UNIVERSITY of RWANDA

COLLEGE OF SCIENCE AND TECHNOLOGY

Regional Centre of Excellence in Biomedical Engineering and e-Health (CEBE)

Design And Develop Leukemia Detection Using Deep Learning

By: NAKURE Salathiel

Reference Number: 221031592

A Dissertation Submitted to the Regional Centre of Excellence in Biomedical Engineering and e-Health (CEBE), University of Rwanda as partial fulfilment of the requirements for the master's degree in biomedical engineering.

Supervised by:

Dr. Kizito NKURIKIYEYEZU

&

Dr. Alfred UWITONZE

DECLARATION

I, **Salathiel NAKURE**, declare that this dissertation entitled "**Design And Develop Leukemia Detection Using Deep Learning**" is my original work based on research and prototype and has not been submitted for any other degree or professional qualification.

Student Name:

NAKURE Salathiel

Student Reference Number: 221031592

Student Signature: _____

Date: July 28, 2024

CERTIFICATE

This is to certify that the project entitled "**The Design and Prototype of Automatic Detection of Leukemia Using Deep Learning**" is a record of original work done by NAKURE Salathiel (Reference number: 221031592), an Msc. Degree student in Biomedical Engineering.

This work has been submitted under the guidance of Dr. Kizito NKURIKIYEYEZU and Dr. Alfred Uwitonze

Main Supervisor: **Co-Supervisor:**

Dr. Kizito NKURIKIYEYEZU **Dr.** Alfred Uwitonze

Biomedical Engineering Master's Program Coordinator

Dr. Gerard Rushingabigwi

ACKNOWLEDGMENTS

I want to sincerely thank my wife for her understanding and constant support throughout this journey. My perseverance was strengthened by your support, and I appreciate your unwavering presence.

With special gratitude to my supervisors, your advice and experience have been crucial throughout the project. Your guidance improved my comprehension of the subject and helped to influence the direction of my work.

We appreciate all the support and encouragement we have received from everyone who has contributed to this project, no matter how small. This work serves as evidence of the group efforts of a helpful network.

Sincere appreciation.

NAKURE Salathiel

ABSTRACT

Effective leukemia detection in resource-limited environments mostly in developing country like Rwanda necessitates the use of advanced machine-learning algorithms. This study, titled "**Design and develop Leukemia Detection Using Deep Learning**" addresses the critical need for timely and accurate leukemia diagnosis that is crucial for successful treatment.

Acknowledging the challenges posed by traditional diagnostic methods, such as reliance on expert pathologists and time-consuming procedures as it should take more than two weeks by culturing the samples, the study has focused on leveraging advanced machine-learning techniques. Despite the absence of a local dataset in Rwanda, this research has utilized open datasets that have been prepared with images obtained from Taleqani Hospital(bone marrow laboratory) in Tehran, Iran. This dataset has 4961 training images divided into two classes: 2483 images from healthy patients and 2478 images from patients affected by blood cancer. We tested the model with a total of 1240 images, with 620 from each class, all with a resolution of 320x240 pixels.

The study's findings underscore the effectiveness of the deep learning system compared to conventional diagnostic techniques. Performance evaluation metrics demonstrate its ability to enhance diagnostic accuracy and efficiency, thus presenting a significant advancement in leukemia detection.

Moreover, the study suggests the integration of this system across district hospitals in Rwanda to address the limited access to leukemia diagnosis, especially considering the sole hospital capable of performing such tests. By doing so, the study aims to increase diagnostic capacities and alleviate strain on healthcare resources.

The completed study offers valuable insights and practical solutions to improve leukemia diagnosis not only in Rwanda but also in similar resource-limited environments globally.

Key Words: Leukemia, Deep Learning, Machine Learning, Diagnosis, Blood Cancer, Rwanda

LIST OF ACRONYMS

ALL: Acute lymphoid leukemia

AML: Acute myeloid leukemia

CLL: Chronic lymphoid Leukemia

CML: Chronic myeloid leukemia

CNN: Convolutional neuron network

CP: Classification Phase

DL: Deep learning

Dr: Doctor

ELISA: Enzyme-linked immunosorbent assay

FEP: Feature Extraction Phase

IEEE: Institute of Electrical and Electronics Engineers

IPP: Image Preprocessing Phase.

LeNet-LLD:LeNet Leukemia Detection

MDS: Myelodysplastic Syndromes

MOH: Ministry of Health

OCNN: Optimized Convolution neuron network

PCR: Polymerase chain reaction

PLDT: Proposed Leukemia Detection Technique

Prof: Professor

RBC: Rwanda Biomedical Center

RGB: RED , GREEN,BLUE

US: United state

WBC: white Blood Cell

WHO**:** World Health Organization

FIGURES

TABLES

TABLE OF CONTENTS

CHAPTER 1. GENERAL INTRODUCTION

1.1 Introduction

One form of blood cancer that is particularly problematic for global health is leukemia, especially in places like Rwanda where resources are scarce, on world life expectance it is ranked as 36 causing deaths in our country, and 317 die every year this makes 0.55% in 2023 ; Rwanda recently stated that only about 3000 new cancer cases are registered each year, while the World Health Organization (WHO) estimates that these cases are approximately 10,000 each year[1]. Leukemia manifests as bleeding, bruises, bone pain, exhaustion, fever, and a higher risk of infection. It is defined by the aberrant development of immature blood cells in the bone marrow. These symptoms are brought on by an insufficient generation of healthy red blood cells. A bone marrow biopsy or blood tests are usually required for the diagnosis. Although the precise origins of leukemia are still unknown, scientists think a combination of hereditary and environmental factors have a role[2].

Effective treatment and positive patient outcomes depend on a timely and correct diagnosis. However, manual microscopy and expert interpretation are typically necessary for conventional leukemia diagnosis approaches, which are labor-intensive, time-consuming, and need specialist knowledge [3]. This paper "**The Design and develop Detection of Leukemia Using Deep Learning**," offers an emphasis on the Rwandan setting, in an attempt to overcome these issues. This case study aims to build an automated approach for leukemia identification from blood samples through the application of deep learning, notably machine learning improvements.

With the ability to analyze medical images and data quickly and accurately, deep learning algorithms have the potential to completely transform the field of medical diagnostics[4]. This work aims to overcome the shortcomings of conventional diagnostic techniques and open the door for more effective and accessible leukemia detection in Rwanda by utilizing deep learning.
This case study's main goals are to tailor deep learning algorithms to Rwanda's particular healthcare environment, compile a large dataset of blood samples from patients.

Our goals for conducting this research project are to improve patient care and outcomes, advance automated leukemia diagnosis, and strengthen diagnostic skills in areas with limited resources. The results of this study show the revolutionary potential of deep learning in medical diagnostics and are anticipated to have an impact not just on leukemia diagnosis in Rwanda but also on healthcare systems globally.

1.2 Problem statement

Cancer is a generic term for a large group of diseases that can affect any part of the body. Other terms used are malignant tumors and neoplasms. One defining feature of cancer is the rapid creation of abnormal cells that grow beyond their usual boundaries[5], and which can then invade adjoining parts of the body and spread to other organs; the latter process is referred to as metastasis. Widespread metastases are the primary cause of death from cancer. Cancer is a leading cause of death worldwide, accounting for nearly 9.6 million deaths in 2018, or nearly one in six deaths [6] Leukemia as one type, a blood cell malignancy, is a major public health issue for Rwandans and across the globe that calls for quick, accurate, and precise detection to provide effective diagnosis and treatment. However, the existing traditional diagnostic methods, such as manual microscopic, Enzyme-linked immunosorbent assay (ELISA), and Polymerase chain reaction (PCR) may be expensive and time-consuming as it takes longer to get results and depends on the availability of skilled pathologists [7].

It is a challenge in resource-constrained settings, such as rural clinics and community health centers, where specialized expertized and equipment availability may be limited.

1.3 Research Questions

1. **Performance Evaluation**
   - ✓ How do conventional diagnostic techniques like blood testing and bone marrow biopsies fare against convolutional neural networks (CNNs) in terms of leukemia diagnosis performance?
2. **Generalization and Adaptation**
   - ✓ To what extent can CNN models trained on diverse global datasets be adapted to effectively diagnose leukemia in the population of Rwanda, considering potential differences in data distribution and patient demographics?
3. **Data Availability and Quality**
   - ✓ What are the obstacles in Rwanda related to gathering and annotating medical imaging data for CNN model training?
   - ✓ In the context of Rwandan healthcare settings, how might data augmentation techniques and transfer learning approaches be used to lessen the limits of tiny or imbalanced datasets?

1.4 Objectives

1.4.1 General Objective

- The present project aims to design and prototype an automated detection of leukemia using a deep learning algorithm.

1.4.2 Specific Objectives

- Develop an effective deep learning algorithm for leukemia detection enabled by open source as a dataset.
- Assess the efficacy of the algorithm in detecting cancer on new, unseen data.
- To evaluate and compare the performance of the proposed LeNet-LLD model for leukemia detection with existing models, based on key metrics such as accuracy, precision, recall, F1 score, and loss value

1.5 Study Scope

This research on automatic leukemia detection using deep learning in Rwanda encompasses several key areas, including

1. **Data Collection**
2. Obtaining diagnostic imaging data from public open dataset in Iran, this dataset was divided into 2 classes. They are in total 4961 images where 2483 images were from healthy patients and 2478 images were from patients affected with blood cancer. We tested the model with 1240 images 620 from each class. These images has resolution of 320*240 and they were ready to be used .
3. **Model Development**

- Creating and putting into practice convolutional neural network (LeNet-LLD) architectures specifically designed to detect leukemia utilizing deep learning frameworks in our case we have used Tensor Flow. Because it offers several advantages that align with the project's objectives and requirements some of them are Robust Deep Learning Framework, High Performance and Efficiency, Flexibility and Customization, and Integration with Python and Jupyter Notebooks
- Using annotated medical imaging data, LeNet-LLD model was trained, with a focus on resolving issues with data imbalance and small dataset sizes. We have used data

augmentation that provides different transformations on images such as rotation, shear, zooming, shift, and flip to resolve the data imbalance and increase model precision

4. Evaluation Metrics

❖ Defining assessment parameters (accuracy, sensitivity, specificity, precision, recall, and F1 score) to measure CNN models' performance.

- **Definition**: Accuracy measures the overall correctness of the LeNet-LLD model's predictions, calculated as the ratio of correctly classified samples to the total number of samples.
- **Formula**: Accuracy=Number of Correct Predictions/Total Number of Predictions

- **Sensitivity (True Positive Rate) same as Recall**:

  - Definition: Sensitivity measures the proportion of actual positive cases that are correctly identified by the LeNet-LLD model.
  - Formula: Sensitivity R=True Positives / True Positives+False Negatives

- **Precision (Positive Predictive Value)**:

  - Definition: Precision measures the proportion of true positive predictions out of all positive predictions made by the LeNet-LLD model.
  - Formula: Precision P=True Positives / True Positives+False Positives

- **F1 Score**:

  - Definition: The F1 score is the harmonic mean of precision and recall, providing a balance between the two metrics.
  - Formula: F1=2×Precision×Recall / Precision+Recall

These assessment parameters collectively provide a comprehensive evaluation of the performance of LeNet-LLD model in classifying medical images for leukemia diagnosis. They help to measure or to understand the model's ability to correctly identify both positive and negative cases, as well as its balance between precision and recall.

- ❖ Performing validation and cross-validation on separate test datasets to guarantee the models' robustness and generalizability

1.6 Significance of the Study

Using machine learning CNN for autonomous leukemia detection in Rwanda is significant in several important ways.

- ✓ **Early Detection and Diagnosis**: The implementation of automated detection technologies has promise in enabling timely intervention and treatment commencement by facilitating early leukemia identification. Increasing survival rates and bettering patient outcomes depend heavily on early diagnosis.
- ✓ **Resource Optimization:** The study can lessen the workload for medical staff, cut down on diagnostic delays, and improve resource efficiency in Rwandan healthcare facilities by automating the diagnosis procedure. Better patient outcomes and more effective healthcare delivery may result from this when deployed in different hospitals.
- ✓ **Enhanced Accuracy and Precision:** Machine learning CNN models have shown the potential to surpass conventional diagnostic techniques in the analysis of medical imaging data due to their high accuracy and precision. This will reduce diagnostic errors and misinterpretations and lead to more consistent and reliable leukemia detection.
- ✓ **Accessible Healthcare Services:** For our country, the implementation of automated diagnostic technologies has the potential to enhance access to specialist healthcare services and surmount geographical obstacles, especially in places that are distant or underprivileged. In addition to ensuring that all patients have access to a prompt and accurate leukemia diagnosis, this promotes healthcare equity.
- ✓ **Research and Innovation:** The study promotes innovation and research in the fields of machine learning applications in healthcare and medical imaging analysis. It encourages cooperation between academic institutions, businesses, and healthcare providers and advances knowledge and technology in Rwanda's healthcare sector.
- ✓ **Public Health Impact:** The study enhances leukemia identification and care, that helps public health practioners initiatives for disease burden reduction, complications avoidance, and population health enhancement. It is in the same line with national healthcare policies and programs that are meant to fight non-communicable illnesses like cancer.

- ✓ **Capacity Building:** The study improves the knowledge and abilities of medical professionals, researchers, and technicians involved in leukemia diagnosis and treatment in the healthcare sector through training and capacity-building activities. This promotes sustainable development in the healthcare industry and fortifies the nation's healthcare personnel.

In brief**,** this study has significant implications for healthcare delivery, patient outcomes, and healthcare system strengthening in Rwanda, paving the way for improved leukemia diagnosis and management through the application of advanced machine learning techniques.

1.7 Organization

Chapter 1: Introduction

A summary of the research issue is given in this chapter, together with background data, the problem statement, the objectives, the research questions, and the study's significance.

Chapter 2: Literature Review

This chapter presents a thorough analysis of the body of research on leukemia diagnosis, machine learning applications in medicine, and earlier research on CNN models' automatic leukemia identification.

Chapter 3: Research Methodology

The research methodology used in the study is described in this chapter, along with the data collection strategies, data pretreatment methods, model construction, and evaluation measures.

Chapter 4: Results , Interpretation and Discussion

This chapter presents and analyzes the study's findings, including how well the constructed CNN model detected leukemia, how it performed in comparison to other diagnostic techniques, and a discussion of the study's main conclusions.

Chapter 5: Conclusion and Recommendations

The study's main conclusions are outlined in this last chapter, which also offers suggestions for additional research and therapeutic applications. Finally, the research study is concluded with closing remarks.

CHAPTER 2. RECENTLY RELATED LITERATURE

2.1 Literature Review

Leukemia disease is a general name for white blood cancer it can have different names derived from which type of affected white blood cell. The disease affects bone marrow and blood cells by uncontrolled production of white blood cells (WBC) also known as Leukocytes, the abnormal is called Leukemia we know that exist other types of blood cell such as **Lymphoma, Multiple Myeloma, Myelodysplastic Syndromes (MDS)** and many more. The white blood cells are responsible to fight against infections and the immune of the whole body [8]

Leukemia is associated with a huge production of white blood cells in bone marrow and tends to replace other types of blood cells. There are two main types of leukemia based on manifestation development, Acute and chronic Leukemia. **Infected white blood cell grows rapidly and do not perform normally in acute leukemia**, while in chronic leukemia they migrate normally and grow less rapidly.it is difficult to distinguish between normal white blood cells and healthy White blood cells. Leukemia cells proliferate primarily in bone marrow and lymphoid tissues and then spread-out side in other tissues. The concerned tissues are lymphoid and myeloid they proliferate from bone marrow [9]. As said before there two types of leukemia one issued from lymphoid and another from myeloid depending on the size and shape of white blood cells, we have again two subspecies Acute and chronic (see above); they become Acute lymphoid leukemia (ALL), Chronic lymphoid Leukemia (CLL), Acute myeloid leukemia, and Chronic myeloid leukemia (CML).

As the main objective of this research project is to develop an automated detection of Leukemia that has different types of white blood cells cancer is important to define each one of them before we go deeper.

2.1 Types and definitions of leukemia

2.1.1. Acute lymphoid leukemia

Acute lymphoid leukemia (ALL) is one type of white blood cell cancer that originates from lymphoid precursors in the bone marrow. Blast cells in peripheral blood and the bone marrow proliferate and accumulate. ALL has mostly immature bone marrow white blood cells; it is a cancer caused by overproduction and endless multiplication of lymphocytes [6]. It usually happens at the age of 2-5 years of kids and it presents flu-like symptoms, fatigue in the bone joints, and weakness and pain [6]. As with other disease, the diagnostic took longer as they used a manual method for analyzing samples; it has been observed that the result also presents much doubt on its accuracy and precision. It is considered that the morphological similarity of those cells makes the diagnosis

of the disease problematic, which can lead to death if undetected and treatment not initiated on time [6]

2.1.2. Acute myeloid leukemia

Acute Myeloid leukemia (AML) [10]is characterized by the replacement of normal myeloid cells by undifferentiable blasts as a result of acquired genetic changes [6]. The abnormality of myeloid develops with the ability of the cells to proliferate and block the differentiation process or deviate from the normal mechanism. Bone marrow blasts begin to create or generate immature white blood cells. They can also produce platelets and red blood cells all of which are not stable or mature ones. The most common symptoms of early-stage AML can be like flu or other common illnesses. It develops in 25% of patients and is frequently observed in men and middle age [6]

2.1.3 Chronic lymphocytic leukemia

Chronic lymphocytic leukemia (CLL) is a type of cancer of blood, bone, and marrow. CLL develops from a type of white blood cell called B cells. It propagates slowly, usually affecting older adults. The same symptoms are fatigue, swollen lymph nodes easy bruising, and pain in the upper left portion of the abdomen, which may be caused by an enlarged spleen. Treatment may include chemotherapy, and stem cell transplantation [6].

2.1.4. Chronic Myeloid leukemia

A slow-progressing and uncommon type of blood-cell cancer that begins in the bone marrow. Chronic myeloid leukemia (CLL) affects older adults and is caused by a chromosome mutation that occurs spontaneously. The cause of that mutation is not yet revealed. Many people don’t show symptoms until later stages and the diagnosis is only made through routine blood work. When symptoms do appear, the patient may present bleeding easily, feeling run down or tired, weight loss, pale skin, and night sweats. Treatment may include biological therapy, stem cell transplantation, chemotherapy, targeted drugs [6].

2.2 **Leukemia Identification Using Deep Learning Algorithms**

Many decades ago, many strategies for automated leukemia detection and identification on microscopic images were established in the literature. We can list a few of them here, such as the traditional machine learning classifiers and deep learning algorithms. In the existing literature, machine learning methods are most frequently used for leukemia cancer detection and they involve several steps such as preprocessing, and feature extraction [11], accompanied by classifications. Some of them require segmentation and feature selection procedures to further enhance the performance [12]. This is a conventional machine learning approach on the other side we can investigate the deep learning approaches in the context of, most researchers even many of them adopt and design several architectures for the automated classification of leukemia cancer. These deep learning techniques are further classified into traditional standalone deep learning models or transfer of learning-based approaches [7]. "For instance, Shaheen et al. suggested the AlexNet-based deep learning model to diagnose Acute Myeloid Leukemia (AML) using blood samples in the form of microscopic images. They have compared the performance of their presented approach with the LeNet-5 model in terms of accuracy, quadratic loss, recall, and precision. Their proposed method shows 98.58% accuracy along with 88.9% of the microscopic images being accurately classified with 87.4% accuracy"[7] Other researchers suggested a CNN architecture comprising several convolution and Max-pooling layers for leukemia cancer[13]

To overcome the limitations of manual screening; hematologists have to come up with an automated approach that could detect malignant lymphocytes. This is to answer a hard and complicated task that hematologists face during the diagnosis of leukemia due long time and inaccuracy that may result when the old method is used [8].

Deep and machine learning, two subfields of artificial intelligence, are both beneficial for detecting malignancies. However, when it comes to diagnosing various types of leukemia, deep learning

appears to out perform because it catches complex patterns and features that are detectable or clearly specified by human specialists [3].

Complex data analysis: Examining several types of data, such as clinical records and genomic information, is critical for diagnosing leukemia. Furthermore, image analysis with a convolutional neural network (CNN) and sequential data analysis with a recurrent neural network (RNN) are powerful deep learning approaches that can deal with complicated structured datasets while effectively recognizing space-time links[14]. Machine learning techniques can find markers for accurate leukemia identification by processing large high-dimensional data sets.[15]

Improved performance: Deep learning models' capacity to understand complicated patterns and representations has resulted in a lot of promise for a variety of sectors, including medical image analysis and genetic data. Consequently, identifying leukemia using deep-learning-based methodologies is more trustworthy due to increased accuracy and sensitivity when compared to typical machine-learning methods. The use of vast volumes of data for training(data augmentation plays important role in this job) contributes to the improved performance of deep learning models. [3], [16]

Generalization capability: The capacity of deep learning models to generalize effectively means that they can utilize what they have learnt from previous experience to generate accurate predictions on new or previously unknown datasets, and the ability to generalize well across diverse patient types and datasets is critical in diagnosing leukemia**.**[10 ,p.24], [17], [18]

2.3 Summary

Several methods for automated leukemia diagnosis and identification utilizing both conventional machine learning and deep learning algorithms are highlighted in the research review. Preprocessing, feature extraction, and classification are the phases involved in traditional machine learning techniques[19]. For better results, segmentation and feature selection processes are frequently needed. Conversely, automated feature extraction from unprocessed data is provided by deep learning methods, namely convolutional neural networks (CNNs), which make it possible to identify intricate patterns and features without the need for human feature engineering.

Numerous research studies have exhibited the efficacy of deep learning models, including CNN and AlexNet-based architectures, in accurately and sensitively identifying leukemia. When it comes to jobs that call for automatic feature extraction, intricate data analysis, enhanced

performance, and the ability to generalize across many datasets, deep learning performs better than classical machine learning.

The insights gained from this chapter highlight the potential of deep learning approaches to improve leukemia diagnosis by means of precise and effective automated detection systems. These results encourage more research and development of deep learning-based methods for leukemia detection with the goal of enhancing patient outcomes, decreasing manual labor, and increasing diagnostic accuracy. The knowledge gathered from prior research serves as the basis for this study's continuation into the upcoming chapters, where new approaches and experiments will be used, suggested and assessed.

CHAPTER 3. RESEARCH METHODOLOGY

In the following section, we will show all the steps and methods used to collect data and group them into two classes and subsets used to test and validate the model and the answer for research question will be discussed. The system design and prototype will appear in this section.

3.1 Research Process

The research process has followed the structured framework, it was designed to facilitate and make guidance to the investigation; during our project assessment and development of automated leukemia detection using deep learning. We will utilize convolution neural networks as the best choice as they can handle multiple tasks from raw input data (Digital Images of blood smear in our case).

The framework includes the following main elements:

1. **Problem Definition**: Current issues revolve around the effective detection of leukemia, a type of blood cancer, especially in resource-limited regions like Rwanda. Diagnosis of leukemia has traditionally relied on manual microscopy, which is time-consuming, labor-intensive, and prone to human error. Furthermore, the availability of qualified hematologists and access to advanced diagnostic facilities may be limited in certain regions, further exacerbating the challenge of timely and accurate diagnosis.

   Our main goal is to address these limitations by developing and implementing a deep learning-based leukemia detection algorithm. This algorithm aims to automate the detection process, which has the potential to reduce dependence on manual microscopy and improve diagnostic accuracy and efficiency.

   Key aspects of the problem include:

   1) **Limited access to advanced diagnostic equipment:** Many health facilities in Rwanda, especially those in rural and remote areas, may lack access to advanced diagnostic equipment and trained staff (pathologists). This makes timely and accurate leukemia diagnosis difficult.
   2) **Challenges of manual microscopy:** Traditional leukemia diagnostic methods rely on manual microscopy of blood samples, which can be subjective, time-consuming, and require specialized knowledge. Inconsistent interpretation and diagnosis can lead to inaccurate results and delays in starting treatment**.** They identify and categorize

leukemia by counting various blood cells and morphological features [20]. This technique is time-consuming for the prediction of leukemia. The pathologist's professional skills and experiences may be affecting this procedure, too. In other cases Cytogenetic testing usually takes about **2 to 3 weeks** because the leukemia cells must grow in lab (microbiological culture) dishes for a couple of weeks before their chromosomes are ready to be looked at. Not all chromosome changes can be seen under a microscope. Other lab tests can often help detect these changes

To summarize, the issue at hand concerns the requirement for a novel, economical, and precise approach to leukemia detection that may surmount the obstacles presented by resource constraints and the prevalence of manual diagnostic procedures in Rwanda. The creation of this algorithm based on deep learning offers a chance to overcome these obstacles and enhance the quality of healthcare for leukemia patients in the area as it has been observed in our country in reviewed leterature.

3.2 Research Design Method

The system design flowchart visually and the model architecture represents the entire research process, from data collection to model evaluation and implementation, providing a comprehensive understanding of the study approach.

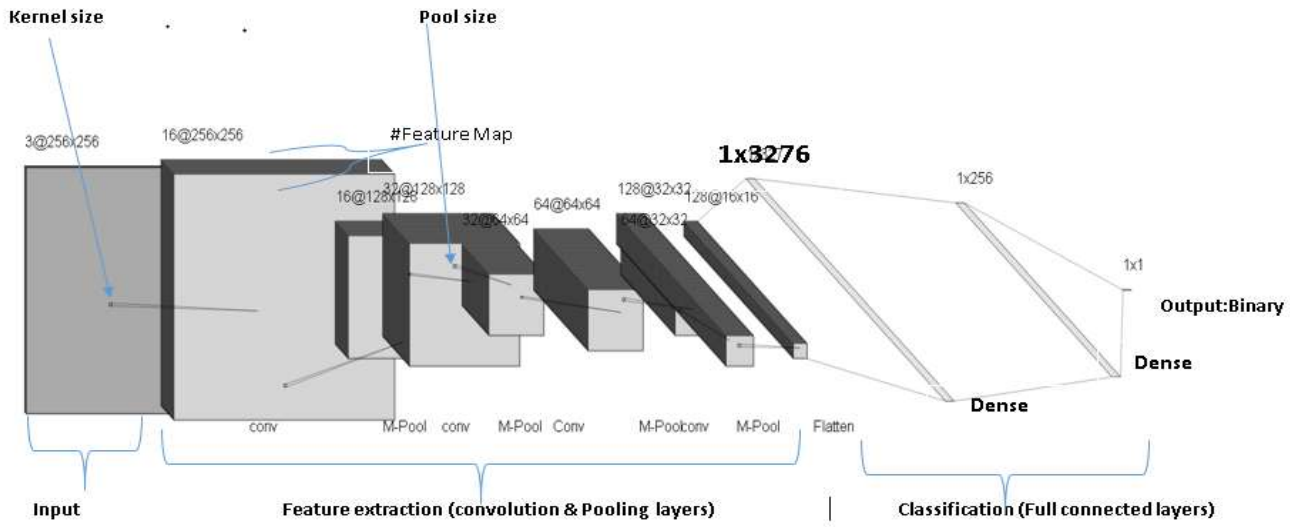


*Figure 1:*Proposed Leukemia detection Algorthm artchitecture based on LeNet (LeNet-LLD)

We have three types of CNN architecture which are the convolutional layers, pooling layers, and fully-connected (FC) layers. When these layers are stacked together, a CNN architecture is formed.In this designed model we have defferent approach we used to increase the metric measurement such as accuracy, precision and other as discussed earler in this paper. We have applied kernel or filters and pooling filter which are small matrix , they are applied on input matrix to filter out some elements and that operation is called convolution . In this section we have difeferent number of filters both on convolution and Max-pooling as we move forward as it can be seen on the figure 2 above .Let have one example on 3rd layer of convolution we have 64@64x64, This layer is a critical component of the Convolutional Neural Network (CNN) architecture, specifically designed to detect more complex patterns and features in the input data. Parameters and configuration of this layers has 64 filters the first number in this **64**@64x64; Each filter is responsible for detecting a specific feature in the input data , the more filters there are, the more features the layer can detect and each of the 64 filters has a height of 3 and a width of 3. This means each filter scans over a 3x3 region of the input image at a time; smaller filter size like 3x3 are common as they are effective in capturing local spatial features while maintaining computational efficiency.

Activation function used in all layers except the last one is ReLu(), it introduces non lineality in the model, allowing the network to learn more complex patterns. The input to this layer comes from the previous MaxPooling layer with input that has 32 feature maps each of size 64x64. In convolution operation each of the 64 filters performs a convolution operation on the 32 input feature maps; the filter slides across the width and height of the input, performing element-wise multiplications and summing them up and this results in 64 output feature maps, each corresponding to a filter ; thereafter the output has 64 feature maps (depth), each of size 64x64. The height and width remain the same as the input due to the "same" padding used and the depth increases from 32 to 64 because we are using 64 filters.

This layer extracts more complex features from the input data, such as edges, textures, and patterns that are more abstract compared to the initial layers. The 64 filters captures different aspects of the features in the input image. As we move deeper into the network, each layer learns higher-level features. The third convolutional layer builds upon the features detected by the previous layers, allowing the model to understand more complex and abstract representations of the input data. In brief third convolutional layer is pivotal in enhancing the model's ability to recognize intricate patterns and structures within the input data. By increasing the number of filters and maintaining

a small filter size, it ensures detailed feature extraction while preserving the spatial dimensions of the input, thanks to the use of padding.
Each layer has its own number of parameters and this was an example taken randomly to explain a bit the figure above.

From convolution and pooling layers that are multidimensional layers we introduce flattening layer for serving as a bridge between multi dimensional layers and one dimension layer called fully connected (Dense) layer as this flatten layer takes the multi-dimensional output from the convolutional or pooling layers and converts it into a one-dimensional array. The input to theflatten layer is depth: 128, Height: 16, Width: 16 and this comes from the last MaxPooling layer; the input is a 3D tensor with 128 feature maps, each of size 16x16. It reshapes the 3D tensor into a 1D tensor (a flat vector); for an input of size (128, 16, 16), the output size will be 128 * 16 * 16 = 32768 units (flattened output is a vector with 32768 elements). The flatten layer simplifies the multi-dimensional data, making it suitable for input into the fully connected layers, which require a one-dimensional vector
Fully connected layers are used to combine features learned by the convolutional layers to predict the final output they act as classifier. Each neuron in the fully connected layer has a weight and a bias; weights are learned during the training process to capture important features.
The fully connected layer produces an output of 256 units, each unit represents a learned feature combination. The output layer produces the final prediction of the network and for binary classification, it outputs a single value representing the probability of one class. Remember we have 256 units from the previous fully connected layer which is input to this last layer with sigmoid function **(σ(x));** this sigmoid function outputs a value between 0 and 1.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad \text{and} \quad x = \sum_{i=1}^{256} w_i z_i + b$$

Formula (a) formula(b)

For each neuron in the output layer, **x** is calculated as formula (b); where, **wi** represents the weight associated with the **i-th** input zi and **b** represents the bias.

In the settings of proposed model the sigmoid activation function in the output layer provides a probability for binary classification, where high values indicate normalcy and low values indicate abnormalcy. The output with sigmoid activation produces a binary classification probability, with values close to 1 indicating a high probability of the image being normal and values close to 0 indicating a high probability of the image being abnormal.

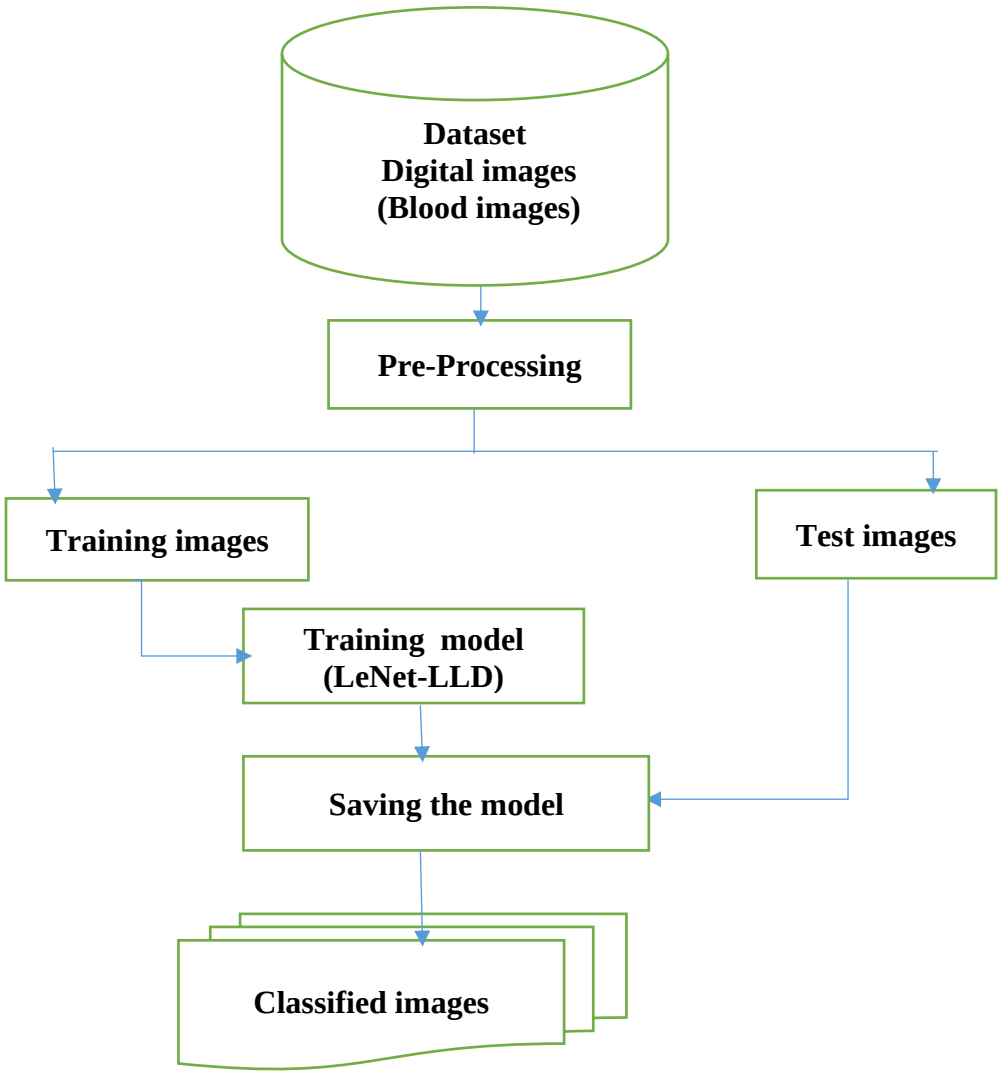


Figure 2:The Process of medical image classification based on the LeNet-LLD

A dataset of microscopic pictures is gathered and then preprocessed to improve the quality this means adjusting the size , normalizing them and ensure the required image extension, as our dataset is limited in data we have used data augmentation as recommended by the previous researchers[21] thereafter a LeNet-LLD architecture is designed for feature extraction and classification. Several measures are used to assess the model's performance this is to monitor training and validation

accuracy on other the loss have been considered in this study . With the testing of the model, we have built a script that enables us to pass new input images one by one to the system to be classified as either negative or positive. Before we go into classification is important to provide some graphs of the model showing the performance of the model in the detection or classification of the images

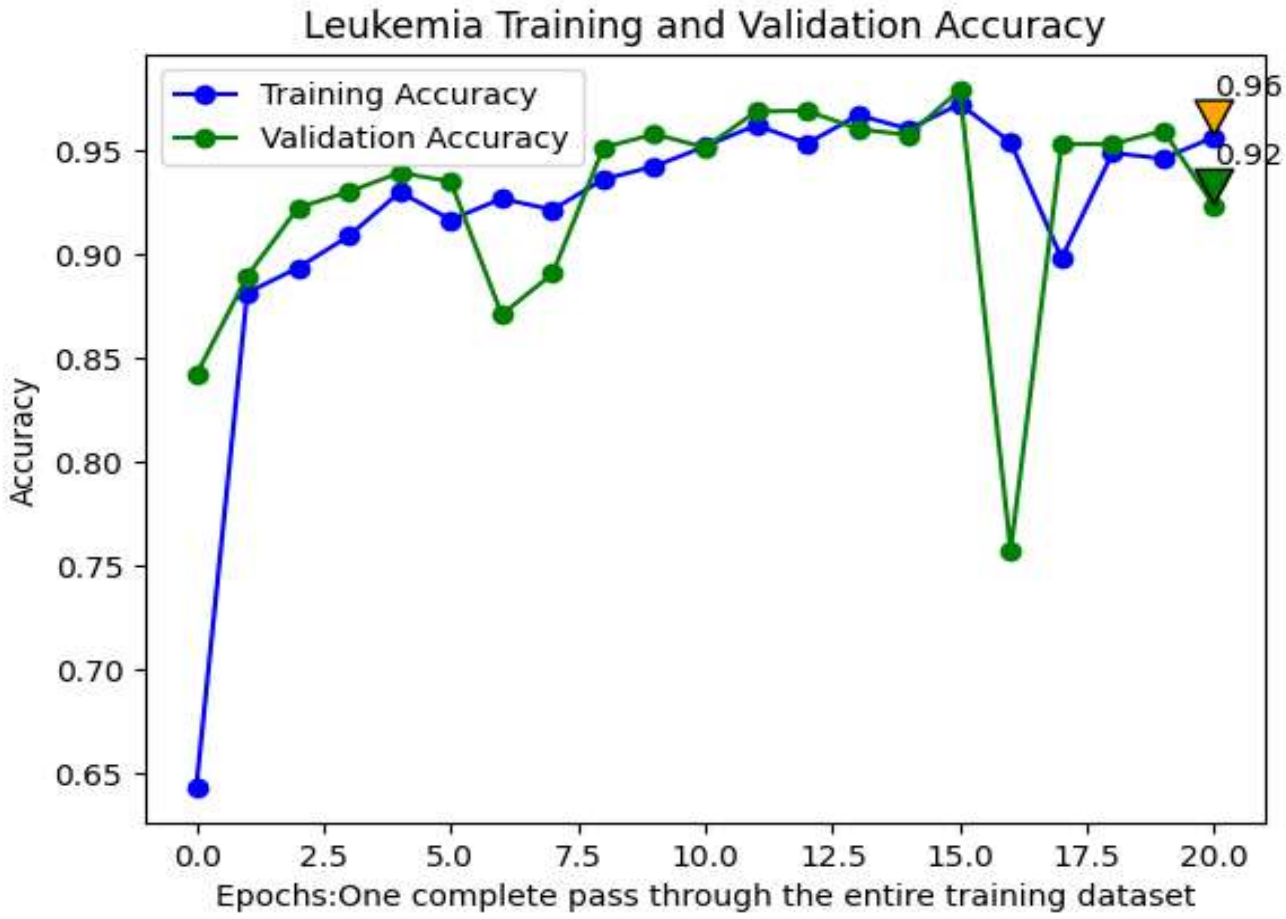


Figure 3: Model accuracy

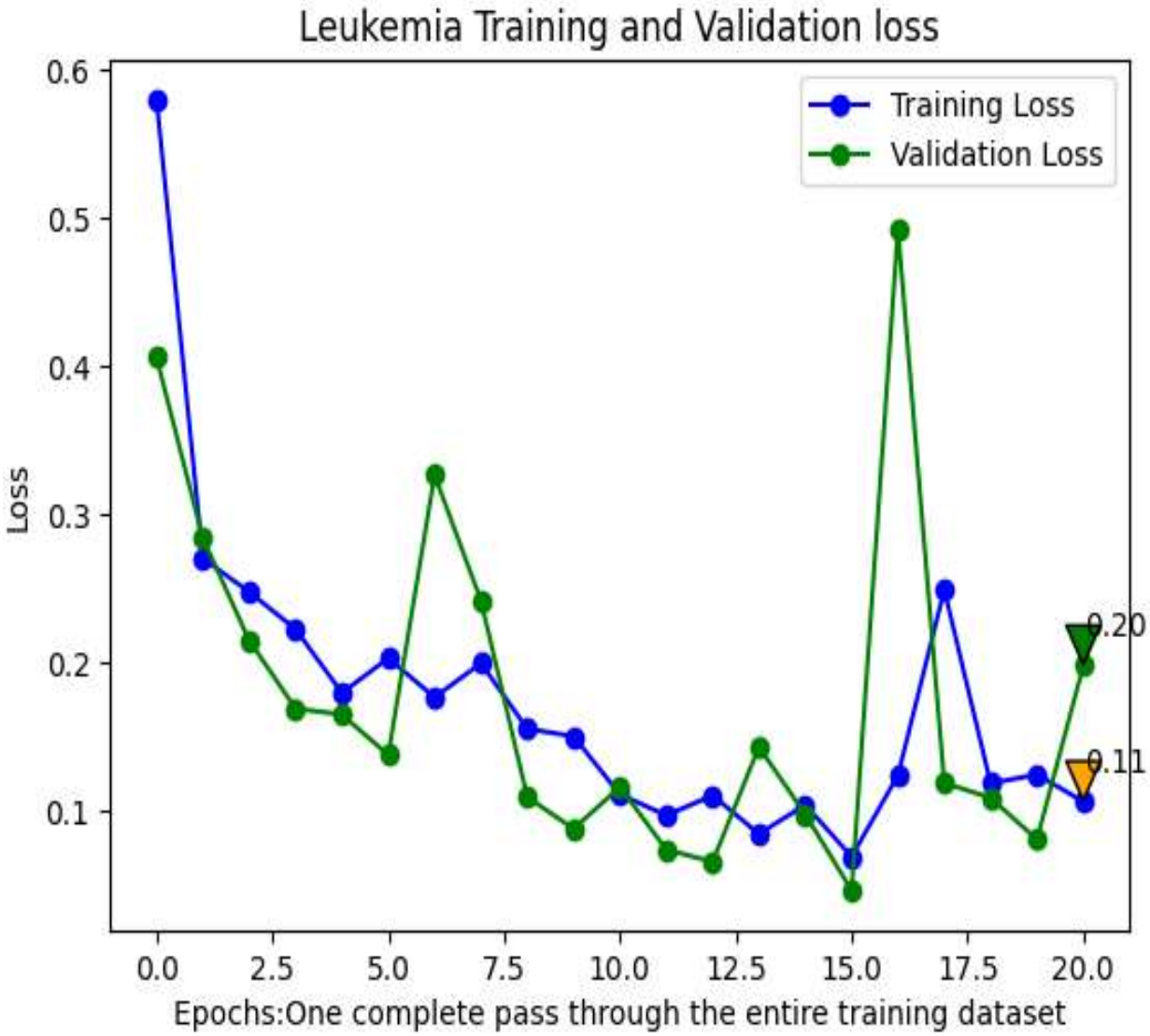


Figure 4: Model Loss

Figure 3 for model accuracy as it appears, it has completed more than 20 epochs. Within this learning process we have monitored both training and validation metrics. The blue curve for training the model and it learns well over the dataset except where it has curve variation that happen mostly between 13-17epochs suggests that the model's performance fluctuated during training, this is possibly caused by optimization process encountering local minima or noisy data. However after 17 epoch the model converged to a stable solution, leading to the consistent increase in accuracy and that tells us that the model has learned in the last epochs and predicts with more confidence. The same for Validation curve we observe more considerable variations in drop at the region near the one for training and stabilize to reach the maximum of 96% of accuracy as shiwn on figure 3; which is good for image classification in general

Figure 4 the loss graph, which measures the discrepancy between expected and actual values, shows how well the model performs. Better performance is shown by lower loss values.

The loss is first rather significant (around 0.6) and varies, showing that the model's predictions are not at all close to the actual values. The loss significantly decreases, indicating a steady improvement in the model's performance. Throughout the whole training phase, this pattern persists.

The model's performance significantly improves, as evidenced by the noticeable decline in loss. This implies that the model has improved its ability to forecast the future, as evidenced by the dramatic rise in accuracy that occurred around the same time. The validation loss shows considerable decrease as training loss they have even the same curvature and their value is shown in comparison table of this paper .

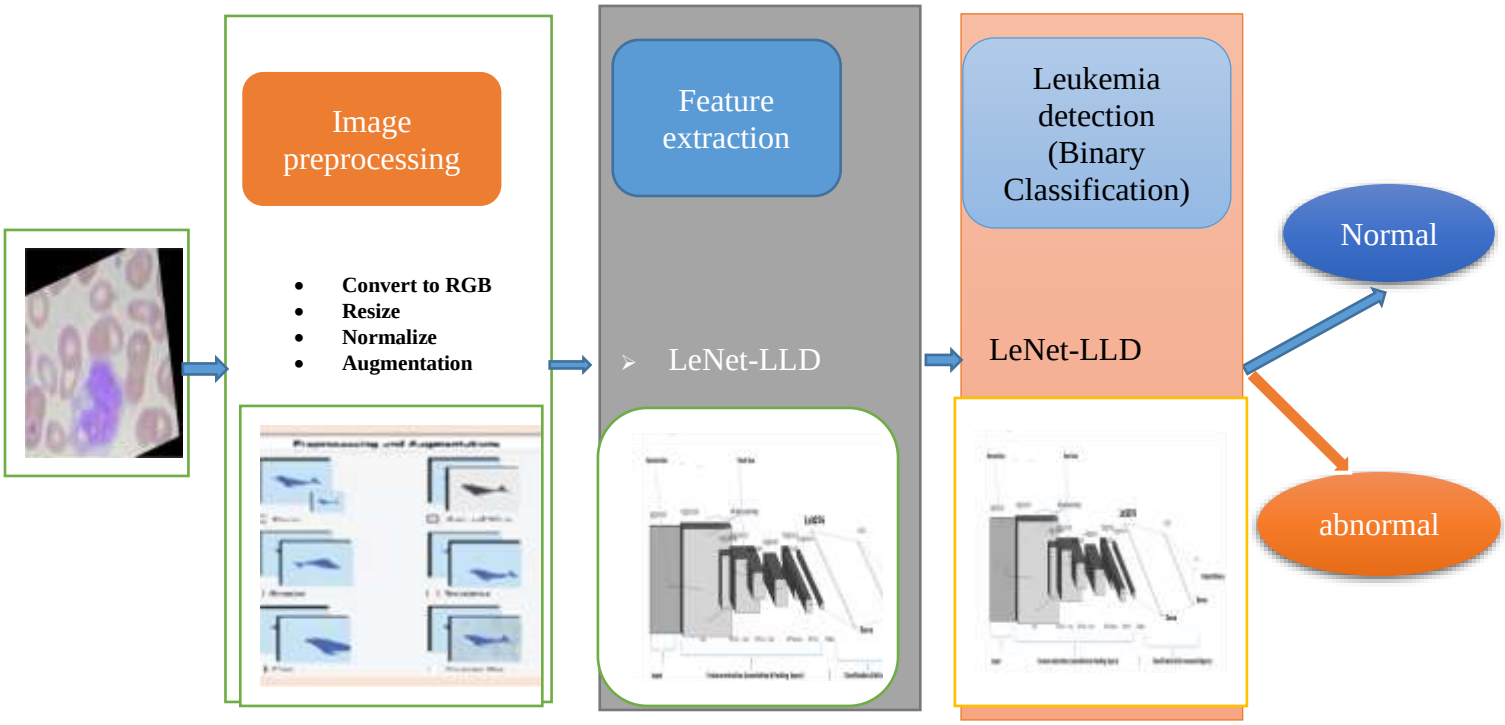


Figure 5:Framework of proposed Leukemia detection system

The Proposed Leukemia Detection Technique (PLDT) consists of three main phases as shown in Fig. 5 which are: (i) Image Preprocessing Phase (IPP), (ii) Feature Extraction Phase (FEP), and (iii) Classification Phase (CP).

3.3 Response to research questions

1 Performance Evaluation

- ✓ The performance is observed on the accuracy graph Figure 3 above, we have discussed the low rate of existing diognistic and the time it takes to get results. It goes to 2 or 3weeks in some case. As in traditional diagnosis the use to culture cells and wait their growth, and test it. On other side this trained model with high accurace has real time response once the image in provided at the input.; it has a good promise for early detection.

2 Generalization and Adaptation

- ✓ LeNet-LLD trained on diverse grobal dataset can not fully take account of all nuances of local population's health profile and disease prevalence. It is very capital to assess the representatives of the training data in relation with Rwandan context as the way and microscopy used to take image can introduce additional parameter.

- ✓ Patient demographics can differ dramatically within groups, including age, ethnicity, genetics, and environmental factors. These regional variations may not be taken into account by LeNet-LLD models trained on datasets from other areas, which could result in biased or erroneous predictions.

3**. Data Availability and Quality**

- ✓ The contacted personnel have refused the availability of such dataset and If is not the case the personnel in charge may not willing to provide
- ✓ Rwanda healthcare in ourdays has advanced in technology but still developing as we now don’t have datacenters for this sector and without this we can be sured on the quality of the data as storage matter.

3.4 Summary

We have used 6202 images divided into two classes’ negative and positive cancer that we have use to train, validate and test the model. With two classes 0 and 1 that represent negative and positive we have chosen two use binary classification and select the specific threshold to separate the two. Choosing the right threshold is essential for medical image classification in order to determine the binary classification outcome, which might be recognizing a particular pathology or differentiating between healthy and diseased tissues; we have used manual thresholding.

CHAPTER 4. THE PROJECT IMPLEMENTATION

This section examines the results from the "Design and Developing leukemia detection using deep learning" project's simulation and/or implementation phases. This section provides an overview of our research endeavors and provides an analysis of the efficiency and functionality of the solutions that were created.

The Design interface of this system is simple what means that it is user friendly system. It has a clean and intuitive layout with two main components:

Image Upload: By just choosing the file from their device, users can quickly upload an image of a blood stain. To reduce extra stages and hassles, this process has been shortened.

Prediction Button: After uploading the image, on the same page user can click the "Predict" button to start the prediction process. When you do this action, the image data is transmitted to the backend and processed by our deep learning model to produce predictions back to the webpage showing the class of the uploaded image and the probability closer to 1 indicate Normal or Negative otherwise Abnormal or Positive.

User experience

Because of its straightforward design, users can navigate and use the interface with little help or explanation. When a user clicks the prediction button, they are immediately notified that their request is being handled. The outcomes are quickly shown on the website after the forecast is finished.

In general, user-friendliness, efficiency, and simplicity are given a top priority in our interface design. We want people utilizing our deep learning-based solution to identify leukemia to have a smooth experience by reducing complexity and optimizing the user interaction process.

Our solution's modular design and usage of widely supported technologies make it simple to integrate into current workflows or systems.

Python and Flask: Our system enables flexibility and compatibility with a wide range of systems and environments by utilizing Python and Flask for the backend implementation. Flask makes it simple to create web services, which facilitates seamless integration.

TensorFlow: With copious documentation and support, TensorFlow is a well-liked deep learning system. Our solution can be seamlessly integrated into current workflows thanks to its compatibility with various technologies.

The trained models and backend components of our system may be readily extracted and deployed in production contexts, such as cloud-based or on-premises infrastructure, even if it was first developed in a notebook environment for Developing and development reasons.

4.1 Running the Leukemia detector system via web browser

As we discussed earlier the system is web based application because we can access it via web browser and display the results within that web browser. It is straightforward process that users with varying levels of technical expertise should be able to follow easily. As a little aside, it could be useful to give some more background, like installing Flask and any required dependencies beforehand, for users who might not be experienced with running Flask apps. An expanded version of the instructions is as follows:

Go to the Directory by navigating there: Launch the command line interface (CMD) and use the cd command to go to the directory containing the Flask application. As an illustration

CD path/to/your/flask/app by here you may need to install some dependencies if not installed and run the app by typing python filename.py . The Flask application is initialized and made usable at the given URL (usually localhost: 5000) by using this command. Let screenshot all of those steps and have preview bellow here.

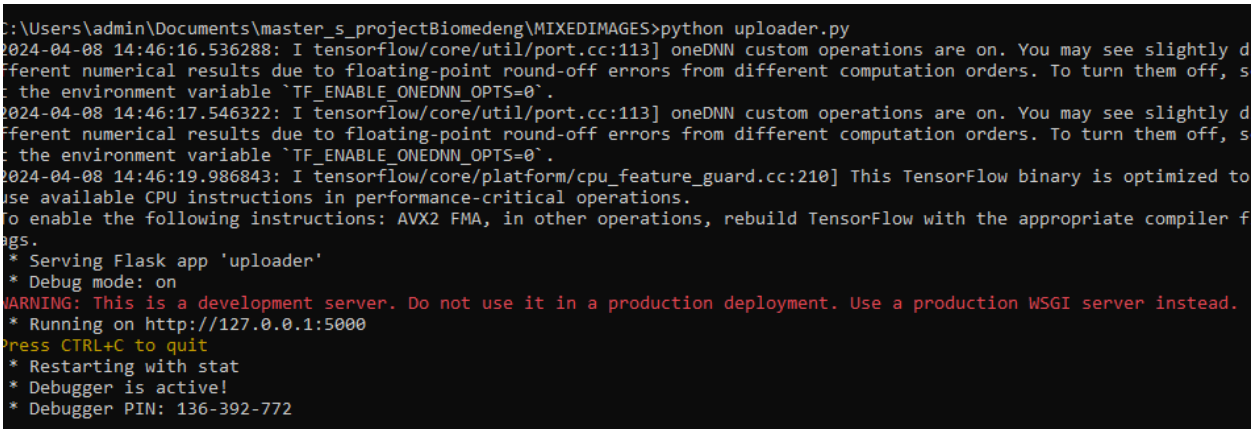


Figure 6:screenshot (Running Flask)

After running with success the Flask App, recall the trained model is included in this flask app and other required code to resize and normalize image before being classified as positive or negative; all code will be provided in the appendix.

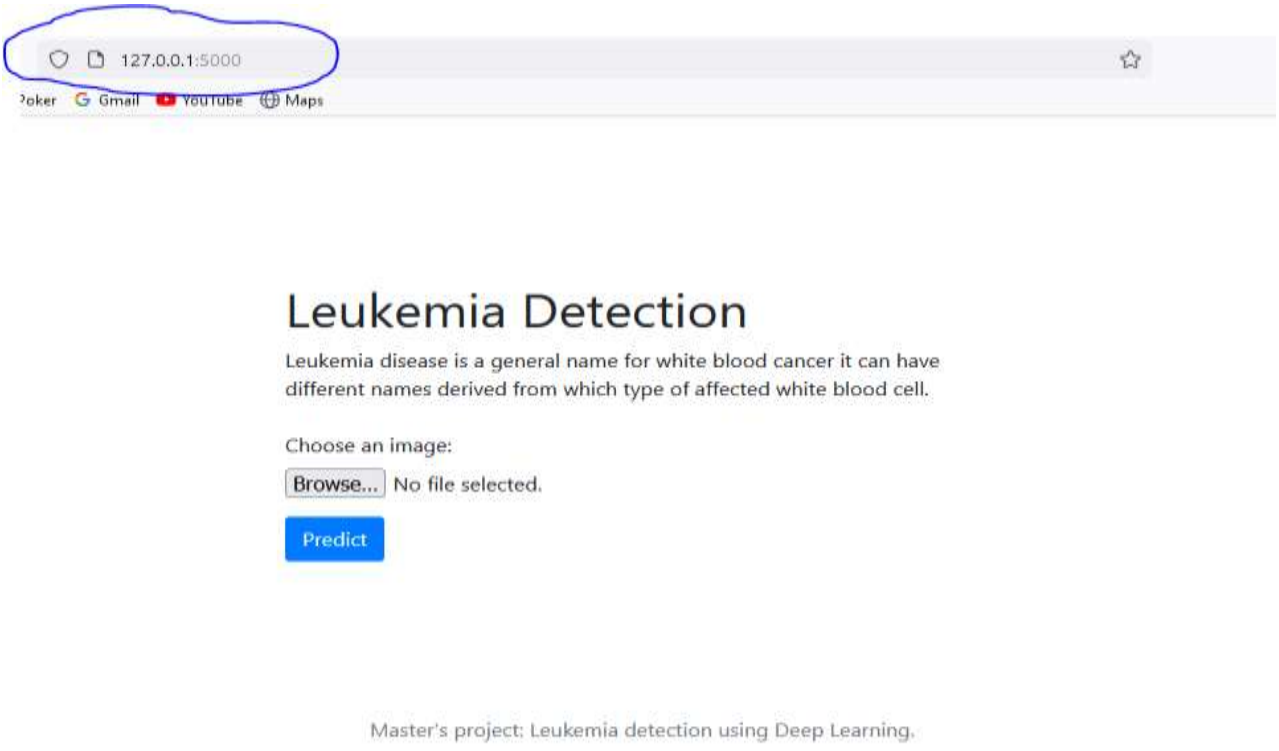


Figure 7 :web interface of Leukemia detection system

Once the image is selected using browse input; the next step is to classify the image by clicking the predict button. Let assume that the image is select next screenshot will show the results that will be explained later.

Figure 8: Prediction results

4.2 interpretation of results and discussion

4.2.1 Results

To distinguish between normal and abnormal photos or medical images from patients , we particularly used binary classification to train our LeNet-LLD model. Let's examine our approach to this task. Let us first explain the approach used to determine the likelihood or degree of confidence that an image is normal. The reason we use the term "normal" is that our model was trained to identify features in an image that are indicative of normalcy. A chosen image is considered normal when its confidence in normalcy is close to 1, otherwise it is considered abnormal. It is important, especially for medical applications, to separate these images using a threshold. We are showing in table below images with their class and confidence using our graphic user interface and this can be reach through model page after run the model code itself.

Table 1: Prediction results

Sample image			**Predictions**		
S/N	**Image name**	**From type**	**Class**	**Confidence**	**%**
1	22.jpeg	Negative	Normal	0.9999664425849915	100%
2	_0_3975.jpeg	Negative	Normal	0.9994944930076599	99.95%
3	_3_9457.jpeg	Positive	Abnormal	1.559542290230*10e-13	0.00%
4	_1_4511.jpeg	Positive	Abnormal	3.312291880774*10e-07	0.00%

The classification outcomes of our model are explained in detail in the table above. It distinguishes between two groups: Abnormal (signaling the existence of cancer) and Normal (signaling people who do not have cancer). We used our system to make predictions on the sample images shown in the table. As evidenced by the encouraging results of the obtained performance metrics, our model may be useful in leukemia diagnosis. Specifically, our model's resilience is demonstrated by its Precision of 0.99115 **(99%),** Recall of 0.96855345 **(96%)** , and F1 Score of 0.97972166 **(97%)**. Our LeNet-LLD model has the potential to accurately diagnose this crippling illness; with an additional training using a large dataset obtained from Rwanda's healthcare system, it can work on local settings.

The proposed LeNet-LLD model for leukemia detection was evaluated and compared with existing models using several key performance metrics in next section of this paper. These metrics include accuracy, precision, recall, F1 score, and loss value. By comparing these criteria, the effectiveness of LeNet-LLD in reliably diagnosing leukemia in medical images can be evaluated and compared

to other advanced algorithms. This comparison helps to emphasize the strengths and potential areas for improvement of the LeNet-LLD model, ensuring a complete grasp of its performance relative to other methods in the field.

4.2.2 Comparison and Discussion

With this expermet we have made comparison with related works that have been used in leterature review of this paper. The comparison of the proposed model with the existing work on leukemia cancer detection as shown in table 2 .It will show defferent evaluation metrics such accuracy,precission , recall and F1 score all of those metrics have been discussed and explain above .

Table 2: Comparison of existing works of deep learning model with the proposed model

#	Author	Model	Loss value	Accurancy %	Precision %	Recall %	F1 score %
1	F. Scotti [22]	CNN	--	92.00	--	--	--
2	M. Bukhari [12]	Squeeze and Excitation	0.117	98.3	98.16	98.16	98.43
3	Anilkumar K, Manoj V [5]	AlexNet LeukNet	--	94.12	--	--	--
4	M. Claro *et al.*[16]	AlertNet-RWD	0.0940	97.18	97.23	97.18	--
5	**Proposed work**	**LeNet-LLD**	**0.11**	**96**	**99**	**96**	**97.9**

The comparison of multiple deep learning models for leukemia detection highlights the advantages and disadvantages of each strategy. F. Scotti's CNN model has an accuracy of 92.00%, but it lacks further performance measures. M. Bukhari's model, which employs a Squeeze and Excitation network, stands out for its high accuracy of 98.3%, precision of 98.16%, recall of 98.16%, and remarkable F1 score of 98.43 percent. Anilkumar K and Manoj V AlexNet LeukNet has an

accuracy of 94.12%, but no other metrics are published. M. Claro et al.'s AlertNet-RWD performs well, with a loss value of 0.0940, accuracy of 97.18%, precision of 97.23%, and recall of 97.18%. In comparison, the proposed LeNet-LLD model has a competitive accuracy of 96%, the maximum precision of 99%, a recall of 96%, and a well-balanced F1 score of 97.9%. This shows that the LeNet-LLD model excels in precision, limiting false positives, and maintaining a solid overall performance, indicating its usefulness in leukemia diagnosis when compared to other models.
The LeNet-LLD model's high precision shows that it is good at reducing false positives, which is especially essential in leukemia detection, where a false positive can cause unneeded stress and medical procedures in patients. The balanced F1 score indicates that the model is capable of handling a variety of detection tasks, including identifying true cases and minimizing errors.

The efficacy of these models can be affected by the variety and magnitude of the datasets utilized, as well as the geographic origin of the data.
Models trained on heterogeneous datasets that encompass changes in patient demographics, medical imaging equipment, and regional healthcare practices may demonstrate enhanced generalizability and resilience.

4.3 Summary

In conclusion, while each model has advantages, the LeNet-LLD model stands out due to its high precision and balanced performance metrics. When comparing these models, it is critical to consider the dataset's diversity as well as geographical differences in medical imaging, both of which can have a major impact on model performance and generalizability. The excellent precision of the LeNet-LLD suggests that it could be a great candidate for deployment in clinical contexts, where avoiding false positives is as critical as optimizing total accuracy.

The table provide detailed predictions made by the trained LeNet-LLD model on sampled images from unseen, showcasing its ability to classify them into their respective classes with varying degrees of confidence. These results, coupled with performance metrics few of them are Precision, Recall, and F1 Score, underscored the robustness of the model. Specifically, with a Precision of 0.99115, Recall of 0.9685, and F1 Score of 0.9797, our model demonstrated promising capabilities in leukemia diagnosis even we recommend further investigation after screening using the system.

The comparison made with related works used in the literature review shows a correlation with the values of the metric measure of the model that is the Squeeze and Excitation on loss value of

0.117 where LeNet-LLD has 0.11 , precision has 0.98 where 0.99 other values can be seen in the table and this ensures the robustness of our trained model

Overall, our model shows potential for effectively diagnosing (classifying) leukemia with further training using extensive datasets sourced from Rwanda's healthcare system.

CHAPTER 5. CONCLUSION AND RECOMMENDATION

5.1 Conclusion

This study presented the outcome of efforts to develop and evaluate a deep learning for leukemia detection system and the system worked as expected. We have developed robust LeNet-LLD model that can accurately identify leukemia from medical images by applying deep learning and sophisticated techniques for medical image analysis. The results of testing and evaluation on comparison made of the system to other related works demonstrate how beneficial they can be for improving patient outcomes and diagnostic accuracy in the detection of leukemia. The LeNet-LLD model's high precision indicates its effectiveness at reducing false positives, which is particularly important in leukemia detection where a false positive can lead to unnecessary stress and medical interventions for patients. The balanced F1 score suggests the model is robust in handling various aspects of detection, from identifying true cases to minimizing errors

The absence of data with in the country created difficulties, but making use of public databases was essential to work around this restriction. We successfully trained and evaluated our deep learning model using publically available datasets, which facilitated the development of automated leukemia detection system. This highlights the importance of open data initiatives in fostering research and innovation, particularly in resource-constrained settings.

5.2 Recommendations

We advise further investigation and improvement of deep learning-based methods for leukemia detection to the scientific community. By collecting and disseminating more varied and representative datasets, more research should be done to overcome the data shortage within the country. The development and assessment of leukemia detection systems can be greatly improved by cooperative efforts focused at gathering and annotating medical imaging data.

We recommend the use of deep learning and AI applications at district hospitals level, because they have the potential to completely transform the way diseases are diagnosed, especially when it comes to early detection of diseases like leukemia. District hospitals can use AI to optimize healthcare delivery, improve patient outcomes, and increase diagnostic accuracy.

This will ultimately help to achieve the goal of providing high-quality, easily accessible healthcare to everyone, regardless of socioeconomic background or location.

REFERENCES

[1] F. Rubagumya *et al.*, "State of Cancer Control in Rwanda: Past, Present, and Future Opportunities," 2020. [Online]. Available: https://ascopubs.org/go/authors/open-access

[2] F. M. Talaat and S. A. Gamel, "Machine learning in detection and classification of leukemia using C-NMC_Leukemia," *Multimed Tools Appl*, Jan. 2023, doi: 10.1007/s11042-023-15923-8.

[3] "Classification of Leukemia Disease in Peripheral Blood Cell Images Using Convolutional Neural Network".

[4] M. Rana and M. Bhushan, "Machine learning and deep learning approach for medical image analysis: diagnosis to detection," *Multimed Tools Appl*, vol. 82, no. 17, pp. 26731–26769, Jul. 2023, doi: 10.1007/s11042-022-14305-w.

[5] K. K. Anilkumar, V. J. Manoj, and T. M. Sagi, "Automated Detection of B Cell and T Cell Acute Lymphoblastic Leukaemia Using Deep Learning," *IRBM*, vol. 43, no. 5, pp. 405–413, Oct. 2022, doi: 10.1016/j.irbm.2021.05.005.

[6] WORLD HEALTH ORGANIZATION: REGIONAL OFFICE FOR EUROPE., *WORLD CANCER REPORT : cancer research for cancer development.* IARC, 2020.

[7] J. Lyons, "The polymerase chain reaction and cancer diagnostics," *Cancer*, vol. 69, no. 6 S, pp. 1527–1531, 1992, doi: 10.1002/1097-0142(19920315)69:6+<1527::AID-CNCR2820691304>3.0.CO;2-N.

[8] M. Z. Ullah *et al.*, "An attention-based convolutional neural network for acute lymphoblastic leukemia classification," *Applied Sciences (Switzerland)*, vol. 11, no. 22, Nov. 2021, doi: 10.3390/app112210662.

[9] E. G. Kazanci and E. Büyükkaya, "ARTIFICIAL INTELLIGENCE THEORY and APPLICATIONS Follow-up of patients Using Artificial Intelligence During The Pandemic and Its Application In The Diagnosis of Leukemia," 2021.

[10] G. B. W. Wertheim, R. Daber, and A. Bagg, "Molecular diagnostics of acute myeloid leukemia it's a (next) generational thing," 2013, *Elsevier B.V.* doi: 10.1016/j.jmoldx.2012.08.002.

[11] C. V. S. A. S.-H. L. and K.-R. K. T. T. P. Thanh, “Leukemia Blood Cell Image Classification Using Convolutional Neural Network”.

[12] M. Bukhari, S. Yasmin, S. Sammad, and A. A. Abd El-Latif, “A Deep Learning Framework for Leukemia Cancer Detection in Microscopic Blood Samples Using Squeeze and Excitation Learning,” *Math Probl Eng*, vol. 2022, 2022, doi: 10.1155/2022/2801227.

[13] A. Rehman, N. Abbas, T. Saba, S. I. ur Rahman, Z. Mehmood, and H. Kolivand, “Classification of acute lymphoblastic leukemia using deep learning,” *Microsc Res Tech*, vol. 81, no. 11, pp. 1310–1317, Nov. 2018, doi: 10.1002/jemt.23139.

[14] M. A. Abu, N. Hazirah Indra, A. Halim, A. Rahman, A. Sapiee, and I. Ahmad, “A study on Image Classification based on Deep Learning and Tensorflow,” 2019. [Online]. Available: http://www.irphouse.com563

[15] S. Ansari, A. H. Navin, A. Babazadeh Sangar, J. Vaez Gharamaleki, and S. Danishvar, “Acute Leukemia Diagnosis Based on Images of Lymphocytes and Monocytes Using Type-II Fuzzy Deep Network,” *Electronics (Switzerland)*, vol. 12, no. 5, Mar. 2023, doi: 10.3390/electronics12051116.

[16] M. Claro *et al.*, “Convolution Neural Network Models for Acute Leukemia Diagnosis,” in *International Conference on Systems, Signals, and Image Processing*, IEEE Computer Society, Jul. 2020, pp. 63–68. doi: 10.1109/IWSSIP48289.2020.9145406.

[17] I. Joseph Maria, T. Devi, and D. Ravi, “Machine Learning Algorithms For Diagnosis Of Leukemia,” *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH*, vol. 9, p. 1, 2020, [Online]. Available: www.ijstr.org

[18] A. Rehman, N. Abbas, T. Saba, S. Ijaz Ur Rahman, Z. Mehmood, and H. Kolivand, “Classification of acute lymphoblastic leukemia using deep learning.” [Online]. Available: http://www.hematology.org/

[19] N. Bibi, M. Sikandar, I. U. Din, A. Almogren, and S. Ali, “IOMT-based automated detection and classification of leukemia using deep learning,” *J Healthc Eng*, vol. 2020, 2020, doi: 10.1155/2020/6648574.

[20] N. Sampathila *et al.*, “Customized Deep Learning Classifier for Detection of Acute Lymphoblastic Leukemia Using Blood Smear Images,” *Healthcare (Switzerland)*, vol. 10, no. 10, Oct. 2022, doi: 10.3390/healthcare10101812.

[21] R. Ashraf *et al.*, “Deep Convolution Neural Network for Big Data Medical Image Classification,” *IEEE Access*, vol. 8, pp. 105659–105670, 2020, doi: 10.1109/ACCESS.2020.2998808.

[22] F. Scotti, “Robust Segmentation and Measurements Techniques of White Cells in Blood Microscope Images,” 2006.

APPENDICES

Appendix 1: The Utilized Codes

1.1 Training the Model code (notebook is written in cells)

```
 for the first time uncomment and install all of those dependancies libraly
!pip install pandas
!pip install opencv-python
!pip install numpy
!pip install tqdm
!pip install scipy

!pip install scikit-learn
!pip install keras
!pip install tensorflow
!pip install pyarrow
!pip install scikit-image
!pip install mlxtend
!pip install focal_loss
!pip list
```

```
port tensorflow as tf


gpus = tf.config.experimental.list_physical_devices('GPU')

if gpus:
    try:
        for gpu in gpus:
            tf.config.experimental.set_memory_growth(gpu, True)
```

```
        logical_gpus = tf.config.experimental.list_logical_devices('GPU')
        print(len(gpus), "Physical GPUs,", len(logical_gpus), "Logical GPUs")
    except RuntimeError as e:
        print(e)


tf.config.list_physical_devices('GPU')

import pandas as pd
import cv2
import numpy as np
import os , sys
import os



from random import shuffle
from tqdm import tqdm
import scipy
import skimage
from skimage.transform import resize
from sklearn.model_selection import train_test_split
import imghdr


# Avoid OOM errors by setting GPU Memory Consumption Growth
gpus = tf.config.experimental.list_physical_devices('GPU')
for gpu in gpus:
    tf.config.experimental.set_memory_growth(gpu, True)

use_cuda = True

print(os.listdir("cancer/train"))

TRAIN_DIR = "cancer/train/"
#TEST_DIR =  "cancer/test/"
```

```
image_exts = ['jpeg','jpg', 'bmp', 'png']

for image_class in os.listdir(TRAIN_DIR):
    for image in os.listdir(os.path.join(TRAIN_DIR, image_class)):
        train_image_path = os.path.join(TRAIN_DIR, image_class, image)
        try:
            img = cv2.imread(train_image_path)
            tip = imghdr.what(train_image_path)
            if tip not in image_exts:
                print('Image not in ext list {}'.format(train_image_path))
                os.remove(train_image_path)
        except Exception as e:
            print('Issue with image {}'.format(train_image_path))
            # os.remove(image_path)

import numpy as np
from matplotlib import pyplot as plt
from keras import backend as K

from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.callbacks import ReduceLROnPlateau, ModelCheckpoint
from tensorflow.keras.optimizers import Adam
import tensorflow as tf




# augmentation parameters
datagen = ImageDataGenerator(
    rotation_range=30,                width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest')
```

```
train_datasets = datagen.flow_from_directory(
    TRAIN_DIR,

    color_mode='rgb',
      target_size=(150, 150),
    batch_size=32,
    class_mode='categorical',
    shuffle=True)

train_dataset= tf.keras.utils.image_dataset_from_directory(TRAIN_DIR)




data_iterator = train_dataset.as_numpy_iterator()
batch = data_iterator.next()

fig, ax = plt.subplots(ncols=4, figsize=(20,20))
for idx, img in enumerate(batch[0][:4]):
    ax[idx].imshow(img.astype(int))
    ax[idx].title.set_text(batch[1][idx])



train_size = int(len(train_dataset)*.500)
val_size = int(len(train_dataset)*.352)
test_size = int(len(train_dataset)*.150)
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Dense, Flatten, Dropout,
BatchNormalization
from tensorflow.keras.callbacks import EarlyStopping, TensorBoard
from tensorflow.keras.layers import RandomFlip, RandomRotation, RandomZoom
```

```
train_dataset= tf.keras.utils.image_dataset_from_directory(TRAIN_DIR,
                                                           validation_split=0.4,
    subset='training',
    seed=123,
    image_size=(256, 256),
    batch_size=32)

# Data augmentation
data_augmentation = tf.keras.Sequential([
    RandomFlip("horizontal_and_vertical"),
    RandomRotation(0.2),
    RandomZoom(0.2),
])
train_dataset    =    train_dataset.map(lambda    x,    y:    (data_augmentation(x,
training=True), y))

val_dataset = tf.keras.utils.image_dataset_from_directory(
    TRAIN_DIR,
    validation_split=0.4,  # 20% for validation
    subset='validation',
    seed=123,
    image_size=(256, 256),
    batch_size=32
)
```

```
import tensorflow as tf
from tensorflow.keras import layers, models
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Dense, Flatten, Dropout
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout,
BatchNormalization
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Dense, Flatten, Dropout,
BatchNormalization
```

```
from tensorflow.keras.callbacks import EarlyStopping
from keras import regularizers

# Define the model architecture
model = Sequential()

#model.add(Conv2D(16, (3, 3), 1, activation='relu', input_shape=(256, 256, 3)))
model.add(Conv2D(16, (3, 3), 1, activation='relu', input_shape=(256, 256, 3)))
model.add(MaxPooling2D())
#model.add(BatchNormalization())
#model.add(Dropout(0.3))

model.add(Conv2D(32, (3, 3), 1, activation='relu') )
model.add(MaxPooling2D())
#model.add(BatchNormalization())
#model.add(Dropout(0.3))

model.add(Conv2D(64, (3, 3), 1, activation='relu'))
model.add(MaxPooling2D())
#model.add(BatchNormalization())
#model.add(Dropout(0.3))

model.add(Conv2D(128, (3, 3), 1, activation='relu'))
model.add(MaxPooling2D())
#model.add(BatchNormalization())
#model.add(Dropout(0.3))

model.add(Flatten())
model.add(Dense(256, activation='relu'))
#model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))

# Compile the model
model.compile(optimizer='adam',              loss=tf.losses.BinaryCrossentropy(),
metrics=['accuracy'])

# Print the model summary
#model.summary()
```

```
# Define early stopping callback

fig = plt.figure()
plt.plot(hist.history['loss'], color='teal', label='train loss')
plt.plot(hist.history['val_loss'], color='orange', label='validation loss')
fig.suptitle('Loss', fontsize=20)
plt.legend(loc="upper left")
plt.show()

fig = plt.figure()
plt.plot(hist.history['accuracy'], color='teal', label='accuracy')
plt.plot(hist.history['val_accuracy'], color='orange', label='val_accuracy')
fig.suptitle('Accuracy', fontsize=20)
plt.legend(loc="upper left")
plt.show()

from tensorflow.keras.models import save_model, load_model


model.save('MythesisModel.keras’, overwrite=True)




from keras.models import load_model
model.summary()
from keras.models import load_model

# Load the model
model = load_model(‘MythesisModel.keras')


from tensorflow.keras.metrics import Precision, Recall, BinaryAccuracy

pre = Precision()
re = Recall()
acc = BinaryAccuracy()
```

```
for batch in test.as_numpy_iterator():
    X, y = batch
    yhat = model.predict(X)
    pre.update_state(y, yhat)
    re.update_state(y, yhat)
    acc.update_state(y, yhat)

print(pre.result(), re.result(), acc.result())

import numpy as np

# Extract features and labels from the training set
x_train = []
y_train = []
for features, labels in train_dataset:
    x_train.append(features.numpy())
    y_train.append(labels.numpy())

# Convert to numpy arrays
x_train = np.concatenate(x_train, axis=0)
y_train = np.concatenate(y_train, axis=0)

# Extract features and labels from the validation set
x_val = []
y_val = []
for features, labels in val_dataset:
    x_val.append(features.numpy())
    y_val.append(labels.numpy())

# Convert to numpy arrays
x_val = np.concatenate(x_val, axis=0)
y_val = np.concatenate(y_val, axis=0)



# Verify the shapes
```

```
print(f"x_train shape: {x_train.shape}")
print(f"y_train shape: {y_train.shape}")
print(f"x_val shape: {x_val.shape}")
print(f"y_val shape: {y_val.shape}")
```

```
x_val = np.array(x_val)
y_val = np.array(y_val)
```

```
from sklearn.metrics import precision_score, recall_score, f1_score

# Let's set threshold = 0.5 for this example

threshold = 0.5
y_pred_probabilities = model.predict(x_test)
y_pred_binary = (y_pred_probabilities > threshold).astype(int)  # Replace
y_pred_probabilities with your actual predicted probabilities

precision = precision_score(y_true, y_pred_binary)
recall = recall_score(y_true, y_pred_binary)
f1 = f1_score(y_true, y_pred_binary)

print("Precision:", precision)
print("Recall:", recall)
print("F1 Score:", f1)
```

```
x_val = []
y_true = []  # Store the true labels here
for features, labels in val:
    x_val.append(features.numpy())
    y_true.append(labels.numpy())
     # Concatenate all batches in x_test along the batch dimension
x_val = np.concatenate(x_val, axis=0)

# Concatenate all batches in y_test along the batch dimension
y_true = np.concatenate(y_true, axis=0)

# Convert the lists to numpy arrays
x_val = np.array(x_val)
y_true = np.array(y_true)

import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import f1_score

# Assuming you have predicted probabilities and true labels
y_pred_probabilities = model.predict(x_val)


thresholds = np.arange(0.1, 1, 0.1)  # Threshold values from 0.1 to 0.9
f1_scores = []

for threshold in thresholds:
    # Convert predicted probabilities to binary predictions using the threshold
    y_pred_binary = (y_pred_probabilities > threshold).astype(int)

    # Calculate F1 score
    f1 = f1_score(y_true, y_pred_binary)
    f1_scores.append(f1)

# Plot F1 curve
plt.plot(thresholds, f1_scores, marker='o')
plt.xlabel('Threshold')
plt.ylabel('F1 Score')
```

```
plt.title('F1 Curve')
plt.grid(True)
plt.show()
```

1.2 Flask code to run the model

```
from flask import Flask, render_template, request, jsonify
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np
import io
from base64 import b64encode

app = Flask(__name__)

# Load the trained model
model = load_model('MyCancer_Mastersprojectymixed.keras', compile=False)

def preprocess_image(img):
    # Resize the image to the expected input shape (256, 256)
    img = img.resize((256, 256))

    # Convert the image to a NumPy array
    img_array = image.img_to_array(img)

    # Normalize pixel values to be between 0 and 1
    img_array = img_array / 255.0

    # Add batch dimension and channel dimension
    img_array = np.expand_dims(img_array, axis=0)

    return img_array

@app.route('/')
def upload_form():
    return render_template('index.html')

@app.route('/predict', methods=['POST'])

@app.route('/predict', methods=['POST'])
def predict():
```

```
    try:
        # Get the image from the request
        img_file = request.files['image']
        img_bytes = img_file.read()
        img_filename = img_file.filename

        # Encode the image data as base64
        img_data = b64encode(img_bytes).decode('utf-8')

        # Load the image
        img = image.load_img(io.BytesIO(img_bytes), target_size=(256, 256))

        # Preprocess the image
        img_array = preprocess_image(img)

        # Make a prediction
        prediction = model.predict(img_array)
        if float(np.max(prediction)) > 0.6:
            classestype = 'NORMAL'
        else:
            classestype = 'ABNORMAL OR NOT NORMAL'

        # Convert the prediction to a human-readable format
        result = {
            'class': classestype,
            'confidence': float(np.max(prediction)),
            'filename': img_filename,
            'image_data': img_data
        }

        # Render the result template with the prediction
        return render_template('result.html', prediction=result)

    except Exception as e:
        return jsonify({'error': str(e)})

if __name__ == '__main__':
    app.run(debug=True)
```

75.

1.3 HTML template for index to upload image

```
3  <!DOCTYPE html>
4  <html lang="en">
5  <head>
6      <meta charset="UTF-8">
7      <meta name="viewport" content="width=device-width, initial-scale=1.0">
8                                                                                <link
   href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.cs
   s" rel="stylesheet">
9      <title>Design And Developing Leukemia Detection Using Deep Learning</title>
10      <style>
11         /* Custom CSS to center the form */
12         .centered-form {
13             display: flex;
14             justify-content: center;
15             align-items: center;
16             height: 80vh; /* Adjust as needed */
17         }
18     </style>
19
20 </head>
21 <body>
22    <div class="container">
23         <div class="row centered-form">
24             <div class="col-md-6">
25                 <h1 class="mt-4">Leukemia Detection</h1>
26                  <p>Leukemia disease is a general name for white blood cancer
   it can have different names derived from which type of affected white blood
   cell.</p>
27                    <form action="http://localhost:5000/predict" method="post"
   enctype="multipart/form-data" class="mt-4">
28                     <div class="form-group">
29                         <label for="imageInput">Choose an image:</label>
30                              <input type="file" name="image" id="imageInput"
   class="form-control-file" accept="image/*" required>
```

```
31                    </div>
32                                    <button   type="submit"   class="btn   btn-
   primary">Predict</button>
33                </form>
34             </div>
35         </div>
36     </div>
37
38     <!-- Footer -->
39     <footer class="footer mt-auto py-3">
40         <div class="container text-center">
41             <span class="text-muted">Master's project: Leukemia detection using
   Deep Learning.</span>
42         </div>
43     </footer>
44 </body>
45 </html>
```

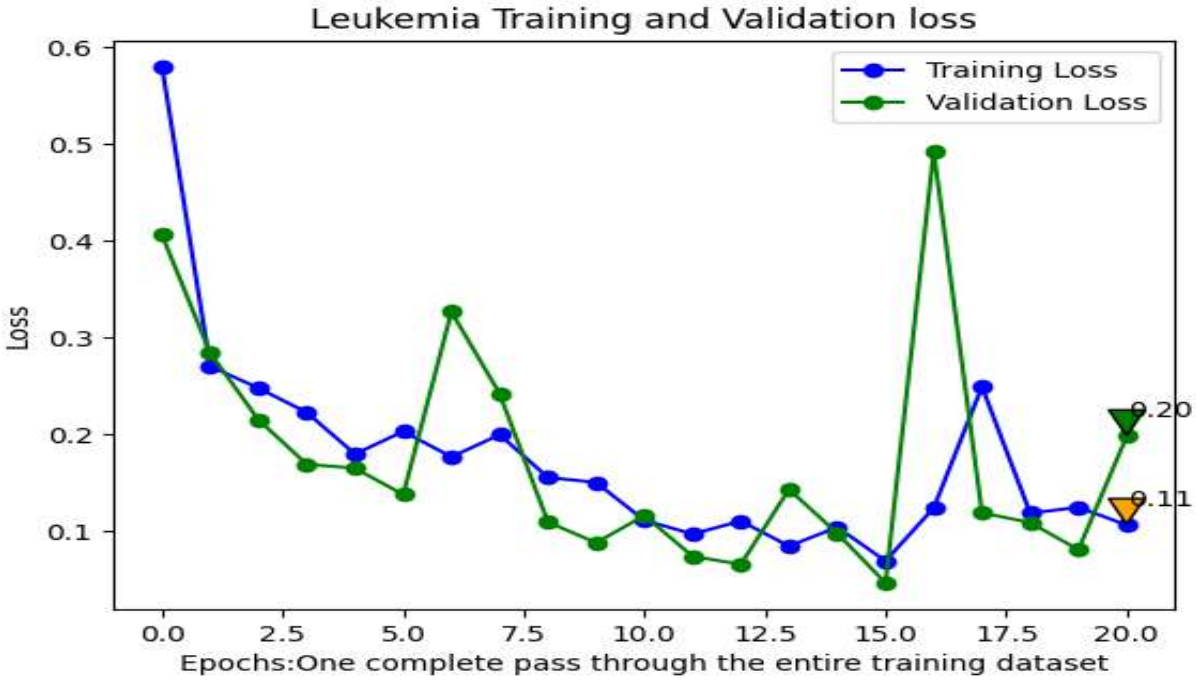
1.4 HTML code to display results

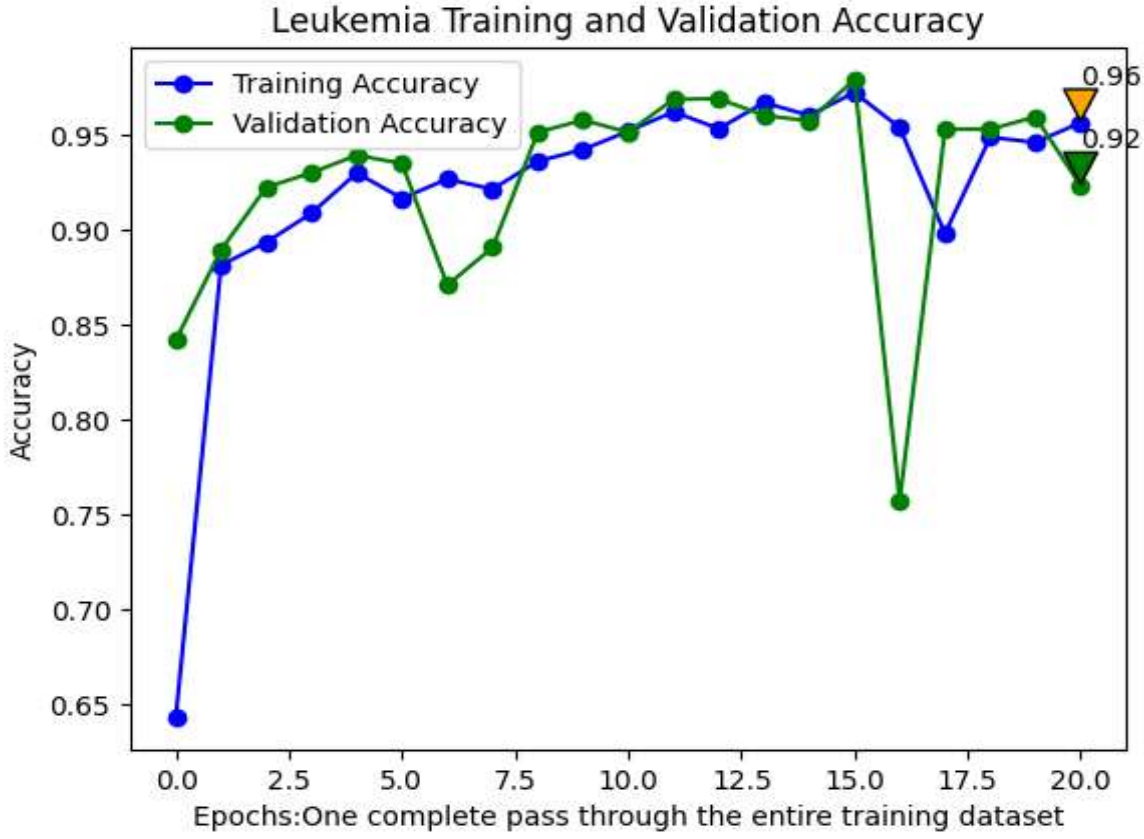
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Image Prediction Result</title>
    <!-- Include Bootstrap CSS -->
                                                                          <link
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css"
rel="stylesheet">
      <style>
        /* Custom CSS to center the form */
        .centered-form {
            display: flex;
            justify-content: center;
```

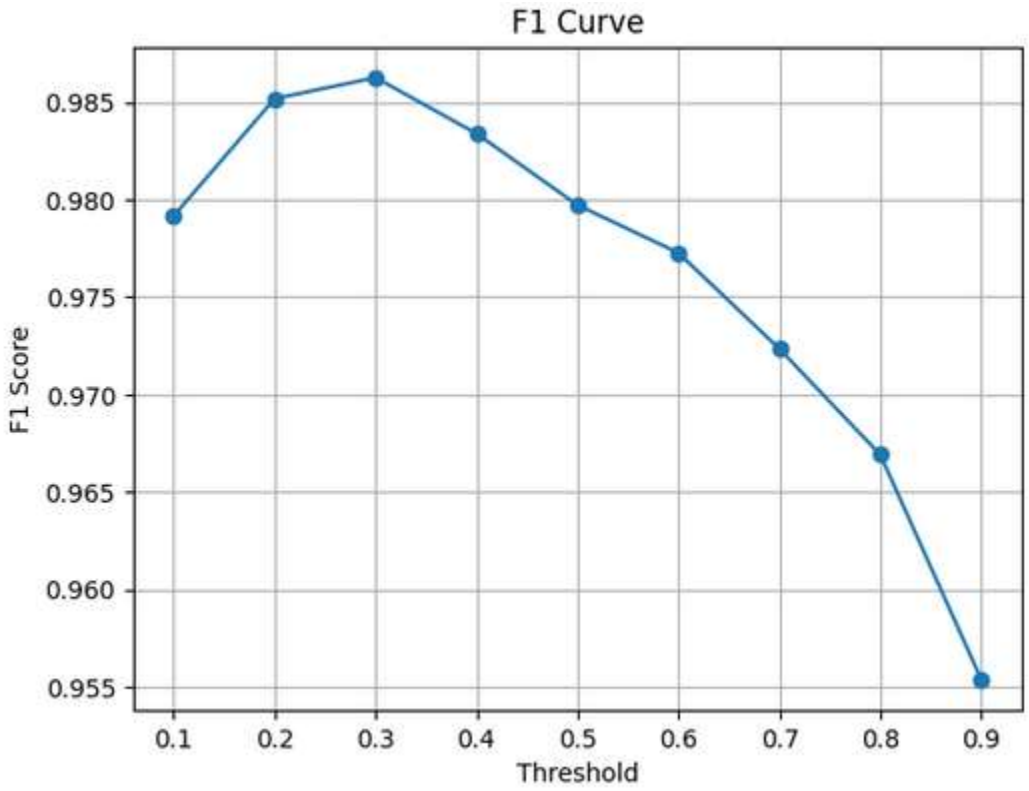
```
            align-items: center;
            height: 80vh; /* Adjust as needed */
        }
    </style>
</head>
<body>
    <div class="container centered-form">
        <div class="form-group">
        <h1 class="mt-4">Prediction Result( Leukemia Detection)</h1>
        <img src="{{ prediction.image_data }}" class="img-
fluid mt-4" alt="Uploaded Image">
        <p class="mt-4">Class: {{ prediction.class }}</p>
        <p>Confidence: {{ prediction.confidence }}</p>
        <p>Filename: {{ prediction.filename }}</p>
        <!-- Add a button -->
        <a href="/" class="btn btn-primary mt-4">Upload Another Image</a>
    </div>
        </div>

      <!-- Footer -->
    <footer class="footer mt-auto py-3">
        <div class="container text-center">
             <span class="text-muted">Master's project: Leukemia detection using
Deep Learning.</span>
        </div>
    </footer>
    </body>
</html>
```

Appendix 2: model training

```
Epoch 1/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 136s 925ms/step - accuracy: 0.5379 - loss: 0.6850 - val_accuracy: 0.8419 - val_loss: 0.4062
Epoch 2/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 104s 883ms/step - accuracy: 0.8702 - loss: 0.3046 - val_accuracy: 0.8895 - val_loss: 0.2842
Epoch 3/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 102s 873ms/step - accuracy: 0.9022 - loss: 0.2388 - val_accuracy: 0.9226 - val_loss: 0.2153
Epoch 4/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 101s 861ms/step - accuracy: 0.9067 - loss: 0.2277 - val_accuracy: 0.9302 - val_loss: 0.1695
Epoch 5/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 100s 856ms/step - accuracy: 0.9389 - loss: 0.1615 - val_accuracy: 0.9395 - val_loss: 0.1653
Epoch 6/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 100s 853ms/step - accuracy: 0.9158 - loss: 0.2032 - val_accuracy: 0.9351 - val_loss: 0.1381
Epoch 7/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 100s 855ms/step - accuracy: 0.9445 - loss: 0.1328 - val_accuracy: 0.8710 - val_loss: 0.3269
Epoch 8/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 100s 849ms/step - accuracy: 0.9256 - loss: 0.2051 - val_accuracy: 0.8907 - val_loss: 0.2424
Epoch 9/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 99s 842ms/step - accuracy: 0.9213 - loss: 0.2015 - val_accuracy: 0.9516 - val_loss: 0.1097
Epoch 10/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 100s 853ms/step - accuracy: 0.9427 - loss: 0.1579 - val_accuracy: 0.9581 - val_loss: 0.0884
Epoch 11/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 101s 861ms/step - accuracy: 0.9538 - loss: 0.1126 - val_accuracy: 0.9516 - val_loss: 0.1164
Epoch 12/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 103s 877ms/step - accuracy: 0.9636 - loss: 0.0928 - val_accuracy: 0.9690 - val_loss: 0.0743
Epoch 13/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 101s 864ms/step - accuracy: 0.9550 - loss: 0.1056 - val_accuracy: 0.9694 - val_loss: 0.0660
Epoch 14/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 99s 841ms/step - accuracy: 0.9650 - loss: 0.0880 - val_accuracy: 0.9605 - val_loss: 0.1435
Epoch 15/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 98s 839ms/step - accuracy: 0.9609 - loss: 0.1108 - val_accuracy: 0.9577 - val_loss: 0.0980
Epoch 16/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 99s 843ms/step - accuracy: 0.9709 - loss: 0.0664 - val_accuracy: 0.9794 - val_loss: 0.0468
Epoch 17/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 98s 839ms/step - accuracy: 0.9658 - loss: 0.0871 - val_accuracy: 0.7573 - val_loss: 0.4926
Epoch 18/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 102s 869ms/step - accuracy: 0.8518 - loss: 0.3320 - val_accuracy: 0.9532 - val_loss: 0.1192
Epoch 19/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 99s 843ms/step - accuracy: 0.9498 - loss: 0.1143 - val_accuracy: 0.9532 - val_loss: 0.1092
Epoch 20/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 101s 862ms/step - accuracy: 0.9455 - loss: 0.1322 - val_accuracy: 0.9597 - val_loss: 0.0811
Epoch 21/23
117/117 ━━━━━━━━━━━━━━━━━━━━ 105s 893ms/step - accuracy: 0.9599 - loss: 0.1015 - val_accuracy: 0.9234 - val_loss: 0.1983
```


Leukemia Training and Validation loss
Training Loss
Validation Loss
Loss
0.6
0.5
0.4
0.3
0.2
0.1
0.20
0.11
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Epochs:One complete pass through the entire training dataset


Leukemia Training and Validation Accuracy
Training Accuracy
Validation Accuracy
Accuracy
0.95
0.90
0.85
0.80
0.75
0.70
0.65
0.96
0.92
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Epochs:One complete pass through the entire training dataset




```python
from sklearn.metrics import precision_score, recall_score, f1_score

# Assuming y_true contains the ground truth labels and y_pred contains the predicted probabilities or predictions
# You need to convert the probability predictions to binary predictions using a threshold
# Let's assume threshold = 0.5 for this example

threshold = 0.5
y_pred_probabilities = model.predict(x_val)
y_pred_binary = (y_pred_probabilities > threshold).astype(int)  # Replace y_pred_probabilities with your actual predicted probabilities

precision = precision_score(y_val, y_pred_binary)
recall = recall_score(y_val, y_pred_binary)
f1 = f1_score(y_val, y_pred_binary)

print("Precision:", precision)
print("Recall:", recall)
print("F1 Score:", f1)
```

```
78/78 ━━━━━━━━━━━━━━━━━━━━ 170s 566ms/step
Precision: 0.9911504424778761
Recall: 0.9685534591194969
F1 Score: 0.9797216699801192
```

Model: "sequential_6"

Layer (type)	Output Shape	Param #
conv2d_16 (Conv2D)	(None, 254, 254, 16)	448
max_pooling2d_16 (MaxPooling2D)	(None, 127, 127, 16)	0
conv2d_17 (Conv2D)	(None, 125, 125, 32)	4,640
max_pooling2d_17 (MaxPooling2D)	(None, 62, 62, 32)	0
conv2d_18 (Conv2D)	(None, 60, 60, 64)	18,496
max_pooling2d_18 (MaxPooling2D)	(None, 30, 30, 64)	0
conv2d_19 (Conv2D)	(None, 28, 28, 128)	73,856
max_pooling2d_19 (MaxPooling2D)	(None, 14, 14, 128)	0
flatten_5 (Flatten)	(None, 25088)	0
dense_8 (Dense)	(None, 256)	6,422,784
dense_9 (Dense)	(None, 1)	257

Total params: 19,561,445 (74.62 MB)

Trainable params: 6,520,481 (24.87 MB)

Non-trainable params: 0 (0.00 B)

Optimizer params: 13,040,964 (49.75 MB)