Detection and Quantification of TB Bacilli in Sputum Smear Images

A PROJECT REPORT

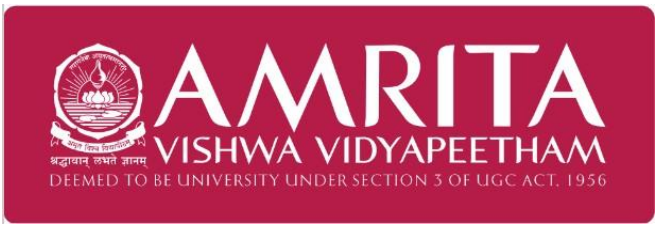
Submitted by

Jeevanantham
CH.EN. U4AIE21118
&
Sona B
CH.EN. U4AIE21152

in partial fulfillment for the award of the degree of
BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING

Under the guidance of
Ms. Rithani M

Submitted to



AMRITA VISHWA VIDYAPEETHAM
AMRITA SCHOOL OF COMPUTING
CHENNAI – 601103
April 2025

ABSTRACT

Tuberculosis (TB) remains a significant global health challenge, with millions of new cases annually, primarily in low and middle-income countries. The traditional diagnostic methods for detecting Mycobacterium tuberculosis bacilli in sputum smear images are labor-intensive, time-consuming, and susceptible to human error, particularly when dealing with low bacillary loads. This project proposes an automated system using advanced image processing and deep learning techniques to detect TB bacilli in Ziehl-Neelsen (ZN) stained sputum smear images. The methodology involves preprocessing the images for color-based segmentation, feature extraction using contour and geometric properties, and classification using machine learning models such as ResNet50, DenseNet, EfficientNet, Vision Transformer (ViT), and Inception 202V3. The system optimizes the most effective model based on accuracy, precision, recall, and F1 score, ensuring robustness across varied datasets. By minimizing expert dependency and improving diagnostic speed and accuracy, the solution aims to enhance TB diagnosis, particularly in resource-limited settings and supports global efforts to combat the spread of TB.

Keywords: Tuberculosis Detection, Sputum Smear Microscopy, Automated Diagnosis, Deep Learning, Image Segmentation, Machine Learning Models, Ziehl-Neelsen Staining, Convolutional Neural Networks (CNN), Vision Transformer (ViT), Medical Image Processing, Computer-Aided Diagnosis (CAD)

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

ACRONYM		ABBREVIATION
TB	-	Tuberculosis
ZN	-	Ziehl-Neelsen
CAD	-	Computer-Aided Diagnosis
RGB	-	Red, Green, Blue (Color space)
HSV	-	Hue, Saturation, Value (Color space)
YCbCr	-	Luminance, Chrominance Blue, Chrominance Red (Color space)
DPI	-	Dots Per Inch
CMOS	-	Complementary Metal-Oxide-Semiconductor
FMF	-	Focus Measure Function
SIFT	-	Scale-Invariant Feature Transform
RANSAC	-	Random Sample Consensus
COR	-	Correlation
SSIM	-	Structural Similarity Index Measure
FSIM	-	Feature Similarity Index Measure
ZNSM-Idb	-	Ziehl-Neelsen Sputum Smear Microscopy Image Database
CNN	-	Convolutional Neural Network
ViT	-	Vision Transformer
MP	-	Megapixel

CHAPTER 1

INTRODUCTION

Tuberculosis (TB) continues to be a leading cause of infectious disease-related deaths worldwide, especially in low-resource settings. Traditional diagnostic methods, reliant on expert analysis and microscopy, are time-consuming and prone to errors. This project proposes an AI-powered automated system to detect TB bacilli in sputum smear images, leveraging advanced deep learning models like ResNet50 and DenseNet. By automating the diagnostic process, the project aims to reduce human error, improve diagnostic speed, and expand access to healthcare in underserved regions. This technology aligns with global efforts to control TB and represents a significant advancement in medical AI applications.

1.1 BACKGROUND ON TUBERCULOSIS AND IT'S GLOBAL IMPACT

Tuberculosis (TB) is a bacterial infection caused by Mycobacterium tuberculosis that primarily affects the lungs but can also impact other parts of the body. It is a major public health issue, with millions of new cases diagnosed globally each year, particularly in low and middle-income countries. According to the World Health Organization (WHO), TB is one of the top 10 causes of death worldwide and the leading cause from a single infectious agent, ranking above HIV/AIDS. The spread of TB is exacerbated by socioeconomic factors such as poverty, malnutrition, and overcrowding, especially in regions with limited healthcare infrastructure. TB's airborne nature further complicates containment efforts, making it essential to have early, accurate, and widespread diagnostic capabilities.

1.2 CHALLENGES IN TRADITIONAL TB DIAGNOSTICS

Traditional TB diagnostics include the Ziehl-Neelsen (ZN) staining technique and fluorescence microscopy, which involve staining sputum smear samples to visualize bacilli under a microscope. While these methods have been used for decades, they require skilled technicians and significant time for manual examination. The dependency on experts creates bottlenecks, especially in resource-limited settings where access to trained personnel is scarce. Additionally, these methods are prone to variability and errors, particularly when the bacterial load in the sample is low. This leads to potential delays in diagnosis, increasing the risk of transmission and worsening patient outcomes. The inconsistency in diagnostic results due to human factors and the labor-intensive nature of these methods indicate a pressing need for a more automated and scalable approach.

1.3 AI AND IT'S ROLE IN REVOLUTIONIZING TB DIAGNOSIS

The development of artificial intelligence (AI) and deep learning (DL) technologies in recent years has shown great promise in medical diagnostics, particularly in automating image analysis tasks traditionally performed by human experts. AI-based systems can process images rapidly, identify patterns, and classify objects with high precision. For TB diagnostics, this means that automated systems can analyze sputum smear images, detect bacilli, and provide diagnostic outcomes without the need for human intervention, significantly speeding up the process. Moreover, AI models can be trained to recognize a wide variety of image conditions, thus improving consistency and reducing diagnostic errors even when bacillary loads are low. The application of AI in this domain not only addresses the limitations of traditional methods but also provides scalable solutions that can be deployed in mobile health clinics and other remote healthcare settings.

1.4 EXPANDING ACCESS AND IMPROVING HEALTH OUTCOMES

The AI-driven diagnostic system proposed in this project aims to have a transformative impact on TB management, particularly in underserved regions. By developing a robust system that can automate the detection of TB bacilli in sputum smear images, the project contributes to reducing the dependency on skilled technicians. This is crucial for expanding access to accurate diagnostics in areas where healthcare resources are limited. The AI system's ability to operate efficiently in various settings allows for deployment in mobile clinics, which can reach remote communities that otherwise lack diagnostic facilities. This project aligns with the WHO's End TB Strategy by providing a technological solution that supports early detection and timely treatment, thus reducing transmission rates and improving patient outcomes. The ultimate goal is to make high-quality TB diagnostics accessible to all, regardless of location or resource availability.

1.5 SIGNIFICANCE OF THE PROJECT FOR GLOBAL HEALTH INITIATIVES

Beyond its immediate application in diagnosing TB, the AI-based system developed through this project holds potential for broader applications in infectious disease management. The success of this project could serve as a model for integrating AI into other diagnostic processes, promoting the use of technology to bridge healthcare gaps worldwide. By demonstrating that AI can enhance diagnostic capabilities in low-resource settings, this project supports global health initiatives aimed at equitable healthcare access. Furthermore, the development and deployment of such technologies contribute to the international effort to eradicate TB and other infectious diseases, showcasing how innovation can be a driving force in achieving global health goals.

CHAPTER 2

LITERATURE REVIEW

Detection of tuberculosis, particularly automated processing of sputum smear pictures, has been one of the significant thrust areas toward enhancement of diagnostic efficiency. Several methods have been proposed using machine learning or image processing, and results vary from excellent to merely workable depending upon the design or application of the same in clinical practice. Therefore, within this context, Sotaquira, in her early work, would establish an algorithm that identifies and measures clusters of bacilli in sputum smear samples, as later to form the basis of developing automated methods for diagnosing tuberculosis through digital image processing technique

2.1 EARLY IMAGE PROCESSING TECHNIQUES FOR TB DETECTION

An early study established an algorithm to identify and measure bacilli clusters in sputum smear samples, forming the groundwork for future automated diagnostic methods[1]. Another research used K-nearest neighbor classifiers to improve the accuracy of TB bacilli detection in microscope images[2]. Deep learning, specifically convolutional neural networks (CNNs), was employed to automate TB bacilli classification, significantly improving precision and reducing the need for manual intervention[3]. Conventional image processing techniques were also explored to detect bacilli in smear images[4]. A recent review emphasized the growing role of artificial intelligence (AI) and image processing in clinical applications, highlighting its increasing accuracy in diagnosing TB[5].

2.2 RISE OF AUTOMATED DETECTION TECHNIQUES

AI-based methods, particularly deep learning models, were explored for TB bacilli detection, which reduced errors associated with manual microscopy[6]. A study demonstrated that convolutional neural networks (CNNs) used for pneumonia detection could also be applied to TB diagnosis, showcasing the versatility of deep learning in medical imaging[7]. Another research critically reviewed AI-based diagnostic methods, emphasizing the use of digitized smear images for improved diagnostic accuracy[8]. Deep learning models, specifically designed to analyze medical images, were highlighted for their ability to reduce manual errors in diagnosing tuberculosis[9]. Advanced AI techniques were proposed for detecting TB in microscopic images, focusing on reducing the margin of error typically associated with traditional methods[10].

2.3 DEEP LEARNING INNOVATIONS IN TB DETECTION

Research highlighted the potential of using bioaerosol analysis for detecting Mycobacterium tuberculosis, introducing a new diagnostic approach beyond traditional microscopy[11]. A crowdsourcing model was proposed to analyze sputum smear samples remotely, decentralizing the TB testing process and enhancing accessibility to diagnostic tools in underserved areas[12]. Quantitative approaches using local linear estimators were developed to quantify TB bacilli in sputum smear images, improving diagnostic accuracy[13]. The use of advanced machine learning models like Faster R-CNN, along with data augmentation techniques, was reported to enhance TB bacilli detection in smear images[14].

2.4 CHALLENGES IN IMPLEMENTING AI FOR TB DIAGNOSIS

Machine learning algorithms, originally developed for other diseases like pneumonia and Gram-stained sputum smears, demonstrated their applicability in TB diagnosis[15]. Deep learning was surveyed as a tool for lung disease detection, presenting a comprehensive review of its application in medical diagnostics, including TB[16]. A publicly available database of Ziehl-Neelsen sputum smear microscopy images was created, serving as a valuable resource for developing TB detection algorithms[17]. Additional studies explored the potential of using hybrid models for detecting diseases beyond TB, such as gastric cancer, through medical image analysis[18].

2.5 FUTURE DIRECTIONS AND POTENTIAL OF AI IN TB DETECTION

The application of quantum-inspired convolutional neural networks (QCNet) for thyroid nodule classification was explored, demonstrating the potential for using these advanced models in medical diagnostics. Another study emphasized the use of explainable AI (XAI) to understand the decision-making processes of CNN models in image classification tasks, underscoring the need for transparency in AI-based medical tools[19]. Further advancements in AI and deep learning models focused on their application in complex medical image analysis, illustrating how these models could be adapted for TB bacilli detection and other medical diagnostics[20].

CHAPTER 3

METHODOLOGY

The overall scope of this project is to annotate the train images, apply image processing techniques to identify the potential bacilli objects, classify the objects into either of the 3 classes (i.e., single bacillus, bacilli cluster and artifacts), count the number of single bacilli in each bacilli cluster and hence calculate the total number of single bacilli on a given image.

Using the ground truth information available on the microscopic images, annotation is performed on each image and a corresponding annotation file is generated for each image. Image processing operations, i.e., image segmentation and image postprocessing are performed on the microscopic images to separate the potential TB objects from the background. Since, bacilli objects belonging to different classes can be distinguished based on their geometrical features (like Relative convex area, Eccentricity, Roughness etc.,), these features can be used as input to the Machine algorithms to classify the bacilli. For each object in the postprocessed image, geometric features are extracted and class label is assigned based on the ground truth information available in the annotation file. Similar procedure is applied to all the images and a final labelled data matrix is created using the afore-mentioned features and class labels. All the machine learning models are trained using the labelled data matrix. For a given test image, each potential TB object is classified into one of the 3 classes (i.e., single bacillus, bacilli cluster and artifacts). Concave points are detected on the bacilli cluster and the same are used to count the number of single bacilli in each bacilli cluster. Finally, an overall count of single bacilli is obtained for a given test image.

3.1 Dataset

For the development of automated detection techniques of TB bacilli on Ziehl-Neelsen sputum smear microscopic images, a standard sputum smear microscopic image database (ZNSM-iDB) [1] is developed by Mohammed Imran Shan and his team. Images belonging to specific folders of this database are used for this project. This section describes the structure, features, and contents of the key files used from the MIMIC-IV dataset and explains how each dataset is used in the model.

According to the World Health Organization (WHO) guidelines, 300 viewfields of a smear slide should be examined in CM within 24 hours of collection of a specimen for accurate diagnosis. As manual identification and counting of bacilli using CM is a very time-consuming and laborintensive task, it takes 40 min to 3 h to analyze even 40 to 100 viewfield images from a single slide to diagnose a patient as tuberculosis positive or negative. Therefore, the sensitivity of tuberculosis detection varies, and it relies on the experience of microbiologists.8 The effectiveness of diagnosis is compromised to a significant extent for extrapulmonary, pediatric, or HIV-patients co-infected tuberculosis. All of these shortcomings can be addressed through an

automated microscope, which will not only increase the accuracy but also reduces the time of diagnosis. Efforts were made to increase the sensitivity of this diagnostic test by incorporating automated methods. However, the attainment of success is limited mainly because of the inadequacy of data and dependency of automated methods on image contents.

3.1.1 Key Components of Automated Microscopy

The automated microscopy for bacilli detection requires efficient algorithms in the following three domains:

i. **Autofocusing:** In a stack of images captured from a single viewfield with different focuses, an image with the best average focus over the entire viewfield is defined as the focused one. The maximum value of the focus measure function (FMF) corresponds to the best-focused image. This method would facilitate automated capturing of the best-focused image. In recent years, various autofocusing algorithms have been proposed and implemented in microscopy images for diverse biological applications.

ii. **Autostitching:** This method stitches viewfields of a smear-slide to form a mosaic or slide map. The WHO recommended analysis of 300 viewfields can be achieved faster and efficiently by automated stitching of overlapping viewfields followed by detection of bacilli in a mosaic (stitched-image) by segmentation methods.17 Bacilli segmentation algorithms use bacilli shape and size as the potential features to segment the bacilli from other objects. Autostitching also facilitates automatic detection of bacilli on the edge by joining half bacillus structures on the boundaries of two different viewfields. Though many autostitching methods were developed, they were not validated on diverse datasets.

iii. **Automatic bacilli segmentation and grading:** It is a process of segmentation and counting of bacilli either from viewfield or mosaic. Pattern recognition and machine learning techniques have been used to detect the bacilli in images, but their efficacy and scopes are limited due to their implementation on nonunified and limited datasets.

Databases and tools have already provided a better diagnosis of cancer and other diseases. Databases were used to develop algorithms/methods, which were implemented in computer- aided diagnosis (CAD) systems to obtain a second opinion about a disease. The databases and CADs are also effectively used for the early detection of diseases. Keeping in view these accomplishments of databases, the Ziehl–Neelsen sputum smears microscopy image database (ZNSM-iDB) has been developed. This database contains diverse categories of image datasets with both medium and high noise backgrounds. It possesses datasets for all three processes required for automated microscope development. Standard protocols were used to acquire the images, and the datasets were validated by various automated microscopy algorithms to establish their robustness. This database can be used to develop and evaluate efficient and robust algorithms related to automated microscopy.

3.1.2 Data collection

Digital images of viewfields from 10 different ZN-stained sputum smear slides (belonging to tuberculosis positive patients) were acquired under the guidance of two expert microscopists. Triplicate data for each category were collected using three different microscopes at 100× magnification. The objective lens with 100× magnification was used as the typical Mycobacterium tuberculosis bacilli width and length of about 0.5 and 2 − 4 μm, respectively. Images were acquired in RGB (red, green, and blue) color space with ".jpg" file-format. Image dimensions in pixels and DPI (dots per inch) are also provided. The dimensions of image represent the number of pixels in an image. Detailed configurations of all the microscopes and acquired image properties are mentioned below:

i. First datasets were acquired using a Labomed Digi 3 digital microscope (MS-1), which features an L × 400 trinocular microscope and an iVu 5100 digital camera module 5.0 megapixel CMOS sensor. The acquired images were 800 × 600 pixels with bit depth and resolution of 24 (eight per channel) and 120 DPI, respectively, at 100× magnification. The physical size (pixel pitch) of each single pixel is 2.2 μm.
ii. Second datasets were acquired using a Motic BA210 digital microscope (MS-2), which features a Siedentopf type Binocular head and Moticam 2500 digital camera module 5.0 megapixel CMOS sensor. The acquired images were 1280 × 1024 and 2592 × 1944 pixels with bit depth and resolution of 24 and 96 DPI, respectively, at 100× magnification. The physical size of a pixel is 2.2 μm.
iii. Third datasets were acquired using an Olympus CH20i digital microscope (MS-3), which features a trinocular microscope and a smartphone digital camera with 16- megapixel BSI-CMOS sensor. The smartphone was attached to the microscope using HY0088 microscope mobile phone interface. The acquired images were 5312 × 2988 (16 MP), 3984 × 2988 (12 MP), and 2048 × 1152 (2.4 MP) pixels with bit depth and resolution of 24 and 72 DPI, respectively, at 100× magnification. The physical size of a pixel is 1.12 μm.

3.1.3 Description of Datasets

The architecture of ZNSM-iDB with its applications is presented in *Fig. 1*. This database contains seven different categories of digital images in triplicate sets (one set from each microscope), which may be separately visualized or retrieved for further processing and applications (*Table 1*). Multiple and diverse datasets were provided so that the robustness of developed algorithms can be evaluated.

		No. of digital images from different microscopes (MS)		
Group	Category of data	MS-1	MS-2	MS-3

1	Autofocusing dataset	9 stacks	10 stacks	30 stacks
2	Overlapping view fields for auto-stitching	7 sets (50 to 90 images/set)	6 set (50 images/set)	10 sets (50 images/set)
3	Manually segmented bacilli in the view field	2 set (50 images/set)	2 set (50 images/set)	2 set (50 images/set)
4	View fields without bacilli	50	50	50
5	Single or few bacilli	100	100	100
6	Overlapping (occluded) bacilli	200	200	200
7	Over-stained view fields with bacilli and artifacts	250	250	250

Table 3.1 Category-wise presentation of datasets available in ZNSM-iDB

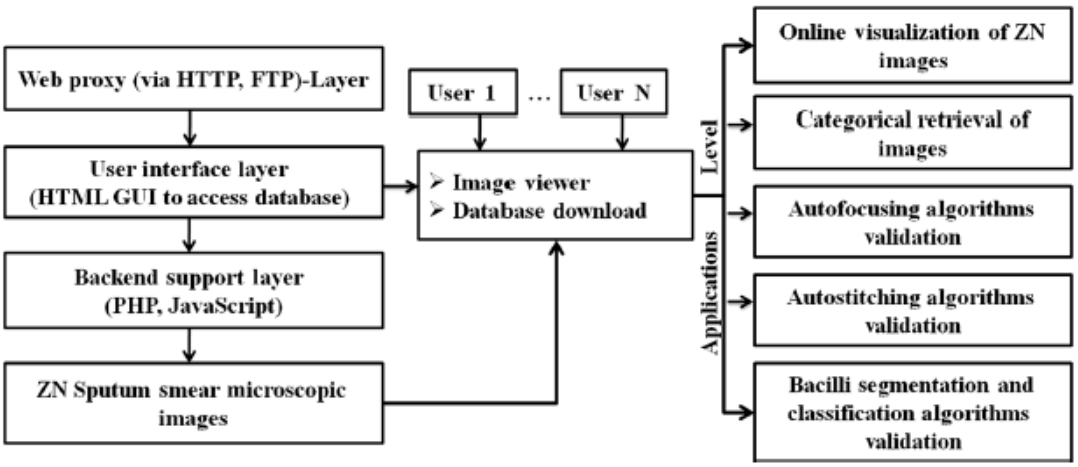


Fig. 3.1 Architecture and applications of ZNSM-iDB.

A detailed description for each category of data is given below.

3.1.3.1 Autofocusing dataset

In this category, every stack of images is restricted to a single viewfield. Each stack contains at least 20 images captured at different focus lengths, in which one is marked as the best focused, while the others are unfocused to different extents. The 10th image is the best-focused one in most of the stacks.

3.1.3.2 Overlapping viewfields for auto-stitching

Adjacent overlapping viewfields can be stitched to make the mosaic or slide map using image processing

techniques. In this dataset, 10 overlapping viewfield images were acquired in a row and then the slide was moved left or right to take the images of the next row *(Fig. 2)*.

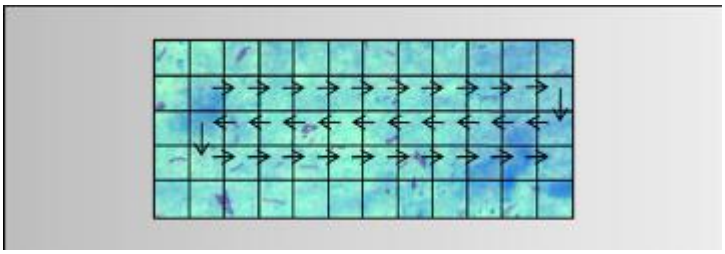


Fig. 3.2 A depiction of direction in which the images were acquired from a ZN-stained slide. Each square box corresponds to a viewfield.

3.1.3.3 Manually segmented bacilli in a viewfield

This dataset contains viewfields with manually segmented or marked bacilli. Different shapes were used for marking such as a circle or oval shape for single bacillus, square or rectangle for occluded bacilli, diamond for unclassified red structures, and hexagon for the *artifacts [Fig. 3(a)]*.

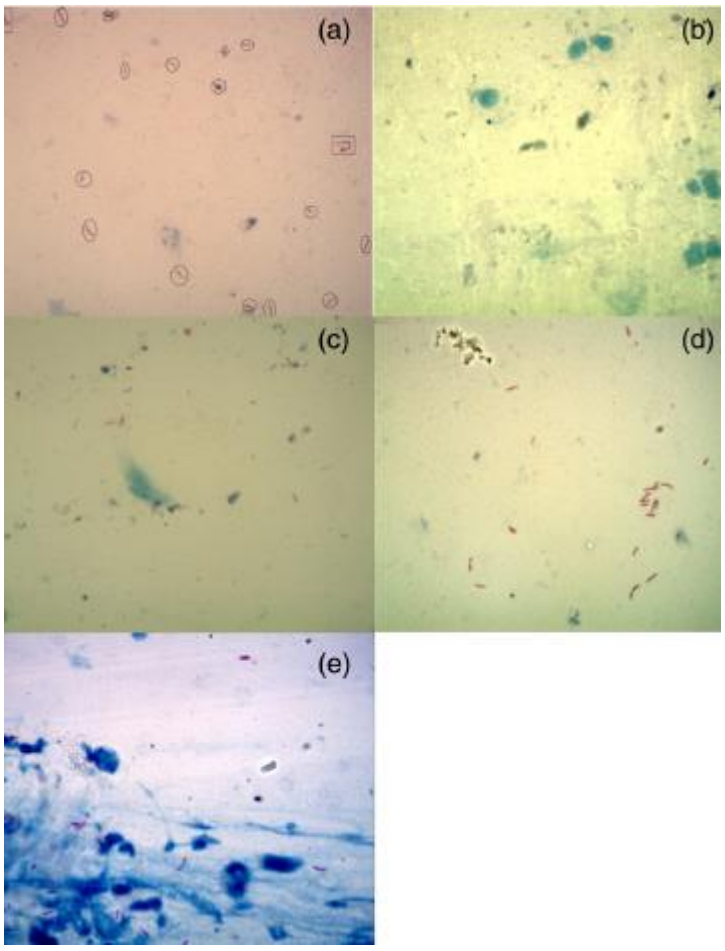


Fig. 3.3 Sample images of five different category datasets available in ZNSM-iDB. (a) Manually

segmented viewfield, (b) viewfield without bacilli, (c) viewfield with single or few bacilli, (d) viewfield with occluded bacilli, and (e) over-stained viewfields with bacilli and artifacts.

3.1.3.4 Viewfields without bacilli

Images in this group range from medium to very high-density background depending upon the presence of over-staining and artifacts, but the viewfield does not contain any bacilli *[Fig. 3(b)]*.

3.1.3.5 Viewfields with a single or few bacilli

In this group of images, the number of bacilli in each viewfield varies from 1 to 10. This group also has medium to high-density backgrounds *[Fig. 3(c)]*.

3.1.3.6 Viewfields with overlapping (occluded) bacilli

In many instances, two or more bacilli are overlapped at the same position and form an occluded bacilli cluster. Images in this category are diverse in terms of medium to high-density backgrounds *[Fig. 3(d)]*.

3.1.3.7 Over-stained viewfields with bacilli and artifacts

Sometimes, the quality of ZN-stained CM images is not good due to the presence of artifacts and over-staining. Therefore, bacilli detection is difficult using a segmentation method, and only robust methods can produce a better performance. In this category, more than 200 images from each microscope are provided that contain over-stained (blue) regions with artifacts and/or bacilli *[Fig. 3(e)]*.

3.1.4 Data Validation

ZNSM-iDB resource contains autofocusing, autostitching, and bacilli segmentation and classification image datasets. Some algorithms/methods reported in these three domains were also implemented on the datasets.

Twenty-four FMFs were implemented on autofocusing datasets to determine the best-focused one in each stack of images. The performance of FMFs in classifying a stack is determined using different criteria such as accuracy, focus error, false maximum, and full-width at half-maximum. The robustness of FMFs to the different imaging conditions (median filtering, noise addition, contrast reduction, saturation increment, and nonuniform illumination addition) was also determined.

Autostitching of overlapping viewfields datasets was performed and reported. The overlapping subparts of the viewfields were stitched together using scale-invariant feature transform (SIFT) feature extraction and random-sample-consensus (RANSAC) selection method. The divide and conquer algorithm was implemented

for faster stitching and mosaic formation. Comparisons of similarities between the original and stitched images were performed using correlation (COR), structural similarity (SSIM), and feature similarity (FSIM) measures.

Bacilli segmentation and classification using the watershed algorithm32 was performed on ZNSM-iDB database. Forty images from three microscopes (MS-1, MS-2, and MS-3) were randomly extracted and grouped into medium- and high-density background datasets. The shape and size of the objects were determined to filter the true bacilli. Similarly, the watershed algorithm was implemented on 30 randomly extracted images from the smartphone enabled microscope (MS-3) to segment bacilli. In both studies, sensitivity and specificity of the watershed algorithm were calculated. In an another study, images were divided into four groups based on the infection level (*Table 2*) and sensitivity and specificity of the watershed segmentation method for classifying an image as TB positive or negative were determined for each group.

Number of bacilli	Number of view fields to be examined	Grading
1 in 9 in 100 view fields	100	Scanty
10 to 99 in 100 view fields	100	1+
1 to 10 in each view field	50	2+
>10 in each view field	20	3+

Table 3.2 Grading of view fields based on infection level.

The sensitivity and the precision rate of this segmentation method for identifying true bacilli were determined for each group. Furthermore, the discordance rate was calculated for the watershed segmentation method to evaluate the percent of pairs where the observation with TB-positive has a lower predicted probability than TB-negative. The predicted probability was calculated in a binary logistic regression model.

3.2 Data Preprocessing

3.2.1 **Image Annotation**

Using the ground truth details available on the microscopic image and the online image annotation tool, bounding box-based annotation is performed on each image to label each object into one of 4 classes, i.e., single bacillus, bacilli cluster, unclassified red structures and artifacts.

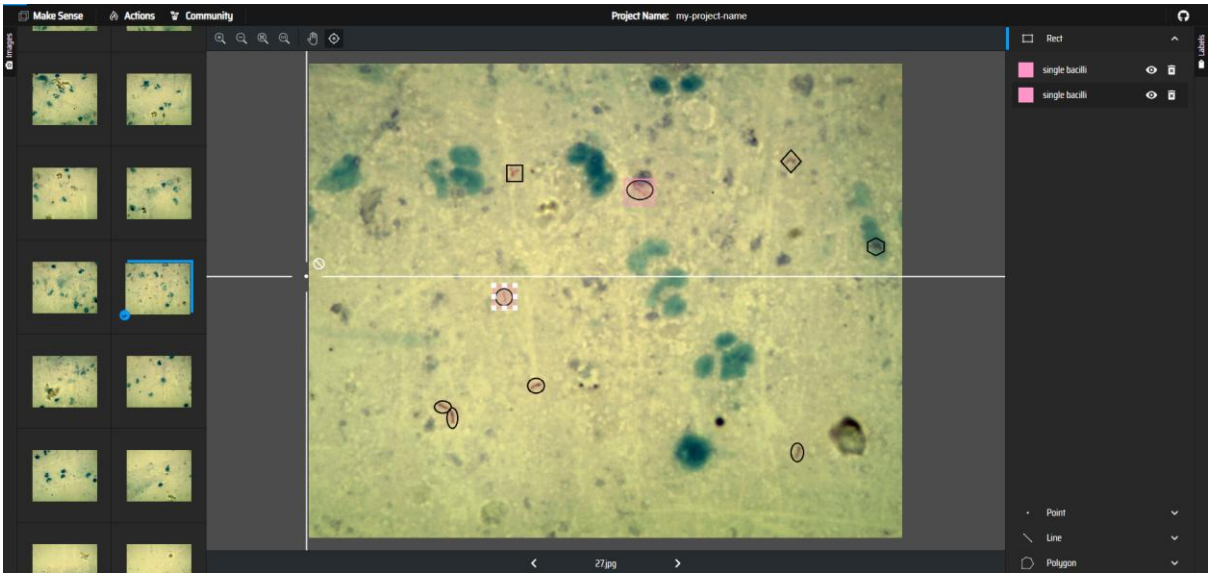


Fig 3.4 Manually Annotating images

An annotation file (in .csv format) is generated for each image. The annotation file consists of details related to class label, coordinates of top left and bottom right corners of bounding box around each object etc., as shown below.

```
Area,Roughness,Relative convex area,Circularity,Compactness,Eccentricity,Class
15.5,1.0,1.0,0.7382922197978818,1.3544772289131861,0.5993709496466937,3
228.5,1.1255157151765909,1.336980306345733,0.4338343718689326,2.3050271367205406,0.4131061971161571,3
74.0,1.1843790393454765,4.675675675675675,0.08504317843388838,11.75873266281303,0.5953386852472928,3
12.0,1.077010253705712,1.2916666666666667,0.4413051398948433,2.266005785085053,0.4691730228810838 7,3
113.5,1.0944759372135369,1.1938325991189427,0.30807489362251106,3.245963954548388,0.2302849029652614,1
22.0,1.042871719453307 7,1.1136363636363635,0.4611289156219007,2.1685909647443204,0.4063295979645399,1
28.5,1.064859931203541,1.2456140350877194,0.5013307442382535,1.9946911524834747,0.4517810443540876 7,3
29.5,1.0647061706572567,1.2881355932203389,0.4647520141515739,2.151685134330285,0.9204823610025437,3
13.5,1.0722067339589492,1.6296296296296295,0.35373288928168695,2.8269918639193126,0.4930151994517307,3
83.0,1.2926443863767714,2.3072289156626504,0.19512024836935715,5.12504472681394,0.5942813248808162,3
32.0,1.0327282395034176,1.0625,0.7953589909308333,1.257293890435147,0.802210198522284 8,3
19.5,1.0298006210597177,1.0769230769230769,0.5980123401348733,1.6722062955665165,0.6147994365153023,3
15.5,1.0320856272814893,1.1935483870967742,0.5852833333392216,1.7085741948172217,0.3841950304237953,3
229.0,1.1777743971343773,1.2183406113537119,0.5214967638232842,1.9175574411404452,0.7255125009086779,1
12.5,1.0180073730171035,1.08,0.7743527505641299,1.2914011079207532,0.8515045855127673,3
33.0,1.073886591157015,1.3181818181818181,0.523612141034979 4,1.9098105670800245,0.6719783392388827,3
29.0,1.1417185453342162,1.5517241379310345,0.3799353267721231,2.6320268991458593,0.7392056611293113,3
12.0,1.1358388693043313,1.7083333333333333,0.2774395639866729,3.604388594151756,0.3462505559175352 3,3
53.5,1.0706677345674367,1.1962616822429906,0.5850287308255506,1.7093177604950645,0.5753678594259044,3
20.0,1.0531166794864952,1.325,0.4970993693317708,2.011670224696235,0.7001454936366974,3
19.0,1.0505899959692524,1.2894736842105263,0.4722444008651822,2.117547604943405,0.4221493096983126,3
42.5,1.063754790132165,1.4470588235294117,0.44283425322737363,2.258181233073109,0.4236982756535829,3
34.0,1.0243469712509918,1.0441176470588236,0.8450689278640103,1.183335426291903,0.9373244780304337,3
44.5,1.254371226456014,1.898876404494382,0.27355164260935094,3.655616871685407,0.743333832428615,3
17.5,1.073170199641299,1.4,0.49530741331104167,2.018948178698111,0.5645417468626833,3
80.0,1.1042649047640436,1.35625,0.4792786226626702,2.086469023893495,0.4556753934587152,1
13.0,1.0190873591792915,1.1538461538461537,0.5768543421982874,1.7335398676018998,0.5234047140924017,3
16.0,1.0948865650138007,1.4375,0.39767949546541664,2.514587780870294,0.3583423243829056 4,3
47.5,1.1036850218271332,1.2421052631578948,0.5300954464448159,1.8864527260263917,0.5659000877245203,3
31.0,1.2022699363143932,1.5161290322580645,0.37707011125020656,2.652026692554387,0.6807812837860517,3
24.0,1.0678437631438806,1.4166666666666667,0.35097774699308826,2.84918348404491,0.3354149714672725,3
58.5,1.2603213803746012,2.2051282051282053,0.227299458661616,4.3994825411736445,0.8117970988742302,3
24.0,1.0510171468726457,1.1875,0.5965192431981249,1.6763918539135292,0.8333823225068192,3
176.0,1.4191429008048941,1.6846590909090908,0.22080029178697788,4.528979522204494,0.4737148578427116 7,3
36.0,1.1455110284542522,2.0,0.2632333153277535,3.7989112387043162,0.5185399645194412,3
18.5,1.0650589697413924,1.1891891891891893,0.6391923625811166,1.564474262430029,0.8296588019868603,3
34.5,1.0077816858633781,1.0144927536231885,0.8145046663457662,1.2277400502644744,0.7651149390307046,3
24.0,1.0994524877668608,1.5416666666666667,0.4042590690710407,2.4736612645399165,0.6225942048783 54,3
40.5,1.1063382035685234,1.4197530864197532,0.2787669902030867,3.5872252997798704,0.2289930196471561 3,3
32.5,1.1047867365968584,2.1076923076923078,0.20731234422695932,4.823639439942033,0.4874293014923391,3
```

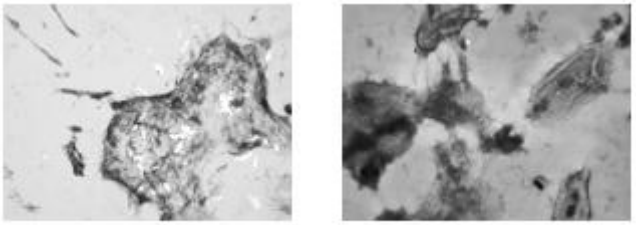
Fig 3.5 CSV Generated from features

3.2.2 **Image Segmentation**

Compared with the fluorescence microscopy images (where the samples are stained with Auramine and the bacilli are clearly defined), the samples obtained via Ziehl Neelsen method and acquired with a

conventional light mi croscope show bacilli and bacilli clusters that, depending on the illumination level used during the acquisition, may have different colors, varying from light fuchsia to dark purple. There are also objects, different from bacilli, that appear under the microscope during the acquisition process (such as food rests, fibers, pollen, etc.); these objects have blue tones (varying from light to dark blue) that also depend on the illumination level. The sample's background also varies from image to image (and is also an illumination-dependant variable), and may take colors such as white, cream and beige. Examples of these characteristics are shown in *Figure 6* (where the bacilli and clusters have been colored, for illustration purposes, with white):

As shown in *Figure 6*, both images backgrounds differ thanks to different illumination levels set during the acqui sition process. It can also be seen (*Figure 6*, left) that de pending on the level of infection there can be many bacilli and clusters (not always easily recognizable) that, as stated previously, make the diagnosis process a difficult and inaccurate task.



(a) High level of infection (b) Low level of infection

Fig 3.6. Typical conventional light microscopy sputum smear images

Besides the characteristics described above, it's important to note that in a sputum smear sample may appear not (a) High level of infection (b) Low level of infection only isolated bacilli but also bacilli agglomerations (or clusters), that make the diagnosis process even more difficult. This is because TB diagnosis relies on bacilli counting, and when there are bacilli clusters is not always a simple task to compute the exact number of bacilli present in a specific cluster.

Having defined the different characteristics and artifacts present in the digital images, is clear that the algorithm has to satisfy these requirements: a segmentation stage that make use of color information, with a high level of efficiency (correct detection of bacilli and bacilli clusters) and a low level of false positives, and also robust against different levels of illumination; a quantification stage able to determine, with relatively high level of accuracy, the number of bacilli present in the image (even when there are bacilli clusters); and a diagnosis stage able to correctly determine the level of infection. Since the number of images to be analyzed, in order to determine the level of infection, varies from 20 to 100, the algorithm must be efficient in terms of computing time. The algorithm, described next, have taken into account all the requirements stated previously.

3.2.2.1 **Color space selection and image segmentation.**

The goal of this stage is to segment most of the objects of interest (bacilli and bacilli clusters) discarding other arti facts present in the image (and, thus, reducing the number of false positives). Since color

tones of the bacilli differ from those on the artifacts and the background, is clear that a color segmentation algorithm has to be implemented.

RGB, HSV, YCbCr, Lab and YIQ color spaces were con sidered. To determine the correct color space to implement in the algorithm, each plane (R, G, B, H, S, V, Y, Cb, etc.) of the images dataset was analyzed. From this analysis it was established that RGB, HSV and YIQ color spaces were not adequate for the segmentation of the image, since all of them have a high number of false positives after the seg mentation stage.

Nevertheless, YCbCr and Lab color spaces offered bet ter results. Specifically the Cr plane (from the YCbCr im age) conveys most of the information related with bacilli 118 Authorized licensed use limited to: Politecnico di Milano. Downloaded on March 06,2010 at 03:43:12 EST from IEEE Xplore. Restrictions apply. and bacilli clusters and rejects most of the artifacts (such as food rests, fibers and pollen particles), but it also con tains artifacts that result from the varying illumination con ditions. On the other side, the Lab color space, and specif ically the a plane, is more robust against illumination ar tifacts but is also unable to reject other objects, different than bacilli, present in the image. Since both planes, Cr and a, contain information related with the bacilli, but also both differ in the type of artifacts detected (Cr plane con tains illumination artifacts, while a plane rejects most of them), it is possible to make a logical AND between both segmented images, in order to obtain the desired results (lo cate the bacilli and clusters and reject all other artifacts and background).

In order to guarantee an adequate segmentation stage, almost illumination-independent, a thresholding technique, based on the first derivative of the histogram, was imple mented. Since a change in the illumination conditions moves the histogram of an image from darker to brighter levels (or viceversa), it is clear that a threshold based on level intensity is not adequate in this case. Nevertheless, the first derivative of the histogram brings information re lated with differences between consecutive points of that histogram; these differences are robust against varying illu mination conditions and therefore guarantee a robust algo rithm. In this case the first derivative is computed over the Cr and a planes of the image. If H[n] (n = 0,1,….2,5,4) is the histogram, then its first derivative is computed as in (1):

$$\Delta H[n] = H[n+1] - H[n] \qquad (1)$$

The threshold is then selected as the maximum level of intensity (nmax) chosen between all the possible levels, n, that satisfy the condition:

$$\Delta H[n] \leq r \qquad (2)$$

In this case the parameter r was established by test ing all the images in the database (both for 640x480 and 1600x1200 image resolutions). The advantage of this method is, as stated previously, that the threshold level is se lected based on the first derivative of the histogram, which is computed between consecutive levels, and thus makes this threshold more robust against variations in the illumination

conditions.

Having described the color spaces selected and the thresholding process, the image segmentation algorithm can be summarized as follows:

Step 1: load the image in RGB space.

Step 2: convert the image to the YCbCr and Lab color spaces. Take only the Cr and a planes. Step 3: compute threshold levels for the Cr and a planes based on their first derivatives.

As an example, *Figure 7* shows the results of the proposed algorithm described above:

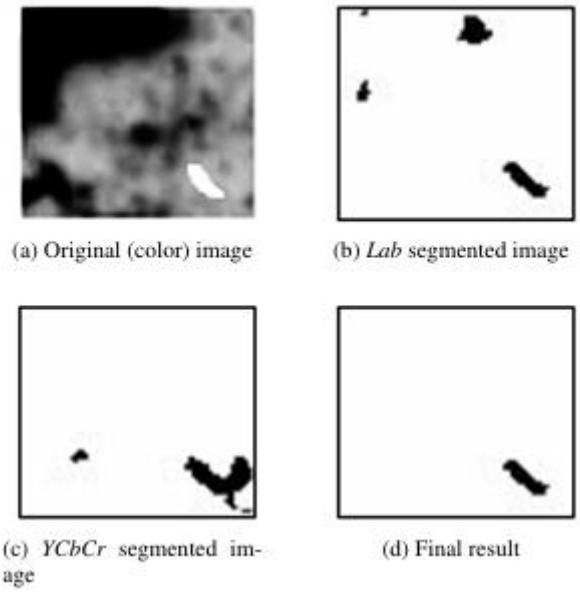


Fig 3.7 An example of the proposed segmentation algorithm.

In *Figure 7(a)* the white-colored area corresponds to the bacillus to be segmented. *Figures 7(b) and 2(c)* show the results of the segmentation algorithm applied to the a and Cr planes respectively. *Figure 2(d)* shows the results of the segmentation process as a result of a logical AND between the images *in Figures 7(b) and 7(c).*

3.2.2.2 ***Segmentation steps***

A. The given RGB microscopic image is transformed into YCbCr and CIE-Lab spaces.

B. Extracted the Cr-component from the YCbCr image and a-component from the Lab image.

C. Computed threshold levels for the Cr-component and a-component based on their first derivatives.

D. Computed segmented images for Cr-component and a-component based on the thresholds obtained in Step 3.

E. Computed a logical AND between the segmented Cr-component and segmented a-component to get the final segmented image.

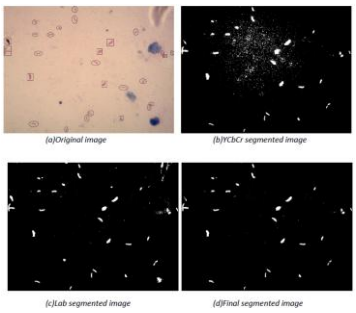


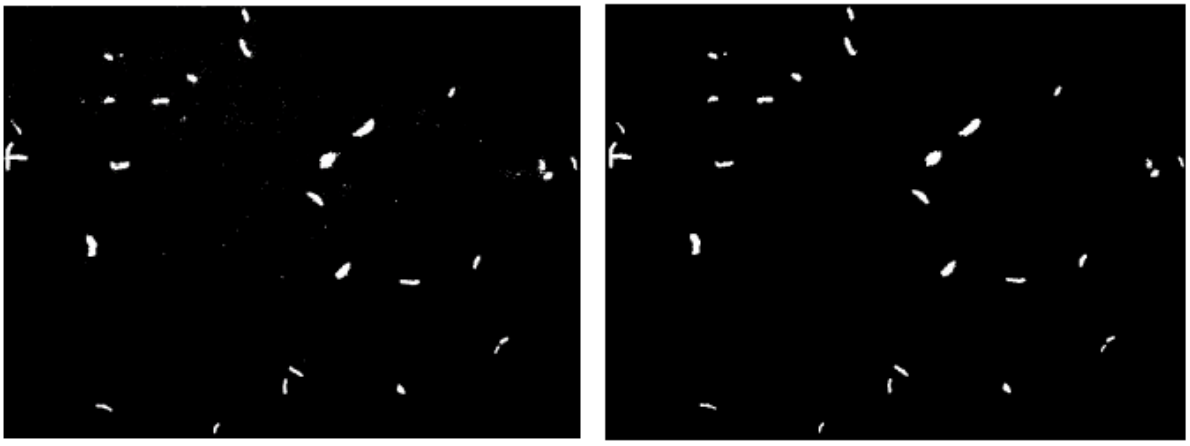
Fig 3.8 Segmentation of Sputum smear Image

3.3 **Image Post-processing**

Along with the true bacilli, the original microscopic image consists of a lot of non-bacillus objects (also called artifacts) having similar color characteristics as that of true bacilli. Due to this, these artifacts are even observed on the segmented image. Also, the count of these artifacts is very high when compared to the count of true bacillus. Removing these artifacts is required to improve the speed of image processing performed in further stages.

The typical size of these artifacts is very small when compared to that of a true bacillus and this fact can be used to remove all these smaller size artifacts.

The segmented image is resized to 600x800. This helps to choose a proper size threshold to reject the small-sized artifacts. After observing many images, it is found that objects with size less than 20 are artifacts. Using morphological operations, objects less than size 20 are removed.

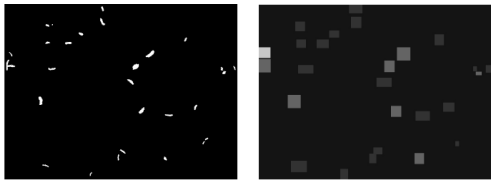


(a)Image after segmentation *(b)Image after post processing*

Fig 3.9 Image post processing of Sputum Smear Image

3.3.1 **Extracting the ground truth info and labelling the objects**

A gray scale image (with all pixel intensities initialized to zeros) is created. Using the details of bounding boxes available in the annotation file, bounding rectangles are drawn on this image at the location of potential TB objects with different intensities (with each intensity corresponding to a class label) to get the image with ground truth information as shown below.



(a)Postprocessed image (b)Image with class label information

Fig 3.10. Visualizing ground truth

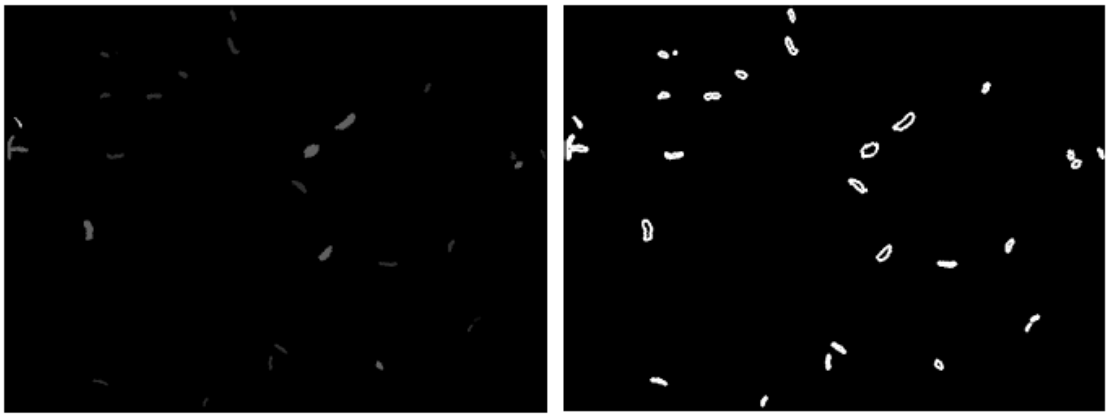
A logical AND is performed between post processed image and the image with class label info to get the labelled processed image (image with labelled objects) as shown below.

(c)Image with labelled objects

Fig 3.11. Labelling with ground truth

3.3.2 Contour finding, feature extraction and formation of labelled data matrix

For each object on the labelled post-processed image, corresponding contour is found and contour properties (i.e., Major axis length, minor axis length, contour area, contour perimeter, perimeter of convex hull, area of convex hull) are obtained. From the contour properties, geometric features [3][4][5] like Area, Roughness, Relative Convex area, Circularity, Compactness, Eccentricity are calculated using the below formulae. (a) Area = Area of contour (b) Roughness = Perimeter of contour/Perimeter of convex hull of contour (c) Relative convex area = Area of convex hull of contour/Area of contour (d) Circularity = 4*pi*area of contour/ (Perimeter of contour) ^2 (e) Compactness = (Perimeter of contour) ^2/ (4*pi*Area of contour) Class label is assigned to each object based on the intensity of corresponding labelled object on Labelled post processed image. A feature vector along with its class label is created for each object in this way. The feature vector along with the class label of all the objects on the labelled postprocessed image are stacked horizontally (row-wise) to get the labelled data matrix for the entire image.



(a)Image with labelled objects (b)Image with contours

Fig 3.12. Contour finding of the bacilli cells

The above process is repeated for all the training images to get their corresponding labelled data matrices and all these labelled data matrices are stacked horizontally to obtain the final labelled data matrix for the entire image dataset.

3.4 Model Architecture:

A pipeline-structured methodology for the identification of tuberculosis bacilli in sputum smear images is proposed using prepossessing, feature extraction, ML classification, and postprocessing. Images of sputum smears were derived from the ZNSM-iDB dataset. Regions of interest were either manually annotated or automatically marked before processing on the pictures. The color-based picture segmentation separates the bacilli from the backdrop by dividing the different regions of the image into various sections according to their color features. After segmentation, extraneous particles or noise that fail to correspond to bacilli are eliminated using postprocessing operations. This focuses attention solely on the regions of interest in the image. After pre-processing, feature extraction is carried out to identify and measure specific features of the bacilli. Contours of the bacilli are located, and several attributes of contours such as edges and shape, are calculated. In addition, the geometric features such as circularity and relative convex area, roughness, and area, are computed.

3.4.1 Random Forest:

The bacilli were classified in this paper from geometric and contour features taken out of the sputum smear images using a conventional ensemble learning technique. In the training phase, this model learns the different decision trees and aggregates their results for making more reliable and accurate predictions. Because it can handle varied bacilli shapes, sizes, and contour features, the use of Random to make a forest is advantageous in this scenario. After segmentation, it reduces the chances of

misclassified areas caused by noise and unwanted information by summing up the decisions taken from several trees.

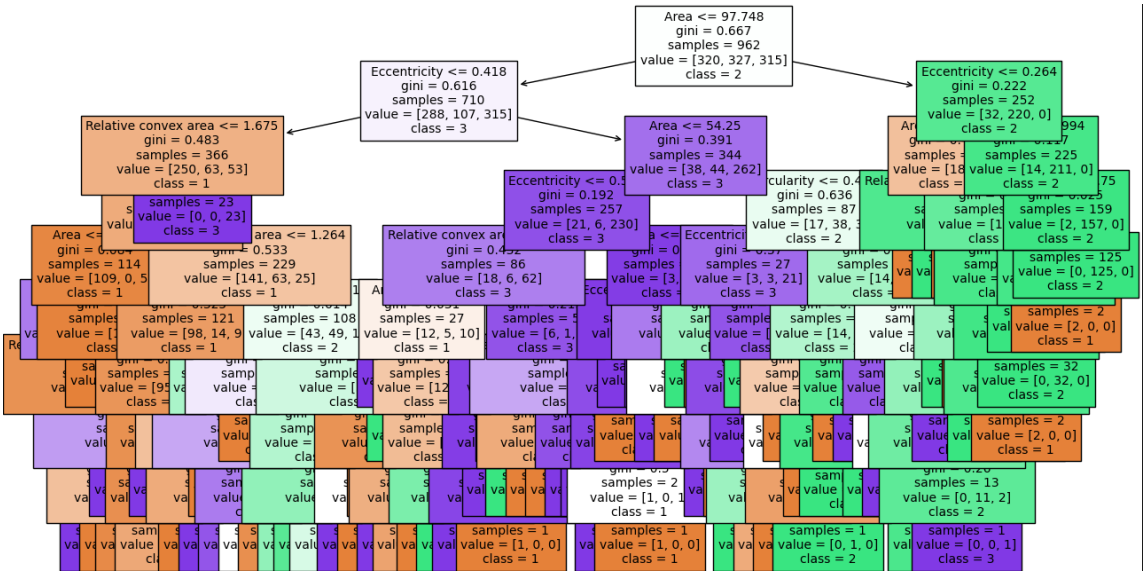


Figure 3.13. Decision tree for the classifier

3.4.2 ResNet50:

The ResNet-50 model is a deep convolutional neural network (CNN) designed for efficient image classification, and its architecture is composed of several key components:

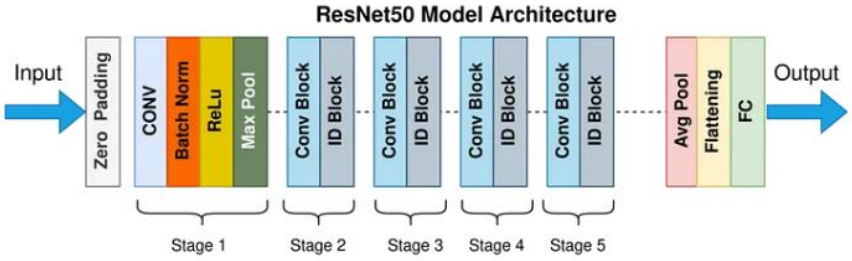


Fig 3.14. ResNet50 Architecture

Convolutional Layers: The network begins with convolutional layers responsible for extracting fundamental features from the input image, such as edges, textures, and shapes. Each convolutional layer is followed by batch normalization and ReLU activation to standardize the output and introduce non-linearity. Max-pooling layers follow to reduce the spatial dimensions of the feature maps, preserving the most important features while reducing computational complexity.

Identity Block: The identity block is a simple yet crucial component of ResNet-50. It allows the input to pass through several convolutional layers, and then the original input is added back to the output (a

process known as residual connection). This mechanism enables the network to learn residual functions that map the input to the desired output, allowing for deeper layers without the risk of vanishing gradients.

Convolutional Block: Similar to the identity block, the convolutional block includes additional transformations. It introduces a 1x1 convolutional layer before the 3x3 convolution to reduce the number of filters, making the architecture computationally efficient. The convolutional block also employs residual connections, allowing the network to learn more complex representations of the input.

Fully Connected Layers: After the feature extraction process through convolutional and pooling layers, the output is fed into fully connected layers for classification. These layers transform the learned features into final class predictions. The final layer uses a softmax activation function to convert the outputs into probabilities across different classes, determining the classification of the input image.

Skip Connections (Residual Connections): A key innovation in ResNet-50 is the use of skip connections, or residual connections, which allow the input to bypass certain layers and add directly to the output. This design helps overcome the vanishing gradient problem, enabling the training of very deep neural networks. Skip connections ensure that the model retains information across layers, improving both accuracy and stability.

3.4.3 DenseNet:

The DenseNet (Densely Connected Convolutional Network) architecture is designed to improve the flow of information and gradients throughout the network, while significantly reducing the number of parameters compared to ResNets. DenseNet achieves this by connecting each layer to every other layer in a feed-forward fashion, promoting feature reuse and enhancing efficiency.

Dense Connectivity: Unlike ResNet, where each layer only connects to the next layer and incorporates residual connections by summing feature maps, DenseNet concatenates the output of all preceding layers as input to the current layer. This dense connectivity ensures that each layer has direct access to gradients from all previous layers, preventing the blocking of information flow and promoting richer feature propagation. This strategy also reduces the risk of vanishing gradients, improving training in deep networks.

residual learning:

$$x_l = H_l(x_{l-1}) + x_{l-1}$$

Dense connectivity equation:

$$x_l = H_l([x_0, x_1, \ldots, x_{l-1}])$$

Narrow Layers: DenseNet uses narrow layers, typically with fewer filters (e.g., 12 filters), compared to other architectures. While ResNets can have large parameter counts due to the independent learning of weights in each layer, DenseNet reduces the parameter count by reusing features from previous layers. As a result, DenseNet models are more compact and computationally efficient.

DenseBlock: DenseNet is composed of several DenseBlocks where each layer within the block is densely connected. All layers in the DenseBlock maintain the same spatial resolution, meaning no downsampling is performed within these blocks. The main components of each DenseLayer within a DenseBlock are:

1x1 Convolutional Layer: This acts as a bottleneck layer, reducing the number of feature maps before the main convolution, optimizing computation.

3x3 Convolutional Layer: After the bottleneck, this layer performs the main convolutional operation, generating the final feature maps.

The outputs of all DenseLayers within a block are concatenated, allowing each subsequent layer to have access to the feature maps generated by all previous layers.

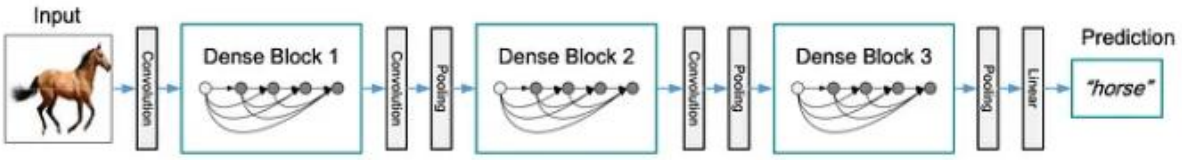


Fig 3.15. DenseNet Architecture

Transition Layers: Between DenseBlocks, Transition Layers are used to downsample the spatial dimensions of the feature maps through average pooling. These layers reduce the size of the feature maps to make the network more computationally manageable as it deepens. Unlike other architectures that lose information during downsampling, DenseNet retains feature reuse by ensuring that even after downsampling, feature maps are densely connected across layers.

Feature Reuse and Model Compactness: The fundamental idea behind DenseNet is that each layer can access the collective knowledge of all previous layers. This dense connectivity results in highly efficient feature reuse across the network, which in turn leads to a more compact model with fewer parameters, making DenseNet computationally efficient without compromising performance.

3.4.4 Efficient Net:

EfficientNet introduces a novel approach to model scaling through a technique known as **compound scaling**. This method effectively balances the model's width, depth, and resolution by uniformly scaling each dimension using a fixed set of scaling coefficients. Instead of randomly adjusting these dimensions, compound scaling ensures a proportional and harmonious increase, which maximizes model performance while maintaining efficiency.

$$\text{Depth } d = \alpha^{\emptyset}, \text{Width } w = \beta^{\emptyset}, \text{Resolution } r = \gamma^{\emptyset}, (1)$$

$$\text{such that } \alpha . \beta^2 . \gamma^2 \approx 2$$

$$\alpha \geq 1, \beta \geq 1, \gamma \geq 1$$

Fig 3.16. Compound scaling equations

EfficientNet Fundamental Building Blocks

EfficientNet is built upon several fundamental components, which enhance its efficiency and effectiveness:

- **MBConv Unit**: The MBConv (Mobile Inverted Bottleneck Convolution) unit is the core building block of EfficientNet, derived from MobileNets. It integrates depthwise and pointwise convolutions within an inverted residual structure. This design significantly improves computational efficiency while facilitating the extraction of more complex features from the input data.
- **Squeeze-and-Excitation Optimization**: This technique enhances the model's ability to focus on informative features by adaptively recalibrating channel-wise feature responses. It enables the network to emphasize essential features while suppressing less important ones, leading to improved representational power.

- **ReLU6 and Swish Activation Functions**: EfficientNet employs ReLU6 and Swish activation functions to enhance non-linearity in the network. These activation functions help the model learn complex patterns more effectively, contributing to overall performance improvements.

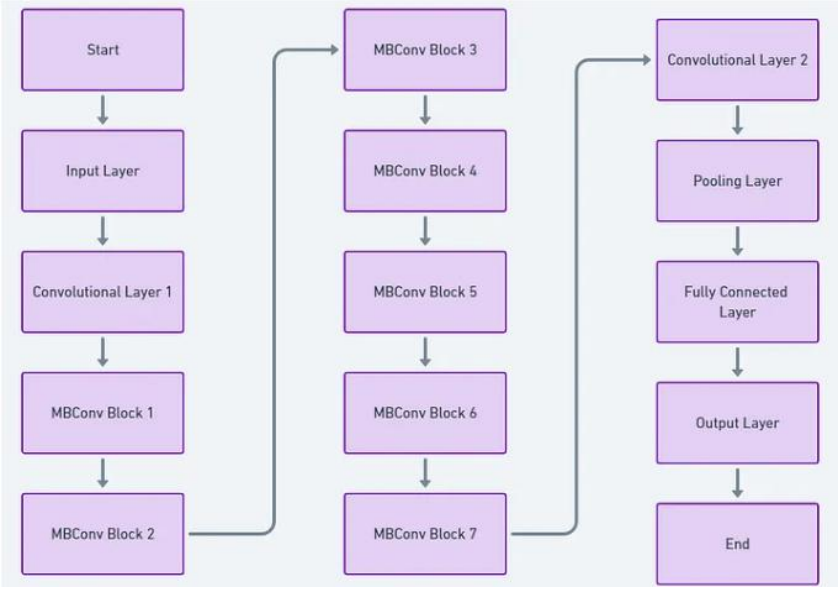


Fig 3.17. EfficientNet Architecture

Through these building blocks and the compound scaling method, EfficientNet achieves state-of-the-art performance on various benchmark tasks while remaining computationally efficient, making it a significant advancement in the field of deep learning.

3.4.5 ViT:

The Vision Transformer (ViT) represents a significant shift in image classification approaches by applying transformer architecture, originally designed for natural language processing, to visual data. ViT processes images by dividing them into patches and transforming these patches into tokens, which allows the model to learn spatial relationships in a fundamentally different way compared to traditional convolutional neural networks (CNNs).

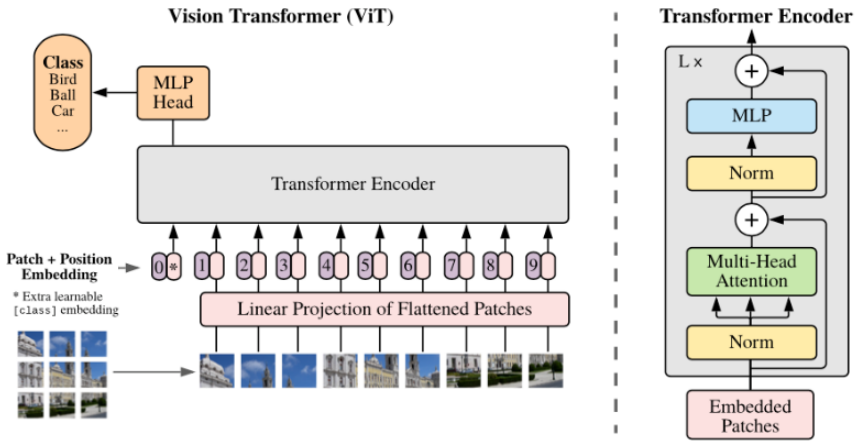


Fig 3.18. ViT Architecture

Key Components of ViT

The operation of ViT can be broken down into several essential steps, each contributing to its effectiveness in image analysis:

1. **Patch Embedding**: The input image is divided into non-overlapping patches. Each patch is then flattened and linearly projected into a vector, creating a set of patch embeddings that represent the image in a lower-dimensional space.
2. **Positional Embedding**: Since transformers do not inherently capture the spatial relationships between patches, positional embeddings are added to the patch embeddings. This step helps the model understand the relative position of each patch within the original image.
3. **Encoder Layers**: ViT utilizes multiple encoder layers that consist of self-attention mechanisms and feedforward neural networks. Each encoder layer processes the input tokens, allowing the model to learn complex representations through the interaction of patches.
4. **Multi-Head Self-Attention**: This mechanism enables the model to attend to different parts of the input simultaneously. By learning relationships among all patches, the self-attention mechanism allows ViT to capture global context, which is crucial for recognizing patterns and relationships between distant patches in the image.
5. **Feedforward Neural Networks**: After the self-attention mechanism, each token passes through a feedforward neural network. This network applies non-linear transformations to further enhance the feature representation learned from the attention layer.
6. **Layer Normalization and Residual Connections**: To stabilize and improve training, layer normalization is applied after each self-attention and feedforward operation. Additionally,

residual connections are incorporated to facilitate the flow of information and gradients, helping to mitigate issues related to vanishing gradients in deep networks.

By leveraging these components, the Vision Transformer effectively captures global context and complex relationships within images, enabling high performance in various vision tasks while moving away from traditional convolutional architectures.

3.4.6 Inception V3:

Inception V3 is a sophisticated deep learning architecture that builds upon its predecessors to address the challenges of increasing the depth of neural networks. While adding more layers can enhance a model's representational power, it also raises concerns such as overfitting, particularly when limited labeled training data is available, and increased computational requirements. Additionally, selecting the appropriate kernel size for convolutions presents a complex challenge.

The Inception architecture provides a solution by allowing the use of filters of varying sizes in parallel without necessitating a deeper network structure. Instead of connecting layers sequentially, the different filter sizes are applied simultaneously within the same module, enabling the model to learn from multiple spatial hierarchies at once.

To mitigate the issue of having an excessively large number of parameters—which can hinder performance and efficiency—Inception V3 incorporates several advanced techniques. One key strategy is the factorization of larger convolutions into smaller ones. For example, a 5×55 \times 55×5 convolution can be broken down into two 3×33 \times 33×3 convolutions, reducing the number of parameters while preserving the model's capability to capture complex features. Furthermore, asymmetric factorizations, such as decomposing a 3×33 \times 33×3 filter into a 1×31 \times 31×3 and 3×13 \times 13×1 filter, are utilized to further optimize parameter usage.

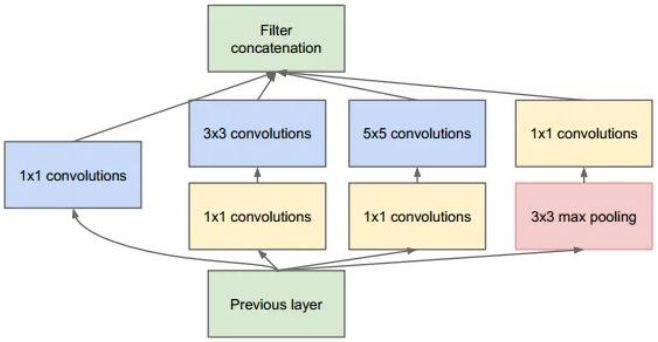


Fig 3.19. Inception Block

An important aspect of Inception V3 is its implementation of depthwise separable convolutions. In this architecture, depthwise separable convolutions perform channel-wise spatial convolution followed by a 1×11 \times 11×1 convolution. However, Inception V3 uniquely applies the 1×11 \times 11×1 convolution first, which can enhance computational efficiency and model performance.

By leveraging these innovative strategies, Inception V3 achieves a powerful balance between model complexity and computational efficiency, making it a popular choice for various image classification tasks in deep learning applications.

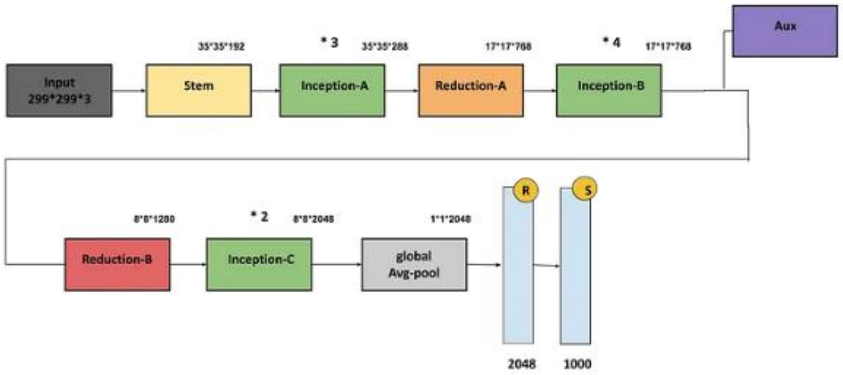


Fig 3.20. Inception V3 Architecture

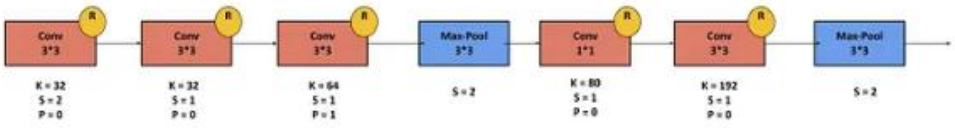


Fig 3.21. Stem Block

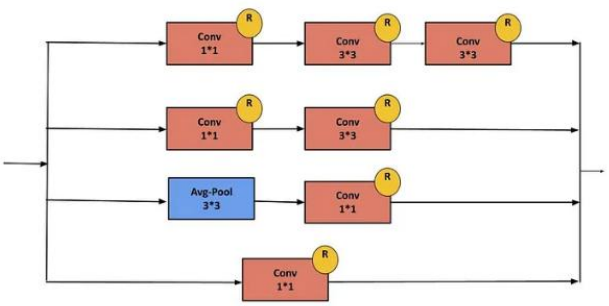


Fig 3.22. Inception A Block

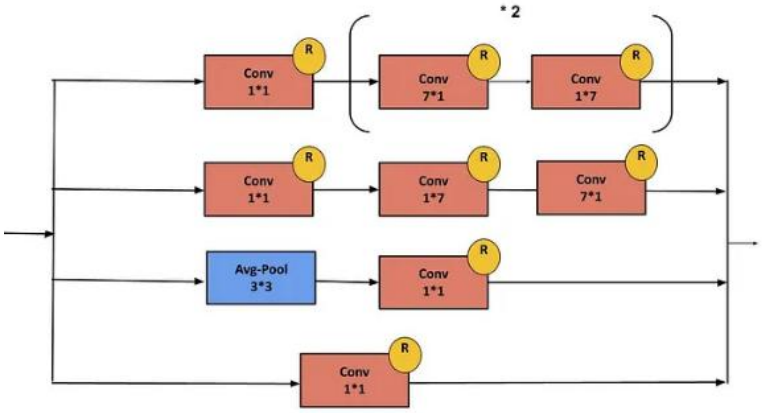


Fig 3.23. Inception B Block

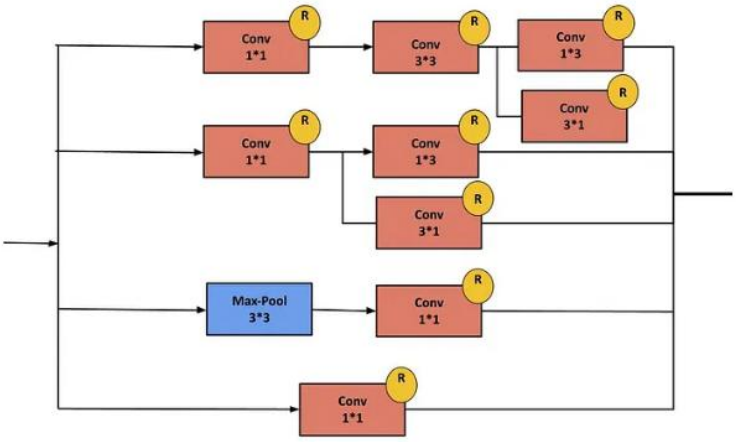


Fig 3.24. Inception C Block

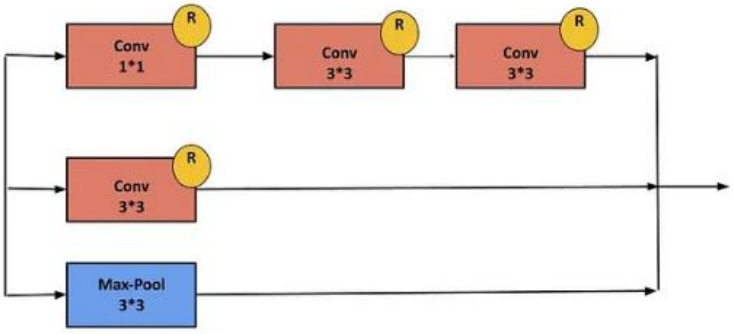


Fig 3.25. Reduction A Block

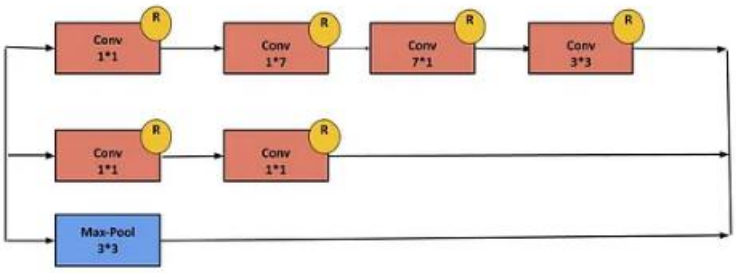


Fig 3.26. Reduction B Block

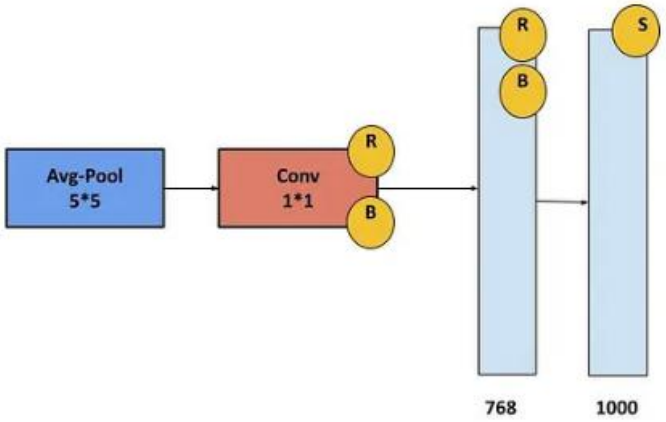


Fig 3.27. Auxiliary classifier Block

3.5 Model Interpretation and Functionality

In this project, several advanced deep learning and machine learning models were employed to effectively classify bacilli cells from sputum smear images. The deep learning models – ResNet50, DenseNet, EfficientNet, Inception V3, and Vision Transformer (ViT) – utilize convolutional layers to extract meaningful patterns from the images. These models identify and classify bacilli by learning features such as shape, texture, and spatial arrangement.

- **ResNet50**: This model leverages residual connections, enabling deeper layers to learn complex features without facing the vanishing gradient problem. It excels in distinguishing between single bacilli and clusters by capturing subtle differences in shapes.
- **DenseNet**: DenseNet creates direct connections between layers to maximize information flow and reuse features, making it effective in recognizing both simple structures like single bacilli and more complex clusters.

- **EfficientNet**: By balancing depth, width, and resolution of the network, EfficientNet achieves a good trade-off between performance and computational efficiency, making it ideal for handling varied-sized bacilli clusters.
- **Inception V3**: This model uses multi-scale processing through factorized convolutions, allowing it to efficiently capture fine details and differentiate between smaller objects like single bacilli and artifacts.
- **Vision Transformer (ViT)**: The ViT model applies self-attention mechanisms, enabling it to understand long-range dependencies in the image, which enhances its capability to detect subtle variations in bacilli clusters and single cells.
- **Random Forest**: This model, unlike the above CNN-based models, focuses on feature extraction from contours. By analyzing contour features such as shape, size, and texture, this model classifies the identified objects into categories like bacilli, clusters, and artifacts. This feature-driven approach provides an alternative to bounding-box methods, focusing instead on interpreting contour characteristics for accurate classification.

3.6 Separation of clustered bacilli objects

The separation of clustered bacilli objects is based on the identification of concave points and the application of ellipse fitting techniques. This method comprises four main components:

- Polygon Approximation

The original contour of clustered bacilli may appear rough and irregular due to overlapping structures. Therefore, Polygon Approximation (PA) is employed to smooth out the irregularities in the contour. This process not only enhances the visual representation of the bacilli but also significantly reduces the number of points along the contour boundary. Consequently, this reduction in points decreases the computational load in subsequent processing phases.

- Concave Point Extraction

Once the contour has been approximated, the next step involves extracting concave points, also known as convexity defects, from the polygon-approximated contour. These concave points play a critical role in segmenting the contour into distinct parts.

- Contour Segmentation

The concave points identified from the approximated contour are utilized to segment the contour into multiple segments. If CCC represents the contour and mmm is the total number of concave points, the segmentation can be mathematically expressed as follows:

$C = \{L_1, L_2, ..., L_m\}$

Here, each segment LiL_iLi is defined by two adjacent concave points as endpoints. These segments are then

prepared for the subsequent ellipse fitting process.

- Ellipse Fitting and Refinement

In the final step, each contour segment undergoes ellipse fitting. During this process, the eccentricity and area of each contour segment are calculated. Only the contour segments that exhibit an area exceeding a predefined threshold and an eccentricity within specified minimum and maximum limits are deemed valid representations of single bacilli. Segments that do not meet these criteria are ignored, ensuring that the analysis focuses on accurately identifying and separating the bacilli within the clustered structures.

- **Flow Chart**

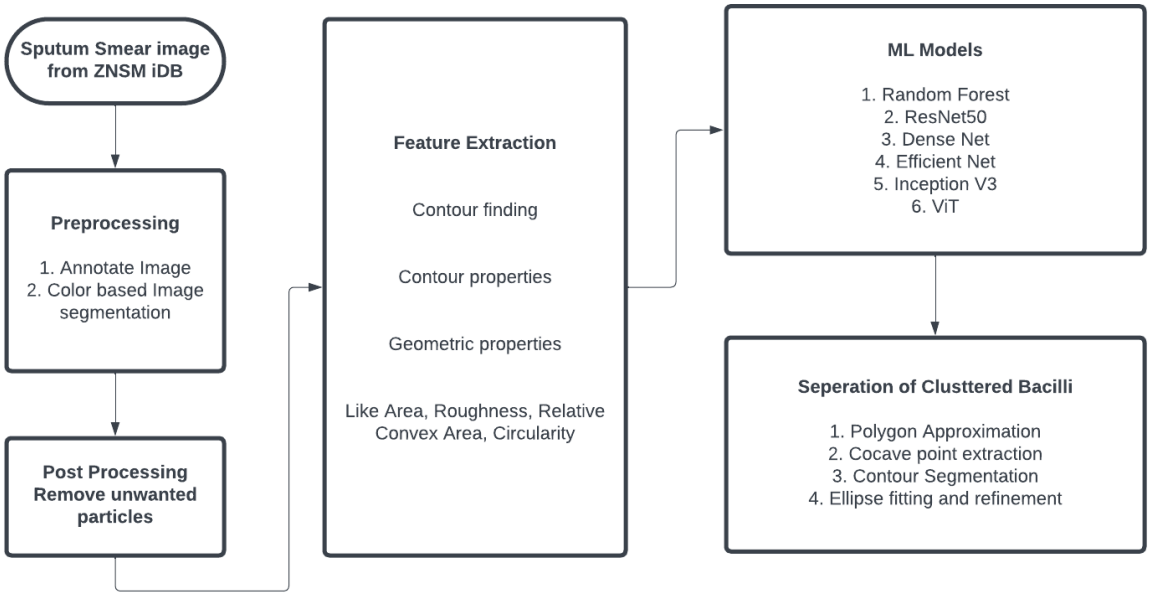


Fig.3.28. Methodology of the project

CHAPTER 4

RESULTS AND DISCUSSION

In an attempt to evaluate the proposed system for the detection of tuberculosis bacilli, several machine learning models have been employed: Random Forest, ResNet50, DenseNet, Inception V3, EfficientNet, and Vision Transformer (ViT). Each model's performance was evaluated through many key performance indicators, such as accuracy, F1-score, precision, and recall. The results of the analysis indicate that each model used to identify and classify the bacilli has its strengths. The top models were the performances of ViT and EfficientNet, exhibiting the best scores in F1 with the highest in terms of accuracy in identifying bacilli from noise and artifacts of sputum smear images. It produced the maximum classification accuracy for this reason, for it processes images in patches and can capture global context and long-range dependencies. It is especially good, therefore, with managing obstructed and clustered bacilli. In particular, in the case of complex images wherein bacilli overlap or appear in clumps, this far outperformed standard convolutional models due to this self-attention mechanism. On the other hand, also performed reasonably well, also while maintaining efficiency in processing high resolution images.

4.1 EXPERIMENTAL SETUP

The experiments aimed to evaluate the performance of several machine learning models in detecting *Mycobacterium tuberculosis* bacilli from sputum smear images. The dataset consisted of ZN-stained microscopy images representing various conditions, ensuring a comprehensive evaluation of model capabilities. Each model—Random Forest, ResNet50, DenseNet, Inception V3, EfficientNet, and Vision Transformer (ViT)—was trained and validated using standard metrics including accuracy, precision, recall, and F1-score.

4.2 RESULTS ON TB BACILLI DETECTION

The evaluation results are summarized in Table I, detailing the performance metrics for each model.

Task	Accuracy	F1 Score
Random Forest	88.4%	84.1%
ResNet50	82.76%	79.4%

DenseNet	91.39%	88.3%
EfficientNet	62.07%	45.3%
Inception V3	79.31%	71.7%
Vision Transformer (ViT)	96.55%	92.3%

Table 4.1. Accuracy and F1 Score of ML models

- **Analysis of Model Effectiveness:**

The performance of these models can be attributed to their architectural strengths. ViT's self-attention mechanism allows it to effectively capture long-range dependencies, which is crucial for accurately identifying individual bacilli amidst clustering. DenseNet, known forint feature reuse, excels in retaining critical information necessary for bacilli detection. In contrast, models like ResNet50 and Inception V3 demonstrated moderate performance, revealing some limitations in managing high background noise and complex image formations.

- **Implications of Findings for TB Diagnostics**

The results indicate that AI-driven models can substantially enhance TB diagnostic workflows by offering high accuracy and low dependency on human intervention. The integration of these technologies into clinical practices can expedite TB diagnoses, facilitating prompt treatment and contributing to improved public health outcomes. Moreover, the successful application of these models supports global health initiatives focused on eradicating TB, particularly in underserved populations.

CHAPTER 5

CONCLUSION

This study has demonstrated the transformative potential of artificial intelligence (AI) in the automated detection of Mycobacterium tuberculosis bacilli in sputum smear images, showcasing how advanced machine learning models can significantly enhance TB diagnostics. Traditional methods, primarily reliant on manual microscopy and expert analysis, have long posed challenges in terms of accuracy, efficiency, and accessibility, particularly in low-resource settings. By leveraging deep learning techniques, this research illustrates a paradigm shift in TB detection that addresses these longstanding issues.

The experimental results highlight the superior performance of the Vision Transformer (ViT) model, which achieved an impressive accuracy of 96.55% and an F1-score of 92.3%. These metrics underscore the model's ability to accurately identify bacilli even in complex imaging conditions, where overlapping and occluded structures pose significant challenges to traditional diagnostic methods. This level of accuracy is crucial for timely diagnosis and treatment initiation, which are vital components in controlling the spread of tuberculosis and improving patient outcomes. The capacity of AI-driven models to reduce the reliance on skilled technicians also represents a significant advancement, particularly in regions where access to trained healthcare professionals is limited.

In addition to demonstrating high diagnostic accuracy, the implementation of AI systems in TB detection aligns with global health initiatives aimed at eradicating the disease. The World Health Organization’s End TB Strategy emphasizes the importance of early diagnosis and treatment to prevent transmission and improve recovery rates. By integrating automated diagnostic tools into clinical practice, healthcare systems can ensure that individuals receive timely and accurate testing, facilitating quicker responses to TB outbreaks and ultimately contributing to the broader goal of reducing TB incidence and mortality worldwide. However, while this study has made significant strides in illustrating the efficacy of AI in TB diagnostics, it also highlights several challenges that must be addressed to fully realize the potential of these technologies. One key limitation is the dependence on high-quality, diverse datasets for training machine learning models. The performance of AI algorithms is inherently tied to the quality and representativeness of the data used in their training. Future research should prioritize the development of comprehensive and publicly accessible datasets that reflect a variety of conditions and populations. This will not only enhance the generalizability of models but also mitigate biases that can arise from limited training data.

Moreover, the computational demands of some advanced models, such as ViT, may pose barriers to their implementation in low-resource healthcare settings. While these models demonstrate superior performance, their complexity and resource requirements can limit accessibility. Future efforts should focus on optimizing these models to achieve high accuracy while minimizing computational overhead. Techniques such as model pruning, quantization, and the use of lighter architectures can help make AI-driven diagnostics more feasible for deployment in diverse environments. The findings of this study also open avenues for further research. The potential for integrating AI diagnostic systems with telemedicine and mobile health applications could enhance accessibility and allow for remote consultations and diagnostics in rural and underserved communities. Such integrations could play a pivotal role in broadening access to healthcare services and improving disease management.

Additionally, the application of AI technologies extends beyond TB detection to encompass a wider array of infectious diseases. Future research should explore the adaptation of similar methodologies for diagnosing other bacterial and viral infections, contributing to a more comprehensive approach to global health challenges. The flexibility of AI systems allows for the incorporation of new data and insights, which can continuously enhance their performance and applicability across various disease contexts. In conclusion, this study illustrates a significant advancement in TB diagnostics through the application of AI and deep learning. The ability to automate the detection of TB bacilli not only improves diagnostic accuracy but also enhances the efficiency of healthcare delivery, particularly in resource-limited settings. By addressing the challenges related to data quality, computational efficiency, and model adaptability, future research can further refine these technologies, contributing to global efforts to combat tuberculosis and improve public health outcomes. The integration of AI into healthcare signifies a promising future, where technology and innovation can bridge gaps in diagnostics and treatment, ultimately leading to healthier communities worldwide.

CHAPTER 6

REFERENCE

[1] Abdeta, J., Diriba, Chala Diriba, and Worku Jimma. ”Pulmonary Tuberculosis Bacilli Detection from Sputum Smear Microscopy Images Using K-Nearest Neighbor Classifier.” (2022).

[2] Chamidah, Nur, et al. ”Identification the number of Mycobacterium tuberculosis based on sputum image using local linear estimator.” Bulletin of Electrical Engineering and Informatics 9.5 (2020): 2109-2116.

[3] Delgado, Lara García, et al. ”Remote analysis of sputum smears for mycobacterium tuberculosis quantification using digital crowdsourcing.” Plos one 17.5 (2022): e0268494.

[4] El-Melegy, Moumen, Doaa Mohamed, and Tarek ElMelegy. ”Automatic detection of tuberculosis bacilli from microscopic sputum smear images using faster r-cnn, transfer learning and augmentation.” Pattern Recognition and Image Analysis: 9th Iberian Conference, IbPRIA 2019, Madrid, Spain, July 1–4, 2019, Proceedings, Part I 9. Springer International Publishing, 2019.

[5] Kieu, Stefanus Tao Hwa, et al. ”A survey of deep learning for lung disease detection on medical images: state-of-the-art, taxonomy, issues and future directions.” Journal of imaging 6.12 (2020): 131.

[6] Kotei, Evans, and Ramkumar Thirunavukarasu. ”Computational techniques for the automated detection of mycobacterium tuberculosis from digitized sputum smear microscopic images: A systematic review.” Progress in Biophysics and Molecular Biology 171 (2022): 4-16.Manoj, NV Sai, M. Rithani, and R. S. SyamDev. ”Enhancing Gastric Cancer Diagnosis Through Ensemble Learning for Medical Image Analysis.” 2023 Seventh International Conference on Image Information Processing (ICIIP). IEEE, 2023.

[8] Panicker, Rani Oomman, et al. ”Automatic detection of tuberculosis bacilli from microscopic sputum smear images using deep learning methods.” Biocybernetics and Biomedical Engineering 38.3 (2018): 691-699.

[9] Patterson, B., et al. ”Detection of Mycobacterium tuberculosis bacilli in bio-aerosols from untreated TB patients. Gates Open Res. 2017 Nov 7; 1: 11.”

[10] Rachna, H. B., and MS Mallikarjuna Swamy. ”Detection of Tuberculosis bacilli using image processing techniques.” International Journal of Soft Computing and Engineering (IJSCE) 3.4 (2013): 2231-2307.

[11] Rangarajan, Prasanna Kumar, et al. ”Detecting AI-generated images with CNN and Interpretation using Explainable AI.” 2024 IEEE International Conference on Contemporary Computing and Communications (InC4). Vol. 1. IEEE, 2024.

[12] Shah, Mohammad Imran, et al. ”Ziehl–Neelsen sputum smear microscopy image database: a resource to facilitate automated bacilli detection for tuberculosis diagnosis.” Journal of Medical Imaging 4.2 (2017): 027503-027503.

[13] Sirohi, Manraj, et al. ”Development of a Machine learning image segmentation-based algorithm for the determination of the adequacy of Gram-stained sputum smear images.” medical journal armed forces india 78.3 (2022): 339-344.

[14] Sotaquira, Miguel, L. Rueda, and Remberth Narvaez. ”Detection and quantification of bacilli and clusters present in sputum smear samples: a novel algorithm for pulmonary tuberculosis diagnosis.” 2009 international conference on digital image processing. IEEE, 2009.

[15] Swamy, Penta Aravinda, M. Rithani, and R. S. SyamDev. ”A Hybrid Model for Disaster Damage Detection Using Satellite Images.” 2023 Seventh International Conference on Image Information Processing (ICIIP). IEEE, 2023.

[16] Swathi, Gunti, Ali Altalbe, and R. Prasanna Kumar. ”QuCNet: Quantum-Inspired Convolutional Neural Networks for Optimized Thyroid Nodule Classification.” IEEE Access (2024).

[17] Ureta, Jennifer, Oya Aran, and Joanna Pauline Rivera. ”Detecting pneumonia in chest radiographs using convolutional neural networks.” Twelfth International Conference on Machine Vision (ICMV 2019). Vol.11433. SPIE, 2020.

[18] Xiong, Yan, et al. ”Automatic detection of mycobacterium tuberculosis using artificial intelligence.” Journal of thoracic disease 10.3 (2018): 1936.

[19] Yang, Xin, et al. ”Automatic comic generation with stylistic multi-page layouts and emotion-driven text balloon generation.” ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM) 17.2 (2021): 1-19.

[20] Zachariou, Marios, Ognjen Arandjelović, and Derek James Sloan. ”Automated methods for tuberculosis detection/diagnosis: A literature review.” BioMedInformatics 3.3 (2023): 724-751.

CHAPTER 7

APPENDICES

```
iporting all the necessary libraries

import cv2
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
from skimage.morphology import
remove_small_holes,remove_small_objects,binary_closing,closing,convex_hull_image
from skimage.morphology import skeletonize, thin
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler,OneHotEncoder,MultiLabelBinarizer
from sklearn.neighbors import KNeighborsClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier,GradientBoostingClassifier
from xgboost import XGBClassifier
from sklearn.tree import DecisionTreeClassifier,plot_tree
from sklearn.metrics import confusion_matrix,roc_auc_score,f1_score,classification_report
from scipy import stats
from imblearn.over_sampling import SMOTE
import pandas as pd
import seaborn as sns
import os
from collections import Counter

# Function definition of image segmentation to separate the potential bacilli objects from
image background

def image_segmentation(img):

    # Transform the input image to YCbCr color space and calculate the histogram and its
first derivative of cr-component
    img_ycrcb = cv2.cvtColor(img, cv2.COLOR_BGR2YCrCb)
    img_cr = img_ycrcb[:,:,1]
    cr_hist,_ = np.histogram(img_cr.ravel(),256,[0,256])
    cr_hist_diff = np.diff(cr_hist)

    # Transform the input image to CIE-Lab color space and calculate the histogram and its
first derivative of a-component
    img_lab = cv2.cvtColor(img, cv2.COLOR_BGR2LAB)
    img_a = img_lab[:,:,1]
    a_hist,_ = np.histogram(img_a.ravel(),256,[0,256])
    a_hist_diff = np.diff(a_hist)

    int_lvls = np.arange(0,255)
```

```
    # Calculating the segmentation threshold for cr-component based on its difference
histogram

    # -18000 is the threshold of cr-component for (1200x900) image. So scaling the
threshold as per input image dimensions
    cr_hist_th = int(-18000*(orig_img_shape[0]*orig_img_shape[1])/(1200*900))
    if (int_lvls[cr_hist_diff<=cr_hist_th].size!=0):
        cr_th_i = np.max(int_lvls[cr_hist_diff<=cr_hist_th])
    # If there is no pixel intensity below -18000, take threshold as peak of cr histogram
    else:
        cr_th_i = np.argmax(cr_hist)

    # Calculating the segmentation threshold for a-component based on its difference
histogram

    # -1000 is the threshold of a-component for (1200x900) image. So scaling the threshold
as per input image dimensions
    a_hist_th = int(-1000*(orig_img_shape[0]*orig_img_shape[1])/(1200*900))
    if (int_lvls[a_hist_diff<=a_hist_th].size!=0):
        a_th_i = np.max(int_lvls[a_hist_diff<=a_hist_th])
    # If there is no pixel intensity below -1000, take threshold as peak of a-histogram
    else:
        a_th_i = np.argmax(a_hist)


    # Thresholding the cr and a components using the above thresholds and calculating
their segmented image outputs
    a_th = a_th_i
    cr_th = cr_th_i
    _, img_a_th = cv2.threshold(img_a, a_th, 255, cv2.THRESH_BINARY)
    _, img_cr_th = cv2.threshold(img_cr, cr_th, 255, cv2.THRESH_BINARY)

    # Performing logical AND between both the segmented images to get the final segmented
image
    img_seg = cv2.bitwise_and(img_cr_th,img_a_th)

    return img_seg

# Function definition of image postprocessing to remove small size artifacts from
segmented image

def image_postprocess(img_seg):

    # Removing small size artifacts on segmented image
    img_bin1 = (img_seg//255).astype(bool)
    img_rem1 =
remove_small_objects(img_bin1,min_size=20,connectivity=8).astype('uint8')*255
    #img_fill =
(remove_small_holes(img_rem1,area_threshold=5000,connectivity=8)).astype('uint8')*255
    #img_bin2 = (img_fill//255).astype(bool)
```

```
    #img_rem2 =
remove_small_objects(img_bin2,min_size=50,connectivity=8).astype('uint8')*255
    img_pp = img_rem1.copy()

    return img_pp

# Function definition to assign class label to each image object

def label_image_objects(img_file,annot_fold_path):

    # Finding annotation file path from image file path
    annot_file = img_file[:-4]+'_annot.csv'
    annot_file_path = os.path.join(annot_fold_path,annot_file)

    annot_cols =
["Label","x_st_pt","y_st_pt","box_width","box_height","Image_name","Image_width","Image_he
ight"]

    lbl_data = pd.read_csv(annot_file_path,names=annot_cols, skiprows=1)

    # Initializtion of labelled image with zero pixel intensities
    img_lbl = np.ones(orig_img_shape[:2], np.uint8)*20

    # Extracting the class label and drawing bounding boxes with different intensities on
labelled image
    for i in range(lbl_data.shape[0]):
        x_st_pt = lbl_data["x_st_pt"][i]
        y_st_pt = lbl_data["y_st_pt"][i]
        x_end_pt = x_st_pt+lbl_data["box_width"][i]
        y_end_pt = y_st_pt+lbl_data["box_height"][i]
        if (lbl_data["Label"][i]=="single bacillus"):
            fill_val = 50
        elif (lbl_data["Label"][i]=="bacilli cluster"):
            fill_val = 100
        elif (lbl_data["Label"][i]=="unclassified red structures"):
            fill_val = 150
        elif (lbl_data["Label"][i]=="artifacts"):
            fill_val = 200
        else:
            print("The actual label is: ", lbl_data["Label"][i])
        st_pt = (x_st_pt,y_st_pt)
        end_pt = (x_end_pt,y_end_pt)
        cv2.rectangle(img_lbl, st_pt, end_pt, fill_val, thickness=-1)

    # Resize the labelled image to standard size of 600x800
    img_lbl = cv2.resize(img_lbl,(img_shape[1],img_shape[0]))

    # Perform logical AND between post processed image and labelled image to get
postprocessed labelled image where each object
    # given different pixel intensities based on its class label
```

```
    img_pp_lbl = cv2.bitwise_and(img_pp,img_lbl)

    return img_lbl,img_pp_lbl

# Function definition to find contours of objects present on the post processed labelled
image

def find_contours(img_pp_lbl):

    # Find contours from the post processed labelled image
    contours,_ = cv2.findContours(img_pp_lbl, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

    # Threshold of contour area to accept or reject the contour
    area_th = 2000

    # Filtering the contours based on the number of contour points, contour area and its
ellipse features
    contours_filt = []
    for cnt in contours:
        if (len(cnt)<5):
            continue
        (x,y),(ma,MA),angle = cv2.fitEllipse(cnt)
        area = cv2.contourArea(cnt)
        area_ell = np.pi*ma*MA/4
        if (x>img_shape[1] or x<0 or y>img_shape[0] or y<0 or area<5 or area>area_th):
            continue
        contours_filt.append(cnt)

    return contours_filt
# Function definition to calculate geometric features of a contour

def extract_features(cnt):

    # Calculate eccentricity of contour by fitting an ellipse
    (x,y),(ma,MA),angle = cv2.fitEllipse(cnt)
    cnt_centre = (int(x),int(y))
    eccentricity = ma/MA

    # Contour perimeter
    perimeter = cv2.arcLength(cnt,True)

    # Contour area (in pixels)
    area = cv2.contourArea(cnt)

    # Convex hull of contour
    hull = cv2.convexHull(cnt)

    # Perimeter of convex hull
    perimeter_hull = cv2.arcLength(hull,True)

    # Roughness of contour
```

```
    roughness = perimeter/perimeter_hull

    # Area of convex hull
    area_hull = cv2.contourArea(hull)

    # Relative convex area
    rel_conv_area = area_hull/area

    # Circularity
    circularity = (4*np.pi*area)/perimeter**2
    #circularity = (4*np.pi*area)/perimeter_hull**2

    # Compactness
    compactness = perimeter**2/(4*np.pi*area)

    # Contour Area (in micro meter)
    area_um = area*2.2

    # Calculating Aspect ratio by fitting bounding box across the contour
    x,y,w,h = cv2.boundingRect(cnt)
    asp_ratio = w/h

    # Gathering all the features
    features =
(cnt_centre,area,roughness,rel_conv_area,eccentricity,circularity,compactness,eccentricity
)

    return features
# Function definition to get class label of the contour based on pixel intensity

def get_class_lbl(img_lbl,cnt_centre):
    if (img_lbl[cnt_centre[1],cnt_centre[0]]==50):
        cls_lbl = "1"
    elif (img_lbl[cnt_centre[1],cnt_centre[0]]==100):
        cls_lbl = "2"
    elif (img_lbl[cnt_centre[1],cnt_centre[0]]==20):
        cls_lbl = "3"
    else:
        cls_lbl = None

    return cls_lbl
```