International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 01 | January - 2024 SJIF Rating: 8.176 ISSN: 2582-3930

Image Classification of White Blood Cells using Deep Learning Models

Lalith Samanthapuri, Sujay Bhagawan Ghadge, Vishaal Pedapatnam, Rama Krishna Chavali, Shashwat Mahodaya, Bathineni Taraka Rameswara K Durga Prasad, Varun Vallabhaneni, Nunna Karthik, Athota Koushik

Abstract

White blood cell classification is critical for medical diagnosis. In practice, a hematologist classifies white blood cells by carefully studying a small blood sample under a microscope. Thecurrent methods for detecting the subtypes f white blood cells are time-consuming and prone toinaccuracy. Computer-assisted white blood cell identificatin and diagnosis cut the time it takesto identify white blood cells in half and eliminate human error. In the realm of white blood cell classification, several deep learning algorithms have been created in recent years that can identify. It is easier for trained people to make conclusions about diseases after classified bloodcells. Since deep learning models perate with large databases, training, and testing networks, consequently, when the model is trained with powerful computing skills accessible, distinct perfect models can be constructed and used in medical analysis and applications dealing with the number of white blood cells and subtypes of white blood cells.

Keywords: White blood cells, deep learning, Convolutional neural network, image classiffica- tion, image processing, neutrophils.

Introduction

Blood is vital and many bodily functions depend on healthy blood. Blood health can be as- sessed by analyzing blood components (cells, etc.).Plasma makes up the remaining 55% of blood's volume, whereas blood cells make up around 45% of it. Platelets, white blood cells, and red blood cells are the three different types of blood cells. White blood cells (WBCs) play a significant role in the immune system of the human body. They play a significant part in the regulation of immune reaction and inflammation. The immune system, which defends against microorganisms and other chemicals, is under their control. WBCs are classified as polynu- clear (neutrophil, eosinophil and basophil) or mononuclear (monocyte and lymphocyte). The body is guarded against bacterial and fungal illnesses by neutrophils, which are separated intotwo functionally different subpopulations: neutrophil-killers and neutrophil-cagers. Allergies, parasite infections, collagen illnesses, and spleen and central nervous system disease can cause an increase in eosinphils. Basophils are primarily responsible for allergy and antigen responses since they release the chemical histamine, which causes blood vessels to dilate.In order to remove germs and other foreign invaders from the body, immune cells work in conjunction with lymphocytes to help. The tissues' foreign substances are phagocytosed by monocytes. Monocytes are produced more abundantly by illnesses caused by bacteria, viruses, listeria, andmalaria. The distribution of these five classes among WBCs in the body is 62 , 2.3 %, 0.4 %, 30
%, and 5.3 % respectively. A kind of blood cell called a leucocyte, sometimes referred to as an

Chapter 1. Introduction

immune cell, is crucial to the functioning of the human immune system. One of the worst dis- eases afflicting people today is leukaemia. It begins in the bone marrow and spreads to WBC, while certain kinds of leukimia begin in other blood cells. The WBC asserts that engineering technology-extracted qualities like as textures, patterns, and features in appearance play a sig- nificant part in the in-depth learning of picture categorization.Techniques for classifying, frag- menting, identifying, and extracting features from images are becoming commonplace. The previously introduced model for Hemorrhagic Analyzer's WBCs image sorting required labo- rious manual processing. Fortunately, as was already indicated, technologies based on machine learning may swiftly close the gaps. Numerous methods for classifying blood cells have been developed by researchers due to the importance of blood cell classification in diagnosis.We are not aware of any large, broadly available white blood cell detection and categorization datasets.To thoroughly and impartially assess the efficacy of the given method, we have gathered some currently available dataset. The two types of WBCs are mononuclear (monocyte, lymphocyte) and polynuclear (neutrophil, eosinophil, basophil).To count the various types of WBC cells, anumber of processes are required, such as detecting and recognising the type in the microscopicpicture. Even for a professional, this process takes time and is prone to mistakes. Computer assisted automatic detection and diagnostic systems allow for the reduction of the amount of time required for diagnosis, the elimination of potential human mistake, and the achievement of a trustworthy outcome. [11].Numerous research in the literature concentrate on classifying WBC pictures according to cell types. The goals of this study are to simultaneously classify many cells using a regionally based convolutional neural network, one of the deep learning techniques, to distinguish between visible and invisible WBC due to overlap or exclusion of particular picture regions, and to detect visible WBC.

By omitting human processing stages, deep learning models like CNN can at times speed up computation. It attempts to categorise all of the photos at once using high level characteristics. This means that the CNN can be considered as a powerful technology for building strong imageclassification.

Overview / Literature Review

Neutrophils, eosinophils, lymphocytes, monocytes, and basophils are the five primary types of white blood cells that make up the immune system, which protects the body against foreign substances. Each type serves a particular function and makes a different contribution to the immune system. Being able to classify and, consequently, count these different constituentsis essential for evaluating patient health and infection risks. White blood cell types are typi- cally determined by laboratory tests. Under the microscope, the staining procedure and manualimage interpretation are laborious and prone to human mistake. The absence of training data that takes into consideration the physical alterations of white blood cells is another key barrier that prevents trained Classifiers from generalising well. In this study, the classification of white blood cells into their five categories is examined utilising advanced deep neural networks (such as VGG-16, ResNet, and DenseNet), generative adversarial networks (GAN), and image pro- cessing techniques.The initialization of the DNNs' weights can either be done at random or by using weights that have already been created for the CIFAR-100 dataset. Unlike other methods that require complex image preprocessing and human feature extraction before classification, our approach uses the produced images right away.As a substitute for the laboratory setting, the paper looks into the automatic classification of white blood cells utilising quick, accurate, andinexpensive DNNs.Kouzehkanan et al.(2021)

We concentrate on the most recent DNNs that have been pretrained on the CIFAR-100 dataset and can classify white blood cells as neutrophils, eosinophils, lymphocytes, monocytes,or basophils. VGG, ResNet, and DenseNet are some of these DNNs. The suggested methodmay successfully classify white blood cells, according to extensive testing. Validation accu-racy is 98.8% when using the top DNN model, DenseNet-169. We discover that the proposedmethodology outperforms competing methods, particularly those that rely on intricate image processing and labor-intensive feature engineering.AC02909904(1987) Several works have employed pre-trained CNNs.17–20 to extract traits needed for WBC classification. When used to the picture features obtained by pre-trained CNNs for the purpose of diagnosing acute lym-phoblastic leukaemia, Rehman et al. 17 investigated the accuracy of three different classifiers.The SVM classifier, they discovered, yields the best results. In18, features from three popular CNN architectures (AlexNet, GoogleNet, and ResNet-50) were mixed. The right features werethen selected using the maximum information coefficient and ridge approaches.

Finally, WBCs were categorised using a quadratic discriminant analysis method. Similar to this, Togacar et al.19 used a pre-trained Alexnet network to extract attributes from a quadratic discriminant analysis model to categorise WBCs. A pre-trained CNN, a statistically improvedsalp swarm approach, an SVM model, and an SVM model were all used by Sahlol et al. in their feature extraction method. You may directly train deep learning neural networks to clas-sify WBCs1,21–26. When classifying WBCs, Hedge et al.1 used pre-trained networks and didnot use them. They discovered that training a network completely from scratch yields superior outcomes versus fine-tuning an AlexNet-pretrained network.Zhao et al.(2017). Writers in 21 tackled the categorization of WBCs by

fine-tuning pre-trained AlexNet and LeNet-5 networksas well as training a new CNN from scratch. They claimed that the innovative network they hadpresented outperformed the previously described tuned networks. W-Net, a brand-new CNNarchitecture created by Jung et al.22, was used to categorise WBCs in the LISC dataset. Bay-dilli and Atila23 categorised the WBCs in the LISC dataset using capsule networks. Banik etal.24 created a fused CNN model to perform a differential WBC count and put it to the testusing the BCCD dataset. Liang et al.25 combined the output feature vector of the fatten layerin a customised CNN and a long short term memory network to identify WBCs in the BCCD dataset. A novel complicated fused CNN that was introduced in 26 was trained entirely fromscratch using 10,253 improved WBCs images from the BCCD dataset.Patil et al.(2021). The proposed CNN in26 is quite sophisticated, although it only has 133,000 parameters. In order to partition the cells, the Zack algorithm estimates a threshold value. The social spider opti- misation method was used to choose the top ten retrieved form, texture, and colour attributes.Last but not least, they used a KNN classifier to divide WBCs into two categories: healthy andsick. Ghane et al. 31 created a brand-new method to segment the nucleus of the WBCs by fusing Otsu's thresholding algorithm, k-means clustering, and a modified watershed algorithmin a novel fashion. They were successful in doing so with a precision of 96.07%.Tavakoli et al. (2021) Laosai and Chamnongthai32 looked on how to recognise acute myelogenous leukaemia and ALL from WBC images.Jung et al.(2019). After locating the nuclei using the k-means cluster- ing method, they obtained form and texture data and then used an SVM classifier to categorise the WBCs. Due to the difficulty of the manual procedure, computer-assisted methods for mea- suring white blood cells (WBC) have gained a lot of popularity. White blood cell segmentationand identification from microscopic blood images can be done with high accuracy, according to recent research. The distribution of the five groups reflects the state of the immune system, despite the fact that categorising the discovered cells remains difficult and in high demand. UseW-Net, a CNN-based WBC classification model, to accurately identify different WBC types. To divide the traits into five categories, our model employs a softmax classifier, two fully con- nected layers, and two fully connected layers. From WBC pictures, information is extracted using three convolutional layers. W-Net performs more precisely than state-of-the-art methods.We compare W-Net to ResNet to show how effective it is at classifying WBC images (He et al., 2016).

We further experimented the LISC public data to show how other researchers might benefit from our trained W-Net model (Rezatofighi and Soltanian-Zadeh 2011).Analysis of WBC count and type is essential for disease diagnosis. We attempted to address the issue that real-world applications still struggle to distinguish between the five different types of WBCs (neutrophils, eosinophils, basophils, lymphocytes, and monocytes), despite the fact that there are numer- ous techniques for doing so from microscopic images of a blood smear. Because of the quick development of computer vision and machine learning, accurate categorization jobs may now be accomplished across a range of industries. This adds W-Net, a CNN-based architecture,to enable accurate categorization of

the five WBC classes. We evaluated the proposed archi- tecture on a real-world dataset and addressed problems like the transfer learning property and class imbalance. W-Net recorded a 97 percent classification accuracy on average. Using a convolutional neural network, normal white blood cells are recognised and categorised.

The two exercises that follow will help the programme learn more about the characteristicsand form of a typical WBC. The first difficulty is identifying distinctive features of a typical white blood cell. The second objective is to categorise various types of healthy white blood cells. The system can identify between aberrant and normal WBCs by contrasting their high- level properties using a Convolutional Neural Network. CNNAlmezhghwi & Serte(2020). In order to make a decision, doctors and other medical professionals may need to employ this process of recognising and classifying WBC. On a dataset of 10,000 blood cell pictures, the suggested network performs up to 96.78% accurately.By bypassing some steps, Convolutional Neural Networks (CNN) reduce processing times. It attempts to classify images while also detecting broad traits.

Convolutional neural networks (CNNs) are therefore a potent tool for creating reliable pic- ture classifiers. A dataset of 10.000 JPEG photos of microscopes is used in this study.Kutluet al.(2020).

The blood dataset is divided into the four main subtypes of white blood cells found inblood—monocytes, lymphocytes, neutrophils, and eosinophils. The photos of the basophils'cells are not part of this dataset. For each category, there are about 2500 pictures.Liang et al.(2018). Using various strategies and methodologies, an image classifier may be built to categorise blood photos into the right classes using this blood dataset. Convolutional neural networks also have the benefit of obtaining high-level characteristics from microscope pictures and then classifying them quickly and with minimal code implementation. Convolutional neu- ral networks are challenging to build and grow because of their extreme complexity in terms ofthe number of layers required to extract useful high level properties from microscope images and how to categorise these features. This makes network building challenging.Huang et al. (2021). This study will present a system that recognises white blood cells in microscope images and then classifies them into one of four groups using a convolutional neural network: Mono-cytes are classified as Class A cells, lymphocytes as Class B cells, neutrophils as Class C cells,and eosinophils as Class D cells. A well-known deep learning architecture that has received alot of recent development is the convolution neural network (CNN). It was influenced by how living organisms naturally perceive the world around them.

This effective identification technique has garnered a lot of interest. Because it avoids thelaborious picture pre-processing and offers end-to-end training, the network is frequently used. CNN possesses three key traits: local convolution, weight sharing, and multiple convolutional kernels.Pullanji & Muthuswamy(2022).These characteristics indicate that the CNN's layout is more like a biological brain network than not. It has the ability to recognise two-dimensional pictures that have been scaled, twisted, and shifted without deforming them.A classifier builtusing the CNN approach is created to recognise different photos of white blood cells. There arealso

certain neural network architectures, such as AlexNet GoogleNet VGGNet, that performbetter when dealing with visual imagery. The properties of natural animals and plants may be extracted from data using AlexNet and GoogleNet. People with normal vision may see thesecharacteristics in animals and plants.

Face recognition is the primary field for which VGGNet's network architecture is intended.Varra(n.d.)The result is unsatisfactory if these network designs are applied directly in the character- ization and categorization of white blood cells. The properties of the white blood cell are not readily apparent since it is a tiny representation of a microbe under a microscope, and the hitherto common network architectures are insufficient to deal with the problem.

Chapter 3 Methodology

3.1 Objectives

- To classify and identify white blood cells with the aid of deep learning methods.
- To work on different pre-trained models and compare the results to analyze their perfor-mances.
- To decide which categorization model is the most appropriate.
- To assess the proposed model's performance using a number of metrics, including Accu-racy, F1-Score, and training time on multiple benchmark datasets.

3.2 Scope of the Project

The ultimate goal of this project concentrating on picture classification of white blood cells using deep learning models is to develop trustworthy and accurate models for automated whiteblood cell classification, which may be useful in medical diagnosis and research.

However, in general, the following tasks can be included in the scope of such a project:

- Data Collection : Gathering enough white blood cell photos with labels to train and test deep learning models.
- Data Preprocessing : Preparing the photos for the models' training may involve processeslike image scaling, normalisation, and augmentation.

• Model Development : Convolutional neural networks (CNNs) and other deep learning models that can classify the white blood cell images into different classes (e.g., neu- trophils, lymphocytes, monocytes, eosinophils, basophils).

• Model Training and Evaluation :employing criteria like accuracy, precision, recall, andF1 score to assess the performance of the created models on the labelled data.

The project's overall objective is to create an accurate and effective deep learning model that can categorise photos of white blood cells into the appropriate kinds, which has a variety of uses in medical diagnosis and research.

PACKAGES SCIKIT-LEARN:

• Initially a third-party addition to the SciPy library, Scikit-Learn is now a stand-alone Python package available on Github. Support vector machines, random forests, gradient boost- ing, and DBSCAN are some of Scikit-Learn's classification, regression, and clustering tech- niques.

NUMPY:

• This Python has some model that gives us a quick maths formula for all kind of problem-atics. Also it reads and manipulate data from numpy package.

TENSORFLOW:

• One of the top Python libraries for deep learning applications is commonly regarded as TensorFlow. It offers a wide range of adaptable tools, libraries, and community resources and was created by the Google Brain Team. TensorFlow may be used to build deep learning modelsand neural networks by both novices and experts.

SEABORN:

• Matplotlib is used by the Python data visualisation programme Seaborn. It offers a so- phisticated drawing tool for creating eye-catching and educational statistical visuals.

PANDAS:

• These packages used to write or read files. Frame lines often used to manipu-late files.

MATPLOT-LIB:

• Set visuals is a useful tool for finding patterns dataset given. It is easy to fuzz functiondata.

3.3 DATASET

The white blood cell Raabin-WBC images from typical peripheral blood samples, which werepublished in 2021, make up the data set.

The dataset carries approximately 30000 WBCs and artefacts (colour spots). “To reassure accurate data, a considerable wide variety of cells had been categorized via way of means of specialists, and the ground truth of nucleus and cytoplasm had been extracted (approximately 1145), as well”.

In 5 distinct files, 5 different kinds of cells had been stored. Detailed information about thedataset is given in the below table.

	Lymphocyte	Monocyte	Neutrophil	Eosinophil	Basophil	Total
All WBCs	3461	795	8891	1066	301	14514
Training set	2427	561	6231	744	212	10175
Test set	1034	234	2660	322	89	4339

Table 3.1: Dataset Information

3.4 Separation Of Dataset

A dataset that was divided into three sets—a training set, a testing set, and a validation set—isreferenced in the information provided.

The total dataset, which consists of 8142 photos, is composed 80% by the training set. Al-though the information supplied does not specify how many photographs are in each category,these photos are grouped into five groups. The quantity of images in each group may be aboutequal, or they may be unevenly distributed, with some groups having more photos than others.The testing set is made up of 4339 images in total, which accounts for around 20% of the entire dataset. Although the information supplied does not state how many photographs are in each class, these photos are also from five distinct classes.

Although the validation set is not specified explicitly, it is standard practise to utilise a val-idation set to assess a model’s performance during training. This is accomplished by regularlyassessing the model against the validation set and modifying the model’s design or parameters as necessary. According to the information given, the validation set contains 2033 images fromfive distinct classes.

The collection as a whole includes 14514 images spread across five classifications. It’s unclear how the images are distributed throughout the classes, thus it’s conceivable that all ofthe classes are represented equally or that certain classes have more photos than others. It is crucial to take the class distribution into account when conducting experiments and analysing findings since the dataset’s balance might have an impact on how well machine learning models perform when trained on it.

3.5 Data Loading

Loading Image Data: The dataset will be loaded one at a time by the generators, who will thentrain on the network and recycle memory.tf.keras.preprocessing.image: ImageDataGenerator: Produce tensor image data in batches while improving the data in real-time.After the image hasbeen resized and improved, the function will be invoked. A single picture (a Numpy tensor of rank 3) should be the only input for the function, which should then output another tensor of the same type validation split: The percentage of photos set aside for verification (strictly between 0 and 1).The photos are divided into two categories: test data and training data.

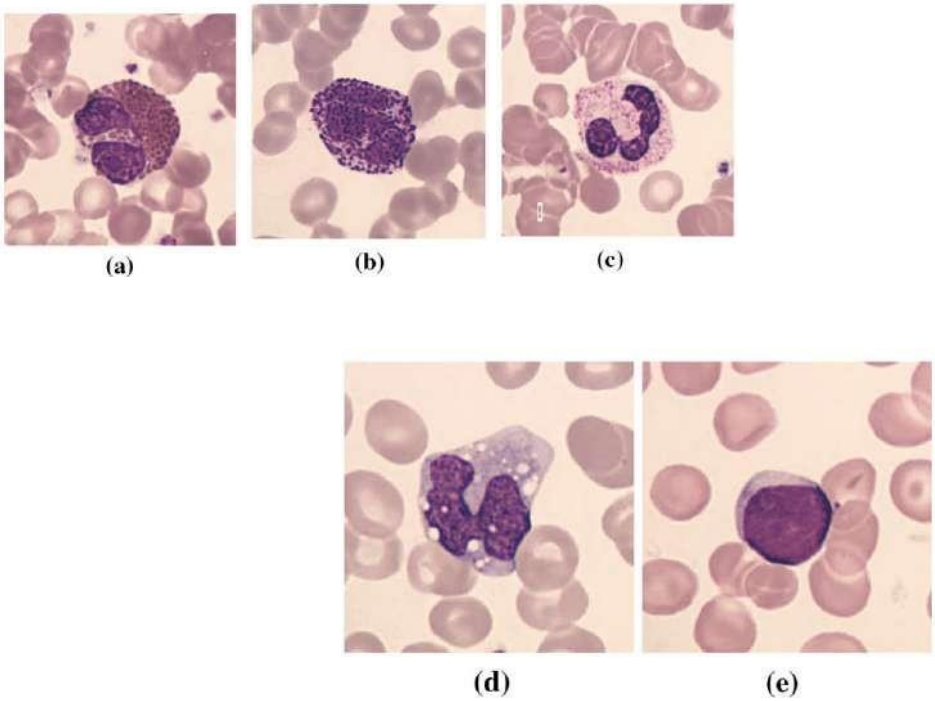

Figure 3.1: Types Of WBCs : (a) eosinophils, (b) bosophils, (c) neutrophils, (d) monocyte, (e)lymphocyte

3.6 Problem Statement:

- Identifying and measuring the quantity of white blood cells is one of the challenges inmedical research.
- The quantity of red blood cells is the important variable. The WBCs account for about 1% of the total blood volume in an adult who is healthy. The work of establishing WBC subtypes is made significantly more

challenging by the difficulty in recognising WBCsdue to the low amount of WBCs in blood.

- Any increase or decrease in the number of categorization subtypes suggests that there isa health issue and that the body is responding to an illness.

- Under a light and electron microscope, blood-smeared slides are often examined to iden-tify WBC kinds.

- Before being examined under a microscope, blood cells are coloured. To accurately diagnose a nucleus, pathologists must look at its structure and evaluate its size in relationto that of RBCs.

- It takes longer and is more error-prone because it is a manual process.

- The trade-off between picture quality and tiny field of vision, as well as the possibil- ity of human error associated with mechanical glass slide scanning, are the two biggestdrawbacks of manual categorization.

.

3.7 Evaluation Metrics/Performance:

The effectiveness of the model is evaluated using metrics. One of the most important principlesof machine learning is how to evaluate your model.Metrics like recall, accuracy, and precisionare helpful for evaluating classification algorithms on balanced datasets. However, other meth-ods, like as ROC/AUC, are more useful in determining the model's performance when the datais uneven.

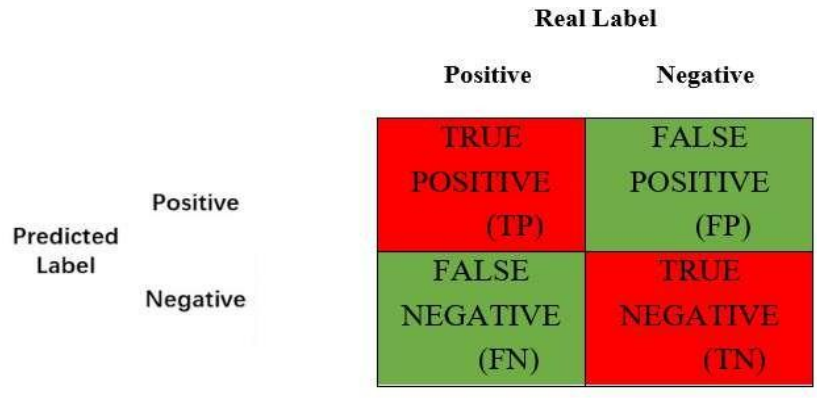

Figure 3.6: Confusion Matrix

A CONFUSION MATRIX is a table layout that helps with visualising the different out- comes and scenarios of a classification original challenge. It generates a table with all of the predicted and actual results of a classifier.

PRECISION-RECALL : Percentage of positive predictions that are correct (i.e.,) The per-centage of accurately anticipated positive observations to all predicted positive observationsis known as precision. The proportion of accurately predicted optimistic parameter estimates. (The prototype accurately predicted the fraction of actual de-faulters.) Sensitivity (Recall) - Yes, the proportion of accurately guessed be use to all findings in the full class is referred to assensitivity.

F1 SCORE : The F1 score, with 1.0 being the best outcome and 0.0 being the worst, is a weighted harmonic mean of recall and accuracy. F1 scores usually perform worse than accuracy tests since they combine precision and recall while determining their results. When contrasting classifier models, the weighted average of F1 is typically recommended rather than overall accuracy.

ACCURACY_SCORE : The quantity of positive predictions pieces of data among all the sample points is did refer to as exactness.

SUPPORT : Support in a dataset refers to the number of actual instances of the class.Unbalanced support in the training data may indicate the need for stratified sampling or rebalancing because
it may cause structural problems with the classifier's reported scores. While diagnosing the evaluation process, support is constant across models.

3.8 Implementation of Deep Learning Models

Machine learning is a subset of deep learning, which is a network with three or more layers. Despite their limitations, these neural networks try to mimic the way the human brain functions and allow it to "learn" from enormous amounts of data. Even while a neural network with only one layer may still produce approximations, more hidden layers can aid in enhancing andoptimising for accuracy.
By executing mental and physical activities without the need for human involvement, deeplearning, a technique that underpins many artificial intelligence (AI) products and services, in- creases automation. Both established products and services (including digital assistants, voice- activated TV remote controls, and credit card fraud detection) and cutting-edge innovations (like self-driving automobiles) are powered by deep learning technology.

In order to comprehend a picture as a whole, image categorization is a fundamental diffi- culty. Giving the image a specific label has the purpose of classifying it. The analysis of photoswith only one object visible is a common topic in the field of image classification. In contrast,object detection examines more realistic circumstances where several objects could be presentin an image and necessitates both classification and localisation tasks.

3.9 Convolutional Neural Network(CNN)

Convolution neural network (CNN), a popular deep learning architectural style, was recently developed and was informed by the way that live organisms naturally see their own systems.

It's a form of successful recognition strategy that's gained a lot of buzz. The network is well- liked because it eliminates the need for time-consuming image pre-processing and allows for full training.CNN's three unique characteristics are local convolution, weight sharing, and mul-tiple convolutional kernels.[2]. These characteristics make CNN's design more like a biologicalbrain network. Two-dimensional pictures that have been rotated, scaled, or moved can be rec-ognized and modified.

The model comprises of a layer of a convolutional neural network that has already been trained and a fully connected layer with Softmax output.
Using a convolutional neural network that has already been trained We utilised the weight values we discovered during initial training on the data set provided by ImageNet for the initial weights of our CNN model. Convolutional and pooling are the first two layers of convolutional neural networks. The convolutional neural network's top layer, which is also its most crucial one, is calculated by performing convolution operations on feature maps from the layer below using convolution windows of variable widths. The feature map of the preceding layer is suc- cessively slid using convolution windows of increasing sizes. Typically, the convolutional layerhas a window size of 33 or 55 pixels. The number of weight parameters also varies. The valuesof the neurons in each feature map of the convolutional layer are convolved using matching windows, and the result is computed using the excitation function of the layer.

Pooling layer: Following a convolution layer is frequently a pooling layer. This layer's main goal is to scale down the complex feature map to save money on equipment.Following a convolution layer is frequently a pooling layer. This layer's main goal is to scale down the complex feature map to save money on equipment. During max pooling, the feature map's most crucial component is extracted.
In a relatively predetermined size image segment, calculated abuse average pooling is the most prevalent element. The entire sum of the elements inside the chosen segment is calculatedusing add pooling. The convolutional and

fc layers are frequently connected via the pooling

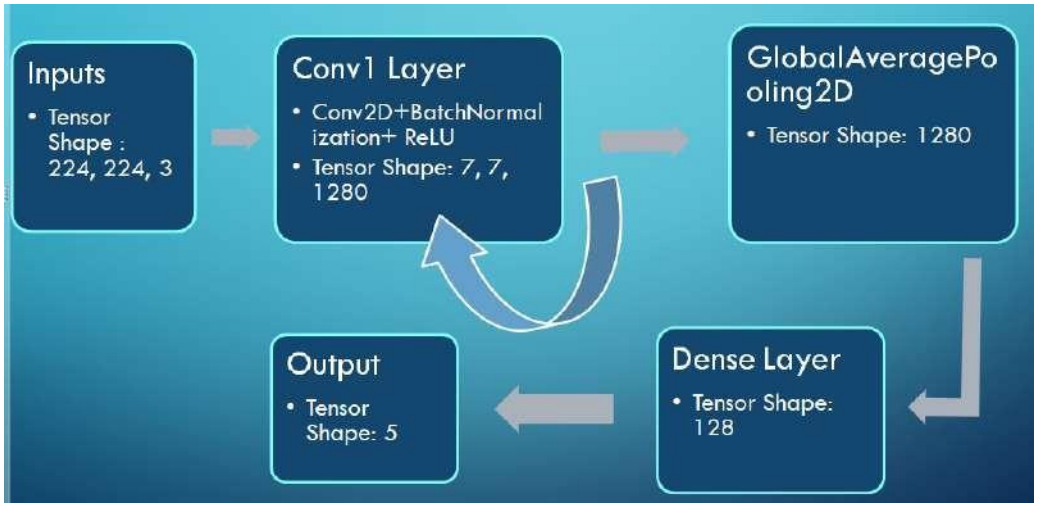

Figure 3.2: CNN Architecture

layer.

The fully connected (fc) layer: In addition to the neurons, it also stores the weights and biases and connects the neurons across layers. The output layer in a CNN design is typically positioned before the last few levels.

The input images from the layers that came before are now 2-dimensional and aligned to the fc layer. The mathematics-related activities are frequently carried out after the flattened vector is scattered over a number of additional fc levels. The categorising process starts at thispoint.

Dropout: After all of the features are attached to the fully connected layer, the training dataset can start to fit too well. The process of over fitting is when a model works so well on the data used for training that it has a detrimental impact on how well it performs on new data.A dropout layer, which reduces the size of the model by removing a small number of neurons from the network of neurons during training, solves this issue. 30 % of the neural network's nodes are randomly removed after passing a dropout of 0.3. Activation functions: In the end, one of among the most important components of the CNN model is the activation process. Any continuous or complex network variable link may be learned and approximated using them. In the network, it offers nonlinearity. The ReLU, Softmax, tanh, and sigmoid activation functionsare most frequently used. Each of the operations has a certain purpose. A CNN model for multi-class classification uses the sigmoid and Softmax functions.

3.10 Residual Network(ResNet)

The vanishing gradient problem may occur if we use too many layers in a deep learning net- work, even if we may get better results. ResNet (Residual Neural Networks) can tackle this problem by jumping over layers via skip connections or short-cuts. Double or triple layer skips are commonly used in ResNet models. ResNet is a CNN with a lot of applications in image processing. In order to train a new model using ReLU, Softmax loss function, and momentum optimizer, we employed the ResNet50 network (50 Conv. layers). The ideal hyperparameters were found to be learning rate = 0.001, decay = 0.0001, momentum = 0.9, batch size = 32, and training epochs = 50.

Residual Block: This design integrates the idea of residual networks to address the problemof vanishing/exploding gradients. A method called skip connections is used in this network. The skip connection links directly to the output after skipping a few training phases. This network allows the network to fit the residual mapping rather than having layers learn the un-derlying mapping. If the network is fitted, H(x):= F(x) + x is formed in place of, for instance, H(x), the initial mapping.

Employing this kind of skip connection has the advantage that regularisation will stop any layer from impairing the functionality of the architectural design. Therefore, when training very deep neural networks, issues caused by vanishing/exploding gradients can be avoided.

Similar ideas are referred to as "highway networks," and these networks also employ skip connections. Parametric gates are used in these skip connections, just as they are in LSTM.

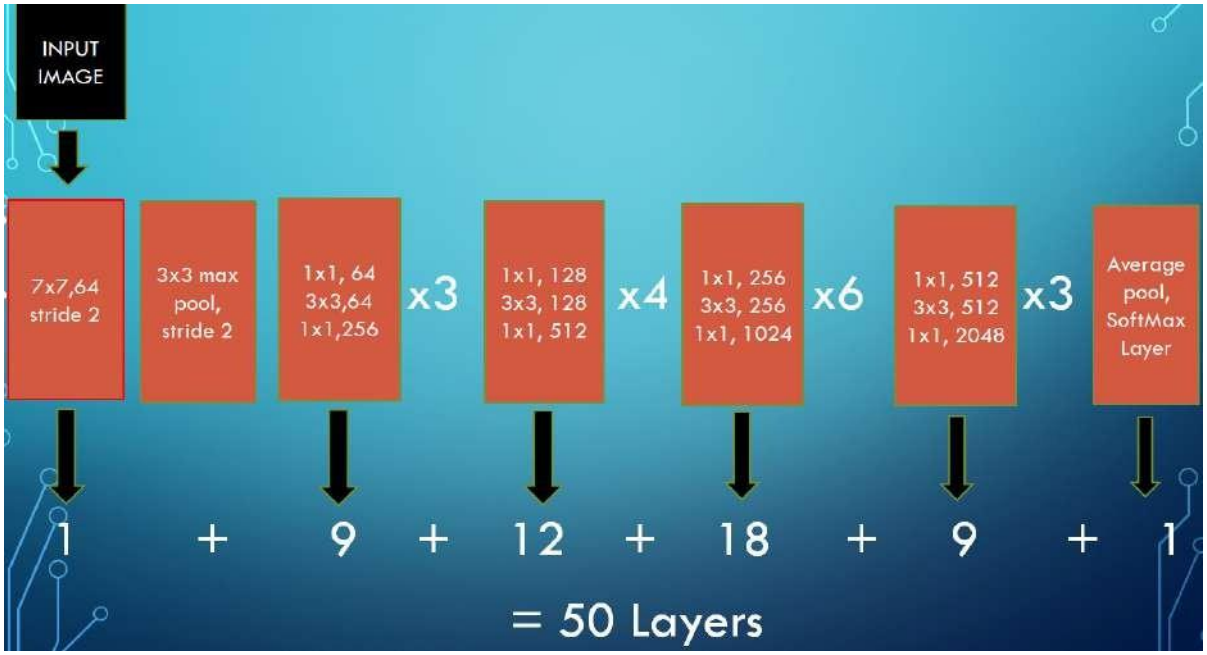

Figure 3.3: ResNet Architecture

These gates control the quantity of data that passes across the skip link. However, in terms of accuracy, this method falls short of ResNet architecture.

3.10.1 Skip Connection In ResNet

In deep neural networks, skip connections are connections that send a layer's output to later levels that are not exactly above or below the layer from which it came.The issue of disappear-ing and inflating gradients grew more and more evident as researchers created neural networks with ever-increasing layers. Because each extra layer is added to the multiplication we must perform to acquire the gradient during backpropagation, disappearing and expanding gradientsare more difficult in deep networks.

The result of multiplying numerous numbers less than 1 together will decrease exponen- tially as you add more terms to the multiplication. If you multiply many terms that are big- ger than 1, the result will grow exponentially with each term you add to the multiplication.

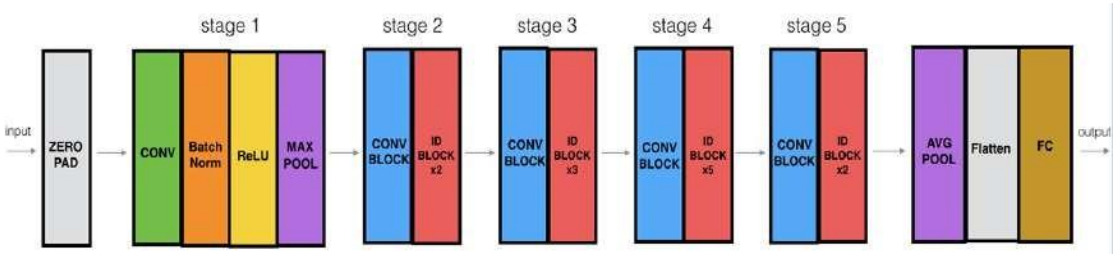

Figure 3.4: Skip Connection

The more layers we have, the more often we must multiply data.Additionally, it has been em- pirically demonstrated that very deep neural networks with conventional topologies typically exhibit performance degradation and accuracy saturation as the network converges.A skip con-nection avoids one or more levels and the activities that go along with them in order to solve these issues.

3.11 Visual Geometry Group(VGG 16)

The benefit of this kind of skip connection is that it enables regularisation to avoid any layer that might reduce architectural performance. As a result, vanishing/exploding gradients are nolonger a problem when training very deep neural networks. Skip connections are also used in "highway networks," which are similar concepts. Parametric gates are used in these skip con- nections, just as they are in LSTM. These gates control the quantity of data that passes across the skip link. However, in terms of accuracy, this method falls short of ResNet architecture. A pooling layer with 7x7x521 nodes, three conv512 levels with 14x14x521 nodes each, and two dense or fully-connected layers with 4090 nodes each follow. The datasets are divided into five categories by a dense or output layer, which has 1000 nodes of varied sizes. In honour of the Oxford Visual Geometry Group, it is often referred to as the OxfordNet model.

- There are 16 layers in all, each with a different weight, as indicated by the number 16.
- Only the Conv and Pooling layers are present.
- A 3 x 3 Kernel should always be used for convolution.

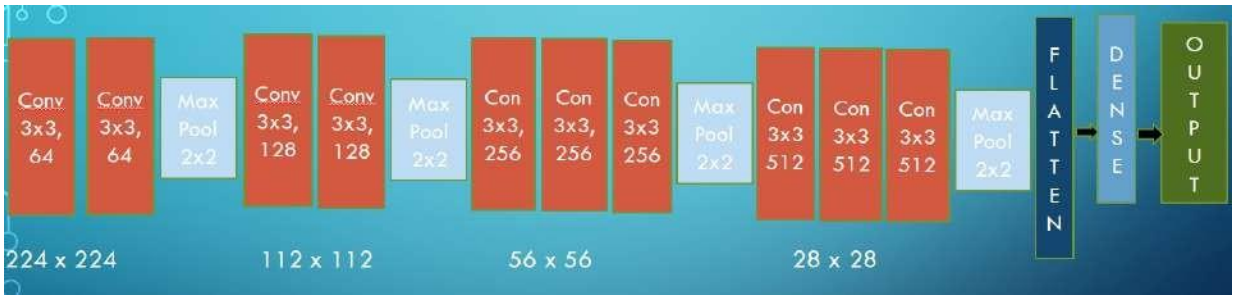

Figure 3.5: Vgg 16 Architecture

- The maximum pool size is 2x2.
- There are around 138 million parameters in all.
- It was trained on ImageNet data and has a 92.7 percent accuracy rate.
- It has a Vgg 19 version with a total of 19 layers and weights.

3.11.1 Limitations Of Vgg 16

- The initial VGG model needed two to three weeks to train on an Nvidia Titan GPU.
- The trained VGG-16 pictureThere is a 528 MB file size for net weights. As a result, it isinefficient because it consumes a lot of bandwidth and disc space.
- The issue of growing gradients is brought on by 138 million parameters.

3.12 Inception V3

The key feature's size within the image frame may vary significantly during an image classifica- tion task. As a result, selecting a kernel size is challenging. Smaller kernels, on the other hand, excel in detecting region-specific properties that are dispersed throughout the image frame. Formore global characteristics that span a large portion of the picture, bigger kernels are preferred.The proper detection of such a variable-sized feature necessitates the use of kernels of diverse sizes. This is done in the film Inception. It increases the layers rather than simply increasingthe number of them. The same layer implements several kernels of various sizes.

Inception expands the network space available for training to select the optimal network. At multiple levels, each inception module can collect key characteristics. While the 3x3 conv layer is more likely to collect dispersed traits, the 5x5 conv layer captures global characteris- tics. The task of collecting low-level features in a neighborhood that stand out belongs to the max-pooling process. All of these characteristics are extracted and concatenated at a specific level before being sent to the next layer. We leave it up to the network/training to determine which attributes have the highest value and weight them appropriately. The trained Inception network

will have much fewer weights than the 5x5 conv kernel if the data set's pictures are abundant in global features but deficient in several low-level qualities.

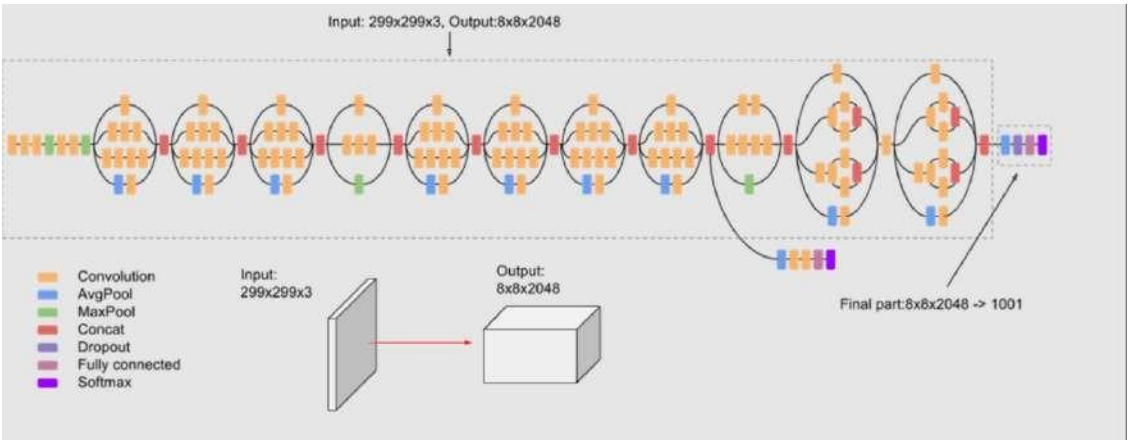

Figure 3.6: Inception V3 Architecture

The Inception v3 model, which was introduced in 2015, differs from its predecessors due to its 42 layers and decreased error rate. Let's examine the several enhancements made by theInception V3 model. The Inception V3 model has undergone significant changes, including

- Reduction to Smaller Convolutions.
- Using Spatial Factorization to Create Asymetric Convolutions.
- Utility of Auxiliary Classifiers.
- Efficient Grid Size Reduction.

3.13 AlexNet

AlexNet was employed for the first time in a public context after winning the ImageNet Large- Scale Visual Recognition Challenge in 2012 (ILSSVRC Competition). A deep convolutional neural network was used by AlexNet to show how to tackle the photo categorization problem in this conflict.

The layers that make up the AlexNet CNN architecture are listed below, along with a briefexplanation of each. Convolution Layer: The productive multiplication of two sets of com- ponents is referred to as convolution in mathematics. Within the convolution layer of deep learning, the convolution process has an influence on the filter/kernel and the picture data ar- ray. As a result, the convolutional layer is essentially a layer in which a convolutional neural network performs a convolutional operation on the filter and the image.

Batch normalisation layer: Batch normalisation is an approach for decreasing the influ- ence of unstable gradients in the neural network by utilising extra layers to perform actions onthe previous layer's inputs. After being normalised and normalised, the input values are alteredusing the resize and move methods. The output of the sub-sampling technique known as max pooling is the pixel with the greatest value that is present in the sub-

sampling layer's receptivefield. The max-pooling method, which is discussed below, slides a 2x2 window across the inputdata to obtain an average of the pixels inside the kernel's receptive field.

Flatten layer:uses an input shape to flatten incoming image data into a one-dimensional array. In a thick layer, any number of units or neurons may be present. One type of neuron is the perceptron.

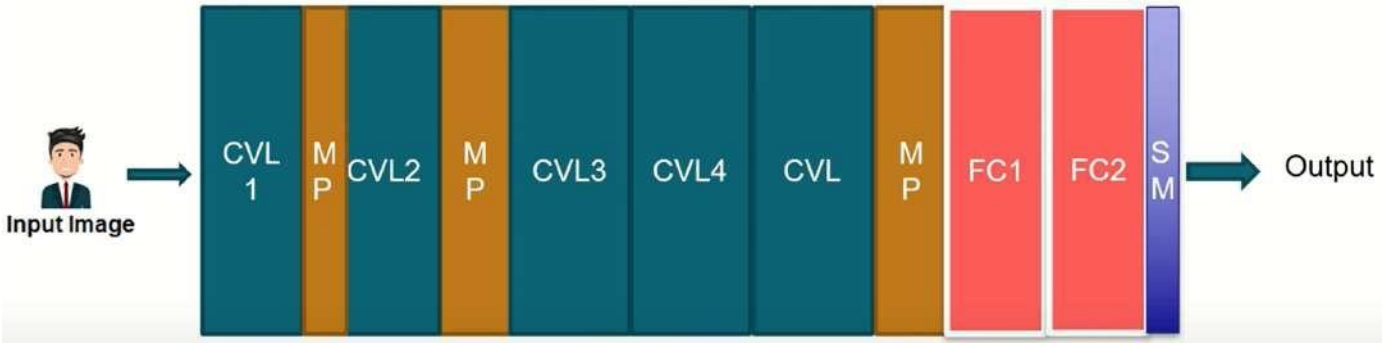

Figure 3.7: AlexNet Architecture

CVL = Convolutional LayersMP = MaxPooling Layers FC = Fully Connected LayerSM = SoftMax Layer

- **Input Layer:** A 224x224x3 RGB image is the input for AlexNet.
- **Convolutional Layer:**A convolutional layer with 96 filters in the first layer has a begin- ning size of 11x11x3, a stride of 4, and padding of 0. A convolutional layer with 256 filters, a dimension of 5x5x48, stride 1, and padding 2 makes up the second layer. The third, fourth, and fifth levels use convolutional layers with 384, 384, and 256 filters of sizes 3x3x256, respectively. There is cushioning of 1 and a stride of 1 in these layers.
- **MaxPooling Layer:** AlexNet's three layers have the maximum pooling. Following the first and second convolutional layers, the first and second pooling layers, with a pool sizeof 3x3 and a stride of 2, respectively. The third pooling layer, with a pool size of 3x3 anda stride of 2, comes after the fifth convolutional layer.
- **Activation Layer:** After every convolutional and fully connected layer, AlexNet em- ploys the Rectified Linear Unit (ReLU) activation function.
- **Dropout Layer:** After the first two fully connected layers, AlexNet conducts dropout regularisation with a probability of 0.5 to prevent overfitting.
- **Fully Connected Layers:**In order to feed two fully linked layers with a combined to- tal of 4096 neurons each, the output of the final pooling layer is flattened into a 4096- dimensional vector. The last layer of the ImageNet dataset is fully connected and consistsof 1000 neurons, or 1000 classes.

- **SoftMax Layer:** For the purpose of determining the probability for each of the 1000 classes. A softmax activation function is applied to the output of the final fully linked layer.

Overall, the AlexNet architecture was a development in computer vision that significantly increased the accuracy of picture identification.

Chapter 4

Results and Discussions

The testing made use of the Python distribution Anaconda, the Google Colaboratory notebook, the Keras library, and the TensorFlow backend engine. Table 2 demonstrates how well different models classified distinct types of white blood cells.

The outcomes of the various methods mentioned above varied. The accuracy of each ap- proach differs. The accuracy variation is primarily due to the dataset's lack of clarity.

Nonetheless, each method has advantages and disadvantages. Every method presented is considerably superior to manual methods, which are time-consuming, demanding, and prone to errors. Some techniques are quite sluggish, which may be fixed by optimizing computer hardware speed or tweaking network configuration and data compression.

$NO	MODEL	ACCURACY
1	CONVOLUTIONAL NEURAL NETWORK(CNN) NEURAL NETWORK	0.94%
2	RESIDUAL NETWORK 50 (RESNET 50)	0.95%
3	VISUAL GEOMETRY GROUP 16 (VGG16)	0.91%
4	INCEPTION V3	0.96%
5	ALEXNET	0.93%

TABLE: Models And Their Accuracy Rates.

Each approach evaluates categorization accuracy based on a unique set of criteria. Along with the number of layers, other factors include the quantity of training sets, testing sets, input size, data label, kernel size in each layer, pooling size in each layer, epoch, activation type, and location in the network. [5]. As a result, the white blood cell classification accuracy is significantly influenced by the quality, labels, quantity, and size of the dataset.

4.1 Results - Screenshots

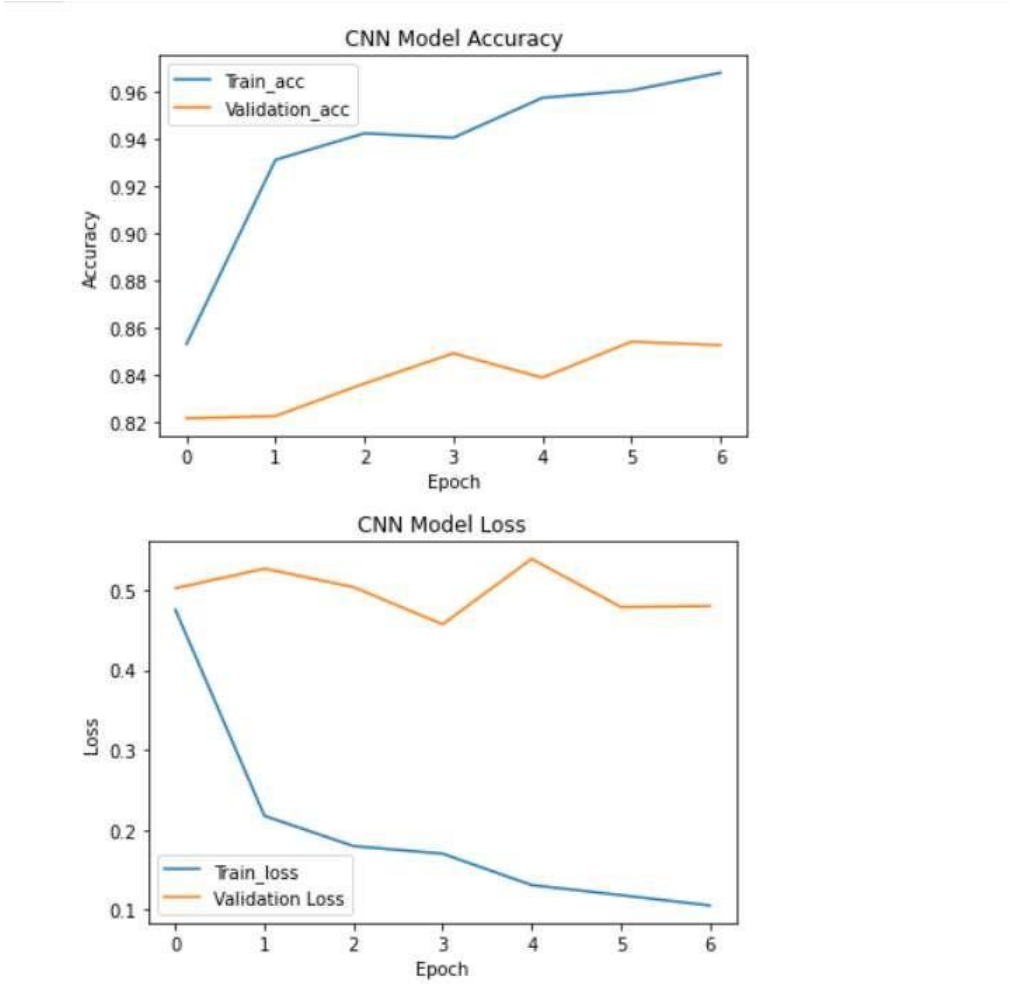

Figure 4.1: CNN Accuracy & Loss Graph

	precision	recall	f1-score	support
Basophil	1.00	1.00	1.00	89
Eosinophil	0.66	0.89	0.76	322
Lymphocyte	0.96	0.94	0.95	1034
Monocyte	0.87	0.77	0.82	234
Neutrophil	0.97	0.95	0.96	2660
accuracy			0.93	4339
macro avg	0.89	0.91	0.90	4339
weighted avg	0.94	0.93	0.93	4339

Figure 4.2: CNN Classification Report

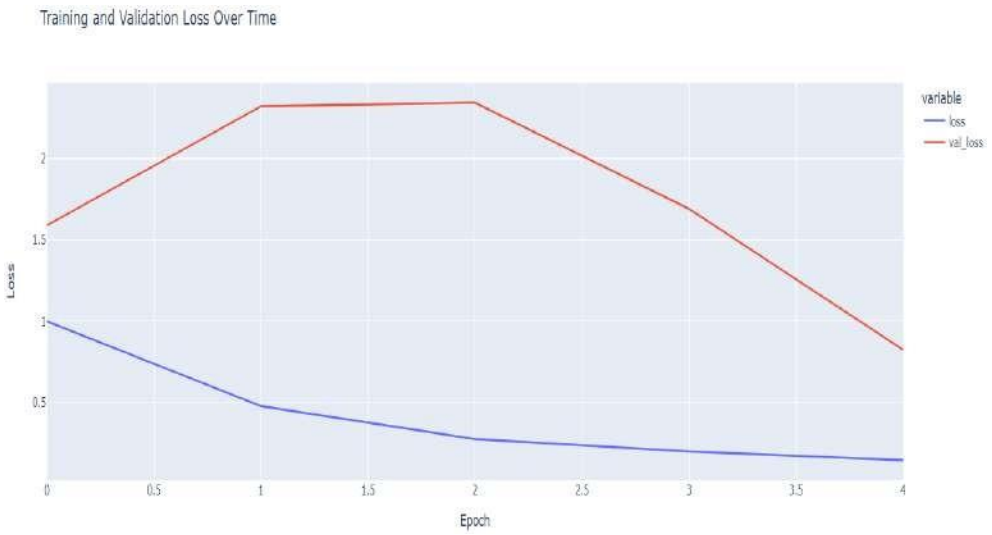

Figure 4.3: Resnet Training Graph

Classification Report:

	precision	recall	f1-score	support
Basophil	0.97	0.82	0.89	89
Eosinophil	0.27	0.44	0.33	322
Lymphocyte	0.42	1.00	0.59	1034
Monocyte	0.95	0.08	0.14	234
Neutrophil	1.00	0.48	0.65	2660
accuracy			0.59	4339
macro avg	0.72	0.56	0.52	4339
weighted avg	0.80	0.59	0.59	4339

Figure 4.4: Resnet 50 Classification Report

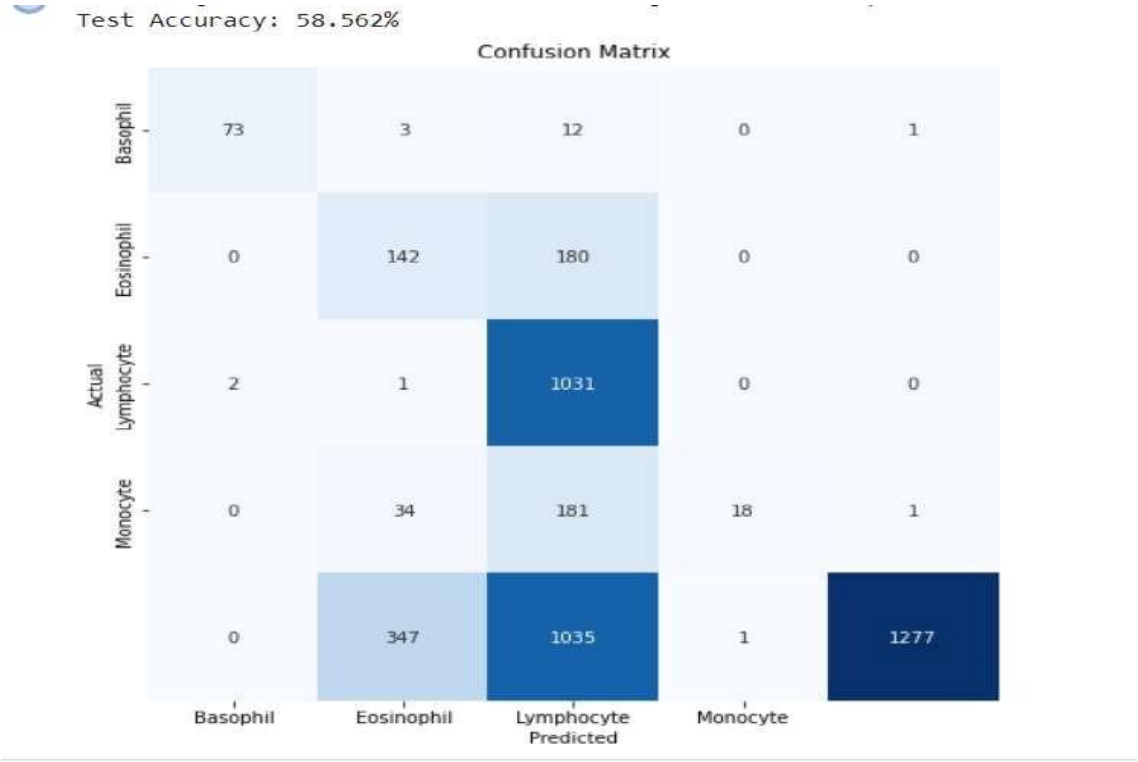

Figure 4.5: Resnet 50 Confusion Matrix

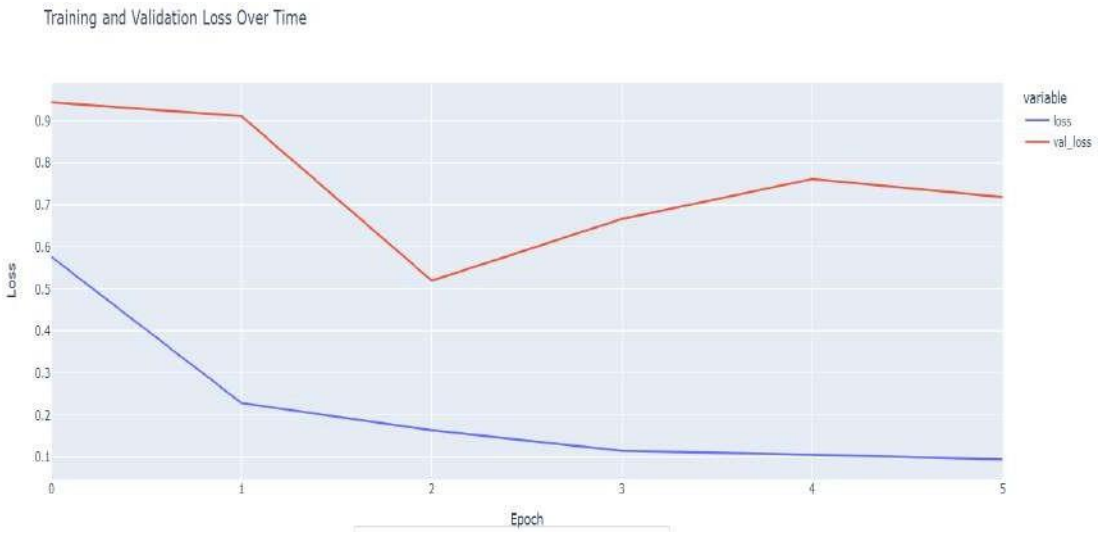

Figure 4.6: Inception V3 Training Graph

Classification Report:

	precision	recall	f1-score	support
Basophil	1.00	0.96	0.98	89
Eosinophil	0.80	0.78	0.79	322
Lymphocyte	0.94	0.96	0.95	1034
Monocyte	0.77	0.84	0.80	234
Neutrophil	0.98	0.96	0.97	2660
accuracy			0.94	4339
macro avg	0.90	0.90	0.90	4339
weighted avg	0.94	0.94	0.94	4339

Figure 4.7: Inception V3 Classification Report

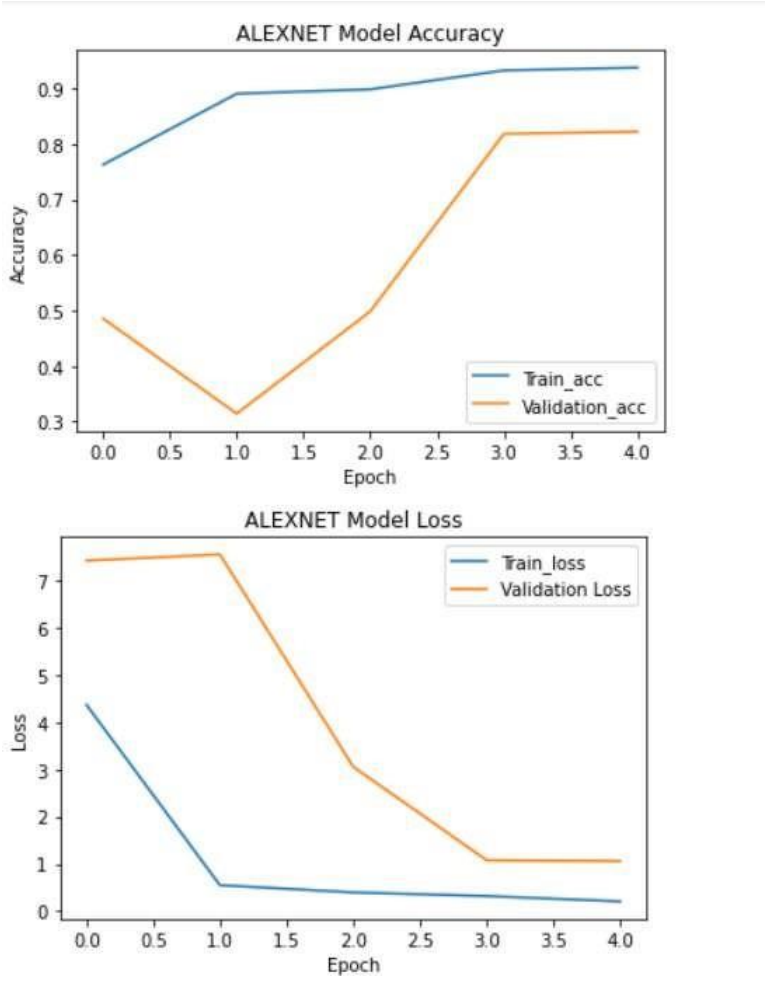

Figure 4.8: Alexnet Accuracy & Loss Graph

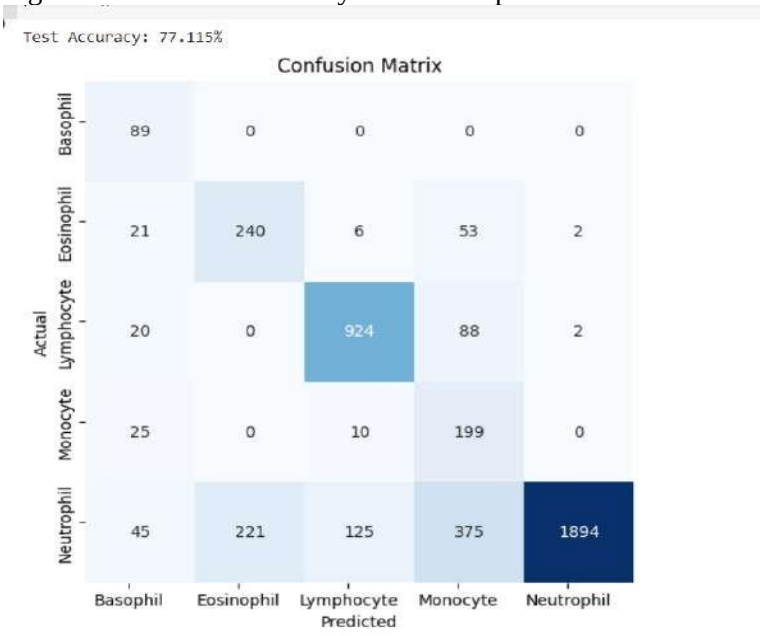

Figure 4.9: Alexnet Confusion Matrix

Chapter 5

Summary and Future Work

The previous few years have seen a lot of interest in deep learning, deep learning models, and deep learning layers. A machine learning technique called deep learning enables computers tocomprehend, absorb, and analyse data in a hierarchical manner. Additionally, artificial neural networks are utilised to mimic the composition and functionality of the brain. Deep learning may operate at numerous layers, which sets it apart from traditional neural networks.

For the diagnosis of diseases, WBC type and count analysis is crucial. WBCs can be recog-nised and counted using a variety of techniques using microscopic pictures of a blood smear. The rapid advancement of machine learning and computer vision has made it possible to solvecomplex categorization problems.

White blood cell identification and categorization are influenced by two factors. The ini- tial factor is the medical unit's dataset, which is followed by the development of each of the methods required to process and categorize the dataset . When compared to traditional lab procedures, the findings of this study aid in the more accurate identification of photographs. The test set was quite accurate, with a 90 percent accuracy rate.When a flawless model is cre- ated and equipped with strong computational skills, it may be applied to medical analysis and applications that deal with WBC quantity and type.

By include red blood cells and blood platelets during training and validation, the suggestedmodel can eventually be generalised even more. To further improve ROC and the efficacy of the

suggested model, several pretrained models and optimisation approaches might also be used. The p-value can also be used.

```
[ ] import numpy as np
    import plotly.express as px
    import matplotlib.pyplot as plt
    import seaborn as sns
    import tensorflow as tf
    from sklearn.metrics import accuracy_score, classification_report
    from keras.callbacks import LearningRateScheduler
    from keras.callbacks import EarlyStopping

[ ] train_dir = '/tmp/WBC_Classifier-main/images/Rabin Data/Train'
    test_dir = '/tmp/WBC_Classifier-main/images/Rabin Data/Test'
```

Figure A.1: Importing libraries & fitting train & test dataset

```
train_images = train_gen.flow_from_directory(
    directory=train_dir,
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
    shuffle=True,
    seed=42,
    subset='training'
)
val_images = train_gen.flow_from_directory(
    directory=train_dir,
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
    shuffle=True,
    seed=42,
    subset='validation'
)

test_images = test_gen.flow_from_directory(
    directory=test_dir,
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
    shuffle=False,
    seed=42
)
```

Figure A.2: Flow Of image data

```
train_gen = tf.keras.preprocessing.image.ImageDataGenerator(
    preprocessing_function=tf.keras.applications.mobilenet_v2.preprocess_input,
    validation_split=0.2
)

test_gen = tf.keras.preprocessing.image.ImageDataGenerator(
    preprocessing_function=tf.keras.applications.mobilenet_v2.preprocess_input
)
```

Figure A.3: Loading the image

```
pretrained_model = tf.keras.applications.MobileNetV2(
    input_shape=(224, 224, 3),
    include_top=False,
    weights='imagenet',
    pooling='avg'
)

pretrained_model.trainable = False
```

Figure A.4: Building Pre-trained Model

```
inputs = pretrained_model.input
x = tf.keras.layers.Dense(128, activation='relu')(pretrained_model.output)
outputs = tf.keras.layers.Dense(5, activation='softmax')(x)

model = tf.keras.Model(inputs=inputs, outputs=outputs)

model.compile(
    optimizer='adam',
    loss='categorical_crossentropy',
    metrics=['accuracy']
)

print(model.summary())
tf.keras.utils.plot_model(
    model,
    to_file="model.png")
```

Figure A.5: Building Classification Model

```
epochs = 25
stepsperepoch=100
validationsteps=1
annealer = LearningRateScheduler(lambda x: 1e-3 * 0.95 ** x)
es = EarlyStopping(monitor='val_loss', mode='max', verbose=1, patience=3)

history = model.fit(
     train_images,
    epochs=epochs,
    callbacks=[annealer,es],
    steps_per_epoch=stepsperepoch,
    validation_data=val_images,
        )
```

Figure A.6: Training

```
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('CNN Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(["Train_acc","Validation_acc"])
plt.show()

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('CNN Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(["Train_loss","Validation Loss"])
plt.show()
```

Figure A.7: Plotting Accuracy & Loss graph

```
from sklearn.metrics import accuracy_score
import seaborn as sns
acc = accuracy_score(test_images.labels, predictions)
cm = tf.math.confusion_matrix(test_images.labels, predictions)
print("Test Accuracy: {:.3f}%".format(acc * 100))

plt.figure(figsize=(8, 8))
sns.heatmap(cm, annot=True, fmt='g', vmin=0, cmap='Blues', cbar=False)
plt.xticks(ticks= np.arange(4) + 0.5, labels=CLASS_NAMES)
plt.yticks(ticks= np.arange(4) + 0.5, labels=CLASS_NAMES)
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
```

Figure A.8: Confusion matrix

A.1 ResNet 50

```
import numpy as np
import plotly.express as px
import matplotlib.pyplot as plt
import seaborn as sns
import tensorflow as tf
from sklearn.metrics import accuracy_score, classification_report
```

```
train_dir = '/tmp/WBC_Classifier-main/images/Rabin Data/Train'
test_dir = '/tmp/WBC_Classifier-main/images/Rabin Data/Test'
```

Figure A.9: Importing Libraries & Fitting train & test dataset

```
train_gen = tf.keras.preprocessing.image.ImageDataGenerator(
    preprocessing_function=tf.keras.applications.mobilenet_v2.preprocess_input,
    validation_split=0.2
)

test_gen = tf.keras.preprocessing.image.ImageDataGenerator(
    preprocessing_function=tf.keras.applications.mobilenet_v2.preprocess_input
)
```

Figure A.10: Creating Generators

```
train_images = train_gen.flow_from_directory(
    directory=train_dir,
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
    shuffle=True,
    seed=42,
    subset='training'
)
val_images = train_gen.flow_from_directory(
    directory=train_dir,
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
    shuffle=True,
    seed=42,
    subset='validation'
)

test_images = test_gen.flow_from_directory(
    directory=test_dir,
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
    shuffle=False,
    seed=42
)
```

Figure A.11: Flow of Image

```
from tensorflow.keras import activations
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras.regularizers import l2
```

```
from tensorflow import keras
# Count total images in training set
count_train = sum([len(files) for r, d, files in os.walk(train_dir)])
count_test = sum([len(files) for r, d, files in os.walk(test_dir)])

img_shape = (224, 224, 3)

model = keras.applications.MobileNet(include_top=True,
    weights=None,
    input_tensor=None,
    input_shape=img_shape,
    pooling='max',
    classes=5, alpha=1, depth_multiplier=2,dropout=.5)

print(model.summary())

lr, epochs = 0.0001, 20
loss = keras.losses.CategoricalCrossentropy()
trainer = keras.optimizers.RMSprop(learning_rate=lr)
steps_per_epoch = count_train//64
validation_steps = count_test//64

save_it= keras.callbacks.ModelCheckpoint(filepath='rabin_a_mobile_v2.epoch{epoch:02d}-loss{val_loss:.2f}.hdf5',monitor='val_loss', verbose=0, save_best_only=True, save_weights_only=Fa

model.compile(loss=loss,optimizer=trainer,metrics=['accuracy'])
my_model = model.fit(train_images, steps_per_epoch=steps_per_epoch, epochs=epochs, validation_data = test_images, validation_steps = validation_steps, verbose=1, callbacks=save_it)
```

Figure A.12: Building & Training the model

```
predictions = np.argmax(model.predict(test_images), axis=1)

acc = accuracy_score(test_images.labels, predictions)
cm = tf.math.confusion_matrix(test_images.labels, predictions)
clr = classification_report(test_images.labels, predictions, target_names=CLASS_NAMES)

print("Test Accuracy: {:.3f}%".format(acc * 100))

plt.figure(figsize=(8, 8))
sns.heatmap(cm, annot=True, fmt='g', vmin=0, cmap='Blues', cbar=False)
plt.xticks(ticks= np.arange(4) + 0.5, labels=CLASS_NAMES)
plt.yticks(ticks= np.arange(4) + 0.5, labels=CLASS_NAMES)
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()

print("Classification Report:\n----------------------\n", clr)
```

Figure A.13: Test Accuracy & classification Report

```
val_images = train_gen.flow_from_directory(
    directory=train_dir,
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
    shuffle=False,
    seed=42,
    subset='validation'
)


predictions = np.argmax(model.predict(val_images), axis=1)

acc = accuracy_score(val_images.labels, predictions)
cm = tf.math.confusion_matrix(val_images.labels, predictions)
clr = classification_report(val_images.labels, predictions, target_names=CLASS_NAMES)

print("Validation Accuracy: {:.3f}%".format(acc * 100))

plt.figure(figsize=(8, 8))
sns.heatmap(cm, annot=True, fmt='g', vmin=0, cmap='Blues', cbar=False)
plt.xticks(ticks= np.arange(4) + 0.5, labels=CLASS_NAMES)
plt.yticks(ticks= np.arange(4) + 0.5, labels=CLASS_NAMES)
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()

print("Classification Report:\n----------------------\n", clr)
```

Figure A.14: Validation Accuracy & Classification Report

References

AC02909904, A. (1987), *International journal of pattern recognition and artificial intelligence*, World Scientific Publishing Company.

Almezhghwi, K. & Serte, S. (2020), 'Improved classification of white blood cells with the generative adversarial network and deep convolutional neural network', *Computational In-telligence and Neuroscience* **2020**.

Huang, X., Jeon, H., Liu, J., Yao, J., Wei, M., Han, W., Chen, J., Sun, L. & Han, J. (2021), 'Deep-learning based label-free classification of activated and inactivated neutrophils for rapid immune state monitoring', *Sensors* **21**(02), 512.

Jung, C., Abuhamad, M., Alikhanov, J., Mohaisen, A., Han, K. & Nyang, D. (2019), 'W-net: a cnn-based architecture for white blood cells image classification', *arXiv preprint arXiv:1910.01091* .

Kouzehkanan, Z. M., Saghari, S., Tavakoli, E., Rostami, P., Abaszadeh, M., Mirzadeh, F., Satl- sar, E. S., Gheidishahran, M., Gorgi, F., Mohammadi, S. et al. (2021), 'Raabin-wbc: a large free access dataset of white blood cells from normal peripheral blood', *bioRxiv* pp. 2021–05.

Kutlu, H., Avci, E. & Özyurt, F. (2020), 'White blood cells detection and classification based on regional convolutional neural networks', *Medical hypotheses* **135**, 109472.

Liang, G., Hong, H., Xie, W. & Zheng, L. (2018), 'Combining convolutional neural network with recursive neural network for blood cell image classification', *IEEE access* **6**, 36188– 36197.

References

Patil, A., Patil, M. & Birajdar, G. (2021), 'White blood cells image classification using deep learning with canonical correlation analysis', *Irbm* **42**(5), 378–389.

Pullanji, V. P. & Muthuswamy, J. (2022), White blood cell image classification using deep learning method, *in* 'AIP Conference Proceedings', Vol. 2494, AIP Publishing LLC, p. 050002.

Tavakoli, S., Ghaffari, A., Kouzehkanan, Z. M. & Hosseini, R. (2021), 'New segmentation and feature extraction algorithm for classification of white blood cells in peripheral smear images', *Scientific Reports* **11**(1), 19428.

Varra, P. (n.d.), 'Classifying white blood cell images using deep learning'.

Zhao, J., Zhang, M., Zhou, Z., Chu, J. & Cao, F. (2017), 'Automatic detection and classificationof leukocytes using convolutional neural networks', *Medical & biological engineering & computing* **55**, 1287–1301.