VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELGAUM - 590018, KARNATAKA

A PROJECT REPORT
ON

"IDENTIFICATION OF LEUKEMIA CELLS FROM MICROSCOPIC IMAGES USING DEEP LEARNING TECHNIQUES"

A dissertation work submitted in partial fulfillment of the requirement for the Award of the degree of

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE & ENGINEERING

Submitted by

SHREERAKSHA	**(1EW17CS141)**
ANITHA PATEL	**(1EW17CS010)**
VISHWANATH C R	**(1EW17CS180)**
SHARADHA BHAT	**(1EW16CS142)**

Under the Guidance of

Prof. DHANRAJ S
Assistant Professor
Dept. of CSE, EWIT

EAST WEST INSTITUTE OF TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
EAST WEST INSTITUTE OF TECHNOLOGY
BENGALURU-560091

2020-2021

EAST WEST INSTITUTE OF TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

(Affiliated to Visvesvaraya Technological University, Belagavi | Recognized by Govt. of Karnataka |

Approved by AICTE, New Delhi)

CERTIFICATE

This is to certify that the project work entitled **"IDENTIFICATION OF LEUKEMIA CELLS FROM MICROSCOPIC IMAGES USING DEEP LEARNING TECHNIQUES"** is a bonafide work carried out by **SHREERAKSHA (1EW17CS141), ANITHA PATEL (1EW17CS010), VISHWANATH C R (1EW17CS180), SHARADHA BHAT (1EW16CS142),** in partial fulfillment for the award of Bachelor of Engineering in Computer Science and Engineering of the Visvesvaraya Technological University, Belgaum during the academic year 2020- 2021. It is certified that all the suggestions/corrections indicated for Internal Assessment have been incorporated in the report deposited in the department library. The project report has been approved as it satisfies the academic requirements in respect to project work prescribed for the said degree.

Signature of Internal Guide	Signature of Head of Department	Signature of Principal
Prof. Dhanraj S	**Prof. Chetana Srinivas**	**Dr. K Channakeshavalu**
Asst. Professor, CSE	Professor & HOD, CSE	Principal/Director, EWIT

EXTERNAL VIVA

Name of External Examiner	**Signature with Date**
1. _____	_____
2. _____	_____

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned our effort with success.

We express our sincere gratitude to our principal **Dr. K Channakeshavulu**, Principal, East West Institute of Technology, Bangalore for his inspiration.

We wish to place on record our grateful thanks to **Prof. Chetana Srinivas**, Professor and head of the department, Computer Science and Engineering, East West Institute of Technology Bangalore for providing encouragement and guidance.

We would also like to thank our guide **Prof. Dhanraj S.**, Assistant prof, Dept. of Computer Science and Engineering, for the timely inspection and guiding me throughout the process.

Finally, express our sincere thanks to our parents, well-wishers and friends for their moral support, encouragement & help throughout the completion of the internship work.

SHREERAKSHA (1EW17CS141)
ANITHA PATEL (1EW17CS010)
VISHWANATH C R (1EW17CS180)
SHARADHA BHAT (1EW16CS142)

DECLARATION

SHREERAKSHA (1EW17CS141), ANITHA PATEL (1EW17CS010), VISHWANATH C R (1EW17CS180), SHARADHA BHAT (1EW16CS142), students of eighth semester Bachelor of Engineering in the Department of Computer Science and Engineering of East West Institute of Technology, Bangalore – 560091, hereby declare that the project entitled "**IDENTIFICATION OF LEUKEMIA CELLS FROM MICROSCOPIC IMAGES USING DEEP LEARNING TECHNIQUES**" has been carried out by us under the supervision of Internal Guide **Prof. DHANRAJ S**, Assistant Professor, Department of Computer Science and Engineering, EWIT, Bangalore submitted in the fulfillment of the course requirements for the award of the degree of Bachelor of Engineering in Computer Science and Engineering of Visvesvaraya Technological University during the academic year 2020-2021

Date:

Place: Bengaluru

SHREERAKSHA (1EW17CS141)	**ANITHA PATEL (1EW17CS010)**	**VISHWANATH C R (1EW17CS180)**	**SHARADHA BHAT (1EW16CS142)**

ABSTRACT

Health is the highly marked statistical research aspect in which cancer is pruning. Cancer is the abnormal production of cells in the body. Now, if an imbalance occurs in blood forming tissue including bone marrow it hinders the body's ability to fight infection. This malfunctioning is termed as Leukemia. Traditionally, a hematologist manually examined the reports with microscopic images to test for Leukemia cells. However, this perusal explains an automated identification of the same by eradicating errors using deep learning methods, mainly Convolutional Neural Network (CNN). The dataset has blood smear images; both infected and non-infected which is initially preprocessed for feature extraction and further analysis. This follows- training the model, prediction of cancer and computing the accuracy. The inferred accuracy records to 97.75%, which finds to be better than other neural networks like ANN, RNN etc. This study with CNN is close enough to the initial architecture with less measurable time and increased automation. Therefore, the model is deployed into an application using Html front end using Django as API where the doctor can effectively work as a tool in identifying Leukemia.

LIST OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The word Leukemia that means 'white blood' derived from the characteristic high white blood cell count that presents in most afflicted people before treatment. The high number of white blood cells is apparent when a blood sample is viewed under the microscope; with extra white blood cells, it indicates frequently cells being immature or dysfunctional. The excessive number of cells can also interfere with the level of other cells, causing further harmful imbalance in the blood count.

In terms of how quickly it develops or gets worse, Leukemia's classification is either acute (fast- growing) or chronic (slow growing). Acute Leukemia rapidly progresses and results in the accumulation of immature, functionless blood cells in the bone marrow. With this, cells begin to reproduce and build up in the bone marrow thus reducing its ability to produce healthy cells. Chronic on the other hand progresses much slowly and results in accumulation of mature, but still abnormal WBCs. Hence, the broadly classified Leukemia types are –Acute lymphoblastic leukemia (ALL), Acute myelogenous leukemia (AML), Chronic lymphoblastic leukemia (CLL) and Chronic myelogenous leukemia (CML). Hematologist used manual methods to find the number of infected cells.

If we look into what can cause leukemia then it can be an ancestry, genetic disorders, and unhealthy habits like smoking or any previous chemotherapy treatments. These eventually gives symptoms like swollen lymph nodes, unintentional weight-loss, fatigue and weakness with excessive sweating.

Leukemia is usually treated by a hematologist-oncologist. These are doctors who specialize in blood disorders and cancer. The treatment depends on the type and stage of the cancer. Some forms of leukemia grow slowly and don't need immediate treatment. However, treatment for leukemia usually involves Chemotherapy, Stem cell transplantation, Radiation therapy. Nevertheless, in the fast growing we look forward for a quick and automatic approach. In such cases, Deep learning becomes the spotlight. Deep learning is a type of machine learning that uses a layered algorithmic architecture to analyze data.

In such models, data filtering happens through a cascade of multiple layers, with each successive layer using the output from the previous one to inform its results.

As they process more, data models become more accurate, essentially learning from previous results to refine their ability to make corrections and connections.

Just as biological neurons connect with one another to process information in the brains of animals, electrical signals travel across subsequent layers of nodes to activate a stimulus in the neighboring neuron.

In particular, the subset of deep learning neural nets have thousands to millions of simple processing that are densely interconnected. Neural networks are organized into layers of nodes and they are "feed forward", meaning that data moves in one direction. An individual node connects to several sub-nodes in the layers beneath it. With these assumptions neural nets we broadly classify into Artificial Neural Networks (ANN), Recurrent Neural Networks (RNN) and Convolutional Neural Network (CNN).

CNN is considered to be more powerful than RNN. RNN includes less feature compatibility when compared to CNN. This CNN takes inputs of fixed sizes and generates fixed size outputs. Compared to its predecessors, the main advantage of CNN is that it automatically detects the important features without any human supervision. This is why CNN would be an ideal solution to computer vision and image classification problems.

The proposed model here focuses on CNN in deriving the required inference. Multiplying pixel values by weights and summing them is "convolution". Along with various components this nets has several convolution layers; in taking outputs from previous layers to feed as input to next layers, CNN plays a major role of classifying and recognizing complex objects. This algorithm has its own benefits and special features that surpass the abilities on choosing the latter, which we shall discuss further.

1.1 WHY MACHINE LEARNING WITH PYTHON?

Machine Learning is simply making a computer perform a task without explicitly programming it. In today's world every system that does well has a machine learning algorithm at its heart. Take for example Google Search engine, Amazon Product recommendations, LinkedIn, Facebook etc., all these systems have machine learning algorithms embedded in their systems in one form or the other. They are efficiently utilizing data collected from various channels which helps them get a bigger picture of what they are doing and what they should do.

Python is a widely used high-level programming language for general-purpose programming.

Apart from being open source programming language, python is a great object-oriented, interpreted, and interactive programming language. Python combines remarkable power with very clear syntax. It has modules, classes, exceptions, very high level dynamic data types, and dynamic typing. There are interfaces to many system calls and libraries.

Python is widely considered as the preferred language for teaching and learning Ml (Machine Learning). Few simple reasons are:

- It's simple to learn. As compared to C, C++ and Java the syntax is simpler and Pythonalso consists of a lot of code libraries for ease of use.
- Though it is slower than some of the other languages, the data handling capacity isgreat.
- Open Source! – Python along with R is gaining momentum and popularity in theAnalytics domain since both of these languages are open source.
- Capability of interacting with almost all the third party languages and platforms.

1.2 WHY DEEP LEARNING TECHNIQUE?

Deep Learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called artificial neural networks.

Deep learning is a class of machine learning algorithms that uses multiple layers to progressively extract higher level features from the raw input. For example, in image processing, lower layers may identify edges, while higher layers may identify the concepts relevant to a human such as digits or letters or faces.

In deep learning, each level learns to transform its input data into a slightly more abstract and composite representation. In an image recognition application, the raw input may be a matrix of pixels; the first representational layer may abstract the pixels and encode edges; the second layer may compose and encode arrangements of edges; the third layer may encode a nose and eyes; and the fourth layer may recognize thatthe image contains a face.

Importantly, a deep learning process can learn which features to optimally place in which level on its own.

1.3 TYPES OF MACHINE LEARNING

As with any method, there are different ways to train machine learning algorithms, each with their own advantages and disadvantages. To understand the pros and cons of each type of machine learning, we must first look at what kind of data they ingest. In ML, there are two kindsof data — labeled data and unlabeled data.

- Labeled data has both the input and output parameters in a completely machine- readable pattern, but requires a lot of human labor to label the data, to begin with.
- Unlabeled data only has one or none of the parameters in a machine-readable form. This negates the need for human labor but requires more complex solutions.

There are also some types of machine learning algorithms that are used in very specific use-cases, but three main methods are used today.

➢ Supervised Learning

In a supervised learning algorithm, the input data is labeled such that the data is organized. The computer is able to follow the examples of input- output pairs and train the model to fit the data with good accuracy. Some ofthe supervised learning algorithms are –

- Linear & Multivariate Regression
- Logistic Regression
- Naive Bayes
- Decision Trees
- K-nearest neighbour
- Linear Discriminant Analysis
- Artificial Neural Networks
- Unsupervised Learning

➢ Unsupervised Learning

In Unsupervised Learning, the data is not labeled or categorized. In Unsupervised learning, the data is able to organize itself after it follows a certain pattern in the way the data is distributed.

Unsupervised learning algorithms are complex and are currently under research. Some of the unsupervised learning algorithms are –

- Clustering Analysis
- Anomaly Detection
- Hierarchical Clustering
- Principal Component Analysis

➢ **Reinforcement Learning**

We use these algorithms to choose an action. Also, we can see that it is based on each data point. Moreover, after some time the algorithm changes its strategy to learn better. Also, achieve the best reward.

Machine Learning is used in various industries that require future prediction, identification of patterns and autonomous decision making. It is widely used in healthcare, finance, banking, manufacturing and transportation sectors.

1.4 FUNDAMENTAL STEPS OF IMAGE PROCESSING IN ML

1. Data Collection

- The quantity & quality of your data dictate how accurate our model is
- The outcome of this step is generally a representation of data (Guo simplifies tospecifying a table) which we will use for training
- Using pre-collected data, by way of datasets from Kaggle, UCI, etc., still fits intothis step

2. Data Preparation

- Wrangle data and prepare it for training
- Clean that which may require it (remove duplicates, correct errors, deal withmissing values, normalization, data type conversions, etc.)
- Randomize data, which erases the effects of the particular order in which we collected and/or otherwise prepared our data
- Visualize data to help detect relevant relationships between variables or classimbalances (bias alert!), or perform other exploratory analysis

3. **Choose a model**

- Different algorithms are for different tasks; choose the right one

4. **Train the model**

- The goal of training is to answer a question or make a prediction correctly asoften as possible
- Linear regression example: algorithm would need to learn values for *m* (or *W*)and *b* (*x* is input, *y* is output)
- Each iteration of process is a training step.

5. **Evaluate the model**

- Uses some metric or combination of metrics to "measure" objective performance of model
- Test the model against previously unseen data

6. **Parameter Tuning**

- This step refers to hyper parameter tuning, which is an "art form" as opposed to ascience
- Tune model parameters for improved performance
- Simple model hyper parameters may include: number of training steps, learning rate,initialization values and distribution, etc.

7. **Make Predictions**

- Using further (test set) data which have, until this point, been withheld from the model (and for which class labels are known), are used to test the model; a better approximation of how the model will perform in the real world.

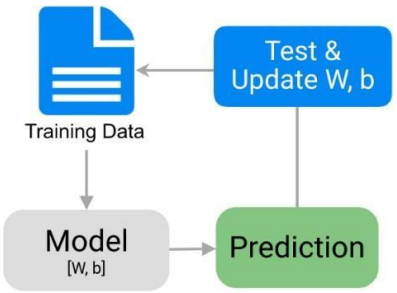

Fig 1.1 Fundamental Steps in Machine Learning

1.5 APPLICATIONS OF MACHINE LEARNING

Machine learning algorithms are used in circumstances where the solution is required to continue improving post-deployment. The dynamic nature of adaptable machine learning solutions is one of the main selling points for its adoption by companies and organizations across verticals.

Machine learning algorithms and solutions are versatile and can be used as a substitute for medium-skilled human labor given the right circumstances. For example, customer service executives in large B2C companies have now been replaced by natural language processing machine learning algorithms known as chatbots. Machine learning algorithms also help to improve user experience and customization for online platforms.

Facebook utilizes recommendation engines for its news feed on both Facebook and Instagram, as well as for its advertising services to find relevant leads. Netflix collects user data and recommends various movies and series based on the preferences of the user. However, as ML continues to be applied in various fields and use-cases, it becomes more important to know the difference between artificial intelligence and machine learning.

1.6 ADVANTAGES OF MACHINE LEARNING

- Easily identifies trends and patterns
- No human intervention needed (automation)
- Handling multi-dimensional and multi-variety data
- Wide Applications

1.7 DISADVANTAGES OF MACHINE LEARNING

- Data Acquisition
- Time and Resources
- Interpretation of Results
- High error-susceptibility

SUMMARY

As a result, we have studied Advantages and Disadvantages of Machine Learning. Also, this blog helps an individual to understand why one needs to choose machine learning.

CHAPTER 2

LITERATURE SURVEY

1. **Cruz JA, Wishart DS. "Applications of machine learning in cancer prediction and prognosis." Cancer Inform. 2017 Feb**

ABSTRACT-

Machine learning is a branch of artificial intelligence that employs a variety of statistical, probabilistic and optimization techniques that allows computers to "learn" from past examples and to detect hard-to-discern patterns from large, noisy or complex data sets. This capability is particularly well-suited to medical applications, especially those that depend on complex proteomic and genomic measurements. As a result, machine learning is frequently used in cancer diagnosis and detection. More recently machine learning has been applied to cancer prognosis and prediction. This latter approach is particularly interesting as it is part of a growing trend towards personalized, predictive medicine. In assembling this review we conducted a broad survey of the different types of machine learning methods being used, the types of data being integrated and the performance of these methods in cancer prediction and prognosis. A number of trends are noted, including a growing dependence on protein biomarkers and microarray data, a strong bias towards applications in prostate and breast cancer, and a heavy reliance on "older" technologies such artificial neural networks (ANNs) instead of more recently developed or more easily interpretable machine learning methods. A number of published studies also appear to lack an appropriate level of validation or testing. Among the better designed and validated studies it is clear that machine learning methods can be used to substantially (15-25%) improve the accuracy of predicting cancer susceptibility, recurrence and mortality. At a more fundamental level, it is also evident that machine learning is also helping to improve our basic understanding of cancer development and progression.

LIMITATIONS -

- It is just a theoretical approach of understanding
- Does not brief on datatypes used for detection

2. "A statistical and ML methods for modelling cancer risk using structured clinical data", JUNE 2018

ABSTRACT-

Advancements are constantly being made in oncology, improving prevention and treatment of cancers. To help reduce the impact and deadliness of cancers, they must be detected early.

Additionally, there is a risk of cancers recurring after potentially curative treatments are performed. Predictive models can be built using historical patient data to model the characteristics of patients that developed cancer or relapsed. These models can then be deployed into clinical settings to determine if new patients are at high risk for cancer development or recurrence. For large-scale predictive models to be built, structured data must be captured for a wide range of diverse patients. This paper explores current methods for building cancer risk models using structured clinical patient data. Trends in statistical and machine learning techniques are explored, and gaps are identified for future research. The field of cancer risk prediction is a high-impact one, and research must continue for these models to be embraced for clinical decision support of both practitioners and patients.

LIMITATIONS-

- Emphasis only on structured clinical data.
- It is a simple generalized study

3. Cruz JA, Wishart DS. Applications of machine learning in cancer prediction and prognosis. Cancer Inform, 2017 Feb

ABSTRACT-

Image classification is a very popular machine learning domain in which deep convolutional neural networks have mainly emerged on such applications. These networks manage to achieve remarkable performance in terms of prediction accuracy but they are considered as black box models since they lack the ability to interpret their inner working mechanism and explain the main reasoning of their predictions.

There is a variety of real world tasks, such as medical applications, in which interpretability and explainablity play a significant role. Making decisions on critical issues such as cancer prediction utilizing black box models in order to achieve high prediction accuracy but without provision for any sort of explanation for its prediction, accuracy cannot be considered as sufficient and ethnically acceptable. Reasoning and explanation is essential in order to trust these models and support such critical predictions. Nevertheless, the definition and the validation of the quality of a prediction model's explanation can be considered in general extremely subjective and unclear. In this work, an accurate and interpretable machine learning framework is proposed, for image classification problems able to make high quality explanations. For this task, it is developed a feature extraction and explanation extraction framework, proposing also three basic general conditions which validate the quality of any model's prediction explanation for any application domain.

The feature extraction framework will extract and create transparent and meaningful high level features for images, while the explanation extraction framework will be responsible for creating good explanations relying on these extracted features and the prediction model's inner function with respect to the proposed conditions. As a case study application, brain tumor magnetic resonance images were utilized for predicting glioma cancer. Our results demonstrate the efficiency of the proposed model since it managed to achieve sufficient prediction accuracy being also interpretable and explainable in simple human terms.

LIMITATIONS-

- An attempt of removing features in the dataset might decrease the overall performance.

4. Pan, L., Liu, G., Lin, F. et al. Machine learning applications for prediction of relapse in childhood acute lymphoblastic leukaemia. Sic Rep 7, 7402 (2017).

ABSTRACT-

The prediction of relapse in childhood acute lymphoblastic leukemia (ALL) is a critical factor for successful treatment and follow-up planning. Our goal was to construct an ALL relapse prediction model based on machine learning algorithms.

Monte Carlo cross-validation nested by 10-fold cross validation was used to rank clinical variables on the randomly split training sets of 336 newly diagnosed ALL children, and a forward feature selection algorithm was employed to find the shortest list of most discriminatory variables. To enable an unbiased estimation of the prediction model to new patients, besides the split test sets of 150 patients, we introduced another independent data set of 84 patients to evaluate the model. The Random Forest model with 14 features achieved a cross-validation accuracy of 0.827 ± 0.031 on one set and an accuracy of 0.798 on the other, with the area under the curve of 0.902 ± 0.027 and 0.904, respectively. The model performed well across different risk-level groups, with the best accuracy of 0.829 in the standard-risk group. To our knowledge, this is the first study to use machine learning models to predict childhood ALL relapse based on medical data from Electronic Medical Record, which will further facilitate stratification treatments

LIMITATIONS-

- Uses random forest, so can use only when tabular data is provided.
- Performance is a concern in random forest.

5. **H. Abedy, F. Ahmed, M. N. Qaisar Bhuiyan, M. Islam, N. Y. Ali and M. Shamsujjoha, "Leukaemia Prediction from Microscopic Images of Human Blood Cell Using HOG Feature Descriptor and Logistic Regression" 2018 16th International Conference on ICT, 2018, pp. 1-6.**

ABSTRACT-

Leukemia is a cancer of blood which originates in bone marrow causing disruption in the production of human blood cells. Earlier detection of leukemia is crucial due to its fatality. Detection of leukemia involves microscopic observation of human blood cells. Application of various image processing methods along with existing machine learning algorithm is a fast and convenient way to detect leukemia.

These methods require extracting features from microscopic images of blood cells and applies machine learning algorithm to train and test a classifier based model which can predict leukemia with an acceptable accuracy. However, scarcity of publicly available image dataset and the inconsistency of the information provided from them make it more challenging while developing a model which can predict leukemia accurately. Besides the size of small datasets and the computational cost, memory evaluation and the required accuracy are also concerning issue. Keeping these drawbacks in mind, we proposed an efficient classifier based model which extract features from image dataset with and classify them accordingly. In this book, we have introduced an edge feature with HOG feature descriptor and Logistic Regression Classifier based model which can detect leukemia with an accuracy of almost 96%.

LIMITATIONS-

- No automatic implementation
- In regression algorithms deep learning methods are not used.

2.2 PROBLEM IDENTIFICATION

Our literature survey helped us deal with challenging unstructured data that is images rather than on simple structured text data. The documentation showed the simple machine learning algorithms made the conclusion to be ambiguous. And further detailed view on certain algorithms such as random forest had left the deployment with less accuracy instead it gave us an idea to use neural networks which has given the highest result precisions along its chronology. The image processing will show the segmentation of the raw input but needs to be taken further for a better clarity of results.

The research flow infers that the above papers have not worked on automatic interface but we wish to master this. Our approach here is to overcome all the problem with specificity of deep learning and neural network over microscopic cell images and provide an end-to-end application for a pathologist. On further interest we would add up image segmentations and scaling to give a vivid picture of what the machine is actually looking into to give a predictable result or stage of Leukaemia currently the victim may have been overcoming with.

2.3 PROJECT OBJECTIVE

Our main objective is to provide the specialist with an application which provides the results with maximum accuracy. Our outlook on the project is to extract a microscopic cell image input from user interface, which would basically be a HTML, CSS interface.

This image would then undergo a set of image processing such as segmentation, edge detection etc. whose batch would be fed to the machine to learn from the main frame of Python code, which would use its major libraries and deep learning neural networks and various ML libraries. Then a derived predicated percentile output would be fed back into the application interface. To summarize the objective to provide a quick access application to the hematologist and the methodology is to use Python libraries and algorithms to attain the objective mentioned.

Through this report, we will show the ways and approach we have taken up to attain the objective, which was help in an organized and comprehensive manner, so that the needed model fulfils the agenda behind our working.

CHAPTER 3

SYSTEM REQUIREMENTS

To be used efficiently, all computer software needs certain hardware components or other software resources to be present on a computer. These prerequisites are known as (computer) system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements: minimum and recommended. With increasing demand for higher processing power and resources in newer versions of software, system requirements tend to increase over time. Industry analysts suggest that this trend plays a bigger part in driving upgrades to existing computer systems than technological advancements.

3.1 NON-FUNCTIONAL REQUIREMENTS

Nonfunctional requirements are the functions offered by the system. It includes time constraints and constraints on the development process and standards. The nonfunctional requirements are as follows:

- **Speed:** The system should process the given input into output within appropriate time.
- **Ease of use:** The software should be user friendly.
  Then the customers canuse easily, so it does not require much training time.
- **Reliability:** The rate of failures should be less then only the system is morereliable
- **Portability**: It should be easy to implement in any system.

3.2 SPECIFIC REQUIREMENTS

The specific requirements are:

- **User Interfaces:** The external users are the clients. All the clients can use this software for sending the request and getting the response back.
- **Software Interfaces:** The Operating Systems can be any version of Windows.
- **Performance Requirements:** The PC's used must be atleast Pentium 4machines so that they can give optimum performance of the product.

3.3 HARDWARE REQUIREMENTS

- Processor : CORE i3
- RAM: 2 GB
- Hard Disk : 20 GB
- Peripherals: Keyboard, Mouse, pen drive or CD ROM
- LAN interconnected systems.

3.4 SOFTWARE REQUIREMENTS

- Python 3.6 or above
- Pycharm IDE
- Keras
- Tensorflow
- Seaborn
- Numpy

3.5 SOFTWARE ANALYSIS

A software requirements definition is an abstract description of the services, which the system should provide, and the constraints under which the system must operate. It should only specify only the external behavior of the system and is not concerned with system design characteristics. It is a solution, in a natural language plus diagrams, of what services the system is expected to provide and the constraints under which it must operate.

3.5.1 Python 3.6 and Above

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophyemphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage-collected. It supportsmultiple programming paradigms, including structured (particularly, procedural), object- oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

Python is a multi-paradigm programming language. Object oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by meta programming and meta objects. Many other paradigms are supported via extensions, including design by contract and logic programming.

Python uses dynamic typing and a combination of reference counting and a cycle- detecting garbage collector for memory management. It also features dynamic name resolution (late binding), which binds method and variable names during program execution.

Important reasons using python

1) Readable and Maintainable Code

2) Multiple Programming Paradigms

3) Compatible with Major Platforms and Systems

4) Robust Standard Library

5) Many Open Source Frameworks and Tools

6) Simplify Complex Software Development

3.5.2 PyCharm IDE

PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as Data Science with Anaconda.Coding assistance and analysis, with code completion, syntax and error highlighting, linter integration, and quick fixes.

PyCharm features:

1) Coding assistance and analysis, with code completion, syntax and errorhighlighting, linter integration, and quick fixed.

2) Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages.

3) Python refactoring: includes rename, extract method, introduce variable, introduce constant, and pull up, push down and others.

4) Google App Engine Python development.

3.5.3 Keras

Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. Keras contains numerous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code. The code is hosted on GitHub, and community support forums include the GitHub issues page, and a Slack channel.

In addition to standard neural networks, Keras has support for convolutional and recurrent neural networks. It supports other common utility layers like dropout, batch normalization, and pooling.

To summarise, Keras layer requires below minimum details to create a complete layer.

- Shape of the input data
- Number of neurons / units in the layer
- Initializers
- Regularizers
- Constraints
- Activations
- Kernel initializer represent initializer to be used.
- Kernel regularizer represent regularizer to be used. None is set as value.
- Kernel constraint represent constraint to be used. MaxNorm function is set as value.
- Activation represent activation to be used. ReLU function is set as value.

3.5.4 Numpy

NumPy is a python library used for working with arrays.
It also has functions for working in domain of linear algebra, fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely. NumPy stands for Numerical Python.

In Python we have lists that serve the purpose of arrays, but they are slow to process. NumPy aims to provide an array object that is up to 50x faster that traditional Python lists. The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy.Arrays are very frequently used in data science, where speed and resources are very important.

3.5.5 Sklearn

Scikit-learn (Sklearn) is the most useful and robust library for machinelearning in Python. It provides a selection of efficient tools for machine learning andstatistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib. It was originally called ***scikits.learn*** and was initially developed by David Cournapeau as a Google summerof code project in 2007. Rather than focusing on loading, manipulating and summarising data, Scikit-learn library is focused on modeling the data.

3.5.6 TensorFlow

TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML-powered applications.

TensorFlow was originally developed by researchers and engineers working on the Google Brain team. The system is general enough to be applicable in a wide variety of other domains, as well.

TensorFlow provides stable Python and C++ APIs, as well as non-guaranteed backward compatible API for other languages

3.5.7 MatplotLib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged.SciPy makes use of Matplotlib.

Pyplot is a Matplotlib module which provides a MATLAB-like interface.Matplotlib is designed to be as usable as MATLAB, with the ability to use Python, and the advantage of being free and open-source.

Seaborn: provides an API on top of Matplotlib that offers sane choices for plot style and color defaults, defines simple high-level functions for common statistical plot types, and integrates with the functionality provided by Pandas

3.5.8 Pandas

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. DataFrame object for data manipulation with integrated indexing.

Tools for reading and writing data between in-memory data structures and different file formats.

Features:

- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Label-based slicing, fancy indexing, and subsetting of large data sets.
- Data structure column insertion and deletion.
- Group by engine allowing split-apply-combine operations on data sets.
- Data set merging and joining.

- Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional data structure.
- Time series-functionality: Date range generation[6] and frequency conversions, moving window statistics, moving window linear regressions, date shifting and lagging.
- Provides data filtration.

3.5.9 Seaborn

Seaborn is an open source, BSD-licensed Python library providing high level API for visualizing the data using Python programming language. Seaborn is built on top of Python's core visualization library Matplotlib. It is meant to serve as a complement, and not a replacement. However, Seaborn comes with some very important features. Let us see a few of them here. The features help in -

- Built in themes for styling matplotlib graphics
- Visualizing univariate and bivariate data
- Fitting in and visualizing linear regression models
- Plotting statistical time series data
- Seaborn works well with NumPy and Pandas data structures
- It comes with built in themes for styling Matplotlib graphics

SUMMARY

The third chapter is about the system analysis. It gives the explanation about functional, non-functional, hardware and software requirements. It also emphasis on analysis on hardware and software requirements.

CHAPTER 4

SYSTEM ANALYSIS

4.1 MODULES CLASSIFIED

This project has mainly 4 modules. They are as follows:

- Selecting Data and pre-processing
- Applying CNN
- Extracting the Features
- Prediction

4.1.1 Selecting Data and Pre-processing

A dataset plays a very important role in training the model in order to make prediction for leukemia. For training, tuning, model selection to testing datasets are majorly used.

In this research work, we can find data in the form of a folder of two data files. One of these files has a number of sample images of blood cells infected by leukemia and the second file has images of those cells, which are healthy. This data further divides into training and testing datasets.

The training dataset trains the model to identify the class to which the input image belongs. The testing dataset, which is comparatively smaller to that of training dataset, uses the trained model in order to test for the correctness of the training results given by the model.

Individual images have 640*480 pixels approximately and are in .jpeg format.

4.1.2 Applying CNN

Convolution is a function derived from two functions by which we determine how the shape of one modifies due to the other.

$$(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau$$

The above is a mathematical part of it where f and g are two functions, which then integrates as shown where τ is the integration function. However, the convolution layer works as follows.

Filter or a Kernel is an m*m matrix used to detect features in this step which has random values of integers. At first, each pixel of the input image is assigned with an integer (random). The image has more than 600 such pixels which are in the form of huge number of matrices aligned one after the other.

The filter which is the feature detector is a comparatively smaller matrix starts from the top left corner of the image and strides over each m*m matrix of its size across the entire image. Usage wide strides for the data in this study

If all the random values of the filter match the values of the pixels matrix then the model puts a '1' into the feature map but if they do not match then it puts a '0'. A feature map is a group of matrices as well which stores 0's and 1's as explained . The creation of a feature map reduces the image pixels to its essential features and CNN develops multiple such maps, which refers to as convolution layers. This map also adjusts to the image, sharpens it and detects edges. There are small sections of input that come together to form a complete feature map called neurons. Neurons in general are mathematical functions that takes input, multiplies them with certain weights and gives an output.

4.1.3 Extracting the Features

The model is initially fit on a training dataset, which is a set of examples used to fit the parameters (e.g. weights of connections between neurons in artificial neural networks) of the model. The mode is trained on the training dataset using a supervised learning method, for example using optimization methods such as gradient descent or stochastic gradient descent. In practice, the training dataset often consists of pairs of an input vector (or scalar) and the corresponding output vector (or scalar), where the answer key is commonly denoted as the target (or label). The current model is run with the training dataset and produces a result, which is then compared with the target, for each input vector in the training dataset.

Then the test dataset will be selected. The test dataset is a dataset that is independent of the training dataset, but that follows the same probability distribution as the training dataset.

If a model fit to the training dataset also fits the test dataset well. A better fitting of the trainingdataset as opposed to the test dataset usually points to overfitting.

Then selecting the microscopic images which need to be checked from that we can get the detailed analysis and results.

4.1.4 Prediction

The prediction of the result is done using the predict function, where the true positive predicts to be the cancerous matching datasets if it is otherwise it is false positive. The cluster graph is found according to similar set of images in the datasets. The genetic algorithm is predicted by passing a function over the datasets given. A pop up message is given on the console following which the gray scaling of the image needs to be given in the application.

Once that is done the console needs to be provided with the all the outputs so that the hematologist can easily detect the faults in the cells as shown in the microscopic images.

To accomplish this the Django API plays a very important role in putting the backend and the frontend together.

SUMMARY

This chapter gives the description about system analysis. It includes modules and algorithms used in the project.

CHAPTER 5

SYSTEM DESIGN

5.1 PROJECT ARCHITECTURE

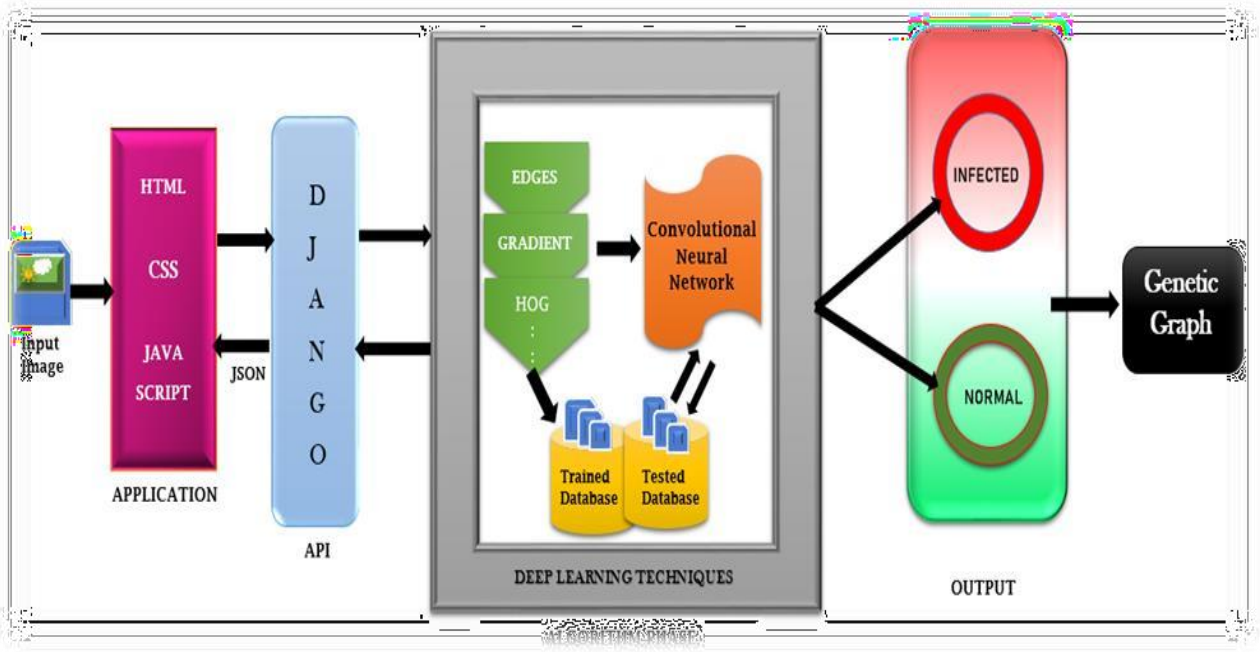

Fig 5.1.1 System Architecture

A microscopic image is given as input in the front-end application. This application is done using the scripting languages like HTML,CSS and JavaScript. HTML basically works like the structure of the webpage. To which styling can be done using the style sheets provided by CSS, where the same style sheets can be reused in building a user interface. The image is taken from the front end and passed on to the Algorithm phase where the user input image is checked with the trained and tested datasets for the matching of features. To extract these features we have do the segmentation, edge detection and HOG methods to get the prominent features to compare with the datasets as mentioned now. Once this algorithm is run an inference is derived from which the input image indication is predicted to be normal or infected as shown in the figure. Along with this a genetic algorithm is also made to run which determines the graph of the infection present or absent.

Now this input and output needs a connecting interface for which a framework is used , here Django is used which takes input from the user interface and processes the algorithm and deploys the result back to the same interface by using the GET and POST methods which is most popularly used over years. Based on this we have divided the project job into modules as explained further.

5.2 MODULE 1- DATA MODELLING WITH CNN ALGORITHM

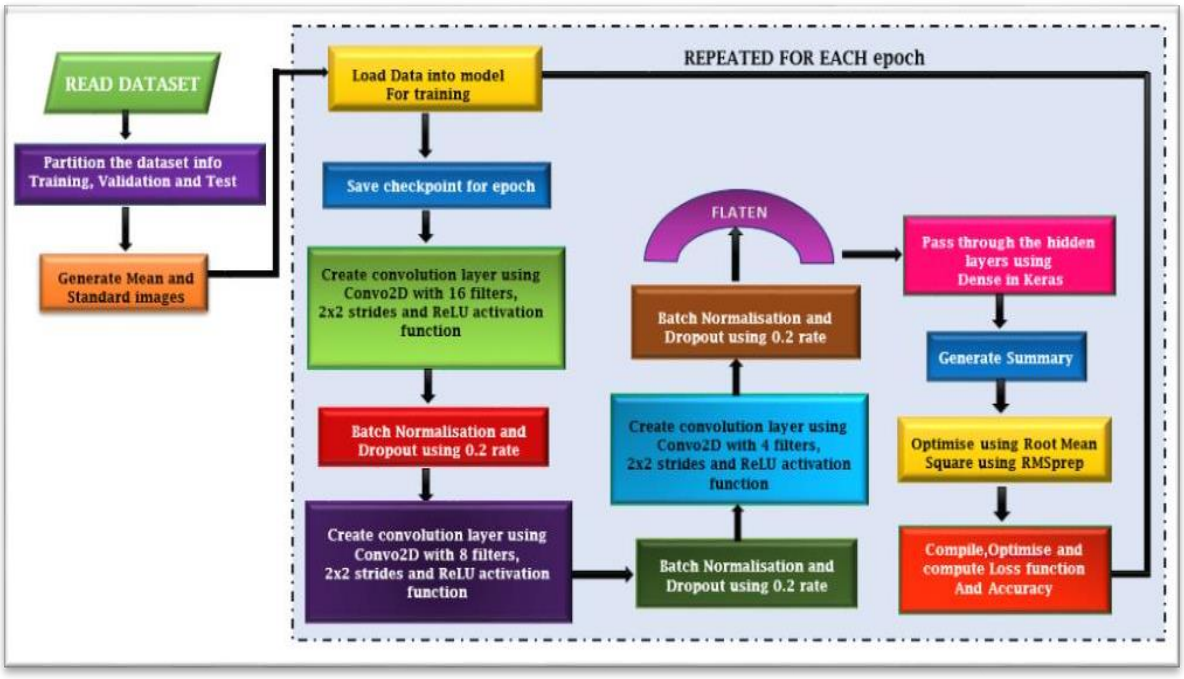

Fig 5.2.1 Data Modeling using CNN algorithm

The dataset is read and is partitioned into separate entities namely training, validation and test datasets. Then a function runs in order to find the standard images and generate a mean. This standard images are with utmost clarity and of same measures and optimal to be loaded to the model next. The data once on loading to the model for training fixes a epoch which is the iteration end point, i.e. a checkpoint .So more the checkpoints more is the accuracy of the model.

Once the checkpoints are sent it is undergoing the convolution part with filtering techniques. Initially it takes 16 filters and 2*2 stride , which can differ from every model. Then on repeatedly running this steps the filters gets optimized to given a smallest matrix with highest count data. This matrix is brought down to a 1D array i.e. flattening.

After flattening the digits are verified with initial vectors in the hidden layers of the Keras library and more like them generating a summary. The mathematical expression is generated using the Root Mean Square which determines ,compiles and generates accuracy. This process is taken up for each epoch value and stops once the image pixel rate is fulfilled.

To represent this on a closer we have divided the module 1 into 3 parts which we will discuss here now.

5.2.1 Module 1-Part A

Download the Data set and load it on to model for further processing. Datasets are downloaded from Kaggle.com

Read the label names and use the libraries and amplify for vivid images. The figures shown below are samples of the infected and non-infected blood smear images.

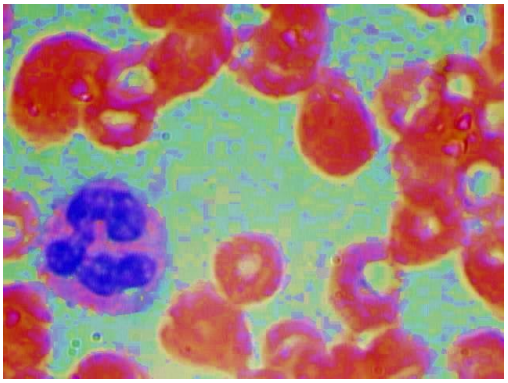

Fig 5.2.1.1 Normal Blood cell image sample

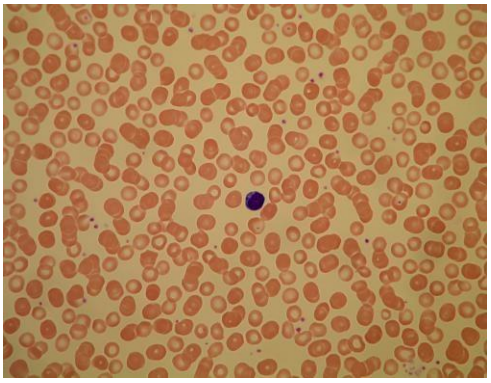

Fig 5.2.1.1 Cancerous Blood cell image sample

5.2.2 Module 1-Part B

CNN, namely means one which has convolutional layers. CNN is widely used for image processing and analysis.CNN is different from any MLP (Multilayer perceptron) in the fact that it has hidden layers called convolutional layers. This also has non convolutional layers but convolutional layers form its basics.

Convolution Layer receives input from a previous layer, performs operation and transfers it to a next layer or output layer which is called the process of convoluting.

CNN and its convolutional layers detect patterns and images with the help of filters. Number of filters are specified for each layer in CNN. This is represented the image below.

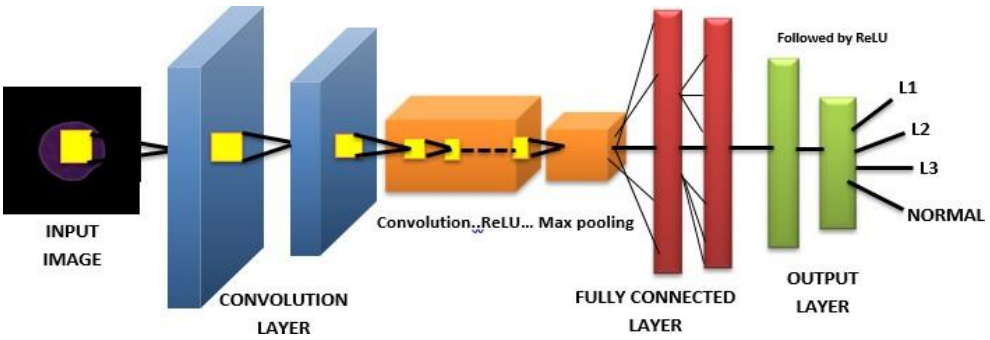

Fig 5.2.2.1 Convolution Network with its layers

Filters detect elements in an image like edges, shapes, objects, textures etc. Example of filters are edge detectors & filters can be specific for one object.

That means complex filters can also detect objects like a full dog, cat, eye etc. in deeper layers.

Filters are technically just rendered as a relatively short matrix for which we decide the number of rows and columns it should have.The values inside the matrix are initialized with random numbers.

Suppose a 3*3 matrix is our filter as shown here:

-1	-1	-1
1	1	1
0	0	0

Fig 5.2.2.2 Filter 3*3 representation

Now this filter slides over every pixel of the image until it traverses every 3*3 block pixels group of the entire image. This sliding is called convolving.

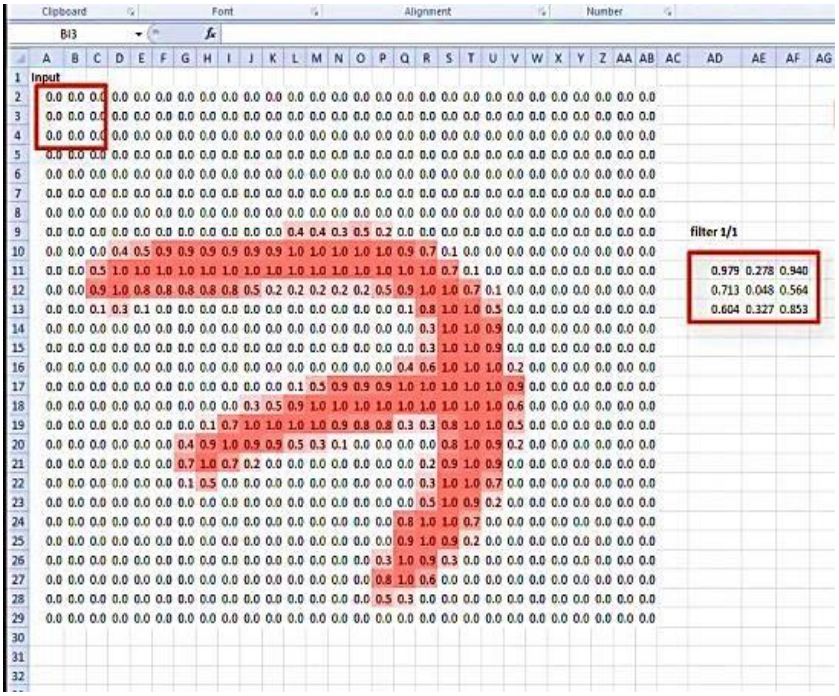

Fig 5.2.2.3 Input matrix sliding over pixels

Filter matrix slides to find the dot product with each block in the image hence to deduce an answer of 9 blocks into one block making a single pixel for the next layer.This loops for all given values until pattern is recognised as shown the figure 5.2.2.4.

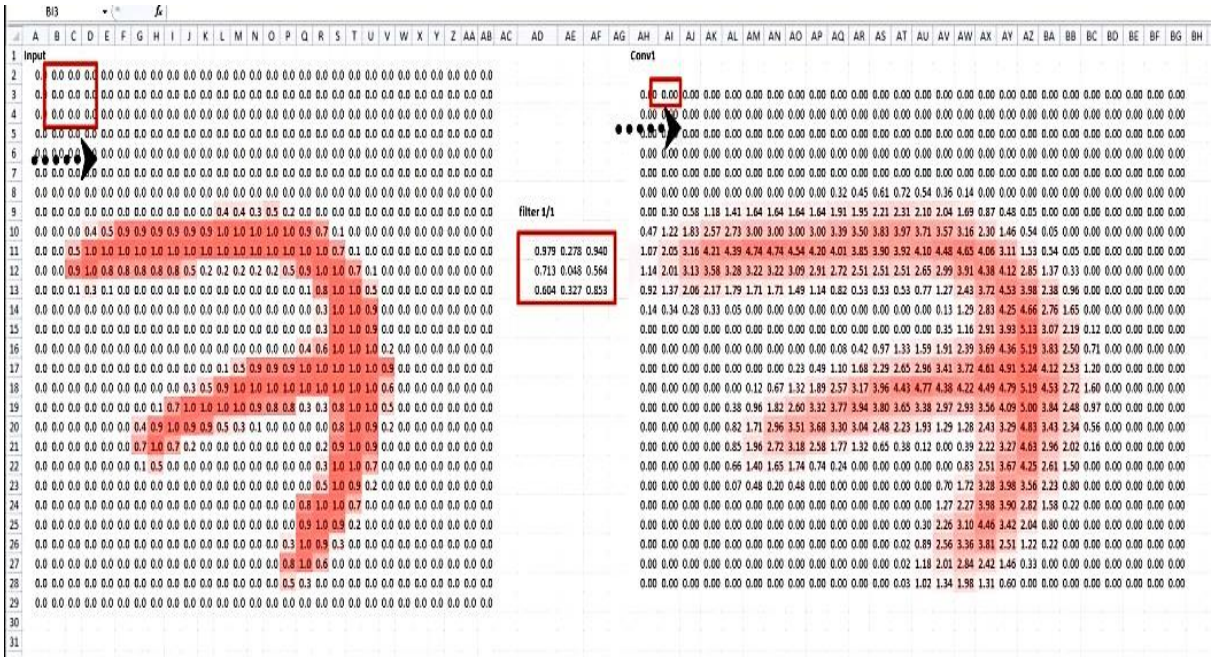

Fig 5.2.2.4 Reduction of the matrix to singleton

Some Internal Functions :

ReLU (Rectified Linear Activation Function):

This is a supplementary step to convolution layer and serves the purpose of increasing non-linearity of images.

Non-linear features can be the transition between pixels, the colors etc. that form negative values, in the image and ReLU returns only non-negative values. ReLU is an activation function used in order to return '0' for negative input values and return same values if input is positive as shown.

F (max) = max (0, x)

It is a linear function that checks the input for negative values, if found makes them 0 and inputs them else it keeps them constant.

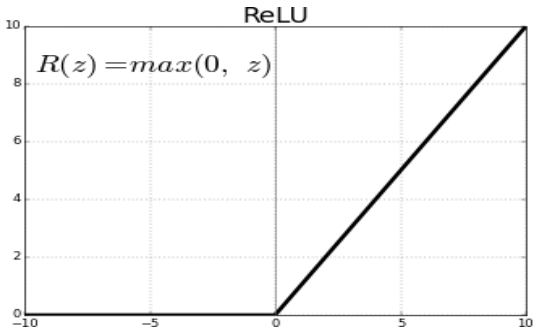

Fig 5.2.2.5 Graph representing ReLU function

Max pooling layer:

The network has to acquire a property called "Spatial Variance". This makes it capable of detecting objects in the image without ambiguity, by the differences in image textures, distances, angles and distortions.

We use max pooling in order to create a pooled feature map, which is a matrix that contains only maximum values of the values in the individual matrices (feature map matrices). Therefore, a^2 of the matrix is reduced to a by picking one value from individual matrices as shown in figure.

This step reduces 75% of the original image information so that the network can proceed towards the result efficiently leading to providing the property of Spatial Variance.

This step solves another major issue of image processing that is "Over fitting" which reduces the complex model for idiosyncrasies.

This is a function that reduces filters by choosing max values inside each matrix.

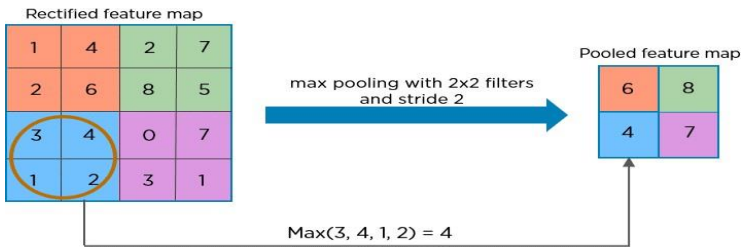

Fig 5.2.2.6 Featured to Pooled Mapping

Flattening Mechanism:

After the completion of the previous steps, it makes a pooled feature map. This step, like its name flattens the pooled feature map into a single column, as shown in the figure. This procedure is undergone to feed the vector of data into the neural network for further processing.

This is a function that converts pooled feature map to a single column.

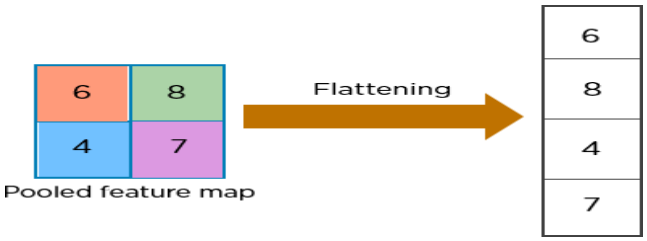

Fig 5.2.2.7 Conversion of Pooled matrix to Flat array.

After applying CNN to different layers pattern recognition can be seen in the following ways. The figure below shows the edge detection in 4 ways using 4 filters.

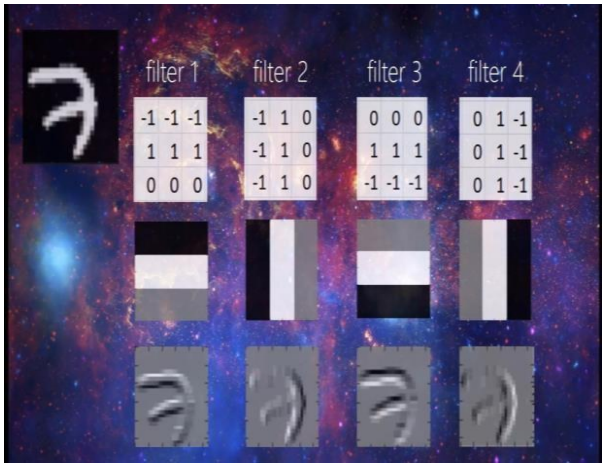

Fig 5.2.2.8 Representation Image filtering

As the number of layers increase the differences are made out as follows , the image becomes more optimised and prominent.

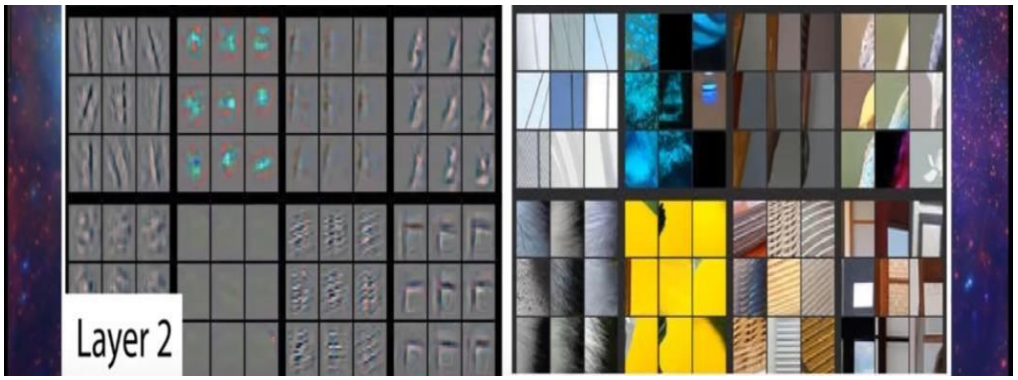

Fig 5.2.2.9 Increased optimisation with increased layers of CNN.

5.2.3 Module 1-Part C

Now, that we have know the working of CNN algorithm, the blood cell microscopic images undergo the various layers of the neural networks as explained in Part B of Module 1.

Once the pattern recognition is activated , the trained dataset if ready for the next step. Using the python libraries we define the predict function.

If the predicted value matches with the normal blood cell results we conclude the result to be normal, if any changes are to be seen from the standardised value then the outcome turns out to be infected (Leukaemia). This data brings out the genetic algorithm graph and cluster analysis of the above model.

Now the model is ready to get an user interface and get deployed.

5.3 MODULE 2- USER INTERFACE DESIGN

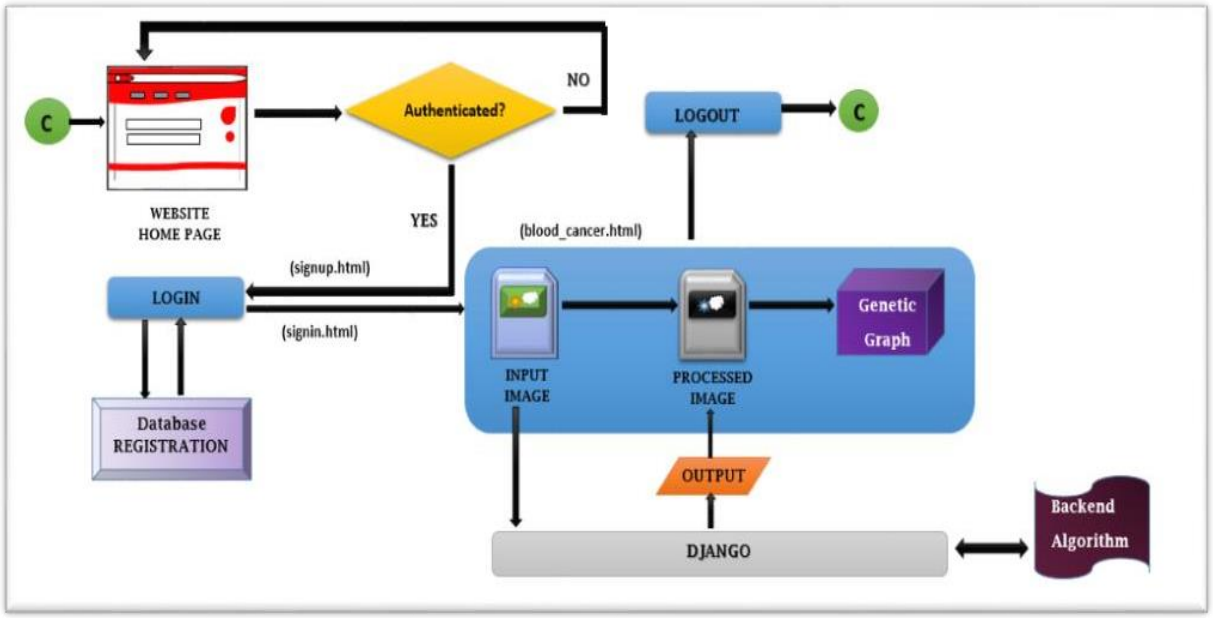

Fig 5.3.1 User Interface Design

The model generates a URL, clicking on which it redirects to the HTML page.A new user can visit the website, a new user can get his details authenticated .

On successful authentication the data gets stored in the database and moves from sign-up towards the sign-in part.

A regular user and authenticated can access the blood-cancer detection main page.

In this webpage the user is asked to enter the blood cell image is given as an input which triggers the Django API and directs to the CNN algorithm to carry out the processes discussed in Module 1.

The interface is designed with bright background and on click result retrieving buttons like cluster analysis, Genetic algorithm which redirects to the hyperlinked location. Once the user has done with his work he can logout. There is also an option for the cient to navigate between various tabs across the page.

The output i.e. the processed image and the genetic algorithm retrieves back from the Backend algorithm via the support of Django API.

5.4 MODULE 3 –DEPLOYMENT PHASE

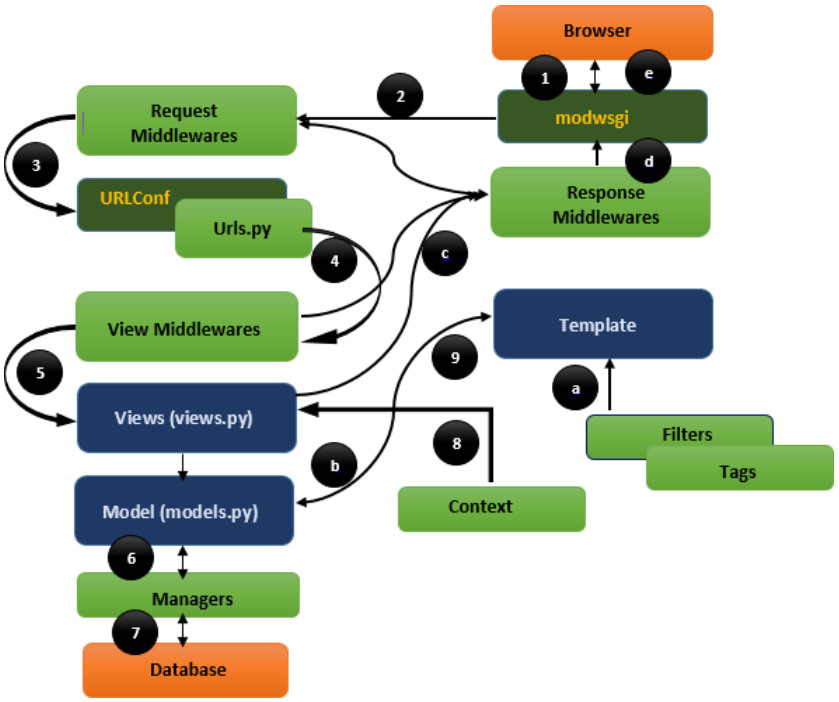

Fig 5.4.1 Deployment using Django API

It is indicated in the form of step numbers and characters matching the same with the figure as explained below, i.e. the Working Of Django API in connecting two ends.

1. User requests a page
2. Request reaches Request Middle wares, which could manipulate or answer the request.
3. The URLConffinds the related View using urls.py
4. View Middlewares are called , which could manipulate or answer the request
5. The view function is invoked
6. The view could optionally access data through models
7. All model-to-DB interactions are done via a manager
8. Views could use a special context if needed
9. The context is passed to the Template for rendering

a. Template uses Filters and Tags to render the output

b. Output is returned to the view

c. HTTPResponse is sent to the Response Middlewares

d. The response middleware can enrich the response or return a completely new response

e. The response is sent to the user's browser.

CHAPTER 6

IMPLEMENTATION

Once the system has been designed, the next step is to convert the designed one in to actual code, so as to satisfy the user requirements as excepted. If the system is approved to be error free it can be implemented. The implemented software should be maintained for prolonged running of the software.

6.1 UPLOADING OF DATASETS

```
def blood_cancer(request):
    if request.method == 'POST' and 'bttwo' in request.POST:
        # import matplotlib.pyplot as plt
        data = pd.read_csv('labels.csv')
        mylist = data.iloc[:, 2].values
        a_ary = []

        for li in mylist:
            if li not in a_ary:
                a_ary.append(li)
        print(a_ary)
        print(len(a_ary))
        X, y = make_blobs(n_samples=len(mylist), centers=len(a_ary),
                          random_state=0, cluster_std=0.60)
    plt.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap='autumn')
        plt.title("Data set Analysis-Cluster Analysis")
        plt.savefig('./assets/static/img/ClusterAnalysis.jpg')
    elif request.method == 'POST' and request.FILES['myfile'] and 'btnone' in
request.POST:
        myfile = request.FILES['myfile']
```

6.2 CONVERSION OF IMAGE TO GREYSCALE

```
# Convert image to grayscale
    img_gs = cv2.imread(save_path, cv2.IMREAD_GRAYSCALE)
    cv2.imwrite('gs.jpg', img_gs)
    im = Image.open(r"gs.jpg")
    im.show()
    img1 = cv2.imread(save_path, 0)
    img2 = cv2.imread('gs.jpg', 0)
```

6.3 FILTERING OF IMAGES

```
    # --- take the absolute difference of the images ---
    res = cv2.absdiff(img1, img2)
    # --- convert the result to integer type ---
    res = res.astype(np.uint8)
    # --- find percentage difference based on number of pixels that are not zero ---
    percentage = (np.count_nonzero(res) * 100) / res.size
    print(percentage)
    # Apply canny edge detector algorithm on the image to find edges
    edges = cv2.Canny(img_gs, 100, 300)

    # Plot the original image against the edges
    plt.subplot(121), plt.imshow(img_gs)
    plt.title('Original Image')
    plt.subplot(122), plt.imshow(edges)
    plt.title('CNN (convolutional neural network)')
    plt.savefig('./assets/static/img/predbc-cnn.jpg')
    # return render(request, 'graphical_page.html')
    # Display the two images
    plt.show()
    sleep(0.5)
```

6.4 APPLYING OF CONVOLUTION NEURAL NETWORK LAYERS

```
 # Initialising the CNN
classifier = Sequential()
# Convolution
classifier.add(Conv2D(32, (3, 3), input_shape=(64, 64, 3), activation='relu'))
# Pooling
classifier.add(MaxPooling2D(pool_size=(2, 2)))
# Adding a second convolutional layer
classifier.add(Conv2D(32, (3, 3), activation='relu'))
classifier.add(MaxPooling2D(pool_size=(2, 2)))
# Flattening
classifier.add(Flatten())
# Full connection
classifier.add(Dense(units=128, activation='relu'))
classifier.add(Dense(units=1, activation='sigmoid'))
# Compiling the CNN
classifier.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
# Fitting the CNN to the images
train_datagen = ImageDataGenerator(rescale=1. / 255,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=True)
test_datagen = ImageDataGenerator(rescale=1. / 255)
```

6.5 DETECTING OF GRAPH

```
def detectgraph(request):
 if request.method == "POST":
    print(2)
    data = pd.read_csv('labels.csv')
    mylist = data.iloc[:, 2].values
    a_ary=[]
    for li in mylist:
       if li not in a_ary:
```

```
            a_ary.append(li)
        print(a_ary)
        print(len(a_ary))
        X, y = make_blobs(n_samples=len(mylist), centers=len(a_ary),
                    random_state=0, cluster_std=0.60)
        plt.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap='autumn')
        fig, ax = plt.subplots()
        fig.savefig('D:\2021 8th project\blood cancer\my_plot.png')
        plt.title("Data set Analysis-SVM Algorithm")
        plt.savefig('./assets/templates/SVM Algorithm')
        plt.show()
        print(len(mylist))
    return render(request, 'cluster_analysis.html')
```

6.6 FUNCTION TO FIND GRAPH ANALYSIS

```
def graph_analysis(request):
    if request.method == "POST":
        print(2)
        data = pd.read_csv('labels.csv')
        mylist = data.iloc[:, 2].values
        a_ary=[]
        for li in mylist:
            if li not in a_ary:
                a_ary.append(li)
        print(a_ary)
        print(len(a_ary))
        X, y = make_blobs(n_samples=len(mylist), centers=len(a_ary),
     random_state=0, cluster_std=0.60)
        plt.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap='autumn') fig, ax = plt.subplots()
        fig.savefig('D:\2021 8th project\blood cancer\my_plot.png')
            plt.savefig('./assets/templates/SVM Algorithm')
        plt.show()
    return render(request, 'graphical_page.html')
```

6.7 DETERMINING THE RESULT PERCENTILE

```
percentage = (np.count_nonzero(res) * 100) / res.size
    print(percentage)
    if percentage >= 10:
      value = "S1 Level Blood Cancer"
    else:
      value = "Not in S1 Level Blood Cancer"
    value1 = 'S1 Level Blood Cancer'
    fuz_ratio = fuzz.ratio(value, value1)
    print(fuz_ratio)
    if fuz_ratio != 100:
      result = False
    else:
      result = True
    print(percentage)
    win32api.MessageBox(0, 'Blood Cancer Detection' + ':' + str(result), 'CNN (convolutional
neural network) Result')
    f=1
```

6.8 DETERMINING THE GENETIC ALGORITHM GRAPH

```
plt.savefig('./assets/static/img/predbc-cnn.png')
    sleep(0.5)
    percentage = (np.count_nonzero(res) * 100) / res.size
    print(percentage)

    def f(X):
      return np.sum(X)
    varbound = np.array([[0.5, 1.5], [1, 100], [0, 1]])
    vartype = np.array([['real'], ['int'], ['int']])
    model = ga(function=f, dimension=3, variable_type_mixed=vartype,
     variable_boundaries=varbound)
    model.run()
   return render(request, 'blood_cancer.html')
```

6.9 RESPONSE TO HTML INDEX PAGE

```
def index(request):
    if request.method=="POST":
        usid=request.POST.get('username')
        pswd = request.POST.get('password')
        try:
            check = UserModel.objects.get(username=usid,password=pswd)
            print(check)
            request.session['userid']=check.id
            return redirect('blood_cancer')
        except:
            pass
    return render(request,'signin.html')
```

6.10 RESPONSE TO HTML SIGN-UP PAGE

```
def signup(request):
    if request.method=='POST':
        try:
            resp_obj=UserModel.objects.create(username=request.POST.get('username'),
                                password=request.POST.get('password'),
                                email=request.POST.get('email'),
                                mobilenumber=request.POST.get('empcode'),
                                fullname=request.POST.get('fullname'),
                                designation=request.POST.get('designation'),
                                )
            id=resp_obj.id
            print(id)
            return redirect('index')
        except:
            pass
    return render(request,'signup.html')
```

6.11 DJANGO CONNECT TO THE MODELS

```
from django.db import models
# Create your models here.
class UserModel(models.Model):
    username=models.CharField(max_length=256)
    password=models.CharField(max_length=11)
    email=models.EmailField()
    mobilenumber = models.BigIntegerField()
    fullname = models.CharField(max_length=256)
    designation = models.CharField(max_length=256)
```

6.12 DJANGO CONNECT TO THE APP-CONFIG

```
from django.apps import AppConfig


class BloodCancerConfig(AppConfig):
    default_auto_field = 'django.db.models.BigAutoField'
    name = 'blood_cancer'
```

6.13 DJANGO CONNECT TO REQUEST PAGE

```
from django.contrib import admin
from django.urls import path
from blood_cancer import views as bloodcancer_view
from django.conf.urls import url
urlpatterns = [
    path('admin/', admin.site.urls),
    url('^$',bloodcancer_view.index, name="index"),
    path('signin',bloodcancer_view.index,name="index"),
    path('signup',bloodcancer_view.signup,name="signup"),
    path('graph_analysis',bloodcancer_view.graph_analysis,name='graph_analysis'),
    path('blood_cancer',bloodcancer_view.blood_cancer,name='blood_cancer'),
    path('detectgraph',bloodcancer_view.detectgraph,name='detectgraph')
```

CHAPTER 7

SYSTEM TESTING

7.1 SOFTWARE TESTING INTRODUCTION

Software testing is a process used to help identify the correctness, completenessand quality of developed computer software. Software testing is the process used to measure the quality of developed software. Testing is the process of executing a program with the intent of finding errors. Software testing is often referred to as verification &validation.

7.2 EXPLANATION FOR SDLC

The software development life cycle (SDLC) is a conceptual model used in project management that describes the stages involved in an information system development project, from an initial feasibility study through maintenance of the completed application.

7.3 PHASES OF SOFTWARE TESTING

* Requirement Analysis * Design * Testing * Maintenance

7.3.1 Requirement Analysis

The requirements of a desired software product are extracted. Based the business scenario the SRS (Software Requirement Specification) document is prepared in this phase.

7.3.2 Design

Plans are laid out concerning the physical construction, hardware, operating systems, programming, communications, and security issues for the software. Design phase is concerned with making sure the software system will meet the requirements of the product.

7.3.3 Testing

Testing is evaluating the software to check for the user requirements.Here the software is evaluated with intent of finding defects.

7.3.4 Maintenance

Once the new system is up and running for a while, it should be exhaustively evaluated. Maintenance must be kept up rigorously at all times. Users of the system should be kept up- to-date concerning the latest modifications and procedures.

7.4 TYPES OF SDLC MODELS

7.4.1 Water fall model

It will be executing one by one of the SDLC process. The design Starts after completing the requirements analysis coding begins after design. It is a traditional model It is a sequential design process, often used in SDLC, in which the progress is seen as flowing steadily downwards (like a waterfall), through the different phases.

7.4.2 Prototype model

Developed from the sample after getting good feedback from the customer. This is the Valuable mechanism for gaining better understanding of the customer needs

7.4.3 Rapid application development model (RAD):

This mechanism will develop from already existing one. If The New requirement is matching in already existing requirement, will developfrom that**.**

7.4.4 Spiral model

This mechanism is update the application version by version. All the SDLC process will update version by version**.**

7.5 EXPLANATION ON STLC

Testing itself has manyphases i.e. is called as STLC. STLC is part of SDLC

- Test Plan
- Test Development
- Test Execution
- Analyse Results
- Defect Tracking
- Summaries Report

7.6 TYPES OF TESTING

- White-Box Testing
- Black-Box Testing
- Grey-Box Testing

7.6.1 White Box Testing

White box testing as the name suggests gives the internal view of the software. This type of testing is also known as structural testing or glassbox testing as well, as the interest lies in what lies inside the box.

7.6.2 Black Box Testing

It is also called as behavioral testing. It focuses on the functional requirements of the software. Testing either functional or non-functional without reference to the internal structure of the component or system is called black box testing.

7.6.3 Grey Box Testing

Grey box testing is the combination n of black box and white box testing. Intention of this testing is to find out defects related to bad design or bad implementation of the system.

7.7 LEVELS OF TESTING USED IN THE PROJECT

7.7.1 Unit Testing

Initialization testing is the first level of dynamic testing and is first the responsibility of developers and then that of the test engineers. Unit testing is performed after the expected test results are met or differences areexplainable/acceptable**.**

All module which make application are tested. Integration testing is to make sure that the interaction of two or more components produces results that satisfy functional requirement**.**

7.7.2 System Testing

To test the complete system in terms of functionality and non- functionality. It is black box testing, performed by the Test Team, and at the start of the system testing the complete system is configured in a controlled environment.

7.7.3 Functional Testing

The outgoing links from all the pages from specific domain under test.Test all internal links. Test links jumping on the same pages. Check for the default values of fields. Wrong inputs to the fields in the forms.

Valid Input: identified classes of valid input must be accepted.

Invalid Input: identified classes of invalid input must be rejected.

Functions: identified functions must be exercised.

Output: identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

7.8 TEST CASE REPRESENTATION

Sl. No.	Test case Description	Expected Result	Actual Result	Remarks
1	Pre-processing the Datasets.	The Model should pre-process the datasets, convert .mat files into .jpg and segregate the two classes of Blood Cells.	The Model is pre-process the datasets, convert .mat files into .jpg and segregate the two classes of Blood Cells.	**Pass**
2	Training the model by Train and Test Datasets.	The Model should Train and Test the Datasets and improve the Accuracy.	The Model is Training and Testing the Datasets and is improving the Accuracy.	**Pass**
3	Grey-Scaling of Images	The model should take input image and provide the grey scaled image in the console.	The model is taking input image and provide the grey scaled image in the console.	**Pass**

4	Inserting Normal Blood cell for check	On processing the console should give the output as FALSE for cancer.	On processing the console is giving the output as FALSE for cancer.	**Pass**
5	Inserting Cancerous Blood cell for check	On processing the console should give the output as TRUE for cancer.	On processing the console is giving the output as TRUE for cancer.	**Pass**
6	Convolution layer Image	On inserting image the console should give a convoluted image for the given input.	On inserting image the console is giving a convoluted image for the given input.	**Pass**
7	Cluster Analysis Graph	On clicking the Cluster Analysis Button the graph must appear on the user interface.	On clicking the Cluster Analysis Button the graph is appearing on the user interface.	**Pass**
8	Genetic Algorithm Graph	At the end of the model processing a genetic algorithm graph must be generated for each input image.	At the end of the model processing a genetic algorithm graph is generated for each input image.	**Pass**

CHAPTER 8

RESULTS

Initially after classification into two classes, cluster analysis plots are as shown in figure. according to the probability percentage of infection in the dataset.

Next, the percentage above gets to give the prediction in terms of true or false. If more than, 50% of the features of the uploaded image match to the infected class then the prediction shows to be true for occurrence of leukemia or false otherwise.

Model accuracy, which is the measurement, used to determine how well the model is able to identify the relationships and patterns between variables in the dataset is ultimately computes the following

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

Here TP and TN refers to true positive and true negative respectively similarly, FP and FN refers to false positive and false negative.

The accuracy of the CNN model in this study approximates to 97.75%, which is prominently higher than major Neural networks as shown in figure. CNN hence compared to Artificial Neural Network (ANN) and Recurrent Neural Network (RNN), finds to be the most accurate and effective for image processing in this research work, which includes 3D microscopic images of blood cells.

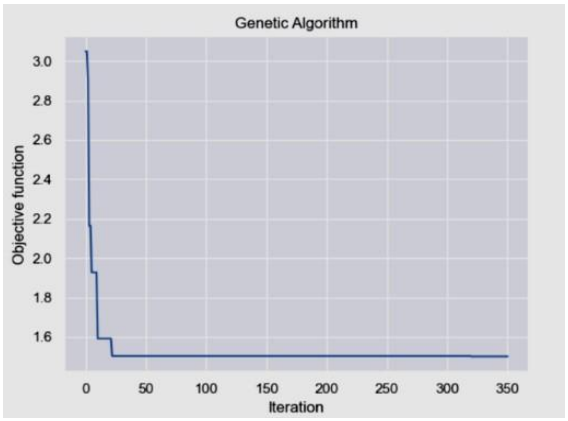

Fig 8.1 Genetic Algorithm Graph

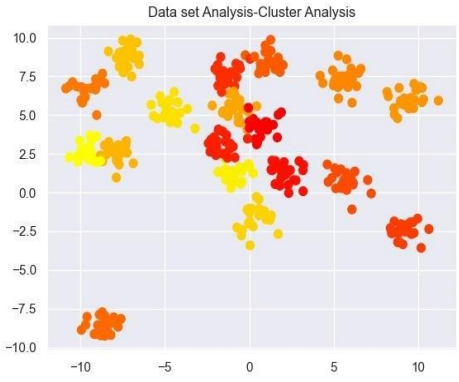

Fig 8.2 Cluster Analysis Graph

8.2 SNAPSHOTS

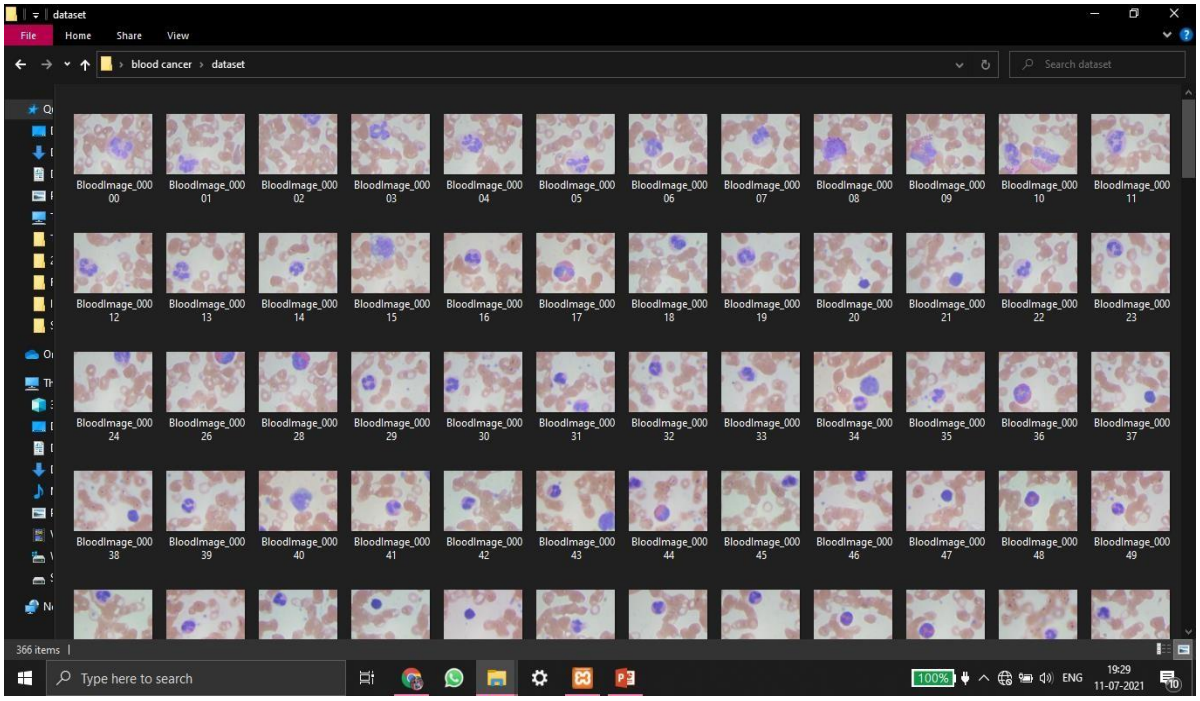

Fig 8.2.1 Normal Cells Dataset

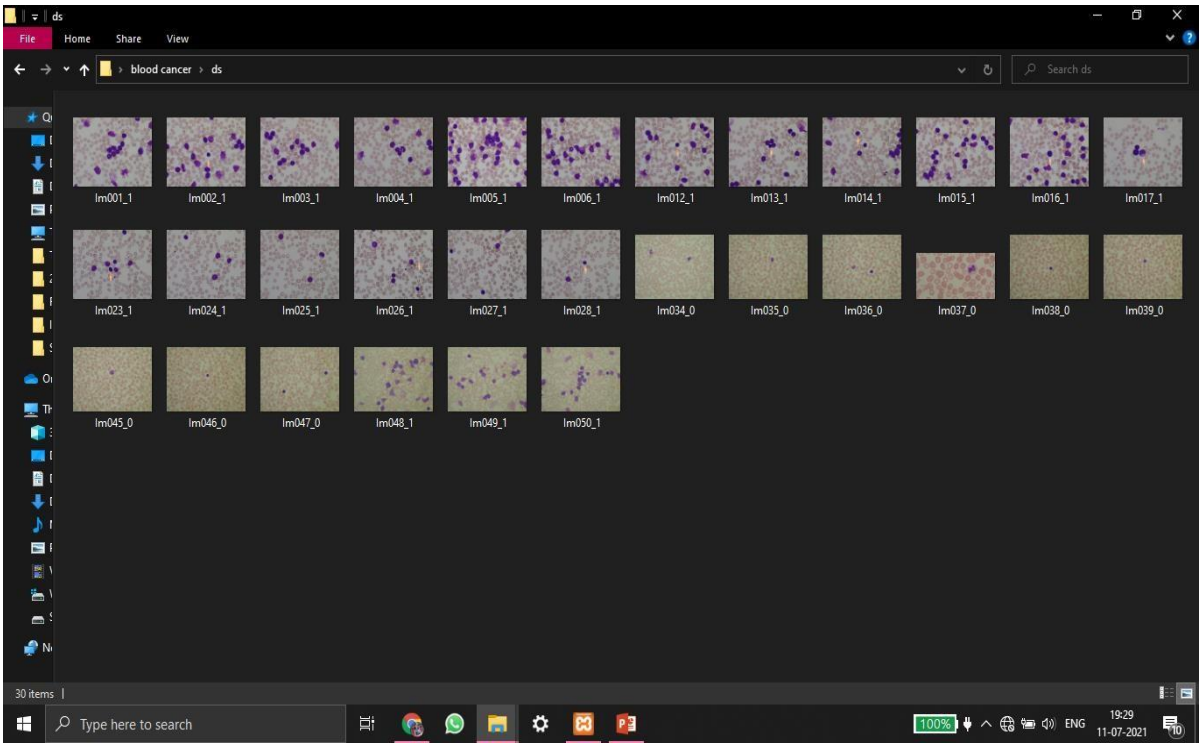

Fig 8.2.2 Cancerous Cells Dataset

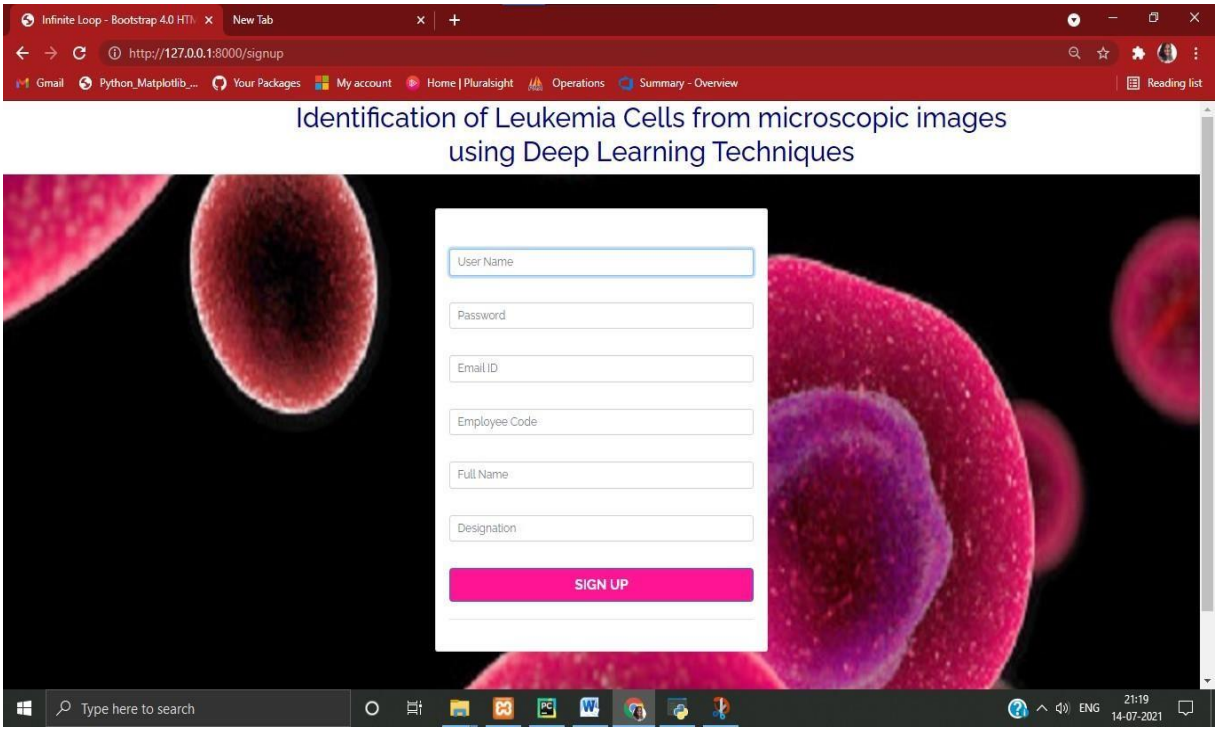

Fig 8.2.3 New User Sign-Up page

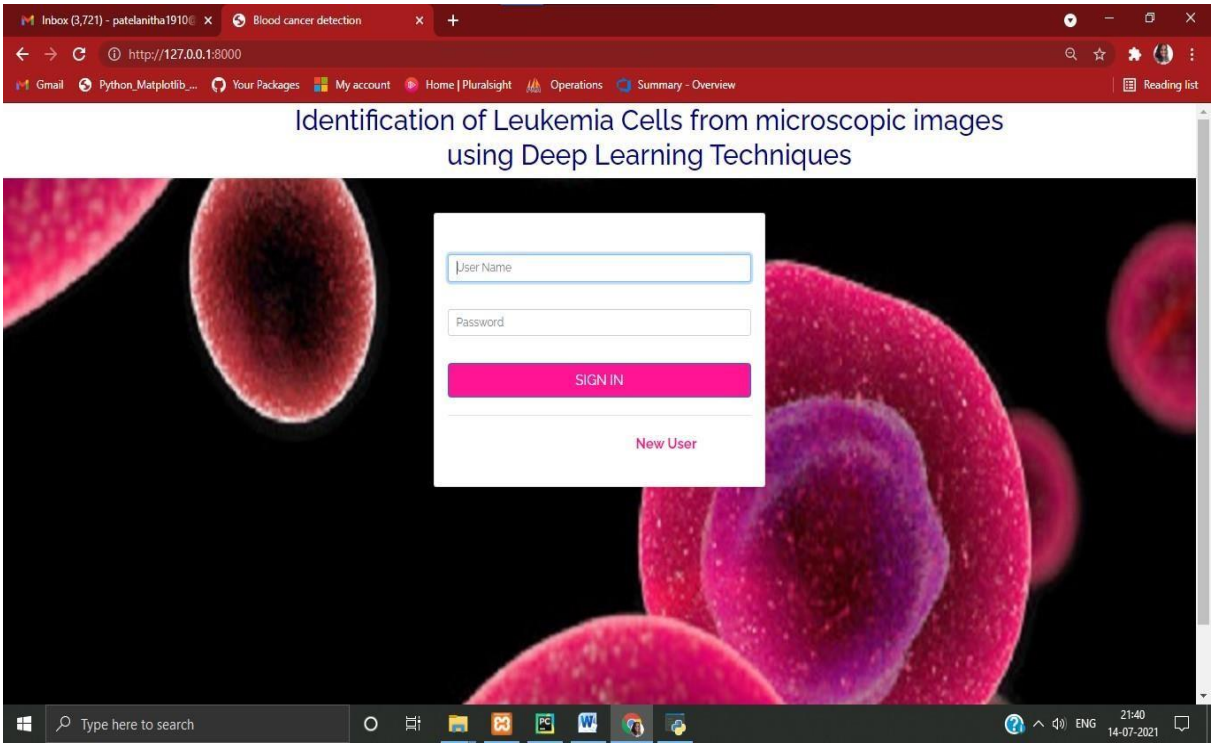

Fig 8.2.4 Existing User Sign-In page

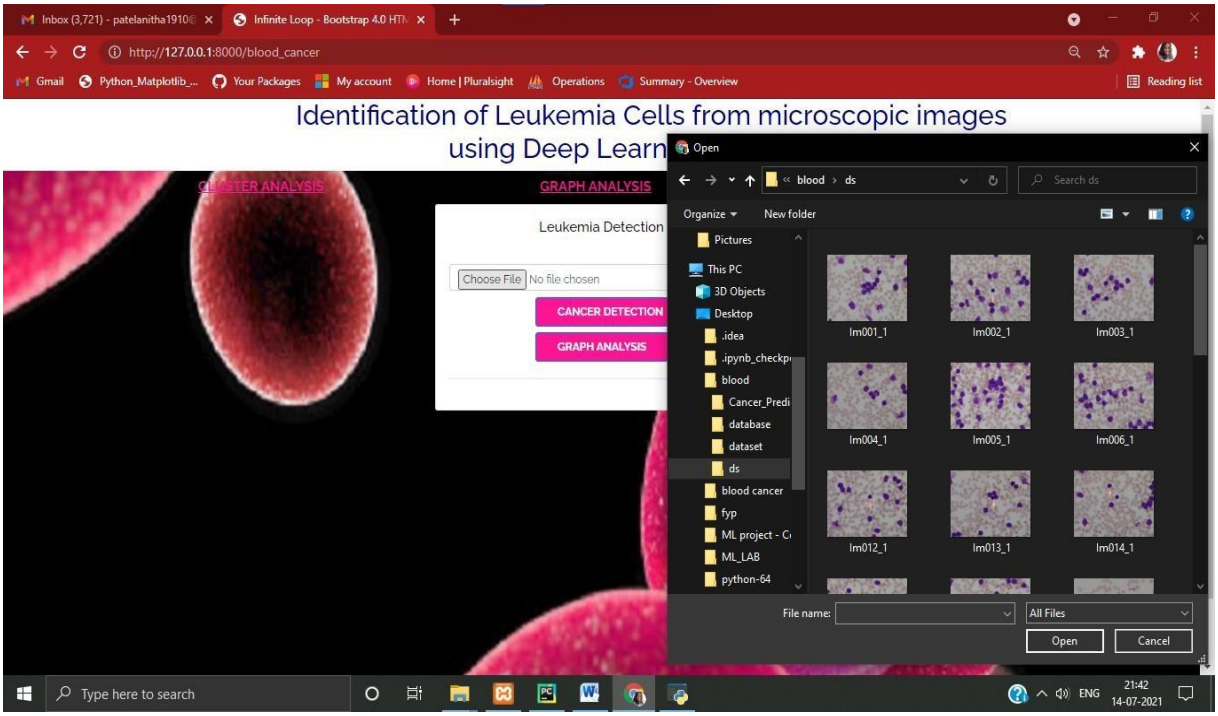

Fig 8.2.5 Uploading Image for Detection

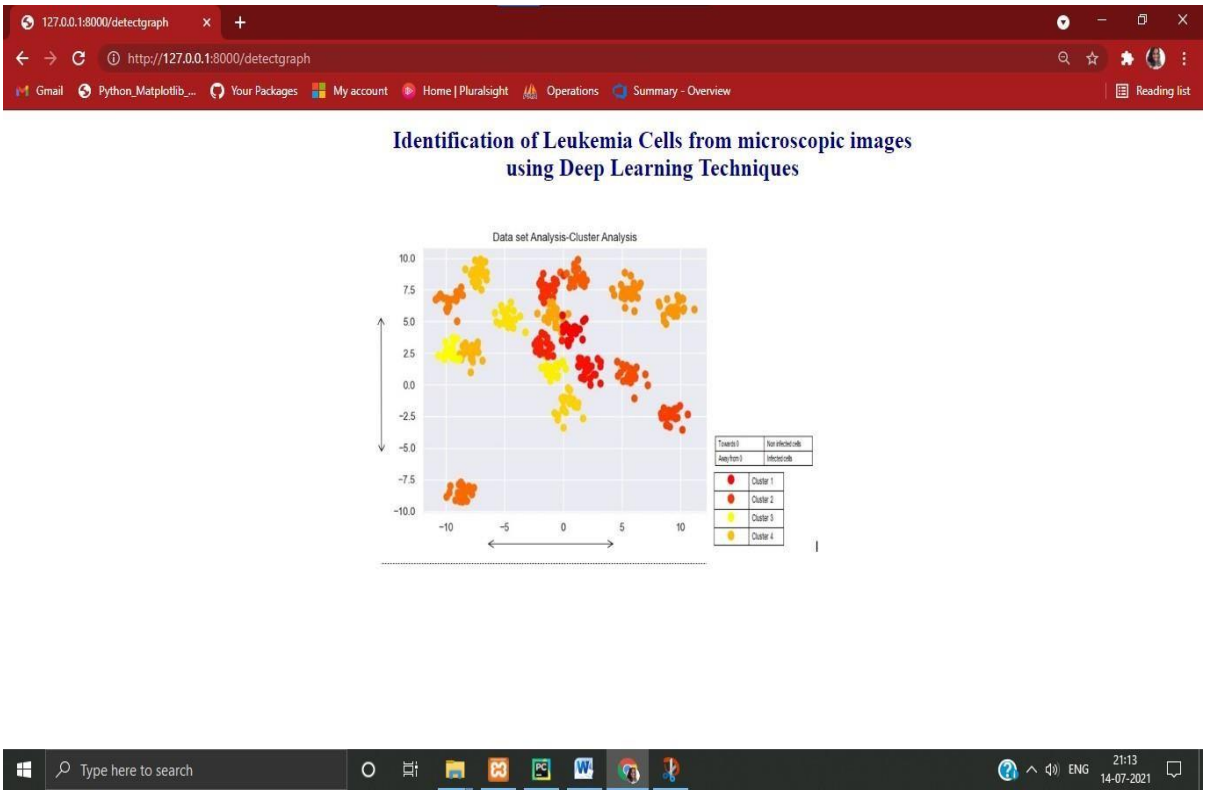

Fig 8.2.6 Deployed Cluster Analysis Graph

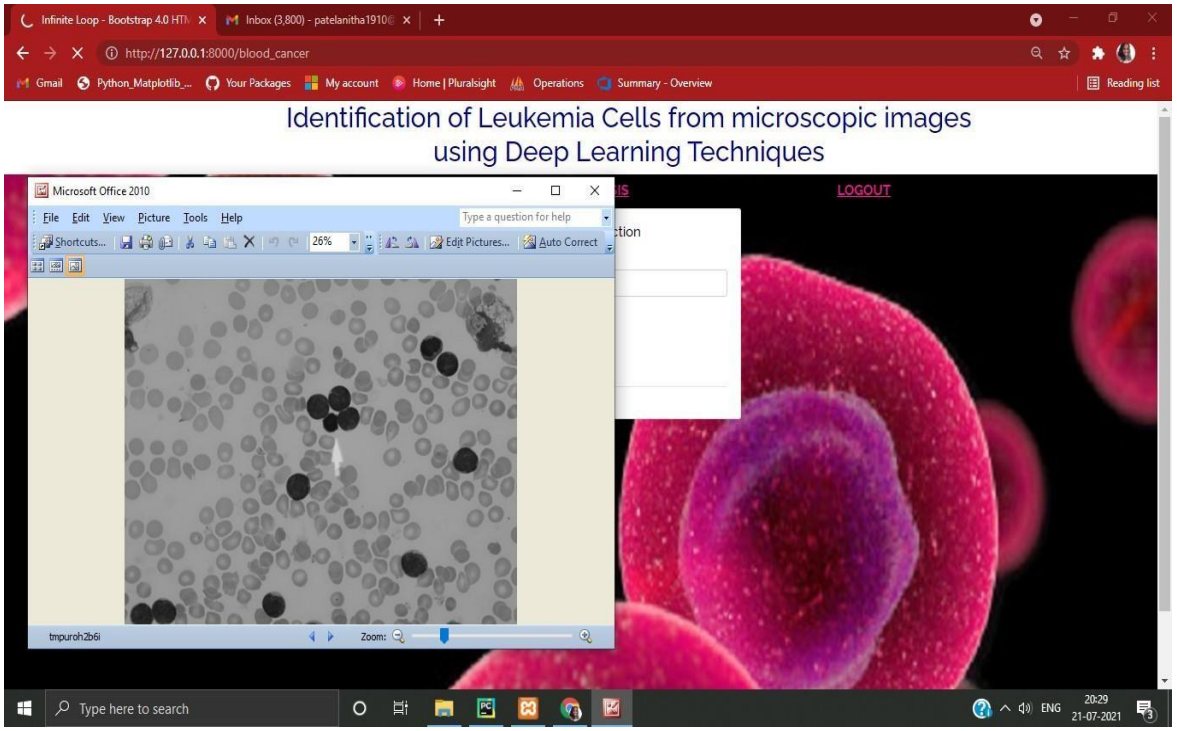

Fig 8.2.7 Grey scaling of Normal Blood cell image

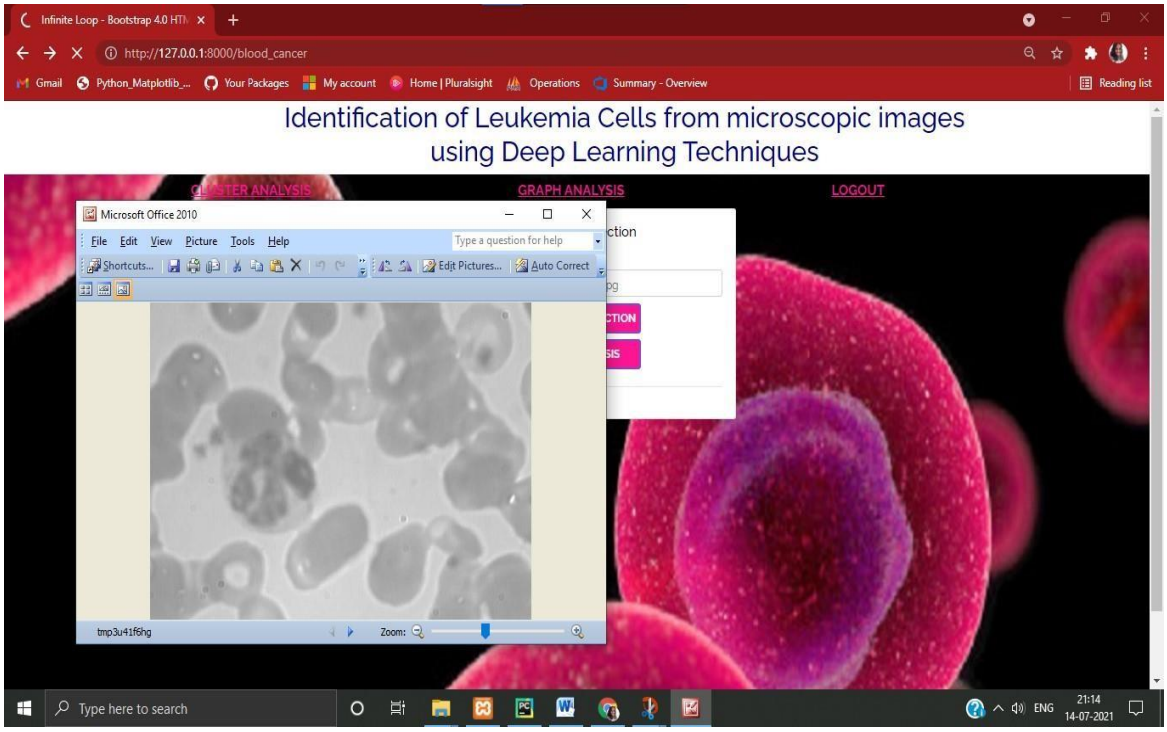

Fig 8.2.8 Grey scaling of Cancer Blood cell image

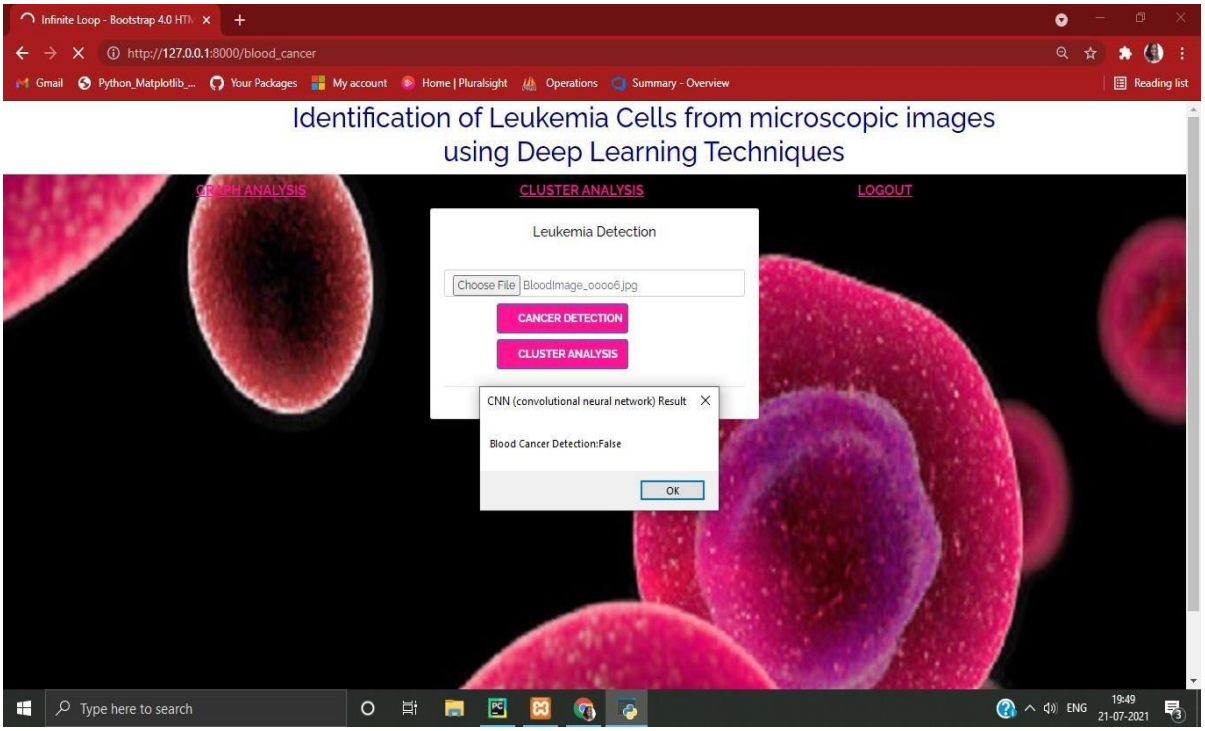

Fig 8.2.9 Result of Normal blood cells-False

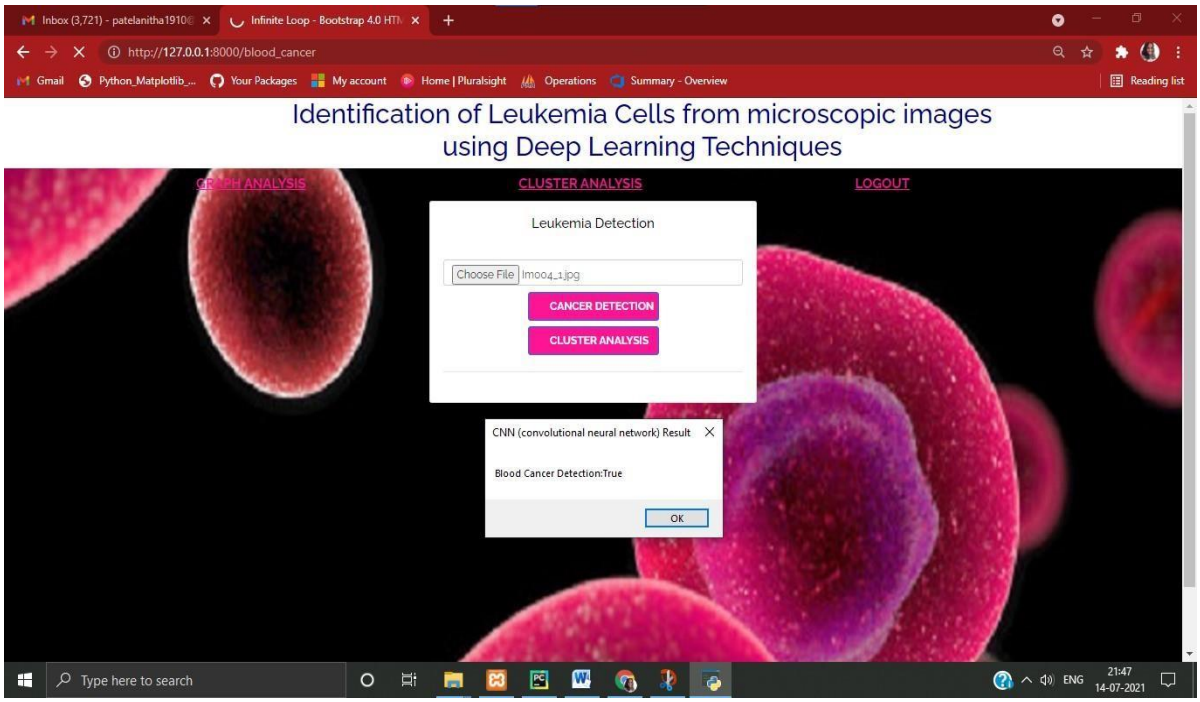

Fig 8.2.10 Result of Cancer blood cells-True

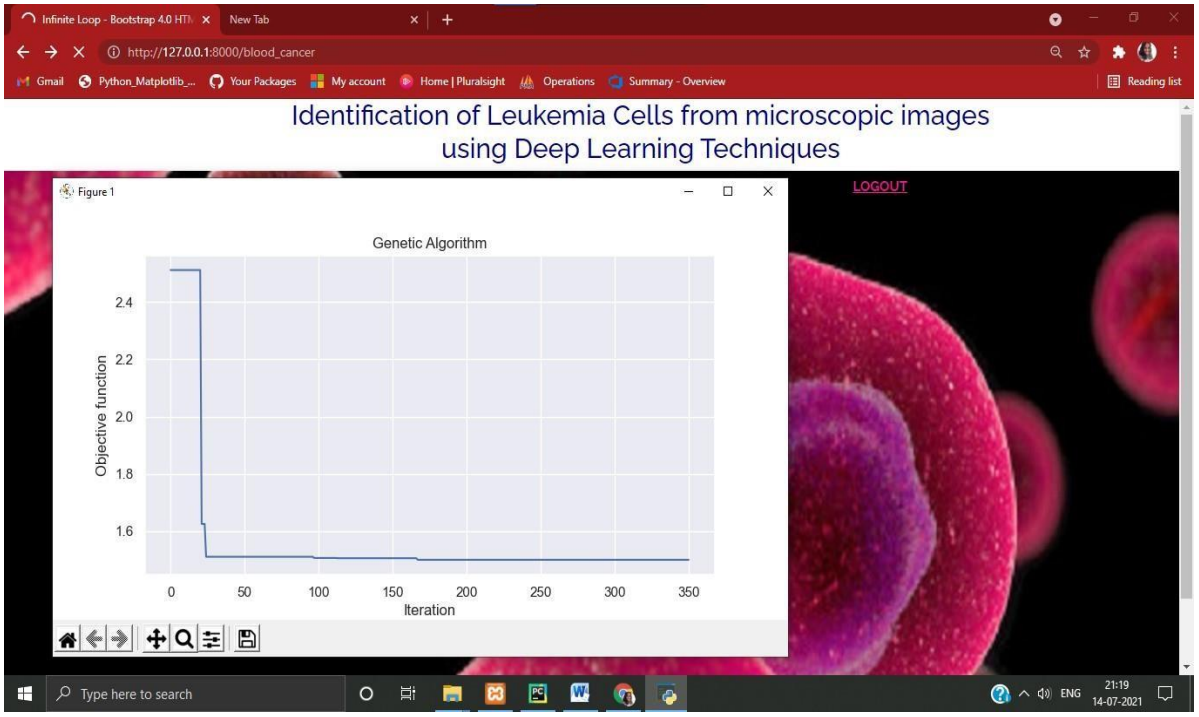

Fig 8.2.11 Genetic Algorithm Graph-Normal Blood cell

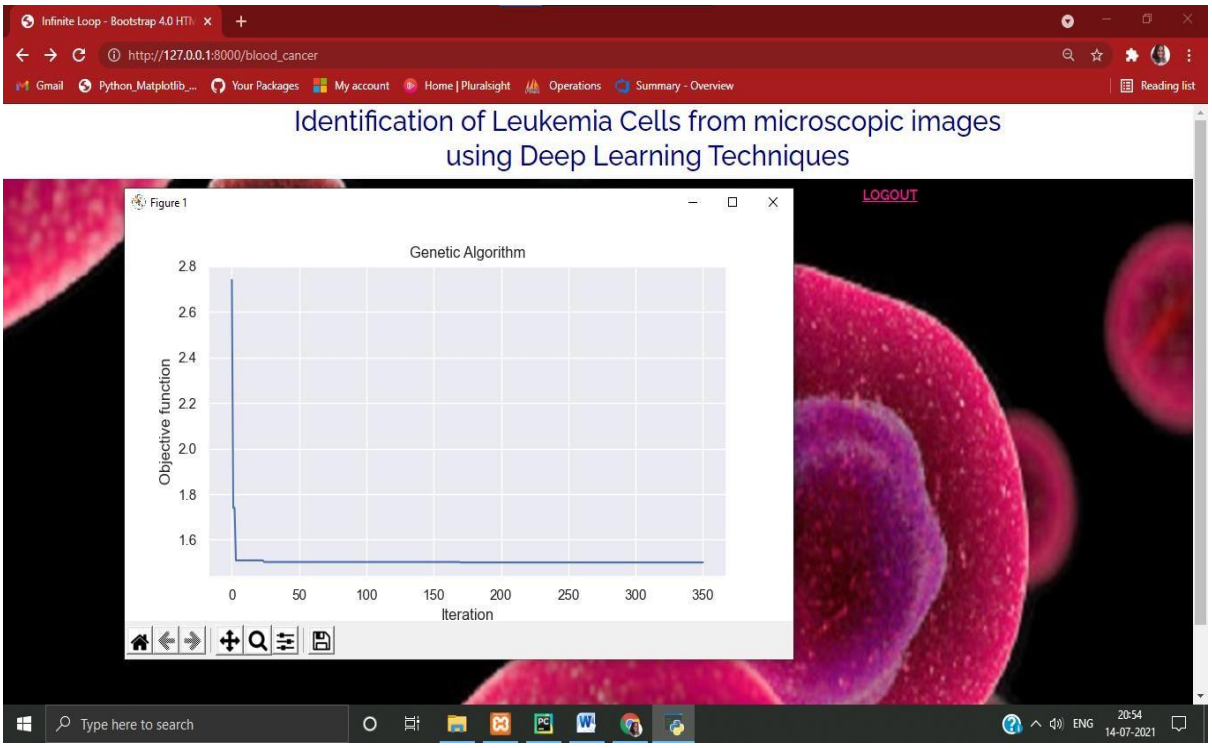

Fig 8.2.12 Genetic Algorithm Graph-Cancer Blood cell

CHAPTER 9

RESULT ANALYSIS

9.1 NORMAL BLOOD CELL IMAGE CONVOLUTING

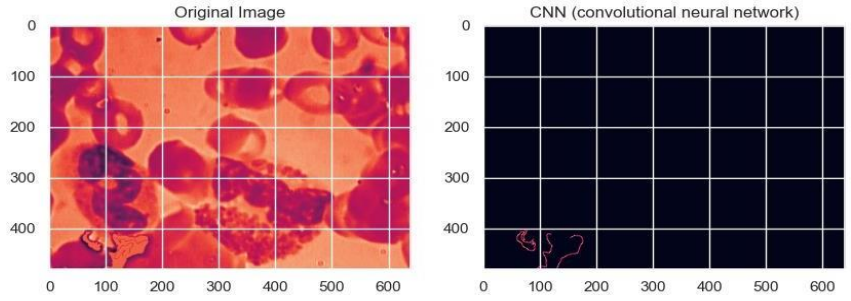

Figure 9.1.1 Normal cell Example 1

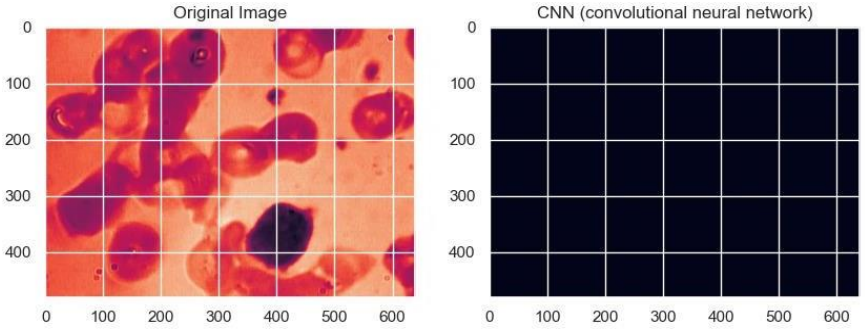

Figure 9.1.2 Normal cell Example 2

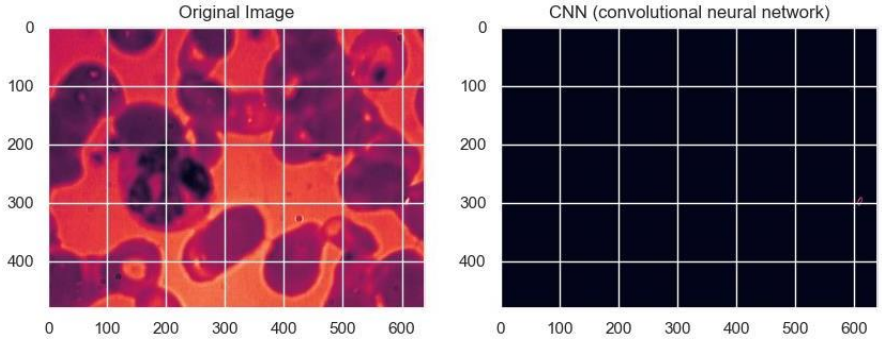

Figure 9.1.3 Normal cell Example 3

9.2 CANCEROUS BLOOD CELL IMAGE CONVOLUTING

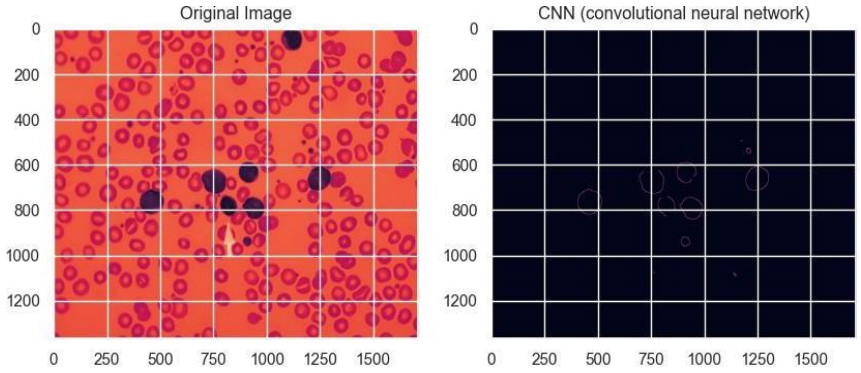

Figure 9.2.1 Cancerous cell Example 1

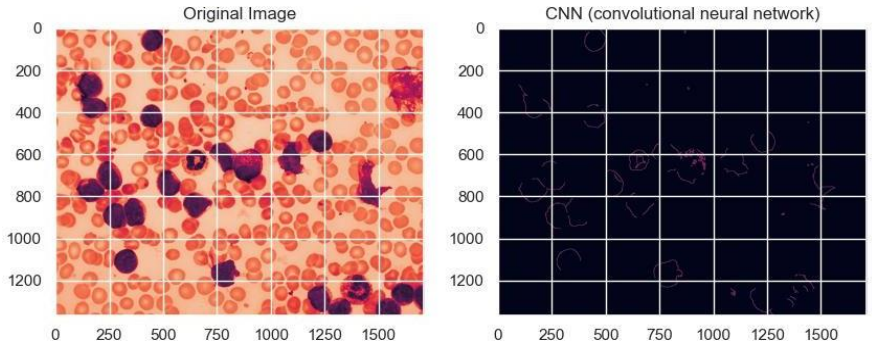

Figure 9.2.2 Cancerous cell Example 2

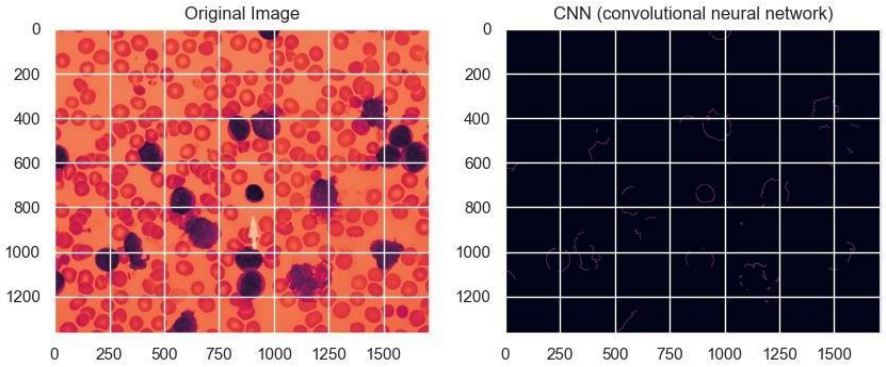

Figure 9.2.3 Cancerous cell Example 3

CONCLUSION

Health care information is a major research topic. Statistical study says around 156 people each day and 6 people per hour, depart this life due to Leukaemia. So doctors would look into an accurate application like the one we tried to project, where a simple image input can give a valuable prediction.

Therefore, as discussed all along the article we have come across a model where a hematologist can eradicate his traditional manual counting method and get an automated output. This is very usual because it initially can satisfy the basic check before a patient takes an extra diagnosis. If the automated prediction tells the being of cancer to be null, then it provide a sense of relief from the fatal disease.

Coming to the accuracy, it mentions to have played a better role in comparison to other ML algorithms. The cluster analysis explains the result vividly. It sums up to 97.75% as per the available datasets with least number of parameters. However, the discussion does not show the proposed model with larger datasets and long training time to achieve great accuracy here. Then looking into future scope, the interface can be included with prediction of reoccurrence of cancer cells, where such models inform patent screening and treatment patterns, potentially improving result outcomes and reducing overall healthcare costs.

REFERENCES

[1] Cruz JA, Wishart DS. "Applications of machine learning in cancer prediction and prognosis." Cancer Inform. 2017 Feb

[2] "A statistical and ML methods for modelling cancer risk using structured clinical data", JUNE 2018

[3] Cruz JA, Wishart DS. Applications of machine learning in cancer prediction and prognosis. Cancer Inform, 2017 Feb

[4] Pan, L., Liu, G., Lin, F. et al. Machine learning applications for prediction of relapse in childhood acute lymphoblastic leukaemia. Sic Rep 7, 7402 (2017).

[5] H. Abedy, F. Ahmed, M. N. Qaisar Bhuiyan, M. Islam, N. Y. Ali and M. Shamsujjoha, "Leukaemia Prediction from Microscopic Images of Human Blood Cell Using HOG Feature Descriptor and Logistic Regression" 2018 16th International Conference on ICT, 2018, pp. 1-6.

[6] Faisal Asadi, Fiqhri Muliandra Putra, Fadhliah Syafria, Okfalisa ,"Implementation of Backpropogation Neural Network and Blood Cells Imagery Extration for acute Leukemia Classification",[IEEE-2017]

[7] Nikhil Shah, Sarika Jain ,"Detection of disease in Cotton Leaf usins Artificial Neural Network",[IEEE-2019]

[8] Nivulina MISCHIE,Adriana ALBU,"Artificial Neural Networks for Diagnosis of Coronary Heart Disease",[IEEE 2020]

[9] Megha Chovatiya,Anushks Dhameliya conference publication,"Prediction of Dengue using Recurrent Neural Network",[IEEE-2019]

[10] Rashmi Panda,N.B.Puhan, Aparna Rao , Ganapathy Panda ," Recurrent neural Network based retinal nerve fiber layer defect detection in early glaucoma",[IEEE-2017]

[11] Rohit Agarwal, Sachinandan Satapathy, Raja Kumar K, "Detection of white blood cells using image processing",[IEEE-2019]

[12] Dharani T,Hariprasath S,"Diagnosis of Leukemia and its types using digital Image processing techniques",[IEEE-2018]

[13] www.stackoverflow.com

[14] www.youtube.com

[15] www.python.org

[16] www.google.com

and more.