonary anomalies that may suggest tuberculosis infection, such as infiltrates, nodules, or cavities. X-rays can identify both active and latent tuberculosis [23].

- b) Active Tuberculosis: Chest X-rays can reveal certain patterns that indicate active tuberculosis [24]. Consolidation (areas of lung tissue filled with fluid or cellular debris), cavitation (development of cavities within the lung), and lymphadenopathy (enlarged lymph nodes in the chest) are examples of these patterns [25]. These findings can help healthcare providers make an accurate diagnosis and choose the best treatment options.
- c) Assessing illness Severity: X-rays can tell you a lot about the breadth and severity of Tb illness in your lungs [25]. They aid in determining the spread of lesions and the involvement of various lung segments. This data assists in selecting the best treatment plan and tracking the disease's progression over time [26].
- d) Chest X-rays are used to check the success of Tb treatment. Serial X-rays can detect changes in the lungs, such as infiltrate clearing or cavity size reduction, indicating a beneficial response to treatment [24]. They are critical for monitoring treatment progress, recognising potential problems, and changing therapy as necessary [27].
- e) Differentiating Tuberculosis from Other Lung illnesses: Chest X-rays can assist in distinguishing tuberculosis from other lung illnesses that may present with similar symptoms [28]. Healthcare providers can distinguish Tb from illnesses such as pneumonia, lung cancer, or other respiratory infections by analysing the typical characteristics associated with the disease [29].

Early diagnosis of active tuberculosis (Tb) with the help of machine learning (ML) techniques can greatly benefit in disease treatment and management [30]. Large volumes of patient data, including clinical, radiographic, and laboratory information, can be analyzed by ML algorithms to find patterns and signs of active tuberculosis infection [24]. Using ML algorithms, healthcare providers can create prediction models that can help identify people who are at high risk of getting active Tb or who may already have the disease. ML algorithms can also help improve the accuracy and efficiency of tuberculosis tests [30]. For example, machine learning (ML) approaches can be used to improve the interpretation of radiological images like chest X-rays or computed tomography (CT) scans, allowing for more exact diagnosis of Tb-related anomalies [31]. This can help radiologists make more accurate diagnosis, which can

lead to earlier intervention and treatment. On top of that, machine learning can aid in the creation of point-of-care diagnostic instruments that are inexpensive, portable, and appropriate for resource-limited situations [\[32\]](#). These techniques can identify tuberculosis quickly and accurately, even in places with limited access to advanced laboratory infrastructure.

2.5 Machine Learning and steps of building a machine learning model

Machine learning encompasses two primary categories, namely supervised and unsupervised learning. Both categories make use of mathematical models in order to provide computers the capacity to learn and carry out particular tasks.

Supervised Learning: Supervised learning makes use of labelled training data, which consists of inputs and associated "labelled" outputs. These labelled outcomes are used by models during training to evaluate and enhance their prediction abilities for brand-new, untainted data [\[33\]](#). Classification and regression methods are frequently the focus of supervised learning. Doctors sometimes make diagnoses of illnesses based on a group of symptoms, which is a problem known as classification. The focus of regression issues, on the other hand, is on making numerical predictions, such as determining how long a patient would stay in the hospital based on vital signs, medical history, and weight. Random forests (RF), decision trees (DT), Naive Bayes models, linear and logistic regression, support vector machines (SVM), and even neural networks are some of the algorithms that fall under the umbrella of supervised learning. An ensemble of independently trained decision trees makes up random forests, a subtype of decision tree. To get a better end result, these trees' projections are blended [\[34\]](#). A random sampling of the data and a random collection of features are used to construct each tree at each candidate split. As a result, predictive traits that are unique to the training set but might not be transferable to fresh data are not overemphasised. Even with noisy data, random forests are reliable and perform well.

A common machine learning algorithm for classification and regression tasks is the decision tree. It is a tree-like model where each leaf node represents the outcome or prediction, each internal node represents a decision based on that feature, and each branch represents a feature. The Decision Tree algorithm creates a collection of decision rules that direct the classification or prediction process by recursively

partitioning the data according to various criteria. Finding the optimal splits at each node that provide the most homogeneous subsets of data is the goal. Popular machine learning method Random Forest falls within the umbrella of supervised learning, more precisely, ensemble learning. It has a solid reputation for being adaptable, durable, and efficient for dealing with complicated data and high-dimensional feature spaces. Decision trees are the foundation of the Random Forest algorithm. To reach a forecast, decision trees partition the data into subsets based on feature values. They resemble trees. Individual decision trees, on the other hand, frequently experience overfitting, which occurs when the model is too tailored to the training data and struggles to generalise successfully to new, unforeseen data [\[35\]](#).

Naive Bayes models are a family of probabilistic classifiers based on Bayes' theorem with an assumption of independence between the features. Despite their simplistic assumptions, Naive Bayes classifiers are widely used in various machine learning tasks, particularly in text classification and spam filtering. The Naive Bayes algorithm calculates the probability of a data point belonging to a particular class based on the probabilities of the individual features given that class. It assumes that the features are conditionally independent, meaning that the presence or absence of one feature does not affect the presence or absence of another feature. This assumption simplifies the computations and makes Naive Bayes models computationally efficient [\[36\]](#). To train a Naive Bayes classifier, the algorithm estimates the prior probabilities of each class and the likelihood probabilities of the features given each class using a training dataset. The prior probability represents the probability of each class occurring in the dataset, while the likelihood probability represents the probability of observing a particular feature given a class. During the prediction phase, the Naive Bayes classifier calculates the posterior probability of each class for a given data point. It selects the class with the highest posterior probability as the predicted class for that data point. The calculation of the posterior probability involves multiplying the prior probability with the likelihood probabilities of the features.

Basic Steps involved in building a Machine Learning Model are [\[37\]](#):

- i. Data Collection: Relevant data is gathered from various sources for use in the machine learning process. The collected data is aimed to be comprehensive and representative of the problem domain.

- ii. **Data Pre-processing:** The collected data is cleaned, transformed, and prepared to ensure its quality and compatibility with machine learning algorithms. Techniques such as handling missing values, dealing with outliers, and normalizing or scaling features are applied to enhance the data.
- iii. **Model Selection:** The appropriate machine learning algorithm or model architecture is chosen that best suits the problem at hand. Consideration is given to factors such as the nature of the problem, availability of labeled data, dataset size, and complexity.
- iv. **Model Training:** The selected model is trained using the prepared data to learn patterns and relationships within the data. The model's internal parameters are adjusted based on the provided inputs and corresponding outputs, optimizing its performance.
- v. **Evaluation:** The performance of the trained model is assessed using appropriate metrics to measure its effectiveness. Metrics such as accuracy, precision, recall, or mean squared error are used to evaluate the model's ability to make accurate predictions.
- vi. **Performance Tuning:** The model's hyperparameters and configuration settings are fine-tuned to optimize its performance. Techniques like grid search, random search, or Bayesian optimization are employed to systematically explore the hyperparameter space and find the optimal combination.
- vii. **Do Predictions:** The trained model is deployed to make predictions on new, unseen data in real-world applications. The model utilizes its learned patterns and relationships to generate predictions or classifications for the given input data.

A branch of machine learning called deep learning is concerned with the creation and use of artificial neural networks, particularly deep neural networks. While both machine learning and deep learning are subfields of artificial intelligence, their methods and capabilities vary. Typically, features for machine learning algorithms must be manually extracted from the input data. These characteristics constitute the foundation for the learning process, in which the algorithm discovers patterns and relationships in the data to execute tasks or make predictions. The excellence and applicability of these hand-crafted features strongly influence the performance of

machine learning models. On the other hand, deep learning algorithms use numerous layers of interconnected neurons to automatically learn hierarchical representations of the input. Since these deep neural networks may learn directly from unprocessed data, explicit feature engineering is not necessary. In order to create more adaptable and potent models, the network learns to extract significant features and representations from the data itself [\[38\]](#).

The volume of data needed is another difference between deep learning and conventional machine learning. When given a significant amount of labelled training data, deep learning algorithms frequently perform better. Deep neural networks are capable of efficiently capturing complex patterns and producing reliable predictions when given a large number of parameters to optimise. Deep learning has also achieved outstanding results in fields including speech recognition, natural language processing, and computer vision [\[39\]](#). Recurrent neural networks (RNNs) have made substantial progress in language comprehension and sequence modelling, while convolutional neural networks (CNNs) have made breakthroughs in object detection and image classification [\[40\]\[46\]](#).

Deep learning does, however, present some difficulties. Deep neural network training can be computationally expensive and calls for a lot of resources, such strong GPUs or specialised hardware [\[40\]](#). Furthermore, deep learning models are frequently referred to as "black boxes," as they are more difficult to interpret than typical machine learning models, which makes it difficult to comprehend how decisions are made.

Property	Machine Learning	Deep Learning
Model	Various algorithms (e.g., decision trees, SVM)	Artificial neural networks
Data Size	Works well with small to medium-sized datasets	Excels with large-scale datasets
Training	Computationally efficient	Computationally expensive

TABLE 2: This table summarizes the differences between Machine Learning (ML) and Deep Learning (DL) in terms of the model, data size, and training process

Deep learning methods, particularly convolutional neural networks (CNNs), are used to recognise and identify objects or features in images. This process is known as image detection using deep learning. By enabling very precise and automated detection methods, deep learning has revolutionised image detection. A deep neural network is trained using a sizable dataset of labelled images. By changing its internal parameters during the training phase, the network gains the ability to recognise patterns, features, and representations in the images. CNNs are neural networks with numerous layers that work together to execute convolution, pooling, and nonlinear activation operations. They were created primarily for image processing. The network gains the ability to recognise visual patterns and characteristics that are connected to particular objects or classes throughout training. This is accomplished by applying optimisation techniques like gradient descent to reduce the discrepancy between the network's anticipated outputs and the ground truth labels of the training images.

By feeding the network with fresh, undiscovered photos after it has been trained, it can be used for image detection. Utilising its previously learned representations, the network analyses the incoming image and outputs the identified objects or features together with the matching class labels or bounding boxes.

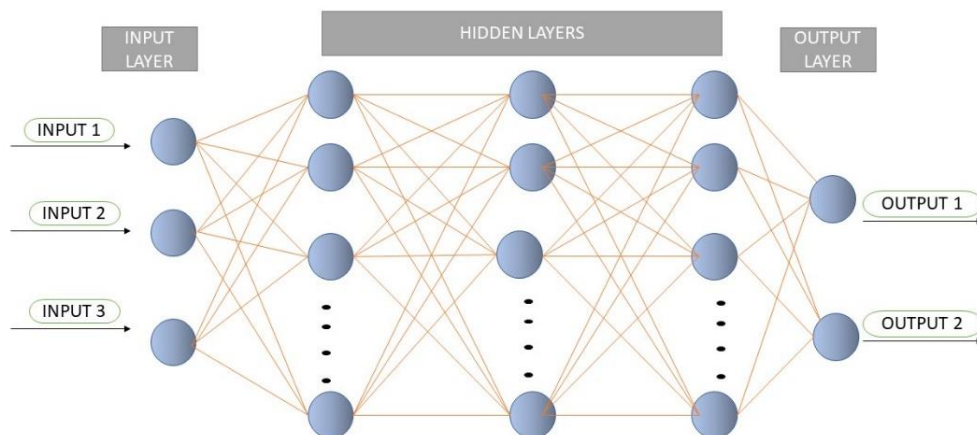


FIGURE 2.5.1: Schematic representation of Neural Network

2.6 Image detection and analysis

Convolutional Neural Networks (CNNs) have revolutionized the field of computer vision by introducing powerful techniques for extracting meaningful features from visual data. They have shown exceptional performance in a wide range of applications, enabling computers to understand and interpret images with remarkable accuracy [42]. One key aspect of CNNs is their ability to automatically learn and extract hierarchical representations of visual features. The network comprises multiple layers, including convolutional layers, pooling layers, and fully connected layers [43][45]. Each layer performs a specific operation on the input data, allowing the network to progressively learn and capture increasingly complex and abstract features [44]. The convolutional layers are responsible for applying convolutional filters to the input image. These filters are small matrices that scan the image and perform element-wise multiplications and summations. By sliding these filters across the input image, the network can detect local patterns and features, such as edges, corners, and textures [42]. These features are essential building blocks for recognizing more complex structures in the image. Pooling layers, such as max pooling or average pooling, are employed to down sample the feature maps obtained from the convolutional layers. Pooling reduces the spatial dimensions of the features while preserving their essential information. This down sampling operation helps in achieving translation invariance and enables the network to focus on the most salient features. The fully connected layers at the end of the network combine the extracted features and perform classification or regression tasks. These layers have connections to all the neurons in the previous layer, allowing the network to learn complex relationships and make predictions based on the extracted features. One significant advantage of CNNs is their ability to learn hierarchical representations of visual data. Lower layers in the network learn simple and low-level features, such as edges and textures, while deeper layers capture more abstract and high-level concepts, such as shapes and objects. This hierarchical representation enables the network to understand the visual data at different levels of abstraction, leading to superior performance in recognizing and interpreting complex images. Moreover, CNNs can be trained on large-scale datasets, such as ImageNet, which contain millions of labeled images. Pre-training on such datasets allows the network to learn generic features that are applicable across a wide range of visual recognition tasks. This pre-trained model can then be fine-tuned on a smaller dataset specific to

the target task, enabling faster convergence and improved performance [41]. In recent years, CNNs have demonstrated remarkable success in various applications. They have achieved state-of-the-art performance in image classification challenges, surpassing human-level accuracy in many cases. CNNs are also widely used in object detection, where they can localize and identify multiple objects within an image. Additionally, CNNs play a crucial role in semantic segmentation, where they assign class labels to each pixel, enabling detailed understanding and analysis of complex scenes [47].

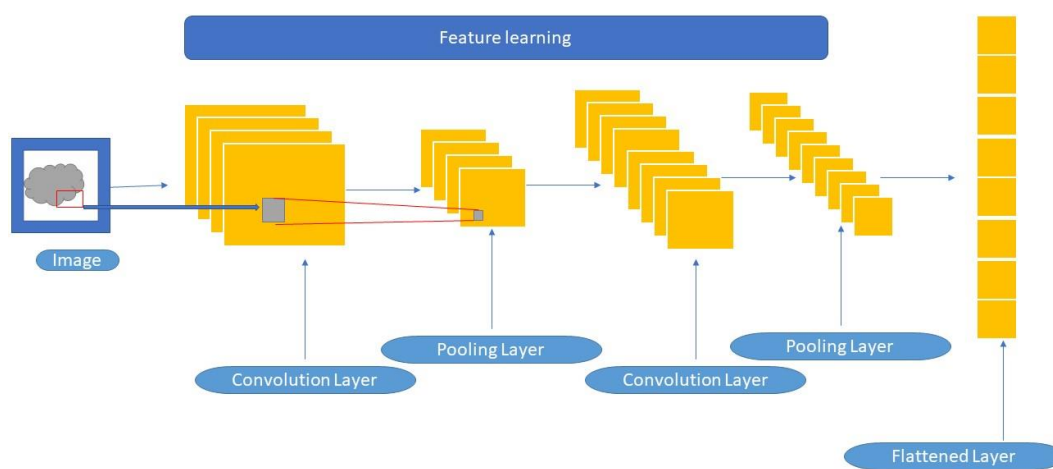


FIGURE 2.6.1: Schematic representation of a CNN with two hidden layers

2.7 Orange Data Mining tool

Orange is a C++ based core model & routines library that supports a wide range of machine learning and data mining methods, both standard and non-standard. It's a free and open-source application for data visualisation, data mining, and machine learning. Orange is a fully programmable system that allows you to quickly prototype new algorithms and test patterns [48]. It's a collection of python-based modules found in the main libraries. It uses Python to provide some functionality for which runtime is not critical [49]. An orange is a component-based technique for machine learning and data mining that combines all of these features. Orange is aimed at both advanced users and experts in data mining and machine learning who wish to design and test their own strategies while reusing the same or more code as possible, and newcomers who can either write short python contents for data analysis [48]. Orange's goal is to serve as a

medium for experiment-based selection, prediction, and reinforcement learning. It's mostly utilised in bioinformatics, genomics, healthcare, and education. It is utilised in education to provide improved teaching techniques for artificial intelligence and machine learning to biology, biomedical, and informatics students [\[50\]](#).

2.7.1 Orange Widgets

Orange widgets are the cells that provide a graphical interface for mining the data and machine learning used by Orange. They include widget for entering data and processing, classifications, prediction, association rules, and clustering, as well as a collection of widgets for model assessment and display of evaluation outcomes, and widgets for transferring models to tool [\[48\]](#). Keys are passed from the sender to the recipient widget when data is exchanged across widgets. A directory widget, for instance, can create data objects which a widget classifier learning widget can accept. The classification tree creates a classification model and provides the data to a widget that displays the tree graphically. The file widgets and objects can provide a data set to an evaluation widget [\[50\]](#).

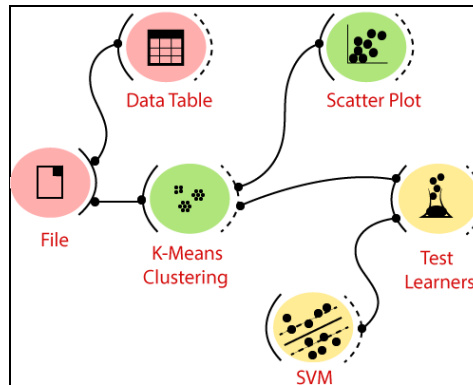


FIGURE 2.7.1: Orange workflow and widgets

Early identification of active tuberculosis allows for the rapid beginning of effective treatment, which is critical for improving patient outcomes and reducing transmission to others. By analyzing diverse data sources and offering automated decision support to healthcare practitioners, ML-based techniques can speed up the diagnosis of active Tb patients. This can help healthcare systems prioritize resources, distribute appropriate interventions, and guarantee that individuals in need receive treatment on time.

CHAPTER 3

METHODOLOGY

3.1 Data Collection

- The data was collected from three primary sources: Sarojini Naidu Medical College (Agra), NSCB Medical College (Jabalpur), and Kaggle.
- The X-ray images were collected from patients who had been diagnosed with tuberculosis as well as those without tuberculosis. This ensured a diverse dataset representing different conditions and demographics.
- The X-ray images were accompanied by accurate labels indicating the presence or absence of tuberculosis for each image.

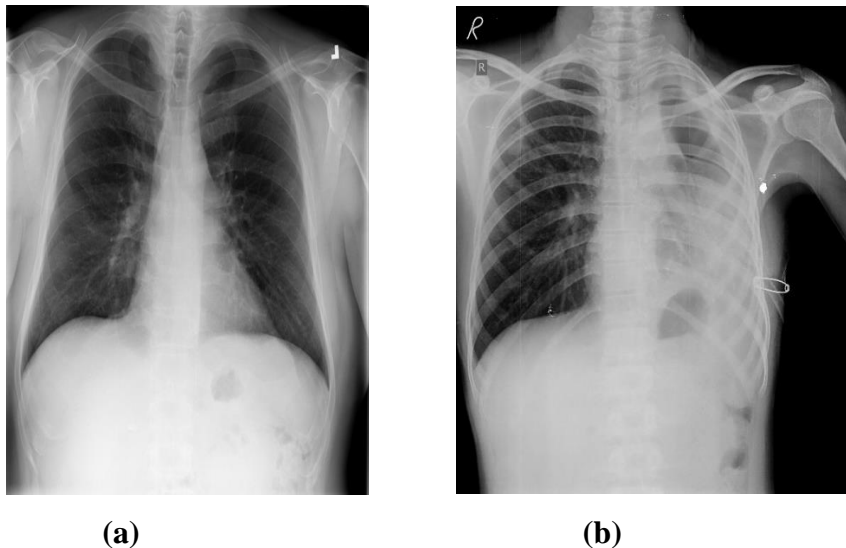


FIGURE 3.1.1: An X-ray visualization of (a) Normal Patient (Tb negative)
(b) Tuberculosis Patient (Tb Positive)

3.2 Data Validation and Cleaning

- The collected dataset underwent a thorough validation process to ensure data quality and integrity.
- Images with poor quality, artifacts, or significant noise were excluded from the dataset to maintain the accuracy and reliability of the analysis.
- The dataset was carefully checked for any duplicates or inconsistencies, and any identified issues were resolved to ensure data consistency and reliability.

3.3 Orange Data Mining Tool for the prediction of best algorithm

- The data was loaded in Orange tool and the best algorithm was detected
- Based on the result Neural Network was found to be most accurate

3.4 Uploading the Data on Google Colab

- After data validation and cleaning, the dataset was prepared for analysis in Google Colab (a cloud-based Jupyter notebook environment).
- For the X-ray images obtained from Sarojini Naidu Medical College Agra and NSCB Medical College Jabalpur, the images were uploaded from the local desktop to the Google Colab environment.

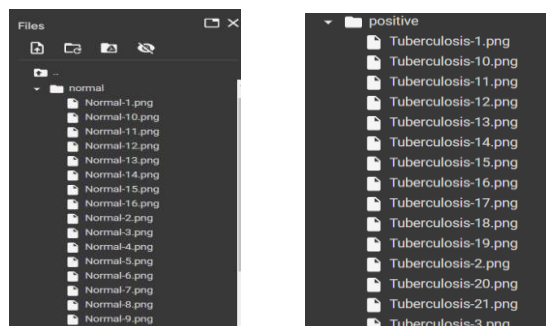
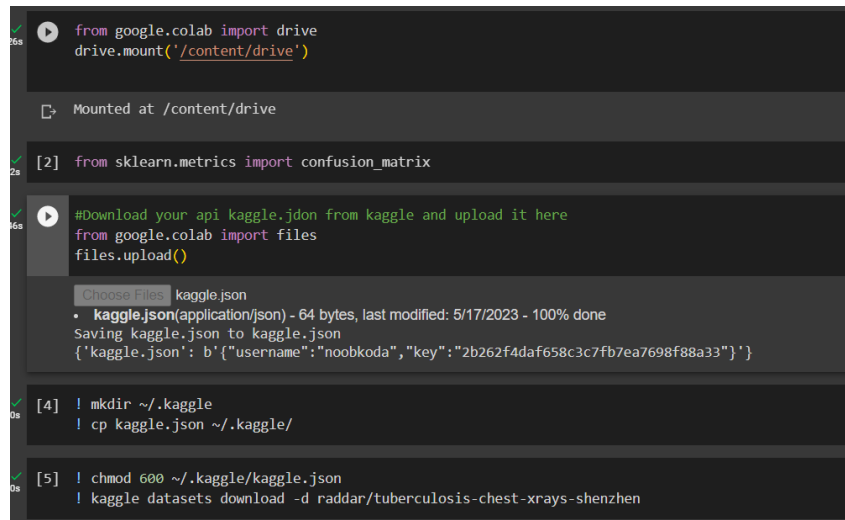


FIGURE 3.4.1: Uploaded X-Ray images from desktop for training the Model

- The Kaggle data, being publicly available, was imported directly into Google

Colab using Kaggle's API



```
from google.colab import drive
drive.mount('/content/drive')

Mounted at /content/drive

[2] from sklearn.metrics import confusion_matrix

#Download your api kaggle.json from kaggle and upload it here
from google.colab import files
files.upload()

Choose Files: kaggle.json
• kaggle.json(application/json) - 64 bytes, last modified: 5/17/2023 - 100% done
Saving kaggle.json to kaggle.json
{'kaggle.json': b'{"username": "noobkoda", "key": "2b262f4daf658c3c7fb7ea7698f88a33"}'}

[4] ! mkdir ~/.kaggle
! cp kaggle.json ~/.kaggle/

[5] ! chmod 600 ~/.kaggle/kaggle.json
! kaggle datasets download -d raddar/tuberculosis-chest-xrays-shenzhen
```

FIGURE 3.4.2: Uploaded X-Ray images from Kaggle for training the Model

3.5 Further data pre-processing and splitting of data into test and train datasets:

- To prepare the data for training the predictive model, additional preprocessing steps were performed. Firstly, the X-ray images obtained from the hospitals were processed to ensure they had the same dimensions. This step involved resizing all images to a uniform width and height.
- Next, the data was split into training and testing datasets. For this purpose, an image data generator provided by the Keras library was utilized. The images were rescaled by dividing their pixel values by 255.0 to bring them within the range of 0 to 1.
- The training data was generated using the `flow_from_directory` method from the image data generator. The directory specified was `'/content/OutputData/train'`, and the target size for the images was set to (150, 150). The `class_mode` was defined as 'sparse' to handle the categorical labels associated with the images.
- Similarly, the testing data was generated using the `flow_from_directory` method. The directory specified was `'/content/OutputData/test'`, and the target size was set to (150, 150). Again, the `class_mode` was set to 'sparse' to align with the categorical labels

```
Processing Image and partitioning data into Train and Test

[ ] i_width = 150
    i_height = 150

    datagen = ImageDataGenerator(rescale=1/255.0)

    training_data_gen = datagen.flow_from_directory(directory='/content/OutputData/train',
                                                    target_size = (i_width, i_height),
                                                    class_mode = 'sparse')
    testing_data_gen = datagen.flow_from_directory(directory='/content/OutputData/test',
                                                    target_size = (i_width, i_height),
                                                    class_mode = 'sparse',)
```

FIGURE 3.5.1: Data pre-processing and splitting the data into train and test

3.6 Deep Learning Model Training:

- A sequential model was created using the Keras library to implement the deep learning model for tuberculosis detection. The model consisted of multiple convolutional layers followed by max pooling layers to extract relevant features from the input images.
- The model architecture included four sets of convolutional layers, each followed by a max pooling layer. The convolutional layers had 32, 64, 128, and 192 filters, respectively, with a kernel size of (3,3). The 'relu' activation function was applied to introduce non-linearity.
- The flattened output from the convolutional layers was passed through dense layers for further feature extraction. The dense layers had 128, 228, and 270 units, respectively, with 'relu' activation functions. Dropout regularization was applied with a rate of 0.4 for the first dropout layer and 0.3 for the subsequent two dropout layers.
- The final dense layer consisted of a single unit with 'sigmoid' activation, representing the output for tuberculosis classification.
- The model was compiled using the 'Adam' optimizer and 'binary_crossentropy' loss function. Accuracy was chosen as the evaluation metric.


```

model = Sequential()

#convolution
model.add(Conv2D(32, (3,3), input_shape = (i_width, i_height, 3), activation='relu', padding='same'))
model.add(MaxPool2D(2,2))

model.add(Conv2D(64, (3,3), activation='relu', padding='same'))
model.add(MaxPool2D(2,2))

model.add(Conv2D(128, (3,3), activation='relu', padding='same'))
model.add(MaxPool2D(2,2))

model.add(Conv2D(192, (3,3), activation='relu', padding='same'))
model.add(MaxPool2D(2,2))

#Dense
model.add(Flatten())

model.add(Dense(128, activation='relu'))
model.add(Dropout(0.4))

model.add(Dense(228, activation='relu'))
model.add(Dropout(0.3))

model.add(Dense(270, activation='relu'))
model.add(Dropout(0.3))

model.add(Dense(1, activation='sigmoid'))

[ ] model.compile(optimizer='Adam', loss='binary_crossentropy', metrics=['accuracy'])

```

FIGURE 3.6.1: Code for building Convolutional layers

3.7 Performance Evaluation:

- The training data generated from the image data generator was used to fit the model. The model was trained for 20 epochs with the specified number of steps per epoch.
- The validation data generated from the image data generator was used for evaluating the model's performance during training.
- The loss and accuracy metrics were recorded during training to monitor the model's learning progress.
- The training and validation loss were plotted to visualize the model's performance over epochs.
- Similarly, the training and validation accuracy were plotted to observe the model's learning and generalization capabilities.

```
Evaluating , CM

# Evaluate the model
_, accuracy = model.evaluate(testing_data_gen, verbose=0)
print('Test Accuracy: %.2f%%' % (accuracy * 100))

# Get predictions
y_pred = model.predict(testing_data_gen)
y_pred = py.round(y_pred).flatten()

# Get true labels
y_true = testing_data_gen.labels

# Create confusion matrix
c_m = confusion_matrix(y_true, y_pred)
print('Confusion Matrix:')
print(c_m)

# Plot confusion matrix
mpt.imshow(c_m, cmap=mpt.cm.Blues)
mpt.title('Confusion Matrix')
mpt.colorbar()
mpt.xlabel('Predicted')
mpt.ylabel('True')
mpt.show()

[ ] model.save('Tuberculosis.h5')

[ ] model = ft.keras.models.load_model("Tuberculosis.h5")
converter = ft.lite.TFLiteConverter.from_keras_model(model)
tflite_model = converter.convert()
open('Tuberculosis.tflite', 'wb').write(tflite_model)
```

FIGURE 3.7.1: This section of code evaluates the built Model

3.8 Model Deployment and Integration:

- The trained model was saved in the '.h5' format for future use.
- To facilitate model deployment, the saved model was converted to the TensorFlow Lite format (.tflite) using the TFLite Converter.
- The TensorFlow Lite model was then loaded, and an Interpreter was created to perform inference on new data.
- An image was loaded, pre-processed, and fed into the TensorFlow Lite model for inference. The output probabilities were obtained.
- A threshold of 0.5 was set for classification, where probabilities above the threshold indicated tuberculosis presence.
- The model deployment process enabled integration into various applications and systems, allowing healthcare professionals to utilize the model for tuberculosis detection efficiently.

```

import tensorflow as tf
import numpy as np
from PIL import Image

# Load the TFLite model
interpreter = tf.lite.Interpreter(model_path="/content/Tuberculosis.tflite")
interpreter.allocate_tensors()

# Get input and output details
input_details = interpreter.get_input_details()
output_details = interpreter.get_output_details()

image_path = "/content/Test/Normal77.png"
# Load and preprocess the image
image = Image.open(image_path).resize((150, 150))
image = np.array(image) / 255.0 # Normalize pixel values to [0, 1]
image = image.astype(np.float32) # Convert to float32
image = np.expand_dims(image, axis=0) # Add batch dimension

# Set the image as the input to the model
interpreter.set_tensor(input_details[0]['index'], image)

# Run the inference
interpreter.invoke()

# Get the predicted probabilities
output = interpreter.get_tensor(output_details[0]['index'])
predicted_probability = output[0][0] # Assuming single output node

# Define the threshold for classification
threshold = 0.5

```

FIGURE 3.8.1: The section of code shown in the picture is to save the model in tflite format and further testing it using a picture which was not used to train the model

3.9 Prediction and threshold setting for classification:

- A threshold value of 0.5 was chosen as a reference point to classify the predicted probabilities into two categories: "Tb Negative" and "Tb Positive"
- If the predicted probability was above the threshold (greater than 0.5), the person was classified as having tuberculosis. Otherwise, if the predicted probability was below or equal to the threshold, the person was classified as Tb negative.

```

# Define the threshold for classification
threshold = 0.5

# Make the prediction
if predicted_probability > threshold:
    prediction = "Tb Positive"
else:
    prediction = "Tb Negative"

print("Prediction:", prediction)
print("Probability:", predicted_probability)

```

FIGURE 3.9.1: The section of code shown in the picture is to predict if new the test image is Tb Positive or Tb negative

**(for the complete code, please refer to the Appendix A)*

CHAPTER 4

RESULT AND DISCUSSION

4.1 Data Collection, Pre-processing and Model building

The deep learning model for tuberculosis detection was trained and evaluated using a dataset collected from Sarojini Naidu Medical College Agra, NSCB Medical College Jabalpur, and Kaggle. The dataset comprised a collection of X-ray images obtained from individuals, with each image annotated to indicate the presence or absence of tuberculosis. The dataset was carefully curated and labelled by medical professionals to ensure accuracy and reliability. To determine the optimal model for tuberculosis detection, various machine learning algorithms, including Support Vector Machines (SVM), K-Nearest Neighbors (KNN), CN2 Rule Induction, and Neural Network, were employed. These algorithms were implemented using the Orange Data Mining tool.

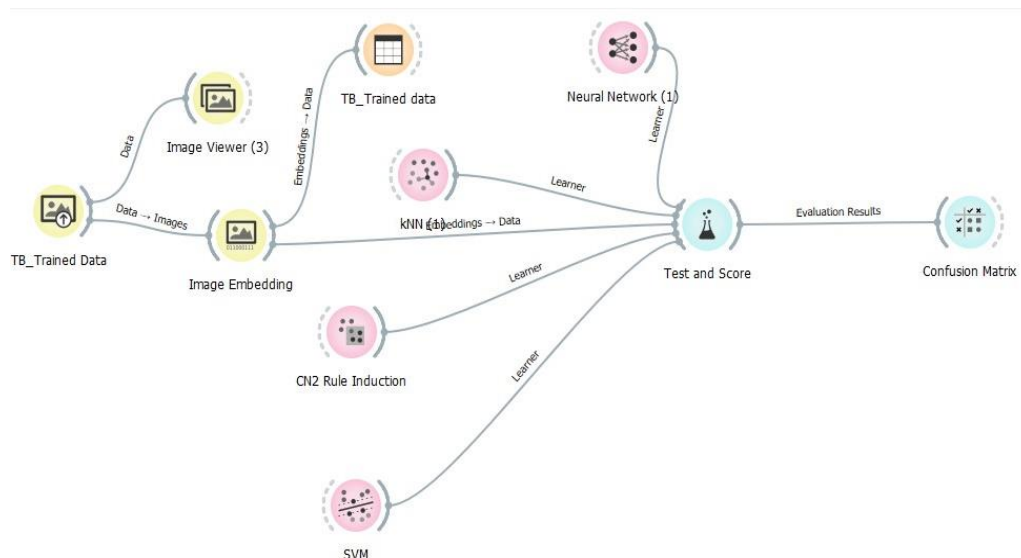


FIGURE 4.1.1: A schematic workflow built on Orange Data mining tool for find the best model for this study

Based on the Literature Reviews and results of Confusion Matrix of each algorithm, Neural Network was observed to show the best result.

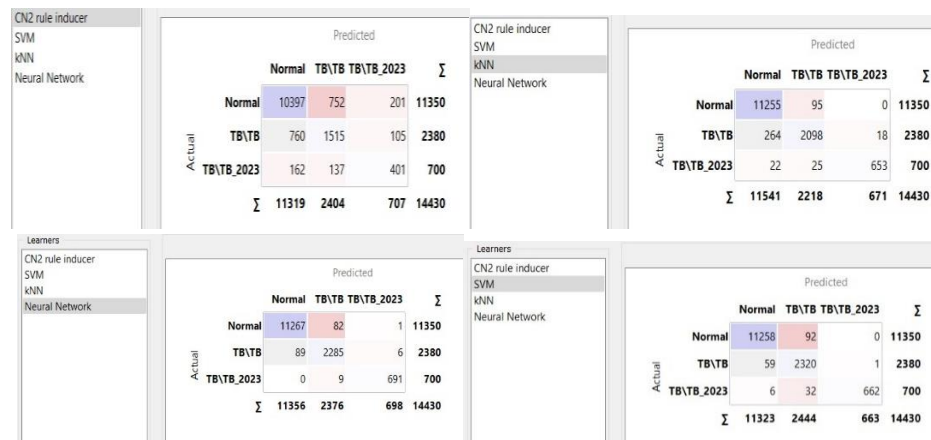


FIGURE 4.1.2: The confusion matrix of CN2 rule inducer, SVM, kNN and Neural Network

It exhibited the ability to effectively learn and recognize patterns within the X-ray images, enabling accurate tuberculosis detection. The Neural Network algorithm was, therefore, chosen to build the model for further development and evaluation. The model was fed the pre-processed dataset, which underwent data cleaning, resizing of images to a consistent dimension, and appropriate normalization. Once the model was trained, it was evaluated using a separate test dataset. This dataset consisted of X-ray images that were not included in the training process. The performance of the model was evaluated by percentage accuracy which provided insights into the model's ability to correctly classify tuberculosis cases and distinguish them from Tb negative images.

4.2 Model Evaluation Results

During the experimentation phase, it was observed that the size of the dataset had a significant impact on the accuracy of the model for tuberculosis detection. Initially, smaller amount of data (662 X-Ray images) was used for training and evaluation.

```
New Directories

[ ] !mkdir data

[ ] !mkdir data/normal
    !mkdir data/positive

[ ] for i in range(len(normal)):
    path = '/content/images/images/' + normal[i]
    !mv $path /content/data/normal

[ ] for i in range(len(positive)):
    path = '/content/images/images/' + positive[i]
    !mv $path /content/data/positive

[ ] input_folder = "/content/data"
    output = "/content/OutputData" #where you want the split datasets saved, one t

splitfolders.ratio(input_folder, output=output, seed=42, ratio=(.8, .0, 0.2))

Copying files: 662 files [00:36, 18.36 files/s]
```

FIGURE 4.2.1: The picture shows that 662 files were uploaded

The accuracy of this model was 78.36%

```
Evaluating , CM

[ ] # Evaluate the model
_, accuracy = model.evaluate(testing_data_gen, verbose=0)
print('Test Accuracy: %.2f%%' % (accuracy * 100))

# Get predictions
y_pred = model.predict(testing_data_gen)
y_pred = py.round(y_pred).flatten()

# Get true labels
y_true = testing_data_gen.labels

# Create confusion matrix
c_m = confusion_matrix(y_true, y_pred)
print('Confusion Matrix:')
print(c_m)

# Plot confusion matrix
mpt.imshow(c_m, cmap=mpt.cm.Blues)
mpt.title('Confusion Matrix')
mpt.colorbar()
mpt.xlabel('Predicted')
mpt.ylabel('True')
mpt.show()

Test Accuracy: 78.36%
5/5 [=====] - 20s 4s/step
Confusion Matrix:
[[31 35]
 [28 40]]
```

FIGURE 4.2.2: The accuracy of this model was 78.36%

To increase the accuracy, a larger and more diverse dataset was employed (3168 X-Ray images).

```
New Directories

!mkdir data

[17] !mkdir data/normal
!mkdir data/positive

[18] for i in range(len(normal)):
    path = '/content/images/images/' + normal[i]
    !mv $path /content/data/normal

[19] for i in range(len(positive)):
    path = '/content/images/images/' + positive[i]
    !mv $path /content/data/positive

[29] input_folder = "/content/data"
    output = "/content/OutputData" #where you want the split datasets saved. one w

    splitfolders.ratio(input_folder, output=output, seed=42, ratio=(.8, .0, 0.2))

Copying files: 3168 files [01:04, 49.06 files/s]
```

FIGURE 4.2.3: The picture shows that 3168 files were uploaded

The accuracy of the model significantly improved to 96.11%.

```
 Evaluating , CM

# Evaluate the model
_, accuracy = model.evaluate(testing_data_gen, verbose=0)
print('Test Accuracy: %.2f%%' % (accuracy * 100))

# Get predictions
y_pred = model.predict(testing_data_gen)
y_pred = py.round(y_pred).flatten()

# Get true labels
y_true = testing_data_gen.labels

# Create confusion matrix
c_m = confusion_matrix(y_true, y_pred)
print('Confusion Matrix:')
print(c_m)

# Plot confusion matrix
mpt.imshow(c_m, cmap=mpt.cm.Blues)
mpt.title('Confusion Matrix')
mpt.colorbar()
mpt.xlabel('Predicted')
mpt.ylabel('True')
mpt.show()

... Test Accuracy: 96.11%
7/24 [=====>.....] - ETA: 31s
```

FIGURE 4.2.4: The accuracy of the model significantly improved to 96.11%.

This improvement in accuracy can be attributed to the increased volume of data available for the model to learn from. With a larger dataset, the model had access to a wider range of tuberculosis cases and non-tuberculosis cases, allowing it to better capture the complex patterns and features indicative of the disease. The increased diversity in the dataset also helped the model generalize better to unseen instances, making it more robust and reliable in real-world scenarios.

4.3 Result and Validation of Model performance by testing it with help of the Data which was not used to train the model

<i>COUNT of Image Number</i>	<i>Model Prediction</i>			
	<i>Actual Result</i>	Healthy	TB	Grand Total
Healthy		206	2	208
TB		15	139	154
Grand Total		221	141	362
<u>% of result predicted correct</u>		<u>95.3038674</u>		

TABLE 3: The table displays the model predictions for a dataset. The rows are the actual findings, while the columns are the model projections.

The table displays the count of images and the corresponding model predictions for a given dataset. The dataset consists of two classes: "Healthy" and "Tb" (Tuberculosis). The rows represent the actual results of the images, while the columns represent the model predictions. According to the table, the algorithm correctly predicted 206 of the 208 healthy images. However, it misidentified two healthy images as Tb. Similarly, out of 154 Tb images, the algorithm correctly predicted 139 as Tb but incorrectly labelled 15 as healthy.

The sample included 221 Tb negative X ray images and 141 Tb positive X ray images. The percentage of accurately anticipated results can be used to assess the model's overall performance. In this case, the model predicted the class labels with an accuracy of roughly 95.30%.

CHAPTER 5

CONCLUSION

The project focused on using machine learning techniques, specifically deep learning models, for tuberculosis (Tb) detection using X-ray images. The study successfully developed and trained a deep learning model using a dataset collected from Sarojini Naidu Medical College Agra, NSCB Medical College Jabalpur, and Kaggle. The model exhibited high accuracy, with a significant improvement observed when a larger and more diverse dataset was used. The model predicted class labels with high accuracy, obtaining an overall accuracy of 95.30%. The majority of healthy images were accurately identified, with only a few mislabelled as TB. Similarly, the model performed well in recognising tuberculosis cases, correctly identifying the majority while misclassifying a few as healthy.

The results demonstrate the potential of machine learning, particularly deep learning, in enhancing the accuracy and efficiency of Tb detection. By leveraging the power of neural networks and image analysis, the developed model showed promising performance in distinguishing between Tb and non-Tb cases based on X-ray images. This has important implications for early detection and prompt initiation of treatment, ultimately improving patient outcomes and reducing the spread of the disease. The future potential of this research lies in further exploration and development of machine learning techniques for tuberculosis (Tb) detection. Areas of potential include incorporating additional data modalities, such as clinical data and demographics, exploring transfer learning and model optimization techniques, addressing class imbalance in datasets, conducting external validation and clinical integration, and considering ethical considerations and fairness in model deployment. These advancements have the potential to enhance the accuracy, efficiency, and real-world impact of Tb detection, ultimately leading to better patient care and control of the disease.

Appendix A

The code presented in the following pages corresponds to the implementation of a project titled "Convolutional Neural Network-assisted detection of primary tuberculosis for improved diagnosis."

```
# Dataset Download
"""

!pip install -U -q kaggle
!mkdir -p ~/.kaggle

#Download your api kaggle.json from kaggle and upload it here
from google.colab import files
files.upload()

! mkdir ~/.kaggle
! cp kaggle.json ~/.kaggle/

! chmod 600 ~/.kaggle/kaggle.json
! kaggle datasets download -d raddar/tuberculosis-chest-xrays-shenzhen

#unzipping the file
from zipfile import ZipFile
file_name = 'content/tuberculosis-chest-xrays-shenzhen.zip'

with ZipFile(file_name, 'r') as zip:
    zip.extractall()
    print('Done')

""""# Libraries

"""

! pip install split-folders

import tensorflow as tf
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Dense, Flatten, Conv2D, BatchNormalization, MaxPool2D,
Dropout
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing import image
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```

import cv2
import splitfolders
from glob import glob

"""# Preprocessing"""

datainfo = pd.read_csv('/content/shenzhen_metadata.csv')

datainfo.head()

normal = []
positive = []

def extract_target(x):
    for i in range(len(x['study_id'])):
        if x['findings'][i] == 'normal':
            normal.append(x['study_id'][i])
        else:
            positive.append(x['study_id'][i])

extract_target(datainfo)

len(normal)

len(positive)

"""# Creating Directories

"""

!mkdir data

!mkdir data/normal
!mkdir data/positive

for i in range(len(normal)):
    path = '/content/images/images/' + normal[i]
    !mv $path /content/data/normal

for i in range(len(positive)):
    path = '/content/images/images/' + positive[i]
    !mv $path /content/data/positive

input_folder = "/content/data"
output = "/content/dataset"

splitfolders.ratio(input_folder, output=output, seed=42, ratio=(.8, .0, 0.2))

"""# Data Visualization """

```

```

tuberculosis = glob('/content/dataset/test/positive/*.png')
normal = glob('/content/dataset/test/normal/*.png')

plt.title('Normal')
plt.imshow(image.load_img(np.random.choice(normal)))
plt.show()

plt.title('Tuberculosis')
plt.imshow(image.load_img(np.random.choice(tuberculosis)))
plt.show()

"""# Image Processing and Data Partition into Train and Test"""

img_width = 150
img_height = 150

datagen = ImageDataGenerator(rescale=1/255.0)

train_data_gen = datagen.flow_from_directory(directory='/content/dataset/train',
                                             target_size = (img_width, img_height),
                                             class_mode = 'sparse')
test_data_gen = datagen.flow_from_directory(directory='/content/dataset/test',
                                             target_size = (img_width, img_height),
                                             class_mode = 'sparse',)

"""# Deep Learning Model"""

model = Sequential()

#convolution
model.add(Conv2D(32, (3,3), input_shape = (img_width, img_height, 3), activation='relu',
padding='same'))
model.add(MaxPool2D(2,2))

model.add(Conv2D(64, (3,3), activation='relu', padding='same'))
model.add(MaxPool2D(2,2))

model.add(Conv2D(128, (3,3), activation='relu', padding='same'))
model.add(MaxPool2D(2,2))

model.add(Conv2D(192, (3,3), activation='relu', padding='same'))
model.add(MaxPool2D(2,2))

#Dense
model.add(Flatten())

model.add(Dense(128, activation='relu'))
model.add(Dropout(0.4))

```

```

model.add(Dense(228, activation='relu'))
model.add(Dropout(0.3))

model.add(Dense(270, activation='relu'))
model.add(Dropout(0.3))

model.add(Dense(1, activation='sigmoid'))

model.compile(optimizer='Adam', loss='binary_crossentropy', metrics=['accuracy'])

r = model.fit_generator(generator=train_data_gen,
                       steps_per_epoch=len(train_data_gen),
                       epochs=20,
                       validation_data= test_data_gen,
                       validation_steps = len(test_data_gen))

"""# Analysis/Evaluation """

plt.title('Loss')
plt.plot(r.history['loss'], label='loss')
plt.plot(r.history['val_loss'], label='val_loss')
plt.legend()

plt.plot('Accuracy')
plt.plot(r.history['accuracy'], label='acc')
plt.plot(r.history['val_accuracy'], label='val_acc')
plt.legend()

"""# Saving the Model

"""

model.save('Tuberculosis.h5')

model = tf.keras.models.load_model("Tuberculosis.h5")
converter = tf.lite.TFLiteConverter.from_keras_model(model)
tflite_model = converter.convert()
open("Tuberculosis.tflite", 'wb').write(tflite_model)

import tensorflow as tf
import numpy as np
from PIL import Image

# Load the TFLite model
interpreter = tf.lite.Interpreter(model_path="Tuberculosis.tflite")
interpreter.allocate_tensors()

```

```

# Get input and output details
input_details = interpreter.get_input_details()
output_details = interpreter.get_output_details()

# Load and preprocess the image
image_path = "/content/test.jpg"
image = Image.open(image_path).resize((150, 150))
image = np.array(image) / 255.0 # Normalize pixel values to [0, 1]
image = image.astype(np.float32) # Convert to float32
image = np.expand_dims(image, axis=0) # Add batch dimension

# Set the image as the input to the model
interpreter.set_tensor(input_details[0]['index'], image)

# Run the inference
interpreter.invoke()

# Get the predicted probabilities
output = interpreter.get_tensor(output_details[0]['index'])
predicted_probability = output[0][0] # Assuming single output node

# Define the threshold for classification
threshold = 0.5

# Make the prediction
if predicted_probability > threshold:
    prediction = "Tuberculosis"
else:
    prediction = "Healthy"

print("Prediction:", prediction)
print("Probability:", predicted_probability)

```

REFERENCES

- [1] W. Zhai, F. Wu, Y. Zhang, Y. Fu, and Z. Liu, "The Immune Escape Mechanisms of Mycobacterium Tuberculosis," *International Journal of Molecular Sciences*, vol. 20, no. 2, p. 340, Jan. 2019, doi: 10.3390/ijms20020340.
- [2] Y.-C. Liu *et al.*, "Immune activation of the host cell induces drug tolerance in Mycobacterium tuberculosis both in vitro and in vivo," *Journal of Experimental Medicine*, vol. 213, no. 5, pp. 809–825, May 2016, doi: 10.1084/jem.20151248.
- [3] M. Coleman, L. Martinez, G. Theron, R. Wood, and B. Marais, "Mycobacterium tuberculosis Transmission in High-Incidence Settings—New Paradigms and Insights," *Pathogens*, vol. 11, no. 11, p. 1228, Oct. 2022, doi: 10.3390/pathogens11111228.
- [4] Y. Ayalew, F. A. Yehualashet, W. A. Bogale, and M. B. Gobeza, "Delay for Tuberculosis Treatment and Its Predictors among Adult Tuberculosis Patients at Debremarkos Town Public Health Facilities, North West Ethiopia," *Tuberculosis Research and Treatment*, Sep. 2020, doi: 10.1155/2020/1901890.
- [5] Zumla, A. J. T. George, V. K. Sharma, N. Herbert, A. Oxley, and M. J. Oliver, "The WHO 2014 Global tuberculosis report—further to go," *The Lancet Global Health*, vol. 3, no. 1, pp. e10–e12, Jan. 2015, doi: 10.1016/s2214-109x(14)70361-4.
- [6] K. J. Seung, S. Keshavjee, and M. W. Rich, "Multidrug-Resistant Tuberculosis and Extensively Drug-Resistant Tuberculosis," *Cold Spring Harbor Perspectives in Medicine*, vol. 5, no. 9, p. a017863, Apr. 2015, doi: 10.1101/cshperspect.a017863.
- [7] J. Caulfield and N. L. Wengenack, "Diagnosis of active tuberculosis disease: From microscopy to molecular techniques," *Journal of Clinical Tuberculosis and Other Mycobacterial Diseases*, vol. 4, pp. 33–43, Aug. 2016, doi: 10.1016/j.jctube.2016.05.005.
- [8] "Sputum smear microscopy in tuberculosis: Is it still relevant?," *Indian Journal for Medical Research*.
- [9] J. O. Sekyere, N. E. Maningi, and P. B. Fourie, "Mycobacterium tuberculosis, antimicrobials, immunity, and lung–gut microbiota crosstalk: current updates and emerging advances," *Annals of the New York Academy of Sciences*, vol. 1467, no. 1, pp. 21–47, May 2020, doi: 10.1111/nyas.14300.
- [10] K. R. Barth, D. G. Remick, and C. A. Genco, "Disruption of immune regulation by microbial pathogens and resulting chronic inflammation," *Journal of Cellular Physiology*, vol. 228, no. 7, pp. 1413–1422, Jul. 2013, doi: 10.1002/jcp.24299.
- [11] K. R. Barth, D. G. Remick, and C. A. Genco, "Disruption of immune regulation by microbial pathogens and resulting chronic inflammation," *Journal of Cellular Physiology*, vol. 228, no. 7, pp. 1413–1422, Jul. 2013, doi: 10.1002/jcp.24299.
- [12] T. Cohen, B. D. Sommers, and M. Murray, "The effect of drug resistance on the fitness of Mycobacterium tuberculosis," *Lancet Infectious Diseases*, vol. 3, no. 1, pp. 13–21, Jan. 2003, doi: 10.1016/s1473-3099(03)00483-3.

- [13]M. De Martino, L. Lodi, L. Galli, and E. Chiappini, "Immune Response to *Mycobacterium tuberculosis*: A Narrative Review," *Frontiers in Pediatrics*, vol. 7, Aug. 2019, doi: 10.3389/fped.2019.00350.
- [14]L. D. Jasenosky, T. J. Scriba, W. A. Hanekom, and A. E. Goldfeld, "T cells and adaptive immunity to *Mycobacterium tuberculosis* in humans," *Immunological Reviews*, vol. 264, no. 1, pp. 74–87, Mar. 2015, doi: 10.1111/imr.12274.
- [15]C. R. Kathryn, R. S. D. Beatriz, C. C. Priscila, S. Alvarez-Arguedas, and M. U. Shiloh, "Pathogenicity and virulence of *Mycobacterium tuberculosis*," *Virulence*, vol. 14, no. 1, Nov. 2022, doi: 10.1080/21505594.2022.2150449.
- [16]C. Bussi and M. G. Gutierrez, "*Mycobacterium tuberculosis* infection of host cells in space and time," *Fems Microbiology Reviews*, vol. 43, no. 4, pp. 341–361, Jul. 2019, doi: 10.1093/femsre/fuz006.
- [17]B. Patterson and R. Wood, "Is cough really necessary for Tb transmission?," *Tuberculosis*, vol. 117, pp. 31–35, Jul. 2019, doi: 10.1016/j.tube.2019.05.003.
- [18]L. D. Maxim, R. Niebo, and M. J. Utell, "Screening tests: a review with examples," *Inhalation Toxicology*, vol. 26, no. 13, pp. 811–828, Nov. 2014, doi: 10.3109/08958378.2014.955932.
- [19]J. W. Uzorka, J. Wallinga, L. J. M. Kroft, T. H. M. Ottenhoff, and S. M. Arend, "Radiological Signs of Latent Tuberculosis on Chest Radiography: A Systematic Review and Meta-Analysis," *Open Forum Infectious Diseases*, vol. 6, no. 7, Jul. 2019, doi: 10.1093/ofid/ofz313.
- [20]J. Caulfield and N. L. Wengenack, "Diagnosis of active tuberculosis disease: From microscopy to molecular techniques," *Journal of Clinical Tuberculosis and Other Mycobacterial Diseases*, vol. 4, pp. 33–43, Aug. 2016, doi: 10.1016/j.jctube.2016.05.005.
- [21]P. Nahid, "Advances in the Diagnosis and Treatment of Tuberculosis," *Annals of the American Thoracic Society*, vol. 3, no. 1, pp. 103–110, Jan. 2006, doi: 10.1513/pats.200511-119jh.
- [22]M. Van Cleeff, L. Kivihya-Ndugga, H. Meme, J. Odhiambo, and P. R. Klatser, "The role and performance of chest X-ray for the diagnosis of tuberculosis: A cost-effectiveness analysis in Nairobi, Kenya," *BMC Infectious Diseases*, vol. 5, no. 1, Dec. 2005, doi: 10.1186/1471-2334-5-111.
- [23]J. Bomanji, N. K. Gupta, P. Gulati, and C. J. Das, "Imaging in Tuberculosis," *Cold Spring Harbor Perspectives in Medicine*, vol. 5, no. 6, p. a017814, Jun. 2015, doi: 10.1101/cshperspect.a017814.
- [24]S. Bhalla, A. Goyal, R. Guleria, and A. Gupta, "Chest tuberculosis: Radiological review and imaging recommendations," *Indian Journal of Radiology and Imaging*, vol. 25, no. 03, pp. 213–225, Jul. 2015, doi: 10.4103/0971-3026.161431.
- [25]L. B. Gadkowski and J. E. Stout, "Cavitary Pulmonary Disease," *Clinical Microbiology Reviews*, vol. 21, no. 2, pp. 305–333, Apr. 2008, doi: 10.1128/cmr.00060-07.

- [26]W. Raghupathi and V. Raghupathi, "Big data analytics in healthcare: promise and potential," *Health Information Science and Systems*, vol. 2, no. 1, Feb. 2014, doi: 10.1186/2047-2501-2-3.
- [27]Jensen-Doss *et al.*, "Monitoring Treatment Progress and Providing Feedback is Viewed Favorably but Rarely Used in Practice," *Administration and Policy in Mental Health*, vol. 45, no. 1, pp. 48–61, Jan. 2018, doi: 10.1007/s10488-016-0763-0.
- [28]Y. T. Kwon, "Chest X-rays in culture-negative pulmonary tuberculosis: early determination is superior to late determination," *The Korean Journal of Internal Medicine*, Aug. 2020, doi: 10.3904/kjim.2020.399.
- [29]M. L. B. Bhatt, S. Kant, and R. Bhaskar, "Pulmonary tuberculosis as differential diagnosis of lung cancer," *South Asian Journal of Cancer*, vol. 01, no. 01, pp. 36–42, Jul. 2012, doi: 10.4103/2278-330x.96507.
- [30]D. Orjuela-Cañón, A. L. Jutinico, C. Awad, E. Vergara, and A. R. Palencia, "Machine learning in the loop for tuberculosis diagnosis support," *Frontiers in Public Health*, vol. 10, Jul. 2022, doi: 10.3389/fpubh.2022.876949.
- [31]F.-J. Zhang, "Application of machine learning in CT images and X-rays of COVID-19 pneumonia," *Medicine*, vol. 100, no. 36, p. e26855, Sep. 2021, doi: 10.1097/md.00000000000026855.
- [32]M. Javaid, A. Haleem, R. P. Singh, R. Suman, and S. Rab, "Significance of machine learning in healthcare: Features, pillars and applications," *International Journal of Intelligent Networks*, vol. 3, pp. 58–73, Jun. 2022, doi: 10.1016/j.ijin.2022.05.002.
- [33]J. Wang and F. Biljecki, "Unsupervised machine learning in urban studies: A systematic review of applications," *Cities*, vol. 129, p. 103925, Oct. 2022, doi: 10.1016/j.cities.2022.103925.
- [34]S. Uddin, A. O. Khan, E. Hossain, and M. A. Moni, "Comparing different supervised machine learning algorithms for disease prediction," *BMC Medical Informatics and Decision Making*, vol. 19, no. 1, Dec. 2019, doi: 10.1186/s12911-019-1004-8.
- [35]P. M. Pardalos and M. Mammadov, "Learning the naive Bayes classifier with optimization models," *International Journal of Applied Mathematics and Computer Science*, vol. 23, no. 4, pp. 787–795, Dec. 2013, doi: 10.2478/amcs-2013-0059.
- [36]H. Habehh and S. Gohel, "Machine Learning in Healthcare," *Current Genomics*, vol. 22, no. 4, pp. 291–300, Dec. 2021, doi: 10.2174/1389202922666210705124359.
- [37]H. Alaskar and T. Saba, "Machine Learning and Deep Learning: A Comparative Review," in *Algorithms for intelligent systems*, Springer Nature, 2021, pp. 143–150. doi: 10.1007/978-981-33-6307-6_15.
- [38]S. Wang, Z. Chen, and S. Chen, "Applicability of deep neural networks on production forecasting in Bakken shale reservoirs," *Journal of Petroleum Science and Engineering*, vol. 179, pp. 112–125, Aug. 2019, doi: 10.1016/j.petrol.2019.04.016.
- [39]J. Chai, H. Zeng, A. Li, and E. W. T. Ngai, "Deep learning in computer vision: A critical review of emerging techniques and application scenarios," *Machine Learning With Applications*, vol. 6, p. 100134, Dec. 2021, doi: 10.1016/j.mlwa.2021.100134.

- [40]C. Chen *et al.*, “Deep Learning on Computational-Resource-Limited Platforms: A Survey,” *Mobile Information Systems*, vol. 2020, pp. 1–19, Mar. 2020, doi: 10.1155/2020/8454327.
- [41]L. Alzubaidi *et al.*, “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions,” *Journal of Big Data*, vol. 8, no. 1, Jan. 2021, doi: 10.1186/s40537-021-00444-8.
- [42]K.-H. Kim, S. H. Choi, and S.-H. Park, “Improving Arterial Spin Labeling by Using Deep Learning,” *Radiology*, vol. 287, no. 2, pp. 658–666, Dec. 2017, doi: 10.1148/radiol.2017171154.
- [43]K. Fukushima, “Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position,” *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, Jan. 1980, doi: 10.1007/bf00344251.
- [44]W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and A. Alsaedi, “A survey of deep neural network architectures and their applications,” *Neurocomputing*, vol. 234, pp. 11–26, Apr. 2017, doi: 10.1016/j.neucom.2016.12.038.
- [45]Adeel, M. Gogate, and A. Hussain, “Contextual deep learning-based audio-visual switching for speech enhancement in real-world environments,” *Information Fusion*, vol. 59, pp. 163–170, Aug. 2018, doi: 10.1016/j.inffus.2019.08.008.
- [46]S. Indolia, A. K. Goswami, S. Mishra, and P. Asopa, “Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach,” *Procedia Computer Science*, vol. 132, pp. 679–688, Jan. 2018, doi: 10.1016/j.procs.2018.05.069.
- [47]Naik and L. Samant, “Correlation Review of Classification Algorithm Using Data Mining Tool: WEKA, Rapidminer, Tanagra, Orange and Knime,” *Procedia Computer Science*, vol. 85, pp. 662–668, Jan. 2016, doi: 10.1016/j.procs.2016.05.251.
- [48]M. Peker, O. Özkaraca, and A. Sasar, “Use of Orange Data Mining Toolbox for Data Analysis in Clinical Decision Making,” in *Advances in bioinformatics and biomedical engineering book series*, IGI Global, 2018, pp. 143–167. doi: 10.4018/978-1-5225-5149-2.ch007.
- [49]Jovic, K. Brkić, and N. Bogunović, *An overview of free software tools for general data mining*. 2014. doi: 10.1109/mipro.2014.6859735.
- [50]Zupan and J. Demšar, “Open-Source Tools for Data Mining,” *Clinics in Laboratory Medicine*, vol. 28, no. 1, pp. 37–54, Mar. 2008, doi: 10.1016/j.cl.2007.10.002.

PAPER NAME

pLAG CHECKing (Final).pdf

Julius

WORD COUNT

7625 Words

CHARACTER COUNT

43213 Characters

PAGE COUNT

44 Pages

FILE SIZE

1.3MB

SUBMISSION DATE

May 31, 2023 11:26 AM GMT+5:30

REPORT DATE

May 31, 2023 11:27 AM GMT+5:30


● **20% Overall Similarity**

The combined total of all matches, including overlapping sources, for each database.

- 9% Internet database
- 2% Publications database
- Crossref database
- Crossref Posted Content database
- 17% Submitted Works database

● **Excluded from Similarity Report**

- Bibliographic material

Summary