Comparative Analysis of Neural Network Models for Peripheral blood cell Image Classification

by

Md Tanzid Mollah
22141053
Mahi Shahriar
19301252
Mohammad Fahim
19301041
Zehan Ahmed
19301243
Kaji Sadman Sakib
19301059

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science and Engineering

Department of Computer Science and Engineering
School of Data and Sciences
Brac University
September 2023

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Md Tanzid Mollah
22141053

Mahi Shahriar
19301252

Mohammad Fahim
19301041

Zehan Ahmed
19301243

Kaji Sadman Sakib
19301059

Approval

The thesis/project titled "Comparative Analysis of Neural Network Models for Peripheral blood cell Image Classification" submitted by

1. Md Tanzid Mollah(22141053)
2. Mahi Shahriar (19301252)
3. Mohammad Fahim (19301041)
4. Kaji Sadman Sakib (19301059)
5. Zehan Ahmed (19301243)

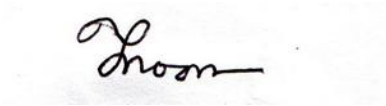
Summer, 2023 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on September 25, 2023.

Examining Committee:

Supervisor:
(Member)

Dr. Muhammad Iqbal Hossain
Associate Professor
Department of Computer Science and Engineering
Brac University

Co-Supervisor 1:
(Member)



Dr. Jannatun Mukta
Assistant Professor
Department of Computer Science and Engineering
Brac University

Co-Supervisor 2:
(Member)

Rafeed Rahman
Lecturer
Department of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

Dr. Md. Golam Rabiul Alam
Associate Professor
Department
Brac University

Head of Department:
(Chair)

Sadia Hamid Kazi, PhD
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Abstract

It's quite difficult to fathom that the future of medicine and diagnosis is more dependent on, and much more likely to be dictated by the growth of technology than the quality of doctors. One of the deadliest diseases today that is ubiquitous all around the world is Leukemia. As deadly as it is, it is one of the most difficult diseases to diagnose. One of the biggest challenges is to identify cells as being affected by this condition and this requires highly trained medical professionals to accomplish such tasks. In this paper we have trained four different image processing models to recognize and identify such cancerous cells. We have used more than 9000 images to do so. After the training processes were over, we evaluated the success of these individual models to assess the difference in their final accuracies, we should bear in mind that these images are rather different than what a usual image-based dataset would look like in that the images are quite similar despite being of different classes. We have used the following models: YOLOv5 (precision = 0.82), CNN (precision = 0.74), YOLOv7 (precision = 0.52), EfficientNet (accuracy = 0.89). From this we can clearly agree upon the dominance of EfficientNet over all the other models.

keywords: medicine, diagnosis, leukemia, YOLOv5, YOLOv7, CNN and EfficientNet

Dedication

We dedicate our work to our respected parents and fellow peers.

Acknowledgement

Firstly, all praise be to the Great Allah for whom our thesis has been completed without any major interruption.
Secondly, to our supervisor Dr. Muhammad Iqbal Hossain sir along with our co-supervisors Rafeed Rahman and Dr. Jannatun Mukta for their kind support and advice in our work. They helped us whenever it was required by us.
And finally to our parents without their continuous support, it may not be possible. With their kind support and prayer, we are now on the verge of our graduation.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Leukemia is a cancer that can be found in blood-forming tissues. It includes the bone marrow and the lymphatic system. It involves the white blood cells (WBC). White blood cells are essential to our immune systems and they normally function by growing and dividing in a regulated manner. Leukemia is a complicated and even lethal form of cancer that affects the bone marrow and blood. Our body face difficulties to fight against infections when it faces unusual production of white blood cells. As a result, it fails to perform the necessary tasks. Effective leukemia therapy and better patient outcomes depend on the early and accurate diagnosis of the disease. Researchers are showing their interest in using artificial intelligence that can recognize and categorize leukemia.

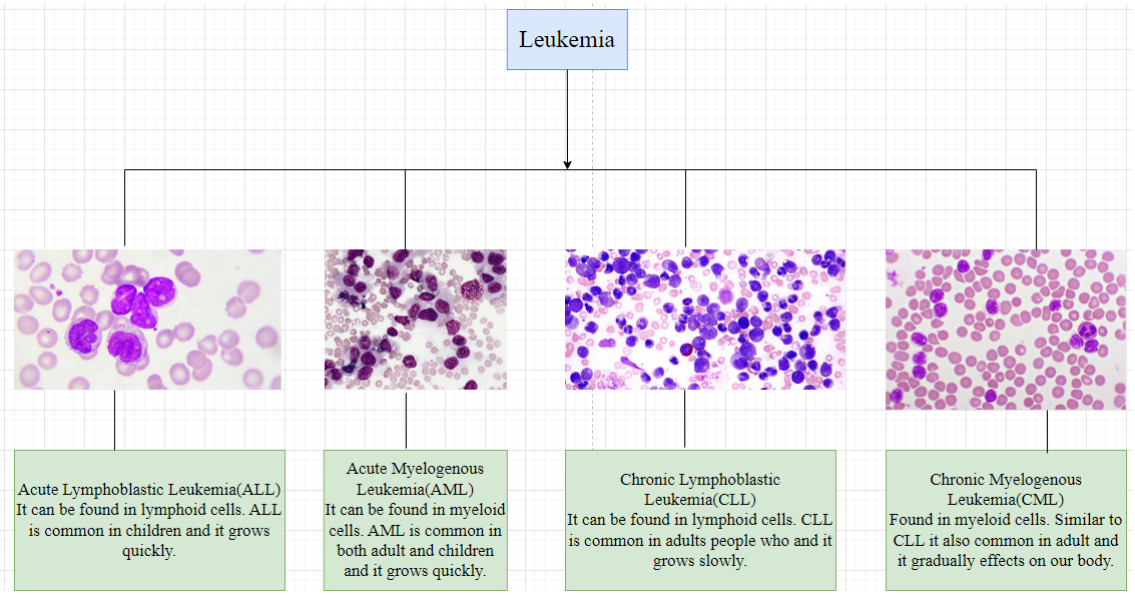


Figure 1.1: Types of Leukemia

However, Leukemia can be classified into four main types. The distribution of these subtypes varies across different populations and age groups. Leukemia can affect individuals of all ages, but certain subtypes have distinct age patterns. For instance, ALL is most commonly diagnosed in children and young adults, while AML and CLL are more prevalent in older adults. According to research, Leukaemia was responsible for about 2.5% and 3.1% of all new cancer cases and deaths worldwide in 2020, respectively [16]. Medical professionals manually investigate these images. It's a very difficult and time-consuming process additionally this process is vulnerable to subjectivity and human error. The development of automated systems can help in the detection and classification of leukemia, resulting in more effective and reliable diagnostics.

1.2 Problem Statement

The problem of "Comparative Analysis of Neural Network Models for Peripheral Blood Cell Image Classification" aims to address the challenges in accurately identifying and classifying leukemia from microscopic images of peripheral blood cells. Leukemia, a type of blood cancer, can manifest in various subtypes with distinct morphological characteristics, making it essential to develop an automated and efficient system that can reliably detect and classify abnormal cells with high precision and sensitivity. The utilization of neural networks presents a promising approach to handling the complexity and variability of blood cell images, but the key challenges lie in training and optimizing these models to achieve robust and accurate results. This research seeks to overcome these obstacles and contribute to the development of a reliable, automated diagnostic tool to assist healthcare professionals in early detection and improved management of leukemia.

1.3 Research Objectives

The primary aim of this research is to investigate the classification and detection of leukemia in peripheral blood cell images using neural networks. The objectives of this research are designed to systematically address key aspects of the research topic and contribute to the advancement of knowledge in this field. The objectives of our research are as follows:

- To develop a dataset of peripheral blood cell images including both healthy and leukemia-affected cells, with proper annotations.
- To implement a deep learning model that can explicitly detect and distinguish leukemia from blood cell images.
- To compare the performance of our model with other methods for detection and classification purposes.
- To determine how well the model can perform on unseen data additionally we will evaluate its performance on external datasets and consider its potential in real-world applications.

Chapter 2

Literature Review

Researchers have developed advanced computer algorithms and neural networks to automate and improve the classification of different types of blood cells but there are still few scopes for further advancement in this field. One of the main challenges is that leukemia is a very diverse disease and there is no single test that can accurately determine all types of leukemia. Additionally, the early stages of leukemia can be difficult to detect which can lead to delays in diagnosis and treatment. Despite these challenges, existing research has introduced various approaches to deep learning for instance CNN, RNN, and BCNN which have demonstrated promising results in accurately diagnosing leukemia and categorizing different subtypes. This approach can minimize the workload of doctors as well as has the potential to improve the accuracy of leukemia diagnosis, especially in the early stages of the disease.

2.1 Related Works

This article [4] describes a system that uses advanced computer algorithms to accurately identify different types of blood cells. Traditionally, identifying blood cells requires expertise, but this system aims to automate and improve the process. The researchers collected a large dataset of 17,092 images of normal blood cells and used them to train a computer model. They tested two different designs. Using VGG-16 and InceptionV3 CNN architectures they achieved high levels of accuracy. Using VGG16 they achieved 86% and achieved 90% accuracy using Inceptionv3. The initial architecture VGG16 was used for feature extraction later these features were used for cell classification by another algorithm. The second design fine-tuned the networks themselves for cell classification. The results showed that both designs performed well, with the second design achieving an overall accuracy of 96.2%. This research demonstrates the effectiveness of using advanced algorithms and neural networks to automate the classification of blood cells.

This research [19] paper focuses on using a special type of deep learning model called Faster R-CNN to diagnose leukemia. The author trained the model using a dataset of microscopic images of blood smears containing both normal and leukemia-affected cells. The model learns to detect and classify leukemia cells by looking at specific regions in the images. The paper explains how the model was trained and evaluated, with bounding boxes drawn around leukemia cells to help the model locate them. The model's accuracy was measured using various metrics and found to be 97%. The experiments showed that the Faster R-CNN model is highly accurate in diagnosing leukemia by effectively identifying and categorizing leukemia cells in the images. This suggests that the model has the potential as an automated tool for leukemia diagnosis.

The research [22] presents a novel approach using Bayesian Convolutional Neural Networks (BCNNs) for the diagnosis of blood cancer. The authors address the challenge of accurately identifying different types of blood cancer through automated systems. The study introduces BCNN models, which integrate the powerful deep learning capabilities of Convolutional Neural Networks (CNNs) with Bayesian inference techniques. This combination enables the models to not only provide accurate predictions but also estimate uncertainty in their predictions. The authors utilize a dataset containing microscopic blood cell images, including both normal and cancer-affected cells, for training and evaluation. They describe the architecture and training process of their BCNN models, including the use of Bayesian techniques to model uncertainty and capture heterogeneity in the data. The performance of the models is evaluated using various metrics such as accuracy, precision, recall, and F1 score. The results demonstrate that the BCNN models achieve high diagnostic accuracy and outperform traditional CNNs. Moreover, the uncertainty estimates provided by the models contribute to the interpretability and reliability of their predictions. By combining deep learning with Bayesian inference, these models provide accurate predictions and valuable uncertainty estimates.

This research [5] focuses on using special computer algorithms called Convolutional Neural Networks (CNNs) to identify different types of leukemia from microscopic images. Accurately classifying leukemia subtypes is important for proper treatment and patient care. The researchers trained a CNN model using a dataset of microscopic images representing various leukemia subtypes, including ALL, AML, CLL, and CML. The paper explains how the model was built and trained, with the dataset divided into training and testing sets. The model learned to recognize the unique features of each leukemia subtype. They evaluated the model's performance using different measurements. In their experiments, the CNN model achieved 88.25% accuracy for one leukemia type and 81.25% accuracy for the other subtypes. These results show that the CNN model is effective in identifying leukemia subtypes. It performs well and can reliably distinguish between ALL, AML, CLL, and CML. These findings demonstrate the potential of using advanced algorithms like CNNs to automate the identification of leukemia subtypes from microscopic images.

In the research work [3], the authors suggest a new convolutional neural network (CNN) design named LeukemiaNet to categorize distinct forms of leukemia, which consist of acute lymphoblastic leukemia (ALL) and acute myeloid leukemia (AML), as well as individuals without cancer. The authors used a dataset of 1188 blood cell images, which were divided into three classes: ALL, AML, and normal. The LeukemiaNet architecture consists of five convolutional layers, followed by two fully connected layers. The convolutional layers use ReLU activation functions, and the fully connected layers use sigmoid activation functions. The LeukemiaNet architecture was trained using a stochastic gradient descent optimizer with a learning rate of 0.0001. The model was trained for 20 epochs, and the best model was selected based on its performance on the validation set. The LeukemiaNet architecture achieved an accuracy of 96.6% on the test set. This accuracy is comparable to the accuracy of other methods that have been used to classify leukemia from blood cell images. The authors also evaluated the LeukemiaNet architecture on a smaller dataset of 49 ALL images and 10 AML images. The model achieved an accuracy of 97.7% on this dataset. The authors stated that the LeukemiaNet architecture is a promising approach for classifying leukemia from blood cell images. The model is accurate and efficient, and it can be used to improve the diagnosis of leukemia. In addition to that, the author also introduced the use of PCA color augmentation to improve the performance of the LeukemiaNet architecture.

In this study [28], Leukemia is the most common type of blood cancer, and it is triggered by an unusually high amount of white blood cells, sometimes referred to as WBCs. It is a major reason for death worldwide and accounts for approximately a quarter of all instances of cancer detected in kids. Acute lymphoblastic leukemia, often known as ALL, is the kind of leukemia that is most commonly discovered in human bone marrow. It is a disease that attacks the bone marrow and ultimately results in the death of white blood cells. The earlier and more accurately cancer is diagnosed, the better the prognosis and the chances of the patient surviving the disease. As a direct consequence of this, medical professionals are now in a position to efficiently diagnose early stages of leukemia by using computer-aided diagnostic (CAD) models. In this research, the authors created a categorization framework using the EfficientNet-B3 CNN model to distinguish ALL as a smart model that adapts the learning rate (LR) automatically. This model differentiates ALL as an automated model. At the beginning of each epoch, they compared the training accuracy with the loss value using a bespoke LR that they developed. On the C-NMC Leukemia dataset, the suggested model was put through its paces. The dataset underwent preprocessing that included both normalization and balancing. The suggested model was assessed and judged in light of more recent classifiers. The suggested model achieved averages of 98.29% precision, 97.83% recall, 97.82% specificity, 98.31% accuracy, and 98.05% Disc similarity coefficient (DSC), respectively. In addition, the model that was presented was used in order to detect the malaria parasite by analyzing microscopic pictures of blood samples. Their suggested model had an average precision of 97.69%, recall of 97.68%, specificity of 97.67%, accuracy of 97.68%, and DSC of 97.68%, respectively. As a result of this, the assessment of the model that was suggested shown that it has an unparalleled perceptive outcome with tuning in comparison to other current existing models.

The following article [23] highlights, White Blood Cell (WBC) identification and counting on microscopic blood cell pictures may enhance human knowledge in Acute Lymphoblastic Leukemia (ALL) diagnosis, making it easier and faster to diagnose the disease. Previous studies often used traditional methods for ALL subtype identification. These methods need many phases, including WBC segmentation, touch cell separation, feature extraction, and classification. The authors offer strategies for object identification and instance segmentation that just need one learning framework and do not require different phases of development. In this research, they assess how well the YOLO and Mask R-CNN models perform in identifying ALL subtypes. They use evaluation measures like accuracy, recall, F1, and mAP to gauge their effectiveness. Based on the results, the YOLOv4 shows superior performance compared to the YOLOv5 and the Mask R-CNN when it comes to identifying ALL subtypes. The YOLOv4 model performs slightly better than both the YOLOv5 model and the Mask R-CNN model, with an F1 score of 89.5% and a mAP score of 93.2%.

In this paper [29], The analysis of blood diseases starts with the recognition and grouping of bone marrow (BM) cells as a crucial foundational element. As a result of the low precision caused by the limited amount of BM-cell data samples, the slight variances between categories, and the compact nature of the objective, pathologists are still required to manually carry out numerous identifications on a daily basis. In this study, the authors present an enhanced technique for identifying BM-cells, which we refer to as YOLOv7-CTA. This is carried out to conquer the challenges that were previously talked about. To begin with, to enhance the model's ability to detect small details, they integrated a new component called CoTLAN into the main network. This provided the model with the capacity to conduct long-term modeling among the different parts of the desired feature data. Afterwards, to work together with the CoTLAN module and give more focus to the characteristics in the targeted region, they combine the coordinate attention (CoordAtt) module within the CoTLAN modules to amplify the model's emphasis on small target features. This is performed to enable the model to focus more on the characteristics in the region that needs to be identified. Ultimately, they employ K-means++ to group the desired boxes of the BM cell dataset in order to produce more suitable anchor boxes. This facilitates the quicker convergence of the enhanced model. Furthermore, they utilize the Focal loss technique instead of the multi-class cross entropy to rectify the inequality between the number of positive and negative samples in the BM-cell images. The findings of the tests indicate that the suggested model outperforms the Faster R-CNN model, the YOLOv5l model, and the YOLOv7 model in terms of mean average precision (mAP). This indicates an enhancement of 12.9%, 8.3%, and 6.7%, correspondingly, above those versions. This shows that the YOLOv7-CTA model is more effective and accomplished in identifying BM-cells.

White blood cells, also known as WBCs, are an important factor in determining a person's overall health as well as the presence or absence of illness in their body. An automated categorization system [14] is able to recognize white blood cells, which may assist medical professionals in correctly diagnosing a variety of conditions, including malaria, anemia, leukemia, and others. Automated examination of blood cells enables quick and reliable results and often includes a comprehensive data set

with no need for performance negotiation. The process takes a lot of time, but the latest computer systems can analyze blood cells automatically using images of blood smears. This is achieved by going through several phases, like extracting characteristics, dividing into sections, and preparing beforehand. This study suggests a successful method for identifying and categorizing images of blood cells in the outer parts of the body. It accomplishes this by combining the algorithms of salp swarm and cat swarm optimization to optimize the convolutional neural networks used in the SSPSO-CNN approach. The goal of this method is to circumvent the issues that have been raised. This research study makes use of the CNN method to automatically categorize five different types of peripheral blood cells, including eosinophils, basophils, lymphocytes, monocytes, and neutrophils. There is no interaction from a human researcher. Another aim of this research [14], is to introduce an improved edition of salp swarm optimizer (SSO) by utilizing particle swarm optimization (PSO) with the objective of attaining satisfactory classification results on the collection of blood cell images. For the purposes of training, the CNN makes use of the VGG19 architecture in this work. When using VGG19 models, an accuracy of 98% may be attained in the classification process. A high level of classification accuracy is achieved using the suggested model, which is based on the CNN technique and optimized using SSPSO. This model also allows automated peripheral blood cell categorization. This technique lays the groundwork for the fine-tuning procedure that is required to construct a classifier that was trained on 10,674 photos gathered from medical practice. The suggested technique improved performance in terms of high precision and F 1-score, achieving an overall classification accuracy of 99% in the process [14].

The examination of white blood cells [20], which serve as the body's first line of defense against pathogenic microorganisms and other agents that might cause illness, is a significant topic in the field of medicine. The quantities of these cells in the blood, which may be broken down into five different groups, namely monocytes, eosinophils, basophils, lymphocytes, and neutrophils, change depending on the kinds of illnesses that are present in the body. The peripheral blood smear is one of the most common methods used for analyzing blood cells. Evaluation using this approach manually is a hard and time-consuming process. At the same time, the performance of the procedure is affected by a wide variety of environmental and humanistic aspects. As a result, a detection procedure that operates in real time has been implemented in this research. To begin, the object recognition framework makes use of the YOLOv5s, YOLOv5x, and Detectron 2 R50-FPN pretrained models. Following this, two novel additions are made to the investigation in order to enhance the functionality of the model. The initial contribution consists of maximizing the activation function, which is both an important criteria in the process of training the model and an arrangement that is supplied in the architecture. The overall recognition rates of all classes see a 0.006 percentage point increase as a result of using this strategy, which is advocated. The combined use of the YOLO and Detectron2 frameworks, both of which are distinct in the object evaluation procedures that they employ, is the second contribution. The outputs acquired from the YOLO and Detectron2 pretrained models were improved by between 3.44% and 14.7% when compared to the success rate attained with this hybrid structure, which offered an improvement of between 3.44% and 14.7%. In addition, the highest accuracy rate

of this hybrid structure on the test dataset for the detection and classification of white blood cells is reached as 98%. This figure was acquired by using the hybrid structure to identify and classify the white blood cells.

Using machine learning techniques in medicine is challenging [12] because it requires a high level of precision and accuracy. Additionally, the risks involved with even the slightest errors are significant. Using these strategies for a more complex area of blood analysis shows great potential, specifically for automatically recognizing different types of blood cells. This could be helpful in detecting blood-related diseases. In this research, they assess 27 of the most well-liked advanced convolutional neural network designs on the small images of blood cells obtained as data. The assortment is available to the general public and includes a large amount of typical blood cells obtained using the CellaVision DM96 machine and identified by qualified doctors into a total of eight different cell categories. They make significant changes to the latest advanced picture sorting models. Blood cell classification carried out using pre-training on the ImageNet dataset. During the training phase, they utilize methods of data augmentation to avoid overfitting and attain generalization. A group of highly skilled models brings significant improvements compared to previous research, achieving remarkable results in the field with a classification accuracy of 99.51 percent. This document provides practical foundations and standards for various common deep-learning structures based on real-world data. for the tiny task of recognizing blood cells from the outer edge.

Chapter 3

Methodology

We divided a methodology into two sole parts mainly workflow and dataset description.Then, further discuss about the model architectures.

3.1 Workflow

In our Research, by collecting proper dataset of relevant image data for our machine learning model we began our workings. We used Roboflow to annotate our data to facilitate our model to be trained, for objects of interest creates ground truth labels. We also pre processed our data with plethora of techniques such as "Min-Max Scaling", resizing and scaling to prepare our data to feed into the models. Just after which divide our data into three portions which are training, testing and validating to evaluate our models. Multiple model architecture was explored which are YOLOv5, YOLOv7, Basic CNN and EfficientNet. Anchor boxes were configured by us and we trained them on pre-processed data with YOLO models. These models were trained on the data which were pre- processed earlier make a comparison on model accuracy and efficiency.

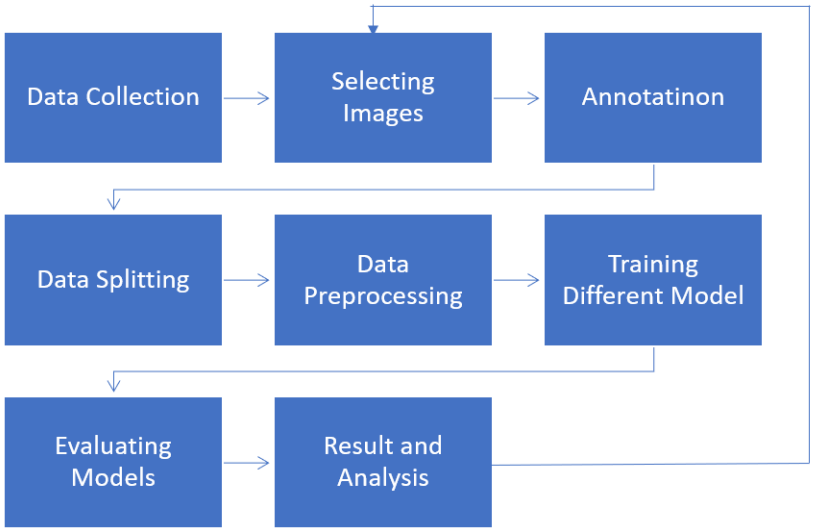


Figure 3.1: Overall Workflow

3.2 Dataset Description

The information set provided through the given link is centered on identifying types of blood cancer known as leukemia, which is a notable objective in the area of medical testing. It provides a complete gathering of genetic data on how genes are used in patients with different forms of blood cancer. The information provided in this dataset is intended to support scientists, analysts, and healthcare experts in creating dependable and effective models to identify and predict the future of leukemia patients. The information provided includes patterns of how genes express themselves, which were garnered from a small-scale array examination. It comprises gauging of gene manifestation degrees for countless genes in every specimen. Every specimen matches a person who has been identified with a specific form of blood cancer. This dataset is used for the task of classifying microscopic images of blood samples into two categories: Acute Lymphoblastic Leukemia (ALL) and Hematopoietic(hem) cells. In the context of this dataset, the ALL category represents images of blood cells affected by Acute Lymphoblastic Leukemia. On the other hand, the hem category represents images of normal hematopoietic cells, which are not affected by Leukemia.The information is given in an organized manner and usually comes as a file that uses commas to separate values, known as a CSV file. Every line corresponds to a specimen from a patient, where the genes' manifested traits are displayed as vertical divisions. Moreover, the set of information might possess other significant details about the specimens, like patient characteristics and medical records. The collection of information covers a considerable quantity of facts, with a notable quantity of examples and hereditary manifestation evaluations. You can find out the precise amount of samples and characteristics by referring to the dataset explanation that's accessible on the Kaggle site. The collection of data is a useful tool to create models of machine learning that can assist in the precise identification of types of leukemia. Using the way genes are expressed, these models may be able to help healthcare providers make knowledgeable choices regarding treatment strategies and individualized attention. Models that have been taught using this collection of data can be utilized to estimate the development and prediction of potential outcomes of disease for patients with leukemia. These patterns can assist in recognizing individuals who may need more intense treatments or those who may react favorably to particular therapies. The collection of data provides scientists and researchers with a useful asset to examine leukemia biology and create new healing methods. It allows for the study of how genes are expressed in various types of leukemia, which could lead to finding new targets for treatment.

Chapter 4

Dataset Pre-processing

4.1 Annotation

We have mainly used Roboflow to preprocess our data. Since we are working in the field of image processing it's essential to label and annotate the images that we are working with and using Roboflow all this can be done with ease. Roboflow uses a bunch of tools such as resizing, cropping, and augmentation to make the images finely tuned to feed to the machine learning model.

From our dataset, the images we received were classified with ALL and Healthy cells. As a result, in order to train our model we annotated it with the help of Roboflow. In order to gain a better accuracy we have to crop the majority of the black portion so that our model can detect it within a short period of time.

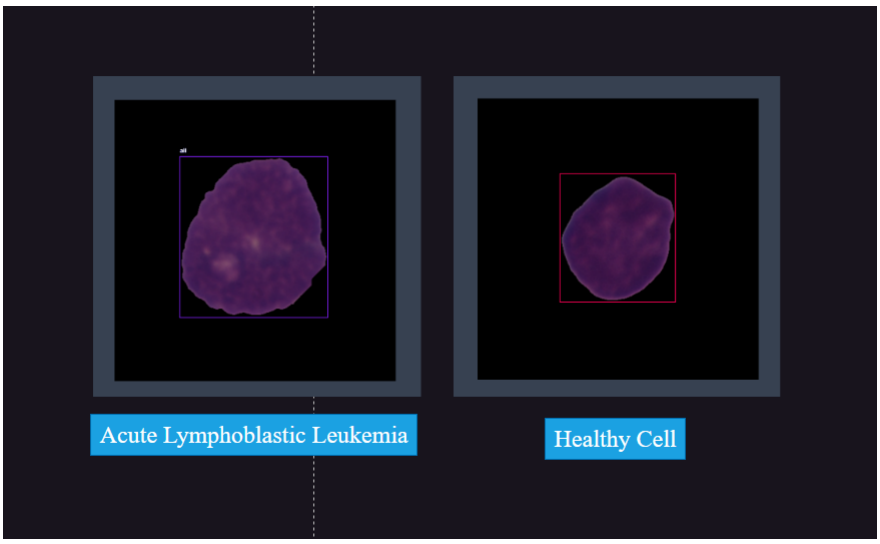


Figure 4.1: Annotation of ALL and Healthy Cell

4.2 Min-Max Scaling

To ensure the prime performance of an individual learning model, proper Data Preprocessing is a very concerned matter. To be more precise, "Min-Max Scaling" technique was used by us to fit out input data for the training and evaluation. This preprocessing step is particularly crucial while working with image data as the problem that arises from different feature scales gets extracted by it . An input component was present for each combination of data points in our dataset which represented the image data and an associated label or target value that served as a target component.

The "Min-Max Scaling" operation [9] was applied on the input data only. In this step, By the highest possible pixel value the value of each pixel of input images gets divided. Which is 255 for both grayscale image and each color channel of an RGB image which is 8-bit. This division ensures the conversion to a uniform pixel value range[0,1] ,which initially was 0 for Black and 255 for White. As for feature scaling and normalization pixel values to a particular scale "Min-Max Scaling" is known but it also additionally serves more essentiality which are as follows:

4.2.1 Stability and Convergence

Model stability got supported while also training an increase in rapid convergence were seen by consistent scales of input features, which makes the learning more enhanced [10].

4.2.2 Model Robustness

Making it less sensitive to changes in the input data, for applying in fresh scenarios , the capability of the model enhances to a different level.

4.2.3 Gradient Descent

Prevention of certain features from predominating the learning process gets enhanced by having characteristics with identical scales, for optimization algorithms like gradient descent[10].
Min-Max Scaling was an integral part of our data preprocessing pipeline which concurrently impacted our machine learning models overall success and performance by also handling image data.

4.3 Data Splitting

It is important to partition our dataset into distinct subsets to train, validate and evaluate our pre-selected models for our research. Partitioning the models ensures machine learning models to be trained on a solid depiction of the data by enabling us to evaluate the performance of models carefully on hypothetical examples.
For training 70%, validation 20% and testing 10% data were designated. For learning patterns the training set was used for machine learning models. Furthermore, the validation set not only helped in running hyper parameters but also prevented the model from overfitting. Finally, for assessing the model's generalization performance a 10% test set was employed.
The data splitting was done to ensure the training should be done with an appropriate portion of data. Due to this precise partitioning, we can get to a conclusion about the model's efficacy and adaptability .

Chapter 5

Proposed Models

5.1 CNN

In this research, we used a widely used powerful deep learning model which is Convolutional Neural Network(CNN) architecture which is mainly designated for image classification tasks [13]. This model can automatically learn and extract some distinct features from input images.

Initially the model starts with an input layer which is made to support 256x256 pixel resolution and RGB color channels. These individual channels record color information, provide source images with a rich representation.

The heart of this architecture is three consecutive convolutional layers which contribute to the feature extraction [2]. With the help of 16 filters and (3,3) sized kernels, the first convolutional layer filters detect local patterns and edges from the input images. To introduce non-linearity and enhance feature representation activation function named The Rectified Linear Unit (ReLU) gets applied [15]. Following the first layer a max-pooling layer is introduced for the first time. It reduces the spatial dimension of the feature map that comes from the first convolutional layer. Which also works on preserving important features with critical information and additionally working on reducing the time complexity.

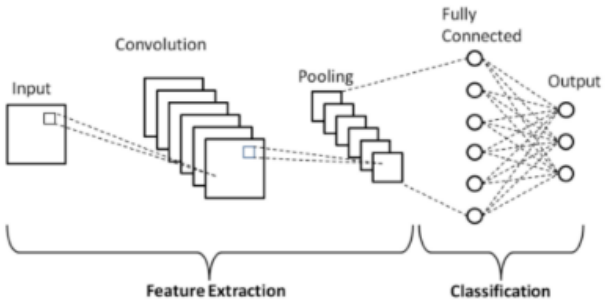


Figure 5.1: The Basic CNN Architecture

Using ReLU activation function with addition to 32 filters with a (3,3) kernel size second convolutional layers involved in the model[11]. Additionally, captures the most complex and abstract features from the input data. Second convolutional layer after that followed by another max-pooling layer which makes the feature map more precise. To capture high-level features, the third convolutional layer comes up with 16 filters with (3,3) kernels to make an impact in the model's ability. Lastly to the output of the third convolutional layer, a final max-pooling layer is applied [15].

After moving through the convolutional and max-pooling layers the feature map then flattened into a one-dimensional vector which is a step to move to fully connected layers from convolutional layers. This operation is known as flattening. To learn high-level representations from the flattened features that got from earlier steps, introduction of two fully connected (dense) layers were done [15]. A compound consisting of 256 units and employing ReLU activation function known as the first dense layer, serves as a feature extractor, determining complex relationships and patterns in the data. With just one unit , in the final dense layer the function of sigmoid activation is used. The output layer is designed for a binary classification task which offers a probability score between 0 and 1. To generate binary predictions, this probability can be used as a threshold. Which can categorize input images into one of two groups.

In a nutshell, starting with low-level patterns and progressing to high-level abstractions ,the design of our CNN architecture allows for the gradual extraction of arranged features from these input images [11]. Then from fully connected layers classification was done by using those extracted features. To ensure that the model can work with new data prediction in an efficient manner appropriate preprocessing techniques, such as data splitting and scaling, were applied.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 254, 254, 16)      448

 max_pooling2d (MaxPooling2D  (None, 127, 127, 16)     0
 )

 conv2d_1 (Conv2D)           (None, 125, 125, 32)      4640

 max_pooling2d_1 (MaxPooling  (None, 62, 62, 32)       0
 2D)

 conv2d_2 (Conv2D)           (None, 60, 60, 16)        4624

 max_pooling2d_2 (MaxPooling  (None, 30, 30, 16)       0
 2D)

 flatten (Flatten)           (None, 14400)             0

 dense (Dense)               (None, 256)               3686656

 dense_1 (Dense)             (None, 1)                 257

=================================================================
Total params: 3,696,625
Trainable params: 3,696,625
Non-trainable params: 0
_________________________________________________________________
```

Figure 5.2: CNN Model Description

5.1.1 CNN Implementation Flow

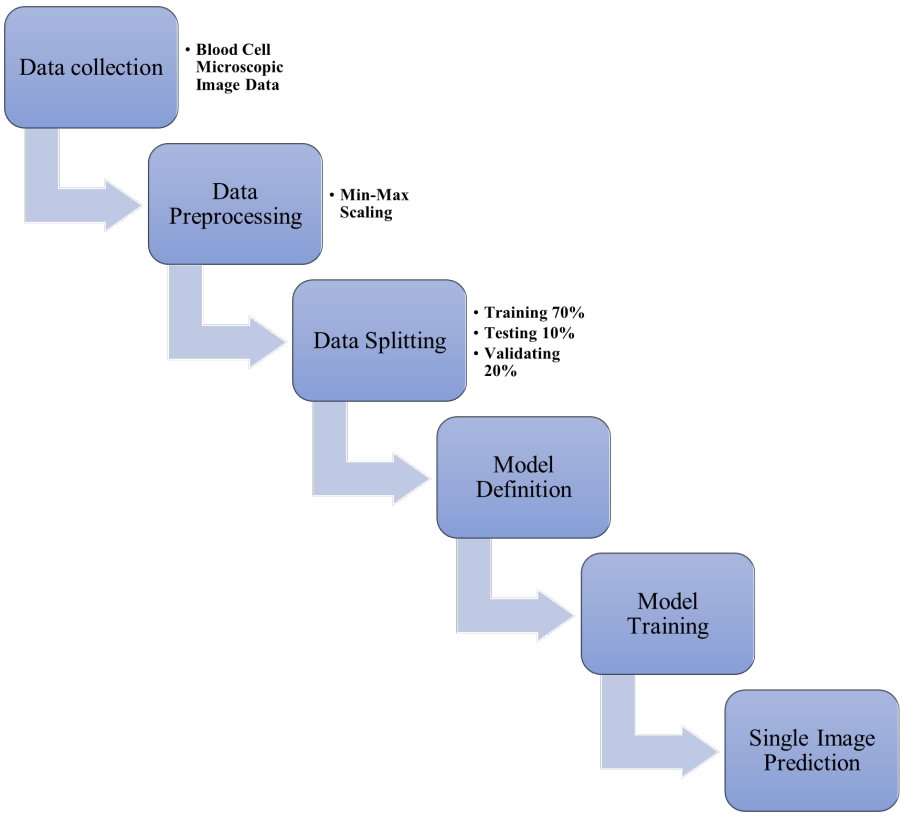


Figure 5.3: Implementation Flow

5.1.2 Training And Validation of CNN

5.1.3 CNN Training

During the time of training , the CNN model was exposed to a sizable collection of images which was annotated, where it collects patterns and characteristics of images. Enhancing accuracy so that images can be correctly classified based on their attributes is the sole purpose here. To make predictions, it needed to make sure that CNN must be trained so that it can differentiate the key components like edges, textures, forms, and colors from data.

5.1.4 Testing

During the testing the model gets tested by some unseen or unknown images that weren't seen by the model previously to see how precisely the model can predict the class. In time of testing the predicted class was compared with the true class to calculate the accuracy of the model.

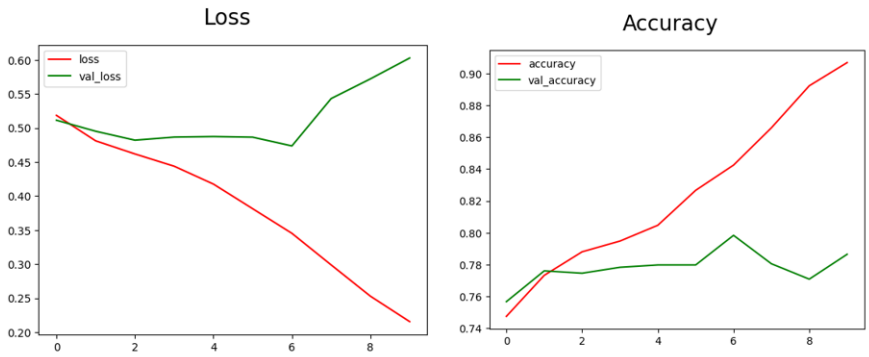


Figure 5.4: Loss and Accuracy of CNN

5.1.5 Result of CNN

```
Epoch 1/10
148/148 [==============================] - 202s 1s/step - loss: 0.5185 - accuracy: 0.7475 - val_loss: 0.5113 - val_accuracy: 0.7567
Epoch 2/10
148/148 [==============================] - 189s 1s/step - loss: 0.4810 - accuracy: 0.7732 - val_loss: 0.4951 - val_accuracy: 0.7760
Epoch 3/10
148/148 [==============================] - 189s 1s/step - loss: 0.4618 - accuracy: 0.7880 - val_loss: 0.4820 - val_accuracy: 0.7746
Epoch 4/10
148/148 [==============================] - 197s 1s/step - loss: 0.4438 - accuracy: 0.7948 - val_loss: 0.4866 - val_accuracy: 0.7783
Epoch 5/10
148/148 [==============================] - 190s 1s/step - loss: 0.4176 - accuracy: 0.8047 - val_loss: 0.4874 - val_accuracy: 0.7798
Epoch 6/10
148/148 [==============================] - 191s 1s/step - loss: 0.3814 - accuracy: 0.8266 - val_loss: 0.4864 - val_accuracy: 0.7798
Epoch 7/10
148/148 [==============================] - 199s 1s/step - loss: 0.3453 - accuracy: 0.8425 - val_loss: 0.4736 - val_accuracy: 0.7984
Epoch 8/10
148/148 [==============================] - 190s 1s/step - loss: 0.2989 - accuracy: 0.8659 - val_loss: 0.5431 - val_accuracy: 0.7805
Epoch 9/10
148/148 [==============================] - 194s 1s/step - loss: 0.2530 - accuracy: 0.8923 - val_loss: 0.5720 - val_accuracy: 0.7708
Epoch 10/10
148/148 [==============================] - 200s 1s/step - loss: 0.2156 - accuracy: 0.9069 - val_loss: 0.6026 - val_accuracy: 0.7865
```

Figure 5.5: Epoch Wise Result of CNN

5.1.6 Point Prediction From CNN

In our work we have done point prediction where we obtained individual predictions from our machine learning model. This is a type of supervised learning especially in the context of regression and classification tasks.

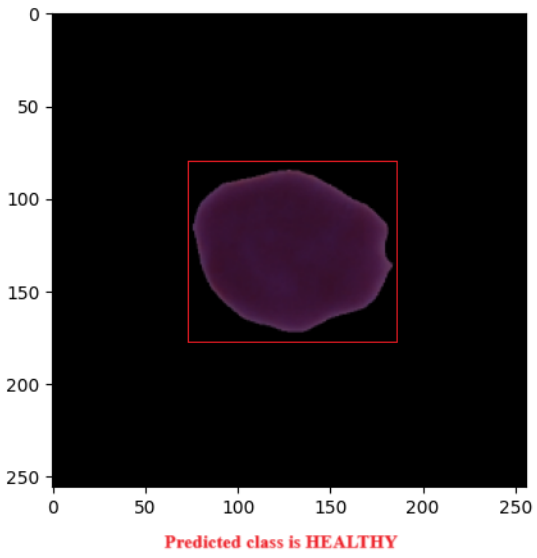


Figure 5.6: Prediction of Class

5.2 YOLOv5

Since YOLOv5 has yet to have a published paper, we are only able to provide a basic explanation of how YOLO functions. Given how similar the YOLOv5 and YOLOv4 models are to one another, a comparison between the two will help to further explain.

As a single regression model, YOLO handles the detection task[18]. It heavily relies on its backbone model, which is responsible for identifying meaningful features from images that are also used in the top-level layers. For example, GoogleNet was used as the backbone for the YOLO model which later on changed to DarkNet networks by Joseph Redmon.

The process was initiated by first dividing the input into an S*S grid cell. Each and every cell is in charge of detecting objects that are right in front of them. Specifically, if the center of an object or something enters the grid cell, that cell's job is to find that specific object.

Bounding boxes, which are rectangular outlines that are used to draw attention to specific objects in images, are provided in multiples for each grid cell. The cell then performs additional calculations to forecast the bounding box's height, width, and center as well as its likelihood to contain an object and the type of object that object will be. Type of object refers to the possibility that a blood cell could be either healthy or leukemic.

Secondly, it eliminates bounding boxes below a predetermined threshold by using the Intersection over Union (IoU) concept. By using this idea, the bounding boxes that are too far apart from the actual boxes can be removed from the model. In the detecting scheme, IoU determines how the bounding boxes overlap.

In order to eliminate overlapping and redundant boxes, the algorithm employs a different technique called Non-Maximum Suppression (NMS), which selects the most efficient boxes. Because there might still be some multiple detection boxes for an object of a similar type after the Intersection over Union.

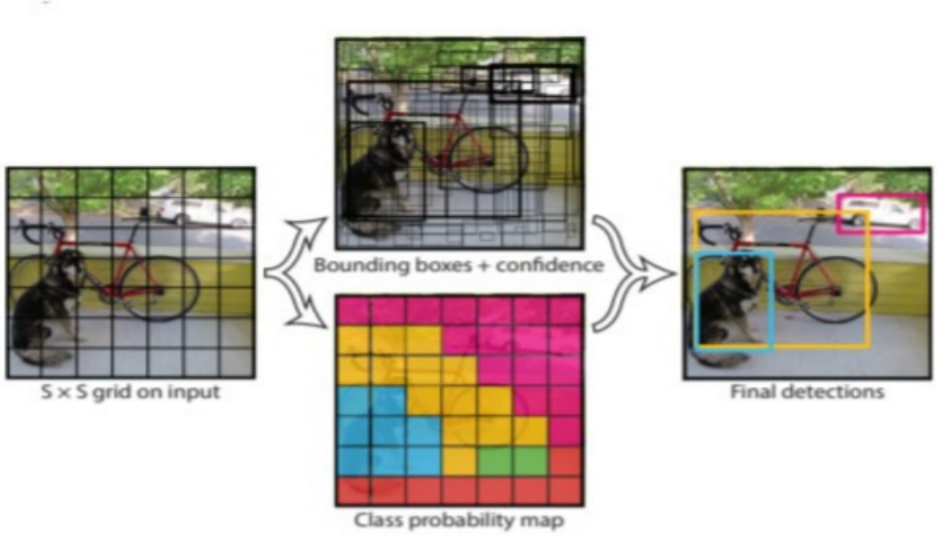


Figure 5.7: YOLO algorithm implementation.

Figure 5.2 taken from [1] demonstrates how the fundamental YOLO algorithm works to predict an object. By anticipating the objects in the bounding boxes, the model was able to correctly identify the type of object in the figure.

The loss function is the most fascinating aspect of YOLO. The model YOLO algorithm optimizes the loss function during training [25].

This is merely a summary of YOLO's fundamentals. Later, YOLO models and versions with new features and enhancements were released. For example YOLOv4, YOLOv5 etc. However, YOLOv5 and YOLOv4 have much more similar architectures than the others. We will go into greater detail about the YOLOv5 differences.

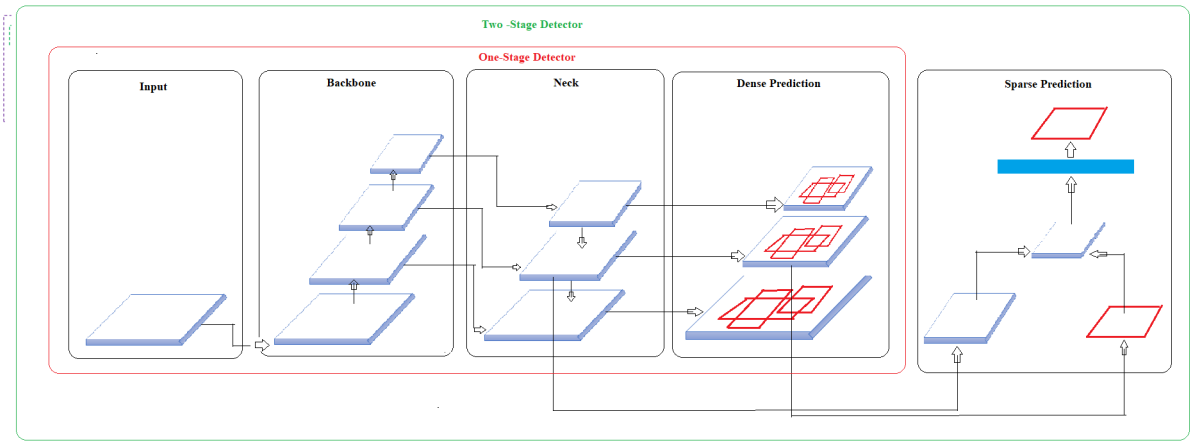


Figure 5.8: Structure of an Object Detector

Figure 5.3 shows the elementary build-up of an object identifier. Backbone was used to predict the classes and bounding boxes for pre-training. One staged or two staged detectors, like YOLO, and SSD, are suitable for dense prediction (which is one staged), while R-CNN is recommended for sparse prediction. A layer called the neck serves as a storage area for feature maps in contemporary object detectors [7].

For YOLOv4 architecture CSPDarknet53 is used as the backbone and an APP block is added to widen the receptive field. Additionally, Persistent Appearance Networks (PAN) were used for parameter aggregation. And just like YOLOv3, YOLOv4 makes use of a similar type of head anchor. It also includes a fresh method of data augmentation Self Adversarial Training (SAT). For instance, employs two forward steps and one backward step and combines four training images. In the first step, the network modifies the image while keeping the weights, and in the second, it is trained to spot an object in the modified image [7].

YOLOv5 has a nearly identical architecture to YOLOv4 [17]. The primary difference, however, is that anchor boxes in YOLOv5 are automatically determined from the distribution of bounding boxes rather than being predefined. This is a significant improvement because the size of the building boxes may vary from the anchors previously specified in the COCO dataset. As opposed to YOLOv4, which made use of the Darknet Framework, YOLOv5 uses the PyTorch Framework. Additionally, there are some differences between these two models in terms of the file types they support. Because YOLOv5 uses a .yaml file and YOLOv4 uses a .cfg file, respectively. The difference between these two types of file conventions is that a .yaml file identifies different types of network layers after which the number of layers present in that block was multiplied.
The architectural workflow of the YOLOv5 algorithm is shown in Fig. 5.4. The three main parameters of the YOLOv5 algorithm are the “Backbone” “Neck” and “Output” [24]. The backbone network is made up of CNN, a version of EfficientNet, which extracts features from the preprocessed inputs. The feature map gathered from the Feature Pyramid Network (FPN) is refined in the Neck. uses both upsampling and concatenation to combine feature maps of various scales. Additionally, it aids in gathering more precise data from various levels of the feature pyramid. Then the detection was completed, and after going through the Non-

Figure 5.9: YOLOv5 implementation flow

Maximum Suppression (NMS) step to remove overlapping and redundant boxes by choosing the most effective one, it provides the most single output that is possible. When compared to earlier iterations, one of YOLOv5's biggest advantages over other YOLO versions is that YOLOV5 is more compact, offers greater precision, and is implemented in the open-source PyTorch framework. There are ten distinct architectures are included in the YOLOv5 model, and they are known as YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x, YOLOv5n6, YOLOv5m6, YOLOv5l6, and YOLOv5x6 + TTA (Jocher, 2020a). However, most of the time only the first five are taken into account for research: small, medium, large, and extra-large [24].

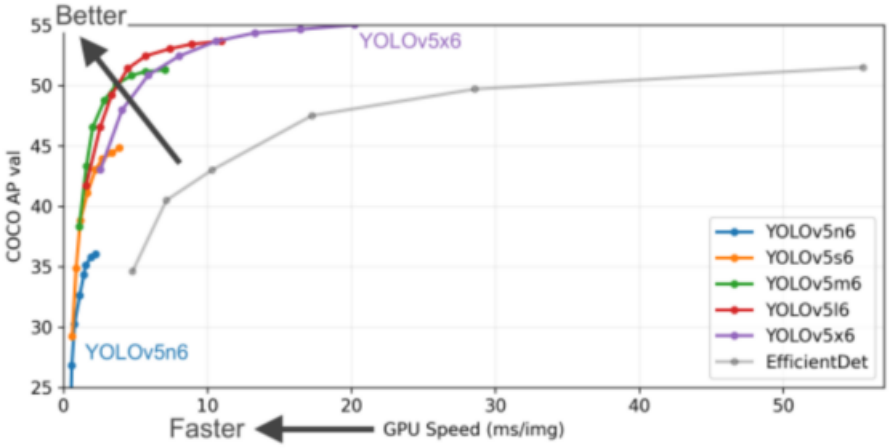


Figure 5.10: Tradeoff between various YOLOv5 versions' speed and accuracy and EfficientNet.

Figure 5.5, which was taken from the GitHub repository of YOLOv5 by Ultralytics, compares the speed and accuracy of various versions of YOLOv5 and EfficientNet. In general, as the curve rises and moves further to the left, it gets faster and more accurate.

5.2.1 Results of YOLOv5

Using a total of 4413 images which amounts to almost 50% of the available data size. The YOLOv5 model was validated by 150 epochs and achieved a moderate level of accuracy. There were 2 classes involved here mainly the “all” and “healthy” and after the data was preprocessed the graphs show some of the results.

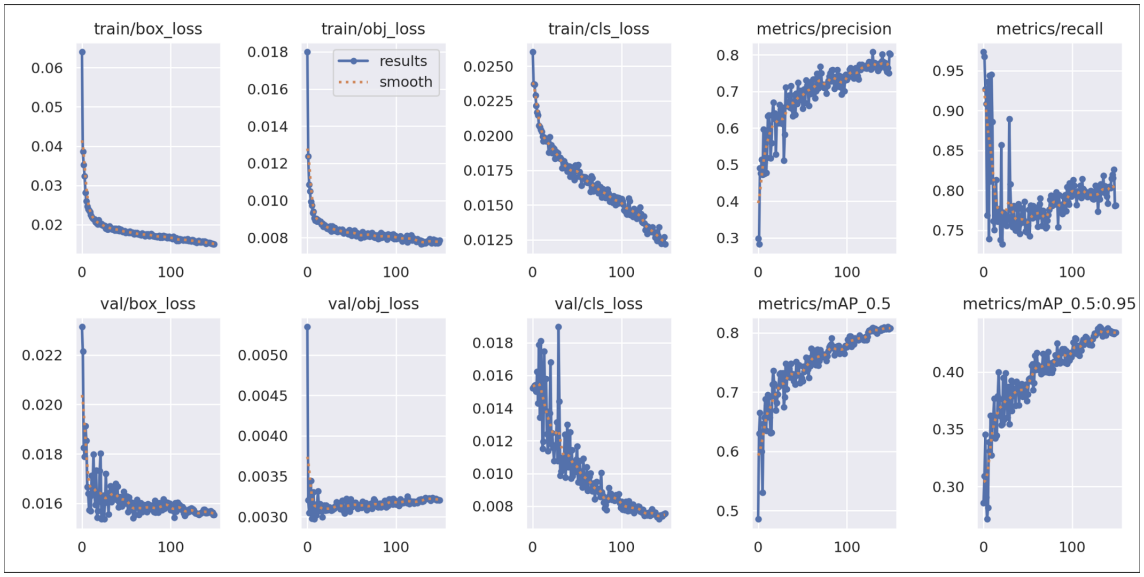


Figure 5.11: Training Data Graphs

5.3 EfficientNet

EfficientNet is a family of convolutional neural network architectures designed to be highly efficient and accurate across a wide range of computer vision tasks. It was introduced by [6] in 2019. The principal idea behind EfficientNet is to achieve maximum performance while minimizing the computational resources required.
EfficientNet's architecture is based on a compound scaling method that systematically balances factors like depth, width, and resolution. These three factors are scaled in a systematic way to create models of different sizes [8].
Convolutional neural networks are scaled up so that we can achieve better accuracies. According to the author of [6], by adding additional layers the ResNet18 model can be expanded into ResNet200. This scaling technique generally enhances accuracy but traditional methods for scaling models tend to be somewhat arbitrary. Few models expanded in terms of depth, others in terms of width and some simply take larger images as input to improve results. Unfortunately, these random scaling approaches require manual adjustment and substantial human effort.
Unlike other convolutional techniques, EfficientNet introduces an effective approach to scaling CNN models to achieve better accuracy and efficiency. It suggests a method known as the "compound coefficient" to scale models. This approach scales all three dimensions: depth, width, and resolution uniformly.

A detailed breakdown of the EfficientNet architecture is given below:
The concept of compound model scaling involves a thorough examination of how different scaling techniques impact a model's performance and efficiency. While scaling individual dimensions can enhance model performance, achieving a balance across all three dimensions, width, depth, and image resolution considering the available computational resources is the key to overall performance improvement.

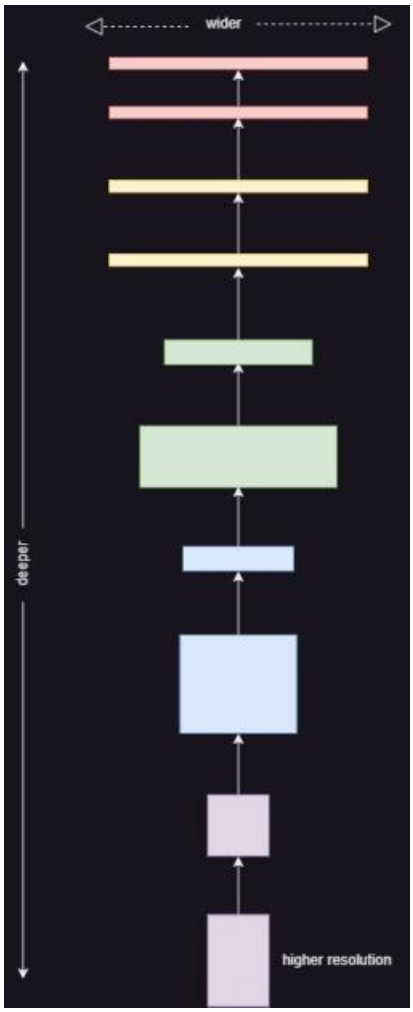


Figure 5.12: Compound Scaling

The compound scaling approach involves maintaining a consistent ratio while scaling the dimensions of width, depth, and resolution to achieve a balanced optimization.

The underlying assumption for these networks is that with larger input images, it's necessary to add more layers to expand the receptive field and increase the number of channels to effectively capture complex patterns present in these larger images [21]. Additionally, the compound scaling technique has proven to be beneficial, enhancing the efficiency and accuracy of preceding CNN models such as MobileNet and ResNet. The compound scaling method outperforms the efficiency and accuracy of earlier CNN models like MobileNet and ResNet, resulting in approximately 1.4% and 0.7% improvement in ImageNet accuracy compared to alternative random scaling techniques.

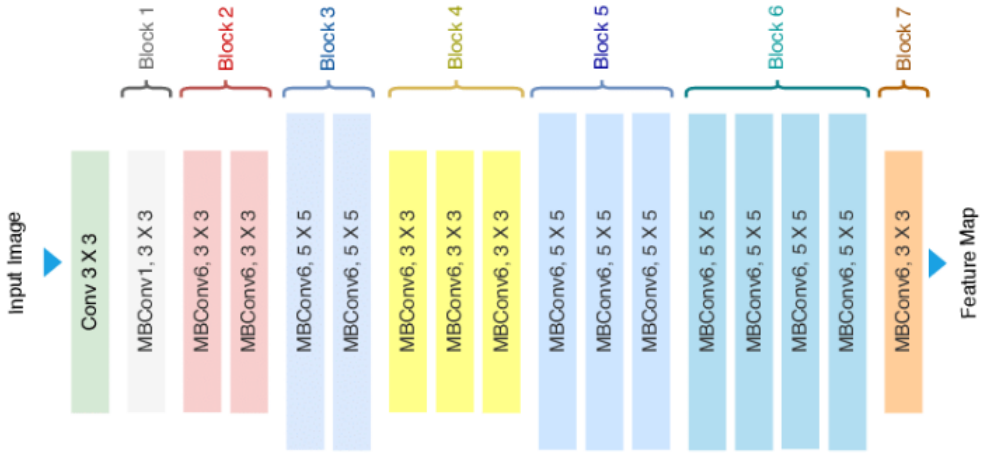


Figure 5.13: EfficientNet Architecture

Compound scaling optimizes accuracy and FLOPS in neural architectures without altering internal layer operations. The EfficientNet-B0, a mobile-size baseline network, employs the mobile inverted bottleneck MBConv as its key building block, featuring the inverted residual block and an additional SE (Squeeze and Excitation) block.
The MBConv aims to enhance information flow in deep neural networks by utilizing an inverted pattern compared to regular residual blocks. The channel progression in a normal residual block is wide → narrow → wide.

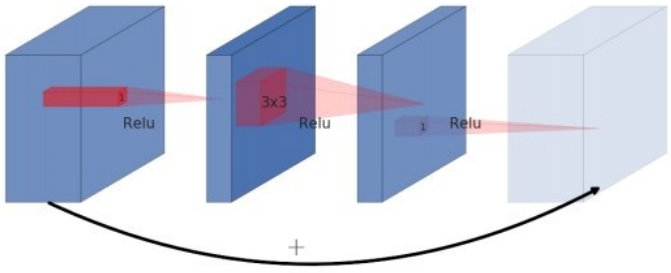


Figure 5.14: Residual Block

On the other hand, an inverted residual block, it's narrow → wide → narrow. This inversion is crucial for efficiency and adaptability in CNNs.

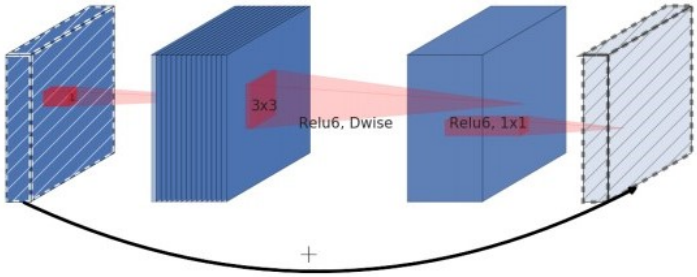


Figure 5.15: Inverted Residual Block

MBConv achieves this through Depth-wise Separable Convolution: widening channels with a 1x1 point-wise convolution, followed by a 3x3 depth-wise convolution to significantly reduce parameters, and finally reducing channels with another 1x1 convolution for addition at the block's start and end.
The SE block, a component for CNNs, dynamically recalibrates feature interdependencies by assigning higher weights to important channels instead of equally weighting all channels.

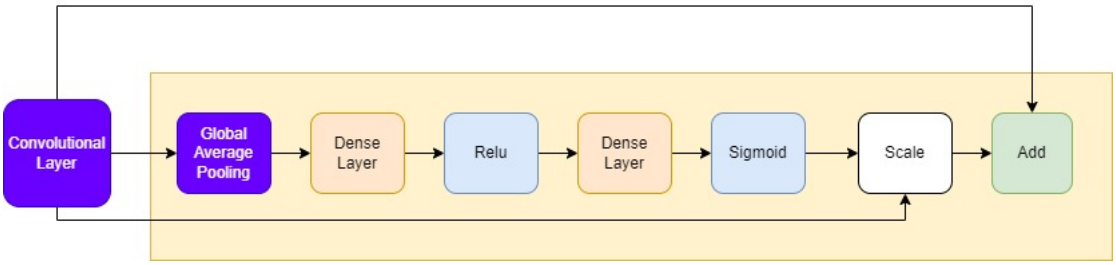


Figure 5.16: Architecture of an SE Block

After that, EfficientNet integrates the SE block with the MBConv block to improve channel-wise interactions, resulting in an efficient and adaptable neural network structure.

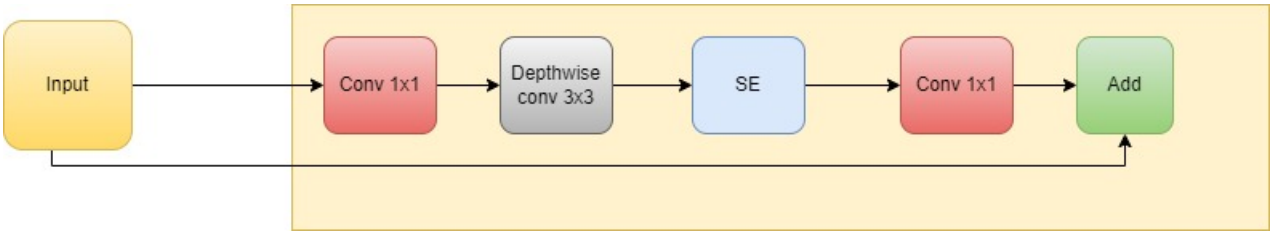


Figure 5.17: Architecture of an MBconv Block

5.3.1 Testing

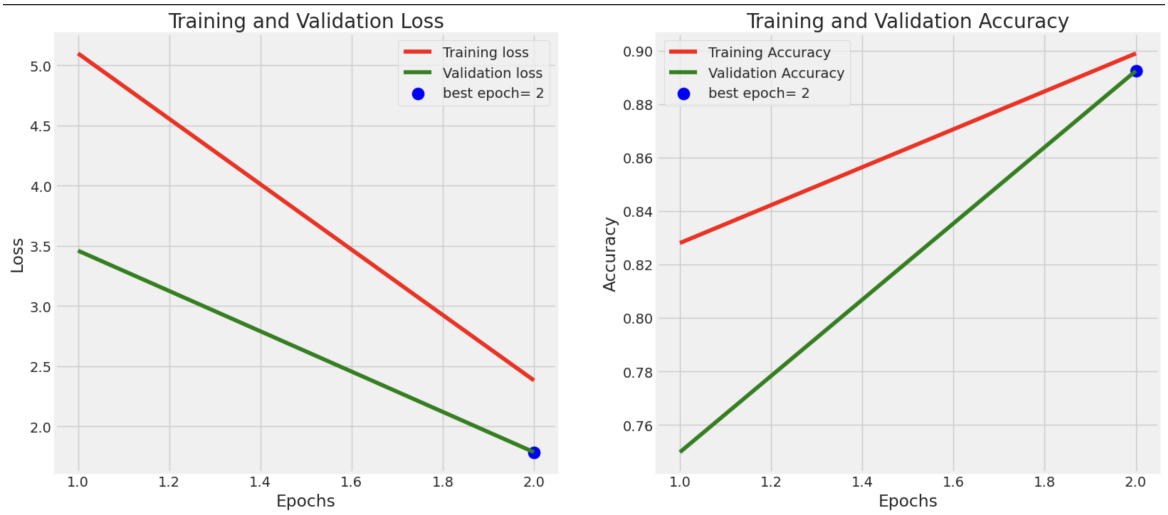


Figure 5.18: Loss and Accuracy Graphs

Here we see a steady decline of the validation loss as the training loss also heads for a steady decline.

5.3.2 Results of EfficientNet

Epoch	Loss	Accuracy	V_loss	V_acc	LR	Next LR	Monitor	% Improv	Duration
1 /2	5.100	82.806	3.46033	74.984	0.00100	0.00100	accuracy	0.00	4617.48
2 /2	2.382	89.895	1.78496	89.243	0.00100	0.00100	accuracy	8.56	4567.32

training elapsed time was 2.0 hours, 33.0 minutes, 11.10 seconds)

Figure 5.19: Epoch Wise Results of EfficientNet

The above diagram shows the class loss and the accuracy values as the model is being trained. We can see the accuracy reach a staggering 89% accuracy after just the 2nd epoch.

5.4 YOLOv7

YOLOv7, alternatively referred to as "You Only Look Once version 7," demonstrates improvements in instant object detection techniques, building upon the achievements of its predecessors [26].

The procedure of YOLOv7 [30] begins by preparing the data. A dataset with pictures that have marked regions is very crucial for properly training the model. This data gathering must precisely portray the things you intend to recognize.

In terms of the arrangement, YOLOv7 utilizes a packed convolutional neural network (CNN) as its primary structure [27]. The choice of CNN architecture, such as Darknet or CSPDarknet53, has a significant impact on how well the model performs.

Adjusting the size of the model is another crucial aspect in YOLOv7. It offers flexibility by providing different options in terms of model sizes, such as small, medium, and large, allowing users to strike a balance between accurate detection and quick processing in live situations.

To improve the detection of objects, YOLOv7 utilizes the grouping of anchor boxes. The dataset's precise predictions are enhanced by grouping the actual bounding boxes together to determine appropriate sizes and shapes for the anchor boxes.

The basic design of YOLOv7 often involves enhancements like feature pyramid networks (FPN) or PANet. These components help in capturing qualities of items at various dimensions, resulting in improved accuracy in detection.

Having a well-defined loss function is highly crucial during training, typically involving a combination of localization loss (such as the average of squared differences) and classification loss (such as cross-entropy). This defeat guides the modification of the model's heaviness during both frontward and backward movements.

Education necessitates a variety of procedures such as information augmentation, elimination, and modification of the rate of acquiring knowledge. These techniques assist in enhancing the model's performance on the dataset, enabling it to excel in identifying objects.

During forecasting, YOLOv7 operates in real-time. It employs a grid with a fixed dimension that shifts through the given image, making estimations for enclosing rectangles, probabilities of classes, and level of certainty for each grid block.The architecture of YOLOv7 includes various important elements.

YOLOv7 [30] depends on a deep CNN main part such as Darknet or CSPDarknet53 to gather hierarchical characteristics from input images.

Certain variations incorporate a neck structure, like FPN or PANet, that combines characteristics from various levels to improve the identification of objects with different dimensions.

This section of YOLOv7 consists of several layers that are in charge of foreseeing bounding boxes and categorization likelihoods. It produces forecasts for anchor boxes at various sizes.

YOLOv7 uses anchor boxes to estimate the sizes and shapes of objects. These anchor boxes are defined by grouping of actual bounding boxes.

The ultimate result includes boxes, category names, and certainty ratings for identified items. After the initial processing, additional steps such as non-maximum suppression (NMS) are used to improve the accuracy of the outcomes.

5.4.1 Results of YOLOv7

```
Epoch   gpu_mem       box       obj       cls     total    labels  img_size
  5/9     11.6G   0.02593  0.005461   0.01012   0.04151        18       640: 100% 245/245 [03:26<00:00,  1.19it/s]
          Class      Images      Labels           P           R      mAP@.5  mAP@.5:.95: 100% 62/62 [00:35<00:00,  1.77it/s]
            all        1976        1973       0.455       0.877       0.483       0.227

Epoch   gpu_mem       box       obj       cls     total    labels  img_size
  6/9     11.6G   0.02297  0.005323  0.008743   0.03704        26       640: 100% 245/245 [03:26<00:00,  1.19it/s]
          Class      Images      Labels           P           R      mAP@.5  mAP@.5:.95: 100% 62/62 [00:34<00:00,  1.78it/s]
            all        1976        1973        0.45        0.91       0.526       0.254

Epoch   gpu_mem       box       obj       cls     total    labels  img_size
  7/9     11.6G   0.02451  0.005478  0.008308    0.0383        27       640: 100% 245/245 [03:27<00:00,  1.18it/s]
          Class      Images      Labels           P           R      mAP@.5  mAP@.5:.95: 100% 62/62 [00:35<00:00,  1.76it/s]
            all        1976        1973       0.484       0.838       0.484       0.238

Epoch   gpu_mem       box       obj       cls     total    labels  img_size
  8/9     11.6G   0.02334  0.005367  0.008215   0.03692        18       640: 100% 245/245 [03:26<00:00,  1.19it/s]
          Class      Images      Labels           P           R      mAP@.5  mAP@.5:.95: 100% 62/62 [00:34<00:00,  1.79it/s]
            all        1976        1973        0.47       0.851        0.52       0.259

Epoch   gpu_mem       box       obj       cls     total    labels  img_size
  9/9     11.6G   0.02326  0.005342  0.007972   0.03657        22       640: 100% 245/245 [03:27<00:00,  1.18it/s]
          Class      Images      Labels           P           R      mAP@.5  mAP@.5:.95: 100% 62/62 [00:38<00:00,  1.62it/s]
            all        1976        1973       0.475       0.836        0.48       0.232
            all        1976        1360       0.638       0.963       0.654       0.326
        healthy        1976         613       0.313       0.709       0.306       0.138
```

Figure 5.20: Epoch Wise Results of YOLOv7

Here, we see that the YOLOv7 model, despite being a much more advanced version of the YOLOv5 model, performs quite poorly. However, a significant reason for this might be the massive difference in the number of epochs used in each case of the YOLO models.

Chapter 6

Comparison and Final Analysis

Model	Accuracy and Precision
YOLOv5	0.82 (precision)
YOLOv7	0.52 (precision)
CNN	0.74 (precision)
EfficientNet	0.89 (accuracy)

Table 6.1: Model Wise Accuracy and Precision

6.1 YOLOv5

1. Precision (Accuracy in object detection): 0.82

2. Observations:

   - YOLOv5 is known for its speed and accuracy in real-time object detection.
   - The precision of 0.82 suggests that it is effective in identifying leukemia cells, but there is room for improvement.

3. Possible Reasons for Performance:

   - When it comes to fine grained object detection, this might not be the best of models to use.
   - Perhaps the data augmentation used here was not optimized for Leukemia cell classification

6.2 YOLOv7

1. Precision: 0.52

2. Observations:

   - YOLOv7 seems to have a lower precision than YOLOv5.

- The precision score might indicate that this model has more false positives or misses.

3. Possible Reasons for Performance:

   - The dataset might be the factor here, since this is a rather unconventional set of images we have and the YOLOv7 model might have an issue here.
   - The YOLOv7 was only trained for 10 epochs while the YOLOv5 for 150 epochs.

6.3 CNN (Convolutional Neural Network)

1. Accuracy: 0.77
2. Precision: 0.74

3. Observations:

   - A much known model is the CNN model which has not disappointed us in any stretch of the imagination and performed quite decently.
   - This model has the ability to identify and isolate important features for leukemia cell detection.

4. Possible Reasons for Performance:

   - CNNs are usually quite effective in such tasks.
   - The hyperparameters might have very well played an important role here.

6.4 EfficientNet

1. Accuracy: 0.89
2. Observations:

   - EfficientNet performs the most impressively among all of the models used.
   - This model is quite well known for it's efficiency, as it's name suggests.
   - The model is somehow quite suited for this specific kind of dataset.

3. Possible Reasons for Performance:

   - EfficientNet is designed such that the computational load is at a minimum while the performance is not being compromised.
   - Transfer learning might be one of the fundamental reasons for it's success.
   - Data augmentation could have made the model even more accurate.

6.5 General Insights

- The fundamental choice of whether we use an older or a much more advanced model does matter as we see in the case of using EfficientNet.
- No matter how good the model is, the data we feed into it is what dictates it all, well preprocessed data and it being of good quality is key to a model's performance.
- Data augmentation, if properly applied, can exaggerate the performance of certain models such as these.
- YOLO models are more used for object detection rather than classification and hence might require certain adjustments for such tasks.
- Class loss, precision and accuracies are metrics that must be carefully observed and controlled to the best of our capacity to ensure the best of results.

6.6 A test of our own

- After we ran all the models we used our base CNN model and changed some of it's parameters to demonstrate and understand the nature of the data.
- We introduced an extra dense layer into the architecture and tried working with that model but to our surprise after running for 8 epochs we gained a validation accuracy of 0.4766 which is significantly lower than the previous model.
- We can conclude from this that while using the CNN model the increasing of neurons is not a factor that necessarily affects the accuracy of the model for our dataset.

```
Epoch 1/10
56/56 [==============================] - 591s 11s/step - loss: 0.6942 - accuracy: 0.5006 - val_loss: 0.6940 - val_accuracy: 0.4941
Epoch 2/10
56/56 [==============================] - 573s 10s/step - loss: 0.6932 - accuracy: 0.5089 - val_loss: 0.6945 - val_accuracy: 0.4902
Epoch 3/10
56/56 [==============================] - 572s 10s/step - loss: 0.6928 - accuracy: 0.5156 - val_loss: 0.6937 - val_accuracy: 0.5039
Epoch 4/10
56/56 [==============================] - 574s 10s/step - loss: 0.6922 - accuracy: 0.5229 - val_loss: 0.6926 - val_accuracy: 0.5176
Epoch 5/10
56/56 [==============================] - 567s 10s/step - loss: 0.6936 - accuracy: 0.5106 - val_loss: 0.6929 - val_accuracy: 0.5117
Epoch 6/10
56/56 [==============================] - 535s 10s/step - loss: 0.6932 - accuracy: 0.5134 - val_loss: 0.6936 - val_accuracy: 0.4980
Epoch 7/10
56/56 [==============================] - 570s 10s/step - loss: 0.6929 - accuracy: 0.5134 - val_loss: 0.6949 - val_accuracy: 0.4805
Epoch 8/10
56/56 [==============================] - 569s 10s/step - loss: 0.6928 - accuracy: 0.5162 - val_loss: 0.6955 - val_accuracy: 0.4766
```

Figure 6.1: Epoch Wise Results of Modified CNN

Chapter 7

Conclusion

7.1 Conclusion

This research sheds light on the intricate details, ramifications, and potential for further advancement of the detection and classification of leukemia in peripheral blood cell images using neural networks. We used a dataset from Kaggle, a well-known website for data science and machine learning enthusiasts, for our investigation and analysis of the blood cells. We investigated the subtle relationships and patterns that existed in the data using this substantial dataset, producing insightful findings and expanding our understanding. We applied these algorithm for detection and classification. This study has successfully identified leukemia cells using the strength of neural networks and computer vision techniques, improving the efficiency and accuracy of diagnosis. Although there has been a noticeable improvement in accuracy, there is still room for improvement. In order for the models to classify various types of leukemia using the data, we plan to expand our dataset in the future. Additionally, use vision transformers and model fusions to try to improve accuracy. We also plan to employ a federated strategy in the long run. But the overall analysis of the various models used shows the scope for more research and understanding that is yet to occur.

Bibliography

[1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[2] H.-C. Shin, H. R. Roth, M. Gao, *et al.*, "Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning," *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1285–1298, 2016.

[3] T. Tran, J.-H. Park, O.-H. Kwon, K.-S. Moon, S.-H. Lee, and K.-R. Kwon, "Classification of leukemia disease in peripheral blood cell images using convolutional neural network," *Journal of Korea Multimedia Society*, vol. 21, no. 10, pp. 1150–1161, 2018.

[4] A. Acevedo, S. Alférez, A. Merino, L. Puigví, and J. Rodellar, "Recognition of peripheral blood cell images using convolutional neural networks," *Computer methods and programs in biomedicine*, vol. 180, p. 105 020, 2019.

[5] N. Ahmed, A. Yigit, Z. Isik, and A. Alpkocak, "Identification of leukemia subtypes from microscopic images using convolutional neural network," *Diagnostics*, vol. 9, no. 3, p. 104, 2019.

[6] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*, PMLR, 2019, pp. 6105–6114.

[7] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[8] G. Marques, D. Agarwal, and I. De la Torre Díez, "Automated medical diagnosis of covid-19 through efficientnet convolutional neural network," *Applied soft computing*, vol. 96, p. 106 691, 2020.

[9] D. Bhatt, C. Patel, H. Talsania, *et al.*, "Cnn variants for computer vision: History, architecture, application, challenges and future scope," *Electronics*, vol. 10, no. 20, p. 2470, 2021.

[10] D. Bhatt, C. Patel, H. Talsania, *et al.*, "Cnn variants for computer vision: History, architecture, application, challenges and future scope," *Electronics*, vol. 10, no. 20, p. 2470, 2021.

[11] D. Bhatt, C. Patel, H. Talsania, *et al.*, "Cnn variants for computer vision: History, architecture, application, challenges and future scope," *Electronics*, vol. 10, no. 20, p. 2470, 2021.

[12] E. Gavas and K. Olpadkar, "Deep cnns for peripheral blood cell classification," *arXiv preprint arXiv:2110.09508*, 2021.

[13] T. Kattenborn, J. Leitloff, F. Schiefer, and S. Hinz, "Review on convolutional neural networks (cnn) in vegetation remote sensing," *ISPRS journal of photogrammetry and remote sensing*, vol. 173, pp. 24–49, 2021.

[14] R. Kumar, S. Joshi, and A. Dwivedi, "Cnn-sspso: A hybrid and optimized cnn approach for peripheral blood cell image recognition and classification," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 35, no. 05, p. 2 157 004, 2021.

[15] A. T. Philip, S. Shifaana, S. Sunny, and P. Manimegalai, "Detection of acute lymphoblastic leukemia in microscopic images using image processing techniques," *Journal of Physics: Conference Series*, vol. 1937, no. 1, p. 012 022, Jun. 2021. DOI: 10.1088/1742-6596/1937/1/012022. [Online]. Available: https://dx.doi.org/10.1088/1742-6596/1937/1/012022.

[16] H. Sung, J. Ferlay, R. L. Siegel, *et al.*, "Global cancer statistics 2020: Globocan estimates of incidence and mortality worldwide for 36 cancers in 185 countries," *CA: a cancer journal for clinicians*, vol. 71, no. 3, pp. 209–249, 2021.

[17] W. Wu, H. Liu, L. Li, *et al.*, "Application of local fully convolutional neural network combined with yolo v5 algorithm in small target detection of remote sensing image," *PloS one*, vol. 16, no. 10, e0259283, 2021.

[18] F. Zhou, H. Zhao, and Z. Nie, "Safety helmet detection based on yolov5," in *2021 IEEE International conference on power electronics, computer applications (ICPECA)*, IEEE, 2021, pp. 6–11.

[19] S. M. ABAS, "Diagnosing the leukemia using faster region based convolutional neural network," *Journal of Applied Science and Technology Trends*, vol. 3, no. 02, pp. 35–38, 2022.

[20] F. Akalin and N. YUMUŞAK, "Detection and classification of white blood cells with an improved deep learning-based approach," *Turkish Journal of Electrical Engineering and Computer Sciences*, vol. 30, no. 7, pp. 2725–2739, 2022.

[21] K. Ali, Z. A. Shaikh, A. A. Khan, and A. A. Laghari, "Multiclass skin cancer classification using efficientnets–a first step towards preventing skin cancer," *Neuroscience Informatics*, vol. 2, no. 4, p. 100 034, 2022.

[22] M. E. Billah and F. Javed, "Bayesian convolutional neural network-based models for diagnosis of blood cancer," *Applied Artificial Intelligence*, vol. 36, no. 1, p. 2 011 688, 2022.

[23] C. Fatichah, N. Suciati, T. Mustaqim, and A. R. Revanda, "Detection of acute lymphoblastic leukemia subtypes using yolo and mask r-cnn," in *2022 5th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, IEEE, 2022, pp. 413–418.

[24] M. Horvat and G. Gledec, "A comparative study of yolov5 models performance for image localization and classification," in *Central European Conference on Information and Intelligent Systems*, Faculty of Organization and Informatics Varazdin, 2022, pp. 349–356.

[25] T. Mostafa, S. J. Chowdhury, M. K. Rhaman, and M. G. R. Alam, "Occluded object detection for autonomous vehicles employing yolov5, yolox and faster r-cnn," in *2022 IEEE 13th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, IEEE, 2022, pp. 0405–0410.

[26] Y. Wang, H. Wang, and Z. Xin, "Efficient detection model of steel strip surface defects based on yolo-v7," *IEEE Access*, vol. 10, pp. 133 936–133 944, 2022. DOI: 10.1109/ACCESS.2022.3230894.

[27] D. Wu, S. Jiang, E. Zhao, *et al.*, "Detection of camellia oleifera fruit in complex scenes by using yolov7 and data augmentation," *Applied Sciences*, vol. 12, no. 22, p. 11 318, 2022.

[28] S. Abd El-Ghany, M. Elmogy, and A. A. El-Aziz, "Computer-aided diagnosis system for blood diseases using efficientnet-b3 based on a dynamic learning algorithm," *Diagnostics*, vol. 13, no. 3, p. 404, 2023.

[29] Z. Cheng and Y. Li, "Improved yolov7 algorithm for detecting bone marrow cells," *Sensors*, vol. 23, no. 17, p. 7640, 2023.

[30] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 7464–7475.