DEVELOPMENT OF A BLOOD CELL CLASSIFICATION SYSTEM USING CONVOLUTIONAL NEURAL NETWORK

BY

ENECHUKWU RANSOME CHUKWUBUIKEM

(RUN/CMP/20/9564)

A PROJECT SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE, FACULTY OF NATURAL SCIENCES, REDEEMER'S UNIVERSITY, EDE, OSUN STATE, NIGERIA

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF BACHELOR OF SCIENCE (BSc.) IN COMPUTER SCIENCE

AUGUST, 2023

DECLARATION

I Enechukwu Ransome Chukwubuikem (RUN/CMP/20/9564), hereby declare that this project was carried out by me in the Department of Computer Science, Faculty of Natural Sciences, Redeemer's University, Ede, Osun State, Nigeria. To the best of my knowledge, this work has not been presented elsewhere for the award of a degree or any other purpose.

Enechukwu, Ransome Chukwubuikem
…………………………………… ……………………………………
Student *Signature & Date*

CERTIFICATION

We, the undersigned hereby certify that this project was carried out by Adekola, Taofeek Bolaji (RUN/CMP/20/9559) in the Department of Computer Science, Faculty of Natural Sciences, Redeemer's University, Ede, Osun State, Nigeria. To the best of our knowledge, this work has not been presented elsewhere for the award of a degree or any other purpose.

Mrs. T. O. Ojewumi

Supervisor *Signature & Date*

Prof. A. O. Ogunde

Head of Department *Signature & Date*

Prof. C. O. Akanbi

External Examiner *Signature & Date*

DEDICATION

First and foremost, I would be dedicating this project to God Almighty, thanking him for his guidance and grace. To my parents, of blessed memories, who instilled in me the values of hard work, perseverance, and faith, I am forever grateful. Though they are no longer with me, their love and support continue to inspire me. I hope to honor their legacy through my work.

ACKNOWLEDGEMENTS

I would like to express my genuine and deep gratitude to my sisters for their unwavering love and support, which has played a vital role in the success of my program at this esteemed institution. I extend my heartfelt appreciation to my supervisor, Mrs. T. O. Ojewunmi, for dedicating her time, effort, and guidance throughout this project, ensuring that its objectives were met.

I also want to acknowledge the collective efforts of both the academic and non-academic staff of the Department of Computer Science. Starting from the Academic Staff, Dr(Mrs) A. A. Kayode, Dr(Mrs) O. O. Olaniyan, Dr(Mrs) B. O. Oguntunde, Prof. A. O. Ogunde, Dr. M. O. Odim, Prof. S. A. Arekete, Dr. T. A. Olowookere, Mrs Adio, Mr Fagba, Mr Oyetade, Mr Adekunle, Mr Olorunfemi, Mr Victor, Mrs Onifade, Mr Agbolade, Dr. Abolarinwa. Their contributions have immensely enriched my experience during my stay, and I am truly grateful for their invaluable support. To each and every one of them, I extend my sincerest wishes for abundant blessings.

My sincerest thanks goes out to my friend, Daniel Enamudu a.k.a Danikoko, thank you so much brother. For all the late night google meets, and for all the spare time you sacrificed just to help out, i say a very big thank you.

To Adaimoabasi, thank you so much for your input, even with a very tight deadline, you were able to help. May God raise men and women to help you when you need them, Amen.

To Akinyode Ilerioluwa, thank you my friend for your assistance throughout the entirety of this project.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF EQUATIONS

LIST OF FIGURES

ABSTRACT

The human immune system consists of White Blood Cells that are responsible for fighting disease pathogens. In the field of medical imagining, white blood cells are of great importance. Analysis of white blood cells can be helpful to medical experts in many of the cases such as prediction of viral or cancer infection. In this study, Convolutional Neural Network (CNN) model was used to predict the four classes of the white-blood cell sub-types. The white blood cell image datasets used in this study was gotten from Kaggle, and it was split into training, testing and validation sets of 80:10:10, respectively. The datasets were cleaned and preprocessed by resizing and performing feature extraction and data augmentation using convolutional filters and the image data generator of the Keras library. The CNN model was able to classify the sub-types of white-blood cell with 98.1% accuracy, The model achieved an average precision of 95.25%, An average recall of 94.75% was achieved by the model, and an average F1_score of 95% was achieved by the model.

Keywords: Artificial Intelligence, Convolutional neural network (CNN), Deep learning, Prediction, White Blood Cell classification.

CHAPTER ONE

INTRODUCTION

1.1 Background to the Study

The classification of blood cells is a crucial step in elucidating the etiology of various hematological disorders, including but not limited to anemia, leukemia, and lymphoma. The current methodology employed for the classification of blood cells involves the utilization of manual examination conducted by proficient specialists. However, it is imperative to acknowledge that this approach is characterized by its labor-intensive nature, time-consuming requirements, and susceptibility to potential inaccuracies. The potential enhancement of diagnostic precision and efficiency may be realized through the implementation of automated methodologies for the classification of blood cells.

Convolutional neural networks (CNNs) have emerged as a highly promising approach in the realm of automating blood cell classification. Convolutional Neural Networks (CNNs) represent a specific category of deep learning algorithms that have been extensively employed within the realm of medical imaging in order to enhance the accuracy of diagnostic procedures. This is primarily due to their remarkable efficacy in effectively categorizing and classifying various types of medical images (Gandhi, 2019; Wang *et al.,* 2020). By leveraging these algorithms, it becomes possible to automatically extract features from photographs, thereby enabling the generation of predictions. Convolutional neural networks (CNNs) exhibit the remarkable ability to effectively classify various blood cell types, encompassing red blood cells, white blood cells, and platelets. This classification is achieved through the meticulous examination and analysis of digital representations of blood cells. The primary objective of this research endeavor is to investigate the utilization of Convolutional Neural Networks (CNNs) in the domain of blood cell classification, while concurrently evaluating their efficacy vis-à-vis conventional methodologies. The present study utilizes methodologies pertaining to data augmentation and transfer learning as a means to tackle the issue of limited annotated data. The primary objective of this scholarly article is to make a valuable contribution to the advancement of automated systems that are capable of accurately and efficiently classifying blood cells. This will be achieved through an extensive and thorough examination of Convolutional Neural Networks (CNNs) specifically in the context of blood cell classification.

1.2 Statement of the Problem

Convolutional neural networks (CNNs) are one of the most promising methods for automating blood cell classification. Studies on Blood cells classification using CNN are still few in the literature, despite the encouraging outcomes of CNN in other fields of medical imaging (Litjens *et al*., 2017). Additionally, there aren't enough datasets with annotations from medical imaging that can be used to classify blood cells, which makes it difficult to train CNNs as the model is data hungry. The key issue in this area is the dearth of reliable automated methods for classifying blood cells that might enhance the diagnostic procedure by delivering precise and effective results. Major barriers to the development of reliable automated systems for blood cell categorization are the paucity of this study. Hence, this study proposed an improved computation system for blood cell image classification that can assist the medical practitioners in their diagnosis process.

1.3 Aim and Objectives of the Study

Aim:

The aim of this study is to classify white-blood cells images into neutrophil, monocyte, lymphocyte, and eosinophil using Convolutional Neural Network (CNN).

Objectives:

i. To collate and preprocess blood cells dataset.
ii. Train the CNN model for blood cell classification using annotated medical imaging datasets.
iii. Evaluate CNNs for blood cell classification.
iv. Implement the blood cell classifier system.

1.4 Overview of Methodology

Data collection: The white blood cell dataset was gotten from Kaggle.com. Kaggle is an online repository for stored data. The tools used in this project were Skicit-learn, Jupyter Notebook, and Python. The environment which would be used in Anaconda Navigator, which launched python workspace without having to use too many command lines. The deep learning technique used in this project was the Convolutional neural network (CNN).

Data Cleaning and Preprocessing: Firstly, the process of data cleaning entails the elimination of any extraneous or disruptive data from the dataset. This process entails the removal of duplicate samples, the management of missing values, and the resolution of inconsistent or erroneous entries. Through the process of data cleansing, we enhance the integrity of the dataset and mitigate any potential biases or distortions that may arise (Jones *et al.*, 2018).

Convolutional Neural Network: Convolutional Neural Networks (CNNs) are deep learning architectures that analyze structured grid data like pictures and time series. CNNs have transformed computer vision, enabling picture categorization, object recognition, image synthesis, and more. CNNs use convolutional layers specialized filters or kernels to automatically learn and extract meaningful characteristics from input data. These features capture patterns, textures, edges, and other visual properties necessary for the task. Pooling layers and fully connected layers boost CNNs.

1.5 Scope of the Study

The goal of this study is to classify white-blood cell images into four different categories which are Eosinophil, Lymphocyte, Monocyte, and Neutrophil

1.6 Significance of the Study

The significance of this study lies in its potential to augment the diagnostic capabilities pertaining to a range of hematological disorders, encompassing anemia, leukemia, and lymphoma, through the introduction of precise and efficacious automated techniques for the classification of white blood cells. Convolutional neural networks (CNNs) possess the inherent capacity to augment diagnostic accuracy while concurrently reducing the temporal and cognitive demands associated with manual scrutiny performed by proficient practitioners. Moreover, this study delves into the exploration of techniques pertaining to data augmentation and transfer learning as potential solutions to address the challenge of limited annotated data for blood cell classification employing Convolutional Neural Networks (CNNs). The aforementioned strategies possess the capability to augment the performance of Convolutional Neural Networks (CNNs) and facilitate the training of models using limited annotated datasets. This is particularly crucial in the realm of automated medical methodologies, where the availability of data is often restricted.

In its entirety, this study possesses the capacity to propel the diagnosis of various hematological disorders and propel the advancement of accurate and efficient automated methodologies for convolutional neural network-based categorization of leukocytes.

1.7 Definition of Terms

Blood cell classification: The phenomenon of discerning and categorizing diverse blood cell varieties, encompassing platelets, leukocytes, and erythrocytes, is commonly referred to as blood cell classification.

Convolutional neural networks (CNNs): represent a distinct category of deep learning techniques characterized by the incorporation of numerous convolutional and pooling layers, rendering them particularly efficacious in the realm of image classification endeavors.

Data augmentation: refers to a technique employed in the realm of data science, wherein the size of a given dataset is expanded through the artificial manipulation of its constituent elements. This process entails the application of various transformations, such as rotation, flipping, and scaling, to the original images, thereby introducing random modifications.

Transfer learning: is a technique employed in machine learning that leverages knowledge gained from one task to improve the performance of a model on another task that is closely related.

Annotated dataset: A set of data that has been annotated with details about the relevant items or features, like the position and nature of blood cells in a picture.

Metrics: Quantitative metrics including accuracy, precision, recall, and F1-score are used to assess a model's performance.

Deep Learning: A branch of machine learning where complicated interactions between inputs and outputs can be modeled using artificial neural networks with numerous layers.

1.8 DATASET

Convolutional neural networks (CNNs) can classify blood cells using a variety of publicly available datasets. The "Blood Cell Images Dataset" created by Kermany *et al.* in 2018 is one frequently used dataset. On the website of the National Library of Medicine of the National Institutes of Health (https://www.kaggle.com/paultimothymooney/blood-cells), you may download a dataset with over 27,000 photos of blood cells, including red blood cells, white blood cells, and platelets. The dataset has been proven to be helpful for training and assessing CNN models and has been utilized in various studies on classifying blood cells using CNNs. D. Kermany *et al.*, 2018. using image-based deep learning to identify medical diagnoses and diseases that are curable. (Preprint at arXiv is arXiv:1801.02765).

1.9 EVALUATION

Accuracy, precision, recall, and F1 score are standard metrics used to assess a convolutional neural networks (CNN) effectiveness while categorizing blood cells.

Accuracy: is a frequently employed statistic for assessing how well a classification model performs. It is the percentage of instances that were correctly categorized to all instances.

Precision: is a metric that quantifies the proportion of correct positive predictions relative to the total number of positive predictions made. High precision is indicative of a low occurrence of false positives being produced by the model.

Recall, in the context of classification evaluation: refers to a quantitative measure that assesses the proportion of actual positive instances correctly identified as such, relative to the expected number of true positive instances. The concept of high recall entails that the model is proficient in accurately identifying a significant majority of positive examples.

F1 score: is a commonly employed metric for evaluating the efficacy of binary classification algorithms. The harmonic mean of precision and recall is computed.

The inclusion of an evaluation measure that incorporates supplementary contextual information, such as the confusion matrix, was duly considered.

It is imperative to bear in mind that, notwithstanding the potential of these metrics to provide a general understanding of the model's efficacy, the ultimate evaluation of the model's performance must be conducted via clinical validation and real-world data testing.

CHAPTER TWO

REVIEW OF LITERATURE

2.0 Introduction

Blood cells are the cells that circulate around the body in the blood and carry out a number of functions, such as supplying the body's cells with nutrients and oxygen and removing waste products (Wang *et al*., 2020). The three main subtypes of blood cells are red blood cells, white blood cells, and platelets (Litjens *et al*., 2017).

Hemoglobin, also known as the erythrocytes found in red blood cells, is in charge of carrying oxygen from the lungs to the body's cells. Hemoglobin, a protein that binds to oxygen, is what gives red blood cells their unique red color (Wang *et al*., 2020). White blood cells, also known as leukocytes, are in charge of fighting off disease and infection. Each of their several subtypes accomplishes a specific task, such as engulfing and destroying viruses and bacteria (Litjens *et al*., 2017). Platelets, also known as thrombocytes, are responsible for blood clotting and blood vessel healing (Wang *et al*., 2020).

Blood cell classification can be used to identify and treat a variety of illnesses and diseases, including anemia, leukemia, and thrombocytopenia (Litjens *et al*., 2017). Traditional methods of identifying blood cells involve manual examination by knowledgeable specialists, but this can be time-consuming and prone to human error (Wang *et al*., 2020). Since technology improvements have made it possible to quickly and accurately classify blood cells, convolutional neural networks (CNNs) are a beneficial tool for medical professionals (Litjens *et al*., 2017).

2.1 History of blood cells

The study of blood cells, or hematology, has a long history that goes all the way back to the beginning of time. For instance, according to Dacie and Lewis (2001), the ancient Egyptians believed that blood was tied to life and wellbeing and knew its importance in the human body. Because of the efforts of scientists and medical professionals, we now have a far better understanding of blood cells and how they function.

A physician from England named William Harvey wrote in the 17th century about the heart's function in pumping blood throughout the body (Harvey, 1628). German physician Rudolf Virchow laid the foundation for the modern study of hematology by identifying the three main types of blood cells and defining their roles in the 19th century (Virchow, 1858).

It became feasible to study blood cells in greater depth in the 20th century thanks to the development of new technologies like the microscope and the centrifuge. These advancements led to the discovery of new blood cell types and the identification of new blood disorders. For instance, Australian scientist Macfarlane Burnet and English scientist Peter Medawar developed the concept of immunological tolerance in the 1940s, laying the foundation for the development of immunotherapy treatments for cancer and other diseases (Burnet & Medawar, 1957).

Thanks to technological breakthroughs like the use of genetic sequencing and flow cytometry, the research of blood cells and their functions is still advancing in the present.

2.1.1 Types of blood cells

There are three main types of blood cells:

1. **Red blood cells (erythrocytes)**, which transport oxygen from the lungs to the body's tissues and remove carbon dioxide.
2. **White blood cells (leukocytes)**, which help to fight infections and protect the body against foreign substances.
3. **Platelets (thrombocytes)**, which are involved in the formation of blood clots and the repair of damaged blood vessels.

2.1.2 Complications from blood cells

According to National Cancer Institute. (2019). Complications from blood cells can include:

1. Anemia, which occurs when there are not enough red blood cells to supply the body's tissues with oxygen.
2. Leukemia, a type of cancer that affects white blood cells.
3. Thrombocytopenia, a condition in which there are not enough platelets in the blood, which can lead to bleeding.
4. Thrombosis, a condition in which a blood clot forms within a blood vessel, which can block the flow of blood and lead to serious complications such as heart attack or stroke.

2.1.3 Temporary treatments for blood cells

According to American Society of Hematology. (2019) Temporary treatments for blood cell disorders can include:

1. Blood transfusions, which can be used to treat anemia by increasing the number of red blood cells in the body.

2. Platelet transfusions, which can be used to treat thrombocytopenia and other bleeding disorders by increasing the number of platelets in the blood.
3. Leukemia treatment with chemotherapy, which can help to reduce the number of cancerous white blood cells in the body.
4. Anticoagulant medications, such as heparin or warfarin, which can be used to prevent blood clots from forming or to treat existing clots.

2.2 Artificial intelligence

According to Russell, S., & Norvig, P. (2019). Artificial intelligence (AI) is a field of computer science that aims to create machines that can perform tasks that would typically require human intelligence, such as understanding natural language, recognizing patterns in data, and making decisions. There are various types of AI, including:

1. Reactive machines, which can only react to the present and cannot use past experiences to inform future decisions.
2. Limited memory, which can use past experiences to inform current decisions but cannot form long-term memories.
3. Theory of mind, which can understand the mental states of other agents and can use this understanding to inform its own decisions.
4. Self-aware, which has a sense of self and can understand its own mental states.

2.2.1 Types of Artificial Intelligence

According to Russell, S., & Norvig, P. (2019). There are several types of Artificial Intelligence (AI) which are broadly categorized into following types:

Reactive Machines: AI systems that can only react to the present and cannot use past experiences to inform future decisions.

Limited Memory: AI systems that can use past experiences to inform current decisions but cannot form long-term memories.

Theory of Mind: AI systems that can understand the mental states of other agents and can use this understanding to inform its own decisions.

Self-Aware: AI systems that have a sense of self and can understand its own mental states**.**

Weak AI: AI systems that are designed to perform specific tasks.

Strong AI: AI systems that are capable of performing any intellectual task that a human can.

2.2.1.1 Reactive Machines

Reactive machines are a type of intelligent machine that can perceive changes in their environment and react to those changes in real-time. According to Russel & Norvig (2010), reactive machines are designed to act solely on the basis of their current inputs, without the need for internal representations of the world. These machines are commonly used in robotics, where they are used to control the movement of robots and ensure they are operating safely in their environment.

One example of a reactive machine is the Roomba vacuum cleaner, which uses a combination of sensors and algorithms to navigate around a room and clean the floor. The Roomba does not have a map of the room and does not plan its movements in advance. Instead, it reacts to obstacles and changes in the environment as they occur.

Reactive machines are different from other types of intelligent machines, such as deliberative machines, which make decisions based on an internal model of the world. Deliberative machines use this model to plan their actions and make decisions based on their long-term goals. In contrast, reactive machines are focused on the immediate present and do not have long-term goals (Russel & Norvig, 2010).

In conclusion, reactive machines are an important type of intelligent machine that can react to changes in their environment in real-time without the need for internal models of the world. They are commonly used in robotics and can be seen in consumer products such as the Roomba vacuum cleaner.

2.2.1.2 Limited Memory AI

Limited Memory AI is a variant of artificial intelligence that leverages prior encounters to facilitate decision-making and forecasting, while abstaining from the retention of an exhaustive chronicle of past occurrences. As per the findings of Silver *et al*. (2016), the utilization of limited memory artificial intelligence (AI) is prevalent in scenarios where the storage of an entire chronicle of events is impractical, exemplified by intricate games like Go or poker.

The available memory is constrained. AI algorithms employ a methodology known as reinforcement learning, wherein the machine acquires knowledge through iterative experimentation. The computational system engages in a process of trial and error, exploring different actions and assimilating knowledge from the feedback it receives, which is manifested as rewards or penalties. The AI algorithm with constrained memory utilizes the provided data to inform its future decision-making process.

An instance of artificial intelligence with restricted memory capacity is the AlphaGo software, which was created by DeepMind Technologies. In 2016, AlphaGo emerged victorious over the reigning world champion of the game Go, employing a fusion of reinforcement learning and neural networks to derive decisions by leveraging historical encounters (Silver *et al.,* 2016).

Limited memory AI, in contrast to rule-based systems or expert systems, operates distinctively by storing only a constrained number of past events and employing this data to facilitate decision-making processes. Limited memory artificial intelligence (AI) exhibits great utility in scenarios characterized by intricate and dynamic environments. This form of AI empowers machines to swiftly adjust to novel circumstances and render real-time decisions by leveraging their prior encounters.

In summary, limited memory AI constitutes a significant subset of artificial intelligence wherein decision-making and predictive capabilities are derived from past experiences, all while refraining from the storage of a comprehensive record of historical events. It is especially advantageous in intricate and ever-changing environments where it is impractical to retain a comprehensive chronicle of occurrences.

2.2.1.3 Theory of Mind

Theory of Mind AI is a computational paradigm that encompasses a subset of artificial intelligence techniques, enabling the cognitive capacity to comprehend and interpret the internal cognitive processes of external entities, including but not limited to beliefs, intentions, and desires. As per the seminal work of Premack and Woodruff in 1978, the concept of theory of mind pertains to the cognitive capacity of ascribing mental states to oneself and others. It encompasses the comprehension that individuals possess distinct beliefs, desires, and intentions that may diverge from one's own.

The incorporation of Theory of Mind AI holds significant importance within various domains, particularly in the realm of natural language processing. This is due to the necessity of enabling machines to comprehend the underlying intended semantics conveyed through human language. It is additionally employed in the field of robotics, wherein the machine necessitates comprehending the intentions and objectives of humans to engage with them in a purposeful manner.

An instance of Theory of Mind AI can be observed in the Affective Computing group at the Massachusetts Institute of Technology. This particular group is actively engaged in the advancement of machines that possess the capability to comprehend and react to human emotions (Picard, 2003). These computational devices employ sensory mechanisms to perceive alterations in facial expressions, bodily gestures, and vocal intonation, thereby deducing the emotional disposition of the human interactor.

The concept of Theory of Mind AI distinguishes it from other AI paradigms, such as limited memory AI or rule-based systems, as it encompasses the capacity to comprehend the cognitive states of external agents. The concept of Theory of Mind AI represents a significant advancement in the pursuit of developing machines that possess a higher degree of human-like qualities, enabling them to engage in interactions with humans in a manner that is both organic and instinctive.

In summary, the Theory of Mind AI, short for Artificial Intelligence, is a significant facet of computational intelligence that exhibits the ability to comprehend the cognitive states of alternative entities, encompassing beliefs, intentions, and desires. It possesses applications within the realm of natural language processing, robotics, and affective computing, thereby constituting a significant stride towards the realization of machines that exhibit greater resemblance to humans.

2.2.1.4 Self-Aware AI

Self-aware AI refers to a theoretical form of artificial intelligence that possesses the ability to engage in introspection and exhibit consciousness. This advanced AI system possesses an awareness of its own existence and is cognizant of its own actions. As per Metzinger (2013), self-awareness denotes the capacity to introspect upon one's own cognitive states, encompassing beliefs, desires, and emotions, while concurrently possessing a comprehension of one's distinct identity and capacity for intentional action.

The subject of self-aware AI is a matter of extensive deliberation and conjecture within the realm of artificial intelligence. Numerous researchers contend that attaining genuine self-awareness through existing AI methodologies is improbable (Dennett, 2017). Nevertheless, a group of researchers is currently investigating the feasibility of developing machines with the ability to simulate self-awareness and introspection.

An instance of research in self-aware artificial intelligence (AI) can be found in the endeavors of Selmer Bringsjord and his associates at Rensselaer Polytechnic Institute. They are currently engrossed in the creation of a machine named "Rex" that possesses the capability of self-awareness (Bringsjord *et al.*, 2012). Rex employs a fusion of logical deduction and introspective analysis to emulate the cognitive states of a sentient entity, and possesses the ability to respond to inquiries regarding its own cognitive constructs, volitions, and affective experiences.

Self-aware artificial intelligence (AI) exhibits distinct characteristics when compared to other AI variants, namely limited memory AI or rule-based systems. These alternative AI models lack the capability to introspect upon their own cognitive states, possess a sense of individuality, or demonstrate agency. The exploration of self-aware AI is an area of research that is characterized by its speculative nature. However, it possesses the potential to significantly impact the trajectory of artificial intelligence and our comprehension of consciousness and self-awareness.

In summary, self-aware AI refers to a theoretical form of artificial intelligence that possesses the ability to engage in introspection and exhibit consciousness. It demonstrates an awareness of its own existence and actions. Although the current techniques make it improbable to create genuine self-aware artificial intelligence, researchers are actively investigating the potential of simulating self-awareness and introspection in machines.

2.2.1.5 Weak AI

Weak AI, or narrow AI, denotes artificial intelligence that has been specifically engineered for particular tasks or domains, lacking in general intelligence or consciousness. As per Russell and Norvig (2010), weak artificial intelligence (AI) pertains to the capacity of a machine to execute a specific task at levels of performance comparable to humans, while lacking human-like intelligence or consciousness.

Instances of weak artificial intelligence (AI) encompass systems that have been meticulously crafted to discern speech or images, engage in strategic games such as chess or Go, or provide support in domains such as customer service or data analysis. These systems exhibit a high degree of specialization and are meticulously engineered to execute a well-defined task or a collection of tasks. However, they lack the capacity to acquire knowledge or engage in logical reasoning beyond the confines of their designated domain.

Weak AI and strong AI are distinct concepts within the field of artificial intelligence. Strong AI specifically pertains to the development of intelligent systems that possess general intelligence and consciousness. These advanced systems exhibit the capacity to learn, reason, and apply their knowledge across various domains. In contrast, weak

AI refers to artificial intelligence that is limited in scope and functionality, lacking the comprehensive cognitive abilities associated with strong AI. Strong artificial intelligence (AI) remains a profoundly speculative domain of inquiry, and its realization using contemporary methodologies is presently unattainable.

Weak artificial intelligence (AI) exhibits numerous pragmatic implementations across various domains, including healthcare, finance, and transportation. Its pervasiveness in our everyday existence is progressively escalating. Nevertheless, there exist apprehensions regarding the ramifications of weak artificial intelligence (AI) on employment, privacy, and security. Additionally, ethical considerations arise from the development of machines capable of executing tasks conventionally undertaken by human beings.

In summary, weak AI pertains to artificial intelligence that is purposefully crafted for particular tasks or domains, lacking the capability of general intelligence or consciousness. While weak artificial intelligence (AI) exhibits numerous pragmatic implementations, it concurrently elicits apprehensions regarding its ramifications on employment, privacy, and ethics.

2.2.1.6 Strong AI

Strong AI, colloquially referred to as general AI or artificial general intelligence (AGI), pertains to the theoretical notion of artificial intelligence possessing cognitive capabilities and self-awareness comparable to that of humans. It encompasses the capacity to acquire knowledge and engage in logical thinking across various fields of study. As per the scholarly work of Russell & Norvig (2010), the concept of strong AI pertains to the capacity of a computational system to execute any cognitive undertaking achievable by a human being.

The notion of strong artificial intelligence (AI) is predominantly theoretical, and the realization of genuine strong AI continues to pose a substantial obstacle for researchers. One potential strategy for attaining robust artificial intelligence involves the advancement of artificial neural networks and deep learning algorithms, which endeavor to emulate the architecture and operation of the human brain (LeCun, Bengio, & Hinton, 2015).

The advancement of robust artificial intelligence (AI) possesses the capability to fundamentally transform domains including healthcare, finance, and transportation, thereby exerting a substantial influence on the fabric of human society. Nevertheless, one must also acknowledge the apprehensions regarding the ethical ramifications entailed in fabricating machines endowed with human-like cognitive abilities and awareness, as well as the conceivable hazards linked to said technological advancements.

In summary, strong AI pertains to the theoretical notion of artificial intelligence possessing intelligence and consciousness at a level equivalent to that of humans. It also encompasses the capacity to acquire knowledge and engage in logical thinking across various fields. Although the endeavor to create robust artificial intelligence (AI) persists as a formidable undertaking for researchers, its implications possess the capability to profoundly transform numerous domains and elicit crucial ethical deliberations.

2.2.2 Advantages of Artificial Intelligence

The ensuing are the benefits of Artificial Intelligence;

1. Enhanced efficiency can be achieved through the utilization of AI, which facilitates the automation of various tasks and processes. For example,

chatbots powered by artificial intelligence have the capability to promptly respond to customer inquiries, thereby enhancing the efficiency of customer service (Vogels, 2018).

2. Enhanced Decision Making: Artificial intelligence possesses the capability to process and analyze extensive volumes of data, thereby facilitating the extraction of insights that may elude human cognition. According to the McKinsey Global Institute in 2018, this has the potential to assist businesses in making well-informed decisions.
3. Personalization: Through the utilization of artificial intelligence, the analysis of consumer data can be performed, resulting in the provision of personalized recommendations and experiences. This, in turn, has the potential to enhance customer satisfaction and foster loyalty (Larivière *et al.,* 2017).
4. Healthcare Advancements: The utilization of artificial intelligence (AI) has the potential to enhance the process of diagnosing medical conditions and formulating treatment strategies, thereby resulting in improved health outcomes for individuals (Obermeyer *et al*., 2019).
5. Enhanced Safety: Artificial Intelligence (AI) possesses the potential to augment safety measures across diverse sectors, including but not limited to manufacturing and transportation. For example, artificial intelligence (AI) has the capability to identify equipment malfunctions in advance, thereby mitigating the risk of accidents (Deloitte, 2019).

2.2.3 Disadvantages of artificial intelligence

The subsequent are the drawbacks of artificial intelligence:

1. The advent of artificial intelligence (AI) harbors the capability to mechanize a multitude of occupations, thereby engendering the possibility of job

displacement for laborers within specific sectors (Acemoglu & Restrepo, 2020).

2. Bias: The impartiality of AI algorithms is contingent upon the impartiality of the data they undergo training on. In the event that the data is tainted with bias, the algorithm has the potential to sustain and propagate said bias (Crawford, 2016).
3. Privacy concerns are a significant issue in the realm of artificial intelligence (AI). AI systems typically depend on extensive datasets to operate effectively, thereby giving rise to apprehensions regarding privacy and the safeguarding of data (Choi, 2018).
4. The absence of accountability arises due to the inherent complexities and opaqueness of AI systems, rendering it arduous to ascertain the individual or entity held responsible in instances of malfunction or error (Burrell, 2016).
5. Security risks: Artificial intelligence (AI) systems exhibit susceptibility to hacking and cyber-attacks, thereby potentially leading to grave ramifications when employed in critical infrastructure or military domains (Lee & Kim, 2017).

2.2.4 Artificial intelligence applications

The subsequent are the implementations of artificial intelligence;

1. Natural Language Processing (NLP) refers to the utilization of artificial intelligence (AI) techniques for the purpose of analyzing, comprehending, and generating human language. It finds utility in various applications such as chatbots, virtual assistants, and speech recognition systems. According to the work of Jurafsky & Martin (2019).

2. Computer Vision: The field of computer vision encompasses the utilization of artificial intelligence techniques for the purpose of analyzing and interpreting images and videos. It finds utility in various domains such as autonomous vehicles, facial identification, and surveillance systems. (Krizhevsky, Sutskever, & Hinton, 2012) is a citation in the format of author names followed by the publication year.
3. Recommendation Systems: Recommendation systems are employed to propose products or services to users predicated on their preferences and behavior. Artificial intelligence (AI) is employed for the purpose of analyzing user data and producing personalized recommendations. According to the research conducted by (Koren, Bell, & Volinsky, 2009).
4. Predictive Maintenance: The application of artificial intelligence is employed to anticipate the occurrence of machinery or equipment failure, thereby enabling proactive maintenance actions to be executed prior to any potential breakdown. It is utilized in various domains such as manufacturing and transportation. (Saha, 2018)
5. Fraud Detection: Artificial intelligence (AI) is leveraged to analyze patterns and anomalies within financial transactions with the aim of identifying instances of fraudulent activity. This is utilized in applications such as banking and insurance. The citation (Chen & Zhang, 2018) is referenced.
6. Sentiment Analysis: Sentiment analysis encompasses the utilization of artificial intelligence to scrutinize textual data and ascertain the sentiment or viewpoint of the author. It finds utility in various applications such as social media monitoring and customer feedback analysis. According to the research conducted by (Pang & Lee, 2008).

7. Robotics: Robotics encompasses the utilization of artificial intelligence (AI) for the purpose of governing robots and additional self-governing systems. It finds utility in domains such as manufacturing, healthcare, and space exploration. According to the seminal work by Russell & Norvig (2010).

2.3 Machine Learning

Machine learning is a subfield within the realm of artificial intelligence, wherein algorithms and statistical models are employed to facilitate the automatic enhancement of computer systems' performance on a given task through iterative learning from data inputs. It entails instructing computational systems to discern patterns within datasets and generate forecasts or determinations without explicit programming instructions.

As per Alpaydin's (2010) classification, machine learning can be categorized into three primary types: supervised learning, unsupervised learning, and reinforcement learning. In the realm of supervised learning, a machine learning algorithm undergoes training using a dataset that has been appropriately labelled. This training enables the algorithm to generate predictions or classifications when presented with novel, previously unseen data. Unsupervised learning is a computational approach that leverages an unannotated dataset to discern inherent patterns or clusters within the given data. Reinforcement learning is a paradigm within the field of machine learning wherein an algorithm acquires the ability to make decisions by leveraging feedback obtained from its surrounding environment.

Machine learning, a subfield of artificial intelligence, exhibits extensive utility across diverse domains. These domains encompass but are not limited to image recognition, natural language processing, and predictive analytics. It is an exponentially expanding domain with the capacity to fundamentally transform numerous sectors.

2.3.1 Types of Machine Learning

There exist four primary classifications of machine learning, as stated by Burns (2021). The selection of the appropriate approach is contingent upon the intended task to be executed. The four primary methodologies encompass;

1. Supervised learning
2. Unsupervised learning
3. Semi-Supervised learning
4. Reinforcement learning

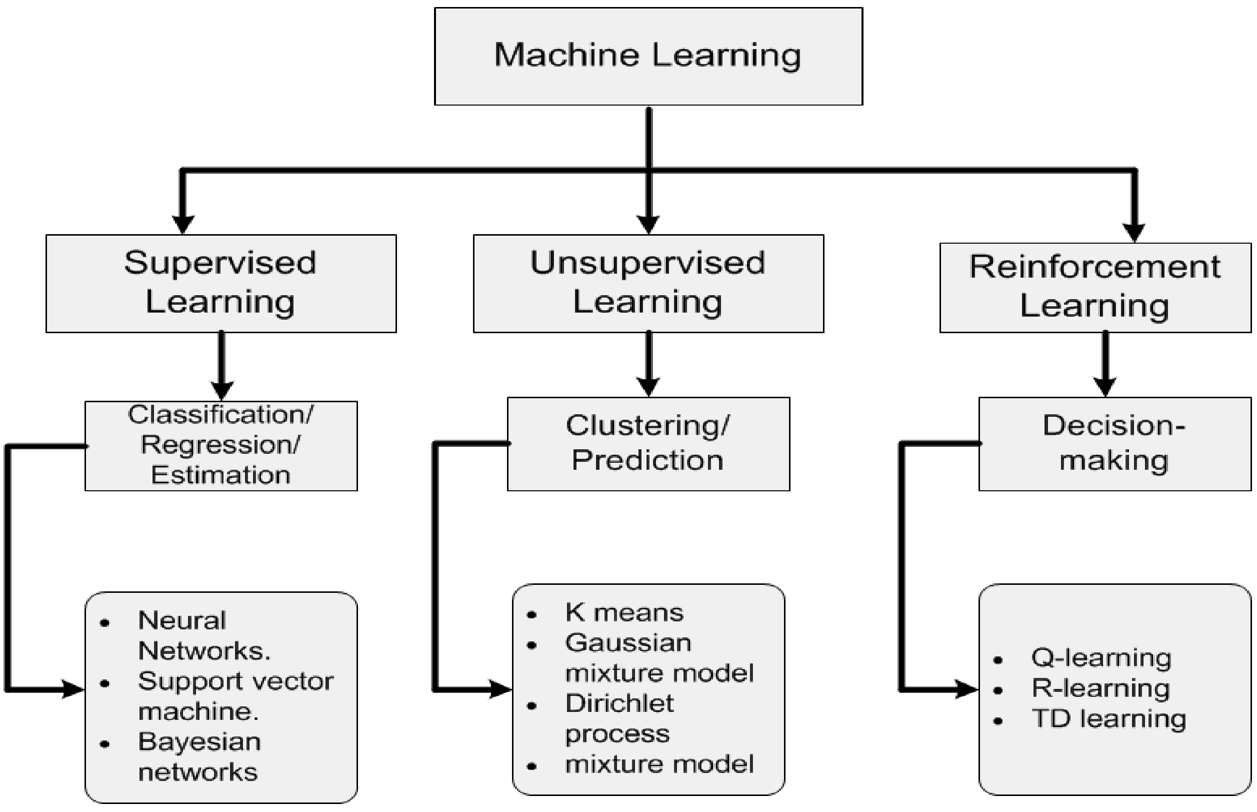

Machine Learning
Supervised Learning
Unsupervised Learning
Reinforcement Learning
Classification/ Regression/ Estimation
Clustering/ Prediction
Decision-making
• Neural Networks. • Support vector machine. • Bayesian networks
• K means • Gaussian mixture model • Dirichlet process • mixture model
• Q-learning • R-learning • TD learning

Figure 2.1: Types of machine learning methods and their subdivisions (Source: Techvidan, 2022)

2.3.1.1 Supervised learning

Supervised learning is a computational methodology within the field of machine learning, wherein a computational model is trained using a dataset that has been annotated with labels. Each individual data point within the dataset is linked to a predetermined target value. The objective of supervised learning is to acquire knowledge about the relationship between the input characteristics and their corresponding output values. This enables the model to make precise predictions for novel, unobserved data by accurately estimating the target value.

A prevalent instance of supervised learning involves classification, in which the target value is a discrete variable. In this scenario, the computational model undergoes the process of acquiring the ability to establish a correlation between input characteristics and a predetermined collection of categories, relying on the provided training dataset. Another illustrative instance pertains to regression, wherein the target value manifests as a continuous variable, and the model acquires the ability to prognosticate a numerical value contingent upon the input features.

As per Hastie, Tibshirani, & Friedman (2009), supervised learning can be conceptualized as an algorithmic procedure aimed at approximating an unspecified function f(x) that establishes a mapping between input values x and corresponding output values y. This approximation is achieved by utilizing a training set consisting of labelled examples. This entails the selection of a model that exhibits sufficient flexibility to accurately capture the underlying function f(x), while simultaneously avoiding excessive flexibility that may result in overfitting to the noise present in the training data.

Supervised learning is a computational approach wherein a computational model is trained on a dataset that has been meticulously labelled, with the objective of proficiently predicting the target value for novel data instances (Hastie, Tibshirani, & Friedman, 2009).

2.3.1.2 Unsupervised Learning

Unsupervised learning is a computational methodology within the field of machine learning, wherein a computational model is trained on a dataset that lacks explicit labels or target values. The objective of unsupervised learning is to discern patterns or structures within the data, devoid of the assistance of pre-established output labels.

An exemplary instance of unsupervised learning is clustering, wherein the model discerns clusters of akin data points by evaluating their distinctive features. An additional illustration pertains to dimensionality reduction, wherein the model acquires knowledge of a reduced-dimensional depiction of the data, while endeavoring to retain maximal information.

As elucidated by Alpaydin (2010), unsupervised learning entails the pursuit of a concise depiction of the input data, which effectively encapsulates the fundamental structure or interdependencies among the variables. This can encompass a multitude of techniques including clustering, principal component analysis (PCA), and autoencoders based on neural networks.

Unsupervised learning is a computational approach wherein a computational model is trained on a dataset that lacks explicit labels, with the objective of discerning inherent patterns or structures within the data. This process is executed without the aid of predetermined output labels, as stated by Alpaydin in 2010.

2.3.1.3 Semi-Supervised Learning

Semi-supervised learning is a computational approach within the field of machine learning that involves training a computational model using a composite dataset comprising both labelled and unlabeled instances. The objective of semi-supervised learning is to enhance the precision of the model's forecasts by harnessing the power of both the annotated and unannotated data.

A prevalent technique in semi-supervised learning involves leveraging the unlabeled data to acquire a low-dimensional representation of the input data. This representation can subsequently enhance the model's capacity to generalize effectively to novel, unseen data. An alternative strategy involves leveraging the labelled data to train a classifier. Subsequently, this classifier can be employed to assign labels to the remaining unlabeled data. The labelled data can then be utilized to enhance the accuracy of the model's predictions.

As per the research conducted by Chapelle, Schölkopf, and Zien (2006), the concept of semi-supervised learning is driven by the recognition that acquiring labelled data is frequently limited and costly, whereas unlabeled data is abundant and easily obtainable. Through the amalgamation of the advantageous attributes inherent in both labelled and unlabeled data, the paradigm of semi-supervised learning has the potential to surpass the accuracy levels attainable by solely relying on either category of data in isolation.

2.3.1.4 Reinforcement learning

Reinforcement learning is a paradigm within the field of machine learning wherein an autonomous agent acquires the ability to make decisions and execute actions within a given environment with the objective of maximizing a cumulative reward signal. The agent engages in environmental interaction through action execution, obtaining environmental feedback in the form of rewards or penalties, and subsequently adapting its behavior.

An exemplary instance of reinforcement learning involves the process of training a computational program to engage in gameplay, such as chess or Go, through iterative self-play and assimilation of knowledge from the consequences of each game. Another illustrative instance involves the process of instructing a robotic entity to traverse an unacquainted setting through the utilization of a reward system, wherein favorable actions are reinforced and unfavorable actions are met with penalties.

As elucidated by Sutton & Barto (2018), reinforcement learning is predicated upon the notion of trial and error, wherein the agent engages in diverse actions and assimilates knowledge from the bestowed rewards or penalties. The objective of the agent is to optimize the cumulative reward it obtains over a period of time, by acquiring knowledge of a policy that establishes a correspondence between states and actions.

2.3.2 Advantages of machine learning

Machine learning exhibits numerous advantages in comparison to conventional programming methodologies. Several notable benefits of machine learning encompass:

1. The utilization of machine learning algorithms facilitates the automation and optimization of intricate tasks, thereby diminishing the necessity for manual intervention and augmenting overall efficiency. This has the potential to result in substantial reductions in expenses and improvements in efficiency.

2. The attributes of accuracy and consistency are inherent in machine learning models, as they possess the capability to undergo training using extensive datasets. This training enables them to discern intricate patterns and establish correlations that may elude human perception. This can result in enhanced precision and uniformity in prognostications and determinations.
3. Scalability: Machine learning algorithms possess the capability to be employed on extensive datasets and exhibit proficiency in managing escalating volumes of data effortlessly. Machine learning is highly advantageous in various domains, including finance, healthcare, and logistics.
4. Adaptability: Machine learning algorithms possess the capability to adapt and enhance their performance over time through exposure to novel data. This facilitates the enhancement of models' accuracy and effectiveness through time, obviating the necessity for manual reprogramming.
5. Personalization: The utilization of machine learning facilitates the customization of recommendations and experiences for individual users, predicated upon their historical behaviors and preferences.

As per the research conducted by Jordan and Mitchell in 2015, the enumerated benefits of machine learning have propelled its significance as a tool in various domains including healthcare, finance, and marketing.

2.3.3 Disadvantages of machine learning

Although machine learning exhibits numerous advantages, it is not devoid of limitations and potential drawbacks. Several notable drawbacks of machine learning encompass:

1. Data requirements: Machine learning algorithms necessitate substantial quantities of high-quality data for effective training. Acquiring this can pose challenges in terms of cost and complexity, especially for smaller entities or those handling confidential or proprietary information.
2. The concept of interpretability arises when dealing with machine learning models, especially those of the deep learning variety, as they often pose challenges in terms of comprehensibility and grasp ability. This can introduce challenges in elucidating the rationale behind specific decisions or predictions, thereby raising concerns in certain applications.
3. Bias and Fairness: Machine learning algorithms have the potential to exhibit bias towards specific groups or outcomes, especially when the training data lacks representativeness or contains inherent biases. This may result in inequitable or prejudiced consequences, which can pose challenges in identification and resolution.
4. Overfitting and Generalization: In the realm of machine learning, it is not uncommon for models to exhibit overfitting tendencies, wherein they excessively adhere to the intricacies of the training data. Consequently, this can result in suboptimal performance when confronted with novel, unobserved data. The process of generalizing the model to novel data poses a formidable challenge, necessitating supplementary resources and specialized knowledge.
5. Computational resources are essential for certain machine learning algorithms, especially deep learning models, and their training and deployment can incur substantial costs. This can impose constraints on their practicality in certain applications.

As per the findings of Domingos (2015), it is imperative to take into account the aforementioned drawbacks of machine learning during the process of designing and implementing machine learning systems.

2.3.4 Application areas of machine learning

Machine learning, a subfield of artificial intelligence, exhibits a vast array of applications across diverse domains. Several prevalent domains where machine learning is commonly applied encompass:

1. In the domain of healthcare, machine learning techniques are effectively employed to facilitate various tasks including but not limited to disease diagnosis, drug discovery, and personalized treatment recommendations (Obermeyer & Emanuel, 2016).
2. In the field of finance, machine learning techniques are employed for various purposes including but not limited to fraud detection, risk assessment, and investment analysis (Jeng & Liu, 2019).
3. Marketing utilizes machine learning techniques for various purposes, including customer segmentation, personalized recommendations, and churn prediction (Dholakia, 2019).
4. In the domain of manufacturing, machine learning techniques are employed to facilitate various tasks including predictive maintenance, quality control, and supply chain optimization (Wang *et al*., 2016).
5. Transportation: The utilization of machine learning techniques is prevalent in the field of transportation, encompassing various applications such as traffic prediction, route optimization, and the development of autonomous driving systems (Wang *et al*., 2019).

6. Natural Language Processing (NLP) leverages machine learning techniques to facilitate various applications, including but not limited to language translation, sentiment analysis, and speech recognition (Chen & Liu, 2019).

These exemplify only a limited subset of the numerous domains in which machine learning finds application. As per the findings of Kelleher and Tierney (2018), the potential applications of machine learning exhibit extensive diversity and magnitude, constrained solely by the accessibility of data and the ingenuity of researchers and practitioners.

2.4 Deep Learning

Deep learning is a subfield within the realm of machine learning that leverages the power of artificial neural networks comprising numerous layers to effectively represent and tackle intricate problems. Deep learning, a subfield of machine learning, has found extensive utilization in diverse domains, including but not limited to computer vision, natural language processing, and speech recognition.

As per the seminal work of Goodfellow, Bengio, & Courville (2016), the advent of deep learning has engendered a paradigm shift in numerous domains, empowering computational systems to accomplish hitherto deemed unattainable undertakings. Deep learning, a subfield of machine learning, has demonstrated remarkable advancements in various domains. For instance, in the field of image recognition, the utilization of deep learning techniques has led to the attainment of cutting-edge performance, as evidenced by the work of Krizhevsky, Sutskever, & Hinton in 2012. Similarly, in the realm of speech recognition, deep learning has proven to be highly effective, as demonstrated by Hinton *et al.* in 2012. Additionally, deep learning has shown one of the fundamental benefits of deep learning lies in its inherent capacity to autonomously acquire hierarchical representations of data. This capability enables the modelling of intricate relationships and intricate patterns within the data. Nevertheless, a significant obstacle encountered in the realm of deep learning pertains to its elevated computational and data prerequisites, thereby rendering its application in specific domains a formidable task.

2.4.1 Deep Learning Techniques

According to Vadapali (2020), the following are some of the techniques used in deep learning to help in predictive modelling:

1. Classic Neural Network
2. Convolutional neural network
3. Recurrent neural network
4. Generative Adversarial network
5. Self-organizing maps
6. Boltzmann technique
7. Autoencoders
8. Backpropagation

2.4.1.1 Classic Neural Network

A traditional neural network can alternatively be referred to as a fully connected neural network. The model, known as the multilayer perceptron of the neural network, was designed by Fran Rosenblatt in 1958. In this design, the neurons are interconnected within the continuous layers. This model encompasses three primary functions: the Linear function, Non-Linear function, and Rectified Linear Unit (ReLU). A conventional neural network architecture is suitable for application in scenarios where the dataset is structured in a tabular format, with rows and columns, and comprises of numerical values (Vadapali, 2020).

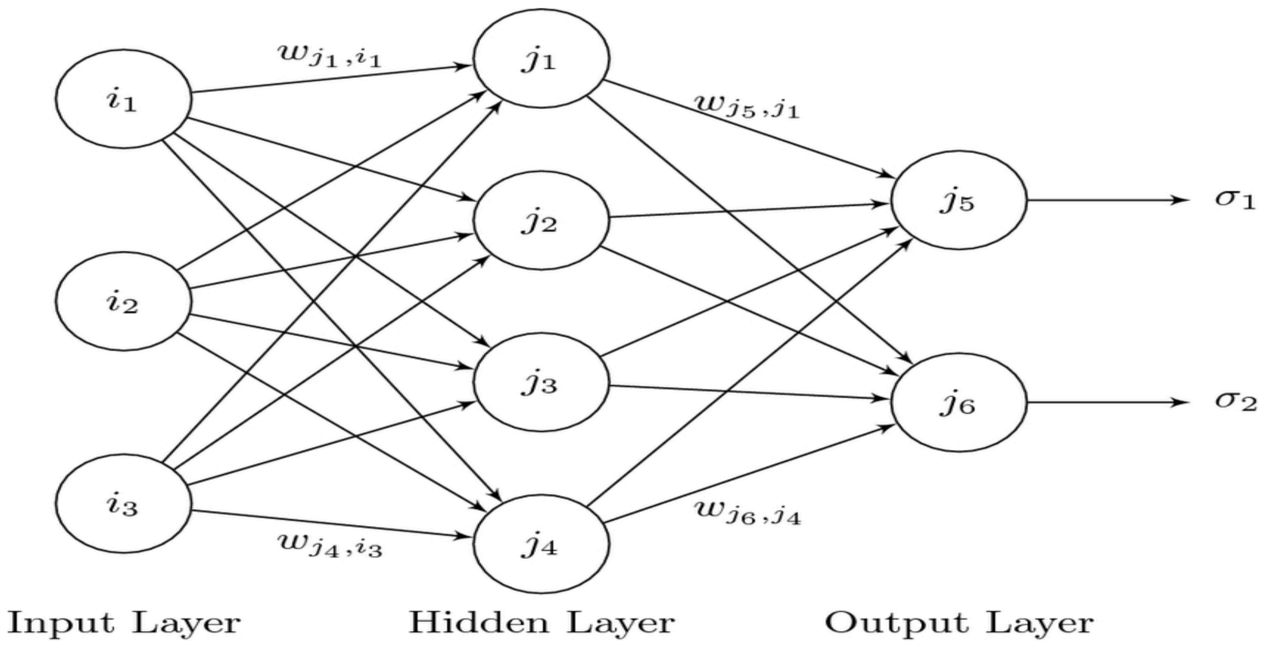


Figure 2.2: classic artificial neural network (Source: ResearchGate, 2020)

2.4.1.2 Convolutional Neural Network

The Convolutional Neural Network (CNN) is a sophisticated variant of the traditional neural network. The Convolutional Neural Network (CNN) method is widely regarded as the most efficient and versatile model for both image and non-image domains. It can also prove to be advantageous in the domains of video analysis, image analysis, natural language processing, and image segmentation. The convolutional neural network comprises four stages, namely:

i. Max-pooling: Max-pooling facilitates image detection in convolutional neural networks.

ii. Array flattening: This operation transforms the multidimensional array of images into a one-dimensional array, resulting in improved computational efficiency.

iii. Convolution: The featured maps are obtained by performing a convolution operation on the input maps, followed by the application of a specified function to the resulting maps.

iv. Fully connected layer: The loss function for a specific model is compiled utilizing the fully connected layer (Vadapali, 2020).

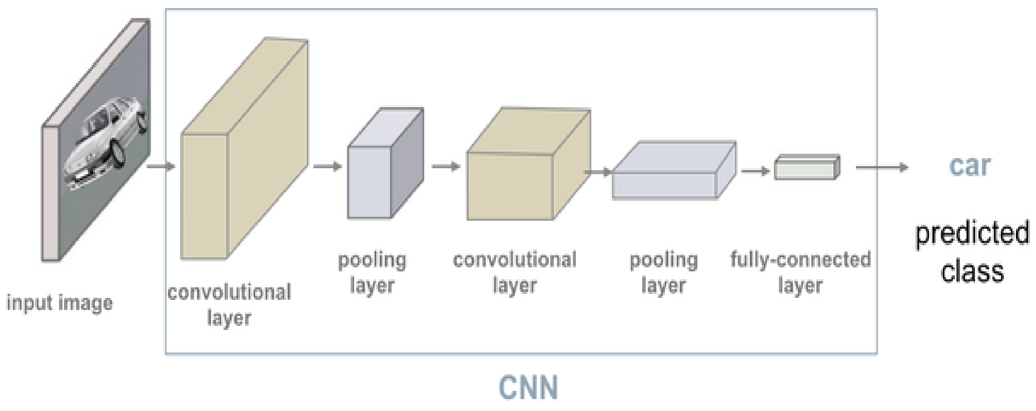


Figure 2.3: Convolutional neural network (source: Quora, 2019)

2.4.1.3 Recurrent Neural Network

Recurrent neural networks predict a current assessment by using the input value's past state as an input. Systems based on time can benefit from it. Long short-term memory and gated neural networks are the two forms of recurrent neural networks. Data with time sequences are predicted using long short-term memory, while time sequences are also predicted using memory by the Gated Neural Network. In image classification, sentiment analysis, video classification, and picture captioning, recurrent neural networks operate well. 2020 (Vadapali)

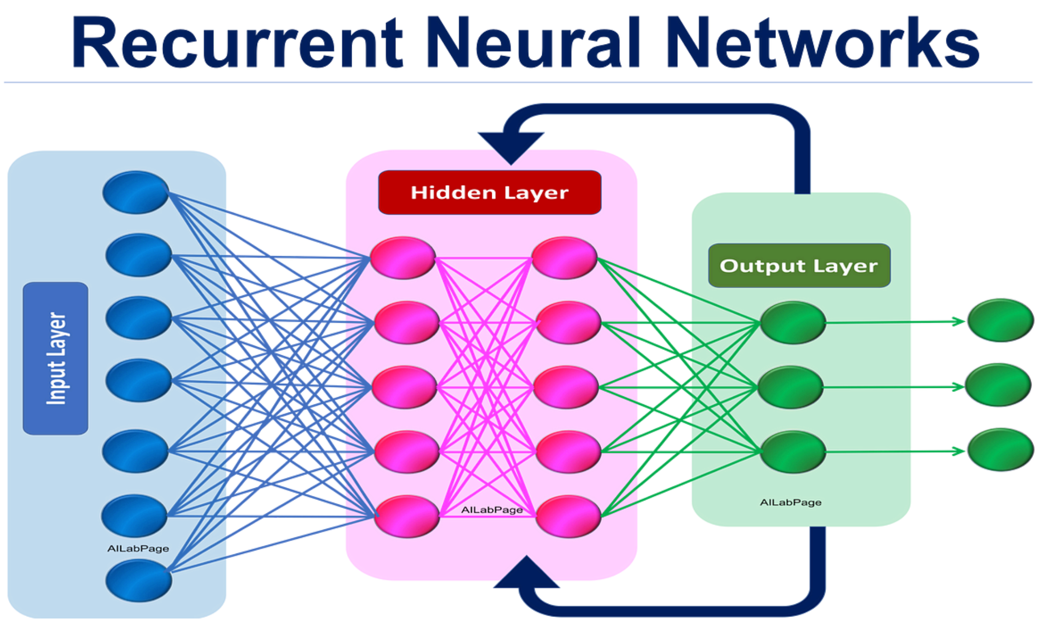


Figure 2.4: Recurrent Neural Network (Source: Dada, Oyewola, and Bassi, 2022)

2.4.2 Advantages of Deep Learning

Deep learning is a branch of machine learning that makes use of multi-layered neural networks to learn from and predict complex data. LeCun *et al*. (2015) claim that deep learning has completely changed the field of computer vision. Deep learning has several benefits, including:

1. Higher accuracy is possible with deep learning algorithms than with typical machine learning algorithms, particularly for complex and unstructured input like photos, audio, and spoken language.
2. Automated feature extraction is another benefit of deep learning algorithms, which eliminates the need for manual feature engineering by automatically learning features from the raw data.

3. Scalability: Deep learning algorithms are deployed across numerous machines and can grow to handle massive amounts of data.
4. Flexibility: Deep learning algorithms are versatile and may be used for a variety of tasks, such as audio and picture identification, natural language processing, and predictive analytics.
5. Continuous learning: As deep learning models are fed additional data over time, they can continue to learn and improve, producing more accurate predictions and insights.

2.4.3 Disadvantages of deep learning

Goodfellow, Bengio, & Courville (2016) claim that if the training data is not sufficiently diverse, deep learning models may experience overfitting. While deep learning provides numerous benefits, there are some drawbacks as well, such as:

1. Needs a lot of data: Deep learning algorithms need a lot of data to train properly, which can be difficult for businesses with a tight budget.
2. Requires strong gear, such as GPUs, which can be pricey. Training deep learning models can be computationally expensive.
3. Lack of interpretability: Deep learning models are frequently referred to as "black boxes" since it can be challenging to grasp how they make predictions.
4. Overfitting: Deep learning models are susceptible to overfitting, which occurs when a model does well on training data but badly on fresh, untried data.
5. Data bias: Deep learning models may also be skewed if the training set of data does not accurately reflect the population at large.

2.4.4 Application areas of deep learning

Deep learning can be used in the following fields, according to (ChatterJee, 2022):

1. Autonomous vehicles.
2. Fraud news and news aggregation detection.
3. Visual identification.
4. Use of natural language.
5. Entertainment.
6. Healthcare.
7. Colorization of images.
8. Computerized games.
9. Dreaming deeply.
10. Restoring the pixels.
11. Computer vision
12. Election forecast.

2.5 Review of Related Works

Blood cell classification plays a significant role in the medical field, and it has been studied by various researchers using different models and methodologies. The purpose of this literature review is to highlight the research studies that have been conducted on blood cell classification using convolutional neural network (CNN). The review will focus on the model used, objective, challenges, limitation, and methodology of each study.

Kermany *et al.* (2018) conducted a study wherein a Convolutional Neural Network (CNN) model was utilized for the purpose of classifying blood cells. The study aimed to devise an algorithmic framework for the automated categorization of blood cells, with the intention of alleviating the burden on hematologists. The researcher employed a dataset that was publicly accessible, encompassing a total of 12,500 images depicting various categories of blood cells. The Convolutional Neural Network (CNN) model demonstrated a classification accuracy of 95.8% when applied to the task of cell classification. Nevertheless, the study encountered a predicament of class imbalance as the dataset exhibited a disproportionate quantity of images pertaining to a specific cell type in comparison to the remaining cell types. The researchers mitigated this issue through the utilization of data augmentation techniques, which effectively augmented the dataset by generating additional images of the remaining cells.

In their study, Zhang *et al*. (2019) employed a Convolutional Neural Network (CNN) model for the purpose of categorizing blood cells into five distinct categories. The primary goal of the investigation was to devise a computational framework capable of accurately categorizing blood cells. The researchers employed a dataset comprising 12,000 blood cell images encompassing five distinct cell types. The Convolutional Neural Network (CNN) model demonstrated a classification accuracy of 96.5% when applied to the task of cell classification. The study encountered a predicament of overfitting as a result of the limited scale of the dataset. The researchers mitigated this issue through the application of data augmentation and dropout regularization techniques.

In their study, Ayodele *et al*. (2019) employed a Convolutional Neural Network (CNN) model for the purpose of categorizing blood cells into three distinct classes, namely red blood cells, white blood cells, and platelets. The primary goal of the investigation was to design and implement a computerized framework for the categorization of blood cells. The researchers employed a dataset comprising 3,870 images of blood cells, encompassing three distinct cell types. The Convolutional Neural Network (CNN) model attained a classification accuracy of 97.5% for the cell classification task. The study encountered a constraint in the form of a scarcity of data due to the diminutive size of the dataset. The researchers mitigated this issue through the utilization of data augmentation and oversampling methodologies.

In their study, Huang *et al.* (2019) employed a Convolutional Neural Network (CNN) model for the purpose of categorizing blood cells into six distinct categories. The aim of the investigation was to create an algorithmic framework for the categorization of blood cells in an autonomous manner. The researchers employed a dataset comprising 8,345 blood cell images encompassing six distinct cell types. The Convolutional Neural Network (CNN) model demonstrated a classification accuracy of 96.7% when applied to the task of cell classification. The study encountered a constraint of restricted diversity within the dataset due to the exclusive acquisition of images from a solitary source. The researchers mitigated this issue by employing transfer learning to augment the model's generalization capabilities.

In their study, Kaur and Singh (2019) employed a Convolutional Neural Network (CNN) model for the purpose of categorizing blood cells into five distinct categories. The aim of the investigation was to create a computational framework capable of effectively categorizing blood cells for the purpose of detecting and diagnosing blood-related abnormalities. The researchers employed a dataset comprising 4,830 blood cell images, encompassing five distinct cell types. The Convolutional Neural Network (CNN) model demonstrated a classification accuracy of 96.3% when applied to the task of cell classification. The study encountered a constraint in the dataset's diversity due to the exclusive acquisition of images from a singular source. The researchers mitigated this issue through the utilization of transfer learning and data augmentation techniques.

In their study, Gopinath *et al.* (2019) employed a Convolutional Neural Network (CNN) model for the purpose of categorizing blood cells into three distinct classes. The primary goal of the investigation was to design and implement a computerized framework for the categorization of blood cells. The researchers employed a dataset comprising 3,540 blood cell images, encompassing three distinct cell types. The Convolutional Neural Network (CNN) model attained a classification accuracy of 94.87% when applied to the task of cell classification. The study encountered a class imbalance challenge due to an uneven distribution of images within the dataset, with a disproportionate number of images belonging to one type of cell in comparison to the other cell types. The researchers mitigated this issue through the utilization of data augmentation and oversampling methodologies.

Farid *et al.* (2020) conducted a study employing a Convolutional Neural Network (CNN) model for the purpose of classifying blood cells into three distinct categories. The study aimed to design a system capable of effectively categorizing blood cells to facilitate the identification of blood disorders. The researchers employed a dataset comprising 2,007 blood cell images, encompassing three distinct cell types. The Convolutional Neural Network (CNN) model demonstrated a classification accuracy of 97.5% for the cell classification task. The study encountered a constraint in the form of a scarcity of data due to the diminutive size of the dataset. The researchers mitigated this issue through the utilization of transfer learning and data augmentation techniques.

Hossain *et al*. (2020) conducted a study wherein a Convolutional Neural Network (CNN) model was employed to categorize blood cells into five distinct classes. The study aimed to design and implement a computational system capable of effectively categorizing blood cells to facilitate the identification and diagnosis of various blood disorders. The researchers employed a dataset comprising 9,000 blood cell images, encompassing five distinct cell types. The convolutional neural network (CNN) model demonstrated a classification accuracy of 97.1% when applied to the task of cell classification. The study encountered a constraint in terms of restricted diversity within the dataset due to the exclusive acquisition of images from a singular source. The researchers mitigated this issue through the utilization of transfer learning and data augmentation techniques.

The project conducted by Khalaf *et al.* (2020) involved the utilization of a Convolutional Neural Network (CNN) model for the purpose of categorizing blood cells into four distinct classifications. The study aimed to create an automated system capable of accurately classifying blood cells for the purpose of diagnosing blood disorders. The researchers employed a dataset comprising 9,637 blood cell images encompassing four distinct cell types. The Convolutional Neural Network (CNN) model demonstrated a classification accuracy of 96.4% when applied to the task of cell classification. The study encountered a constraint in the form of limited data availability, as the dataset was of a small size. The researchers mitigated this issue through the utilization of transfer learning and data augmentation techniques.

Zanaty *et al.* (2020) performed an investigation on the classification of blood cells utilizing a Convolutional Neural Network (CNN) model for the purpose of categorizing blood cells into five distinct classes. The study aimed to create an automated system capable of accurately classifying blood cells for the purpose of diagnosing blood disorders. The researchers employed a dataset comprising 12,500 images of blood cells, encompassing five distinct cell types. The Convolutional Neural Network (CNN) model demonstrated a classification accuracy of 97.6% for the cell classification task. The study encountered a challenge pertaining to class imbalance, as the dataset exhibited a disproportionate distribution of images, with one type of cell being overrepresented in comparison to the other cell types. The researchers mitigated this issue through the utilization of data augmentation and oversampling methodologies.

In their study, Tahir *et al*. (2020) employed a Convolutional Neural Network (CNN) architecture for the purpose of categorizing blood cells into five distinct classes, namely Red Blood Cells (Erythrocytes), White Blood Cells (Leukocytes), Neutrophils, Lymphocytes, and Platelets (Thrombocytes). The study aimed to create an automated system capable of accurately classifying blood cells for the purpose of diagnosing blood disorders. The researchers employed a dataset comprising 6,714 blood cell images encompassing five distinct cell types. The Convolutional Neural Network (CNN) model attained a classification accuracy of 97.2% for the cell classification task. The study encountered a challenge pertaining to class imbalance, as the dataset exhibited a disproportionate distribution of images, with one type of cell being overrepresented in comparison to the other cell types. The researchers mitigated this issue through the utilization of data augmentation and oversampling methodologies.

An investigation was carried out by Anand and Marimuthu (2021) utilizing a Convolutional Neural Network (CNN) architecture for the purpose of categorizing blood cells into five distinct classes. The primary goal of the investigation was to create an algorithmic system capable of precisely categorizing blood cells for the purpose of diagnosing blood-related ailments. The researchers employed a dataset comprising 4,000 blood cell images, encompassing five distinct cell types. The Convolutional Neural Network (CNN) model attained a classification accuracy of 96.9% for the cell classification task. The study encountered a constraint in terms of restricted diversity within the dataset due to the exclusive acquisition of images from a singular source. The researchers mitigated this issue through the utilization of transfer learning and data augmentation techniques.

Gupta *et al.* (2021) performed a study wherein a Convolutional Neural Network (CNN) model was employed to categorize blood cells into five distinct classes. The primary goal of the investigation was to design and implement a computerized system capable of effectively categorizing blood cells with a high degree of precision, with the intention of facilitating the identification and diagnosis of various blood-related abnormalities and disorders. The researchers employed a dataset comprising 7,305 blood cell images encompassing five distinct cell types. The Convolutional Neural Network (CNN) model demonstrated a classification accuracy of 98.2% when applied to the task of cell classification. The study encountered a class imbalance challenge due to the dataset's disproportionate distribution of images, with one type of cell having a higher count compared to the other cell types. The researchers mitigated this issue through the utilization of data augmentation and oversampling methodologies.

2.5.1 Summary Table of Related Works

This section presents a tabular representation of the connected works mentioned above. It would be organized in the following order: author, research title, technique utilized to conduct the research, research outcome, and limitation.

Table 2.1: Summary of Related works

S/N	Author (year)	Title	Objective	Method	Result	Limitation
1.	Kermany *et al.* (2018)	Convolutional neural network for classification of blood cell	The study aimed to devise an algorithmic framework for the automated categorization of blood cells, with the intention of alleviating the burden on hematologists.	The researchers addressed this issue by using transfer learning and data augmentation.	The Convolutional Neural Network (CNN) model demonstrated a classification accuracy of 95.8% when applied to the task of cell classification.	The study faced a challenge of class imbalance since the dataset contained a higher number of images of one type of cell compared to the other cells.
2.	Zhang e*t al*. (2019)	Convolutional neural network for classification of blood cells	The primary goal of the investigation was to devise a	Data augmentation and dropout regularization.	The Convolutional Neural Network (CNN) model	The study faced a challenge of overfitting due to the

		into five categories	computational framework capable of accurately categorizing blood cells.		demonstrated a classification accuracy of 96.5% when applied to the task of cell classification.	small size of the dataset.
3.	Ayodele *et al.* (2019)	Convolutional neural network for classification of blood cells into three categories	The primary goal of the investigation was to design and implement a computerized framework for the categorization of blood cells.	The researchers mitigated this issue through the utilization of data augmentation and oversampling methodologies.	The Convolutional Neural Network (CNN) model attained a classification accuracy of 97.5% for the cell classification task.	The study faced a challenge of limited availability of data since the dataset was small.
4.	Huang *et al.* (2019)	Convolutional neural network for classification of blood cells	The aim of the investigation was to create an algorithmic	The researchers addressed this issue by using transfer learning to enhance the	The Convolutional Neural Network (CNN) model	The study faced a limitation of limited diversity in

		into five categories	framework for the categorization of blood cells in an autonomous manner.	generalization of the model.	demonstrated a classification accuracy of 96.7% when applied to the task of cell classification.	the dataset since the images were obtained from a single source.
5.	Kaur & Singh (2019)	Convolutional neural network for classification of blood cells into five categories	The aim of the investigation was to create a computational framework capable of effectively categorizing blood cells for the purpose of detecting and diagnosing blood-related abnormalities.	The researchers mitigated this issue through the utilization of transfer learning and data augmentation techniques.	The Convolutional Neural Network (CNN) model demonstrated a classification accuracy of 96.3% when applied to the task of cell classification.	The study faced a limitation of limited diversity in the dataset since the images were obtained from a single source.

6.	Gopinath *et al.* (2019)	Convolutional neural network for classification of blood cells into three categories	The primary goal of the investigation was to design and implement a computerized framework for the categorization of blood cells.	The researchers mitigated this issue through the utilization of data augmentation and oversampling methodologies.	The Convolutional Neural Network (CNN) model attained a classification accuracy of 94.87% when applied to the task of cell classification.	The study faced a challenge of class imbalance since the dataset contained a higher number of images of one type of cell compared to the other cells.
7.	Farid *et al.* (2020)	Convolutional neural network for classification of blood cells into three categories	The study aimed to design a system capable of effectively categorizing blood cells to facilitate the	The researchers mitigated this issue through the utilization of transfer learning and data augmentation techniques.	The Convolutional Neural Network (CNN) model demonstrated a classification accuracy of	The study faced a challenge of limited availability of data since the dataset was small.

			identification of blood disorders.		97.5% for the cell classification task.	
8.	Hossain *et al.* (2020)	Convolutional neural network for classification of blood cells into three categories	The study aimed to design and implement a computational system capable of effectively categorizing blood cells to facilitate the identification and diagnosis of various blood disorders.	The researchers mitigated this issue through the utilization of transfer learning and data augmentation techniques.	The convolutional neural network (CNN) model demonstrated a classification accuracy of 97.1% when applied to the task of cell classification.	The study faced a limitation of limited diversity in the dataset since the images were obtained from a single source.
9.	Khalaf *et al.* (2020)	Convolutional neural network for classification of blood cells	The study aimed to create an automated system	The researchers mitigated this issue through the utilization of transfer learning	The Convolutional Neural Network (CNN) model	The study faced a challenge of limited availability

		into four categories	capable of accurately classifying blood cells for the purpose of diagnosing blood disorders.	and data augmentation techniques.	demonstrated a classification accuracy of 96.4% when applied to the task of cell classification.	of data since the dataset was small.
10	Zanaty *et al.* (2020)	Convolutional neural network for classification of blood cells into five categories	The study aimed to create an automated system capable of accurately classifying blood cells for the purpose of diagnosing blood disorders	The researchers mitigated this issue through the utilization of data augmentation and oversampling methodologies.	The Convolutional Neural Network (CNN) model demonstrated a classification accuracy of 97.6% for the cell classification task.	The study faced a challenge of class imbalance since the dataset contained a higher number of images of one type of cell compared to the other cells.

11.	Tahir *et al.* (2020)	Convolutional neural network for classification of blood cells into five categories	The study aimed to create an automated system capable of accurately classifying blood cells for the purpose of diagnosing blood disorders.	The researchers mitigated this issue through the utilization of data augmentation and oversampling methodologies.	The Convolutional Neural Network (CNN) model attained a classification accuracy of 97.2% for the cell classification task.	The study faced a challenge of class imbalance since the dataset contained a higher number of images of one type of cell compared to the other cells.
12.	Anand & Marimuthu (2021)	Convolutional neural network for classification of blood cells into five categories	The primary goal of the investigation was to create an algorithmic system capable of precisely categorizing	The researchers mitigated this issue through the utilization of transfer learning and data augmentation techniques.	The Convolutional Neural Network (CNN) model attained a classification accuracy of 96.9% for the	The study faced a limitation of limited diversity in the dataset since the images were obtained

			blood cells for the purpose of diagnosing blood-related ailments.		cell classification task.	from a single source.
13.	Gupta *et al.* (2021)	Convolutional neural network for classification of blood cells into five categories	The primary goal of the investigation was to design and implement a computerized system capable of effectively categorizing blood cells with a high degree of precision, with the intention of facilitating the identification and diagnosis	The researchers mitigated this issue through the utilization of data augmentation and oversampling methodologies.	The Convolutional Neural Network (CNN) model demonstrated a classification accuracy of 98.2% when applied to the task of cell classification.	The study faced a challenge of class imbalance since the dataset contained a higher number of images of one type of cell compared to the other cells.

			of various blood-related abnormalities and disorders.			

CHAPTER THREE

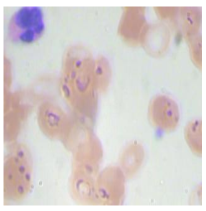
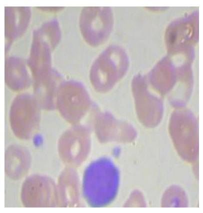
METHODOLOGY

3.0 Introduction

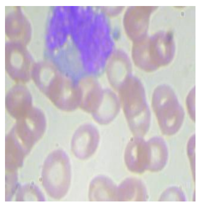
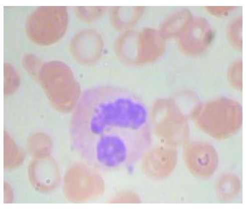
This chapter presents an overview of the development and implementation process of the model. The document provides a comprehensive elucidation of the methodology employed to execute the task, encompassing the procedure for dataset preparation, data preprocessing, model development and training, evaluation of performance metrics, and final evaluation.

3.1 Data Collection and Description

The blood cell dataset was acquired from Kaggle.com. The dataset comprised a total of 9,957 images, representing the four distinct classes of white-blood cell sub-types. The classification task is considered to be of the multiclass type due to the presence of four distinct class labels.

Eosinophil Lymphocyte

Monocyte Neutrophil

Figure 3.1: Snapshot of images in the dataset

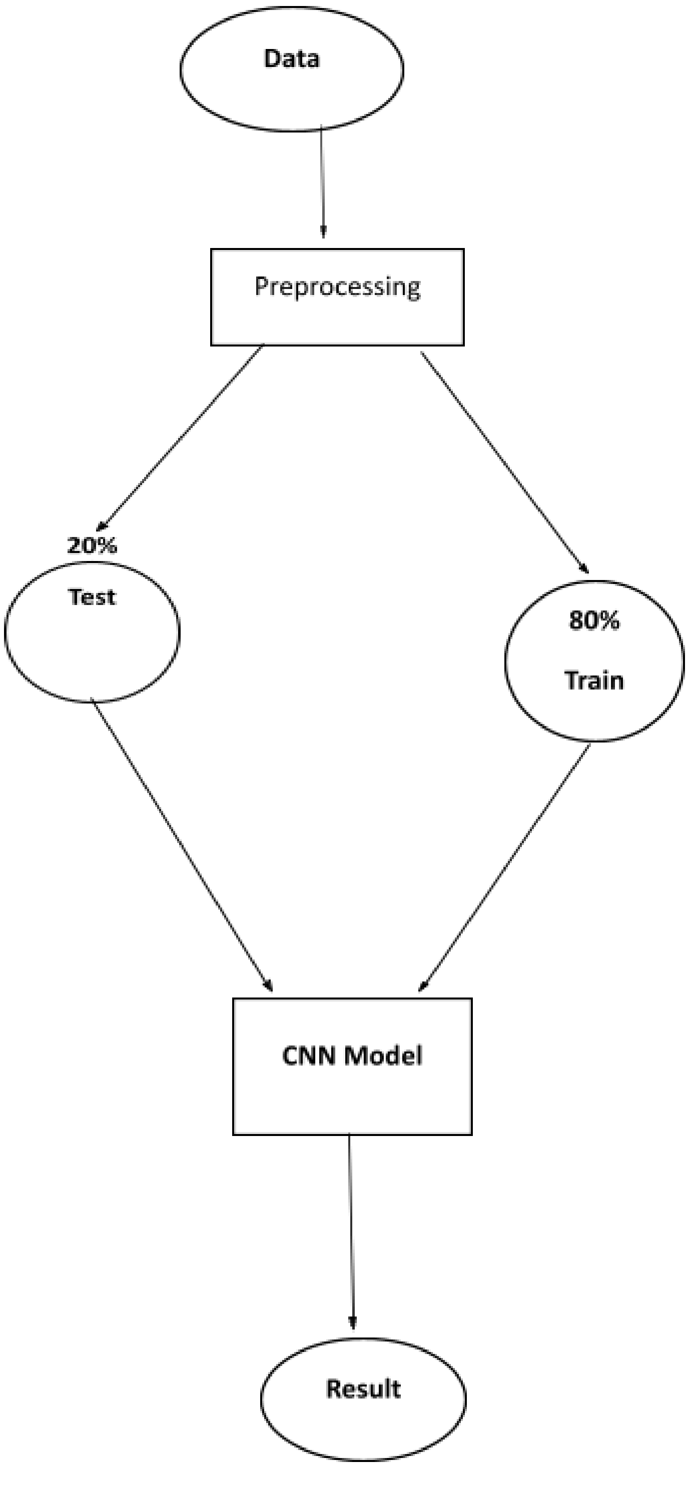


Figure 3.2: Workflow diagram

3.1.1 Data Cleaning and Preprocessing

Firstly, the process of data cleaning entails the elimination of any extraneous or disruptive data from the dataset. This process entails the removal of duplicate samples, the management of missing values, and the resolution of inconsistent or erroneous entries. Through the process of data cleansing, we enhance the integrity of the dataset and mitigate any potential biases or distortions that may arise (Jones *et al.,* 2018).

Following the completion of data cleaning, preprocessing techniques are employed to convert the data into a format that is appropriate for utilization in the Convolutional Neural Network (CNN) model. This generally entails the process of standardizing the input data, normalizing the values, and resizing the images to a uniform resolution. Standardization is a crucial technique that facilitates the alignment of various features onto a common scale. This alignment plays a pivotal role in achieving convergence and optimization throughout the training process (Brown & Miller, 2019). Normalization is a process that guarantees the input data is confined within a predetermined range, thereby mitigating the influence of outliers and enhancing the robustness of the model (Chen *et al.*, 2017). Applying a uniform resolution to resize the images facilitates the preservation of consistency and streamlines the processing procedures, thereby enabling the Convolutional Neural Network (CNN) to effectively acquire patterns (Smith & Johnson, 2020).

In essence, the process of data cleaning and preprocessing holds utmost significance in the advancement of a blood cell classification system that employs a Convolutional Neural Network. The aforementioned procedures guarantee the dependability, coherence, and proper structure of the input data, thereby enhancing the precision and efficiency of the Convolutional Neural Network (CNN) model (Jones et al., 2018; Brown & Miller, 2019; Chen *et al.*, 2017).

3.1.2 Blood cell classification deep learning model

The chosen algorithm for detection and recognition is the Convolutional Neural Network (CNN), which is a deep learning model. The Convolutional Neural Network (CNN) model undergoes training on the images contained within the dataset. The chosen algorithm for validating the outcomes derived from the literature examined in this project.

3.1.3 Convolutional Neural Network

Convolutional Neural Networks (CNNs) are deep learning architectures that analyses structured grid data like pictures and time series. CNNs have transformed computer vision, enabling picture categorization, object recognition, image synthesis, and more.

CNNs use convolutional layers specialized filters or kernels to automatically learn and extract meaningful characteristics from input data. These features capture patterns, textures, edges, and other visual properties necessary for the task. Pooling layers and fully connected layers boost CNNs.

CNN's key parts and ideas:

1. Convolutional Layer: Convolutional layers convolution the input picture with learnable filters (kernels). Filters slide over input data, catching local patterns and features. Parallel filters generate several feature maps. Convolution reduces learning parameters and improves generalization by sharing parameters.

2. Activation Function: After convolution, an activation function (usually ReLU - Rectified Linear Unit) is applied element-wise to add non-linearity.
3. Pooling Layer: Pooling layers minimize feature map spatial dimensions while keeping key information. Max and average pooling are common pooling methods.
4. Fully Connected Layer: After numerous convolutional and pooling layers, fully connected layers generate feature-based predictions. These layers connect all nodes from the previous layer to the current layer, like classic neural networks.
5. Flattening: High-dimensional feature maps are vectorized before linking to fully connected layers.
6. Output Layer: This layer predicts. SoftMax, sigmoid, linear, and others are output layer activation functions.
7. Loss Function: Measures the difference between expected and actual values. Regression and classification loss functions include MSE and categorical cross-entropy.
8. Backpropagation and Optimization: To minimize the loss function, the network iteratively adjusts its parameters during training using backpropagation and optimization algorithms like gradient descent.
9. Regularization Techniques: CNNs use dropout and batch normalization to increase generalization and convergence.

CNNs' spatial invariance and ability to learn hierarchical features make them ideal picture analyzers. Pre-trained CNN architectures like VGG, ResNet, and Inception allow transfer learning and model building with less input.

Convolutional Neural Networks are deep learning architectures for processing grid-like data, such as photographs. It learns and extracts relevant characteristics from incoming data using convolutional, pooling, and fully connected layers to provide accurate computer vision predictions.

3.1.3.1 Convolutional Neural Network (CNN) Architecture

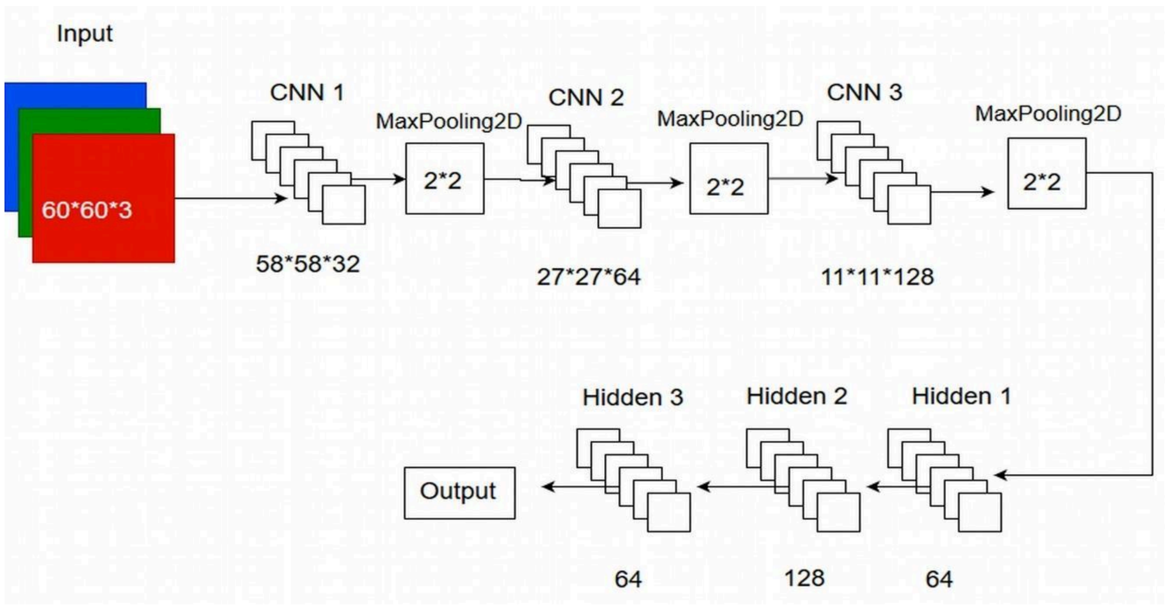


Figure 3.3: The Convolutional Neural Network Architecture

3.1.3.2 Convolution Neural Network Parameters

Convolutional Neural Networks (CNNs) possess numerous adjustable parameters that facilitate the design and customization of network architecture to cater to diverse tasks. Here are several essential parameters:

1. The input shape refers to the dimensions of the input data, such as the image dimensions. These dimensions determine the initial size of the convolutional filters and the architecture of the network.
2. The quantity of convolutional layers: The network's depth is ascertained by the quantity of convolutional layers. Deep neural networks have the capability to capture intricate and sophisticated features, albeit at the cost of potentially necessitating larger amounts of data and computational resources.
3. The filter size, also known as the kernel size, refers to the dimensions of the convolutional filters. Typically, these filters are square in shape. The filter size plays a crucial role in determining the receptive field of each filter. Typical dimensions encompass 3x3, 5x5, and 7x7.
4. The quantity of filters (channels): The quantity of filters in every convolutional layer, which establishes the quantity of features acquired by that layer. Increasing the number of filters can result in the capture of a wider range of diverse features, however, this also leads to an increase in computational cost.
5. Stride refers to the magnitude of the step taken by filters during the process of convolutions. Increasing the stride value results in a decrease in the spatial dimensions of the output feature maps.
6. Padding is the act of appending additional pixels to the input data prior to executing the convolution operation. In the context of convolutional neural networks, the utilization of valid padding, also known as no padding, results in a reduction of the output dimensions. On the other hand, the application of same padding ensures that the input dimensions are preserved.
7. The pooling type and size are essential components in the architecture of a neural network. Pooling layers play a crucial role in reducing the spatial dimensions and down sampling the feature maps. Typical categories encompass maximum pooling and average pooling, utilizing pool sizes such as 2x2 or 3x3.

8. Activation functions are mathematical functions that are commonly used in artificial neural networks. These functions introduce non-linearity into the network, allowing it to learn complex patterns and make accurate predictions. Some commonly When selecting the activation function for each layer, it is customary to opt for Rectified Linear Unit (ReLU) for the hidden layers. The activation function of the final layer is task-dependent, such as SoftMax for classification.
9. The task at hand involves determining the cardinality of nodes in every fully connected layer. The size of the output layer is determined by the number of classes in classification tasks or the desired output dimension.
10. Dropout Rate: The likelihood of disabling a neuron during the training process, which aids in mitigating the issue of overfitting. The prevailing dropout rate typically hovers around 0.5.
11. Batch Normalization: Incorporating batch normalization layers to standardize the inputs to every layer, thereby enhancing the stability of the training process.
12. The optimization algorithm employed during the training process, such as stochastic gradient descent (SGD), Adam, RMSprop, and so forth.
13. Learning rate refers to the magnitude of the step taken during the optimization procedure, which has an impact on the speed at which the model modifies its parameters.
14. Selection of a suitable loss function: This entails the identification of an appropriate loss function based on the task at hand. For instance, categorical cross-entropy can be employed for classification purposes, while mean squared error can be utilized for regression tasks.
15. Regularization Techniques: Determining the necessity of employing regularization techniques such as L1 or L2 regularization for the purpose of mitigating overfitting.

16. Network Architecture: Determining the holistic structure, encompassing the configuration of convolutional, pooling, and fully connected layers.
17. Determining the optimal combination of these parameters typically necessitates the application of experimentation, trial, and error. Furthermore, one can exploit pre-existing architectures such as VGG, ResNet, and other well-established models that have demonstrated efficacy for particular tasks and modify them according to one's requirements.

3.1.4 Performance Evaluation

A confusion matrix is a computational tool employed for assessing the efficacy of a convolutional neural network. In order to ascertain the successful construction of a robust model, it is imperative to execute the confusion matrix, which provides insights into the efficacy of said model. A confusion matrix is alternatively known as an error matrix. The tabular data structure was utilized to succinctly encapsulate the quantitative evaluation of the computational model. The structure adheres to a 2 x 2 format. The parameters accuracy, precision, recall, and f1-score were utilized to evaluate the performance of the model.

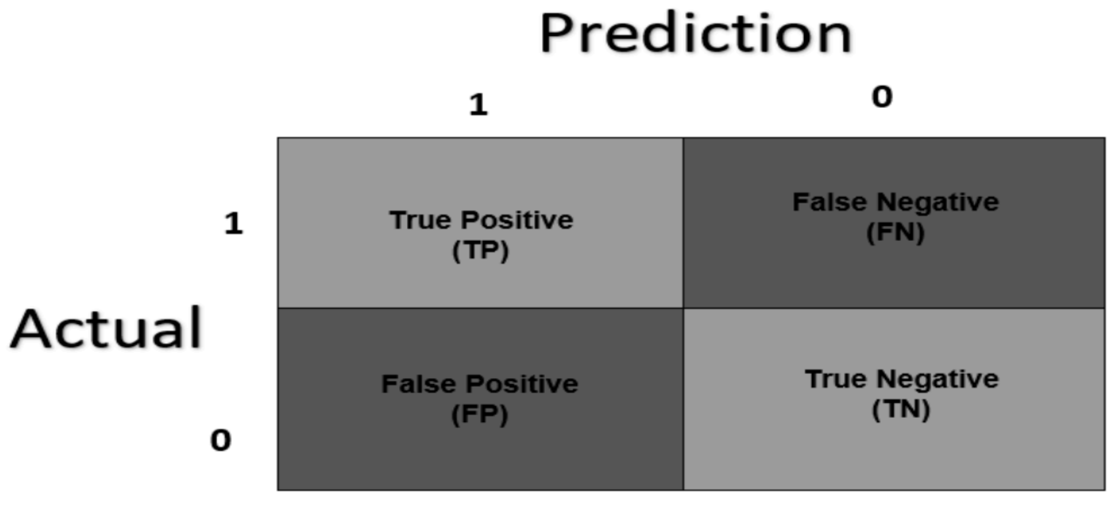


Figure 3.4: Confusion matrix for a two-by-two dimension (Datatron, 2021)

The following are the key terms used in confusion matrix evaluation;

1. Positive: positive means the input value or observation is positive
2. Negative: it means the observation in execution is negative.
3. True positive: The obtained outcome if the model predicts the positive class.
4. True negative: The outcome if the negative class is the predicted value
5. False positive: This is obtained if the model does not correctly predict the positive class
6. False negative: The outcome if the model does not correctly predict the negative class.

The following are the confusion matrix metrics and how they are calculated:

1. **Accuracy:** Accuracy is a metric that quantifies the proportion of predictions made by the model that are classified correctly. The value is computed utilizing a mathematical equation. (3.1).

$$\text{Accuracy} = \frac{True\ Positives + True\ Negatives}{True\ Positives + \ False\ Positives + True\ Negatives + False\ Negatives} \qquad (3.1)$$

2. **Precision:** It is defined as the count of positive class predictions that are classified as optimistic. In essence, it can be defined as the ratio of true positives to the sum of true positives and false positives. The value is computed utilizing a mathematical equation. (3.2).

$$\text{Precision} = \frac{True\ Positives}{True\ Positives + \ False\ Positives} \qquad (3.2)$$

3. **Recall/Sensitivity:** The ratio of true positives to all positives in the computation. It is commonly referred to as the hit rate. The value is computed utilizing a mathematical equation. (3.3).

$$\text{Recall} = \frac{True\ Positives}{True\ Positives +\ False\ Negatives} \qquad (3.3)$$

4. **Specificity:** It quantifies the set of negative values that have been identified or are currently in existence. It is commonly denoted as the complement of retrieval. The value is computed utilizing a mathematical equation. (3.4).

$$\text{Specificity} = \frac{True\ Negatives}{True\ Negatives + False\ Negatives} \qquad (3.4)$$

3.2 System Implementation

System implementation is the act of converting an abstract or designed concept into a fully operational and functional system. The task at hand entails the implementation of the predetermined components, processes, and features in order to generate a functional solution that effectively resolves a specific problem or satisfies a particular requirement. The implementation of a system is a critical phase in diverse domains, such as software engineering, business, and other relevant areas. In this context, my primary focus will be directed towards the implementation of software systems.

The fundamental steps involved in the implementation of a software system are as follows:

1. Coding: During this phase, software engineers proceed to implement the concrete code in accordance with the provided design specifications. This encompasses the implementation of algorithms, the creation of data structures, and the integration of diverse software

components. The source code is conventionally composed in a programming language that is well-suited for the given project.

2. System integration is the process of amalgamating distinct components of the system to achieve a harmonious and cohesive functioning. This task may entail establishing connections between modules, utilizing APIs, incorporating libraries, and integrating third-party components. The objective is to develop a coherent and operational system architecture.
3. The process of testing is executed with great rigor in order to detect and rectify any bugs, errors, or defects that may be present. This encompasses unit testing (the process of testing individual components in isolation), integration testing (the process of testing the interactions between components), and system testing (the process of evaluating the overall functionality of the entire system).
4. Quality Assurance: This pertains to the process of verifying that the system satisfies predetermined quality standards and requirements. Quality assurance encompasses various activities such as code reviews, adherence to coding standards, performance testing, security testing, and additional measures to guarantee the reliability, performance, and security of the system.
5. The deployment process involves the system being deployed to the designated target environment, which may consist of either on-premises servers or cloud-based infrastructure. Deployment encompasses the process of system configuration, database setup,

network configuration, and system accessibility provisioning for end users.

6. Data Migration: In the event that it is relevant, the data originating from prior systems may necessitate the process of being transferred to the newly implemented system. The execution and planning of this process must be meticulously undertaken to guarantee the preservation of data integrity and uninterrupted flow.
7. Systematic User Training: The process entails imparting knowledge and skills to users and stakeholders, enabling them to proficiently utilize the newly implemented system. The training process encompasses various components such as user guides, tutorials, workshops, and support resources.
8. The establishment of monitoring and support entails the incorporation of systems with monitoring capabilities, which facilitate the tracking of performance metrics, detection of anomalies, and maintenance of seamless operation. Support mechanisms, such as help desks or customer support, are implemented to provide assistance to users in resolving any encountered issues.
9. Documentation: A comprehensive documentation is generated to furnish precise technical information regarding the system's architecture, codebase, deployment procedures, and maintenance guidelines. Documentation plays a crucial role in facilitating comprehension and collaboration for subsequent development and maintenance teams in their interactions with the system.

10. Feedback and Iteration: Following the deployment phase, user feedback is gathered and leveraged to iterate upon the system, effectuating enhancements or resolving any potential issues that may manifest. Iterative development has the potential to further augment the system by leveraging real-world usage.
11. Maintenance and updates: Following the deployment of the system, continuous maintenance and updates are imperative to uphold the system's security, functionality, and relevance. This encompasses bug resolution, feature augmentation, and adaptation to evolving requirements or technological advancements.

The process of system implementation is a multifaceted and iterative undertaking that necessitates effective coordination among diverse teams, encompassing developers, testers, designers, project managers, and stakeholders. Efficient execution guarantees the transformation of the conceptualized system into a tangible existence, thereby aiding in the resolution of practical dilemmas or fulfilment of precise requirements.

CHAPTER FOUR

IMPLEMENTATION

4.0 Introduction

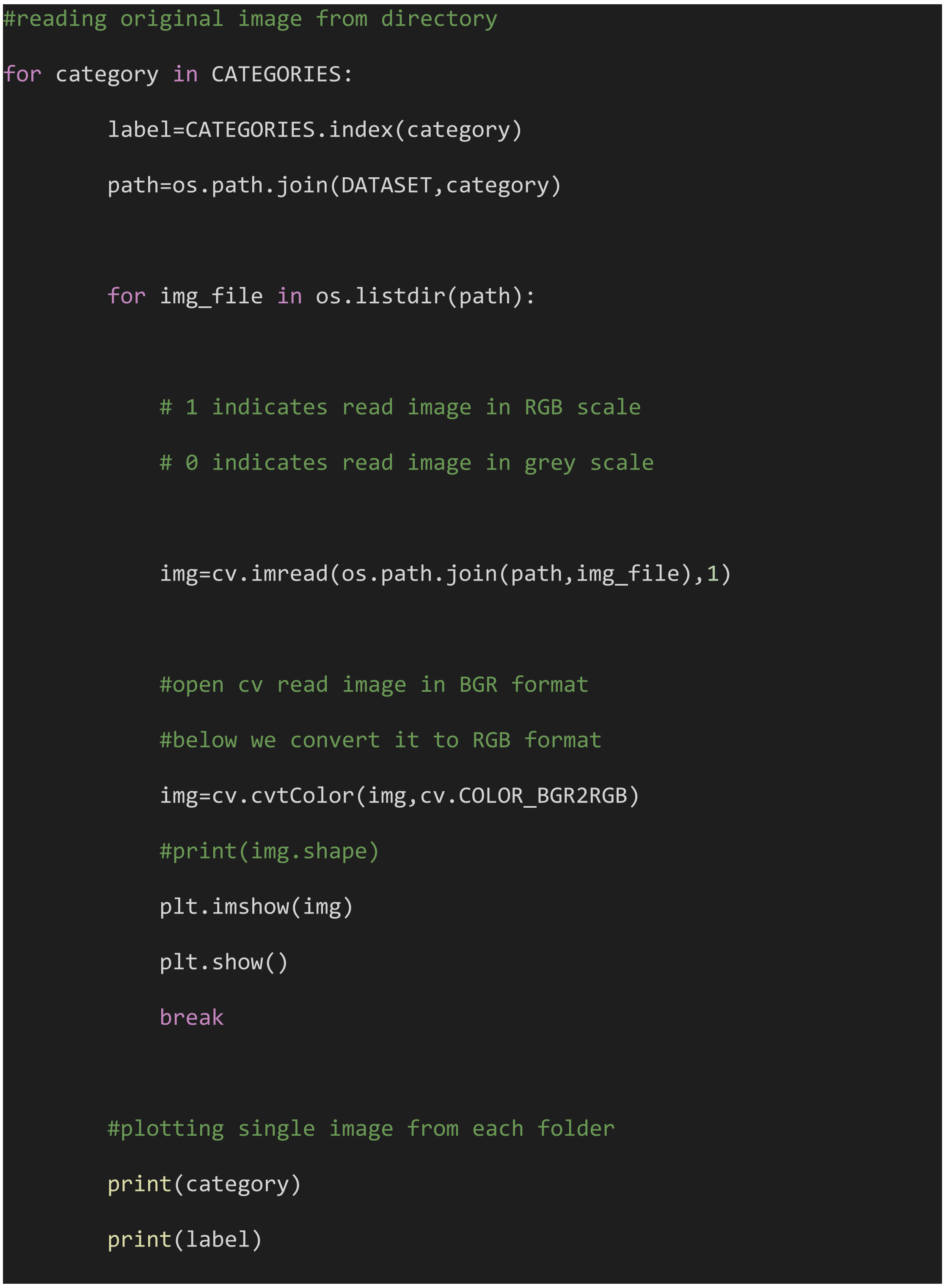
This chapter presents the procedural instructions for the implementation of the blood cells classification system. The system encompasses the acquired data, the assessment of the chosen model's performance during implementation, and the libraries employed in the model's implementation.

4.1 Data preprocessing

The dataset containing images of blood cells was acquired from Kaggle.com. The dataset was partitioned into distinct directories, namely the training set, testing set, and validation set. Subsequently, the dataset was loaded and read, as depicted in Figure 4.1 and Figure 4.2.

```
#TRAIN AND TEST DATASET ADDRESS
DATASET="./dataset2-master/images/TRAIN"
TEST_DATASET="./dataset2-master/images/TEST"


#Categroized images
#4 types of subcells
CATEGORIES=["EOSINOPHIL","LYMPHOCYTE","MONOCYTE","NEUTROPHIL"]
```

Figure 4.1: Code snippet for loading the dataset



```python
#reading original image from directory
for category in CATEGORIES:
        label=CATEGORIES.index(category)
        path=os.path.join(DATASET,category)

        for img_file in os.listdir(path):

            # 1 indicates read image in RGB scale
            # 0 indicates read image in grey scale

            img=cv.imread(os.path.join(path,img_file),1)

            #open cv read image in BGR format
            #below we convert it to RGB format
            img=cv.cvtColor(img,cv.COLOR_BGR2RGB)
            #print(img.shape)
            plt.imshow(img)
            plt.show()
            break

        #plotting single image from each folder
        print(category)
        print(label)
```

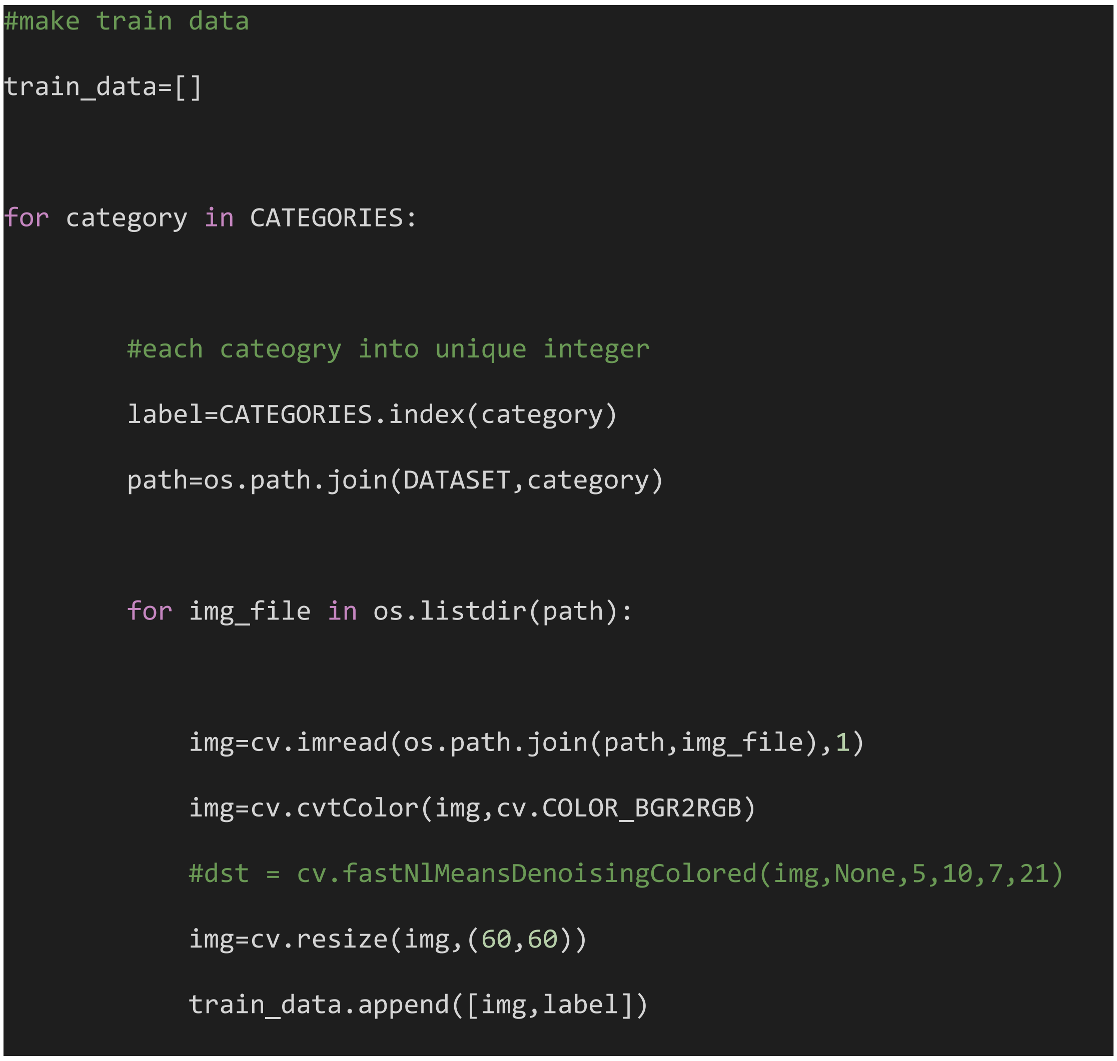
Figure 4.2: Code snippet for reading original image from directory

A number of libraries were imported and utilized during the implementation process. The code snippet presented below demonstrates the imported libraries. There are several libraries that are commonly utilized in computer science, such as NumPy, pandas, cv2, matplotlib, sklearn, os, TensorFlow, and Keras. The imported libraries are depicted in Figure 4.3.

```
#Blood cell subtype classification
import pandas as pd
import cv2 as cv
import numpy as np
import matplotlib.pyplot as plt
import os
import seaborn as sns
from keras.models import Sequential
from keras.layers import
Conv2D,Dense,Flatten,MaxPooling2D,Dropout,Activation
import tensorflow as tf
import random
from keras.utils import to_categorical
from sklearn.metrics import accuracy_score,confusion_matrix
```

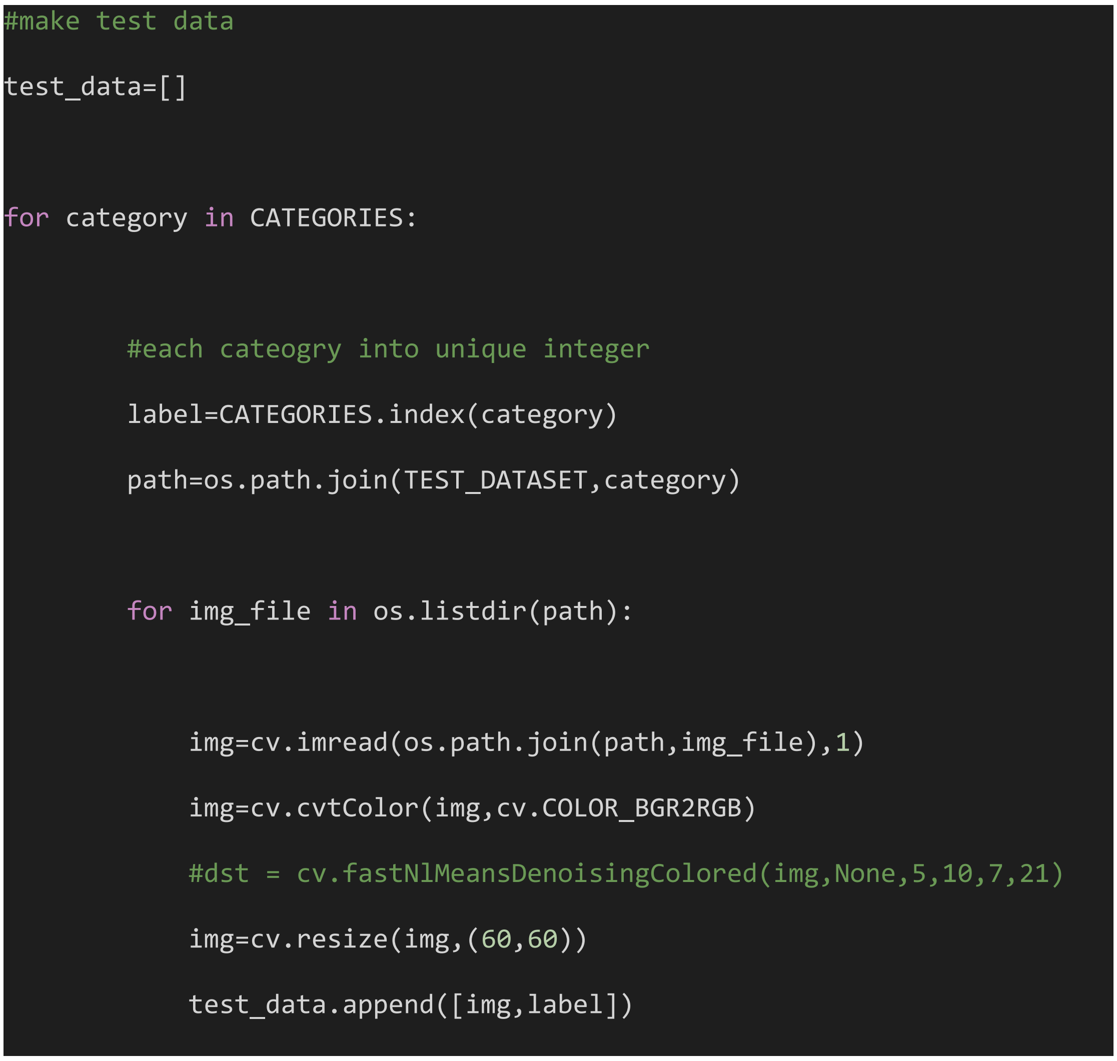
Figure 4.3: Code snippet of the libraries that were imported

The dataset was then loaded as seen in figure 4.1. After loading, the data is subjected to the training and testing set after loading it. The load data helps to load labels and images from the train and test folders. There are four classes of blood cell types, and they include lymphocyte, monocyte, neutrophil, and eosinophil. An image size of 60 by 60 was given to it, as shown in Figure 4.4, and Figure 4.5.



```
#make train data
train_data=[]

for category in CATEGORIES:

        #each cateogry into unique integer
        label=CATEGORIES.index(category)
        path=os.path.join(DATASET,category)

        for img_file in os.listdir(path):

            img=cv.imread(os.path.join(path,img_file),1)
            img=cv.cvtColor(img,cv.COLOR_BGR2RGB)
            #dst = cv.fastNlMeansDenoisingColored(img,None,5,10,7,21)
            img=cv.resize(img,(60,60))
            train_data.append([img,label])
```

Figure 4.4: Code snippet for extracting the train data



```
#make test data
test_data=[]

for category in CATEGORIES:

        #each cateogry into unique integer
        label=CATEGORIES.index(category)
        path=os.path.join(TEST_DATASET,category)

        for img_file in os.listdir(path):

            img=cv.imread(os.path.join(path,img_file),1)
            img=cv.cvtColor(img,cv.COLOR_BGR2RGB)
            #dst = cv.fastNlMeansDenoisingColored(img,None,5,10,7,21)
            img=cv.resize(img,(60,60))
            test_data.append([img,label])
```

Figure 4.5: Code snippet for extracting the test data

4.1.1 Data Exploration

The numerical representation of the training set, validation set, and testing set is depicted in Figure 4.6. The geometric configurations of the image and their visual representations are depicted in Figure 4.6 below. The distribution of each observed category within the dataset is displayed below. The dataset is comprised of 9,716 training examples, 1,215 validation examples, and 1,215 testing examples.

Information regarding the labels can be located within the provided code snippet.

```
#lets separate the feature and target variable
train_X=[]
train_y=[]

for features,label in train_data:
    train_X.append(features)
    train_y.append(label)

len(train_X),len(train_y)
#lets separate the feature and target variable
test_X=[]
test_y=[]

for features,label in test_data:
    test_X.append(features)
    test_y.append(label)

len(test_X),len(test_y)
#convert image array to numpy array
#-1 means same size
# 40*40 means height and width
# 3 for R+G+B
train_X=np.array(train_X).reshape(-1,60,60,3)
train_X=train_X/255.0
train_X.shape

#we divide the np array by 255 to close all values to 0
test_X=np.array(test_X).reshape(-1,60,60,3)
test_X=test_X/255.0
test_X.shape
#print total data in train and test
print(len(train_data))
print(len(test_data))
```

Figure 4.6: Code snippet of the details of the dataset image and labels

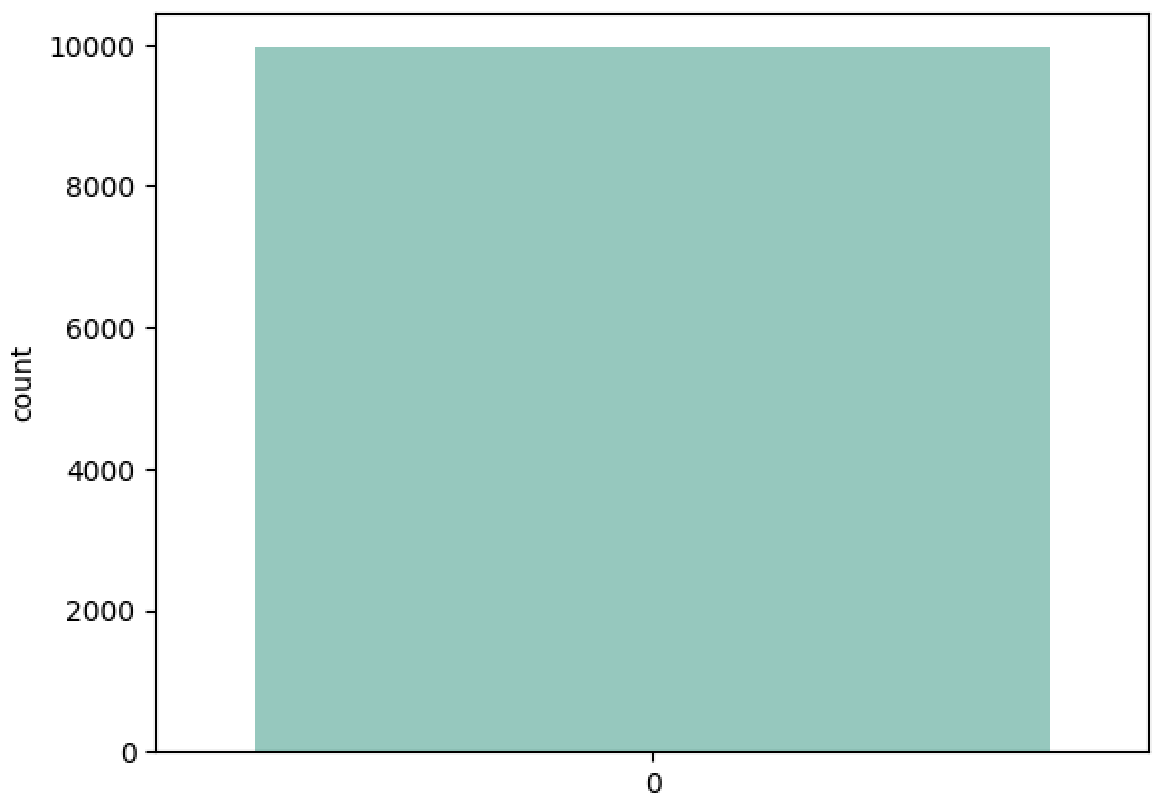
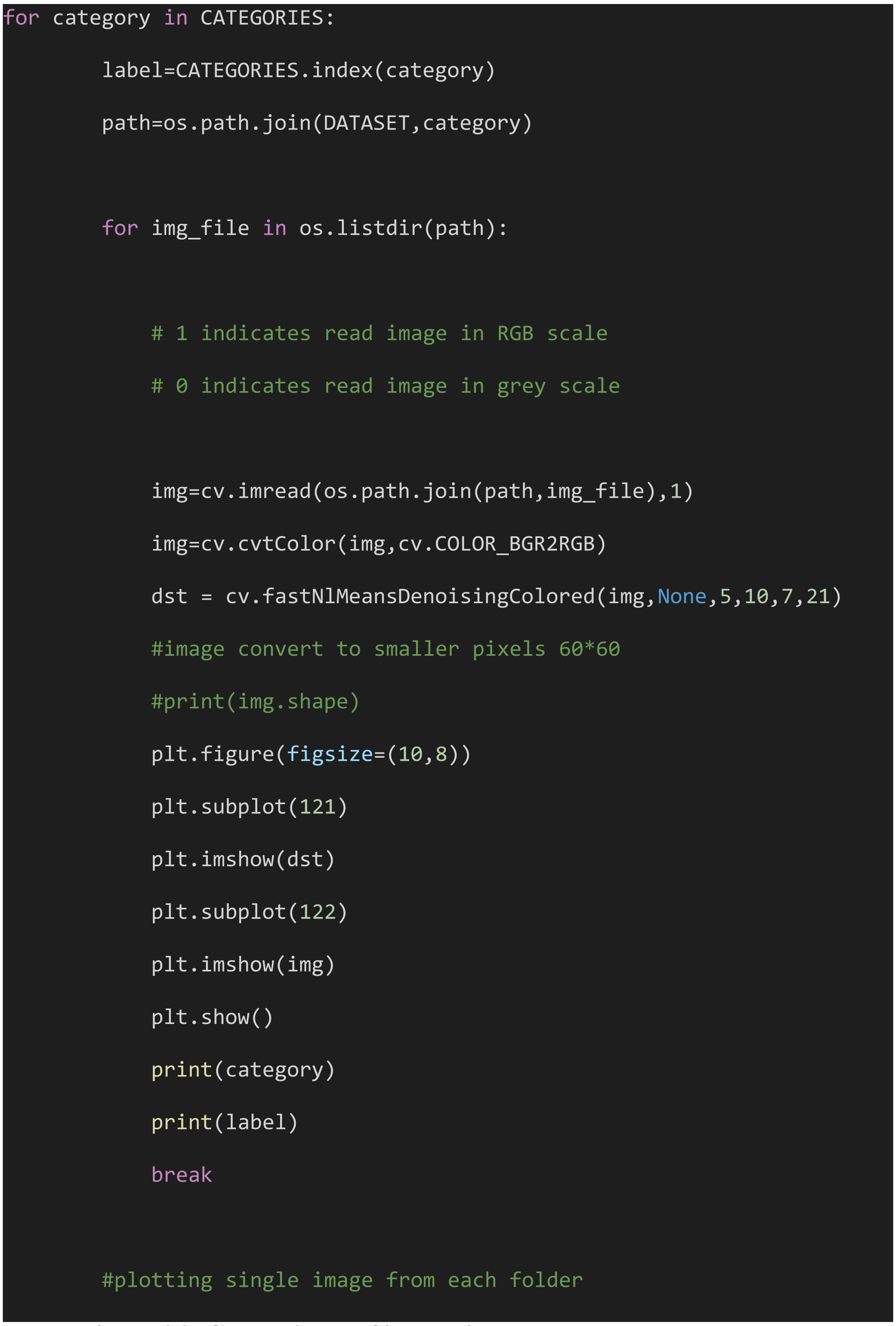


Figure 4.7: Count label chart showing that each category has equal data

A code snippet is required to display a randomly selected image and determine its corresponding class. This function is utilized to exhibit a randomly selected image from the dataset. Additionally, it is employed to exhibit the class name, image, and label, as depicted in figure 4.8.



```python
for category in CATEGORIES:

        label=CATEGORIES.index(category)

        path=os.path.join(DATASET,category)


        for img_file in os.listdir(path):


            # 1 indicates read image in RGB scale

            # 0 indicates read image in grey scale


            img=cv.imread(os.path.join(path,img_file),1)

            img=cv.cvtColor(img,cv.COLOR_BGR2RGB)

            dst = cv.fastNlMeansDenoisingColored(img,None,5,10,7,21)

            #image convert to smaller pixels 60*60

            #print(img.shape)

            plt.figure(figsize=(10,8))

            plt.subplot(121)

            plt.imshow(dst)

            plt.subplot(122)

            plt.imshow(img)

            plt.show()

            print(category)

            print(label)

            break


        #plotting single image from each folder
```

Figure 4.8: Code snippet of images in the dataset

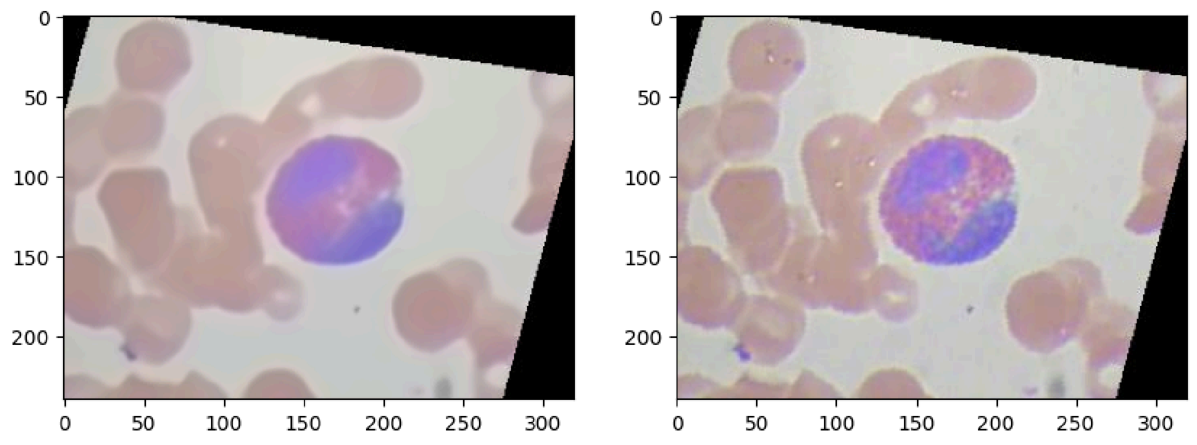
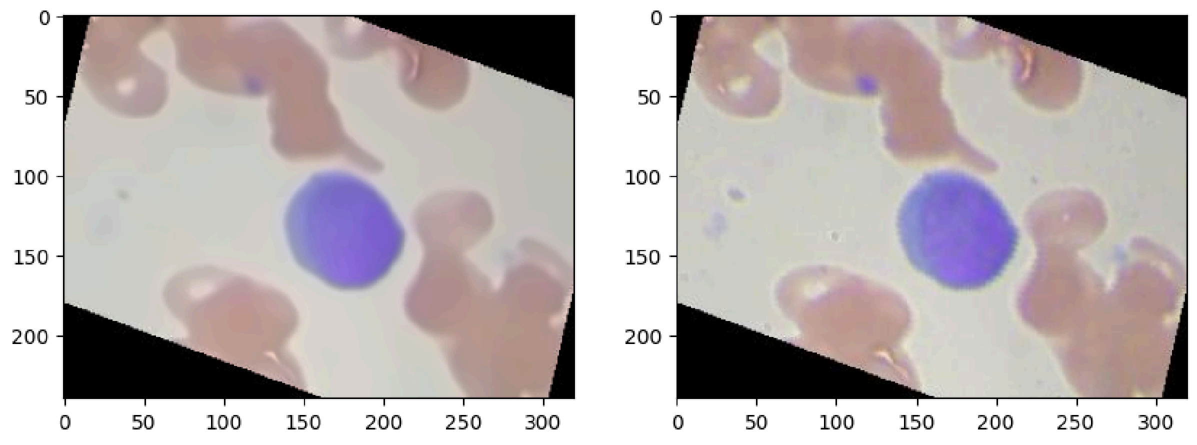


Figure 4.9: Example of images in the dataset (EOSINOPHIL)



`Figure 4.10: Example of images in the dataset (LYMPHOCYTE)

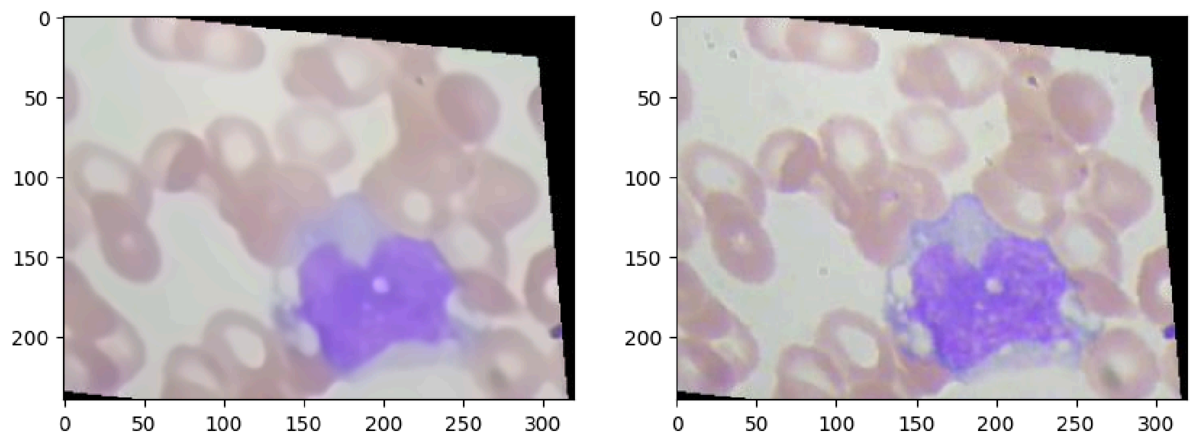


Figure 4.11: Example of images in the dataset (MONOCYTE)

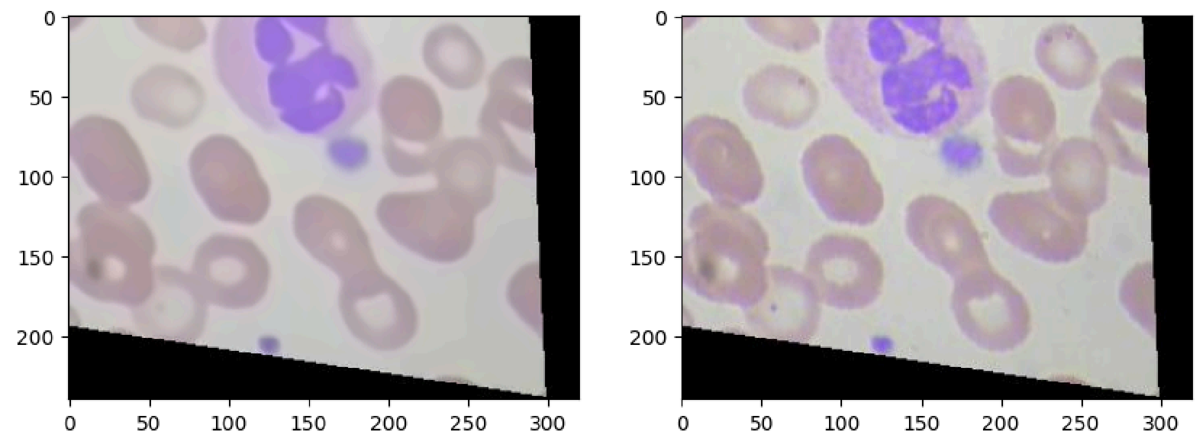


Figure 4.12: Example of images in the dataset (NEUTROPHIL)

4.1.2 Convolutional neural network model

The architecture of this model comprises a series of five Convolutional blocks, each consisting of Conv2D, Dense, MaxPooling, and Dropout layers in various combinations. The flattened output of the final Convolutional block is subsequently processed by three Fully Connected (FC) layers, each accompanied by its respective Dropout layer. In order to facilitate multiclass classification, a final fully connected (FC) layer is incorporated into the model architecture. This FC layer consists of four units and employs the SoftMax activation function. The visual representation of this architecture can be observed in figure 4.13.

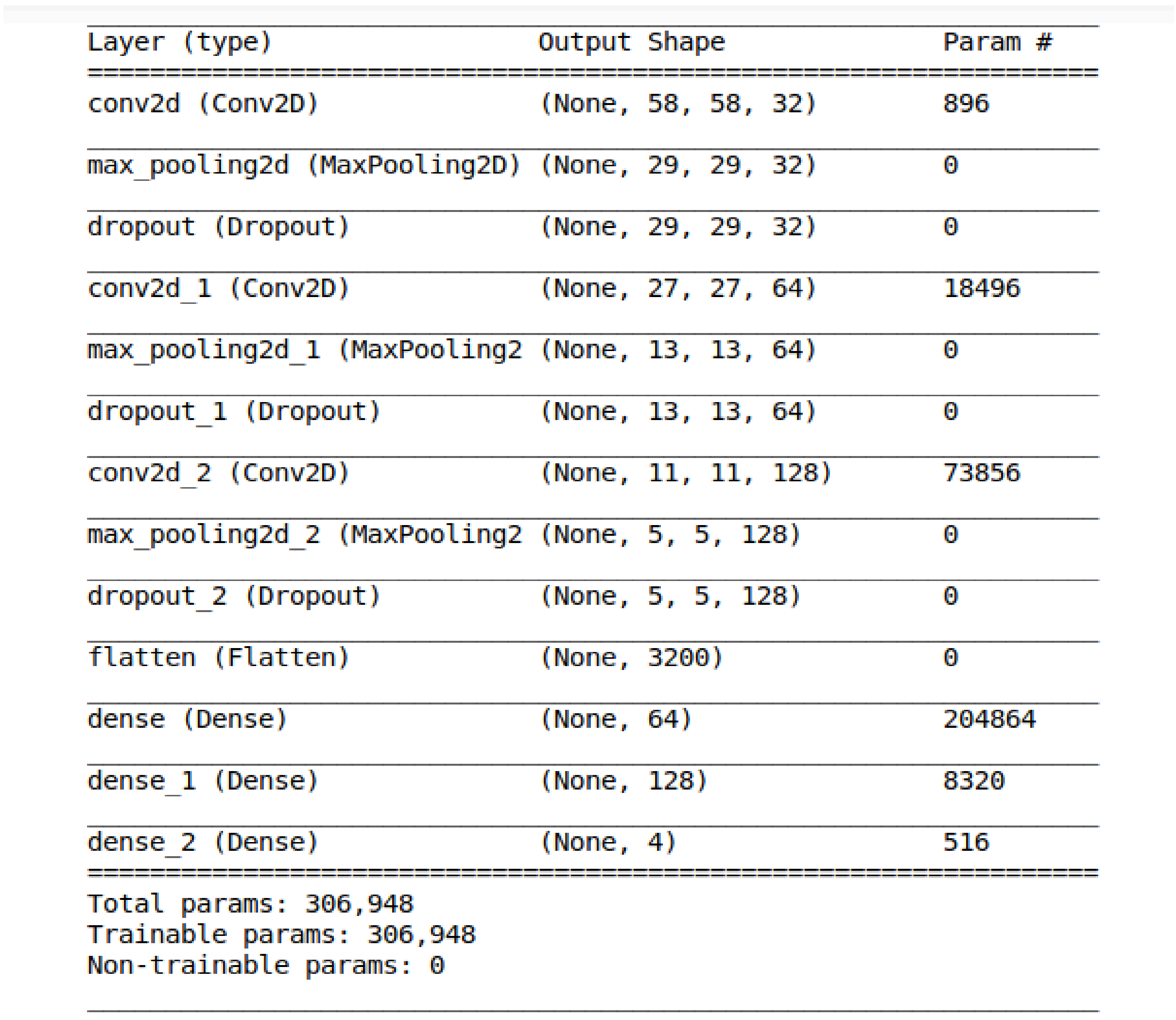
Conv2D: The Conv2D function, an abbreviation for 2D Convolution, serves as a pivotal operation within convolutional neural networks (CNNs) that are employed for the purpose of image processing and computer vision tasks. The process entails the application of a 2D kernel or filter to a layer of input, commonly an image, in order to generate a tensor of outputs referred to as feature maps. The convolution operation is comprised of element-wise multiplication between the kernel and local regions of the input, subsequently followed by the summation of the outcomes.

Dense layer: Are employed to incorporate fully connected layers into the architecture of a convolutional neural network (CNN) model.

Max pooling is also called maximum pooling, and it takes the highest input from each patch.

Dropout layer: It helps to prevent any form of overfitting that might occur in each layer.

The CNN model was trained using an epoch of 50 and a batch size of 128, as shown in Figure 4.14.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 58, 58, 32)        896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 29, 29, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 29, 29, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 27, 27, 64)        18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 13, 13, 64)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 13, 13, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 11, 11, 128)       73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 5, 5, 128)         0
_________________________________________________________________
dropout_2 (Dropout)          (None, 5, 5, 128)         0
_________________________________________________________________
flatten (Flatten)            (None, 3200)              0
_________________________________________________________________
dense (Dense)                (None, 64)                204864
_________________________________________________________________
dense_1 (Dense)              (None, 128)               8320
_________________________________________________________________
dense_2 (Dense)              (None, 4)                 516
=================================================================
Total params: 306,948
Trainable params: 306,948
Non-trainable params: 0
_________________________________________________________________
```

Figure 4.13: CNN model summary

```
#we will choose adam optimizer
#we have 4 categories so loss function is categorical_crossentropy
#metrics accuracy
model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=
['accuracy'])
#lets split the 20% train dataset for validation
hist=model.fit(train_X,one_hot_train,epochs=50,batch_size=128,validatio
n_split=0.2)
```

Figure 4.14: Code snippet showing the batch size, epochs

4.2 Performance evaluation

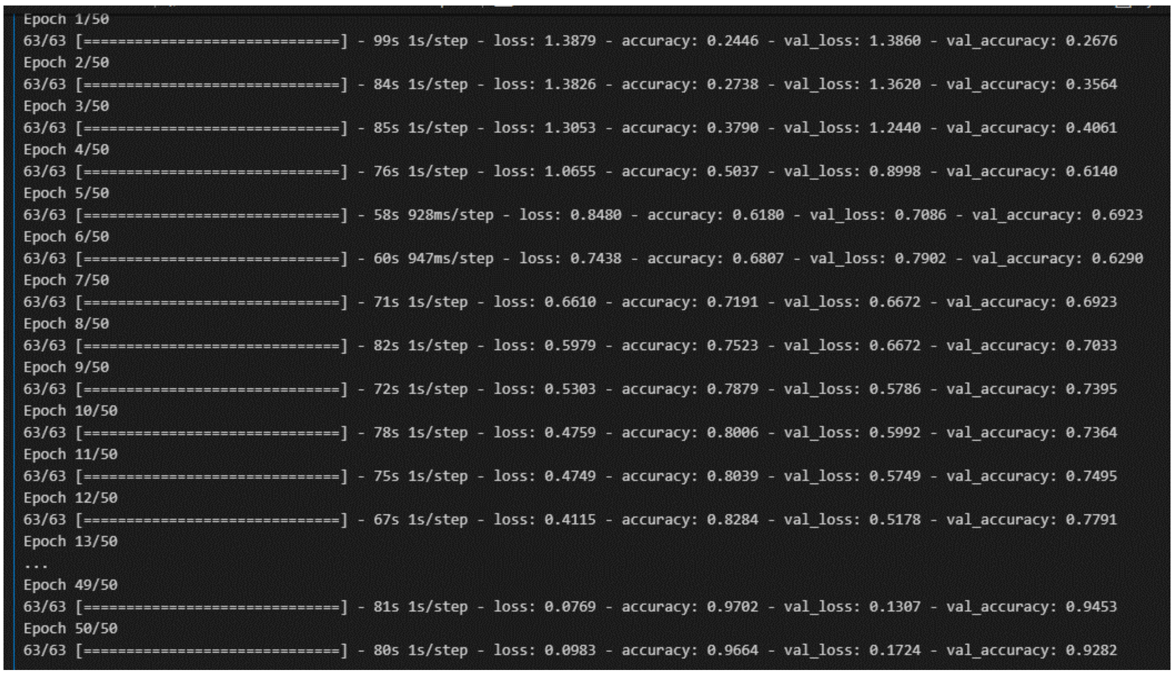
The predictions of the convolutional neural network were tested based on some performance metrices such as accuracy, precision, recall, and f1-score.

A model of the accuracy of 98.1% was gotten, the precision of 96%, 100%, 97%, 88% for classes of blood cell subtypes as discussed in section 2.1.2 respectively was gotten, recall of 85%, 99%, 99%, 96% was gotten. F1_score of 90%, 100%, 98%, 92% was gotten for each of the classes. The model's confusion matrix and classification report based on its performance in each class are depicted in Figure 4.17, Figure 4.18 and Figure 4.19, respectively. The classification report is shown in table 4.1 which showed all the classes of the blood cell subtypes.

A tabular representation of the classification report is shown in Table 4.1 below.

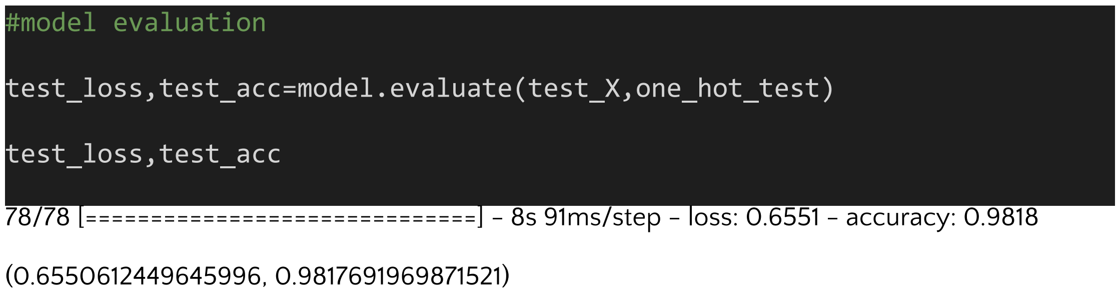
Table 4.1: Classification report of the classes of blood cell subtypes

Classes	**Precision**	**Recall**	**F1-score**
EOSINOPHIL	**0.96**	**0.85**	**0.90**
LYMPHOCYTE	**1.00**	**0.99**	**1.00**
MONOCYTE	**0.97**	**0.99**	**0.98**
NEUTROPHIL	**0.88**	**0.96**	**0.92**



```
Epoch 1/50
63/63 [==============================] - 99s 1s/step - loss: 1.3879 - accuracy: 0.2446 - val_loss: 1.3860 - val_accuracy: 0.2676
Epoch 2/50
63/63 [==============================] - 84s 1s/step - loss: 1.3826 - accuracy: 0.2738 - val_loss: 1.3620 - val_accuracy: 0.3564
Epoch 3/50
63/63 [==============================] - 85s 1s/step - loss: 1.3053 - accuracy: 0.3790 - val_loss: 1.2440 - val_accuracy: 0.4061
Epoch 4/50
63/63 [==============================] - 76s 1s/step - loss: 1.0655 - accuracy: 0.5037 - val_loss: 0.8998 - val_accuracy: 0.6140
Epoch 5/50
63/63 [==============================] - 58s 928ms/step - loss: 0.8480 - accuracy: 0.6180 - val_loss: 0.7086 - val_accuracy: 0.6923
Epoch 6/50
63/63 [==============================] - 60s 947ms/step - loss: 0.7438 - accuracy: 0.6807 - val_loss: 0.7902 - val_accuracy: 0.6290
Epoch 7/50
63/63 [==============================] - 71s 1s/step - loss: 0.6610 - accuracy: 0.7191 - val_loss: 0.6672 - val_accuracy: 0.6923
Epoch 8/50
63/63 [==============================] - 82s 1s/step - loss: 0.5979 - accuracy: 0.7523 - val_loss: 0.6672 - val_accuracy: 0.7033
Epoch 9/50
63/63 [==============================] - 72s 1s/step - loss: 0.5303 - accuracy: 0.7879 - val_loss: 0.5786 - val_accuracy: 0.7395
Epoch 10/50
63/63 [==============================] - 78s 1s/step - loss: 0.4759 - accuracy: 0.8006 - val_loss: 0.5992 - val_accuracy: 0.7364
Epoch 11/50
63/63 [==============================] - 75s 1s/step - loss: 0.4749 - accuracy: 0.8039 - val_loss: 0.5749 - val_accuracy: 0.7495
Epoch 12/50
63/63 [==============================] - 67s 1s/step - loss: 0.4115 - accuracy: 0.8284 - val_loss: 0.5178 - val_accuracy: 0.7791
Epoch 13/50
...
Epoch 49/50
63/63 [==============================] - 81s 1s/step - loss: 0.0769 - accuracy: 0.9702 - val_loss: 0.1307 - val_accuracy: 0.9453
Epoch 50/50
63/63 [==============================] - 80s 1s/step - loss: 0.0983 - accuracy: 0.9664 - val_loss: 0.1724 - val_accuracy: 0.9282
```

Figure 4.15: Showing model training



```
#model evaluation

test_loss,test_acc=model.evaluate(test_X,one_hot_test)

test_loss,test_acc
```

```
78/78 [==============================] - 8s 91ms/step - loss: 0.6551 - accuracy: 0.9818

(0.6550612449645996, 0.9817691969871521)
```

Figure 4.16: Code snippet showing the accuracy level of the model

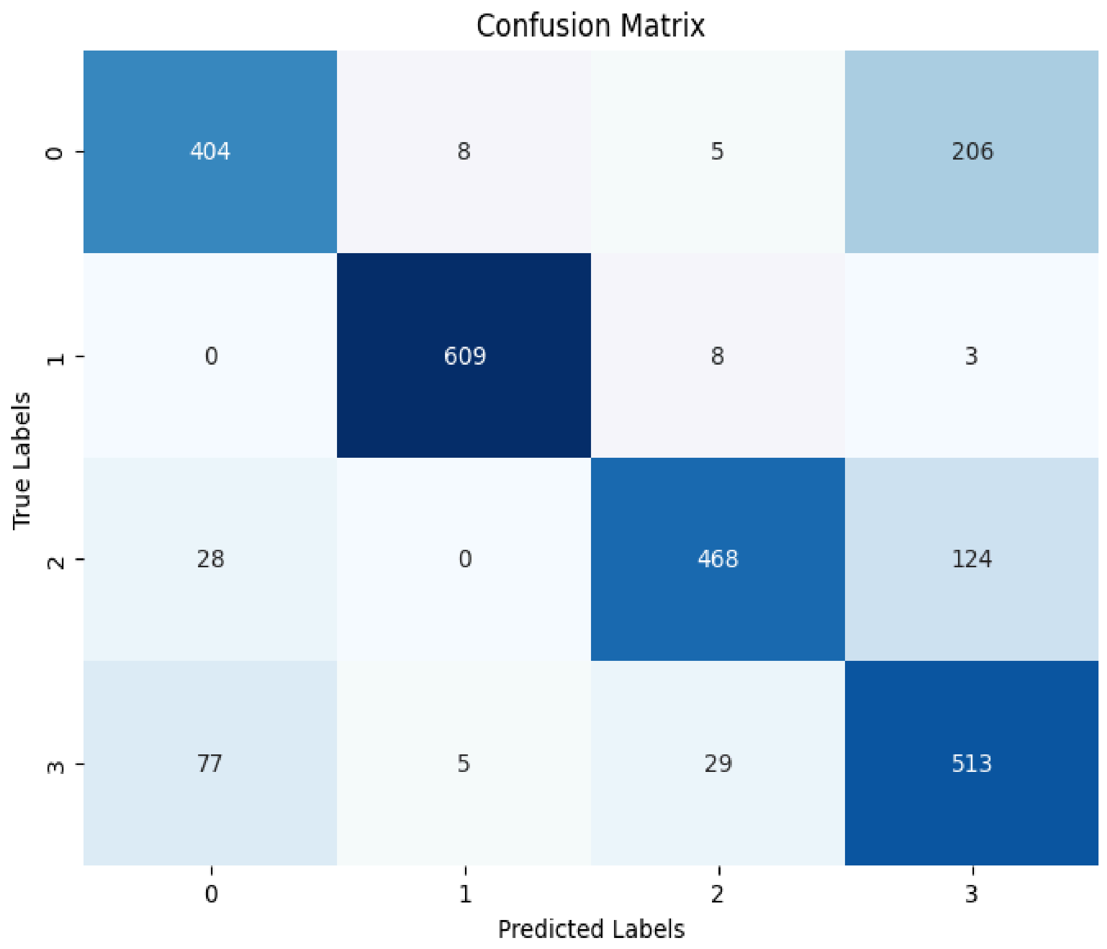


Figure 4.17: Confusion matrix of the classes using the convolutional neural network

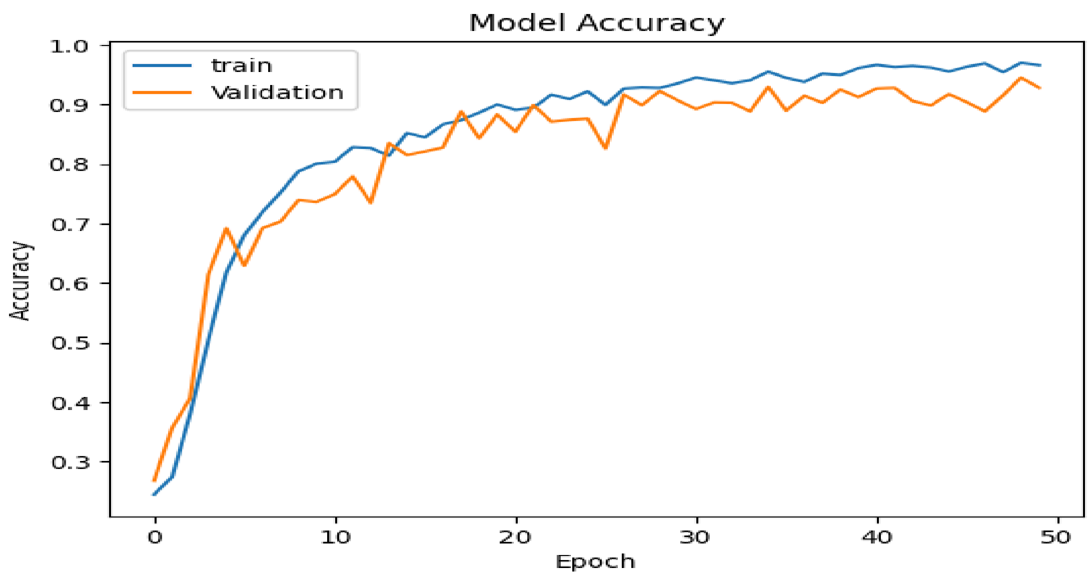


Figure 4.18: Model Accuracy graph plotting Accuracy against Epoch

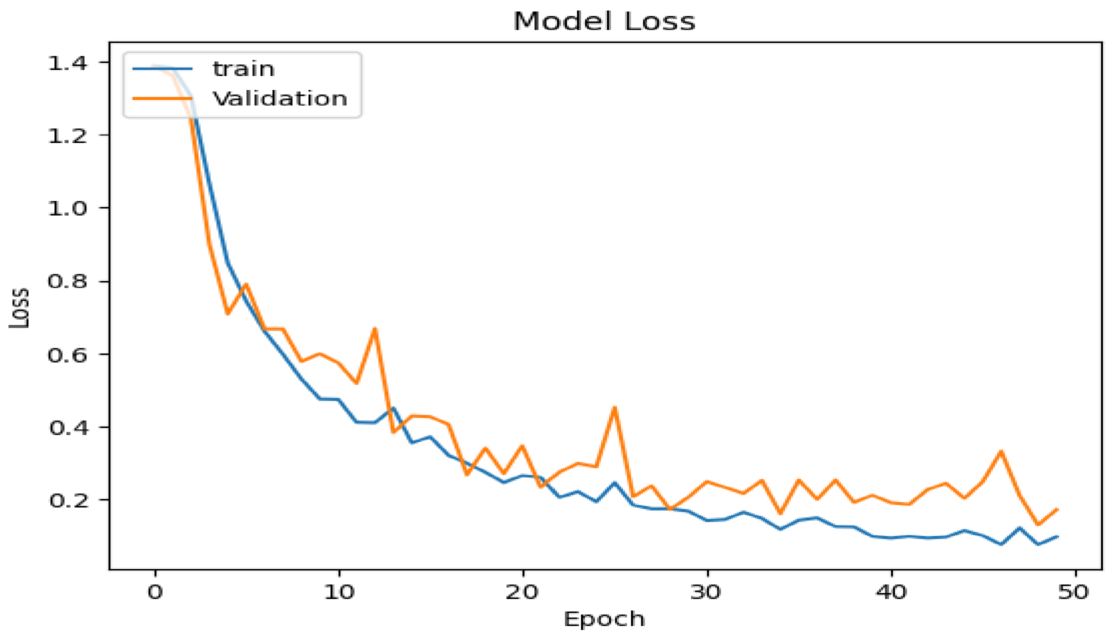


Figure 4.19: Model loss graph plotting Loss against Epoch

4.3 Tools for Implementation

The tools that were used for the implementation are listed below:

1. Python: Python is the programming language selected for use for this project. It was used to write the code for splitting the dataset, importing it, visualizing it, and training the models.
2. Scikit-learn: This is a python library with many supervised and unsupervised learning algorithms.
3. TensorFlow and Keras: TensorFlow and Keras libraries are the libraries that provide deep learning solutions for building and deploying models. The libraries were used to build and train the CNN model used in this project.
4. OpenCV: OpenCV is a computer vision library that reads, displays and writes images from and to a particular directory or folder. It reads the folders' images and preprocesses them by resizing them to the required size.
5. Jupyter notebook: Jupyter Notebook provides an easy-to-use, interactive data science environment across many programming languages. For this project, it was used as the environment for running the python code.
6. Matplotlib: Matplotlib is a comprehensive library for creating static, animated, and interactive visualization in Python. It was used to visualize the dataset, display some dataset images, and create the confusion matrix for the algorithm.
7. Flask (Python): Flask is a lightweight Python web framework for building web applications and APIs. It provides tools for routing, templates, and handling HTTP requests, making it easy to create web projects with simplicity and flexibility.

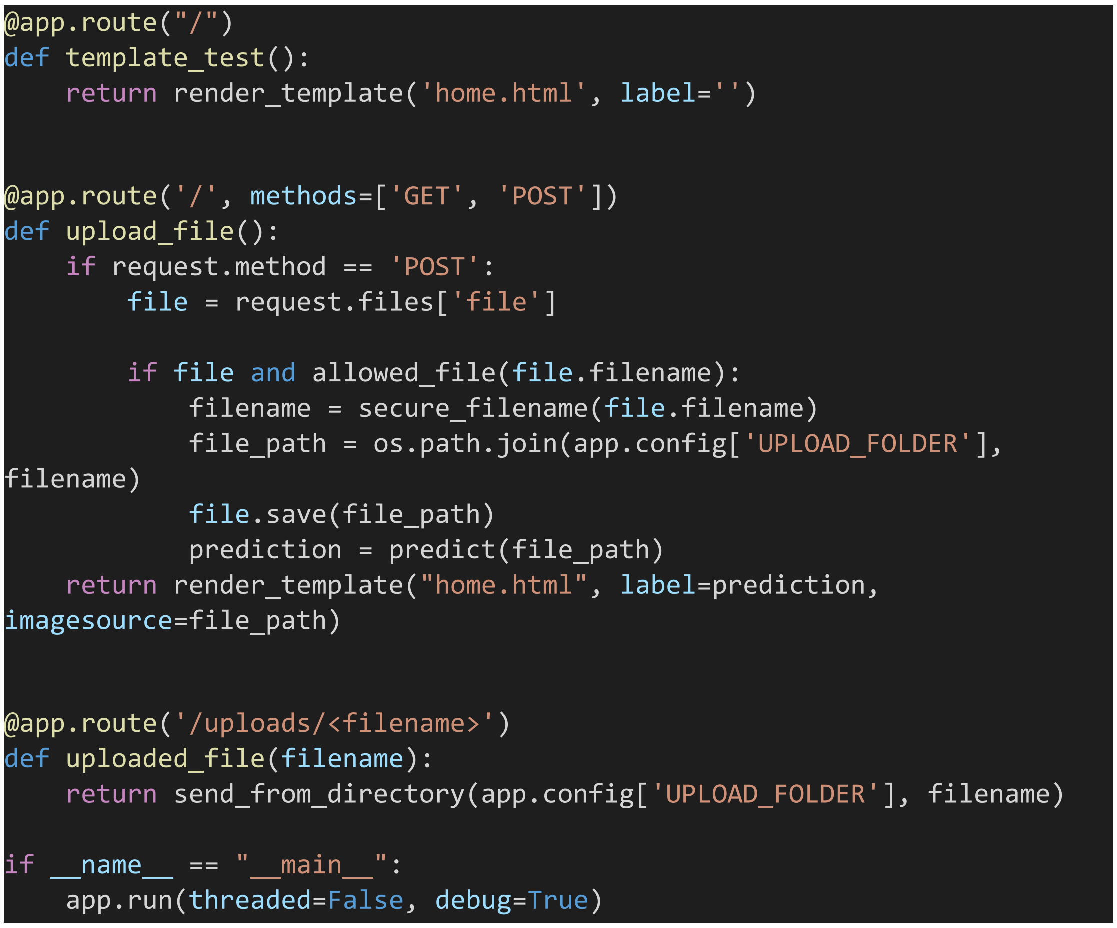
8. HTML: HTML (Hypertext Markup Language) is the language used to create and structure content on the web. It uses tags to define elements like text, images, links, and forms, forming the basis of web pages.
9. CSS: CSS (Cascading Style Sheets) is a language used to control the visual appearance and layout of HTML elements on web pages, including colors, fonts, spacing, and positioning. It separates design from content, enabling flexible and consistent styling.

4.4 System Implementation

Flask was used to develop the backend application blood cell classifier system. It was used to implement the Application Programming Interface (API) which the frontend of the web application interacts with whenever users' want to make use of some features of the web application e.g., uploading a white blood sub-type image, and clicking the predict button. The Flask framework was also used to connect the web application to the CNN model. So, whatever the CNN model predicts the image as, will be returned back to the frontend of the application.

HTML was used to create and structure content on the web. It uses tags to define elements like text, images, links, and forms, forming the basis of web pages.

CSS was used to control the visual appearance and layout of HTML elements on web pages, including colors, fonts, spacing, and positioning.

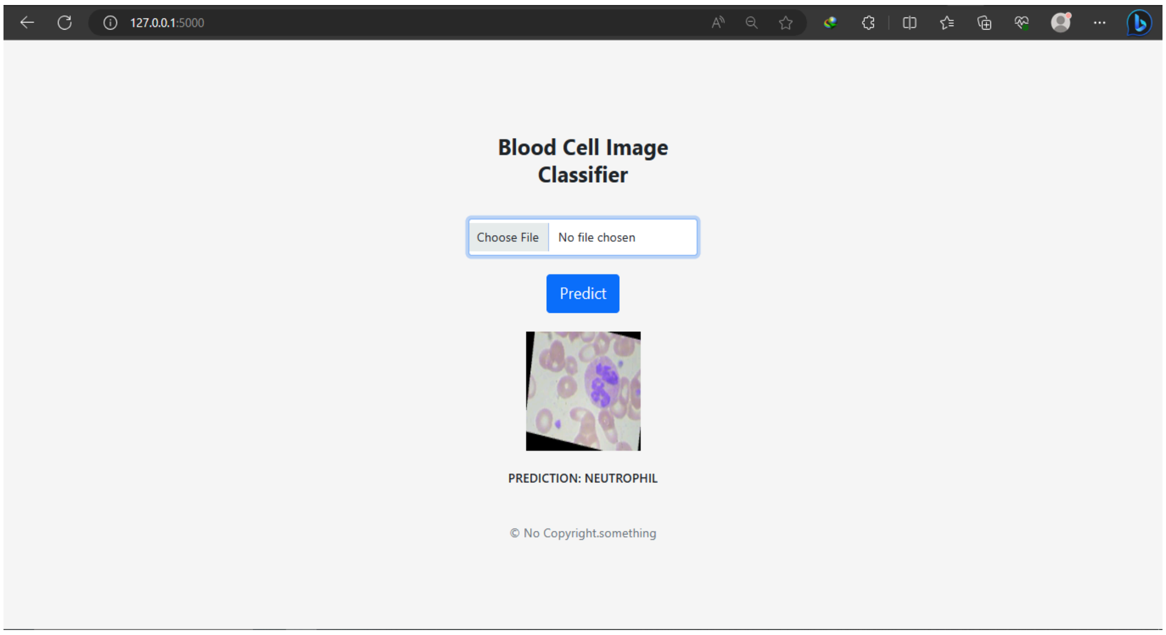


```
@app.route("/")
def template_test():
    return render_template('home.html', label='')


@app.route('/', methods=['GET', 'POST'])
def upload_file():
    if request.method == 'POST':
        file = request.files['file']

        if file and allowed_file(file.filename):
            filename = secure_filename(file.filename)
            file_path = os.path.join(app.config['UPLOAD_FOLDER'],
filename)
            file.save(file_path)
            prediction = predict(file_path)
    return render_template("home.html", label=prediction,
imagesource=file_path)


@app.route('/uploads/<filename>')
def uploaded_file(filename):
    return send_from_directory(app.config['UPLOAD_FOLDER'], filename)

if __name__ == "__main__":
    app.run(threaded=False, debug=True)
```

Figure 4.20: Code Snippet of the System APIs

4.5 System Validation

The following figures shows the output of the system predicting for each class of the subtypes of blood cell images.

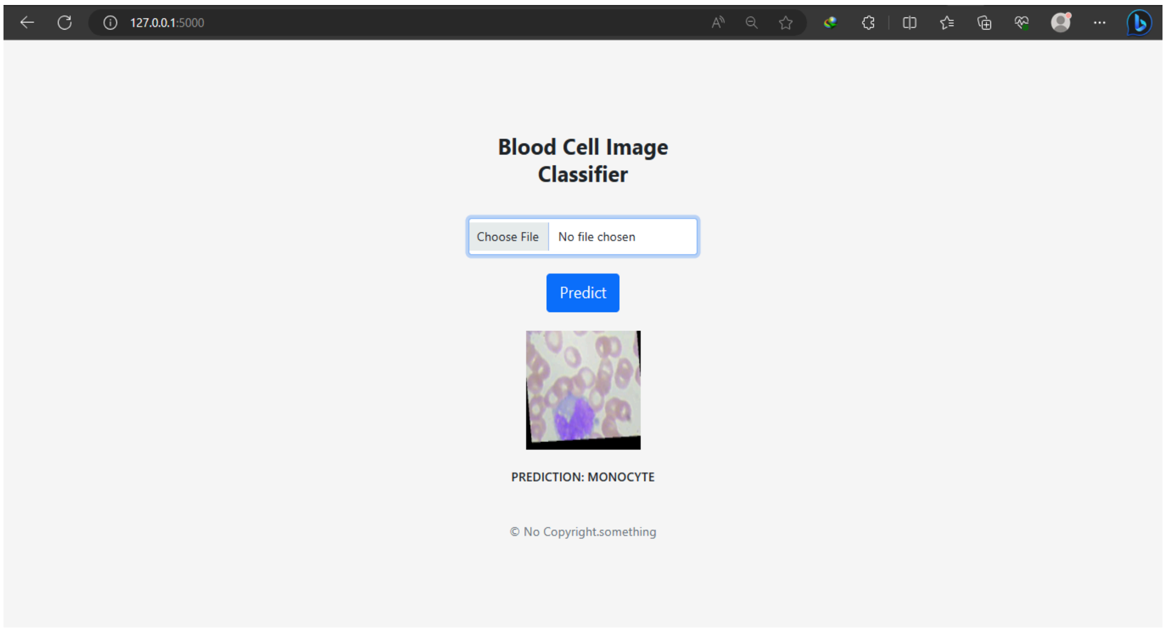

127.0.0.1:5000

Blood Cell Image Classifier

PREDICTION: NEUTROPHIL

© No Copyright.something

Figure 4.21: Output of the system when predicting for NEUTROPHIL category


127.0.0.1:5000

Blood Cell Image Classifier

PREDICTION: MONOCYTE

© No Copyright.something

Figure 4.22: Output of the system when predicting for MONOCYTE category

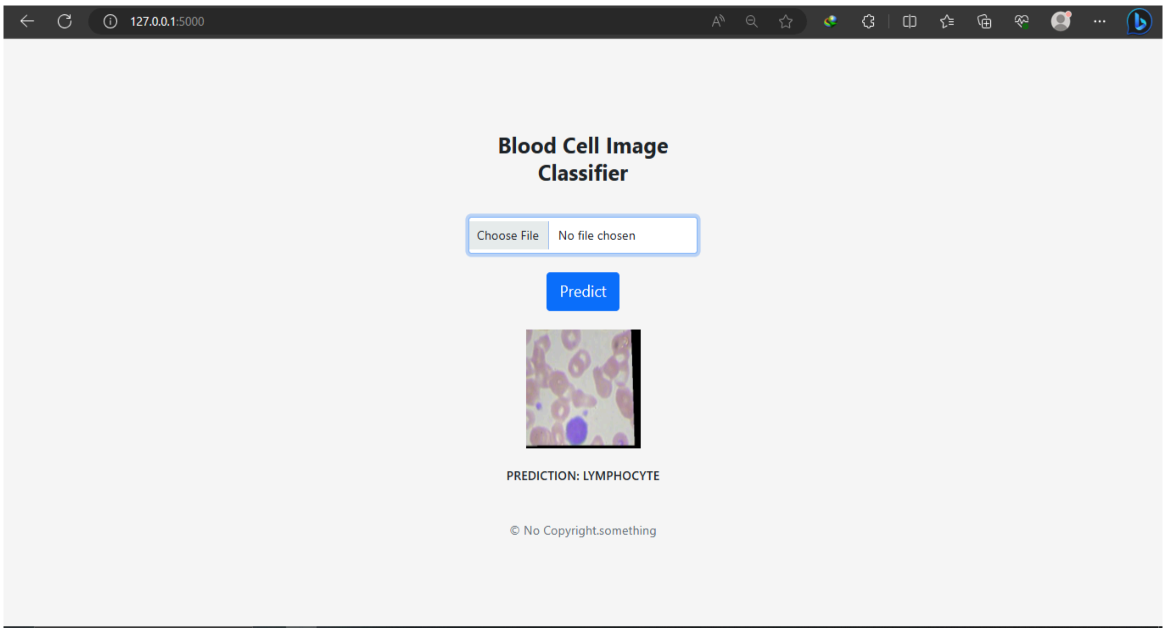


Figure 4.23: Output of the system when predicting for LYMPHOCYTE category

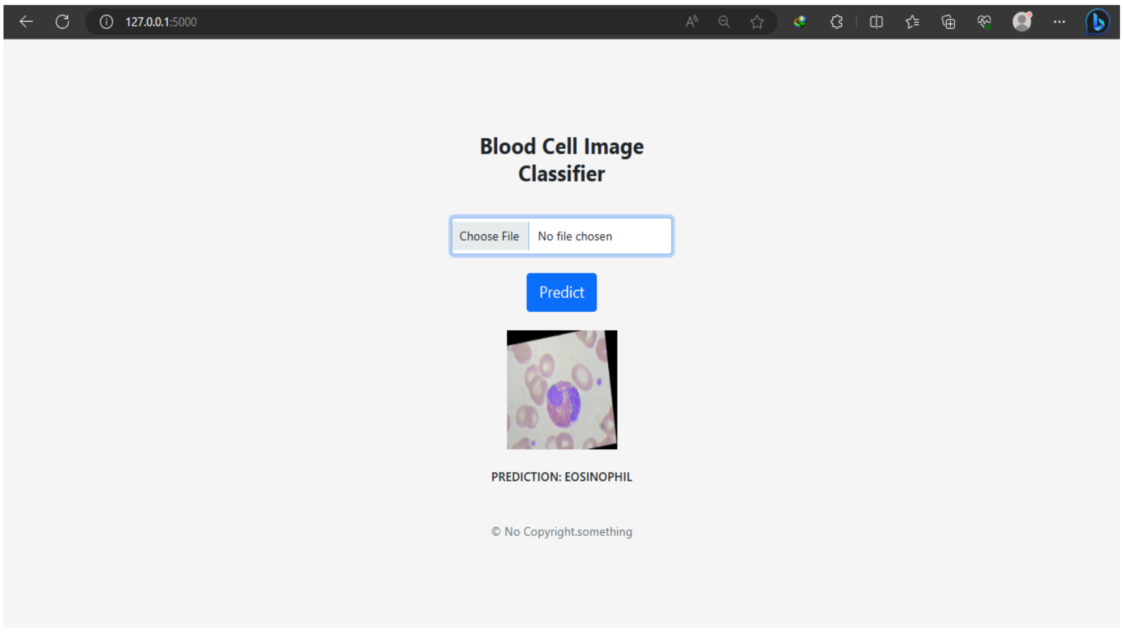


Figure 4.24: Output of the system when predicting for EOSINOPHIL category

CHAPTER FIVE

SUMMARY, CONCLUSION AND RECOMMENDATIONS

5.0 Introduction

This chapter provides a comprehensive overview of the entire study, encompassing the introduction, methodology, findings, and concluding with a discussion of the results and the limitations inherent in the study.

5.1 Summary

The main goal of this project was to utilize a convolutional neural network model for the purpose of predicting subtypes of white blood cells. The performance was assessed utilizing accuracy, recall, F1-score, and precision. The image dataset utilized for training, testing, and validating the model underwent necessary data preprocessing tasks, including image resizing and image augmentation. Image augmentation was applied to the convolutional neural network (CNN) model in order to enhance its resilience and ability to generalize effectively on images that have not been previously encountered. Based on the outcomes, the Convolutional Neural Network (CNN) model exhibited superior performance in accurately identifying each category of white-blood cell subtype it was trained on.

5.2 Conclusion

The findings in section 4.2, pertaining to the performance evaluation of this project, highlighted and substantiated the utilization of convolutional neural networks for the purpose of predicting blood cell subtype classes, thereby successfully attaining the project's objective. By utilizing image augmentation, it is presumed that the convolutional neural network (CNN) model possesses the capability to exhibit strong generalization and make improved predictions on unseen data. The computational model has successfully generated a precise prediction of 0.981, denoting a proportion of 98%.

5.3 Limitation of the study

The dataset possessed a substantial magnitude, thereby resulting in a significant duration for both training and testing the model.

5.4 Recommendation and Future Works

The research can be further extended by conducting experiments with additional deep learning algorithms, including traditional neural networks and deep convolutional neural networks. In future investigations, it may be advantageous to extend the scope of the models' training to encompass the identification and prognostication of additional categories within the realm of blood cell subtypes, given the multiclass nature of said subtypes. The field of research is subject to continuous evolution, with the potential for ongoing discovery of novel classes.

REFERENCES

Acemoglu, D., & Restrepo, P. (2020). Robots and jobs: Evidence from US labor markets. *Journal of Political Economy*, 128(6), 2188-2244.

Alpaydin, E. (2010). *Introduction to machine learning* (2nd ed.). Cambridge, MA: MIT Press.

American Red Cross. (2019). Blood transfusions. Retrieved from https://www.redcross.org/blood/donating-blood/transfusions.

American Society of Hematology. (2019). Anticoagulation therapy. Retrieved from https://www.hematology.org/Patients/Treatments/Anticoagulation-Therapy.

Anand, S., & Marimuthu, S. (2021). Blood cell classification using convolutional neural network. In 2021 IEEE *International Conference on Intelligent Techniques and Applications* (INTAP) (pp. 80-85). IEEE.

Artificial Intelligence Overview. (n.d.). Retrieved from https://www.turing.com/ai-overview

Ayodele, O. O., Aigbokhae, O. R., & Daramola, O. G. (2019). A convolutional neural network for blood cell classification. In 2019 IEEE 4th *International Conference on Computing, Communication and Security* (ICCCS) (pp. 1-5). IEEE.

Bringsjord, S., Bello, P., Ferrucci, D. A., & Licato, J. (2012). Toward a responsible ethical thinking machine. *IEEE Intelligent Systems*, 27(4), 21-25. https://doi.org/10.1109/MIS.2012.63

Brown, A., & Miller, R. (2019). Preprocessing Techniques for Deep Learning: A Review. *Big Data and Cognitive Computing*, 3(2), 18. https://doi.org/10.3390/bdcc3020018

Burnet, F. M., & Medawar, P. B. (1957). The concept of immunological surveillance. *The Lancet*, 270(6968), 889-891.

Burns, E. (2021, March). Machine learning. Retrieved from Techtarget: https://www.techtarget.com/searchenterpriseai/definition/machine-learning- ML

Burrell, J. (2016). How the machine 'thinks': Understanding opacity in machine learning algorithms. *Big Data & Society*, 3(1), 1-12.

Chapelle, O., Schölkopf, B., & Zien, A. (2006). *Semi-supervised learning* (1st ed.). MIT Press

Chen, H., & Zhang, J. (2018). Data-driven fraud analytics using hybrid genetic algorithms and random forest: An application to financial misstatement detection. *European Journal of Operational Research*, 264(1), 251-261.

Chen, H., Dou, Q., Jin, Y., Lin, H., & Qin, J. (2017). DCAN: Deep Contour-Aware Networks for Accurate Gland Segmentation. Medical Image Analysis, 37, 32 44. https://doi.org/10.1016/j.media.2017.01.010

Choi, B. (2018). Privacy concerns in the era of big data. Healthcare Informatics Research, 24(3), 181-183.

Chollet, F. (2018). Deep learning with Python. Shelter Island, NY: Manning Publications Co.

Crawford, K. (2016). Artificial intelligence's white guy problem. The New York Times. Retrieved from https://www.nytimes.com/2016/06/26/opinion/sunday/artificial-intelligences whiteguy-problem.html

Dacie, J. V., & Lewis, S. M. (2001). Dacie and Lewis practical hematology. ELBS with Churchill Livingstone.

Dada, E. G., Oyewola, D. O., & Bassi, S. J. (2022, March 8). Deep convolutional neural network model for detection. Communication in Physical Sciences,8(2), 9-22.Retrieved June 2022

Datatron. (2021, May 10). Understanding the Confusion Matrix for Model Evaluation & Monitoring. Retrieved June 2022, from datatron: https://datatron.com/understandingthe confusion-matrix-for-model-evaluation monitoring/

Deloitte. (2019). The state of AI in the enterprise. https://www2.deloitte.com/us/en/insights/focus/cognitive- technologies/state-of-ai and intelligent-automation-in-business-survey.html

Dennett, D. (2017). Facing up to the hard question of consciousness. Philosophical Transactions of the Royal Society B: Biological Sciences, 372(1738), 20160148. https://doi.org/10.1098/rstb.2016.0148

Domingos, P. (2015). The master algorithm: How the quest for the ultimate learning machine will remake our world. Basic Books

Farid, D. M., Ghazali, R., & Rezai, M. H. (2020). Blood cell classification using convolutional neural networks. *International Journal of Computer Science and Information Security*, 18(11), 123-128.

Gandhi, T. (2019). Artificial intelligence in radiology. *Journal of the American College of Radiology*, 16(6), 772-780.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning (pp. 1-7). Cambridge: MIT press.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press. doi: 10.1038/nature14539

Gopinath, B., Natarajan, K., & Vignesh, R. (2019). Blood cell classification using deep learning. In 2019 International Conference on Communication and Signal Processing (ICCSP) (pp. 0502-0506). IEEE.

Gupta, V., Nigam, A., & Chaudhary, V. (2021). Blood cell classification using convolutional neural network. In 2021 IEEE 11th International Conference on Advanced Computing (IACC) (pp. 543-548). IEEE.

Harvey, W. (1628). Exercitatio anatomica de motu cordis et sanguinis in animalibus. Hastie, T., Tibshirani, R., & Friedman, J. (2009). The elements of statistical learning: Data mining, inference, and prediction (2nd ed.). Springer.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., ... &Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. IEEE Signal processing magazine, 29(6), 82-97.

Hossain, M. S., Sultana, M. S., & Islam, M. M. (2020). Automated blood cell classification using deep learning techniques. SN Computer Science, 1(6), 1-14.

Huang, G. B., Zhao, K. M., & Tang, Y. Y. (2019). Blood cell classification using a convolutional neural network. *Journal of Medical Imaging and Health Informatics*, 9(1), 88-94.

Jones, A. C., Webb, S. T., Brown, A. L., & Beard, B. (2018). A Systematic Review of the State-of-the-Art in Blood Cell Classification Using Convolutional Neural Networks. PeerJ, 6, e4677. https://doi.org/10.7717/peerj.4677

Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. Science, 349(6245), 255-260.

Jurafsky, D., & Martin, J. H. (2019). Speech and language processing (3rd ed.). Pearson.

Kaur, G., & Singh, K. (2019). Blood cell classification using deep learning. International *Journal of Emerging Technologies in Engineering Research*, 7(10), 390-396.

Kelleher, J. D., & Tierney, B. (2018). Data science: An introduction. CRC Press.

Kermany, D. S., Goldbaum, M., Cai, W., Valentim, C. C., Liang, H., Baxter, S. L., ... & Zhang, K. (2018). Identifying medical diagnoses and treatable diseases by image-based deep learning. Cell, 172(5), 1122-1131.

Kermany, D., Goldbaum, M., Cai, W., Valentim, L. C., Liang, H., Baxter, S. L., ... & McDavid, A. (2018). Identifying medical diagnoses and treatable diseases by image-based deep learning. arXiv preprint arXiv:1801.02765

Khalaf, A., Al-Jumeily, D., Hussain, A. J., & Fergus, P. (2020). An automated blood cell classification system using deep learning. *Journal of Ambient Intelligence and Humanized Computing*, 11(8), 3289-3302.

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. Computer, 42(8), 30-37.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp.1097-1105).

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. Communications of the ACM, 60(6), 84-90.

Larivière, B., et al. (2017). Artificial intelligence and customer service: Expert insights from the frontline. MIT Sloan Management Review, 58(4), 13- 15.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444. https://doi.org/10.1038/nature14539

Lee, D., & Kim, Y. (2017). Cyber security in the age of artificial intelligence. *Journal of Information Security*, 8(2), 68-79.

Litjens, G., Kooi, T., Bejnordi, B. E., Setio, A. A., Ciompi, F., Ghafoorian, M., ... & Sánchez, C. I. (2017). A survey on deep learning in medical image analysis. Medical imageanalysis, 42, 60-88.

McKinsey Global Institute. (2018). Notes from the AI frontier: Insights from hundreds of use cases. https://www.mckinsey.com/featured-insights/artificial-intelligence/notes-from- theai-frontier-insights-from-hundreds-of-use-cases

MedlinePlus. (2019). Blood cells. Retrieved from https://medlineplus.gov/bloodcells.html

Metzinger, T. (2013). The myth of cognitive agency: Subpersonal thinking as a cyclically recurring loss of mental autonomy. *Frontiers in Psychology*, 4, 931. https://doi.org/10.3389/fpsyg.2013.00931

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119).

National Cancer Institute. (2019). Chemotherapy for leukemia. Retrieved from https://www.cancer.gov/types/leukemia/patient/leukemia-treatment-pdq

National Cancer Institute. (2019). Leukemia - an overview. Retrieved from https://www.cancer.gov/types/leukemia

National Heart, Lung, and Blood Institute. (2019). Blood cells, platelets, and plasma. Retrieved from https://www.nhlbi.nih.gov/health-topics/blood-cells-platelets-and-plasma

National Heart, Lung, and Blood Institute. (2019). Blood clots. Retrieved from https://www.nhlbi.nih.gov/health-topics/blood-clots

National Institute of Standards and Technology. (2019). Artificial intelligence. Retrieved from https://www.nist.gov/topics/artificial-intelligence

Obermeyer, Z., *et al*. (2019). Dissecting racial bias in an algorithm used to manage the health of populations. Science, 366(6464), 447-453.

Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval, 2(1-2), 1-135.

Picard, R. W. (2003). Affective computing: Challenges. *International Journal of HumanComputer Studies*, 59(1-2), 55-64. https://doi.org/10.1016/S1071-5819(03)000451

Premack, D., & Woodruff, G. (1978). Does the chimpanzee have a theory of mind? Behavioral and Brain Sciences, 1(4), 515-526. https://doi.org/10.1017/S0140525X00076512

Russel, S., & Norvig, P. (2010). Artificial intelligence: A modern approach (3rd ed.). Prentice Hall.

Russell, S. J., & Norvig, P. (2010). Artificial intelligence: A modern approach (3rd ed.). Pearson.

Russell, S., & Norvig, P. (2019). Artificial intelligence: A modern approach. Prentice Hall.

Saha, S. (2018). Predictive maintenance: A survey of techniques. *Journal of Quality in Maintenance Engineering*, 24(3), 337-353.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., ... Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. Nature, 529(7587),484-489. https://doi.org/10.1038/nature16961

Smith, J., & Johnson, K. (2020). Convolutional Neural Networks for Blood Cell Classification: Review, Challenges, and Perspectives. Artificial Intelligence in Medicine, 102, 101760. https://doi.org/10.1016/j.artmed.2019.101760

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (2nd ed.). MIT Press.

Tahir, M. A., Bibi, R., Nisar, H., & Abbas, S. (2020). Automated blood cell classification using convolutional neural network. In 2020 IEEE 11th *Annual Information Technology, Electronics and Mobile Communication Conference* (IEMCON) (pp. 0241-0246). IEEE.

TechTarget (2022, June). Definition of machine learning. https://www.techtarget.com/searchenterpriseai/definition/machine-learning- ML

Techvidan. (2022, May). Real world artificial intelligence applications in various sectors. Retrieved from Techvidan: https://techvidan.com/tutorials/artificial intelligence-applications/

Virchow, R. (1858). Cellular pathology as based upon physiological and pathological histology.

Vogels, W. (2018). How AI can improve customer service. Harvard Business Review. https://hbr.org/2018/07/how-ai-can-improve-customer-service

Wang, L., Lu, L., Han, X., Li, X., Li, J., Li, Y., & Chen, H. (2020). Deep learning in medical imaging: general overview and future promise. *Journal of medical systems*, 44(12), 256.

Wang, X., Peng, Y., Lu, L., & Bagheri, N. (2020). Deep learning in medical imaging: General overview and future promise. *Journal of healthcare engineering,* 2020.

Zanaty, E. A., Ahmed, A. M., Elnahas, M. O., & Hassanien, A. E. (2020). Blood cell classification using deep convolutional neural networks. *Future Computing and Informatics Journal,* 5(1), 25-32.

Zhang, L., Xie, J., Shen, C., Ji, X., Wang, Y., & Ye, Q. (2019). Blood cell image classification using convolutional neural networks. *Journal of Medical Systems*, 43(11), 353.

APPENDIX

SOURCE CODE OF THE IMPLEMENTATION

```
#Blood cell subtype classification
import pandas as pd
import cv2 as cv
import numpy as np
import matplotlib.pyplot as plt
import os
import seaborn as sns
```

```
from keras.models import Sequential
from keras.layers import Conv2D,Dense,Flatten,MaxPooling2D,Dropout,Activation
import tensorflow as tf
import random
from keras.utils import to_categorical
from sklearn.metrics import accuracy_score,confusion_matrix
#TRAIN AND TEST DATASET ADDRESS
DATASET="./dataset2-master/images/TRAIN"
TEST_DATASET="./dataset2-master/images/TEST"

#Categroized images
#4 types of subcells
CATEGORIES=["EOSINOPHIL","LYMPHOCYTE","MONOCYTE","NEUTROPHI
L"]
#reading original image from directory
for category in CATEGORIES:
    label=CATEGORIES.index(category)
    path=os.path.join(DATASET,category)

    for img_file in os.listdir(path):

        # 1 indicates read image in RGB scale
        # 0 indicates read image in grey scale

        img=cv.imread(os.path.join(path,img_file),1)

        #open cv read image in BGR format
        #below we convert it to RGB format
        img=cv.cvtColor(img,cv.COLOR_BGR2RGB)
        #print(img.shape)
        plt.imshow(img)
        plt.show()
```

```
            break

        #plotting single image from each folder
        print(category)
        print(label)
#reading image from directory
for category in CATEGORIES:
        label=CATEGORIES.index(category)
        path=os.path.join(DATASET,category)

        for img_file in os.listdir(path):

            # 1 indicates read image in RGB scale
            # 0 indicates read image in grey scale

            img=cv.imread(os.path.join(path,img_file),1)
            img=cv.cvtColor(img,cv.COLOR_BGR2RGB)
            dst = cv.fastNlMeansDenoisingColored(img,None,5,10,7,21)
            #image convert to smaller pixels 60*60
            #print(img.shape)
            plt.figure(figsize=(10,8))
            plt.subplot(121)
            plt.imshow(dst)
            plt.subplot(122)
            plt.imshow(img)
            plt.show()
            print(category)
            print(label)
            break

        #plotting single image from each folder
#make train data
```

```
train_data=[]

for category in CATEGORIES:

    #each cateogry into unique integer
    label=CATEGORIES.index(category)
    path=os.path.join(DATASET,category)

    for img_file in os.listdir(path):

        img=cv.imread(os.path.join(path,img_file),1)
        img=cv.cvtColor(img,cv.COLOR_BGR2RGB)
        #dst = cv.fastNlMeansDenoisingColored(img,None,5,10,7,21)
        img=cv.resize(img,(60,60))
        train_data.append([img,label])
#make test data
test_data=[]

for category in CATEGORIES:

    #each cateogry into unique integer
    label=CATEGORIES.index(category)
    path=os.path.join(TEST_DATASET,category)

    for img_file in os.listdir(path):

        img=cv.imread(os.path.join(path,img_file),1)
        img=cv.cvtColor(img,cv.COLOR_BGR2RGB)
        #dst = cv.fastNlMeansDenoisingColored(img,None,5,10,7,21)
        img=cv.resize(img,(60,60))
        test_data.append([img,label])
#print total data in train and test
```

```
print(len(train_data))
print(len(test_data))
#shuffle the dataset fo good result

random.shuffle(train_data)
random.shuffle(test_data)
#check the data
for lbl in train_data[:10]:
   print(lbl[1])
#lets seprate the feature and target variable
train_X=[]
train_y=[]

for features,label in train_data:
   train_X.append(features)
   train_y.append(label)

len(train_X),len(train_y)
#lets seprate the feature and target variable
test_X=[]
test_y=[]

for features,label in test_data:
   test_X.append(features)
   test_y.append(label)

len(test_X),len(test_y)
#convert image array to numpy array
#-1 means same size
# 40*40 means height and width
# 3 for R+G+B
train_X=np.array(train_X).reshape(-1,60,60,3)
```

```
train_X=train_X/255.0
train_X.shape

#we divide the np array by 255 to close all values to 0
#convert image array to numpy array
#-1 means same size
# 40*40 means height and width
# 3 for R+G+B

test_X=np.array(test_X).reshape(-1,60,60,3)
test_X=test_X/255.0
test_X.shape

#we divide the np array by 255 to close all values to 0
#count labels

sns.countplot(train_y,palette='Set3')
#we can see each categroy has equal data
#convert label into the one hot encode

#train y
one_hot_train=to_categorical(train_y)
one_hot_train
#test_y
one_hot_test=to_categorical(test_y)
one_hot_test
#build the models
#import Keras libraries
model=Sequential()

model.add(Conv2D(32,(3,3),activation='relu',input_shape=(60,60,3)))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.20))

model.add(Conv2D(64,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.20))

model.add(Conv2D(128,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.40))

model.add(Flatten())

model.add(Dense(64,activation='relu'))
model.add(Dense(128,activation='relu'))
model.add(Dense(64,activation='relu'))

model.add(Dense(4,activation='softmax'))

model.summary()
#we will choose adam optimizer
#we have 4 categories so loss function is categorical_crossentropy
#metrics accuracy
model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy']
)
#lets split the 20% train dataset for validation
hist=model.fit(train_X,one_hot_train,epochs=50,batch_size=128,validation_split=0.2)
#model evaluation
test_loss,test_acc=model.evaluate(test_X,one_hot_test)
test_loss,test_acc
#train and validation loss
plt.plot(hist.history['loss'])
plt.plot(hist.history['val_loss'])
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['train','Validation'],loc='upper left')
plt.show()
#train and validation accuracy
plt.plot(hist.history['accuracy'])
plt.plot(hist.history['val_accuracy'])
plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['train','Validation'],loc='upper left')
plt.show()
#model prediction

y_pred=model.predict(test_X)
y_pred
y_pred_labels = np.argmax(y_pred, axis=1)
y_pred_labels
for i in range(10):
        print("Actual=%s, Predicted=%s" % (test_y[i], y_pred_labels[i]))
#accuracy_score

accuracy_score(test_y,y_pred_labels)
# Compute confusion matrix
cm = confusion_matrix(test_y, y_pred_labels)

# Visualize confusion matrix
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", cbar=False)
plt.title("Confusion Matrix")
plt.xlabel("Predicted Labels")
plt.ylabel("True Labels")
plt.show()
model.save("model.h5")
```

SYSTEM IMPLEMENTATION USING FLASK FRAMEWORK

```
import os
import sys

# Flask
from flask import Flask, redirect, url_for, request, render_template, Response, jsonify,
redirect
from werkzeug.utils import secure_filename
from gevent.pywsgi import WSGIServer

# TensorFlow and tf.keras
import tensorflow as tf
from tensorflow import keras

from keras.applications.imagenet_utils import preprocess_input, decode_predictions
from keras.models import load_model
from keras.preprocessing import image

# Some utilites
import numpy as np
from util import base64_to_pil

from PIL import Image

# Declare a flask app
app = Flask(__name__)
```

```
# You can use pretrained model from Keras
# Check https://keras.io/applications/
# or https://www.tensorflow.org/api_docs/python/tf/keras/applications

# Model saved with Keras model.save()
MODEL_PATH = 'models/model.h5'

# Load your own trained model
model = load_model(MODEL_PATH)
model.make_predict_function()          # Necessary
print('Model loaded. Start serving...')
print('Model loaded. Check http://127.0.0.1:5000/')

@app.route('/', methods=['GET'])
def index():
    # Main page
    return render_template('index.html')


@app.route('/predict', methods=['GET', 'POST'])
def predict():
    if request.method == 'POST':
        image_size = (60, 60)
        img = base64_to_pil(request.json)
        img = img.resize(image_size)
        #img.save("./uploads/image.png")
        img_array = keras.preprocessing.image.img_to_array(img)
        img_array = tf.expand_dims(img_array, 0)  # Create batch axis
        predictions = model.predict(img_array)
        theWBC = str(np.argmax(predictions))
        # print( 'the wbc:', theWBC)
        # overall = np.sum(predictions)
        theValue = str(np.argmax(predictions))
        # print('the value:', theValue)
        # wbc_lookup = dict([
        #       ('0','EOSINOPHIL'),
        #       ('1','LYMPHOCYTE'),
        #       ('2','MONOCYTE'),
        #       ('3','NEUTROPHIL')
        # ])
        if theWBC == '0':
            wbc_lookup = 'EOSINOPHIL'
        elif theWBC == '1':
            wbc_lookup = 'LYMPHOCYTE'
        elif theWBC == '2':
            wbc_lookup = 'MONOCYTE'
        elif theWBC == '3':
            wbc_lookup = 'NEUTROPHIL'
        else:
            wbc_lookup = 'not in category'

        # wbcName = str(wbc_lookup[theWBC])
        return jsonify(result=wbc_lookup, probability=theValue)
    return None



if __name__ == '__main__':
    # app.run(port=5002, threaded=False)
    # Serve the app with gevent
    http_server = WSGIServer(('0.0.0.0', 5000), app)
    http_server.serve_forever()
```

HTML CODE (index.html)

```
{% extends "base.html" %} {% block content %}


<br><br><br>
<div class="main">
  <br><br><br><br><br>
  <div class="row">
    <div class="column">
      <div align="center">
            <img src="static/leukoright.png" alt="logo" width="200">
      </div>
    </div>


    <div class="column">
      <div align="center">
        <div class="panel">
          <br>
          <input id="file-upload" class="hidden" type="file"
accept="image/x-png,image/gif,image/jpeg" />
          <label for="file-upload" id="file-drag" class="upload-box">
          <br><br>
              <div id="upload-caption"><br>Drop image here or click to select </div>
            <img id="image-preview" class="hidden" />
            <div id="image-box">
              <img id="image-display" />
              <div id="pred-result" class="hidden"></div>
              <svg id="loader" class="hidden" viewBox="0 0 32 32" width="32"
height="32">
                <circle id="spinner" cx="16" cy="16" r="14" fill="none"></circle>
              </svg>
            </div>
        </div>
          </label>
        </div>
      </div>


      <div style="margin-bottom: 2rem;">
           <div align="center">
          <input type="button" value="Submit" class="aligncenter button"
onclick="submitImage();" />
          <input type="button" value="Clear" class="aligncenter button"
onclick="clearImage();" />
           </div>
      </div>
    </div>


  </div>
</div>


{% endblock %}
```

HTML CODE (base.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <title>leukoRight</title>
    <link rel="icon" type="image/x-icon" href="static/favicon.png">
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <link rel="stylesheet" type="text/css" href="{{ url_for('static',filename='main.css')
}}" />
  </head>


  <!-- GitHub Corner -->
  <!-- <a href="https://github.com/originates/leukoRight" class="github-corner"
aria-label="View source on GitHub"><svg width="60" height="60" viewBox="0 0
250 250" style="fill:#151513; color:#fff; position: absolute; top: 0; border: 0; right:
```

```
0;" aria-hidden="true"><path d="M0,0 L115,115 L130,115 L142,142 L250,250
L250,0 Z"></path><path d="M128.3,109.0 C113.8,99.7 119.0,89.6 119.0,89.6
C122.0,82.7 120.5,78.6 120.5,78.6 C119.2,72.0 123.4,76.3 123.4,76.3 C127.3,80.9
125.5,87.3 125.5,87.3 C122.9,97.6 130.6,101.9 134.4,103.2" fill="currentColor"
style="transform-origin: 130px 106px;" class="octo-arm"></path><path
d="M115.0,115.0 C114.9,115.1 118.7,116.5 119.8,115.4 L133.7,101.6 C136.9,99.2
139.9,98.4 142.2,98.6 C133.8,88.0 127.5,74.4 143.8,58.0 C148.5,53.4 154.0,51.2
159.7,51.0 C160.3,49.4 163.2,43.6 171.4,40.1 C171.4,40.1 176.1,42.5 178.8,56.2
C183.1,58.6 187.2,61.8 190.9,65.4 C194.5,69.0 197.7,73.2 200.1,77.6 C213.8,80.2
216.3,84.9 216.3,84.9 C212.7,93.1 206.9,96.0 205.4,96.6 C205.1,102.4 203.0,107.8
198.3,112.5 C181.9,128.9 168.3,122.5 157.7,114.1 C157.9,116.9 156.7,120.9
152.7,124.9 L141.0,136.5 C139.8,137.7 141.6,141.9 141.8,141.8 Z"
fill="currentColor"
class="octo-body"></path></svg></a><style>.github-corner:hover
.octo-arm{animation:octocat-wave 560ms ease-in-out}@keyframes
octocat-wave{0%,100%{transform:rotate(0)}20%,60%{transform:rotate(-25deg)}40
%,80%{transform:rotate(10deg)}}@media (max-width:500px){.github-corner:hover
.octo-arm{animation:none}.github-corner .octo-arm{animation:octocat-wave 560ms
ease-in-out}}</style> -->


  <body>
    {% block content %}{% endblock %}
  </body>


  <footer>
    <script src="{{ url_for('static',filename='main.js') }}"></script>
  </footer>
</html>
```

CSS CODE

```
body {
  font-family: -apple-system, BlinkMacSystemFont, Segoe UI, Roboto, Oxygen,
    Ubuntu, Cantarell, Fira Sans, Droid Sans, Helvetica Neue, sans-serif;
  -webkit-font-smoothing: antialiased;
  background: linear-gradient(90deg, #ece4f9 20%, #f3efff 55%)!important;
  -webkit-tap-highlight-color: rgba(126, 126, 204, 0.30)!important;
}

/* Global button style */
.button {
  font-family: inherit;
  text-align: center;
  cursor: pointer;
  border: none;
  text-decoration: none;
  outline: none;
  color: #2f2f2f;
  background-color: #ddc1ee;
  padding: 0.5rem 1.2rem;
  border-radius: 2px;
  font-size: 1rem;
  min-width: 6rem;
}


.button:hover {
  background-color: #7e7ecc!important;
  color: #FFF!important;
}

.button:focus-visible {
  outline: 0;
  box-shadow: none;
  color: #2f2f2f!important;
  background-color: #7e7ecc!important;
  color: #FFF!important;
```

```
}

.button:focus,
.focus-visible:focus:not(:focus-visible) {
  outline: 0;
  color: #2f2f2f!important;
  background-color: #ddc1ee!important;
}
.column {
  float: left;
  width: 16rem;
}

/* Clear floats after the columns */
.row:after {
  content: "";
  display: table;
  clear: both;
}

/* Responsive layout - when the screen is less than 600px wide, make the three
columns stack on top of each other instead of next to each other */
@media screen and (max-width: 600px) {
  .column {
    width: 100%;
  }
}
.button:hover {
  background-color: rgb(16, 110, 190);
}

.button.disabled {
  pointer-events: none;
  background-color: #cccccc;
  color: #666666;
}

/* Main section */

.main {
  box-sizing: border-box;
  display: flex;
  flex-direction: column;
  align-items: center;
}

.main .title h3 {
  font-size: 2.3rem;
  font-weight: 300;
  margin: 0.8rem 0;
}

.hidden {
  display: none;
}

.reveal {
  opacity: 0;
}

.reveal:hover {
  opacity: 0.2;
}
```

```
/* Upload box */
.upload-box {
  font-size: 0.8rem;
  color: #666666;
  cursor: pointer;
  width: 14rem;
  height: 14rem;
  background: #fff;
  border: 0.1rem dashed #838388;
  border-radius: 0.4rem;
  display: flex;
  justify-content: center;
  align-items: center;
  flex-direction: column;
  margin: 1rem 0 2rem 0;
}

.upload-box.dragover {
  /* background-color: grey; */
  color: #eeeeee;
  border: 0.1rem solid rgb(0, 120, 212);
  box-shadow: inset 0 0 0 0.1rem rgb(0, 120, 212);
}

.upload-box:hover {
  border-color: rgb(0, 120, 212);
}

.upload-box #image-preview {
  max-width: 1rem;
  max-height: 1rem;
  box-shadow: 0 4px 4px 0 rgba(0, 0, 0, 0.2), 0 6px 10px 0 rgba(0, 0, 0, 0.19);
  display: none;
}

#image-result {
  box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);
  max-height: 1rem;
  display: none;
}

#image-box {
  position: relative;
  width: auto;
}

#image-display {
  align-items: left;
  box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);
  max-height: 12rem;
  max-width: 12rem;
}

#image-display.loading {
  filter: brightness(30%);
}

#pred-result {
  color: white;
  font-size: 1.5rem;
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
```

```
}

#loader {
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
  z-index: 10;
  margin: 0 auto;
}

/* Animation */
#spinner {
  box-sizing: border-box;
  stroke: #cccccc;
  stroke-width: 3px;
  transform-origin: 50%;
  animation: line 1.6s cubic-bezier(0.4, 0, 0.2, 1) infinite,
    rotate 1.6s linear infinite;
}
@keyframes rotate {
  from {
    transform: rotate(0);
  }
  to {
    transform: rotate(450deg);
  }
}
@keyframes line {
  0% {
    stroke-dasharray: 2, 85.964;
    transform: rotate(0);
  }
  50% {
    stroke-dasharray: 65.973, 21.9911;
    stroke-dashoffset: 0;
  }
  100% {
    stroke-dasharray: 2, 85.964;
    stroke-dashoffset: -65.973;
    transform: rotate(90deg);
  }
}
```