# 

# MALARIA DETECTION USING DEEP LEARNING

*A project report submitted in partial fulfilment of the requirement for the award of the degree of*

## BACHELOR OF TECHNOLOGY

**IN**

## COMPUTER SCIENCE AND ENGINEERING

Submitted by

|  |  |
| --- | --- |
| PRAYAGA SUBRAHMANYA ASHOK | (5191411045) |
| DANTINALU VAMSI KRISHNA | (5191411014) |
| ESWAKOTA BALA SUBRAHMANYAM | (5191411018) |
| KAPIL YADAV | (5191411062) |

Under the Esteemed Guidance of

### Dr. A.S.S.V. RAM KUMAR, M.Tech, Ph.D

**Associate** **Professor**



### DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING ENGINEERING AND TECHNOLOGY PROGRAM

**GAYATRI VIDYA PARISHAD COLLEGE FOR DEGREE AND PG COURSES (A)**

### Rushikonda, Visakhapatnam – 45

(Approved by AICTE| Accredited by NBA| Accredited by NAAC| Affiliated to Andhra University ) An ISO 9001:2015 Certified Institution

### 2019-2023

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING ENGINEERING AND TECHNOLOGY PROGRAM**

### GAYATRI VIDYA PARISHAD COLLEGE FOR DEGREE AND PG COURSES (A)

**Rushikonda, Visakhapatnam - 45**



# CERTIFICATE

This is to certify that the project report entitled **“MALARIA DETECTION USING DEEP LEARNING”** being submitted by PRAYAGA SUBRAHMANYA ASHOK (5191411045), DANTINALU VAMSI KRISHNA (5191411014), ESWAKOTA BALA SUBRAHMANYAM (51914110118), KAPIL YADAV (5191411062) in the partial fulfillment for the award of the Degree of Bachelor of Technology in Computer Science and Engineering to the Engineering and Technology Program, Gayatri Vidya Parishad College for Degree and PG Courses (A), Visakhapatnam is a record of bonafide work carried out under my guidance and supervision.

### Project Guide Head of the Department

Dr. A.S.S.V. Ram Kumar Dr. N.V. Ramana Murty

M.Tech, Ph.D M.Tech, Ph.D

### Associate Professor Professor

**External Examiner**

# VISION AND MISSION OF THE INSTITUTE

## VISION

**Creating Human Excellence for a Better Society**

## MISSION

**Unfold into a World class organization with strong academic and research base, producing responsible citizens to cater to the changing needs of the society.**

# VISION AND MISSION OF THE DEPARTMENT

## VISION

**“Create and sustain as a Centre of excellence in Computer Science and Engineering and allied areas of research to pave the way for providing better technocrats to the society.”**

## MISSION

**“To provide strong *conceptual base (M1)* for acquiring programming skills, to develop high *computational innovative skills (M2)* and to empower *leadership qualities* with strong *ethical base (M3)”***

# PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

### PEO1: Ability to develop hardware and software problem solving capabilities by inculcating strong fundamental, analytical, and communication skills.

**PEO2: Ability to build confidence to address the challenges in their work and to apply innovative technical tools through interaction with professional bodies.**

### PEO3: Ability to gain sufficient depth of knowledge, Self-learning skills, technical skills and leadership qualities through participating in personality development activities, so as to sustain in multidisciplinary environment.

**PEO4: To be able to respond to societal needs in professional and ethical concerns.**

# PROGRAM SPECIFIC OUTCOMES (PSOs)

### PSO 1: System Inception and Elaboration: Conceptualize the software and/or hardware systems, system components and processes/procedures through requirement analysis, Modelling/Design of the system using various architectural/design patterns, Standard Notations, procedures and algorithms.

**PSO 2: System Construction: Implement the systems, Procedures and Processes using the state of the art technologies, standards, tools and Programming Paradigms.**

### PSO 3: System Testing and Deployment: Verify and Validate the Systems, Procedures and Processes using various testing and verification techniques and tools.

**PSO 4: Quality and Maintenance: Manage the quality through various product development strategies under revision, transition and operation through maintainability, flexibility, testability, portability, reusability, interoperability, correctness, reliability, efficiency, integrity and usability to adapt the system to the changing structure and behavior of the systems/environments.**

# DECLARATION

We hereby declare that the project entitled **“MALARIA DETECTION USING DEEP LEARNING*”*** submitted in partial fulfillment of the requirements for the award of Bachelor of Technology in Computer Science and Engineering, to Engineering and Technology Program, Gayatri Vidya Parishad College for Degree and PG Courses (A). We assure that this project is

not submitted in any other University or College.

**Name & Signature of the Students**

|  |  |
| --- | --- |
| PRAYAGA SUBRAHMANYA ASHOK | (5191411045) |
| DANTINALU VAMSI KRISHNA | (5191411014) |
| ESWAKOTA BALA SUBRAHMANYAM | (5191411018) |
| KAPIL YADAV | (5191411062) |

# ACKNOWLEDGEMENTS

With great pleasure we want to take this opportunity to express our heartfelt gratitude to all the people who helped in making main project work a grand success.

First of all we express our deep sense of gratitude to Dr. A.S.S.V. Ram Kumar, Associate Professor, for his constant guidance throughout our main project work.

We are grateful to Project Co-ordinator Sri. G. Kalyan Chakravarthi , Assistant Professor, for his valuable suggestions and guidance given by him during the execution of this main project.

We would like to thank Dr. N.V. Ramana Murty, Professor, Head of the Department of Computer Science of Engineering, for being moral support throughout the period of our study.

We are highly indebted to Dr.B.V. Ramana Murty, Director and Prof. S Rajani, Principal for giving us the permission to carry out this main project in the college.

We would like to thank all the Teaching and Non-Teaching staff of CSE Department for sharing their knowledge with us.

|  |  |
| --- | --- |
| PRAYAGA SUBRAHMANYA ASHOK | (5191411045) |
| DANTINALU VAMSI KRISHNA | (5191411014) |
| ESWAKOTA BALA SUBRAHMANYAM | (5191411018) |
| KAPIL YADAV | (5191411062) |

# ABSTRACT

As the most lethal disease in the world, malaria presents a significant challenge for the health department. Malaria has long been identified via a blood smear. For human red blood cells that were parasite-infected, competent specialists performed microscopy in a lab setting. The doctor's ability and knowledge are used to make the diagnosis. A well-informed person is required for the test. In this project, Malaria is being evaluated from cell pictures using deep learning methods. This well-researched deep learning model uses a CNN that recognizes and classifies objects on previous study thin blood smears with infected cells on a collection of microscope slides with five-fold convolution layers improvement in sensitivity within the cross-validation layer. A CNN trained on 27,560 single-cell images is the method used in this study to understand the cell's parameters. Once the image is submitted, a UI is created that allows the trained model to determine whether the image contains something or not.

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| **CHAPTER** | **Page No** |
| 1. Introduction | 1 |
| 1.1 Introduction to Malaria Detection | 1 |
| 1.2 Applications | 1 |
| 1.3 Introduction to CNN | 4 |
| 2. Literature Survey | 6 |
| 3. Requirements and Analysis | 10 |
| 3.1 Requirements Elicitation | 10 |
| 3.2 Problem Statement | 11 |
| 3.3 Scope of the System | 11 |
| 3.4 Functional Requirements | 12 |
| 3.5 Non Functional Requirements | 12 |
| 4. System Design | 15 |
| 4.1 Design Goals | 15 |
| 4.2 System Architecture | 16 |
| 4.3 Data Flow Diagram | 17 |
| 4.4 Analysis | 18 |
| 4.4.1 UML Diagrams | 28 |
| 5. Implementation | 23 |
| 5.1 Dataset | 24 |
| 5.2 Feature Extraction | 26 |
| 5.3 Performance Matrices | 27 |

|  |  |
| --- | --- |
| 1. Testing    1. Testing Activities    2. Testing Types    3. Test Cases | 28  28  28  30 |
| 1. Result    1. Epoch    2. Graphs    3. UI design results 2. Sample Code    1. Test case Image Results 3. Conclusion 10.References | 31    31  32  33  34  40  42  43 |

**LIST OF TABLES**

|  |  |
| --- | --- |
| **Title** | **Page No** |
| 6.3 Test cases | 30 |

**LIST OF FIGURES**

|  |  |
| --- | --- |
| **Title** | **Page No** |
| Fig 1.3: CNN architecture | 4 |
| Fig 4.2.1 System Architecture Fig | 16 |
| Fig 4.3: Data Flow Diagram | 17 |
| Fig.4.4.1(a) Use case diagram | 19 |
| Fig.4.4.1 (b) Activity diagram | 20 |
| Fig 4.4.1(c) Class Diagram | 21 |
| Fig 4.4.1 (d) Sequence Diagram | 22 |
| Fig 5.1 (a) Infected Image | 25 |
| Fig 5.1 (b) Uninfected Image | 25 |
| Fig 5.3 Sequential CNN | 28 |
| Fig.6.2.4 Black box testing | 30 |
| Fig7.1: 20 Epoch | 32 |
| Fig 7.2(a) Model accuracy graph | 33 |
| Fig 7.2(b) Model loss graph | 33 |
| Fig 7.3(a) Malaria infected image | 34 |
| Fig 7.3(b) Malaria uninfected image prediction | 34 |
| Fig 8.3(a) Test case-1 | 44 |
| Fig 8.3(b) Test case-2 | 44 |
| Fig 8.3(c) Test case-3 | 45 |
| Fig 8.3(d) Test case-4 | 45 |

# INTRODUCTION

## Introduction to Malaria Detection

The origins of intestinal disease can be traced back to Africa's landmass. The intestinal issue began as a result of plasmodium falciparum infection, which is the cause of this illness. The mosquitoes' organism has spread from one side of the globe to the other, contaminating the environment. Contamination can survive in a warm, delicate environment, but not in a frigid climate. The contamination was 40 million years old and came from a particularly ancient epoch. Wilderness fever can affect any animal, as well as people of all ages, from infants to the elderly. Starting with a fever, to a drowsy state, to death. The disease targets the human body's platelets, breaking the white platelets and preventing the organs from functioning properly. Only by obtaining blood tests from the individual and viewing them in the amplified focal point can intestinal disease be detected. Individual sporozoites migrate quickly into the liver, where they multiply abiogenetically in the liver cells for 7 to 10 days. The parasites will then become merozoites, which will travel through the circulatory system and settle in lung vessels. The merozoites then proceed through the red platelets and copy even more, after which the cell explodes. This will be accomplished by communicating with another sound person an example of a mosquito carrier. If a person is suffering from wilderness fever, he or she will be able to tell by the indicators that his or her human body will emit as a notification image. The human body will begin to produce white platelets in order to protect itself from malarial cells. It causes a high fever, headache, nausea, vomiting, stomach pain, and even exorbitant lethargies. Regardless matter how many AI models there were to foresee intestinal ailment. A large learning model is employed in the proposed work to predict wilderness fever with high precision.

## Applications

The use of deep learning for malaria detection has several advantages. First, it is highly accurate, with reported accuracies of up to 97% on some datasets. Second, it is automated, meaning that it can analyze large numbers of images quickly and efficiently. Finally, it is relatively cheap and accessible, requiring only a microscope, camera.

Overall, the application of deep learning in malaria detection shows great potential for improving the speed and accuracy of diagnosis, which can ultimately save lives..

## Natural Language Processing

Deep learning is used to improve language models for tasks such as sentiment analysis, speech recognition, and language translation. Natural language processing (NLP) deals with building computational algorithms to automatically analyze and represent human language. NLP-based systems have enabled a wide range of applications such as Google’s powerful search engine, and more recently, Amazon’s voice assistant named Alexa. NLP is also useful to teach machines the ability to perform complex natural language related tasks such as machine translation and dialogue generation. Other Applications include chatbots, and language learning tools.

## Speech Recognition

Speech recognizers are made up of a few components, such as the speech input, feature extraction, feature vectors, a decoder, and a word output. The decoder leverages acoustic models, a pronunciation dictionary, and language models to determine the appropriate output. They integrate grammar, syntax, structure, and composition of audio and voice signals to understand and process human speech . Applications such as automated voice assistants, dictation software, and speech-to-text systems.

## Computer Vision

Computer vision needs lots of data. It runs analyses of data over and over until it discerns distinctions and ultimately recognize images. For example, to train a computer to recognize automobile tires, it needs to be fed vast quantities of tire images and tire-related items to learn the differences and recognize a tire, especially one with no defects.

Two essential technologies are used to accomplish this: a type of machine learning called deep learning and a convolutional neural network (CNN). Applications include self-driving cars, security and surveillance systems, and medical imaging.

## Robotics

Deep learning has been used to improve perception in robotics. Perception refers to the ability of robots to perceive and understand their environment, and deep learning has made significant contributions to this area. Deep learning has also been used to improve motion planning and control in robotics. Motion planning refers to the process of determining the path that a robot should follow to achieve a given task, and deep learning can help to optimize this process. Additionally, deep learning algorithms have been used to develop more precise and efficient control systems for robots, which can improve their accuracy and speed.

## Finance

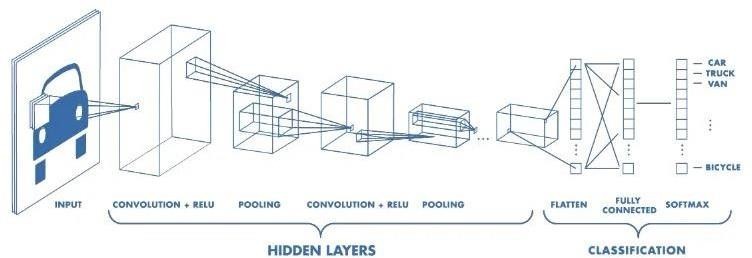
Finance companies can use deep technologies to automate tasks such as paperwork, calculations, data monitoring, and claims processing. This can free up employees to focus on more value-added activities. Deep learning is used in finance for fraud detection, risk assessment, and trading strategies. Deep learning models can analyze customer data to personalize financial services and improve customer experience. Applications include chatbots and virtual assistants for customer service, customer segmentation and profiling, and personalized marketing. Applications include credit scoring, insurance underwriting, and investment management.

## Gaming

Deep learning models can analyze player behavior and game data to provide insights into player engagement, identify areas for improvement, and optimize game design, to analyze natural language inputs such as voice commands or chat interactions. This can be used to create more intuitive game controls and more immersive game environments. Deep learning models can be used to analyze game visuals, such as identifying objects or characters, recognizing facial expressions, and tracking eye movements. Applications include character animation, game design, and player behavior modeling.

## Introduction to Convolution Neural Networks (CNN)

Neural network is a set of algorithms that mimic the human brain and it finds a relationship between the data to get solutions using these algorithms. CNN is a type of Neural Network where the mathematical operation used to find the relationship between the data is Convolution. Traditional neural network fails when coming to complex problems such as image classification, video classification, pattern recognition, etc. but CNN has achieved great success in these applications, yielding good accuracy. CNN consists of mainly 4 Layers, viz. convolutional, pooling, dropout and fully connected layers. The convolutional layer consists of small patches, which transforms the entire image basedon the filter values. ReLu, or rectified linear unit layer, is present, and these layers are linked to other completely connected layers. In order to ensure non-linearity as the data passes through each layer of the network, the ReLu layer serves as an activation function. Without it, the data would no longer have the desired level of dimension . These layers together extract the features from the input data. The algorithm learns from the feature, where the features of interest are represented by each convolution filter. A feature map with a convolution is produced by applying a filter to a set of pixels in the convolution layer. A feature map's size is decreased by the pooling layer to facilitate processing .This is accomplished by utilize average pooling, which uses the average value, and max pooling, which uses output=1, min pooling, which uses output=0.



**Fig 1.3:** CNN architecture

With the rapid development of computer technology and the improvement of computer hardware performance, deep learning has made great progress. Artificial neural networks have been widely used in different fields due to their excellent performance in image classification and recognition. CNN is a multilayer feedforward neural network with a convoluted structure, which has good fault tolerance and self-learning capabilities. It can deal with problems in complex environments and unclear backgrounds. Its generalization ability is significantly better than that of other methods. CNN generally consists of an input layer, several convolution layers, pooling layers, a fully connected layer, and an output layer. It can carry out supervised learning and unsupervised learning and is applied to computer vision, natural language processing, and other fields.

# LITERATURE SURVEY

### Comparision of Machine Learning algorithms for Malaria Detection using microscopic images

This paper is published in the year : 2019 The purpose of this paper is to make a study on Saiprasath G, Naren Babu R, ArunPriyan J, Vinay Kumar R, Sowmya V, Soman K P. The purpose of this paper is to make a study on the weighted KNN (K-Nearest Neighbors) algorithm is trained with the learned features using the Bayesian classifier, whose purpose is staining pixels for Identifying parasites. A color segmentation technique is used for separating the pixels into, parasites, and the background, which is based on classical supervised classification models. Supervised classification algorithms like Support Vector Machines, K-Nearest Neighbour, and Naive Bayes were evaluated using different color models namely the RGB model (Red Green Blue), normalized RGB model, and HSV model (Hue Saturation Value) respectively.

### Machine Learning approach for automated screening of malaria parasite using light microscopic images

This paper is published in the year : 2012 Authors: Dev Kumar Das a, Madhumala Ghosha, Mallika Pal b, Asok K. Maiti b, Chandan Chakraborty,. The purpose of this paper is to make a study on recent to address the development of computer-assisted malaria parasite characterization and classification using a machine-learning approach based on light microscopic images of peripheral blood smears. Bayesian learning and support vector machine (SVM) in order to provide higher classification accuracy using the set of discriminating features. Results show that the design approach provides the highest accuracy i.e., 84% for malaria classification by selecting the 19 most significant uses while SVM provides the highest accuracy i.e., 83.5% with 9 the most significant features. Finally, the performance of these two classifiers under the feature selection framework has been compared toward malaria parasite classification.

### Malaria Detection using Deep Learning

This paper ispublished in the year : 2020 Authors: Gautham Shekar, S. Revathy, Ediga Karthick Goud. This paper describes detection of malaria is to find whether a blood cell is infected or not. The concept of image processing using OpenCV and the contour detection concept are used on the blood cell to find the attributes of the blood cell. The Convolutional Neural Network is used for the detection of a parasite using the microscopic images that are taken.

### Automatic Detection of Malaria Parasites Using Unsupervised Technique

This paper is published in the year : 2019 Authors: Itishree Mohanty, P. A. Pattanaik and Tripti Swarnkar .The focus of this paper is towards comparing the computational paradigms of two unsupervised data reduction techniques, namely Auto encoder and Self Organizing Maps. The domain of inquiry in this paper is for automatic malaria identification from blood smear images, which has a great relevance in healthcare informatics and requires a good treatment for the patients. Extensive experiments are performed using the microscopically thick blood smear image datasets. Our results reveal that the deep learning based Auto encoder technique is better than the Self Organizing Maps technique in terms of accuracy of 87.5%.The Auto encoder technique is computationally efficient, which may further facilitate its malaria identification.

### A Deep Convolutional Neural Network for Detection of Malaria Parasite in Thin Blood Smear Images

This paper is published in the year : 2021 Authors: Milind Raj, Rohan Sharma, Deepak Sain. This paper describes a Deep Learning based image classification approach for the detection of malarial parasite existing in thin blood smear images, using Convolutional Neural Network (CNN) for effective feature extraction and accurate classification. The proposed CNN model can automatically extract intrinsic and discriminative features from images provided. Our custom Deep CNN model can classify the parasitized an uninfected with 93.47 percent accuracy .The CNN performs perfectly with image data. This paper annotates a comparison of the observed accuracy of proposed model using three different optimizers on training and validation data.

### Malaria Detection using Deep Learning Algorithms

This paper is published in the year : 2020 Authors: Gautham Shekar,S. Revathy,Ediga Karthick Goud. The malaria detected from the traditional method that is bringing the samples and analyzing the cell growth requires more time. So in the proposed work, deep learning model has been constructed to predict Malaria with high accuracy rate and low time duration. The Fine-Tuned CNN provided high accuracy rate compared to the other CNN models. The future work will be, working on the disease detection like pneumonia, breastcancer using CNN and planning for the detection of COVID19 smears in the lungs of the human body.

### Malaria Disease Recognition through Adaptive Deep Learning Models of Convolutional Neural Network

This paper is published in the year : 2019 Authors: Sammy V. Militante The study applies the malaria detection using deep learning methods through computer vision using four convolutional neural network models namely: AlexNet, GoogleNet, ResNet and VGGNet. Each layers of CNN architectures are evaluated physically such that the context to be used are in the forms of feature extraction and finetuning. The malaria dataset is obtained from the National Library of Medicine (NLM) database. 80% of acquired images is used as training data while the remaining 20% is used for testing and validation. Performance results of each generated malaria detection modes are compared in the context of their training time, learned features, accuracy and losses in recognizing the images.

**2.8 A Novel Data Augmentation Convolutional Neural Network for Detecting Malaria Parasite in Blood Smear Images**

This paper is published in the year : 2020 Authors: David Opeoluwa Oyewola, Emmanuel Gbenga Dada, Sanjay Misra & Robertas Damasevicius A new deep learning model, called the data augmentation convolutional neural network (DACNN) was proposed in this paper. The proposed model was trained by reinforcement learning to tackle this problem. The paper compared DACNN with other variations of CNN to investigate its performance. Simulation results show that DACNN performs better than the convolutional neural network (CNN) and the directed acyclic graph convolutional neural network (DAGCNN). The result shows that DACNN outperforms the previous techniques used in earlier studies in image processing and classification. DACNN achieves 94.79% classification accuracy while utilizing CNN achieved just 68.61%, and 72.62% accuracy, respectively.

**2.9** **Image analysis and machine learning for detecting malaria**

This paper is published in the year : 2018 Authors: Mahdieh Poostchi , Kamolrat Silamut,Richard J.Maude.e. One of the barriers toward a successful mortality reduction has been inadequate malaria diagnosis in particular. To improve diagnosis, image analysis software and machine learning methods have been used to quantify parasitemia in microscopic blood slides. This article gives an overview of these techniques and discusses the current developments in image analysis and machine learning for microscopic malaria diagnosis. We organize the different approaches published in the literature according to the techniques used for imaging, image preprocessing, parasite detection and cell segmentation, feature computation, and automatic cell classification. Readers will find the different techniques listed in tables, with the relevant articles cited next to them, for both thin and thick blood smear images. We also discussed the latest developments in sections devoted to deep learning and smartphone technology for future malaria diagnosis.

**2.10 Evaluations of Deep Convolutional Neural Networks for Automatic Identification**

**of Malaria Infected Cells**

This paper is published in the year : 2017 Authors: Yuhang Dong, Zhuocheng Jiang, Hongda Shen, W. David Pan This paper studied automatic identification of malaria infected cells using deep learning methods. We used whole slide images of thin blood stains to compile an dataset of malaria-infected red blood cells and non-infected cells, as labeled by a group of four pathologists. We evaluated three types of well-known convolutional neural networks, including the LeNet, AlexNet and GoogLeNet. Simulation results showed that all these deep convolution neural networks achieved classification accuracies of over 95%, higher than the accuracy of about 92% attainable by using the support vector machine method. Moreover, the deep learning methods have the advantage of being able to automatically learn the features from the input data.

# REQUIREMENTS AND ANALYSIS

## Requirements Elicitation

**Requirements Elicitation** is a feature that the system must have or a constraint that it must satisfy to be accepted by the clients. Requirements Engineering aims at defining the requirements of the system under construction. It includes two main activities namely Requirements Elicitation and Analysis.

### Identifying Actors

During this activity we identified the actors. The actors in our project are user and admin.

### Identifying Scenarios

During this activity, we observed the users and developed a set of detailed scenarios for functionality of the users.

### Identifying Use Cases

Once we and the users agreed on the set of scenarios, we derived a set of use cases that completely represent the future system.

### Refining Use Cases

During this activity, we ensured that the requirements specification is complete by detailing each use case and describing the behaviour of the system in the presence of errors and exceptional conditions.

### Identifying relationships among use cases

During this activity, we identified the dependencies among use cases and also consolidated the use case model by factoring out common functionality.

### Identifying Non-functional requirements

During this activity, developers, users and clients agree on aspects like performance of system, documentation, resources security and its quality.

## Problem Statement

With the regular manual diagnosis of blood smears, it is an intensive manual process requiring proper expertise in classifying and counting the parasitized and uninfected cells. Typically this may not scale well and might cause problems if we do not have the right expertise. Various machine-learning algorithms are used for the detection of malaria but were found to be having less accuracy so we are now using a deep-learning model for the prediction of malaria.

## Scope of System

The scope of the system is placed where this project can be used. Generally, this is used in facial image recognition system.

### Existing System

Microscopic Diagnosis Systems, examine the blood cells under the microscope to determine whether the person has been infected with malaria. It requires the presence of a trained microscopist. It is also a time taken process. These methods are entirely dependent on the course of infection, parasite load, and skilled technical resources. Various machine learning algorithms like SVM and Bayesian Learning are used to detect malaria. These algorithms were found to be having an accuracy of 83% for SVM and 84 % for Bayesian Learning. These were earlier done using a feature selection process so some important features may lost.

### Proposed System

The microscopic observation system and machine-learning algorithms have some defects due to which a deep learning model is proposed. The deep learning model was built using conventional neural networks to detect malaria. Using the CNN model to detect whether the smear cell is a parasite or a non-parasite. We will take the microscope-scanned smear cell images as our input and train the algorithm to detect whether that image is infected or un- infected.

## Functional Requirements

The functional requirements describe the inputs and outputs of the application. The Functional requirements of this project are as follows:

* + - Input : Loads Image of the malaria
    - Output : Predicts image consists of malaria or not.

## Non Functional Requirements

### Speed and Robustness:

The algorithm for the system should be robust and comparatively faster in recognizing the infected cells.

### Accuracy:

The data for testing when performed on the trained system should produce the exact or most accurate

### Performance:

The appropriate classification algorithm applied to the system should return the most relevant results while classifying the data.

### Software Requirements

Coding Language : Python

Software Environment : Jupyter Notebook, VsCode

HDD: NVIDIA

* + 1. **Hardware Requirements**

Operating System: Windows7

Processor : Intel Core i5-2348

CPU Speed : 2.30 GHz

Memory : 4 GB(RAM)

### Python

Python is an interpreter, high-level, general purpose programming language. Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural. It also has a comprehensive standard library.

Python interpreters are available for many operating systems. C Python, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of Python’s other implementations. Python uses dynamic typing, and a combination of reference counting and a cycle-detecting garbage collector for memory management.

### Jupiter Notebook

The Juypter Notebook is an open source web application that you can use to create and share documents that contain live code, equations, visualizations, and text. Juypter Notebook is maintaine by the people at Project Juypter. Juypter Notebooks are a spin-off project from thePython project, which used to have an Python Notebook project itself. The name, Juypter, comes from the core supported programming languages that it supports: Julia, Python, and R. Juypter ships with the Python kernel, which allows you to write your programs in Python, butthere are currently over 100 other kernels that you can also use.

### Tensor flow

TensorFlow is an open-source machine learning library developed by Google. It is designed to facilitate the development of machine learning models and deep neural networks. TensorFlow provides a wide range of tools and features for building, training, and deploying machine learning models. One of the key features of TensorFlow is its ability to handle computations on multi-dimensional arrays, or tensors, which makes it well-suited for working with high-dimensional data such as images, audio, and text. TensorFlow also provides a flexible and scalable framework for building and training various types of neural networks, such as convolutional neural networks, recurrent neural networks, and transformer networks.

TensorFlow supports a range of programming languages, including Python, C++, Java, and Go, and has a large and active community of users and contributors. It is widely used in research and industry for various applications such as image and speech recognition, natural language processing, and robotics. TensorFlow requires a 64-bit version of Python 3.6, 3.7, 3.8, or 3.9. You can check your Python version by opening a terminal or command prompt and running the command python --version.Install TensorFlow using pip, the Python package manager. Open a terminal or command prompt and run the following command:

pip install tensorflow

This will download and install the latest version of TensorFlow.

# SYSTEM DESIGN

System Design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. In System design, developers:

* + - * Define design goals of the project
      * Decompose the system into smaller sub systems
      * Design hardware/software strategies
      * Design persistent data management strategies
      * Design global control flow strategies
      * Design access control policies
      * Design strategies for handling boundary conditions.

System design is not algorithmic. It is decomposed of several activities. They are:

* + - * Identify Design Goals
      * Design the initial subsystem decomposition
      * Refine the subsystem decomposition to address the design goals.

System Design is the transform of analysis model into a system design model. Developers define the design goals of the project and decompose the system into smaller subsystems that can be realized by individual teams. Developers also select strategies for building the system, such as the hardware/software platform on which the system will run, the persistent data management strategy, the goal control flow the access control policy and the handling of boundary conditions. The result of the system design is model that includes a clear description of each of these strategies, subsystem decomposition, and a UML deployment diagram representing the hardware/software mapping of the system.

**4.1 Design Goals**

Design goals are the qualities that the system should focus on. Many design goals can be inferred from the nonfunctional requirements or from the application domain.

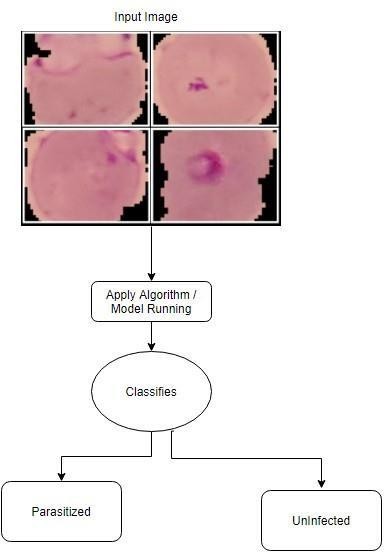
* + - **Accuracy:** In simplest terms, given a set of data points from repeated

measurements of the same quantity, the set can be said to be precise if the values are close to each other, while the set can be said to be accurate if their average is close to the true value of the quantity being measured.

* + - **Speed**: The speed of the system is accurate and efficient.
    - **Consistency**: If there are any errors caused by the system then that errors are rectified and maintained consistent.
    - **Completeness**: The system will be efficient without any errors and is consistent.

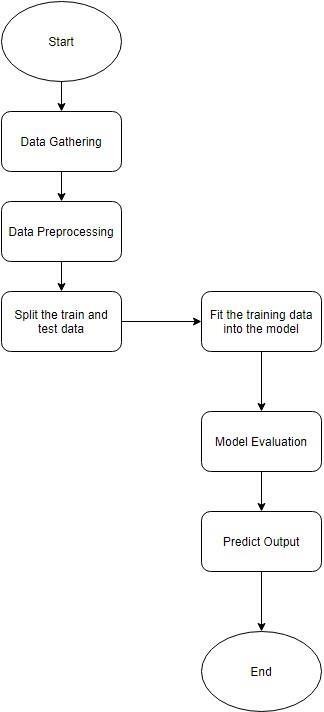
## System Architecture

A system architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviours of the system.



**Fig.4.2.1:** System Architecture

## DATA FLOW DIAGRAM



**Fig 4.3:** Data Flow Diagram

## ANALYSIS

The sequential Convolutional Neural Networks (CNNs) algorithm is used to detect the presence of malaria parasites in blood smear images. CNN is a type of artificial neural network that can analyze images by applying filters to identify patterns and features, allowing for highly accurate and efficient recognition. A large dataset of blood smear images is collected both with and without malaria parasites. The dataset is then used to train the Sequential CNN model to recognize patterns and features associated with the presence or absence of malaria parasites. Once the model is trained, it can be used to analyze new blood smear images and accurately detect the presence or absence of malaria parasites. The Fine-Tuned CNN provided a high accuracy rate of 98.17 on running for 20 epochs and got a model loss of 0.53. An interface is developed where a user can upload an image of a blood smear to a web application. The image is then processed by the Sequential CNN model, and the result whether malaria is present or not is displayed to the user.

### 4.4.1 UML Diagrams

1. **Use Case Diagram**

### Actors

Actors represent external entities that interact with the system. An actor can be human or an external system. During this activity, developers identify the actors involved in this system. In this project, the actors are as follows.

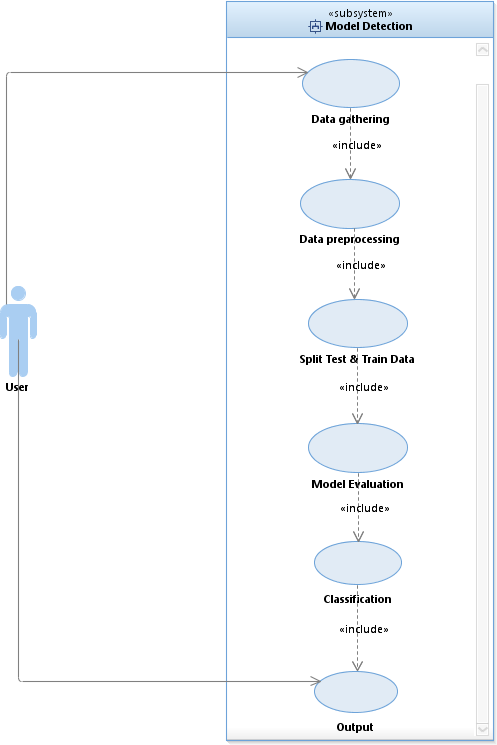
### User

The user actor is the one who uses the project by first getting registered, that is the user’s frontal face image is being captured and trained after which the user is identified by unique name and id.

### Use case

Use cases are used during requirement elicitation and analysis to represent the functionality of the system. A use case describes a function provided by the system that yields a visible result for an actor. An actor describes any entity that interacts with the system. The identification of actors and use cases results in the definition of the boundary of the system, which is, in differentiating the tasks accomplished by the system and the tasks accomplished by its environment. The actors are outside the boundary of the system, whereas the use cases are inside the boundary of the system. Actors are external entities that interact with the system. Use cases describe the behavior of the system as seen from an actor’s point of view. Actors initiate a

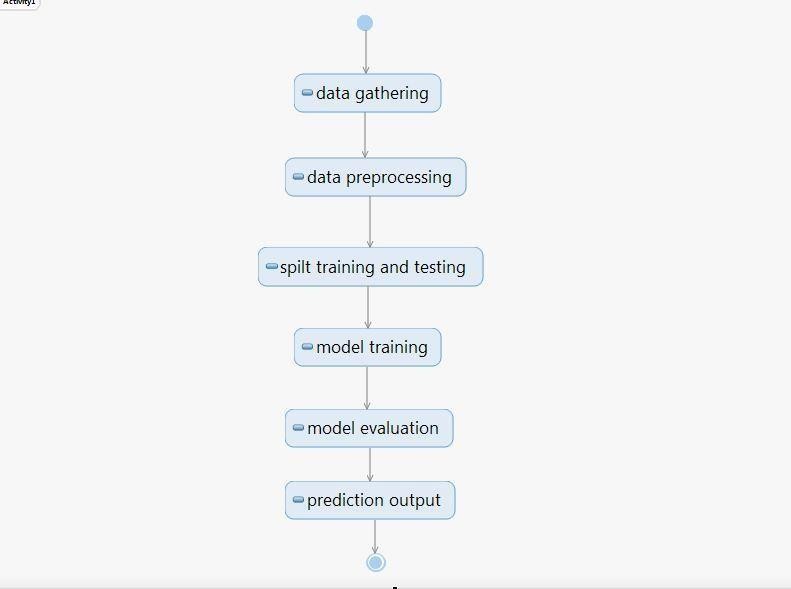
use case to access the system functionality. The use case then initiates other use cases and gathers more information from the actors. When actors and use cases exchange information, they are said to communicate.



**Fig 4.4.1 (a) :** Use Case Diagram

### Activity Diagram

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another.



**Fig.4.4.1 (b):** Activity diagram

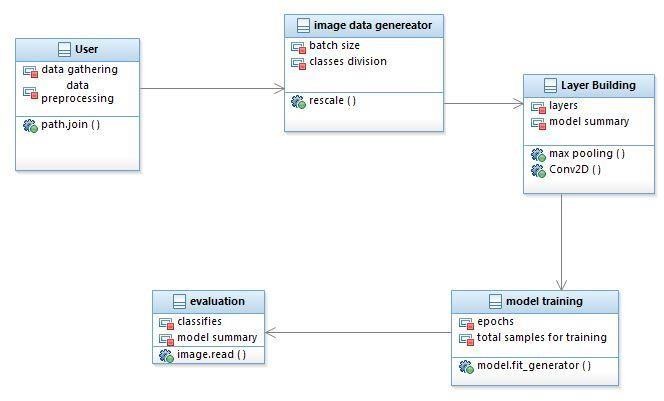
## Class Diagram

Class diagrams are one of the most useful types of diagrams in UML as they clearly map out the structure of a particular system by modelling its classes, attributes, operations, and relationships between objects. It is a static diagram that represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

The class shape itself consists of a rectangle with three rows. The top row contains the name of the class, the middle row contains the attributes of the class, and the bottom section expresses the methods or operations that the class may use. Classes and subclasses are grouped together to show the static relationship between each object.

The purpose of the class diagram can be summarized as:

* + Analysis and design of the static view of an application.
  + Describe responsibilities of a system.
  + Base for component and deployment diagrams.
  + Forward



**Fig.4.4.1(c) :** Class Diagram

## Sequence Diagram

A sequence diagram is the most commonly used interaction diagram. It simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

Sequence diagrams can be useful references for businesses and other organizations.

Purpose of sequence diagrams are:

* + Represent the details of a UML use case.
  + Model the logic of a sophisticated procedure, function, or operation.
  + See how objects and components interact with each other to complete a process.
  + Plan and understand the detailed functionality of an existing or future scenario.



**Fig.4.4.1 (d) :** Sequence Diagram

# 

# IMPLEMENTATION

## Coding Approach

The objective of the coding or programming phase is to translate the design of the system produced during the design phase into code in a given programming language, which can be executed by a computer and that performs the computation specified by the design. The coding phase affects both testing and maintenance. The goal of coding is not to reduce the implementation cost, but the goal should be to reduce the cost of later phases.

## Information Handling

Any system requires some amount of information during its operation selection of appropriate data structures can help us to produce the code so that objects of the system can better operate with the available information decreased complexity.

## Programming Style

Programming style deals with act of rules that a programmer must follow so that the characteristics of coding such as Traceability, Understand ability, Modifiability, and Extensibility can be satisfied. In this current system, we followed the coding rules for naming the variables and methods.

## Verification and Validation

Verification is the process of checking the product built is right. Validation is the process of checking whether the right product is built. During the Development of the system, Coding for the object has been thoroughly verified from different aspects regarding their design, in the way they are integrated etc. Every component in the proposed system has verified by giving different input data during the working process of all modules.

## Algorithm

Stage 1: Gather the preprocessed images and group them into a single report for easy transit.

Stage 2: Split the photos according to the train and test dataset.

Stage 3: Apply ImageDataGenerator to images and like the limitations of the pictures,then conduct the design region example.

Stage 4: To avoid acknowledgement pushing, process the image with string pool expert. Stage 5: Create a basic CNN model without arranging it and fit it.

Stage 6: Incorporate the images into the model and demonstrate how to use the tensor stream and Keras pack with the model.

Stage 7: In the preparation, use Epoch = (number of emphases \* bundle size)/total number of photos.th the model.

Step 8: Make the Frozen CNN model and fit it Step 9: Check the accuracy of the model.

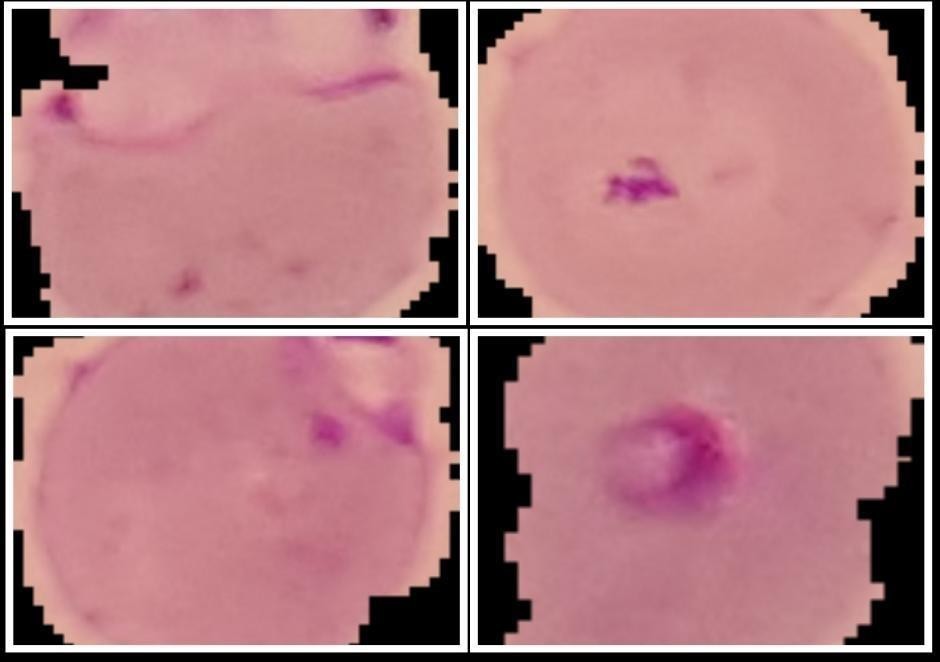
Step 10: Stop here if the precision is good and calculate the exactness rate.

### Dataset

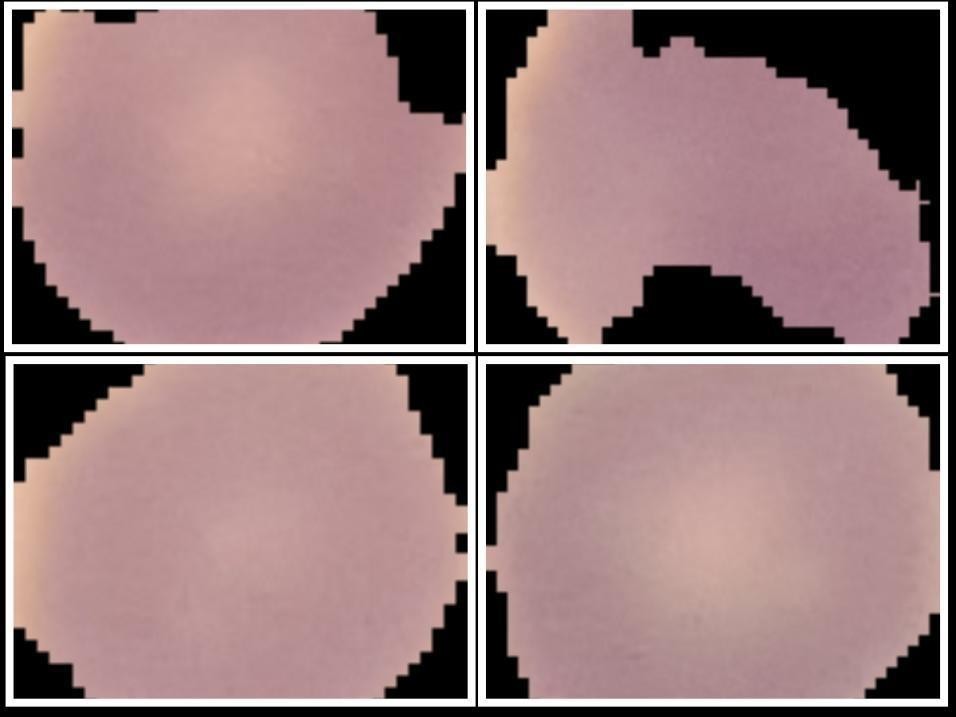
Malaria detection using deep learning is an active area of research aimed at developing automated systems for the accurate and efficient diagnosis of malaria from microscopic images of blood smears. The goal is to leverage the power of deep learning algorithms to detect malaria parasites in blood cells and identify the disease accurately.There are several publicly available datasets for malaria detection using deep learning, including the following:

Malaria Cell Dataset: This dataset contains about 27,558 images of blood smears collected from 150 different patients. The images are labeled as either infected or uninfected.

Malaria Image Classification Dataset: This dataset contains about 13,780 images of blood smears collected from different sources. The images are labeled as either infected or uninfected.



**Fig 5.1 (a):** Infected Image



**Fig 5.1 (b):** Uninfected Image

## Feature Extraction C onvolution

In this layer the features of the image are get extracted. First of all, a part of image is connected to Convolution layer to perform operation ,calculating the dot product between receptive field and the filter We will repeat the same process again and again until we go through the whole image. The output will be the input for the next layer. Convolution layer also contains Rectified Linear Activation Function (ReLU) activation to make all negative value to zero.

### Pooling

Pooling layer is used to reduce the spatial volume of input image after convolution. It is used between two convolution layer. If we apply FC after Convolution layer without applying pooling or max pooling, then it will be computationally expensive and we don’t want it. So, the max pooling is only way to reduce the spatial volume of input image. There is no parameter in pooling layer but it has two hyper parameters — Filter(F) and Stride(S).

### Activation function (Relu)

The Rectified Linear Activation Function (ReLU), it is a piece wise linear function that will output the input directly if it is positive, otherwise it will output zero. It has become the default activation function for many types of neural networks because a model that uses it is easier to train and often achieves better performance.

### ReLU = Max(0, X)

**Classifier (softmax)**

The classifier uses the cross-entropy loss. The Softmax classifier gets its name from the softmax function, which is used to squash the raw class scores into normalized positive values that sum to one, so that the cross-entropy loss can be applied.

## Performance metrics

**ACCURACY:**

Accuracy is one metric for evaluating classification models.Accuracy is a measure of how well a deep learning model is able to correctly predict the outcome of a task based on the input data.

The formula for accuracy is:

Accuracy = (Number of correctly predicted examples) / (Total number of examples)

A Sequential CNN model is trained to detect the presence of malaria parasites in blood smear images. The model is trained on a dataset of 27,558 images. The model predicted 27,053 images correctly. The model got an accuracy of 98.17 for 20 epochs.

Accuracy = (27,053) / (27,558)

= 98.17%

Therefore, the accuracy of the Sequential CNN model for malaria detection is 98.17% for 20 epochs.

**Model Loss :**

Model Loss is one metric for evaluating classification models.Model loss, also known as training loss, is a measure of how well a machine learning model is able to predict the target variable on the training data. It represents the difference between the predicted output and the actual output for each training example and is used as a measure of how well the model is fitting the training data. In the case of a Sequential CNN model, the model loss is typically calculated using cross-entropy loss. The formula for cross-entropy loss is:

loss = -1/n \* Σ [ y \* log(y\_pred) + (1-y) \* log(1-y\_pred) ]

Where:

n is the number of training examples

y is the true label (either 0 or 1)

y\_pred is the predicted label (a value between 0 and 1)

When trained the algorithm the model loss is 0.532 for 20 times(epochs=20).



**Fig 5.3:** Sequential CNN Diagram

# TESTING

Testing is the process of finding differences between the expected behaviour specified by system models and the observed behaviour of the system. Testing is a critical role in Quality assurance and ensuring the reliability of development and these errors will be reflected in the code, so the application should be thoroughly tested and validated. Testing with respect to the project Automated Library Book using Face Recognition is mainly testing for storing the details and their facial features of the student that will be tracked and trained by the admin, it stores the timestamp of the student i.e. in time and out time will be stored in the database and reports will be generated .

## Testing Activities

Testing a large system is a complex activity and like any complex activity. It has to be broke into smaller activities. Thus, incremental testing was performed on the project i.e., components and subsystems of the system were tested separately before integrating them to form the subsystem for system testing.

# Testing Types

## Unit Testing

Unit testing focuses on the building blocks of the software system that is the objects and subsystems. In this testing, we check if the individual components are working or not, for example focus first on modules to locate errors. These errors are corrected and verified, to ensure the code is perfect.

## Integration Testing

Integration testing detects faults that have not been detected. During unit testing by focusing on small groups on components two or more components, which are integrated and tested and once tests do not reveal any new faults; additional components are added to the group. I have used the following approach to implements and integrated testing. Top- down testing strategy unit tests the components of the top layer and then integrated the components of the next layer down. When all components of the new layer have been tested together, the next layer is selected. This was repeated until all layers are combined and

involved in the test. In this testing all the modules on which unit testing is performed are integrated together and tested.

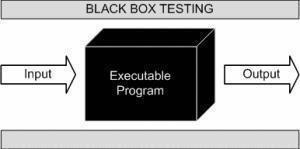
## 6.2.3 Validation Testing:

The systems completely assembled as package, the interfacing have been uncovered and corrected, and a final series of software tests are validation testing. The validation testing is nothing but validation success when system functions in a manner that can be reasonably expected by the customer. The system validation had done by series of Black-box test methods.

## Black Box Testing

Also known as Behavioral Testing, is a software testing method in which the internal structure/design/implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional. This is the final testing with respect to our project. After the unit testing and integration testing we implement black box testing for the final testing where we test for the final output that is expected out of the developed system. This testing is done irrespective of the knowledge of the internal system by the person doing the testing. This method attempts to find errors in the following categories:

* + - 1. Incorrect or missing functions
      2. Interface errors
      3. Errors in data structures or external database access
      4. Behaviour or performance errors
      5. Initialization and termination errors



**Fig.6.2.4:** Black box testing

* 1. **Test cases**

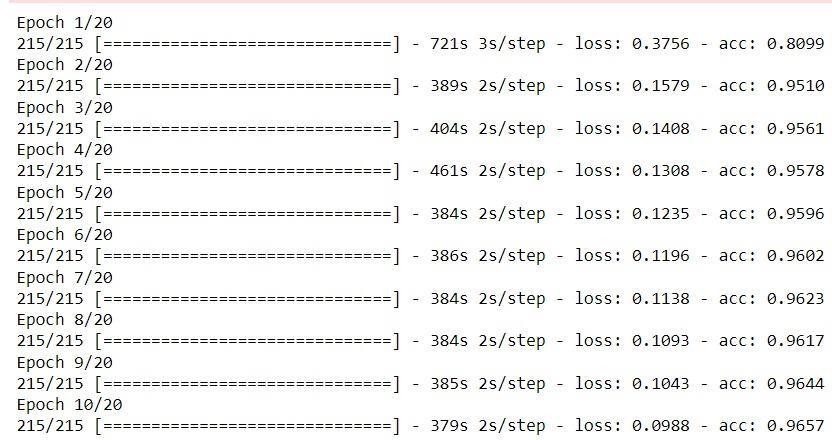
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S.NO | Test Case | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Load | Loading the image on the website | Image has to be Loaded | Image is loaded on website | PASS |
| 2 | Image consists of Malaria | The image consists of spots of malaria in a coloured way | Malaria is present | Malaria is detected in image | PASS |
| 3 | Image does not consists of Malaria | The image is plain & does not consists of any spots | Malaria is not present | Malaria is not present in the image is detected. | PASS |

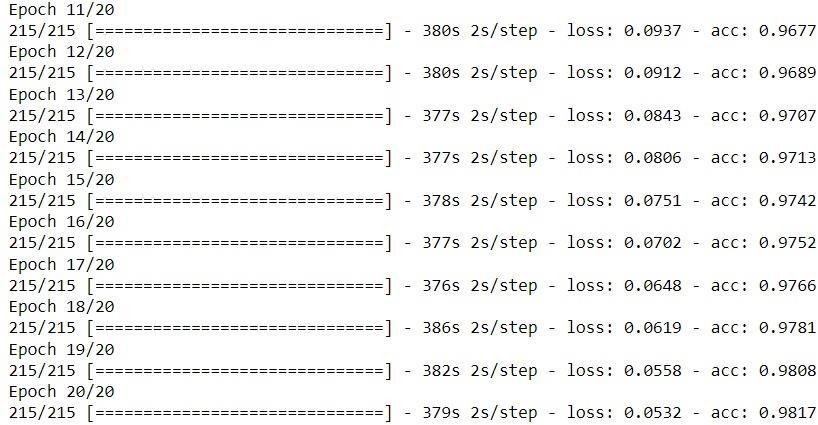
**Table 6.3 :** Tase Cases Table

**7.RESULTS**

# Epoch:

1 epoch is when an entire dataset is passed forward and backward through the neural network only once. Since, 1 epoch is too big to fed to the computer at once we divide it in several smaller batches. The model give the 98.17% accuracy while training the model. Below are 20 epochs of training process with batch size of 128 images.





**Fig 7.1 :** 20 Epoch

# 7.2 Graphs:

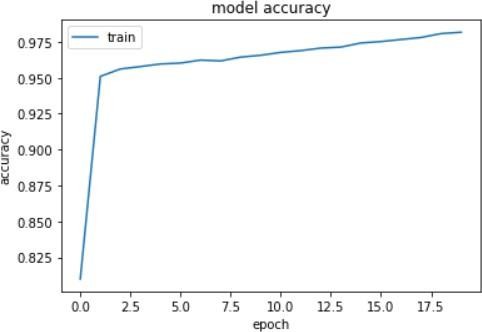
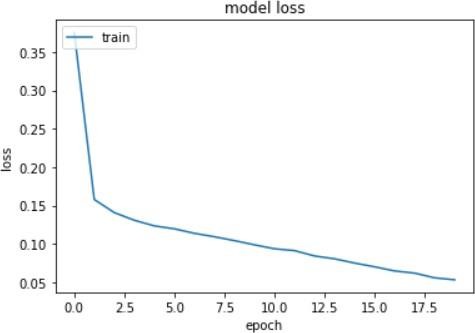
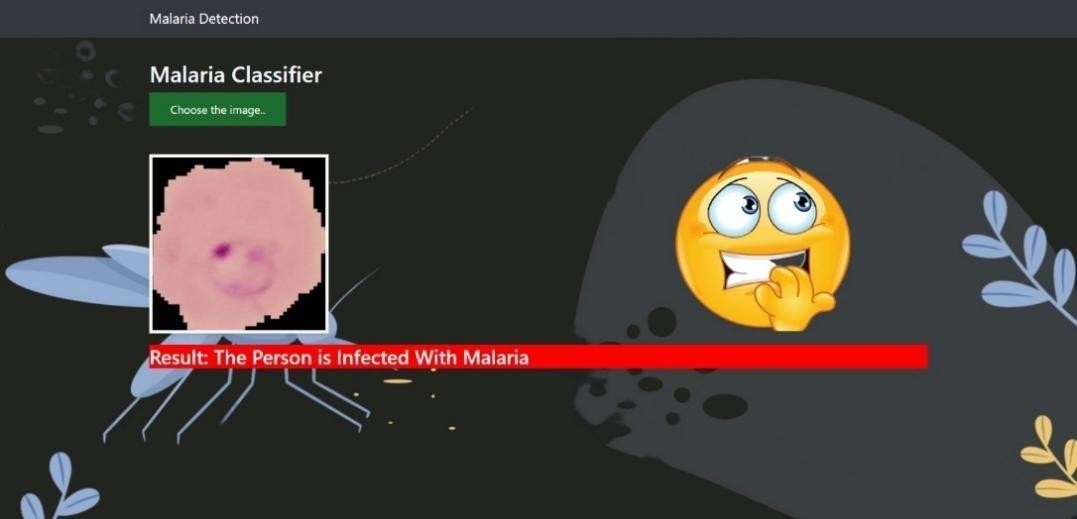


Fig 7.2 (a): Model Accuracy Graph

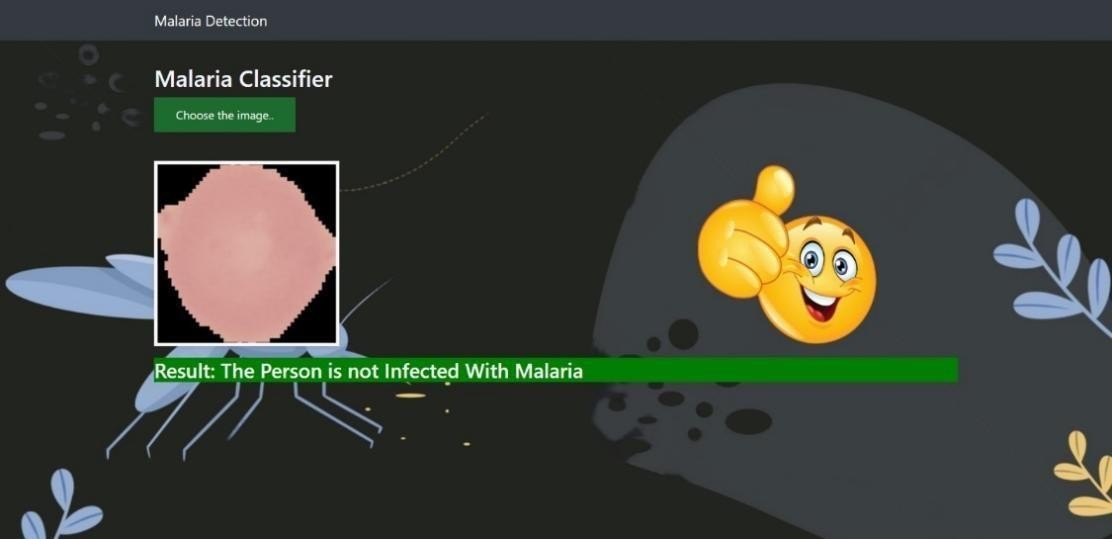


**Fig 7.2 (b):** Model Loss Graph

**7.3 UI Design Result:**



**Fig 7.3 (a):** Malaria Infected Image Prediction



**Fig 7.3 (b):** Malaria Uninfected Image Prediction

# 8.SAMPLE CODE

## 

## Python code

import os

import numpy as np

from sklearn.metrics import classification\_report, confusion\_matrix import seaborn as sn

from glob import glob

os.chdir(r'D:\Malaria\cell\_images\Dataset\\') print(os.path.join(os.getcwd()))

Parasitized = os.path.join(r'D:/Malaria/cell\_images/Dataset/train/Parasitized') Uninfected=os.path.join(r'D:/Malaria/cell\_images/Dataset/train/Uninfected') print(Parasitized)

print(Uninfected)

print('total Parasitized images:', len(os.listdir(Parasitized))) print('total Uninfected images:', len(os.listdir(Uninfected)))

import matplotlib.pyplot as plt import matplotlib.image as mpimg batch\_size = 128

from tensorflow.keras.preprocessing.image import ImageDataGenerator # All images will be rescaled by 1./255

train\_datagen = ImageDataGenerator(rescale=1/255)

# Flow training images in batches of 128 using train\_datagen generator train\_generator = train\_datagen.flow\_from\_directory(

'train/', # This is the source directory for training images target\_size=(200, 200), # All images will be resized to 200 x 200 batch\_size=batch\_size,

# Specify the classes explicitly classes = ['Parasitized','Uninfected'],

# Since we use categorical\_crossentropy loss, we need categorical labels class\_mode='categorical')

import tensorflow as tf

model = tf.keras.models.Sequential([

# Note the input shape is the desired size of the image 200x 200 with 3 bytes color # The first convolution

tf.keras.layers.Conv2D(16, (3,3), activation='relu', input\_shape=(200, 200, 3)),

tf.keras.layers.MaxPooling2D(2, 2), # The second convolution

tf.keras.layers.Conv2D(32, (3,3), activation='relu'), tf.keras.layers.MaxPooling2D(2,2),

# The third convolution tf.keras.layers.Conv2D(64, (3,3), activation='relu'), tf.keras.layers.MaxPooling2D(2,2),

# The fourth convolution tf.keras.layers.Conv2D(64, (3,3), activation='relu'), tf.keras.layers.MaxPooling2D(2,2),

# The fifth convolution

tf.keras.layers.Conv2D(64, (3,3), activation='relu'), tf.keras.layers.MaxPooling2D(2,2),

# Flatten the results to feed into a dense layer tf.keras.layers.Flatten(),

# 128 neuron in the fully-connected layer tf.keras.layers.Dense(128, activation='relu'),

# 5 output neurons for 4 classes with the softmax activation tf.keras.layers.Dense(2, activation='softmax')])

model.summary()

from tensorflow.keras.optimizers import RMSprop model.compile(loss='categorical\_crossentropy', optimizer=RMSprop(lr=0.001),

metrics=['acc']) total\_sample=train\_generator.n n\_epochs = 10

history = model.fit\_generator( train\_generator,

steps\_per\_epoch=int(total\_sample/batch\_size), epochs=n\_epochs,

verbose=1)

from keras.models import model\_from\_json

model\_json = model.to\_json()

with open("model.json", "w") as json\_file: json\_file.write(model\_json)

# serialize weights to HDF5 model.save\_weights("model.h5")

print("Saved model to disk") json\_file = open('model.json', 'r') loaded\_model\_json = json\_file.read() json\_file.close()

loaded\_model = model\_from\_json(loaded\_model\_json) # load weights into new model loaded\_model.load\_weights("model.h5") print("Loaded model from disk") loaded\_model.summary()

# from keras.preprocessing.image import load\_img, img\_to\_array, ImageDataGenerator from tensorflow.keras.utils import load\_img, img\_to\_array

from keras.applications.vgg16 import preprocess\_input my\_image=0

#load the image

# my\_image = load\_img('train/Parasitized/C33P1thinF\_IMG\_20150619\_114756a\_cell\_179.png', target\_size=(200, 200))

my\_image = load\_img(r"D:\Malaria\cell\_images\train\Parasitized\C33P1thinF\_IMG\_20150619\_114756a\_cell

\_17

9.png", target\_size=(200, 200)) # my\_image =

load\_img(r"D:\Malaria\cell\_images\train\Uninfected\C1\_thinF\_IMG\_20150604\_104722\_cell\_9.p ng” target\_size=(200, 200))

#preprocess the image

my\_image2 = img\_to\_array(my\_image)

my\_image2 = my\_image2.reshape((1, my\_image2.shape[0], my\_image2.shape[1], my\_image2.shape[2]))

my\_image2 = preprocess\_input(my\_image2) result=loaded\_model.predict(my\_image2) import numpy as np class\_value=np.argmax(result)

if class\_value == 0:

print("Image is infected with Malaria") else:

print("Image is unifected") from PIL import Image #creating a object

im=Image.open(r'D:\Malaria\cell\_images\train\Parasitized\C33P1thinF\_IMG\_2015 0619\_114756a\_cell\_179.png')

im.show()

## Index.html

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

<h2 style="color:aliceblue">Malaria Classifier</h2>

<div>

<form id="upload-file" method="post" enctype="multipart/form-data">

<label for="imageUpload" class="upload-label"> Choose the image..

</label>

<input type="file" name="file" id="imageUpload" accept=".png, .jpg, .jpeg">

</form>

<div class="image-section" style="display:none;">

<div class="img-preview">

<div id="imagePreview">

</div>

</div>

<div style="height: 250px; width: 250px;" id="parent-emoji">

<div id="emoji-1" style="height: 250px; width: 250px;">

</div>

<div id="emoji-2" style="height: 250px; width: 250px;">

</div>

</div>

</div>

<div >

<button type="button" class="btn btn-primary btn-lg " id="btn-predict" style="display:none;" >Predict!</button>

</div>

<div class="loader" style="display:none;"></div>

<h3 id="result">

<span> </span>

</h3>

</div>

{% endblock %}

**Base.html**

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Malaria Detection</title>

<link href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css" rel="stylesheet">

<script src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"></script>

<script src="https://cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></script>

<script src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>

<link href="{{ url\_for('static', filename='css/Ashok14.css') }}" rel="stylesheet">

</head>

<body>

<nav class="navbar navbar-dark bg-dark">

<div class="container">

<a class="navbar-brand" href="#">Malaria Detection</a>

</div>

</nav>

<div class="container">

<div id="content" style="margin-top:2em">{% block content %}{% endblock %}</div>

</div>

</body>

<footer>

<script src="{{ url\_for('static', filename='js/mains13.js') }}" type="text/javascript"></script>

</footer>

</html>

## Flask code

from \_future\_ import division, print\_function # coding=utf-8

import sys import os import glob import re

import numpy as np

# Keras

from tensorflow.keras.applications.imagenet\_utils import preprocess\_input, decode\_predictions

from tensorflow.keras.models import load\_model

from tensorflow.keras.preprocessing import image from keras.models import model\_from\_json

# from tensorflow.keras.utils import load\_image, image\_to\_array from keras.applications.vgg16 import preprocess\_input

# Flask utils

from flask import Flask, redirect, url\_for, request, render\_template from werkzeug.utils import secure\_filename

#from gevent.pywsgi import WSGIServer

# Define a flask app app = Flask(\_name\_)

MODEL\_PATH =r"D:\Malaria\cell\_images\Dataset\model.h5" mmodel\_json = r"D:\Malaria\cell\_images\Dataset\model.json"

# Load your trained model

# model = load\_model(MODEL\_PATH)

json\_file = open(mmodel\_json, 'r', encoding='utf-8') loaded\_model\_json = json\_file.read() json\_file.close()

loaded\_model = model\_from\_json(loaded\_model\_json)

# load weights into new model

model = loaded\_model.load\_weights(MODEL\_PATH)

def model\_predict(img\_path, model): my\_image=0

my\_image = image.load\_img(img\_path, target\_size=(200, 200))

# Preprocessing the image

my\_image2 = image.img\_to\_array(my\_image)

my\_image2 = my\_image2.reshape((1, my\_image2.shape[0], my\_image2.shape[1], my\_image2.shape[2]))

my\_image2 = preprocess\_input(my\_image2)

result=loaded\_model.predict(my\_image2) class\_value=np.argmax(result)

if class\_value==0:

preds="The Person is Infected With Malaria" else:

preds="The Person is not Infected With Malaria"

return preds

@app.route('/', methods=['GET']) def index():

# Main page

return render\_template('index.html')

@app.route('/predict', methods=['GET', 'POST']) def upload():

if request.method == 'POST':

# Get the file from post request f = request.files['file']

# Save the file to ./uploads basepath = os.path.dirname(\_file\_) file\_path = os.path.join(

basepath, 'uploads', secure\_filename(f.filename)) f.save(file\_path)

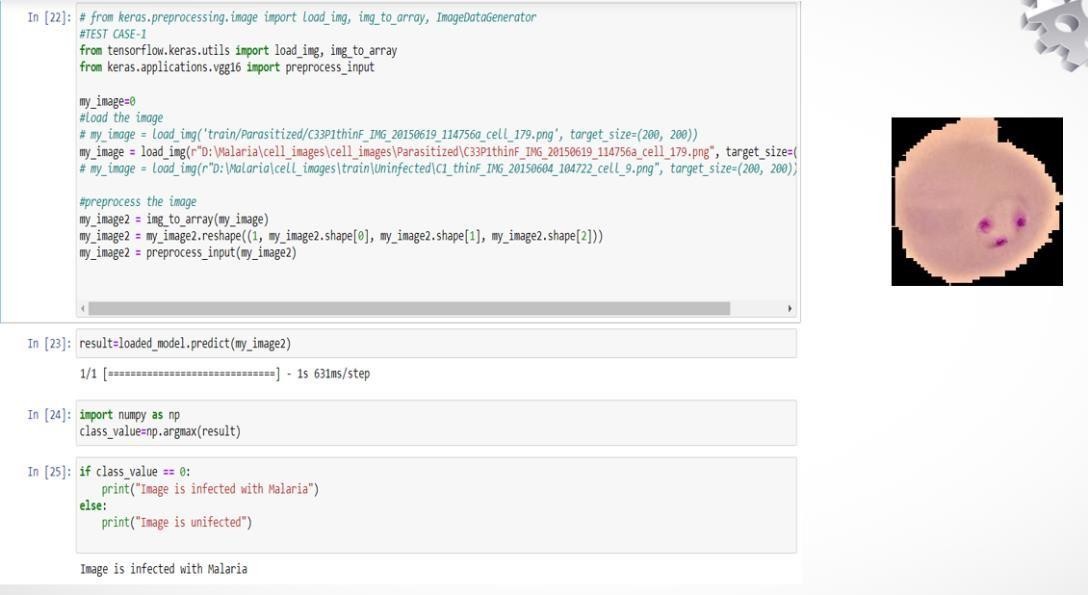
# Make prediction

preds = model\_predict(file\_path, model) result=preds

return result return None

if \_name\_ == '\_main\_': app.run(debug=True)

* 1. **Test case Image results**



**Fig 8.3 (a):** Test Case-1



**Fig 8.3 (b):** Test Case-2



**Fig 8.3 (c):** Test Case-3



**Fig 8.3 (d):** Test Case-4

# 

# 9.CONCLUSION

Malaria is a deadly disease that has taken countless lives and is on a verge to take more. It not only affects humans but also affects a lot of organisms. Malaria detected from the traditional method that is bringing the samples and analyzing cell growth requires more time. So in the proposed work, a deep-learning model has been constructed to predict Malaria with a high accuracy rate and low time duration. The Fine-Tuned CNN provided a high accuracy rate of 98.17 on running for 20 epochs and got a model loss of 0.53. Futher more we have developed a web based GUI model to utilize our deep learning based Malaria Detection model in which a user will be giving a image which is analyzed the result whether Malaria is present in that image or not is given.

# 10.REFERENCE

1. Shuying Liu and Weihong Deng, "Exceptionally profound convolutional neural organization based picture arrangement with little preparing test size," in "Extremely profound convolutional neural organization based picture order with little preparing test size," in "profound convolutional neural organization based picture grouping with little preparing test size," in " The third Asian Conference on Pattern Recognition, coordinated by the International Association for Pattern Recognition (IAPR), occurred in 2015. (ACPR).
2. Nima Hatami, Yann Gavet, and Johan Debayale, "Grouping of time-series pictures utilizing profound convolutional neural organizations." The ICMV 2017 is the 10th worldwide meeting on machine vision (Vol.10696).
3. "Mechanized picture preparing method for the location and classification of jungle fever on meager blood spreads," Nicholas E. Ross, Charles J. Pitchard, David M. Rubin, and Adriano G. Duse. Vol.44 of the IEEE(2020).
4. K.M. Khatri, V.R. Ratnaparkhe, S.S. Agarwal, and A.S. Bhalchandra, "Picture handling strategy for malarial parasite recognizable proof," International Journal of Computer Application, vol (IJCA) (2014).
5. Vishnu V. Makkapati and Raghuveer M. Rao, IEEE International Conference on Acoustics, Speech and Signal Processing, "Division of malarial parasites in fringe blood smear pictures." (2020)
6. Corentin Dallet, Saumya Kareem, and Izzet Kale, "Ongoing blood picture preparing application for jungle fever location using cell phones," IEEE International Symposium on Circuits and Systems (ISCAS) (2014).
7. Zhaoui Liang, Andrew Powell, Ilker Ersoy, Mahdieh Poostchi, KamolrautSilmaut, and Kannapan Palani, "CNN-based picture investigation for jungle fever finding," IEEE International Conference on Bioinformatics and Biomedicine (BIBM) (2022).
8. Yuhang Dong, Zhuocheng Jiang, HongdaShen,W. David Pan, Lance A. Williams, and Vishnu V. B. Reddy, "Assessments of profound convolutional neural organization for mechanized discovery of jungle fever contaminated cells," IEEE EMBS International Conference on Biomedical and Health Informatics (BHI) (2017).
9. GorthiSai Siva, Gopakumar, and MuraliSwetha "From a pile of blood smear pictures got utilizing a specially fabricated slide scanner, convolutional neural organization based jungle fever determination" Journal of Biophontics, Vol. 11 No.3 (2017).
10. JutamaadSatayavivad, MathurosRuchirawat, and Peter Gascoyne Acta Tropica, Vol. 89, No. 3, "Microfludic strategies to intestinal sickness identification." (2019)
11. A. Vadavalli and R. Subhashini, "Profound Learning based truth revelation technique for exploring the credibility of a given book corpus," International Journal of Recent Technology and Engineering, 2019.
12. R. Subhashini, T.N.R. Nithin, and U.M.S. Koushik, "Diabetic Retinopathy Detection Using Image Processing (GUI)", International Journal of Recent Technology and Engineering, 2019.
13. MadhuKeerthana, Y., and BevishJinila., "A Review on Rough Set Theory in Medical Images," Research Journal of Pharmaceutical, Biological, and Chemical Sciences, Vol.7, Issue 1, pp.815-822, ISSN: 0975-8585. (2016)