

BLOOD CELL COUNT USING OPEN-CV PYTHON

Submitted in partial fulfilment of the requirements for the award of a
Bachelor of Technology degree in Biomedical Engineering.

By,

DILIP E.K (39240015)

SABARI.K (39240033)



**DEPARTMENT OF BIOMEDICAL ENGINEERING SCHOOL OF
BIO AND CHEMICAL ENGINEERING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI-600119

APRIL – 2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF BIOMEDICAL ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of SABARI K (39240033) and DILIP EK (Reg no:39240015) who carried out the project entitled "BLOOD CELL COUNT USING OPEN-CV PYTHON" under my supervision from October 2022 to April 2023.

Internal Guide

(Dr. J. PREMKUMAR)

Head of the department

(Dr. T. SUDHAKAR)

HEAD OF THE DEPARTMENT

DEPARTMENT OF BIOMEDICAL ENGINEERING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Chennai - 600 076

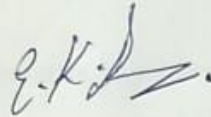
Submitted for Viva voce Examination held on 18.04.2023

Internal examiner

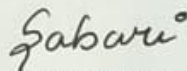
External examiner

DECLARATION

We, **SABARI K** and **DILIP EK** here by declare that the Project Report entitled "**BLOOD CELL COUNT USING OPEN-CV PYTHON**" done by us under the guidance of **Dr. J. PREMKUMAR** of Biomedical Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfilment of the requirements for the award of Bachelor of Technology degree in Biomedical engineering.



1. **DILIP EK**



2. **SABARI K**

DATE: 18-04-2023

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATES

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to the **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

We convey our thanks to **Dr. T. Sudhakar, M.Sc, Ph.D, Heads of the Department, Biomedical Engineering** for providing us necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our interdisciplinary Project Guide **Dr. J.Premkumar, M.Sc., Ph.D.** Department of Biomedical Engineering for his valuable guidance, suggestions, and constant encouragement paving the way for the successful completion of our project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the Department of Biomedical Engineering who were helpful in many ways for the completion of the Project.

ABSTRACT

Blood cells both white and red are important part of the immune system. These cells help fight infections by attacking bacteria, viruses, and germs that invade the body. White blood cells originate in the bone marrow but circulate throughout the bloodstream, while red blood cell helps transport oxygen to our body. Accurate counting of those may require laboratory testing procedure that is not usual to everyone. Generating codes that will help counting of blood cells that produce accurate response via images gives a relief on this problem. In this study, the images were processed and a blob detection algorithm was used to detect and differentiate RBCs from WBCs. A cell counting method was also used to provide an actual count of the RBCs and WBCs detected. The automation comes with a graphical user interface backed-up with a working database system to keep the records of the users (e.g. patients, respondents). The performance of the system was statistically described as accurate compared to the manual method of counting. Results show an accuracy of platelet, and RBCs and WBCs show the best accuracy of the system. Hence, the proposed system can benchmark with the manual methods of detection and counting of platelets, RBCs and WBCs in blood samples. Detection of White Blood Cell and Red Blood Cell is very useful for various medical applications, like counting of WBC, disease diagnosis, etc. Circle detection is the most suitable approach. This article is the implementation of suitable image segmentation and feature extraction techniques for blood cell identification, on the obtained enhanced images.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO
	ABSTARCT	5
	LIST OF ABBREVIATIONS	8
1	INTRODUCTION	9
2	LITRATURE SURVEY	10
3	AIM AND SCOPE	16
3.1	AIM	16
3.2	EXISTING SYTSTEM	16
3.3	DISADVANTAGES	17
3.4	PROPOSED SYSTEM	17
3.5	ADVANTAGES	17
4	MATERIALS AND METHODS	18
4.1	SYSTEM REQUIREMENTS	18
4.1.1	HARDWARE	18
4.1.2	SOFTWARE	19
4.2	MACHINE LEARNING	19
4.2.1	FEATURES	20
4.2.2	CLASSIFICATION	20
4.3	TKINTER	21
4.4	OPENCV-PYTHON	22
4.5	PYTHON	23
4.5.1	ANACONDA	24
4.5.2	ANACONDA NAVIGATOR	25
4.5.3	JUPYTER NOTEBOOK	28
4.6	INSTALLATION OF PYTHON	30
4.7	SETUP OF PYTHON	33
4.8	INSTALLATION OF ANACONDA	36
5	RESULTS AND DISCUSSION	42
5.1	SYSTEM MODULE	42

5.2	COLLECTION OF DATASETS	43
5.3	CODING FOR SEGMENTATION	45
6	CONCLUSION AND FUTURE WORK	48
6.1	CONCLUSION	48
	REFERENCE	49
	APPENDIX	50

LIST OF FIGURES

FIGURE	DESCRIPTION	PAGE NO
4.6.1	Installing of python	27
4.6.3	Installing setup progress of python	28
4.7.2	Pycharm community edition setup	29
4.7.3	Pycharm community install location	30
4.7.4	Pycharm community Edition installation	31
4.7.5	Start menu folder for pycharm	32
4.7.6	Pycharm installed	33
4.8.1	Anaconda Installer	34
4.8.2	Choose anaconda install location	35
4.8.3	License agreement setup	36
4.8.4	Anaconda installation type	37
4.8.5	Anaconda install location	38
4.8.6	Recommended and alternate approach	39
4.8.7	Anaconda installation completed	40
4.9.1	Command prompt	41
4.9.3	Anaconda prompt	42
5.2.1	Collected dataset	45
5.3.1	Coding for segmentation	46
5.3.3	Final output of detected blood cells	47

LIST OF TABLES

TABLE	DESCRIPTION	PAGE NO
4.1	SYSTEM REQUIREMENTS	18
4.1.1	HARDWARE REQUIREMENTS	18
4.1.2	SOFTWARE REQUIREMENTS	18

LIST OF ABBREVIATIONS

ACC	-	ACCURACY
AC	-	ACTIVE CONTOUR
AI	-	ARTIFICIAL INTELLIGENCE
BLOB	-	BINARY LARGE OBJECT
HTML	-	HYPER TEXT MARKUP LANGUAGE
URL	-	UNIFORM RESOURCE LOCATOR
VGG	-	VISUAL GEOMETRY GROUP
XML	-	EXTENSIBLE MARKUP LANGUAGE
XUL	-	XML USER INTERFACE LANGUAGE

CHAPTER 1

INTRODUCTION

Platelets are one of the blood cells that stops the bleeding in the body from blood clotting. Platelets can detect if any blood vessels are damaged. Red blood cells are also tiny blood cells that is also important in the health of human through carrying fresh oxygen throughout the body whereas white blood cells helps protect the body from infections. Complete Blood Count (CBC) involves blood testing to determine the healthiness of the major components of blood which are platelets, red blood cells and white blood cells. Abnormalities of result based from references of normal count of cells may indicate an underlying medical condition that needs further evaluation. For this past few years, CBC counting is one of the most studied areas of research due to accuracy problem. Laboratories in the hospital in the Philippines are still using the traditional method of counting blood cells. This was done in either manual method through haemocytometer or by automated method through flow-cytometer. In this study, it uses images of the blood to calculate the number of cells since research on medical images is new technology. Image processing is a method which involves signal processing and mathematical procedure to change the image into another form of desired image. Image analysis is the extraction of significant information from an image. Hence, this paper does not involve image processing only but analysis as well. Nowadays, there are many ways of image processing and analysis of blood cell images. However, the quest for the highest accuracy is still one of the aims of the researchers. With so many studies, the researchers will present another way of counting blood cells through the use of strong level of algorithm with the help of python OpenCV programming language. This study used colour filtering to keep a specific hue while desaturating the rest of the image. It also involves image segmentation to convert the image into multiple parts to identify which of the cells are platelets, red blood cells or white blood cells. Blob detection plays important role in this study which primarily detects the differences of each blood cells before the cells will be counted.

CHAPTER 2

LITERATURE SURVEY

[1] Title: Platelet count using image processing

Authors: Prabhjot kaur , Vishakha sharma

Description: Platelet count is a blood test which plays a very important role to evaluate the health as well as to diagnose and follow the process of treatment for a wide range of diseases including leukemia, anemia etc. Typically in primary health care centre's, platelet counting is performed manually which gives inaccurate and unreliable results depending upon the skill of the technician. Another method used for counting platelet is Advia hematology analyzer which is a very expensive machine, not affordable by many rural/remote areas. The intent of our research is to address these challenges. This paper introduces a cost effective automatic platelet counter using a microscope. It attached with an automated camera which is connected to a computer and shoot blood sample images. We considered a concept for counting of platelets in microscopic blood cell images using Circular Hough Transform which uses features such as size and shape of platelets for counting mechanism. Normal blood cells images are used for the results and the examined data produced the accuracy rate of 96% on comparing with manual and machine counting.

[2] Title: Computer Aided System for Human Blood Cell Identification, Classification and Counting

Authors: M Annalatha ,N Christy Evangeline

Description: Evaluation of Blood Smear is one of the most basic laboratory procedures in Hematology as deviations of blood cell count from their reference values can be indicators of various disease conditions. Digital Image Processing can help the technologists in the analysis and classification of blood cells and diagnosis

of certain health conditions based on the CBC values and their morphology. This project aims at generating software that can identify, classify and count various blood cells from the peripheral blood smear samples. Locating, identifying and counting of Blood cells manually can be tedious, time consuming and inaccurate. On the other hand, Automatic Hematology Analyzers that work on Flow cytometry principle for classification are expensive and can perform only quantitative analysis. Thus taking into consideration, the wide scope of Image processing in the field of medicine, this proposed work aims to generate a preliminary framework of Automatic Analysis of Human Blood smear for identification, classification and counting of blood cells by using various Image processing.

[3] Title: Determination of blood components (WBCs, RBCs, and Platelets) count in microscopic images using image processing and analysis

Authors: C Jennifer , Dela Cruz ,Valiente

Description: Blood is one of the most essential parts of the human body, and it comprises of the RBCs, WBCs, and Platelets. Complete blood count characterizes the condition of well-being. Hence, segmentation and identification of blood cells is very important. Up to this day, many hospitals and health centers still use the old conventional method which involves manual counting of blood cells. This technique is time-consuming and prone to erroneous outcomes. On the other hand, there are some costly machines like Hematology Analyzer, which health centers cannot afford. This paper presents a raspberry-pi based image analysis system that is designed to segment and count blood cells from microscopic images of blood using Hue, Saturation, and Value (HSV) thresholding method and connected component labeling, respectively. Detection and counting of RBCs, WBCs, and Platelets have been done on ten microscopic images. Statistical analysis was performed to compare the values measured by the proposed system to the actual complete blood count test result of each patient. It shows that the proposed system has a 90% and up accuracy with respect to the actual CBC tests result. In addition, an android application was also developed to aid the user, especially those who are in rural areas, in assessing the number of blood cells, and to send the results from GUI to a doctor or specialist through short message service (SMS) for remote diagnosis.

[4] Title: Implementation of Blood Cell Counting Algorithm using Digital Image Processing Techniques

Authors: Vilas B Inchur ,L S Praveen , Preetham Shankpal

Description: Blood is a connective tissue with cellular components like RBC, WBC, and platelets. If the blood cells are below the standard range it leads to various health diseases and hence proper blood cell counting technique plays importance in the pathology department. The most widely used method for counting blood cells is the microscopic technique which gives better results however, it takes more time to count the number of blood cells. Blood cell count using a digital image processing technique can be considered to automate the microscopic method and minimize the time taken by the conventional method. In this paper, a digital image processing technique is proposed to accurately count all the blood cells in a given slide with maximum accuracy. A method such as morphological operator, texture-based classification method and Circular Hough Transformation (CHT) used to count the RBC which gives accuracy around 90%. Similarly, Morphological Operator, Vegetation method, and Texture Object-Based classification are implemented to count the WBC and Platelet. However, Texture Object-Based classification method gave 100 % accuracy compared to Morphological Operator and Vegetation method. From the results discussed in this paper conclude that for RBC count using Circular Hough Transformation gives better result and WBC, platelets count using the Texture Object-Based classification method gives better results.

[5] Title: An automated system for segmenting platelets from microscopic images of blood cells

Authors: Ratnadeep Dey, Kaushiki Roy, Pramit Ghosh

Description: Platelet count is a very important diagnosis test to identify diseases like Dengue, Malaria, Yellow fever, and others. For dengue patient monitoring, platelet count is often needed. A suspected dengue patient needs a very quick diagnosis to give an accurate result of how critical is the condition of the patient. In most laboratories, Leishman's stained blood slides are used to count platelets. However, these manual platelet counting requires expert lab technician and the

overhead increases manifold when huge blood samples are to be checked by lab technicians that make the entire process time-consuming. So, in this paper, an automated system is proposed which can extract platelets from the microscopic image of blood cells, and that makes platelet counting task easy. Microscopic images of stained blood slides are captured using a light microscope. Then using color-based segmentation and morphological operation, platelets can be extracted. A comparative study between the platelet counts obtained before and after segmentation along with calculation of the efficiency of the system has shown this method to be robust and effective for automation of platelet count system

[6] Title: White blood cells nucleus segmentation from microscopic images of strained peripheral blood film during leukemia and normal condition

Authors: R. Ahasan, A.U. Ratul, A.S.M. Bakibillah,

Description: Counting of White blood cells (WBC) & characterizing their nucleus can provide valuable information to the doctors in order to identify different diseases or stage of a particular disease. The manual method is a tiresome process and has a lot of inaccuracy. On the other hand, the machine (hematological analyzer) based method are very expensive. Digital image processing can be a less time consuming and cost effective method for counting and characterizing the WBC. In order to Count or characterize WBC by image processing, proper segmentation is the key challenge. In this paper we proposed an algorithm to segment WBC nucleus from microscopic images of stained peripheral blood film during leukemia and normal condition. The proposed algorithm involves different steps such as color space conversion, color thresholding, filtering, marker controlled watershed and different morphological operations. The accuracy of the result obtained is 88.57%

[7] Title: "Detection of abnormal blood cells on the basis of nucleus shape and counting of WBC"

Authors: K.K. Jha, B.K. Das, H.S. Dutta,

Description: The counting of Red Blood Cells (RBC), White Blood Cells (WBC) and platelet of human blood in microscopic image can provide useful information concerning the health of the patient. The conventional methods used in laboratories

to detect and diagnose the problem were very slow and prone to get inaccuracy in result. Digital image processing can detect and recognize different diseases and make it easy to analyze and diagnose the actual problem in a fast and cost effective manner. The major intension of this paper is to count number of WBC and also identify the abnormal ones. The proposed work is useful to detect different kind of diseases like Chronic Obstructive Pulmonary Disease, Immune system disorders, Neutropenia, HIV/AIDS, Lymphocytopenia, leukemia etc on the basis of counts of different blood constituents. This paper is going to provide way to recognize types of different kind of WBC in its normal and abnormal form. This will also helpful for hematologists for clear identification and counting of blood cells. Finally the proposed work designed to obtain effective and more accurate result. The result indicates 85% accuracy for identification and counting of different type of blood cells.

[8] Title: Counting and Classification of White Blood Cell using Artificial Neural Network (ANN)

Authors:S. Manik, Lalit, M. Saini, N. Vadera

Description: Quantitative microscopy has improved conventional diagnostic method through better comprehension of microscopic features from a clinical point of view. The extraction of the nucleus from the blood smear images of white blood cells (WBC) gives the profitable data to specialists for classification of various types of disease as the vast majority of the illnesses present in the body can be distinguished by investigating blood. It is extremely time-consuming and tedious to segment the nucleus manually and also classification which is done on the premise of the instruments which are utilized by specialists for segmentation and classification of white blood cells are not economical for every doctor or hospital; so proposed method is ideal which decreases the execution time of segmentation and classification. In this paper; a new framework is proposed to enhance detection and classification of Leukocytes i.e. Nucleus Enhancement by finding Intensity maxima and then classified on the basis of various features extracted from segmented images. Classification is done by Artificial Neural Network (ANN).

[9] Title: Classification of White Blood Cells Based on Morphological Features

Authors: Gautam, Anjali, Bhadauria, Harvindra

Description: The extraction of nucleus from the blood smear images of white blood cells (WBC) provides the valuable information to doctors for identification of different kinds of diseases as most of the diseases present in body can be identified by analyzing blood. Manually it very soporific and tiresome to segment the nucleus and after that classification is done on the basis of that besides that the instruments which are used by experts for segmentation and classification of white blood cells are not affordable by every hospitals and clinics, so automatic system is preferable which reduces the times of segmentation and classification. In our research we focus on segmentation of nucleus from blood smear images using Otsu's thresholding technique applied after contrast stretching and histogram equalization of image followed by minimum filter for reducing noise and increasing brightness of nucleus, mathematical morphological is done to remove the components which are not WBCs, then shape based features are extracted on the basis of that classification rule is applied to classify them in their five category. The classification of nucleus is necessary as they are used to identify different kind of diseases which are related to each type of white blood cells and also help in differential blood count of cells.

[10] Title: "Microscopic images dataset for automation of RBCs counting",

Authors: S. Abbas

Description: A method for Red Blood Corpuscles (RBCs) counting has been developed using RBCs light microscopic images and Matlab algorithm. The Dataset consists of Red Blood Corpuscles (RBCs) images and there RBCs segmented images. A detailed description using flow chart is given in order to show how to produce RBCs mask. The RBCs mask was used to count the number of RBCs in the blood smear image

CHAPTER 3

AIM AND SCOPE

3.1 AIM:

The aim of this project is to Analyzing the Blood Cells Counting using Python OpenCV and predict output for White blood cell ,Red blood cell and Platelets The primary objective of this to present a more accurate counting of blood cells using the python OpenCV programming language. It covers image processing and analysis of platelets, red blood cells and white blood cells.

3.2 PROBLEM STATEMENT:

Detection and counting of white blood cells in blood samples were also presented through computer-aided and mobile-cloud-assisted blood analysis The primary objective of this to present a more accurate counting of blood cells using the python OpenCV programming language. It covers image processing and analysis of platelets, red blood cells and white blood cells. This study was able presented a more accurate counting of the specified types of cell. Further, it has presented a new algorithm in counting these important types of cell found in the human body.

3.3 EXISTING SYSTEM:

Method in counting and classification of white blood cells was also introduced using Artificial Neural Network (ANN). This method decreases the time of executing the segmentation and classification of WBC. ANN was used to classify and confirm the white blood cells. The steps involved image acquisition, image segmentation,

feature extraction and classification using ANN and counting. It involves separation of color components in the image, then histogram was performed. Thresholding was done after the histogram. If object area is greater than the desired value, area from the image corresponding to the triple line square is then extracted. Finally, number of RBCs will be counted from the extracted area of the image. Further, blood smears are stained with Leishman stain. Images are acquired using light microscope connected to digital camera.

DISADVANTAGES:

- Less number of Data set collection
- Feature handled is little complex
- Consuming huge time

3.4 PROPOSED SYSTEM

The proposed method operates on binary images taken from initial segmentation and consists of several detailed steps. Canny edge detection, the most popular edge detection, was done in the image. Morphological operations, the connectedness of the pixels in the image, were then performed. After that, the feature was extracted. After extraction, classification was done. Then clumped cells were separated. OpenCV programming language. It covers image processing and analysis of platelets, red blood cells and white blood cells. The Proposed method using the Tkinter Frame Work on opencv of Blood Cells Counting

ADVANTAGES:

- To improve this accuracy

CHAPTER 4

MATERIALS AND METHODOLOGY

4.1 SYSTEM REQUIREMENTS

Operating System	Windows 7,8,10 (64 bit)
Software	Anaconda Package
Tools	Jupyter notebook
Programming Language	Python

4.1.1 SOFTWARE REQUIREMENTS:

Operating System	Windows 7 or later
Simulation Tool	Open-CV
Documentation	Ms – Office

4.1.2 HARDWARE REQUIREMENTS:

CPU type	Intel Pentium
Ram size	4GB

Hard disk capacity	80 GB
Keyboard type	Internet keyboard
Monitor type	15 Inch colour monitor
CD -drive type	52xmax

4.1.3 ARTIFICIAL INTELLIGENCE:

Artificial intelligence (AI) is the ability of a computer program or a machine to think and learn. It is also a field of study which tries to make computers "smart". As machines become increasingly capable, mental facilities once thought to require intelligence are removed from the definition. AI is an area of computer sciences that emphasizes the creation of intelligent machines that work and reacts like humans. Some of the activities computers with artificial intelligence are designed for include: Face recognition, Learning, Planning, Decision making etc.,

Artificial intelligence is the use of computer science programming to imitate human thought and action by analysing data and surroundings, solving or anticipating problems and learning or self-teaching to adapt to a variety of tasks.

4.2 MACHINE LEARNING

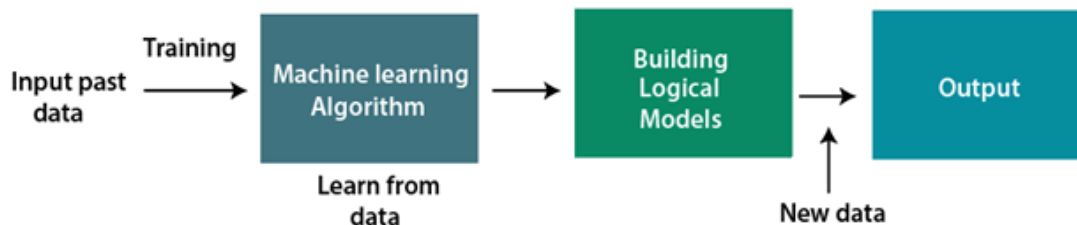
Machine learning is a growing technology which enables computers to learn automatically from past data. Machine learning uses various algorithms for building mathematical models and making predictions using historical data or information. Currently, it is being used for various tasks such as image recognition, speech recognition, email filtering, Facebook auto-tagging, recommender system, and many more.

Machine Learning is said as a subset of artificial intelligence that is mainly concerned with the development of algorithms which allow a computer to learn from the data and past experiences on their own. The term machine learning was first introduced by Arthur Samuel in 1959. We can define it in a summarized way as: "Machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly

programmed”.

A Machine Learning **system** learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it. The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately.

Suppose we have a complex problem, where we need to perform some predictions, so instead of writing a code for it, we just need to feed the data to generic algorithms, and with the help of these algorithms, machine builds the logic as per the data and predict the output. Machine learning has changed our way of thinking about the problem. The below block diagram explains the working of Machine Learning algorithm:



4.2.1 Features of Machine Learning:

- Machine learning uses data to detect various patterns in a given dataset.
- It can learn from past data and improve automatically.
- It is a data-driven technology.
- Machine learning is much similar to data mining as it also deals with the huge amount of the data.

4.2.2 Classification of Machine Learning

At a broad level, machine learning can be classified into three types:

1. Supervised learning
2. Unsupervised learning

3. Reinforcement learning

1) Supervised Learning

Supervised learning is a type of machine learning method in which we provide sample labeled data to the machine learning system in order to train it, and on that basis, it predicts the output.

The system creates a model using labeled data to understand the datasets and learn about each data, once the training and processing are done then we test the model by providing a sample data to check whether it is predicting the exact output or not.

The goal of supervised learning is to map input data with the output data. The supervised learning is based on supervision, and it is the same as when a student learns things in the supervision of the teacher. The example of supervised learning is **spam filtering**.

Supervised learning can be grouped further in two categories of algorithms:

- **Classification**
- **Regression**

2) Unsupervised Learning

Unsupervised learning is a learning method in which a machine learns without any supervision. The training is provided to the machine with the set of data that has not been labeled, classified, or categorized, and the algorithm needs to act on that data without any supervision. The goal of unsupervised learning is to restructure the input data into new features or a group of objects with similar patterns.

In unsupervised learning, we don't have a predetermined result. The machine tries to find useful insights from the huge amount of data.

It can be further classified into two categories of algorithms:

- **Clustering**
- **Association**

4.3 TKINTER

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task. This Python framework provides an interface to the Tk toolkit and works as a thin object-oriented layer on top of Tk. The Tk toolkit is a cross-platform collection of 'graphical control elements', aka widgets, for building application interfaces. If you want to take advantage of the latest version of Tkinter, you'll need to install a version of Python that supports Tcl/Tk 8.5 or greater. This will provide you with the Ttk (Tile extension integrated into Tk), which is required in order to run the current Tk widget set.

4.4. OPENCV-PYTHON

OpenCV-Python is a library of Python bindings designed to solve computer vision problems. Python is a general purpose programming language started by **Guido van Rossum** that became very popular very quickly, mainly because of its simplicity and code readability. It enables the programmer to express ideas in fewer lines of code without reducing readability.

Compared to languages like C/C++, Python is slower. That said, Python can be easily extended with C/C++, which allows us to write computationally intensive code in C/C++ and create Python wrappers that can be used as Python modules. This gives us two advantages: first, the code is as fast as the original C/C++ code (since it is the actual C++ code working in background) and second, it is easier to code in Python than C/C++. Open CV Python is a Python wrapper for the original OpenCV C++ implementation.

OpenCV-Python makes use of **Numpy**, which is a highly optimized library for numerical operations with a MATLAB-style syntax. All the Open CV array structures are converted to and from Numpy arrays. This also makes it easier to integrate with other libraries that use Numpy such as SciPy and Matplotlib.

```
pip install opencv-python
```

```
pip install numpy
```

```
pip install matplotlib
```

To read the images `cv2.imread()` method is used. This method loads an image from the specified file. If the image cannot be read (because of missing file, improper permissions, unsupported or invalid format) then this method returns an empty matrix.

Syntax: `cv2.imread (path, flag)`

Parameters:

path: A string representing the path of the image to be read.

flag: It specifies the way in which image should be read. Its default value is `cv2.IMREAD_COLOR`

Return Value: This method returns an image that is loaded from the specified file.

Note: The image should be in the working directory or a full path of image should be given.

All three types of flags are described below:

cv2.IMREAD_COLOR: It specifies to load a color image. Any transparency of image will be neglected. It is the default flag. Alternatively, we can pass integer value `1` for this flag.

cv2.IMREAD_GRAYSCALE: It specifies to load an image in grayscale mode. Alternatively, we can pass integer value `0` for this flag.

cv2.IMREAD_UNCHANGED: It specifies to load an image as such including alpha channel. Alternatively, we can pass integer value `-1` for this flag.

4.5 PYTHON:

Python is a dynamic, high level, free open source and interpreted programming language. It supports object-oriented programming as well as procedural oriented

programming. In Python, we don't need to declare the type of variable because it is a dynamically typed language.

For example, `x=10` .Here, x can be anything such as String, int, etc.

Python is an interpreted, object-oriented programming language similar to PERL, that has gained popularity because of its clear syntax and readability. Python is said to be relatively easy to learn and portable, meaning its statements can be interpreted in a number of operating systems, including UNIX-based systems, Mac OS, MS-DOS, OS/2, and various versions of Microsoft Windows 98. Python was created by Guido van Rossum, a former resident of the Netherlands, whose favourite comedy group at the time was Monty Python's Flying Circus. The source code is freely available and open for modification and reuse. Python has a significant number of users.

Features in Python

There are many features in Python, some of which are discussed below

- Easy to code
- Free and Open Source
- Object-Oriented Language
- GUI Programming Support
- High-Level Language
- Extensible feature
- Python is Portable language
- Python is Integrated language
- Interpreted Language

4.5.1. ANACONDA:

Anaconda distribution comes with over 250 packages automatically installed, and over 7,500 additional open-source packages can be installed from PyPI as well as the conda package and virtual environment manager. It also includes a GUI, Anaconda Navigator,^[12] as a graphical alternative to the command line interface (CLI).

The big difference between conda and the pip package manager is in how

package dependencies are managed, which is a significant challenge for Python data science and the reason conda exists.

When pip installs a package, it automatically installs any dependent Python packages without checking if these conflict with previously installed packages. It will install a package and any of its dependencies regardless of the state of the existing installation. Because of this, a user with a working installation of, for example, Google Tensorflow, can find that it stops working having used pip to install a different package that requires a different version of the dependent numpy library than the one used by Tensorflow. In some cases, the package may appear to work but produce different results in detail.

In contrast, conda analyses the current environment including everything currently installed, and, together with any version limitations specified (e.g. the user may wish to have Tensorflow version 2.0 or higher), works out how to install a compatible set of dependencies, and shows a warning if this cannot be done.

Opensource packages can be individually installed from the Anaconda repository, Anaconda Cloud (anaconda.org), or the user's own private repository or mirror, using the conda install command. Anaconda, Inc. compiles and builds the packages available in the Anaconda repository itself, and provides binaries for Windows 32/64 bit, Linux 64 bit and MacOS 64-bit. Anything available on [PyPI](https://pypi.org) may be installed into a conda environment using pip, and conda will keep track of what it has installed itself and what pip has installed.

Custom packages can be made using the conda build command, and can be shared with others by uploading them to Anaconda Cloud, PyPI or other repositories.

The default installation of Anaconda2 includes Python 2.7 and Anaconda3 includes Python 3.7. However, it is possible to create new environments that include any version of Python packaged with conda.

4.5.2 Anaconda Navigator:

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda

Repository, install them in an environment, run the packages and update them. It is available for Windows, macOS and Linux.

The following applications are available by default in Navigator:

- JupyterLab
- Jupyter Notebook
- QtConsole
- Spyder
- Glue
- Orange
- RStudio
- Visual Studio Code

4.5.3 JUPYTER NOTEBOOK:

Jupyter Notebook (formerly IPython Notebooks) is a web-based interactive computational environment for creating Jupyter notebook documents. The "notebook" term can colloquially make reference to many different entities, mainly the Jupyter web application, Jupyter Python web server, or Jupyter document format depending on context. A Jupyter Notebook document is a JSON document, following a versioned schema, containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media, usually ending with the ".ipynb" extension.

Jupyter Notebook can connect to many kernels to allow programming in different languages. By default, Jupyter Notebook ships with the IPython kernel. As of the 2.3 release^{[11][12]} (October 2014), there are currently 49 Jupyter-compatible kernels for many programming languages, including Python, R, Julia and Haskell.

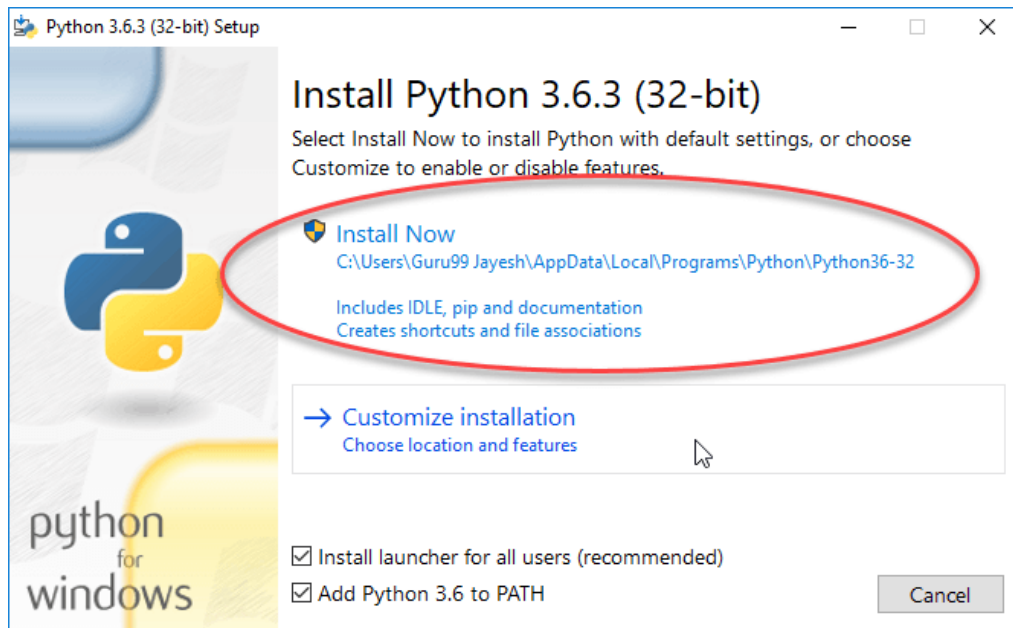
The Notebook interface was added to IPython in the 0.12 release^[14] (December 2011), renamed to Jupyter notebook in 2015 (IPython 4.0 – Jupyter 1.0). Jupyter Notebook is similar to the notebook interface of other programs such as Maple, Mathematica, and SageMath, a computational interface style that originated with Mathematica in the 1980s. According to *The Atlantic*, Jupyter interest overtook the popularity of the Mathematica notebook interface in early 2018.

4.6 Below is a step by step process on how to download and install Python on Windows:

4.6.1 Step 1) To download and install Python, visit the official website of Python <https://www.python.org/downloads/> and choose your version. We have chosen Python version 3.6.3

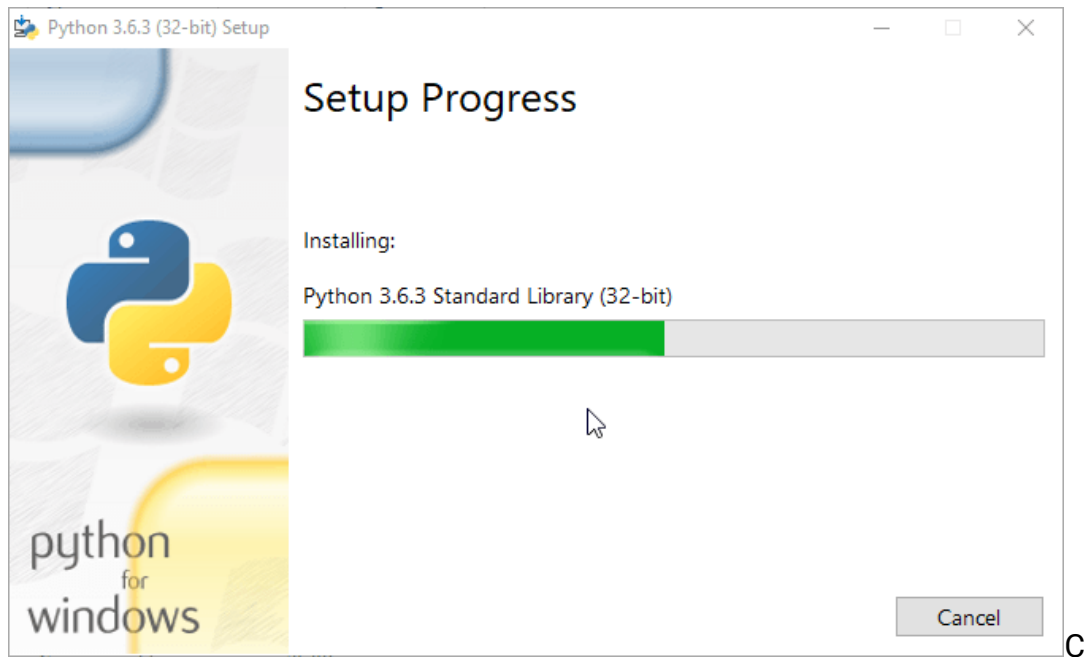


Step 2) Once the download is completed, run the .exe file to install Python. Now click on Install Now.



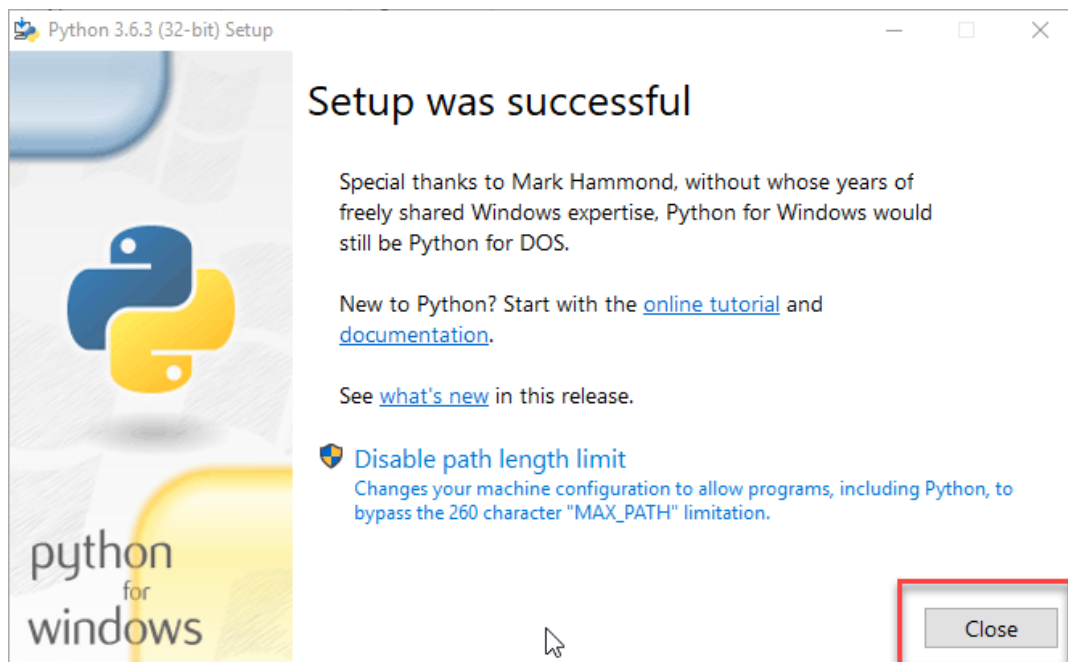
4.6.2 Installing of Python

Step 3) You can see Python installing at this point.



4.6.3 Installing Setup Progress of Python

Step 4) When it finishes, you can see a screen that says the Setup was successful. Now click on "Close".

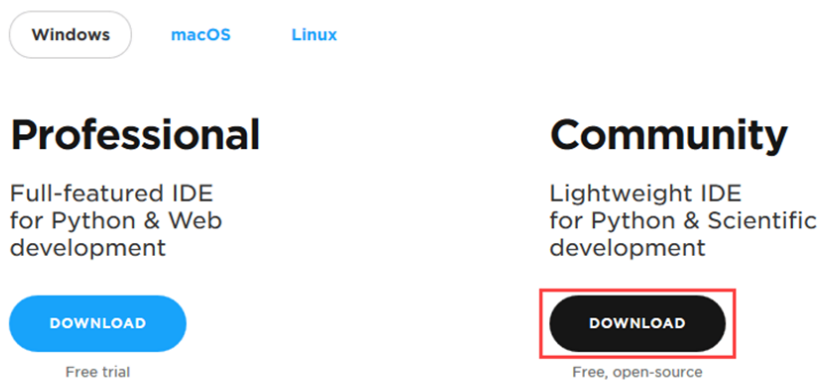


4.7. Setup Progress of Python Completed

4.7.1 Here is a step by step process on how to download and install Pycharm IDE on Windows:

Step 1) To download Pycharm visit the website <https://www.jetbrains.com/pycharm/download/> and Click the "DOWNLOAD" link under the Community Section.

Download PyCharm



The screenshot shows the PyCharm download page. At the top, there are three tabs: "Windows" (selected), "macOS", and "Linux". Below the tabs, there are two main sections: "Professional" and "Community". The "Professional" section describes it as a "Full-featured IDE for Python & Web development" and has a blue "DOWNLOAD" button with "Free trial" text below it. The "Community" section describes it as a "Lightweight IDE for Python & Scientific development" and has a black "DOWNLOAD" button with "Free, open-source" text below it. The "DOWNLOAD" button for the Community edition is highlighted with a red rectangle.

Windows macOS Linux

Professional
Full-featured IDE
for Python & Web
development
DOWNLOAD
Free trial

Community
Lightweight IDE
for Python & Scientific
development
DOWNLOAD
Free, open-source

Download Pycharm Community

Step 2) Once the download is complete, run the exe for install PyCharm. The setup wizard should have started. Click "Next".

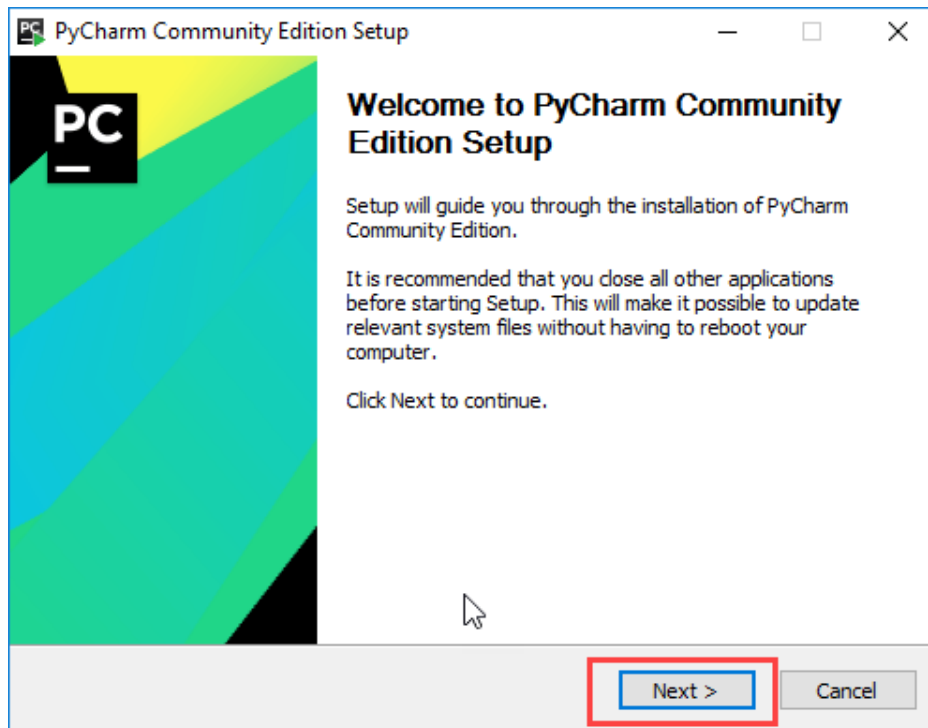


Fig 4.7.2 Pycharm Community Edition Setup

Step 3) On the next screen, Change the installation path if required. Click "Next".

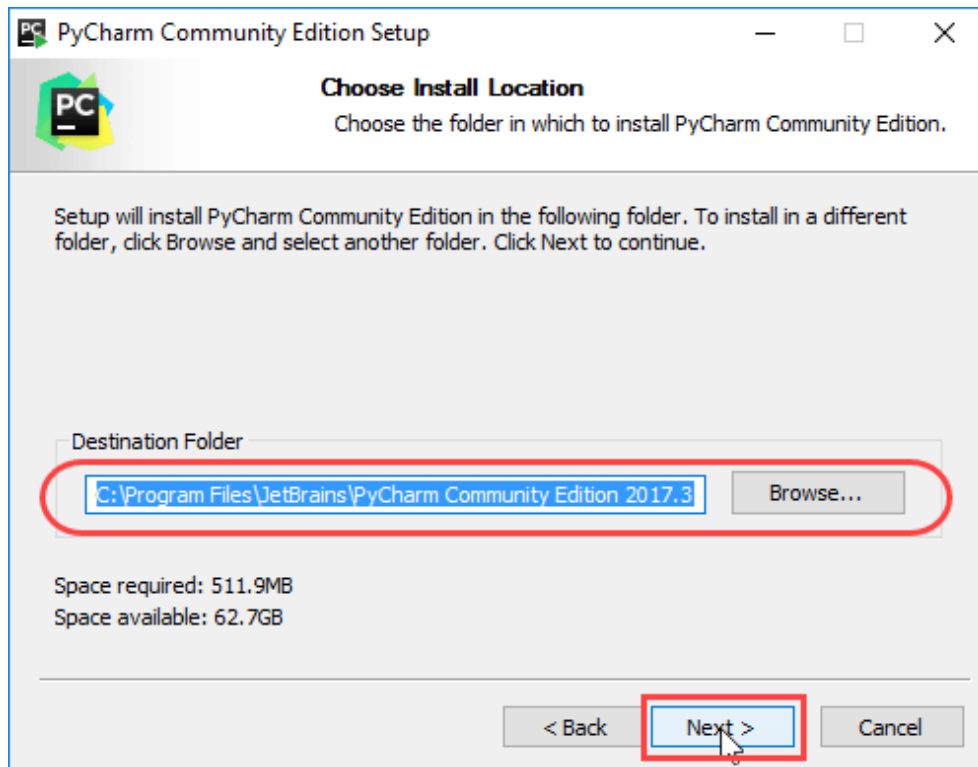


Fig 4.7.3 Pycharm Community Install Location

Step 4) On the next screen, you can create a desktop shortcut if you want and click on "Next".

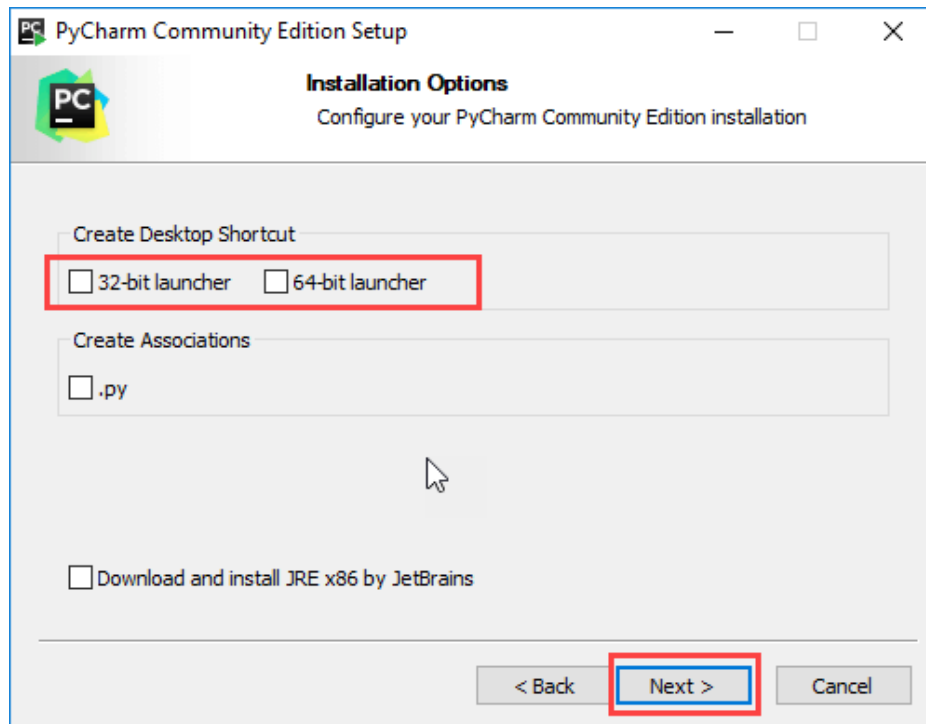


Fig 4.7.4 Pycharm Community Edition Installation

Step 5) Choose the start menu folder. Keep selected JetBrains and click on “Install”.

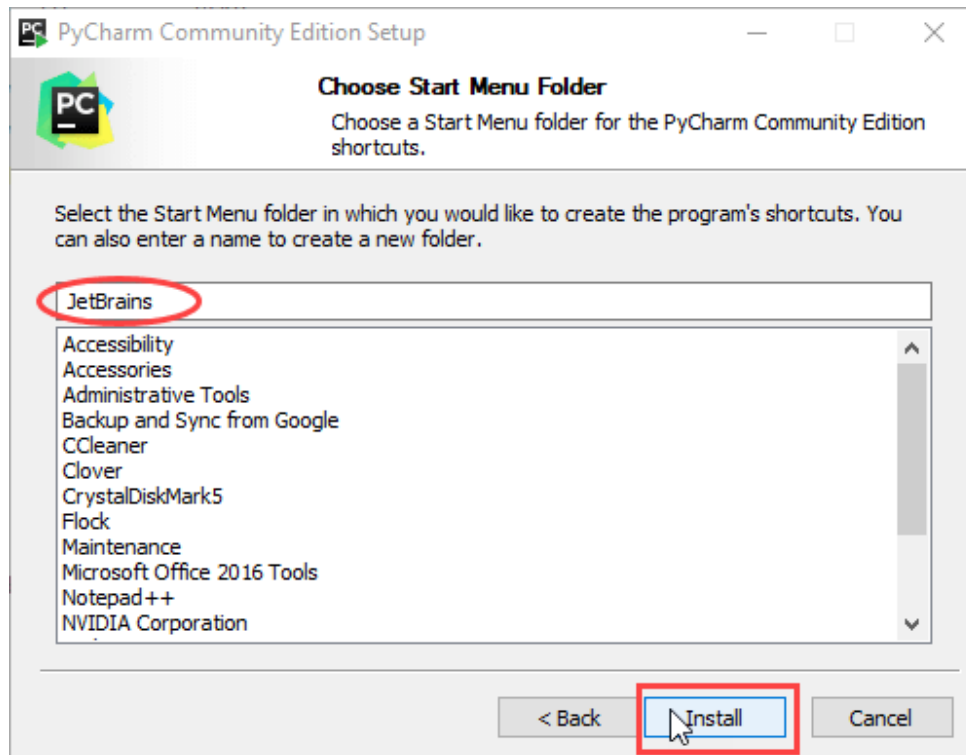


Fig 4.7.5 Start Menu Folder for Pycharm Community
Step 6) Wait for the installation to finish.

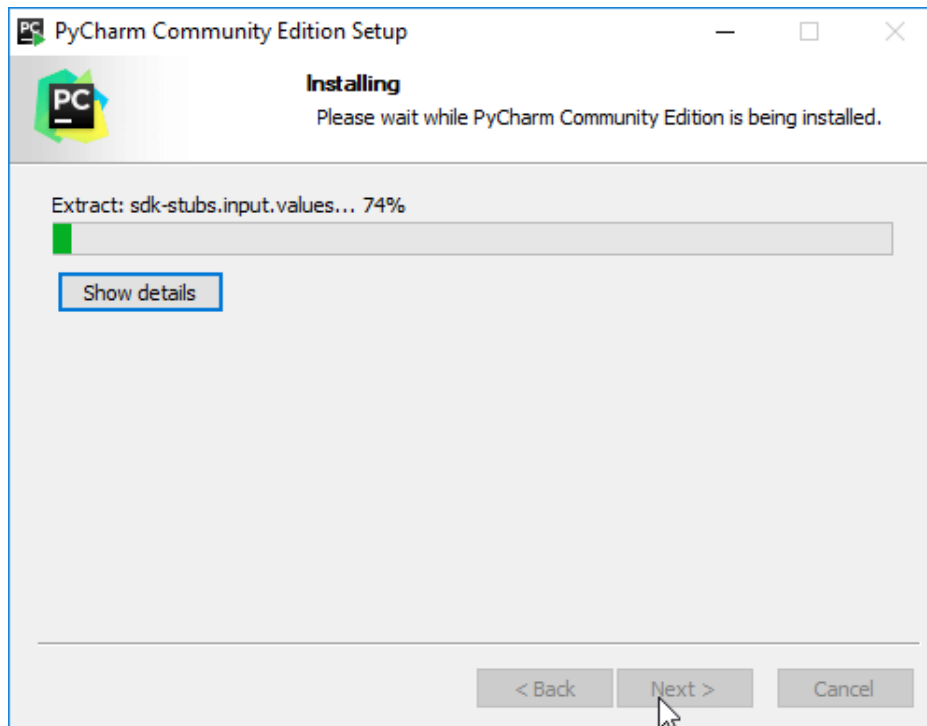


Fig 4.7.6 Pycharm Community Edition is being installed

Step 7) Once installation finished, you should receive a message screen that PyCharm is installed. If you want to go ahead and run it, click the "Run PyCharm Community Edition" box first and click "Finish".

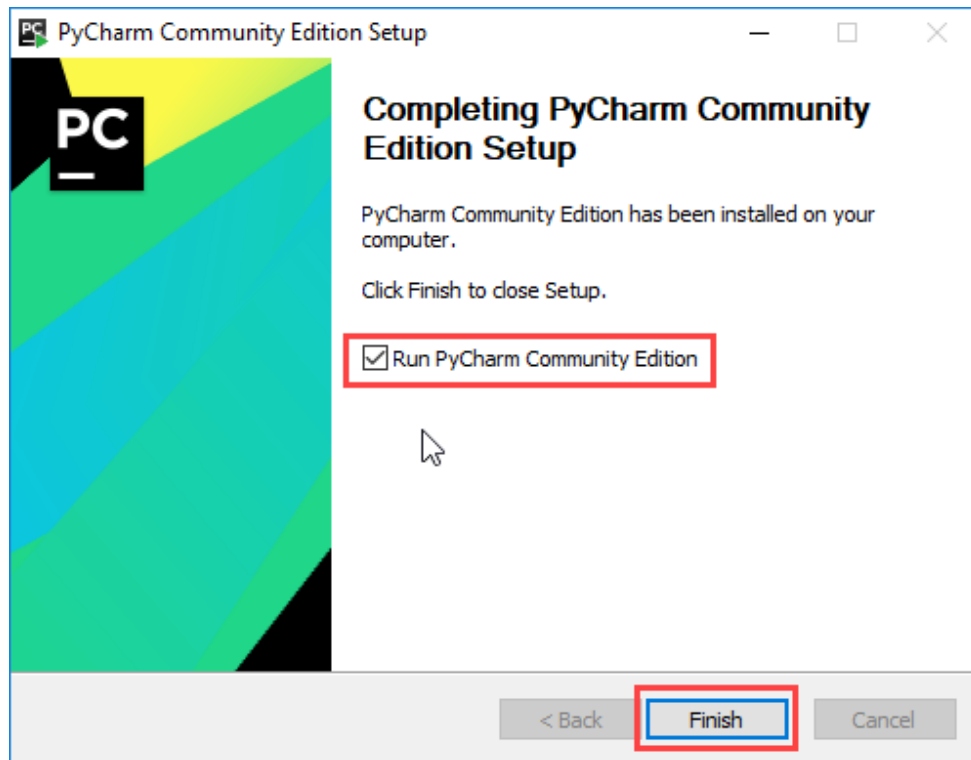


Fig 4.7.7 Completing Pycharm Community Edition Setup

Step 8) After you click on "Finish," the Following screen will appear.

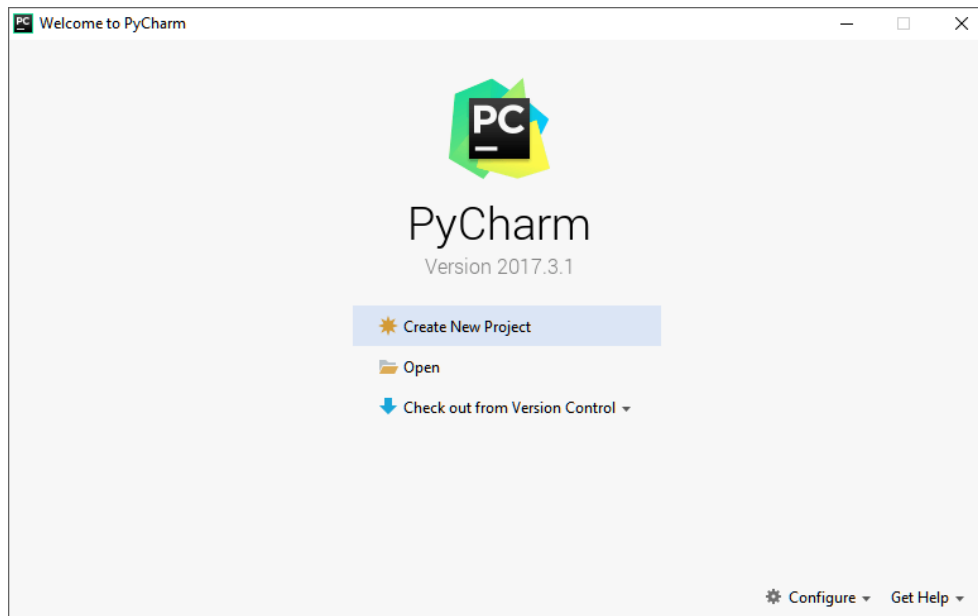


Fig 4.7.8 Pycharm Successfully Installed

Click on Configure > Settings to open up settings in PyCharm

Search for "Project Interpreter". Click on Add local via the settings on the right side

Select "conda environment"

Click on "Existing environment" and navigate to the environment that you want to use. Note that you have to select the bin/python file inside the conda environment for PyCharm to be able to recognize the environment

Make sure to click the "Make available to all projects" if you want the interpreter to be used by multiple projects

Click ok and you are done

4.8 How to download and install anaconda

1. Go to the **Anaconda Website** and choose a Python 3.x graphical installer (A) or a Python 2.x graphical installer (B). If you aren't sure which Python version you want to install, choose Python 3. Do not choose both.

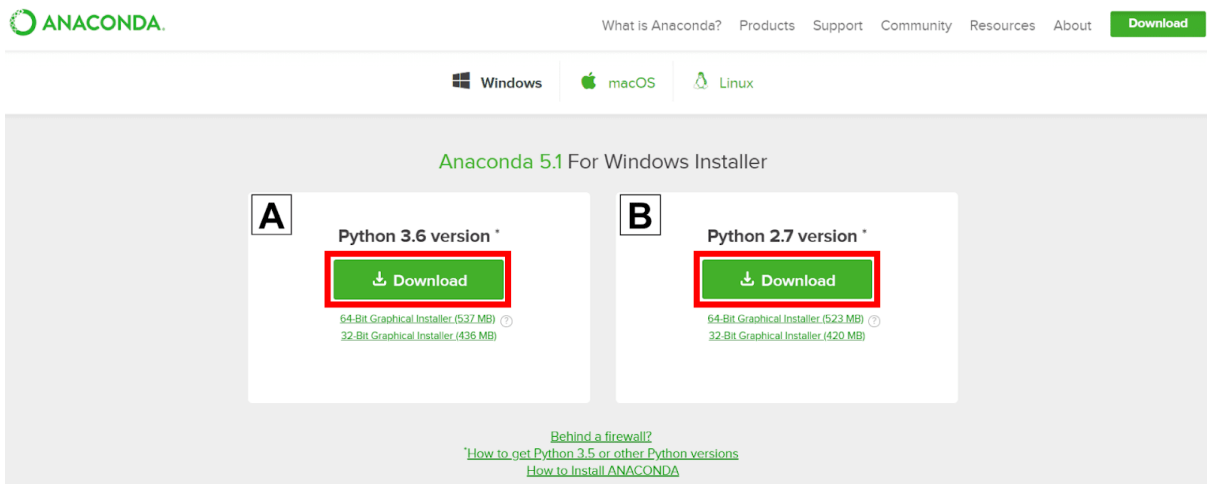


Fig 4.8.1 Anaconda Installer

2. Locate your download and double click it.

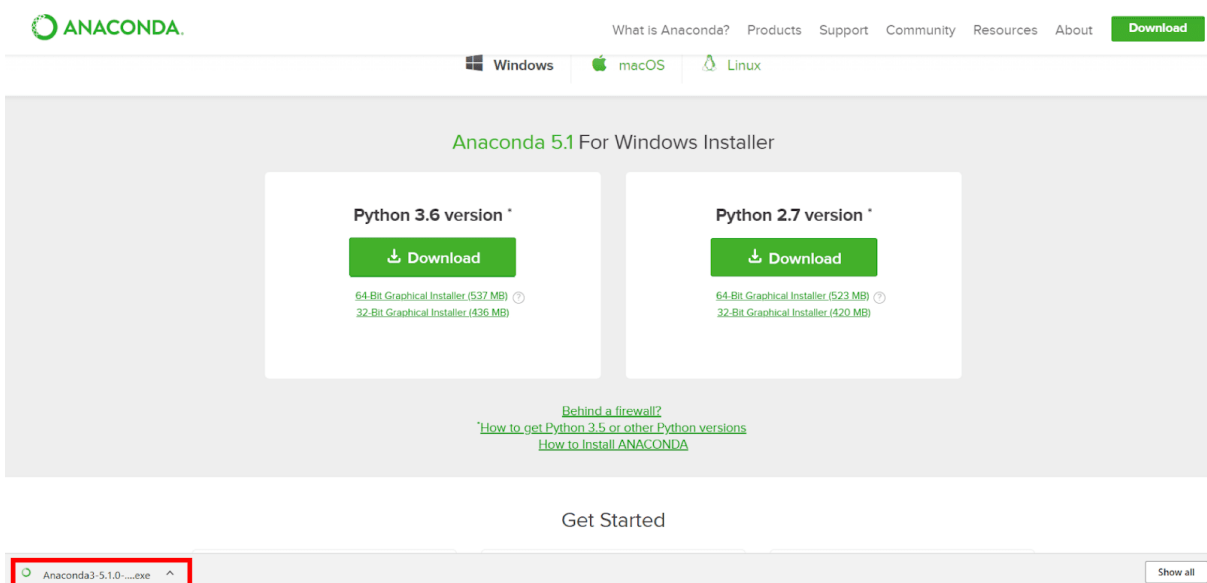


Fig 4.8.2 Choose Anaconda Install Location

When the screen below appears, click on Next.

3. Read the license agreement and click on I Agree.

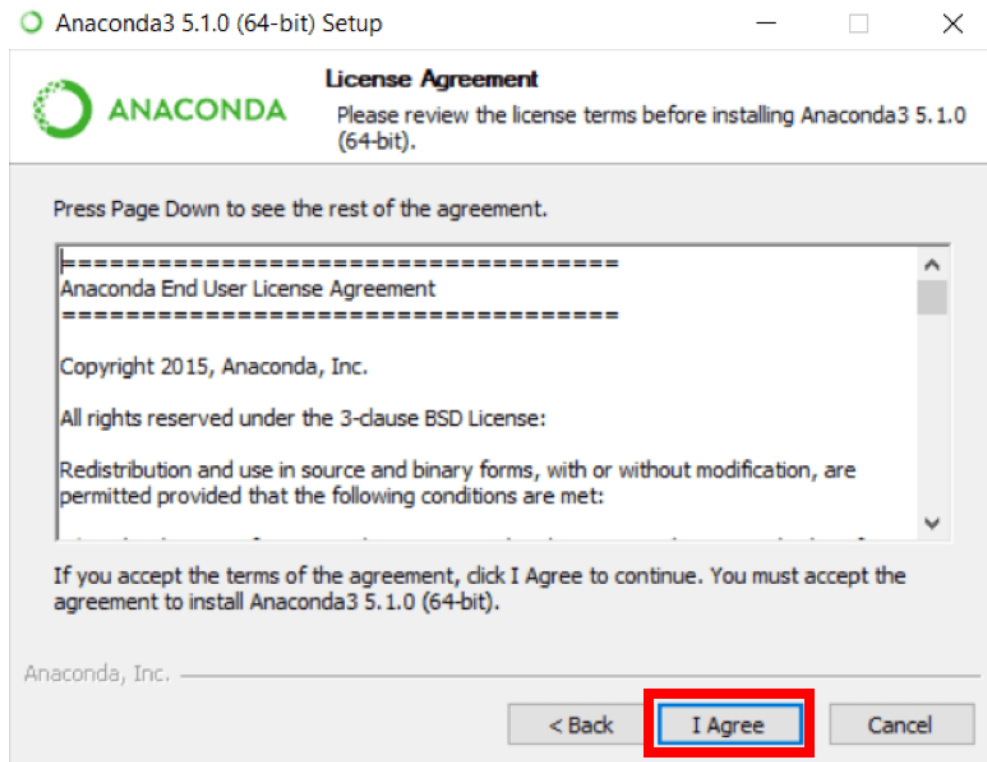


Fig 4.8.3 License Agreement Setup of Anaconda

4. Click on Next.

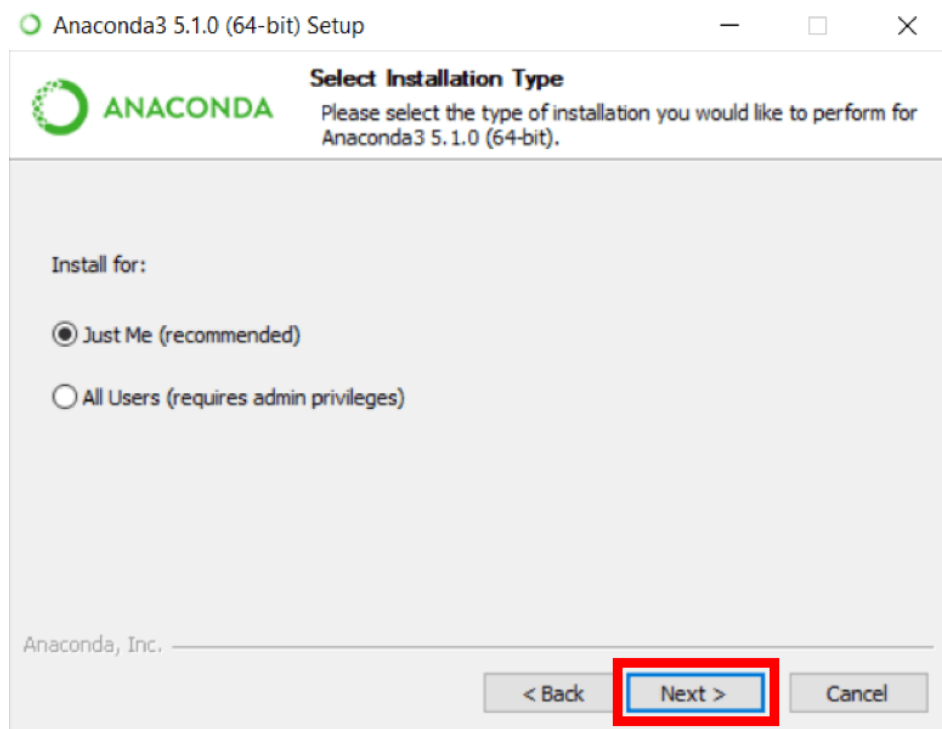


Fig 4.8.4 Anaconda Select Installation Type

5. Note your installation location and then click Next.

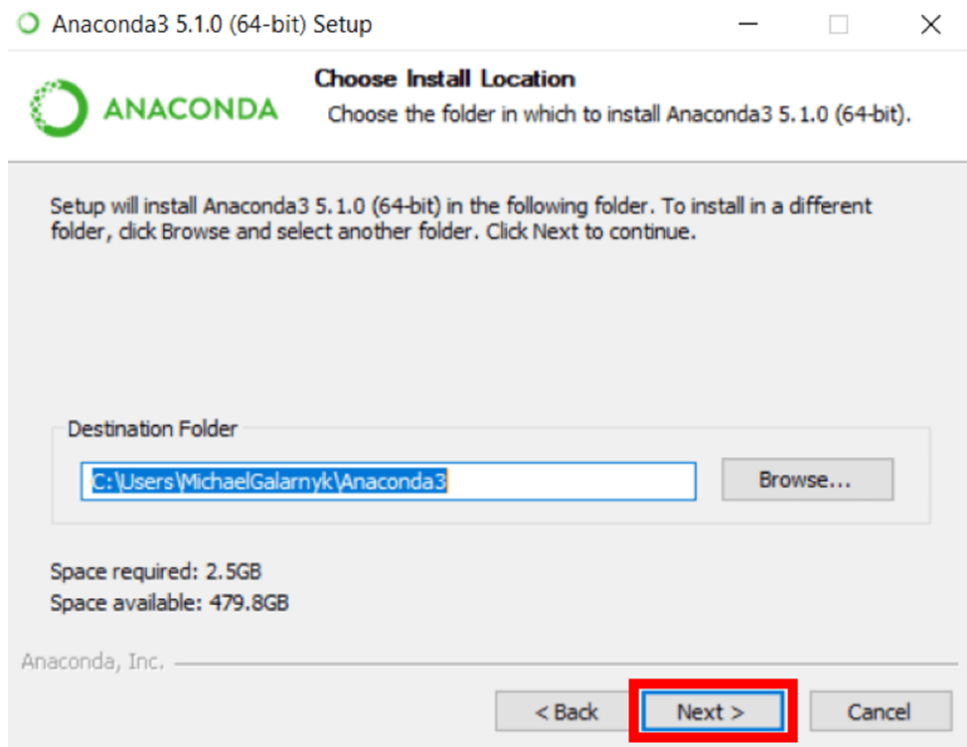


Fig 4.8.5. Anaconda Install Location

6. This is an important part of the installation process. The recommended approach is to not check the box to add Anaconda to your path. This means you will have to use Anaconda Navigator or the Anaconda Command Prompt (located in the Start Menu under "Anaconda") when you wish to use Anaconda (you can always add Anaconda to your PATH later if you don't check the box). If you want to be able to use Anaconda in your command prompt (or git bash, cmd, powershell etc), please use the alternative approach and check the box

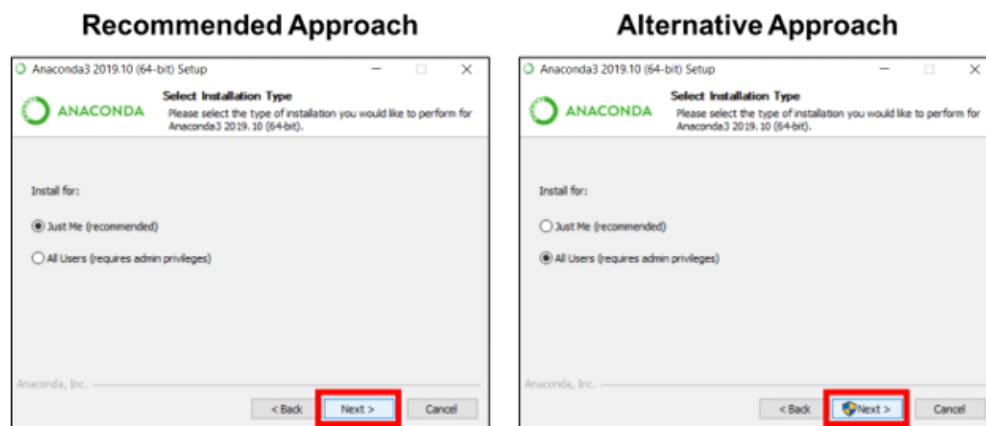


Fig 4.8.6 Recommended and Alternative Approach

7. Click on Next

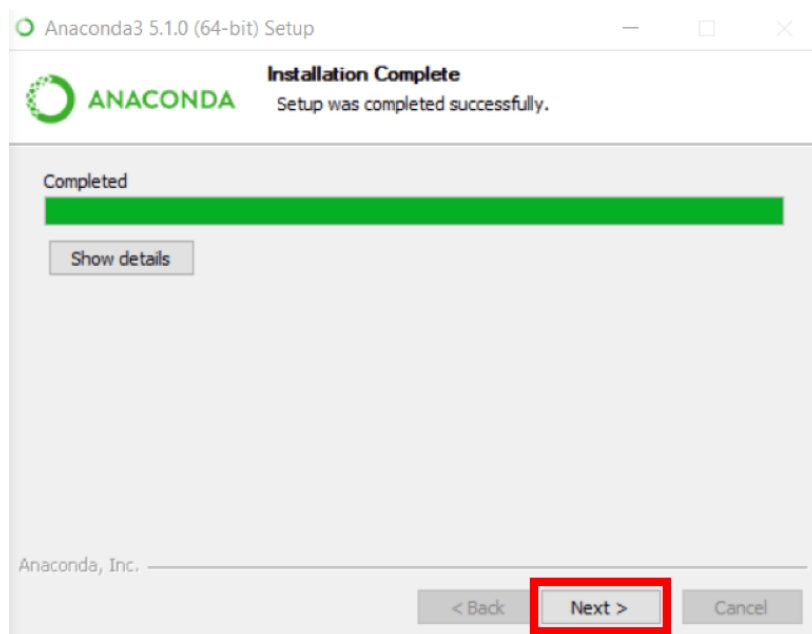


Fig 4.8.7 Anaconda Installation Setup was Completed

8. You can install Microsoft VSCode if you wish, but it is optional.

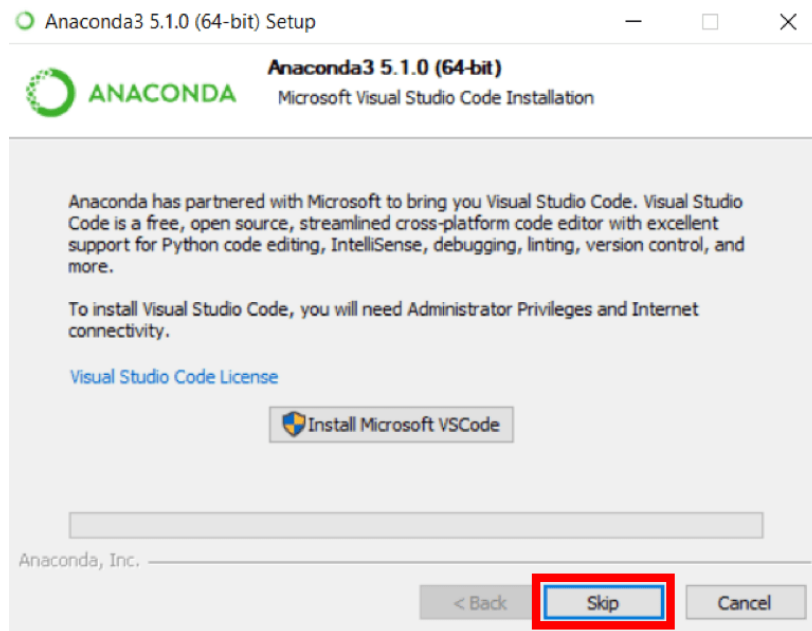


Fig 4.8.8 Microsoft Visual Studio Code Installation

9. Click on Finish.

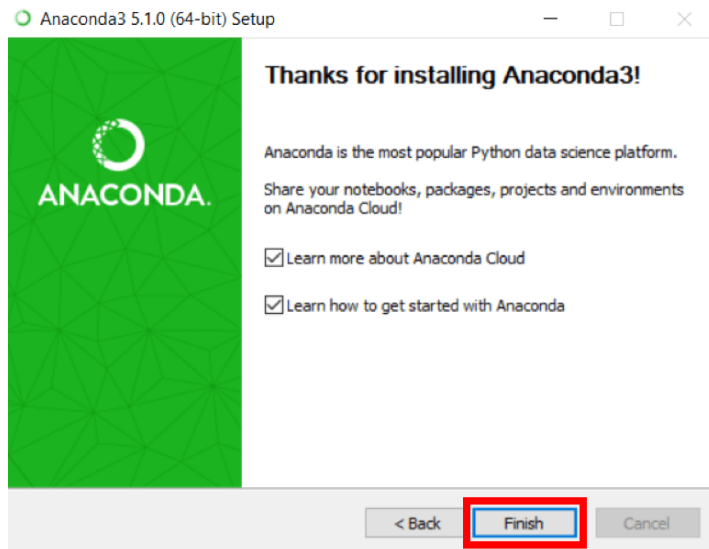


Fig 4.8.9 Anaconda Installation Completed

4.9 How to Add Anaconda to Path (Optional)

This is an **optional** step. This is for the case where you didn't check the box in step 6 and now want to add Anaconda to your Path. The advantage of this is that you will be able to use Anaconda in your Command Prompt, Git Bash, cmd etc.

1. Open a Command Prompt.

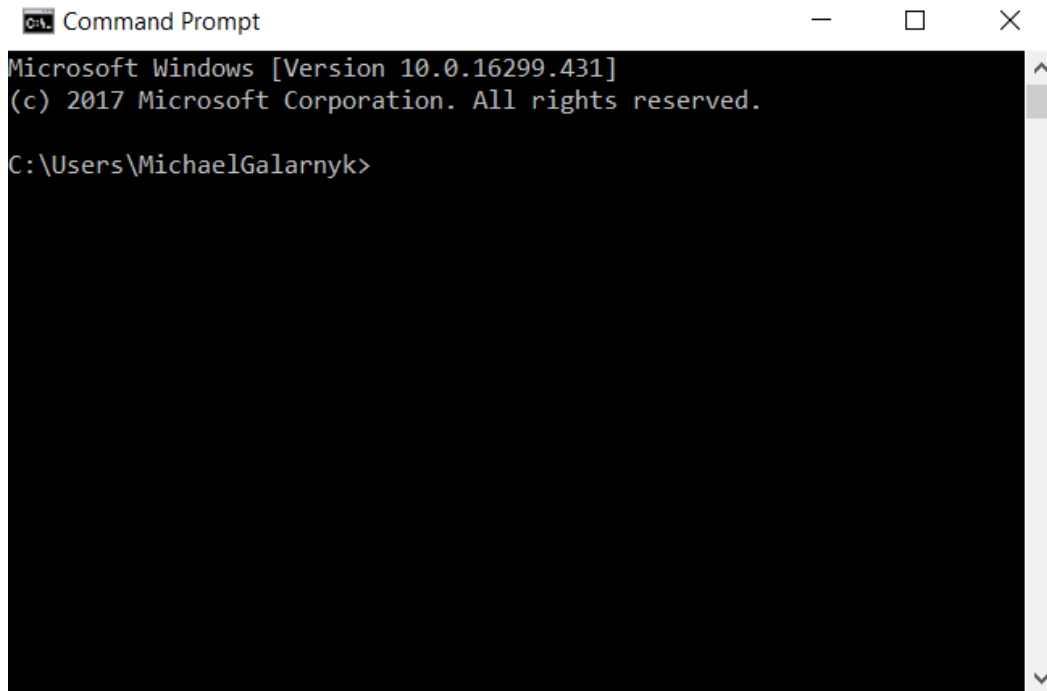


Fig 4.9.1 Command Prompt

2. Check if you already have Anaconda added to your path. Enter the commands below into your Command Prompt. This is checking if you already have Anaconda added to your path. If you get a command **not recognized** error like in the left side of the image below, proceed to step 3. If you get an output similar to the right side of the image below, you have already added Anaconda to your path.

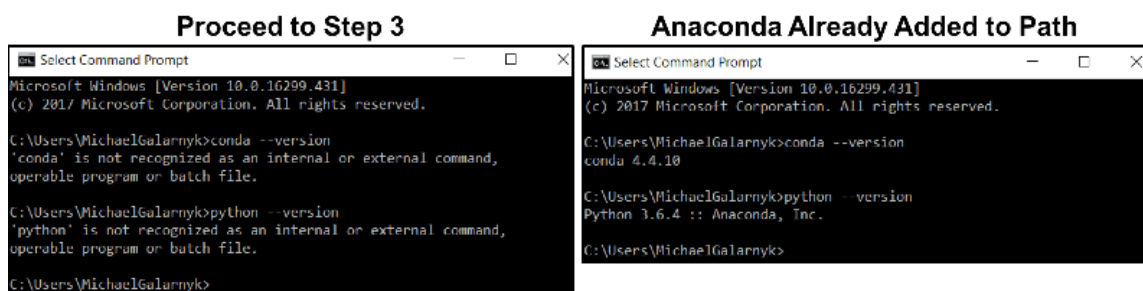
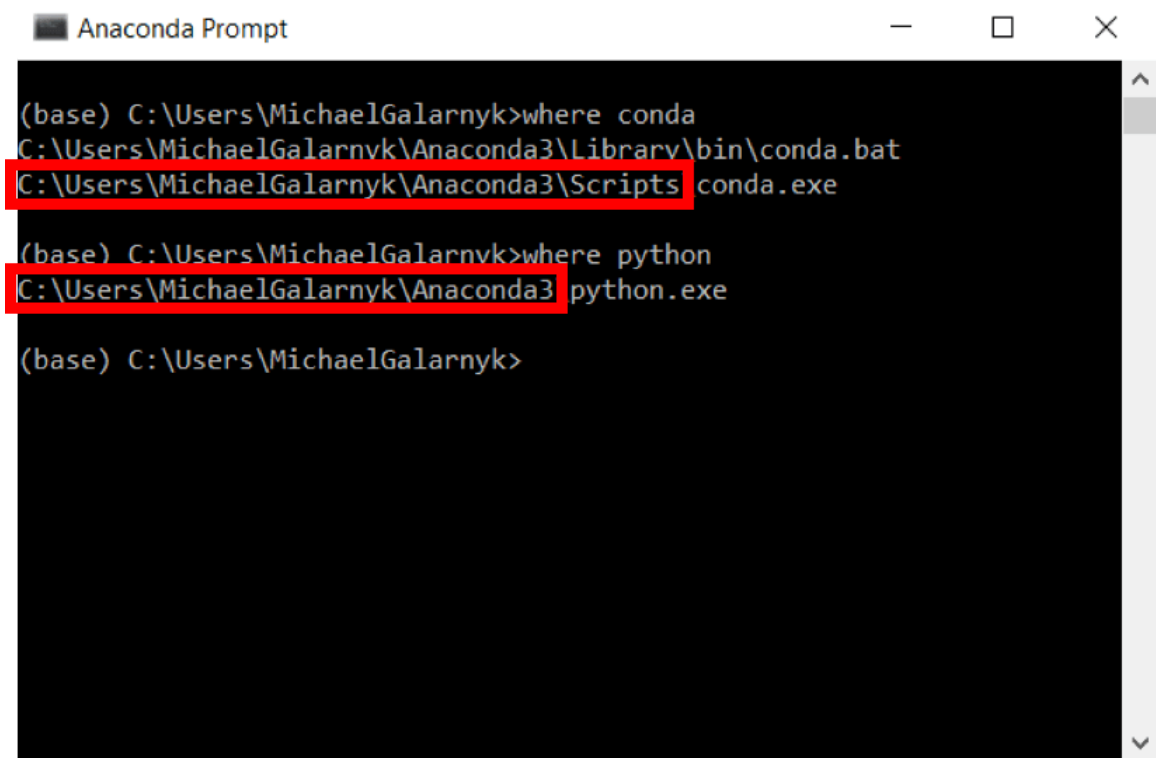


Fig 4.9.2 Anaconda Path

3. If you don't know where your conda and/or python is, open an **Anaconda Prompt** and type in the following commands. This is telling you where conda and python are located on your computer.



```
Anaconda Prompt
(base) C:\Users\MichaelGalarnyk>where conda
C:\Users\MichaelGalarnyk\Anaconda3\Library\bin\conda.bat
C:\Users\MichaelGalarnyk\Anaconda3\Scripts conda.exe

(base) C:\Users\MichaelGalarnyk>where python
C:\Users\MichaelGalarnyk\Anaconda3 python.exe

(base) C:\Users\MichaelGalarnyk>
```

Fig 4.9.3 Anaconda Prompt

4. Add conda and python to your PATH. You can do this by going to your Environment Variables and adding the output of step 3 (enclosed in the red rectangle) to your path. If you are having issues, here is a short video on adding conda and python to your PATH.

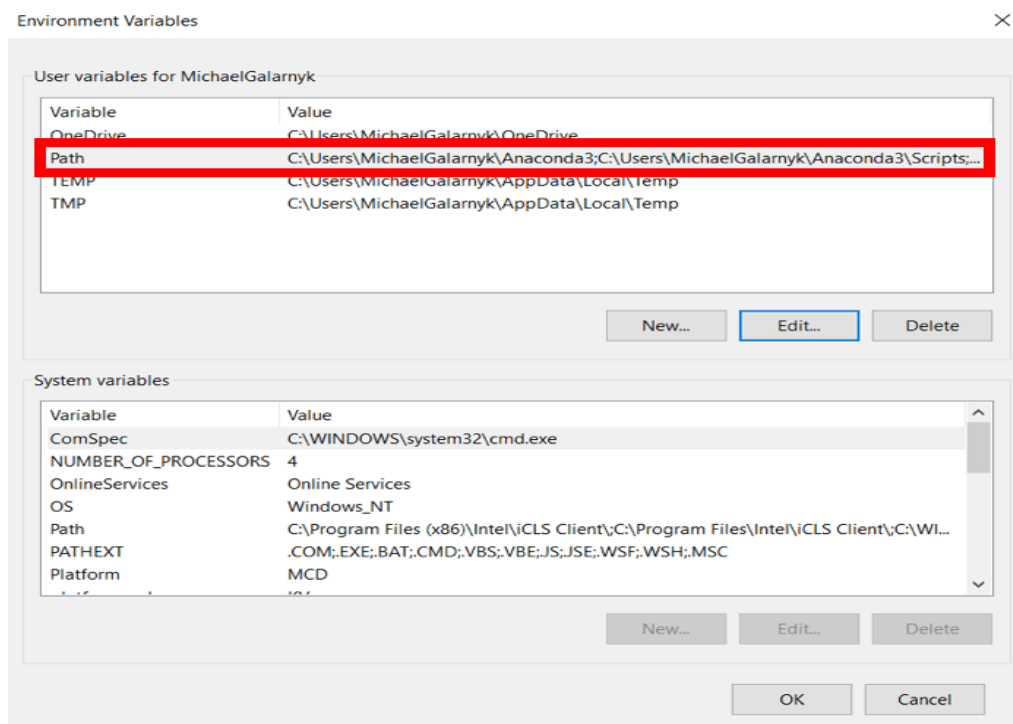


Fig 4.9.4 Environment Variables

5. Open a new Command Prompt. Try typing `conda --version` and `python --version` into the Command Prompt to check to see if everything went well.

```

Microsoft Windows [Version 10.0.16299.431]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\MichaelGalarnyk>conda --version
conda 4.4.10

C:\Users\MichaelGalarnyk>python --version
Python 3.6.4 :: Anaconda, Inc.

C:\Users\MichaelGalarnyk>
  
```

Fig 4.9.5 Select Command Prompt

CHAPTER 5

RESULTS AND DISCUSSION

5.1 SYSTEM MODULES:

- Module 1: Upload Image
- Module 2: Color Filtering
- Module 3: Image Segmentation
- Module 4: Blob Detection
- Module 5: Counting cells

Module 1:

A dataset (or data set) is a collection of data, usually presented in tabular form. Each column represents a particular variable. Each row corresponds to a given member of the dataset in question. It lists values for each of the variables, such as height and weight of an object. Each value is known as a datum.

We have chosen to use a publicly-available Healthcare dataset which contains a relatively small number of inputs and cases. The data is arranged in such a way that will allow those trained in medical disciplines to easily draw parallels between familiar statistical and novel ML techniques. Additionally, the compact dataset enables short computational times on almost all modern computers.

Module 2: Colour Filtering

- OpenCV has some built-in functions to perform Color detection and Segmentation operations. Colour detection is a technique of detecting any

colour in a given range of HSV (hue saturation value) colour space. To filter a colour using OpenCV, we have to specify the hue range of the colour, we need to filter, and create a mask for the colour.

- Color image filtering and enhancement refer to the process of noise reduction in the colour image and enhancement of the visual quality of the image input. Noise encountered into the image data reduces the perceptual quality of an image and thus limits the performance of the imaging system.

Module 3: Segmentation

Import the libraries. Reading the sample image on which we will be performing all the operations. The process of splitting images into multiple layers, represented by a smart, pixel-wise mask is known as Image Segmentation. It involves merging, blocking, and separating an image from its integration level.

Module 4: Blob Detection

Blob stands for Binary Large Object where the term “Large” focuses on the object of a specific size, and that other “small” binary objects are usually considered as noise. In computer vision, blob detection methods are aimed at detecting regions in a digital image that differ in properties, such as brightness or colour, compared to surrounding regions.

Module 5: Counting the Cell

By using Tkinter framework based on OpenCV method to count the RBC, WBC and platelet of the system. Cell counts are important for monitoring cell health and proliferation rate, assessing immortalization or transformation, seeding cells for subsequent experiments, transfection or infection, and preparing for cell-based assays.

RBC: Your body may increase red blood cell production to compensate for any

condition that results in low oxygen levels, including: Heart disease (such as congenital heart disease in adults) Heart failure. A condition present at birth that reduces the oxygen-carrying capacity of red blood cells (hemoglobinopathy).

WBC: White blood cells are part of the body's immune system. They help the body fight infection and other diseases. Types of white blood cells are granulocytes (neutrophils, eosinophils, and basophils), monocytes, and lymphocytes (T cells and B cells). Function of White Blood Cells. White blood cells are a part of your immune system that protects your body from infection. These cells circulate through your bloodstream and tissues to respond to injury or illness by attacking any unknown organisms that enter your body.

Platelet: Platelets, or thrombocytes, are small, colorless cell fragments in our blood that form clots and stop or prevent bleeding. Platelets are made in our bone marrow, the sponge-like tissue inside our bones. Bone marrow contains stem cells that develop into red blood cells, white blood cells, and platelets.

5.2 COLLECTED DATASET

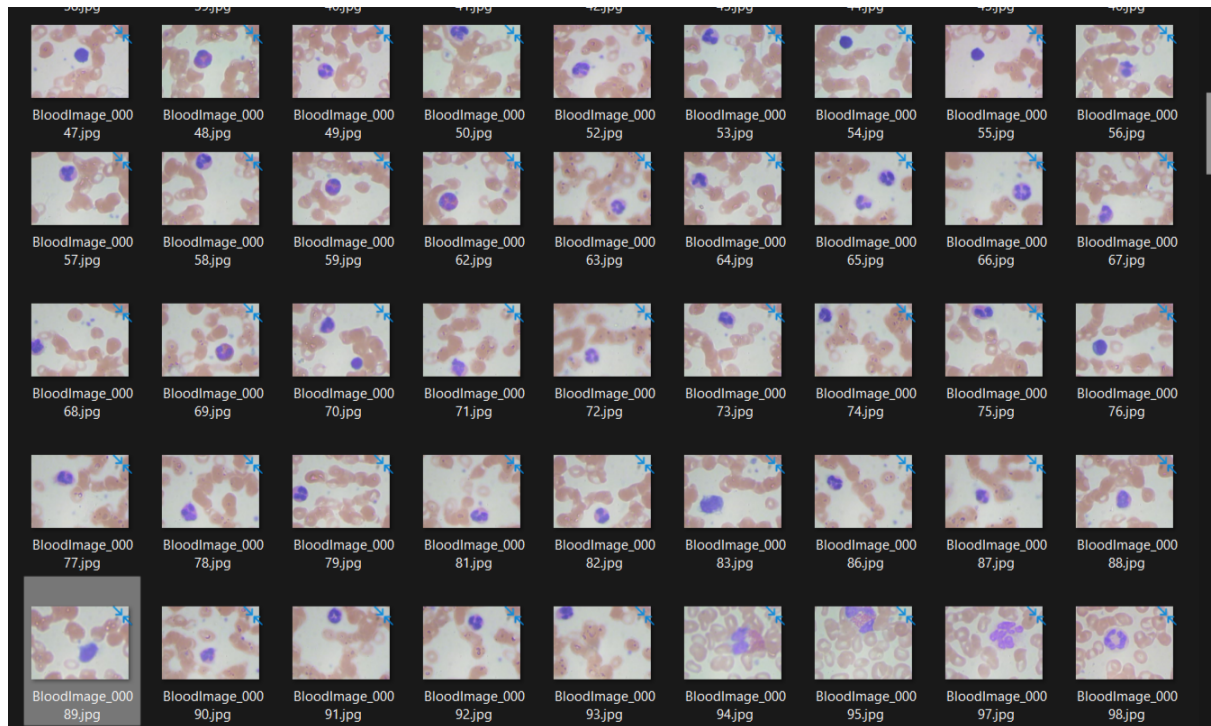


Fig 5.2.1

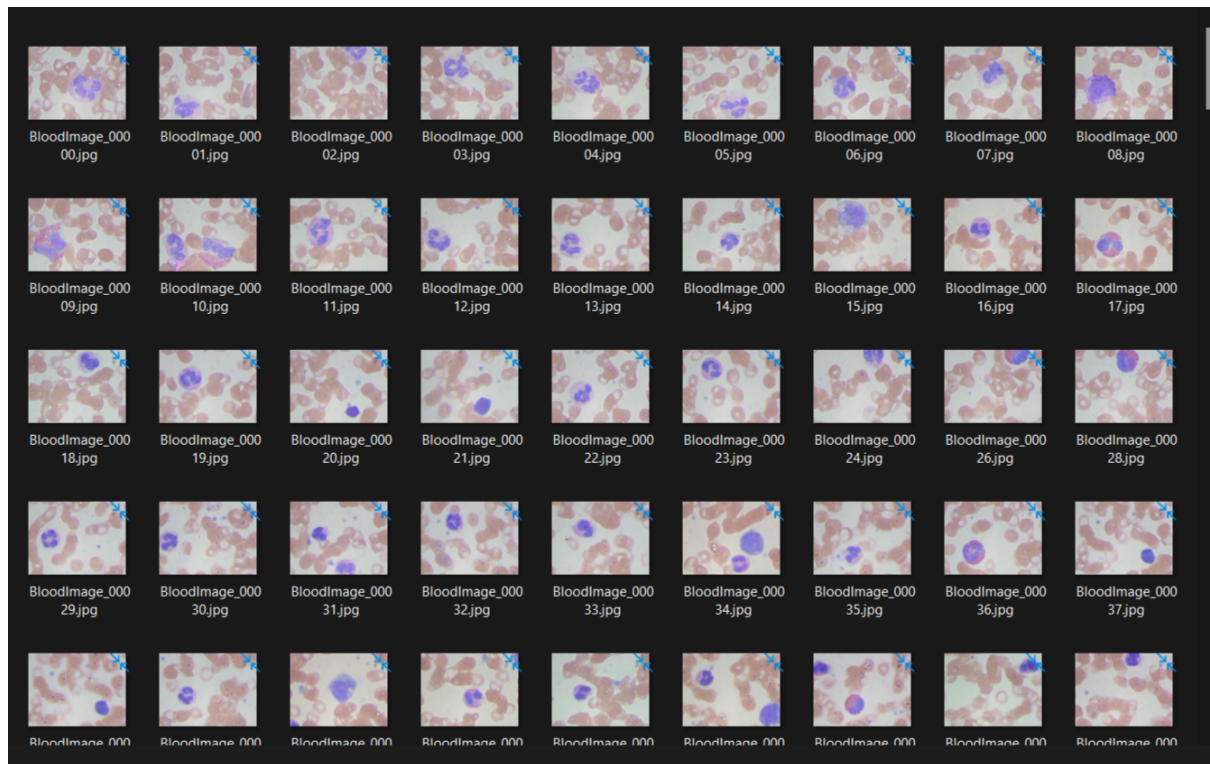


Fig 5.2.2

5.3 CODING FOR SEGMENTATION

```
Anaconda Prompt (anaconda) x + -
(base) C:\Users\messi>cd C:\blood cell\SOURCE CODE
(base) C:\blood cell\SOURCE CODE>python visualize.py
python: can't open file 'C:\blood cell\SOURCE CODE\visualize.py': [Errno 2] No such file or directory
(base) C:\blood cell\SOURCE CODE>cd C:\blood cell\SOURCE CODE\scripts
(base) C:\blood cell\SOURCE CODE\scripts>python visualize.py
```



Fig 5.3.1

CHOOSING SAMPLE IMAGE FROM COLLECTED DATA SET

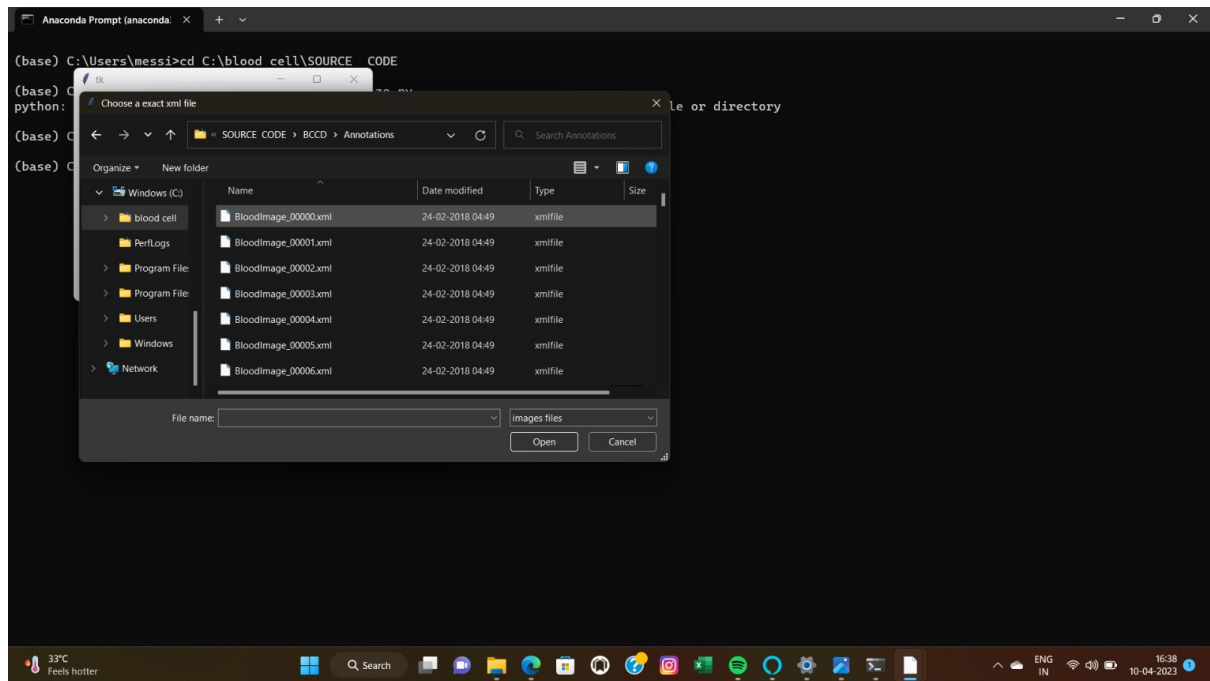


Fig 5.3.2

FINAL OUTPUT OF DETECTED BLOOD CELLS

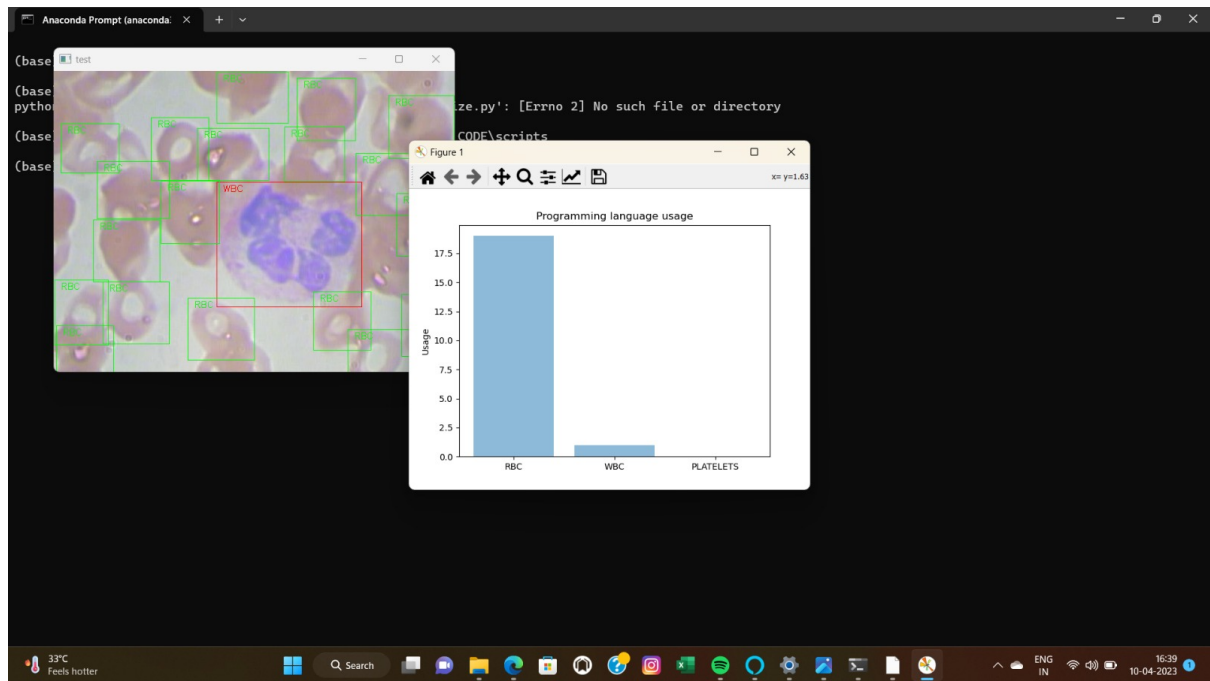


Fig 5.3.3

CHAPTER 6

CONCLUSION AND SUMMARY

6.1 CONCLUSION

The researchers presented more accurate blood cell counting using a new algorithm with the help of python OpenCV programming language. The implementation of image processing and analysis for the platelet, red blood and white blood cells was made possible and resulted to high level of accuracy. The algorithm was able to meet the margin of error of less than 10%.

FUTURE WORK

Furthermore, improved accessibility and portability of the software used were obtained through its on-line database support and free ware feature.

REFERENCE

- [1] Abbas S, "Microscopic images dataset for automation of RBCs counting", Data in Brief 5 2015., pp. 35-40.
- [2] Ahasan R, Ratul A.U, A.S.M. Bakibillah, "White blood cells nucleus segmentation from microscopic images of strained peripheral blood film during leukemia and normal condition", 2016 5th International Conference on Informatics, Electronics and Vision, ICIEV 2016, pp. 361-366.
- [3] Adagale S.S, Pawar S.S, "Image Segmentation using PCNN and Template Matching for", IEEE 2013.
- [4] Gautam, Anjali, Bhadauria, Harvindra, "Classification of White Blood Cells Based on Morphological Features", 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 2363-2368.
- [5] Hari J, Prasad A.S, Rao S.K, "Separation and counting of blood cells using geometrical features and distance transformed watershed", Proceedings of the IEEE International Caracas Conference on Devices, Circuits and Systems, ICCDCS 2014 pp. 1-5.
- [6] Jha K.K, Das B.K, Dutta H.S, "Detection of abnormal blood cells on the basis of nucleus shape and counting of WBC", Proceeding of the IEEE International Conference on Green Computing, Communication and Electrical Engineering, ICGCCEE 2014.
- [7] Kaur P, Sharma V, N. Garg, "Platelet Count using Image Processing", 2016 International Conference on Computing for Sustainable Global Development, 2016, pp. 2574-2577.

- [8] Manik S, Lalit, M. Saini, N. Vadera, "Counting and Classification of White Blood Cell using Artificial Neural Network (ANN)", 1st IEEE International Conference on Power Electronics. Intelligent Control and Energy Systems (ICPEICES-2016, pp. 1-5.
- [9] Loddo, Andrea, Putzu, Lorenzo, Ruberto, Cecilia DI, Fenu, Gianni, "A Computer-Aided System for Differential Count from Peripheral Blood Cell Images", Proceedings - 12th International Conference on Signal Image Technology and Internet-Based Systems, SITIS 2016, pp. 112-118.
- [10] Lou J, M. Zhou, Q. Li, C. Yuan, H. Liu, "An automatic red blood cell counting method based on spectral images", 2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), pp. 1391-1396.
- [11] Sajjad M, S. Khan, Z. Jan, K. Muhammad, H. Moon, J.T. Kwak, S. Rho, Seungmin, W. Sung I. Mehmood, "Leukocytes Classification and Segmentation in Microscopic Blood Smear: A Resource-Aware
- [12] Tulsani, Hemant, Gupta, Rashmi, Kapoor, Rajiv, "An Improved Methodology for Blood Cell Counting", IEEE Impact 2013, pp. 88- 92.

APPENDIX

```
import os, sys, random
import xml.etree.ElementTree as ET
import mxnet as mx
```

```
classes = ["RBC", "WBC", "Platelets"]
```

```
ratio = 0.9
```

```
path = 'D:/1. PYTHON 2019 BACKUPS/BLOOD CELL COUNTING USING PYTHON OPEN-  
CV/SOURCE CODE/BCCD'
```

```

def gen_det_rec(classes, dataset_dir, ratio=1):
    assert ratio <= 1 and ratio >= 0
    img_dir = os.path.join(dataset_dir, "JPEGImages")
    label_dir = os.path.join(dataset_dir, "Annotations")
    img_names = os.listdir(img_dir)
    img_names.sort()

    label_names = os.listdir(label_dir)
    label_names.sort()
    file_num = len(img_names)
    assert file_num==len(label_names)

    idx_random = list(range(file_num))
    random.shuffle(idx_random)
    idx_train=idx_random[:int(file_num*ratio)+1]
    idx_val=idx_random[int(file_num*ratio)+1:]

    with open("train.lst", "w") as train_lst:
        print("Writing in train.lst...")
        if idx_val:
            with open("val.lst", "w") as val_lst:
                print("Writing in val.lst...")

        for idx in range(file_num):
            each_img_path = os.path.join(img_dir, img_names[idx])
            each_label_path = os.path.join(label_dir, label_names[idx])
            tree = ET.parse(each_label_path)
            root = tree.getroot()
            size = root.find('size')
            width = float(size.find('width').text)
            height = float(size.find('height').text)
            label = []
            label.append(str(idx))
            label.append('4\t5\t'+str(width)+'\t'+str(height))

```

```

for obj in root.iter('object'):
    cls_name = obj.find('name').text
    if cls_name not in classes:
        continue
    cls_id = classes.index(cls_name)
    xml_box = obj.find('bndbox')
    xmin = float(xml_box.find('xmin').text) / width
    ymin = float(xml_box.find('ymin').text) / height
    xmax = float(xml_box.find('xmax').text) / width
    ymax = float(xml_box.find('ymax').text) / height
    for i in [cls_id, xmin, ymin, xmax, ymax]:
        label.append(str(i))

label.append(each_img_path)
label = '\t'.join(label)
if idx in idx_train:
    train_lst.write(label+'\n')
else:
    val_lst.write(label+'\n')

```

```

gen_det_rec(classes, path, ratio)
from mxnet import gluon
from mxnet import image
from mxnet import nd
import matplotlib as mpl
import matplotlib.pyplot as plt

```

```

data_shape = (480, 640)
batch_size = 64
data_dir = '../.. /BCCD'

```

```

def get_iterators(data_shape, batch_size):
    class_names = ["RBC", "WBC", "Platelets"]

```

```

num_class = len(class_names)
train_iter = image.ImageDetIter(
    batch_size=batch_size,
    data_shape=(3, data_shape[0], data_shape[1]),
    path_imgrec=data_dir+'train.rec',
    path_imgidx=data_dir+'train.idx',
    shuffle=True,
    mean=True,
    rand_crop=1,
    min_object_covered=0.95,
    max_attempts=200)
val_iter = image.ImageDetIter(
    batch_size=batch_size,
    data_shape=(3, data_shape[0], data_shape[1]),
    path_imgrec=data_dir+'val.rec',
    shuffle=False,
    mean=True)
return train_iter, val_iter, class_names, num_class
train_data, test_data, class_names, num_class = get_iterators(
    data_shape, batch_size)

batch = test_data.next()
def box_to_rect(box, color, linewidth=3):
    """convert an anchor box to a matplotlib rectangle"""
    box = box.asnumpy()
    return plt.Rectangle(
        (box[0], box[1]), box[2]-box[0], box[3]-box[1],
        fill=False, edgecolor=color, linewidth=linewidth)
_, figs = plt.subplots(3, 3, figsize=(6,6))
for i in range(3):
    for j in range(3):
        img, labels = batch.data[0][3*i+j], batch.label[0][3*i+j]
        # (3L, 256L, 256L) => (256L, 256L, 3L)
        img = img.transpose((1, 2, 0))

```

```

img = img.clip(0,255).asnumpy()/255
fig = figs[i][j]
fig.imshow(img)
for label in labels:
    label[1] *= data_shape[1]
    label[3] *= data_shape[1]
    label[2] *= data_shape[0]
    label[4] *= data_shape[0]
    if label[0] == 0:
        rect = box_to_rect(label[1:5], 'red', 0.5)
    elif label[0] == 1:
        rect = box_to_rect(label[1:5], 'green', 0.5)
    elif label[0] == 2:
        rect = box_to_rect(label[1:5], 'blue', 0.5)
    else:
        rect = box_to_rect(label[1:5], 'black', 2)
    fig.add_patch(rect)
fig.axes.get_xaxis().set_visible(False)
fig.axes.get_yaxis().set_visible(False)
plt.show()

```