A PROJECT REPORT

on

"HEALTH DISCERNMENT SYSTEM"

Submitted to

KIIT Deemed to be University

In Partial Fulfillment of the Requirement for the Award of

BACHELOR'S DEGREE IN INFORMATION TECHNOLOGY

BY

ANISH KUMAR- 1706200
INDRILA BASAK- 1706224
PRANAV JAIN- 1706243
RITAM BARIK- 1706254
RITURAJ SAHA- 1706256

UNDER THE GUIDANCE OF

DR. ALEENA SWETAPADMA

SCHOOL OF COMPUTER ENGINEERING
KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY
BHUBANESWAR, ODISHA - 751024

May 2020

KIIT Deemed to be University

School of Computer Engineering
Bhubaneswar, ODISHA 751024

CERTIFICATE

This is to certify that the project entitled

"HEALTH DISCERNMENT SYSTEM"

submitted by

ANISH KUMAR -	1706200
INDRILA BASAK-	1706224
PRANAV JAIN -	1706243
RITAM BARIK -	1706254
RITURAJ SAHA -	1706256

is a record of bonafide work carried out by them, in the partial fulfillment of the requirement for the award of Degree of Bachelor of Engineering (Computer Science & Engineering OR Information Technology) at KIIT Deemed to be University, Bhubaneswar. This work is done during the year 2019-2020, under my guidance.

Date: 08/06/2020

(Dr. Aleena Swetapadma)
Project Guide

ACKNOWLEDGEMENTS

We are profoundly grateful to Dr ALEENA SWETAPADMA for her expert guidance and continuous encouragement throughout to see that this project reaches its target since its commencement to its completion.

ANISH KUMAR
INDRILA BASAK
PRANAV JAIN
RITAM BARIK
RITURAJ SAHA

ABSTRACT

With broad data development in biomedical and healthcare sectors, detailed analyzes of medical data support early detection of illness, patient care and community services. However, the quality of the study is lowered when the content of the medical data is incomplete. Also, various regions exhibit unique features of certain regional diseases. This can hinder disease outbreak forecasting. In this project, we streamline deep learning algorithms to effectively predict chronic disease outbreaks in populations with recurrent diseases. The diagnosis of diseases is a critical and central aspect of medicinal science. Doctors breakdown side effects in the human body more often than not to foresee diseases. In recent times, numerous research strategies have been used with a specific goal to make it more accurate. This system will help to predict the medical results efficiently. In this system, we will provide a user-friendly interface that can be used by the users to detect whether their medical test results are positive or normal, i.e. it will detect the disease.

There is a great growing interest in the domain of deep learning techniques for identifying and classifying images with various dataset. This deep learning project is based on a user interface and its application of the Health Discernment System in real life. It will also describe how the system will perform and under what it must operate. In this document, the user interface will also be shown. Both the stakeholders(users) and the developers of the interface can benefit from this approach.

Keywords: Disease Detection, Feature Selection, Convolutional Neural Network, Deep Learning, Tensorflow.

Contents

List of Figures

Chapter 1

Introduction

The days are long gone when data on health-care used to be small. The advancement level in devices for the acquisition of images is quite large and that is what makes image processing difficult and fascinating. This significant growth of medical images and techniques requires comprehensive and exhaustive efforts from a medical professional who is susceptible to human error and the result can also vary widely among various experts. The alternative to this approach is to use machine learning or deep learning strategies for automating the detection process of various diseases.

Machine Learning (ML) and Artificial Intelligence (AI) have made significant progress over the past few years. ML and AI techniques have influenced medical fields such as medical image processing, image recognition, computer-aided diagnosis, image segmentation, and image fusion to name a few. While automated disease detection based on conventional medical imaging methods demonstrated significant accuracies for decades, breakthroughs in machine learning approaches have sparked a growth in deep learning. Deep learning-based algorithms demonstrated remarkable outcomes in various fields such as computer-aided diagnosis, speech recognition etc.

For our project, we have used the above stated idea behind disease detection, to construct a system using Convolutional Neural Network that detects the diseases quickly and also guarantees it to be free of error. By doing so we meant to minimize the human efforts that are required to detect a medical test report. We have tried to make the system user-friendly with the help of GUI, so that it can be used not only by the medical professionals but also by the population at large.

1.1 MOTIVATION

Disease detection plays a very important role in the process of diagnosis. So our motive is to classify and thus detect the diseases based on the medical test images. By doing this we want to minimize the chances of errors that generally happens due to the doctor's misjudgment.

Therefore, we built a system that will not only help in detecting the diseases efficiently but also save the time and effort of the medical practitioners. It will also save the patients from running to the doctor to get their medical reports verified.

Chapter 2

Literature Survey

2.1 BREAST CANCER

There are various ways of detecting breast cancer including mammography, MRI scans, computed tomography (CT) scans, ultrasound, and nuclear imaging. Although, none of these approaches provides a perfectly accurate cancer prediction. Tissue-based diagnosis is done primarily using a method of staining. Some staining elements, usually hematoxylin and eosin (H&E), is being used to colour elements of tissues in this method. Accordingly, cell structures, types and other foreign elements are stained and easily identifiable in high resolution. Pathologists then analyze the stained tissue slides under a microscope or use images taken from the camera in high resolution. A histopathology test is important for the identification of tumours. It is an old method for predicting invasive cancer cells from stained H&E tissues. There are different weaknesses in this procedure because it includes intra-observer variation, cancer cells and tissues can also have multiple appearances, and many other cell figures have the same hyperchromatic characteristics that make identification difficult. The selection of area is also a consideration as the procedure is conducted only on a specific tissue region, so the area chosen should be in the periphery of the tumour. Using deep learning techniques one can solve the aforementioned problems. Deep learning is a popular subcategory of machine learning technology that is inspired by the functioning of the human brain to examine unstructured patterns. Deep learning models have a high chance of success as they train on representations in the hierarchy. They can also extract and organize unique attributes, and therefore do not require any prior knowledge of the domain. On the other side, trivial methods require rigorous feature engineering to acquire features which involve expertise in the domain. Many methods of deep learning have been proposed for predicting the tumour class. These are mostly binary classification [1,2] but some have used multi-variable classification [3]. Deep learning algorithms just need the data in the correct format and some suitable network parameters for the problem. Pre-designed networks such as AlexNet, MobileNet, Inception and many more can also be used [4]. Different scholars have proposed different methods and manual networks for classifying breast cancer besides the pre-designed networks mentioned above.

Artificial neural networks rely, for example, on MLE (Maximum Likelihood Estimation) [5]. RBF Neural Networks on paper [6], the GRU-SVM model which is an ML algorithm coupled with a type of recurrent neural network (RNN) and gated recurrent unit (GRU) with support vector machine (SVM) [7]. Other scholars have developed methodologies including these techniques to achieve better results with less computational complexity. To reduce the size of the input feature, Karabatak et al. have proposed the AR + NN method which reduces the number of features by implementing association rules [8]. A combination of NN and multivariate adaptive regression splines (MARS) is also used for cancer detection [9]. Another method is the Fuzzy-artificial immune system and the K-NN algorithm listed in Ref. [10]. Descriptors like CLBP, GLCM, LBP, LPQ, ORB, PFTAS are defined in paper [11] with breast cancer classification up to 85.1% accuracy. As with the BreakHis data set released in 2015, this has only been used by some scholars. For example, Fabio A. Spanhol [12] describes parameters and network configuration that has been accurate between 80 and 85%. The proposed method mentioned herein further reinforces this. Also, we present summaries of other methods along with their accuracies in the Discussion section. A series of tasks are implemented in deep learning algorithms. The first step is the preprocessing of images which is necessary to translate data into the format in which it can be input directly into the network. This step involves multiple image channeling, and then segmentation [13] is done (only if required, e.g. where regions of interest need to be separated from the background or parts which are not needed for training are omitted). At this point, data is ready for use in training, either in a supervised way or in an unsupervised way. The next step is feature extraction. Features represent the image's visual content for histopathology. In the case of supervised extraction of features, the features are known and various methods are used to discover them [14, 15, 16], but in the case of unsupervised feature extraction methods, features are not known and implicitly acquired through the Convolutional Neural Network (CNN) in the proposed solutions. The last phase is classification, which places an image in the respective category (benign or malignant) and can be done utilizing SVM (support vector machine) or using an activation function such as Softmax with a fully connected layer

2.2 PNEUMONIA

Recent advances in deep learning models and the access to large datasets have enabled algorithms to outshine medical workers in various diagnostic imaging tasks, such as detection of skin cancer [17], haemorrhage identification [18], arrhythmia detection [19], and diabetic retinopathy detection [20]. Automated diagnosis enabled by chest X-rays has taken on huge interest. These algorithms are progressively used to detect pulmonary lung nodule [21] and pulmonary tuberculosis classification [22]. The performance of many convolutional models on various abnormalities relying on the OpenI database available to the public [23] discovered that the same deep convolutional network architecture

doesn’t well work across all abnormalities [24], ensemble models dramatically improved classification accuracy compared to a single model, and finally, the deep learning method improved accuracy compared to rule-based approaches.
Statistical dependence amongst labels [25] was studied to arrive at more accurate predictions, thereby outshining other approaches on given 13 images that were selected from 14 classes [26]. Algorithms for mining and predicting labels originating from radiology images were reviewed, as were studies [27–29], however, the image labels were usually limited to disease tags, and therefore lacked contextual information. Disease detection from X-ray images was investigated [30–32], classifications on chest X-ray image views were performed [33], and segmentation of body parts from chest X-ray images and computed tomography was done [29, 34]. Conversely, the learning of image features using text and the creation of image descriptions are yet to be explored concerning what a person would say.

2.3 MALARIA

Malaria is commonly diagnosed by microscopic blood cell analysis using blood films[6]. Nearly 167 million blood films had been tested for malaria using microscopy during 2010 which was less expensive and less complex than the diagnosis based on polymerase chain reaction [36]. Though widely used, a microscopic diagnosis has many drawbacks like- malaria is generally linked to poverty and mostly arises in low-economic countries [37], where most laboratories or diagnostic facilities do not have standard testing facilities. Also, the diagnosis depends on the individual's ability to examine the blood film and the level of the parasites present thereon. Also, the monotonicity of the examination greatly influences the quality of the examination, especially towards the later part of a batch, when the lot has many specimens. Global pathology shortage [38] in general, seriously impacts the health care system in developing nations and malaria is no exception. Several Bangladeshi citizens opt for treatment abroad because of the lack of dependable diagnostic facilities [39] that is sadly not financially viable for the majority of people. Today's modern computer-aided systems utilise deep learning algorithms to analyze medical images [40]. There is a development around the world to simplify the diagnostic method with the assistance of various machine learning techniques to help human specialists make the right diagnosis. Liang et al . have proposed an approach based on deep learning for classifying cells infected with malaria from red blood smears. Their proposed method is based on a 16-layer convolutional neural network that uses the AlexNet architecture[41] pre-trained on the CIFAR-100 [42] dataset to outshine their transfer learning-based model. Dong et al. [43] used a dataset consisting of only about 1000 samples of training and testing and has used transfer learning and reported the outcomes on LeNet[44], AlexNet[45] and GoogleNet[46] architectures.

Jane et. al [47] developed a different, object detection based approach and used a Faster Region-based Convolutional Neural Network (Faster R-CNN) that was pre-trained on Imagenet [48] and fine-tuned on their dataset. Bibin et. al [49] proposed a classic approach on Deep Belief Network (DBN) [50] comprising of 4 hidden layers pre-trained by stacking restricted Boltzmann machines [51] using contrastive divergence method [52] for pre-training. Razzak et. al [53] have put forward an automated process that recognizes both the tasks of malaria parasite segmentation as well as classification. Their segmentation network is composed of a Deep Aware CNN [54], and the classification network uses an Extreme Learning Machine (ELM) approach [55]. In several works [56][57][58], the Convolutional Neural Networks were used to diagnose malaria parasites from microscopic images. Also, various approaches were proposed by Shen et. al [59] as well as Mohanty et. al [60] that employ unsupervised machine learning approaches using stacked auto-encoders to automatically learn the features of infected and uninfected cell input images. Mehanian et. al [56] have used a series of computer vision techniques, such as global white balance and adaptive nonlinear grayscale to present a novel augmentation scheme showing state-of-the-art performance in the evaluation of automatic diagnostic methods. Since medical images/datasets are typically smaller in size and thus often deemed inadequate for learning, the strength of transfer learning has also been exploited in the literature. In the literature, Var et. al [57] and Rajaraman et. al [58] suggested methods for computer-aided diagnosis based on pre-trained convolutional neural networks to identify parasites of malaria as feature extractors. In this classification task, classical machine learning algorithms also did well. Bayesian learning, supporting vector machines, logistic regression, and k-nearest neighbour algorithm perform well in this context, as shown by Das et. al [61] and Park et. al [62]. Furthermore, attempts have been made to remove the stains from the peripheral blood smear images as well as to reduce impulse noise.

2.4 SKIN CANCER

Deep learning algorithms have recently gained huge success in various computer vision issues. Krizhevsky et al. in 2012 [63]inbuilt a novel technique (AlexNet) using convolutional neural networks for classifying a large data (1,2 million images) containing 1000 categories of objects in the 2010 ImageNet Large Scale Visual Recognition Challenge (ILSVRC2010) and delivers the highest result and, therefore, tremendous interest among academics in the field of computer vision. [68] Esteva et al. made significant progress on the classification of skin cancer through a pre-trained model of GoogleNet Inception v3 CNN to categorize 129,450 clinical images of skin cancer including 3,374 dermatoscopic images. Yu et al. [64] developed a convolutional neural network for the classification of malignant melanoma with over 50 layers on ISBI 2016 challenge dataset. Haenssle et al., in 2018 [65] used a deep, convolutional neural network to identify the binary diagnostic group of melanocytic images dermoscopy.

Dorj et al. [67] developed ECOC SVM with such a deep, convolutional neural network approach to classify 4 diagnostic categories of images of clinical skin cancer.[66]The clinical images of 12 skin diseases were classified by a deep convolutional neural network by Han et al. To our best understanding in the prior studies, we found no work on the use of skin cancer dermoscopy dataset with ample diversity for the classification of dermatoscopic skin cancer images in multiclass.

Chapter 3

Software Requirements Specification

3.1 INTRODUCTION

3.1.1 DOCUMENT PURPOSE

This document presents a detailed explanation of the objectives, features, user interface and application of the Health Discernment System in real life. It will also describe how the system will perform and under what it must operate. In this document, the user interface will also be shown. Both the stakeholders(users) and the developers of the interface can benefit from this document.

3.1.2 PRODUCT SCOPE

This system will help to detect the medical results efficiently. In this detecting system, we will provide a user-friendly interface that can be used by the users to detect whether their medical test results are positive or normal, i.e. it will detect the disease. This will ultimately help the patients to save time as they won't have to run to the doctor just to know what their medical reports say.

In case the test results are negative, it will greatly help the patient to save money, as there won't be any requirement to visit the doctor again. Decisions are often made based on the doctor's intuition and experience and sometimes that may not be completely correct. In this interface the predictions will be free of unwanted biases and errors- so it will be completely trustworthy. The doctors can also use this system to predict the results better. The lab technicians and the other health-care professionals can also use this predicting system to guide the people.

3.1.3 INTENDED AUDIENCE AND DOCUMENT OVERVIEW

This document is intended for different types of readers such as system designer, system developer and tester. By reading this document a reader can learn about what the project is implemented for and how it will present it's basic ideas.

This document has a sequential overview of the whole project so if a reader reads the document from top to bottom, he will get a clear idea about the project.

3.2 OVERALL DESCRIPTION

3.2.1 PRODUCT PERSPECTIVE

The Health Discernment System helps not only the patients but also the health-care professionals to predict the medical test results more effectively and in a bias-free manner by computerizing the prediction process.

Predictions or results will be generated from the medical test images which will help the patients to make appropriate decisions regarding the course of their treatment. For example, if the patient fails to get the doctor's appointment, they can still know the results of their medical tests by simply using this interface and this in term saves time and money.

3.2.2 PRODUCT FUNCTIONALITY

All the functions of the system will be performed in this order-

- Upload image
- Read image
- Transform image
- Evaluate image using the saved model
- Determine and analyze the output
- Display the output

3.2.3 USERS AND CHARACTERISTICS

The Health Discernment System has one active actor(patients along with health-care professionals) and one cooperating system at the back end. The patients and health-care professionals like doctors and interns can access the system through a desktop for predicting diseases.

3.2.4 OPERATING ENVIRONMENT

Operating System: Minimum Windows XP or Windows VISTA. Better environment Windows 7, 8, 8.1, 10.

Language: Python

A workstation which will run on any Windows OS is needed to predict the diseases using this system.

3.2.6 ASSUMPTIONS AND DEPENDENCIES

If the user has a workstation that works on Windows OS then that person can easily use the detecting interface. The user also has to have the medical test images in JPEG, JPG or PNG format for prediction purpose.

Chapter 4

Requirement Analysis

4.1 SPECIFIC REQUIREMENTS

4.1.1 EXTERNAL INTERFACE REQUIREMENTS

There are many types of interfaces that are supported by this system. Namely- User Interface and Software Interface.

- USER INTERFACE

The user interface will be implemented using any desktop running on Windows OS. This interface will be very user friendly so that people from different strata can use it to detect their disease without any difficulty by just uploading their medical test image.

- SOFTWARE INTERFACE

A software interface running on Windows OS. It should have Python compiler(3.6).

4.1.2 FUNCTIONAL REQUIREMENTS

- UPLOAD IMAGE

The user can upload the medical test image through a workstation running on Windows OS. The image should be in jpeg, png or jpg format.

- READ IMAGE

The image will be scanned before augmentation takes place.

- TRANSFORM IMAGE

The scanned image is then transformed into a format that is needed by the saved custom model.

- EVALUATE IMAGE USING SAVED MODEL

The saved custom model creates a feature map of the uploaded medical test image and predicts the output.

- DETERMINE AND PREDICT THE OUTPUT

The predicted output is then analyzed (0 means normal; 1 means positive) and converted to a user-friendly language.

- DISPLAY THE OUTPUT

The analyzed result is then displayed to the user.

4.1.3 BEHAVIOUR REQUIREMENTS

- Use Case View

The use cases for the users (patients and health-care professionals) and the system (at the back end) are described in this section and shown in Fig. 4.1.3.

1. User Use Case

Use Case: Select and Upload Image

The patients, as well as the health-care professionals like interns and doctors, can upload the medical images and easily detect the disease.

2. System Use Case

Use Case: Perform internal operations at the back-end.

The system will read the image uploaded by the user, augment it and will use the saved custom model to detect whether the disease is present or not in the patient and thus display the result in a user-friendly language.

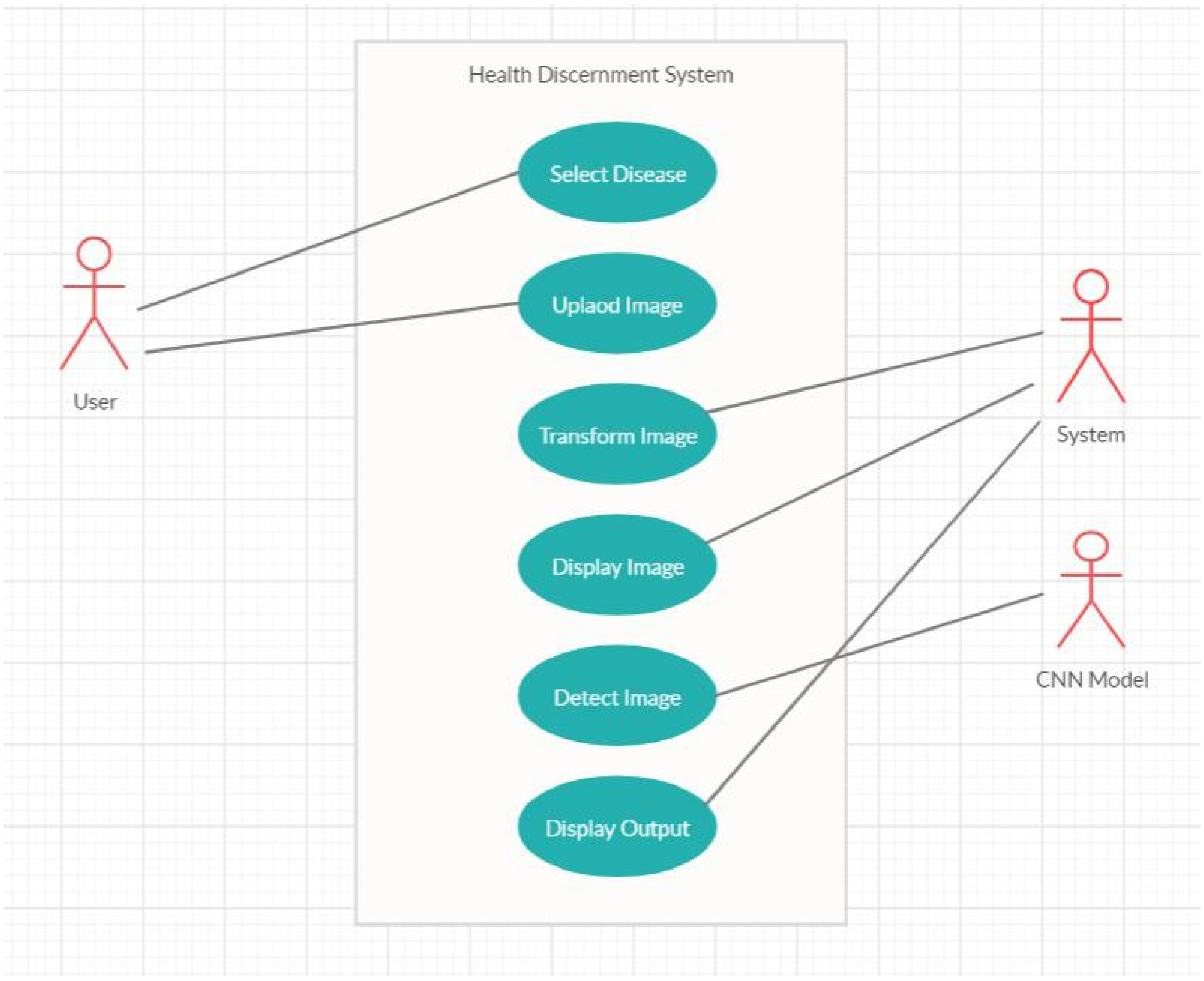


Figure 4.1.3 : Use Case View of the system

4.2 NON FUNCTIONAL REQUIREMENTS

4.2.1 PERFORMANCE REQUIREMENTS

- The product performance will be based on a local system.
- Image detection will take some time.
- The evaluation performance will depend on some software and hardware components.

4.2.2 SAFETY AND SECURITY REQUIREMENTS

The security of the predicting interface depends directly on the security of the workstation.

Chapter 5

System Design

The proposed idea was implemented using Python in the form of a desktop application having a GUI and made using the *Model-View-Controller (MVC)* design pattern. The flow chart of the proposed idea is given in figure Fig. 5.1.1. The application is *”exe”* installable. The Model and the View can run independently using the MVC pattern. The Model module can be used separately in other programs.

5.1 THE MVC MODEL

Model-view-controller (MVC) is an architectural pattern widely used to design user interfaces, which divides an application into three interrelated components. This is done to distinguish internal information representations from how information is communicated to and accepted from the user. The design pattern of MVC decouples these major components to allow efficient reuse of code and also allows parallel development of each of the component. This architecture is mostly used for desktop graphical user interfaces (GUIs) and is popular in designing web applications.

5.1.1 THE MODEL

A *Model* is the application's principal central component. It represents the behaviour of the application, independent of the user interface, in terms of the problem domain. It manages the applications data, logic and rules directly. It is responsible for administering the application's logic and data. It receives user inputs and commands via the View component and uses the logic to generate outputs, which is shown again via the View component. Our Model comprises of various sub-models which represent each disease. In our project, we have used four different diseases, each having its own model.

i. **MALARIA-** *model*

For designing the malaria disease model, we have taken a dataset comprising of cell images of 50x50 pixels in .png format. The dataset is then divided into the train set and test set of 22046 and 5512 images respectively.

Then Convolutional Neural Network (CNN) has been applied, having 3 convolutional layers and one fully-connected layer. 'relu' and 'softmax' activation functions are applied to the hidden layers and the output layer respectively. While compiling 'categorical_crossentropy' has been used as the loss function and 'adam' as an optimization function.

ii. **PNEUMONIA-** *model*

For pneumonia model, a dataset comprising of chest x-rays has been taken. The images are of 224x224 pixels. We then divided the dataset into the train set and test set and as the input images of both the sets are not balanced, we have used class-weight technique to balance it. To the dataset, CNN was applied, having 10 convolutional layers and one fully-connected layer. For activation purpose, 'relu' and 'sigmoid' functions are used in the hidden and output layer respectively. While compiling 'binary_crossentropy' has been used as the loss function and 'adam' as an optimization function.

iii. **BREAST CANCER-** *model*

For this, we have taken a dataset of cell images of micro-anatomy that contains RGB images of 50x50 pixels. To the dataset, we applied a CNN model of 6 convolutional layers and one fully-connected layer. To the output layer and the hidden layers, 'sigmoid' and 'relu' activation functions are applied.

iv. **SKIN CANCER-** *model*

We have taken a dataset comprising of RGB images of skin samples. The images were of 224x224 pixels. The dataset is then divided into the train set and test set of 2637 and 660 images respectively. We have then applied a CNN model of 4 convolutional layers and one fully-connected layer. For activation, 'softmax' and 'relu' are applied to the output layer and hidden layers respectively. While compiling, 'categorical_crossentropy' has been used as loss function and 'adam' as an optimization function.

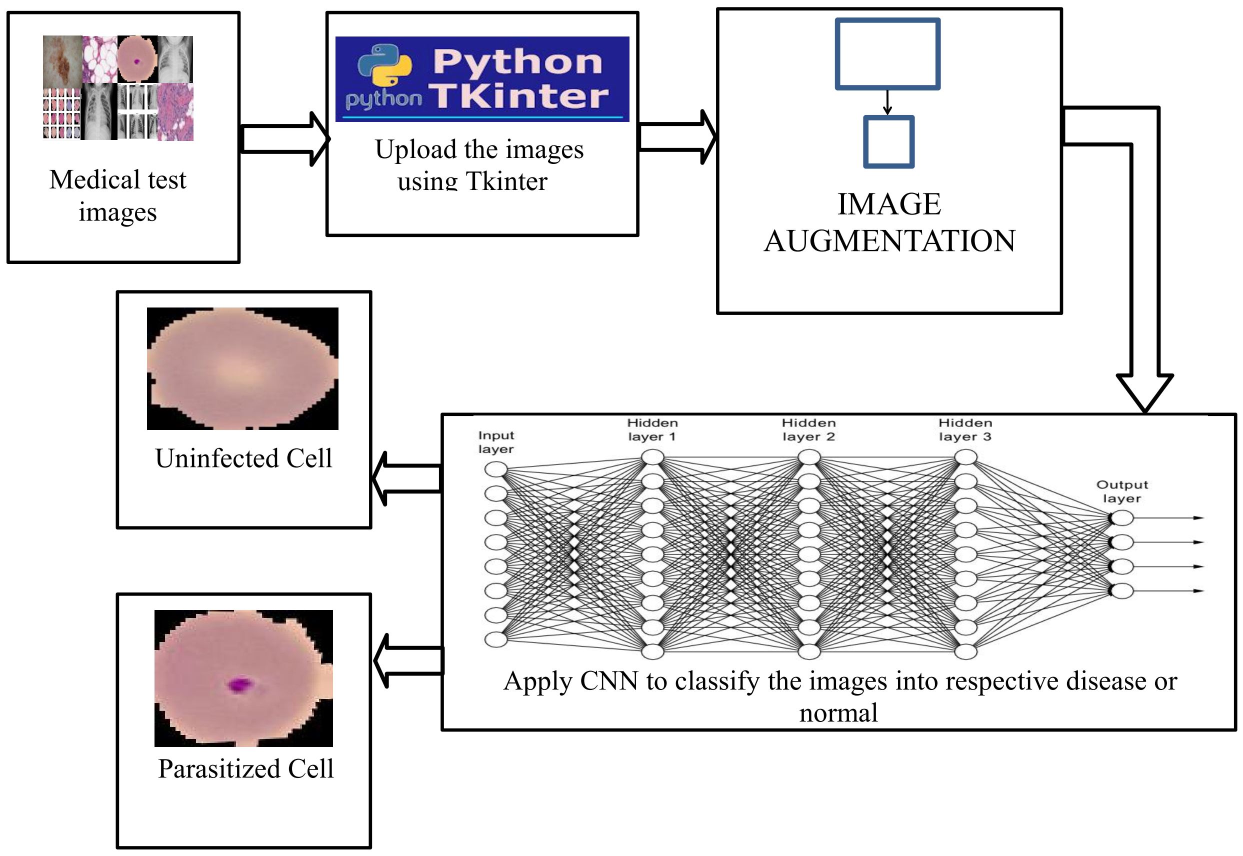


Figure 5.1.1: Flowchart of the proposed method.

5.1.2 THE VIEW

A *View* is something available to the user. It reflects the user interface with which the user is communicating while using an application. While the View has buttons, it, itself remains unaware of the fundamental interaction that exists with the back-end. It helps UI / UX people to operate in parallel with the people at the back-end of the user interface.

5.1.3 THE CONTROLLER

A *Controller* is a master that synchronizes the Model along with View. It obtains the user's interaction with the View, transmits them on to the Model that then processes the input information for output production. Through the View, the outputs (results) are then shown to the user.

5.2 HOW ‘MVC’ FITS IN THIS PROJECT

There are mainly three files in the project: main.py, detector and upload which contains the GUI, Detector and Controller components, respectively. The GUI component is concerned for the View, the Detector component represents the Model and the Controller brings these together by receiving the communication from the GUI component and carrying them on to the Detector component. With the help of these, we can alter and run the GUI component separately without compromising the functionality of the other components. Similarly, by importing separately from the GUI we can modify the Detector component and use it as a module as well.

Chapter 6

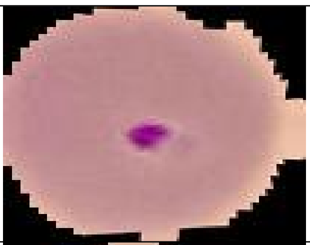
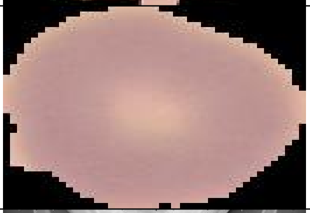
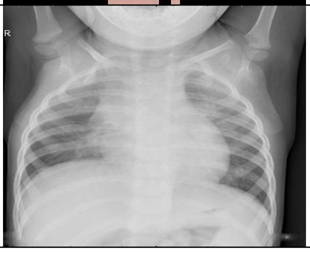
System Testing

The system's basic functionality is to read the user-uploaded image, augment it, and use the stored custom model to detect the disease in the individual, thus displaying the result in a user-friendly format.

The system has been tested to detect malaria, pneumonia, breast cancer and skin cancer by taking various cell, x-ray, tissue and skin image specimens of patients respectively. Table 1 represents the different test cases and their respective results.

Each medical model (CNN) detects the disease with their corresponding accuracy, and none of the models used is mutually dependent. The accuracy of the system in detecting the disease in the patient is 95.21% for Malaria, 90.06% for Pneumonia, 86.88% for Breast cancer and 84.6% for Skin cancer.

6.1 Test Cases and Test Results

Test ID	Test Case Title	Test Condition	System Behaviour	Expected Result
T01	Malaria Detection I		Parasitized Cell Detected	Parasitized Cell Detected
T02	Malaria Detection II		Uninfected Cell Detected	Uninfected Cell Detected
T03	Pneumonia Detection I		Normal Detected	Normal Detected

z

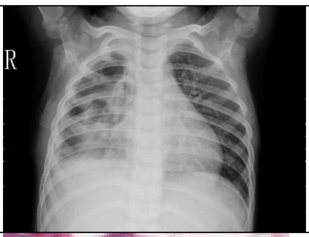
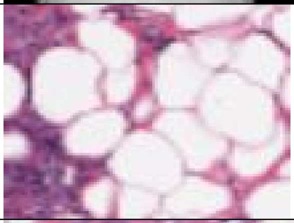
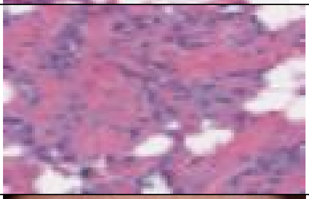
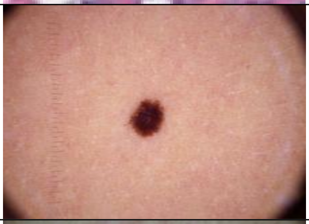
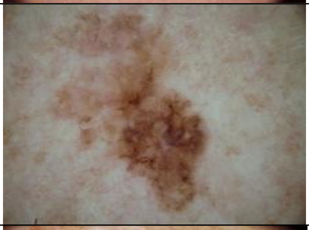
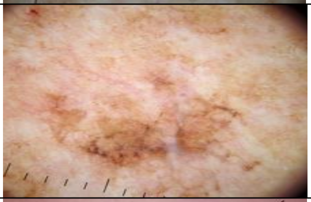
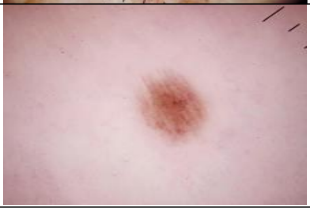
T04	Pneumonia Detection II		Pneumonia Detected	Pneumonia Detected
T05	Breast Cancer Detection I		Benign Tissue Detected	Benign Tissue Detected
T06	Breast Cancer Detection II		Malignant Tissue Detected	Malignant Tissue Detected
T07	Skin Cancer Detection I		Benign Skin Detected	Benign Skin Detected
T08	Skin Cancer Detection II		Malignant Skin Detected	Malignant Skin Detected
T09	Skin Cancer Detection III		Benign Skin Detected	Malignant Skin Detected
T10	Skin Cancer Detection IV		Malignant Skin Detected	Benign Skin Detected
T11	Non-image file uploaded	Any video (.mp4, .mkv etc.), audio (.mp3, .wav etc.), and other file formats (.exe, .py, .txt etc.)	Image File Not Uploaded	Image File Not Uploaded
T12	None uploaded	System idle	Image File Not Uploaded	Image File Not Uploaded

Table 1. Different test cases and their results.

Note: Testing are performed manually

Chapter 7

Project Planning

This project has given us several challenging tasks, such as getting acquainted with the various topics related to our subject area to ensure that the achieved product is at a high quality and predicting the length of tasks throughout the project execution.

7.1 PLANNING

The research process started early during the first month of 2020 and the main emphasis was on tasks such as setting up the system, conducting initial tests to understand the algorithms and acquiring contextual information. Running a few tests has allowed us to understand how much time it would take for test estimation. We used available resources to develop our knowledge and conducted our literature survey on the subject. The first month was spent on studying and understanding image formation and classification process via the use of feature detection. Observing how the network interacts with a broad variety of Convolutional Neural Networks architectures to identify, recognize or distinguish categories, we were able to extract knowledge and code about how a classifier operates on a given dataset. Udemy was also used as an online resource to enhance our understanding of deep neural networks and classifiers. Not only did it provide us with insightful knowledge about how the weights of the network are learned and distributed throughout the network, but it also helped us understand the mathematics and code behind different approaches used on convolutional layers. The environment was set up in the next month, and the dataset of the diseases was found from various sources and the tests were started. These results were continuously updated in a notebook, and their estimates were revised at all stages of the project to reflect the current state of the project.

Chapter 8

Implementation

8.1 WORKING

On opening "*HDS.exe*" a GUI based menu displays four buttons (for each disease). When the user clicks on the button of a particular disease, function call immediately executes the "*application_[disease name].py*" file to initialize a new Tkinter window. This new window consists of the *‘Upload’* button, the *'Detect'* button along with an *ImageView* and *TextView*. This *'Upload'* button calls upon the *open_img()* function which reads the path of the image that is to be uploaded using *filedialog.askopenfilename()*, and stores it in the file named 'Upload' by using the *wrt()* function. The uploaded image is then resized into 325 x 200 pixels automatically before displaying it to the user at the *ImageView* with the help of the 'PIL' module.

When the user clicks on the *‘Detect’* button, the button calls upon the *callback()* function which then retrieves the path of the uploaded image from the 'Upload' file using *red()* function. The image is then transformed into the target size before loading. The corresponding saved CNN model is then used to classify the image. The result is displayed to the user on the screen at the *TextView*. In this way, the user can classify different images of a particular disease in the same window or can go back to the menu to detect different diseases.

8.2 SOFTWARE REQUIREMENT

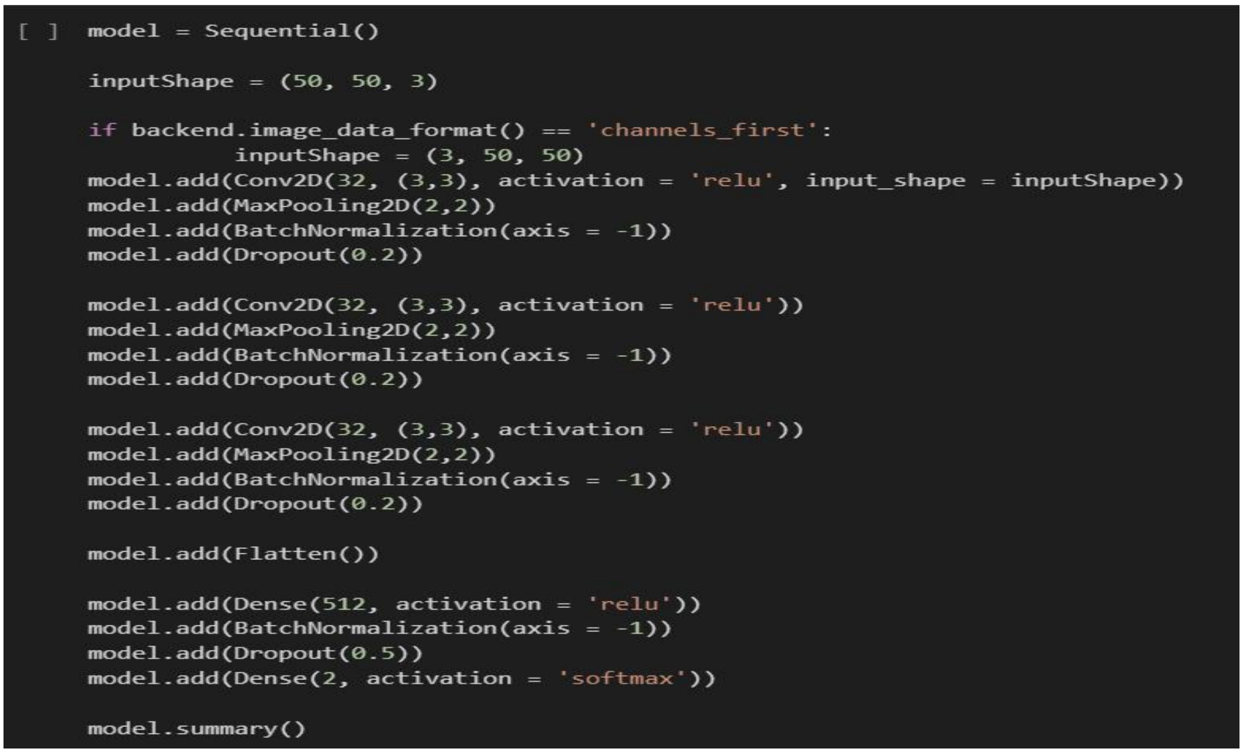
- Python 3.6 or 3.7
- Pip version 19 or 20
- Python modules include Tensorflow, numpy, pandas, PIL, tkinter.

8.3 INSTALLATION

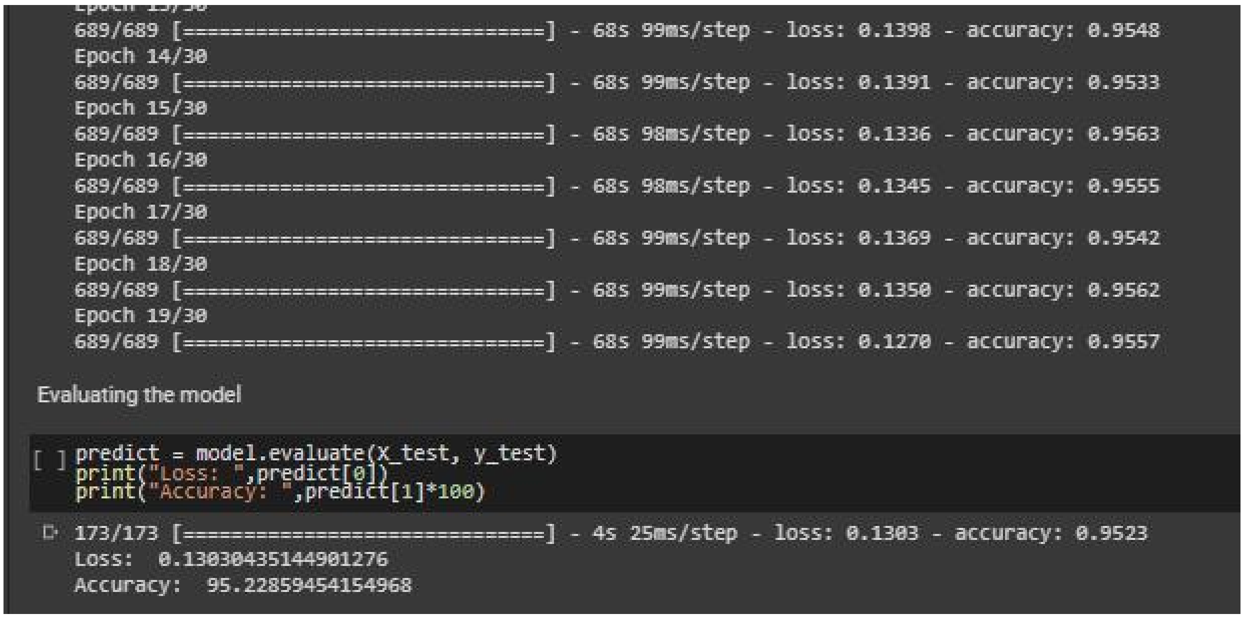
The application is *"exe"* installable and can be installed in the following way-

- Click on the package named *"setup.exe"*, choose the path where you want to install the application and then press the *Install* button.
- You will see the application package installing and after it is done press *Close*.
- Go to the selected path and open the "Health Discernment System" folder.
- Open *"requirements.exe"* to check and install the required libraries. This is only a one-time execution. This may take some time if the required libraries are not installed.
- Open the *"HDS.exe"* file to run the application.

NOTE: Python compiler is the primary requirement.



```
[ ]  model = Sequential()

     inputShape = (50, 50, 3)

     if backend.image_data_format() == 'channels_first':
               inputShape = (3, 50, 50)
     model.add(Conv2D(32, (3,3), activation = 'relu', input_shape = inputShape))
     model.add(MaxPooling2D(2,2))
     model.add(BatchNormalization(axis = -1))
     model.add(Dropout(0.2))

     model.add(Conv2D(32, (3,3), activation = 'relu'))
     model.add(MaxPooling2D(2,2))
     model.add(BatchNormalization(axis = -1))
     model.add(Dropout(0.2))

     model.add(Conv2D(32, (3,3), activation = 'relu'))
     model.add(MaxPooling2D(2,2))
     model.add(BatchNormalization(axis = -1))
     model.add(Dropout(0.2))

     model.add(Flatten())

     model.add(Dense(512, activation = 'relu'))
     model.add(BatchNormalization(axis = -1))
     model.add(Dropout(0.5))
     model.add(Dense(2, activation = 'softmax'))

     model.summary()
```



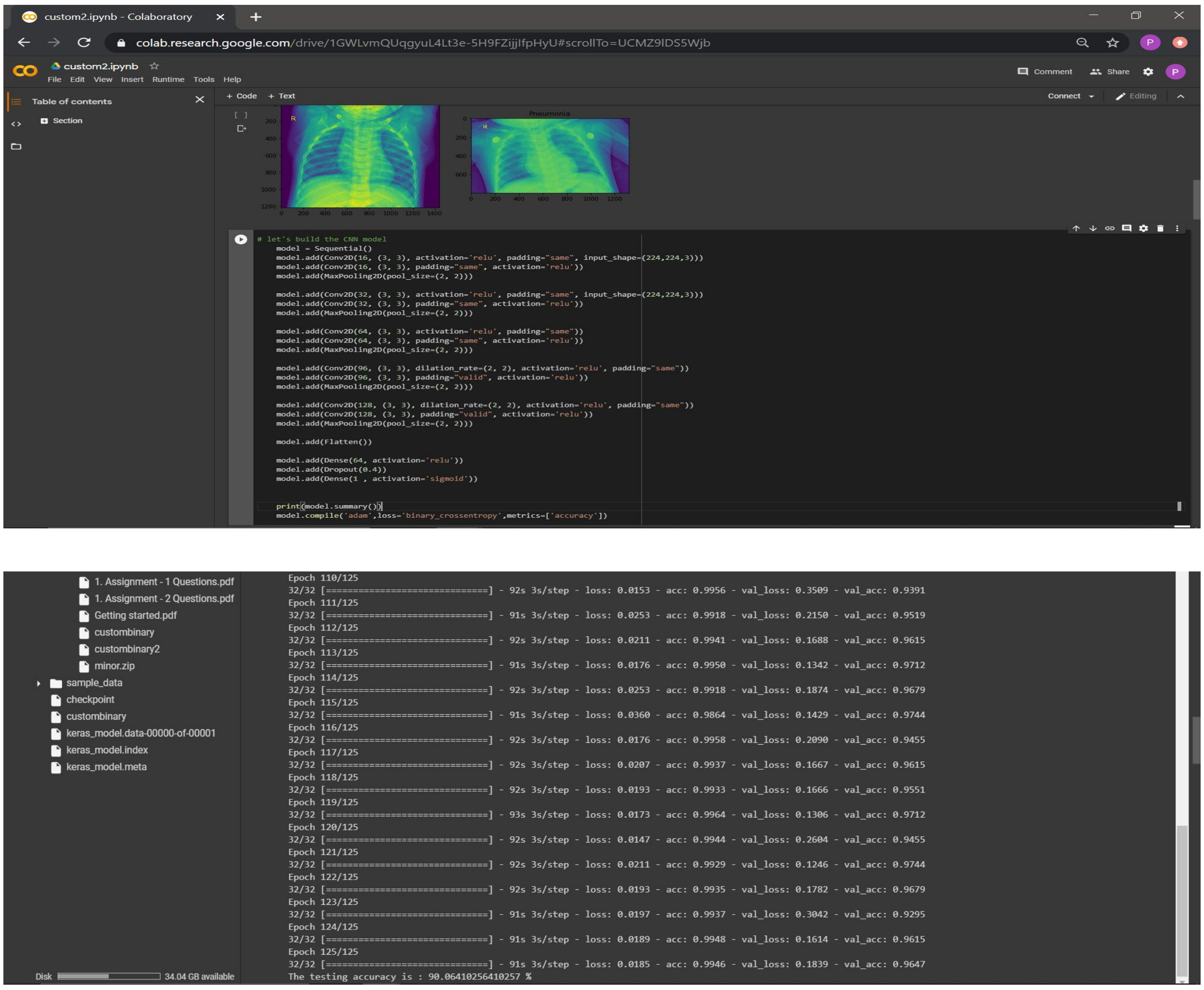
```
689/689 [==============================] - 68s 99ms/step - loss: 0.1398 - accuracy: 0.9548
Epoch 14/30
689/689 [==============================] - 68s 99ms/step - loss: 0.1391 - accuracy: 0.9533
Epoch 15/30
689/689 [==============================] - 68s 98ms/step - loss: 0.1336 - accuracy: 0.9563
Epoch 16/30
689/689 [==============================] - 68s 98ms/step - loss: 0.1345 - accuracy: 0.9555
Epoch 17/30
689/689 [==============================] - 68s 99ms/step - loss: 0.1369 - accuracy: 0.9542
Epoch 18/30
689/689 [==============================] - 68s 99ms/step - loss: 0.1350 - accuracy: 0.9562
Epoch 19/30
689/689 [==============================] - 68s 99ms/step - loss: 0.1270 - accuracy: 0.9557
```

Evaluating the model

```
predict = model.evaluate(x_test, y_test)
print("Loss: ",predict[0])
print("Accuracy: ",predict[1]*100)
```

```
173/173 [==============================] - 4s 25ms/step - loss: 0.1303 - accuracy: 0.9523
Loss:  0.13030435144901276
Accuracy:  95.22859454154968
```

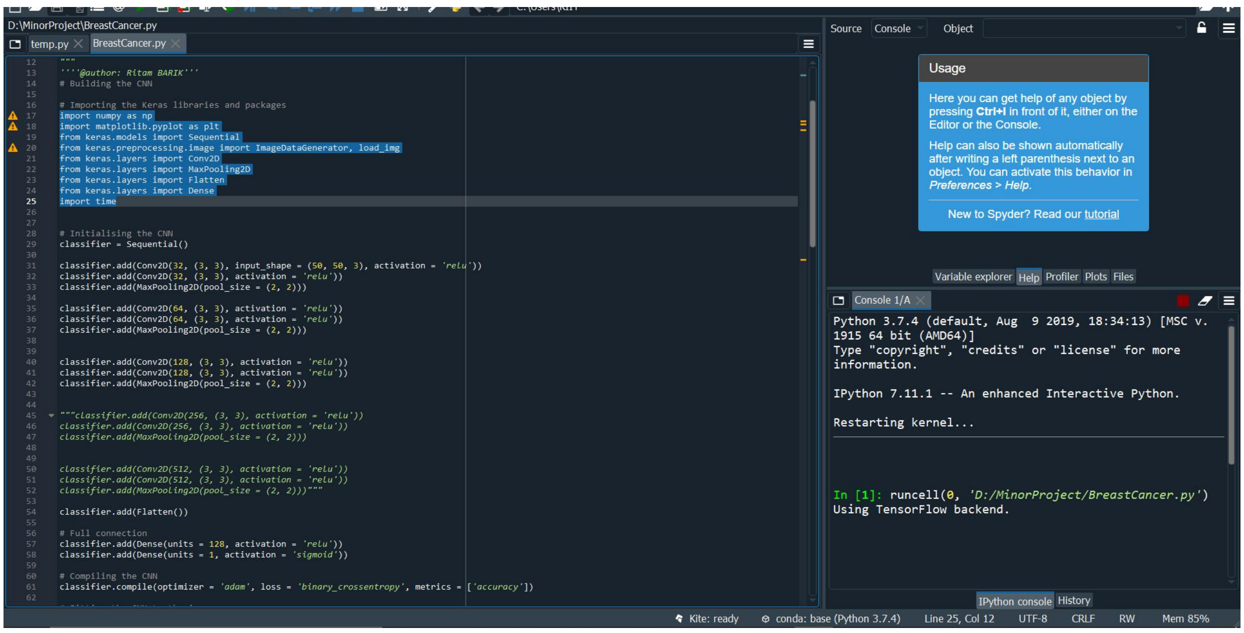
Figure 8.1: Code snippet & output for MALARIA model


custom2.ipynb - Colaboratory
colab.research.google.com/drive/1GWLvmQUqgyuL4Lt3e-5H9FZijjlfpHyU#scrollTo=UCMZ9IDS5Wjb
custom2.ipynb
File Edit View Insert Runtime Tools Help
Comment
Share
Table of contents
Section
+ Code + Text
Connect
Editing
Pneumonia
1. Assignment - 1 Questions.pdf
1. Assignment - 2 Questions.pdf
Getting started.pdf
custombinary
custombinary2
minor.zip
sample_data
checkpoint
custombinary
keras_model.data-00000-of-00001
keras_model.index
keras_model.meta
Disk 34.04 GB available

```
# let's build the CNN model
    model = Sequential()
    model.add(Conv2D(16, (3, 3), activation='relu', padding="same", input_shape=(224,224,3)))
    model.add(Conv2D(16, (3, 3), padding="same", activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Conv2D(32, (3, 3), activation='relu', padding="same", input_shape=(224,224,3)))
    model.add(Conv2D(32, (3, 3), padding="same", activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Conv2D(64, (3, 3), activation='relu', padding="same"))
    model.add(Conv2D(64, (3, 3), padding="same", activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Conv2D(96, (3, 3), dilation_rate=(2, 2), activation='relu', padding="same"))
    model.add(Conv2D(96, (3, 3), padding="valid", activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Conv2D(128, (3, 3), dilation_rate=(2, 2), activation='relu', padding="same"))
    model.add(Conv2D(128, (3, 3), padding="valid", activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Flatten())

    model.add(Dense(64, activation='relu'))
    model.add(Dropout(0.4))
    model.add(Dense(1 , activation='sigmoid'))


    print(model.summary())
    model.compile('adam',loss='binary_crossentropy',metrics=['accuracy'])
```

```
Epoch 110/125
32/32 [==============================] - 92s 3s/step - loss: 0.0153 - acc: 0.9956 - val_loss: 0.3509 - val_acc: 0.9391
Epoch 111/125
32/32 [==============================] - 91s 3s/step - loss: 0.0253 - acc: 0.9918 - val_loss: 0.2150 - val_acc: 0.9519
Epoch 112/125
32/32 [==============================] - 92s 3s/step - loss: 0.0211 - acc: 0.9941 - val_loss: 0.1688 - val_acc: 0.9615
Epoch 113/125
32/32 [==============================] - 91s 3s/step - loss: 0.0176 - acc: 0.9950 - val_loss: 0.1342 - val_acc: 0.9712
Epoch 114/125
32/32 [==============================] - 92s 3s/step - loss: 0.0253 - acc: 0.9918 - val_loss: 0.1874 - val_acc: 0.9679
Epoch 115/125
32/32 [==============================] - 91s 3s/step - loss: 0.0360 - acc: 0.9864 - val_loss: 0.1429 - val_acc: 0.9744
Epoch 116/125
32/32 [==============================] - 92s 3s/step - loss: 0.0176 - acc: 0.9958 - val_loss: 0.2090 - val_acc: 0.9455
Epoch 117/125
32/32 [==============================] - 92s 3s/step - loss: 0.0207 - acc: 0.9937 - val_loss: 0.1667 - val_acc: 0.9615
Epoch 118/125
32/32 [==============================] - 92s 3s/step - loss: 0.0193 - acc: 0.9933 - val_loss: 0.1666 - val_acc: 0.9551
Epoch 119/125
32/32 [==============================] - 93s 3s/step - loss: 0.0173 - acc: 0.9964 - val_loss: 0.1306 - val_acc: 0.9712
Epoch 120/125
32/32 [==============================] - 92s 3s/step - loss: 0.0147 - acc: 0.9944 - val_loss: 0.2604 - val_acc: 0.9455
Epoch 121/125
32/32 [==============================] - 92s 3s/step - loss: 0.0211 - acc: 0.9929 - val_loss: 0.1246 - val_acc: 0.9744
Epoch 122/125
32/32 [==============================] - 92s 3s/step - loss: 0.0193 - acc: 0.9935 - val_loss: 0.1782 - val_acc: 0.9679
Epoch 123/125
32/32 [==============================] - 91s 3s/step - loss: 0.0197 - acc: 0.9937 - val_loss: 0.3042 - val_acc: 0.9295
Epoch 124/125
32/32 [==============================] - 91s 3s/step - loss: 0.0189 - acc: 0.9948 - val_loss: 0.1614 - val_acc: 0.9615
Epoch 125/125
32/32 [==============================] - 91s 3s/step - loss: 0.0185 - acc: 0.9946 - val_loss: 0.1839 - val_acc: 0.9647
The testing accuracy is : 90.06410256410257 %
```

Figure 8.2: Code snippet & output for PNEUMONIA model



D:\MinorProject\BreastCancer.py

```python
"""
'''@author: Ritam BARIK'''
# Building the CNN

# Importing the Keras libraries and packages
import numpy as np
import matplotlib.pyplot as plt
from keras.models import Sequential
from keras.preprocessing.image import ImageDataGenerator, load_img
from keras.layers import Conv2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
import time


# Initialising the CNN
classifier = Sequential()

classifier.add(Conv2D(32, (3, 3), input_shape = (50, 50, 3), activation = 'relu'))
classifier.add(Conv2D(32, (3, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2, 2)))

classifier.add(Conv2D(64, (3, 3), activation = 'relu'))
classifier.add(Conv2D(64, (3, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2, 2)))


classifier.add(Conv2D(128, (3, 3), activation = 'relu'))
classifier.add(Conv2D(128, (3, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2, 2)))


"""classifier.add(Conv2D(256, (3, 3), activation = 'relu'))
classifier.add(Conv2D(256, (3, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2, 2)))


classifier.add(Conv2D(512, (3, 3), activation = 'relu'))
classifier.add(Conv2D(512, (3, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2, 2)))"""

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'])
```

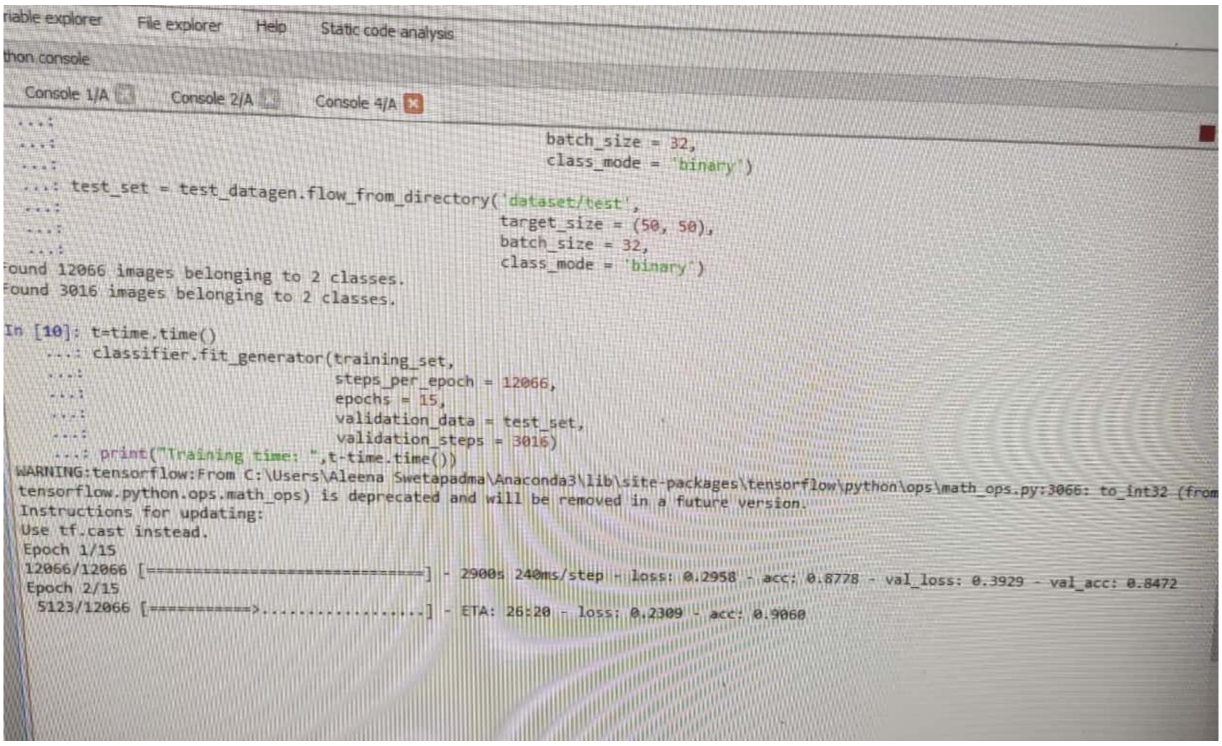
Usage

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help.*

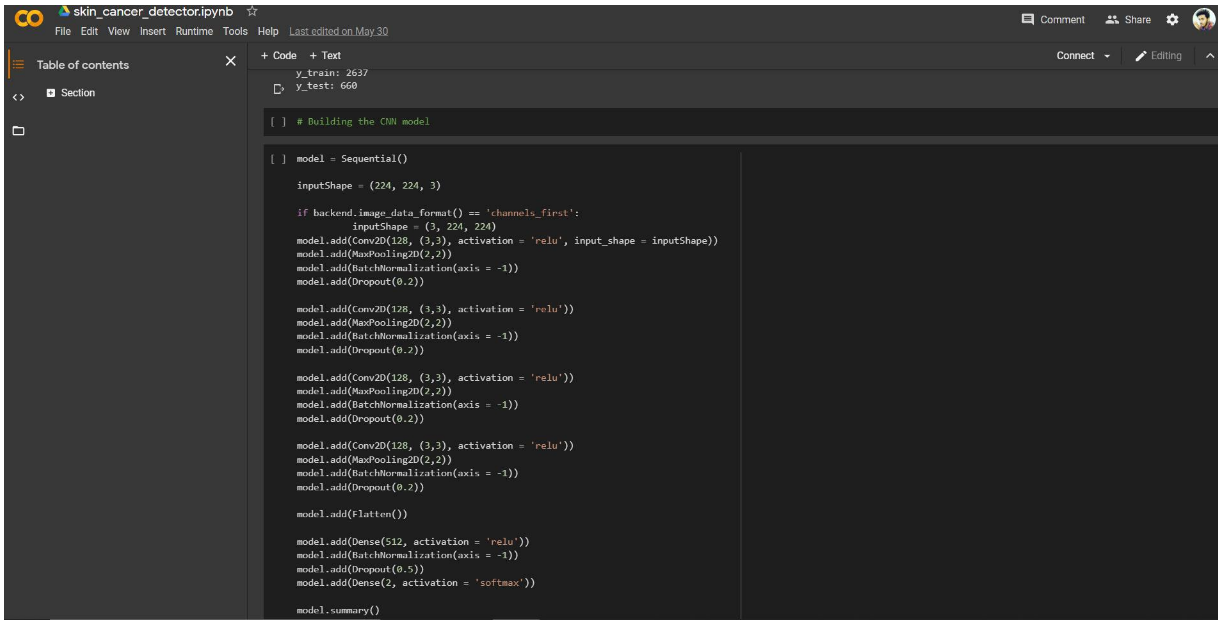
New to Spyder? Read our tutorial

```
Python 3.7.4 (default, Aug  9 2019, 18:34:13) [MSC v.
1915 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more
information.

IPython 7.11.1 -- An enhanced Interactive Python.

Restarting kernel...


In [1]: runcell(0, 'D:/MinorProject/BreastCancer.py')
Using TensorFlow backend.
```



```
...:
...:                                        batch_size = 32,
...:                                        class_mode = 'binary')
...: test_set = test_datagen.flow_from_directory('dataset/test',
...:                                        target_size = (50, 50),
...:                                        batch_size = 32,
...:                                        class_mode = 'binary')
ound 12066 images belonging to 2 classes.
ound 3016 images belonging to 2 classes.

In [10]: t=time.time()
    ...: classifier.fit_generator(training_set,
    ...:                          steps_per_epoch = 12066,
    ...:                          epochs = 15,
    ...:                          validation_data = test_set,
    ...:                          validation_steps = 3016)
    ...: print("Training time: ",t-time.time())
WARNING:tensorflow:From C:\Users\Aleena Swetapadma\Anaconda3\lib\site-packages\tensorflow\python\ops\math_ops.py:3066: to_int32 (from
tensorflow.python.ops.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Epoch 1/15
12066/12066 [==============================] - 2900s 240ms/step - loss: 0.2958 - acc: 0.8778 - val_loss: 0.3929 - val_acc: 0.8472
Epoch 2/15
 5123/12066 [===========>..................] - ETA: 26:20 - loss: 0.2309 - acc: 0.9060
```

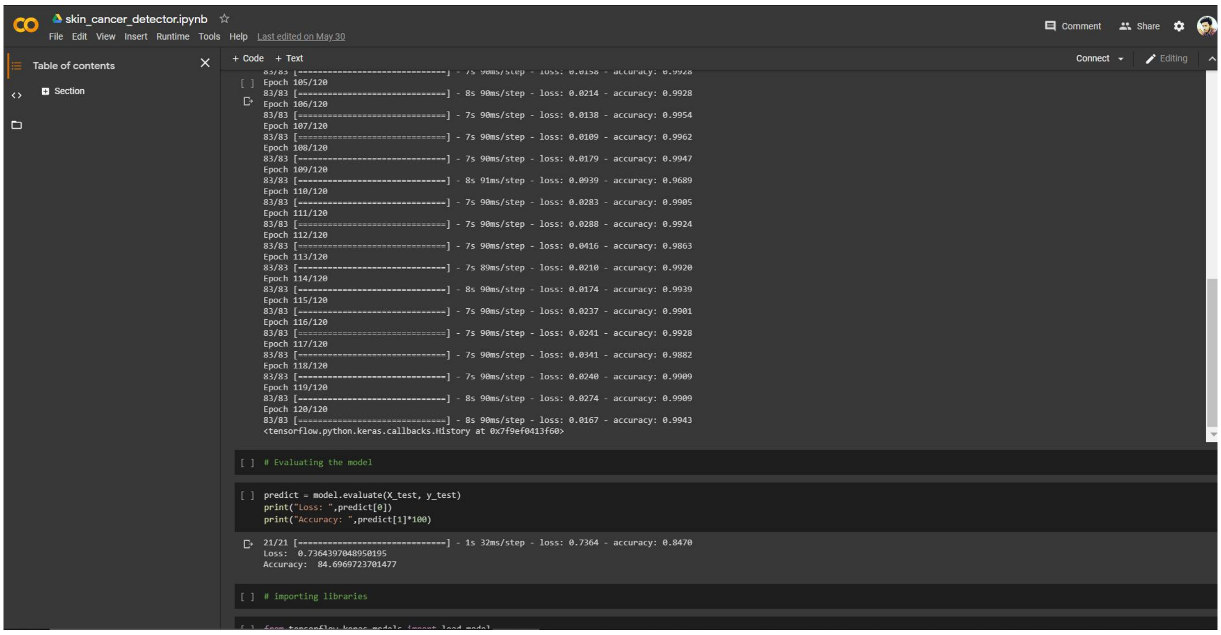
Figure 8.3: Code snippet & output of BREAST CANCER model


skin_cancer_detector.ipynb
File Edit View Insert Runtime Tools Help Last edited on May 30
Comment Share
Table of contents
Section
+ Code + Text
Connect
Editing

```
y_train: 2637
y_test: 660
```

```
# Building the CNN model
```

```
model = Sequential()

inputShape = (224, 224, 3)

if backend.image_data_format() == 'channels_first':
        inputShape = (3, 224, 224)
model.add(Conv2D(128, (3,3), activation = 'relu', input_shape = inputShape))
model.add(MaxPooling2D(2,2))
model.add(BatchNormalization(axis = -1))
model.add(Dropout(0.2))

model.add(Conv2D(128, (3,3), activation = 'relu'))
model.add(MaxPooling2D(2,2))
model.add(BatchNormalization(axis = -1))
model.add(Dropout(0.2))

model.add(Conv2D(128, (3,3), activation = 'relu'))
model.add(MaxPooling2D(2,2))
model.add(BatchNormalization(axis = -1))
model.add(Dropout(0.2))

model.add(Conv2D(128, (3,3), activation = 'relu'))
model.add(MaxPooling2D(2,2))
model.add(BatchNormalization(axis = -1))
model.add(Dropout(0.2))

model.add(Flatten())

model.add(Dense(512, activation = 'relu'))
model.add(BatchNormalization(axis = -1))
model.add(Dropout(0.5))
model.add(Dense(2, activation = 'softmax'))

model.summary()
```


skin_cancer_detector.ipynb
File Edit View Insert Runtime Tools Help Last edited on May 30
Comment Share
Table of contents
Section
+ Code + Text
Connect
Editing

```
83/83 [==============================] - 7s 90ms/step - loss: 0.0158 - accuracy: 0.9928
Epoch 105/120
83/83 [==============================] - 8s 90ms/step - loss: 0.0214 - accuracy: 0.9928
Epoch 106/120
83/83 [==============================] - 7s 90ms/step - loss: 0.0138 - accuracy: 0.9954
Epoch 107/120
83/83 [==============================] - 7s 90ms/step - loss: 0.0109 - accuracy: 0.9962
Epoch 108/120
83/83 [==============================] - 7s 90ms/step - loss: 0.0179 - accuracy: 0.9947
Epoch 109/120
83/83 [==============================] - 8s 91ms/step - loss: 0.0939 - accuracy: 0.9689
Epoch 110/120
83/83 [==============================] - 7s 90ms/step - loss: 0.0283 - accuracy: 0.9905
Epoch 111/120
83/83 [==============================] - 7s 90ms/step - loss: 0.0288 - accuracy: 0.9924
Epoch 112/120
83/83 [==============================] - 7s 90ms/step - loss: 0.0416 - accuracy: 0.9863
Epoch 113/120
83/83 [==============================] - 7s 89ms/step - loss: 0.0210 - accuracy: 0.9920
Epoch 114/120
83/83 [==============================] - 8s 90ms/step - loss: 0.0174 - accuracy: 0.9939
Epoch 115/120
83/83 [==============================] - 7s 90ms/step - loss: 0.0237 - accuracy: 0.9901
Epoch 116/120
83/83 [==============================] - 7s 90ms/step - loss: 0.0241 - accuracy: 0.9928
Epoch 117/120
83/83 [==============================] - 7s 90ms/step - loss: 0.0341 - accuracy: 0.9882
Epoch 118/120
83/83 [==============================] - 7s 90ms/step - loss: 0.0240 - accuracy: 0.9909
Epoch 119/120
83/83 [==============================] - 8s 90ms/step - loss: 0.0274 - accuracy: 0.9909
Epoch 120/120
83/83 [==============================] - 8s 90ms/step - loss: 0.0167 - accuracy: 0.9943
<tensorflow.python.keras.callbacks.History at 0x7f9ef0413f60>
```

```
# Evaluating the model
```

```
predict = model.evaluate(X_test, y_test)
print("Loss: ",predict[0])
print("Accuracy: ",predict[1]*100)
```

```
21/21 [==============================] - 1s 32ms/step - loss: 0.7364 - accuracy: 0.8470
Loss:  0.7364397048950195
Accuracy:  84.6969723701477
```

```
# importing libraries
```

```
from tensorflow.keras.models import load_model
```

Figure 8.4: Code snippet & output for SKIN CANCER model

Chapter 9

Screenshots of Project

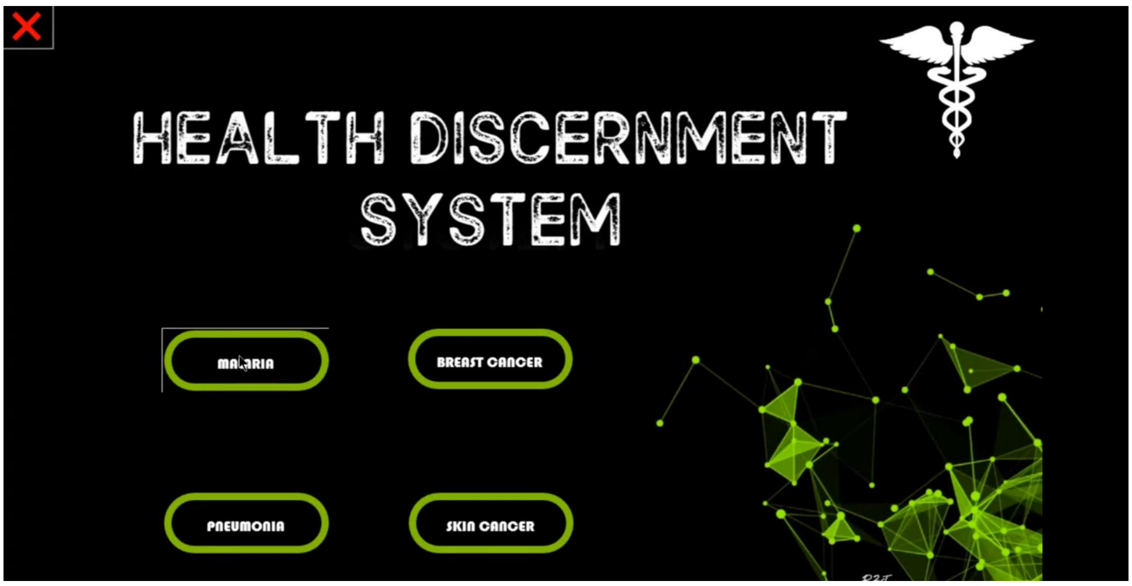


Figure 9.1: Health Discernment System interface I

On opening the *"HDS.exe"* file, the above opening page will be displayed on the screen as shown in figure 9.1. It consists of four button, each representing a disease. The user will click on the disease they want to detect.

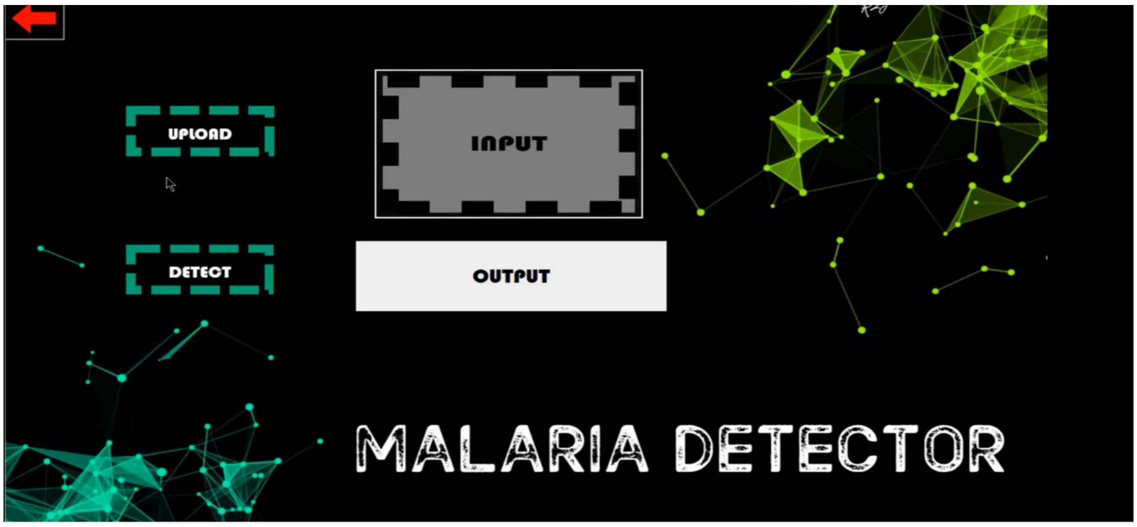


Figure 9.2: Health Discernment System interface II

On selection of the the disease, the page shown in figure 9.2 opens. It consists of the *'UPLOAD'* button and *'DETECT'* button accompanied with the *INPUT* area and the *OUTPUT* area.

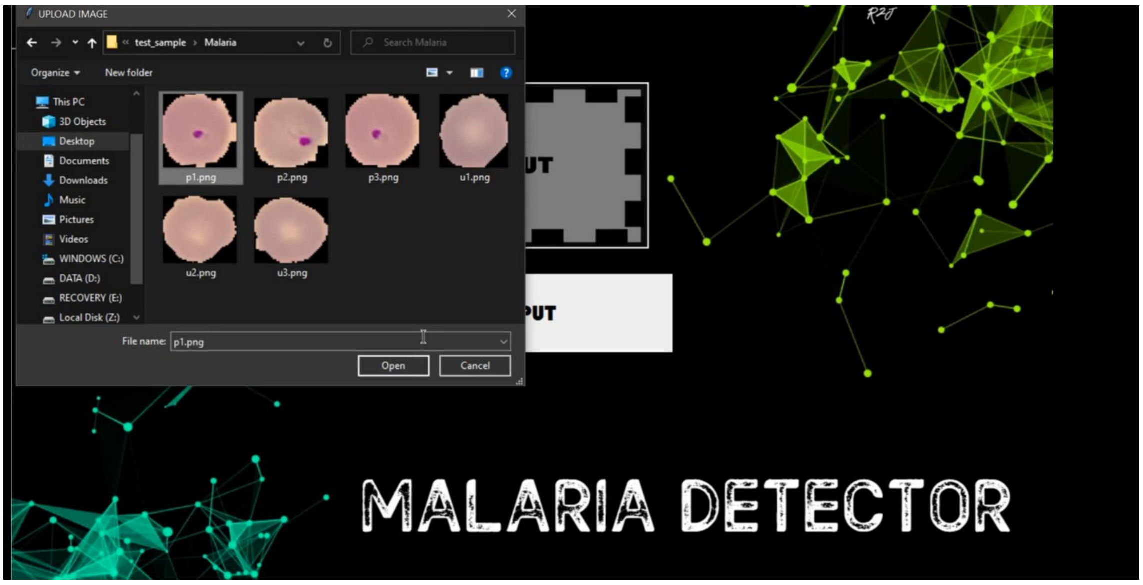


Figure 9.3: Selecting the disease image

When the user clicks on the *'UPLOAD'* button, a file opens where the user selects the image of the disease that is to be predicted, as shown in figure 9.3.

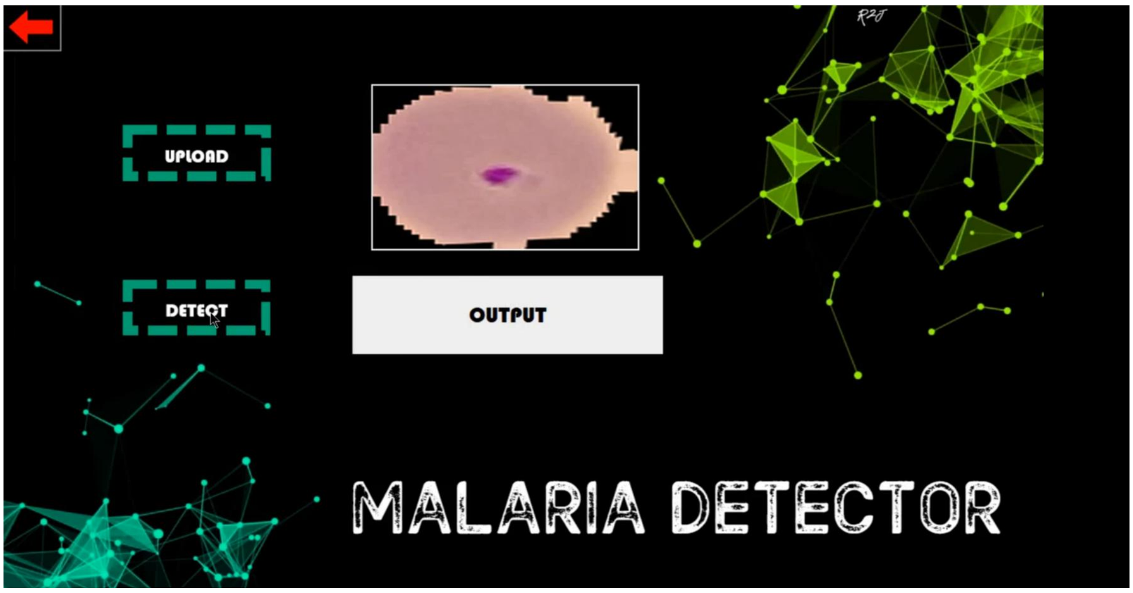


Figure 9.4: Selected image displayed on the screen

The selected image is displayed on the screen to the user, as visible in figure 9.4.

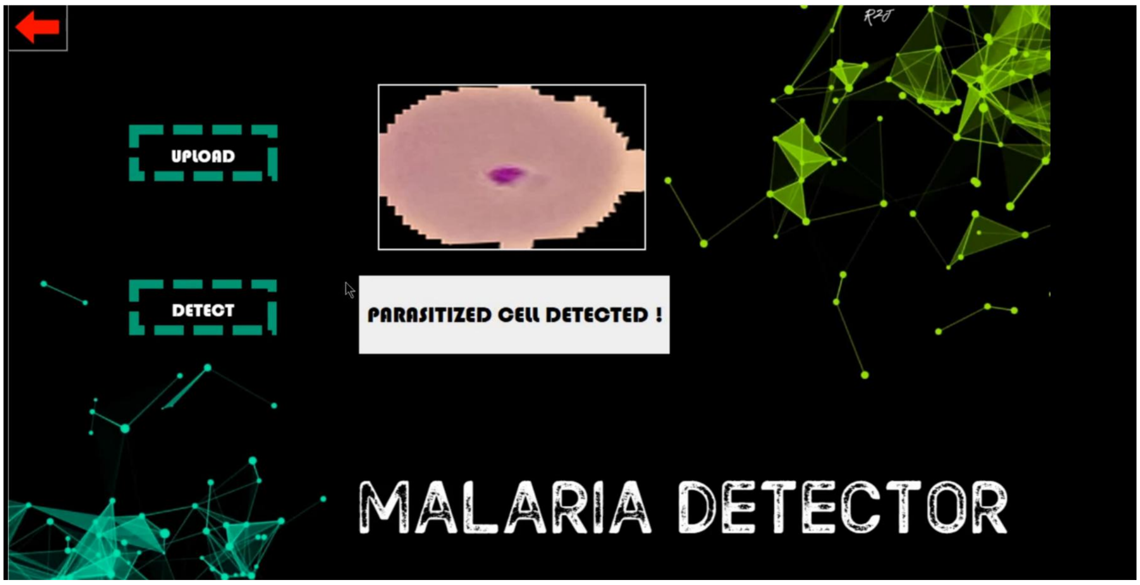


Figure 9.5: Output displayed

When the user clicks on the *'DETECT'* button, the output/result is displayed to the user as shown in figure 9.5.

Chapter 10

Conclusion and Future Scope

10.1 CONCLUSION

In this work, a health discernment system has been proposed for image classification that will work in real-life scenarios. The proposed method is based on an MVC architecture and the model is responsible for the behavioral aspect of the application. Different sub-models pertaining to the four diseases (malaria, pneumonia, breast cancer, and skin cancer) have been designed using a convolutional neural network (CNN). Each of the sub-models has its own specific set of activation and loss function with a single optimization function across all the models. Each of the models has been trained separately and the performance of the proposed model has been validated using a completely separate test set. Among all the four diseases, the model for skin cancer recorded the lowest accuracy at 84.6%. While classification accuracy to detect the presence of parasitized cells of malaria is seen to be highest at 95.21%. Pneumonia and breast cancer models showed performance accuracy of 90.47% and 86.88% respectively. Although the computational time of training each of the models is high, once the trained model is deployed, testing time is minimal. A GUI based web application has been designed with the help of python Tkinter such that real-life input can be given which will be passed on to the trained model for prediction. Therefore, the proposed work is suitable to be used in real-life situations as it is user friendly and cost-effective in nature.

10.2 FUTURE SCOPE

1. The future scope of the work lies on implementing various other algorithms and using several optimization techniques. Also more data will be collected in order to recognize the features more accurately.

2. Major attention will be given to increase the accuracy such that our proposed system can be used to detect a large number of chronic and critical diseases.

3. When these enhancements are done, the system can be integrated with an android application to make it more convenient and easily portable. This will allow people from all strata to use it effectively even when they do not have a personal computer.

References

[1] F.A. Spanhol, L.S. Oliveira, P.R. Cavalin, C. Petitjean, L. Heutte; Deep features for breast cancer histopathological image classification. In 2017 IEEE international conference on systems, man, and cybernetics, SMC 2017, Banff, AB, Canada, October 5-8, 2017 (2017), pp. 1868-1873.

[2] F.A. Spanhol, L.S. Oliveira, C. Petitjean, L. Heutte; Breast cancer histopathological image classification using convolutional neural networks. In 2016 international joint conference on neural networks, IJCNN 2016, Vancouver, BC, Canada, July 24-29, 2016 (2016), pp. 2560-2567,

[3] Z. Han, B. Wei, Y. Zheng, Y. Yin, K. Li, S. Li; Breast cancer multi-classification from histopathological images with structured deep learning model.
Sci Rep, 7 (1) (2017), p. 4172

[4] J. Sun, A. Binder; Comparison of deep learning architectures for h&e histopathology images. In 2017 IEEE conference on big data and analytics (ICBDA), IEEE (2017), pp. 43-48

[5] A. Alias, B. Paulchamy, Detection of breast cancer using artificial neural network, International Journal of Innovative Research in Science 3 (3).

[6] M.G. Kanojia, S. Abraham; Breast cancer detection using RBF neural network. In Contemporary computing and informatics (IC3I), 2016 2nd international conference on, IEEE (2016), pp. 363-368

[7] A.F. Agarap; On breast cancer detection: an application of machine learning algorithms on the Wisconsin diagnostic dataset, CoRR abs/1711.07831

[8] M. Karabatak, M.C. Ince; An expert system for detection of breast cancer based on association rules and neural network. Expert Syst Appl, 36 (2) (2009), pp. 3465-3469

[9] S. Chou, T. Lee, Y.E. Shao, I. Chen; Mining the breast cancer pattern using artificial neural networks and multivariate adaptive regression splines. Expert Syst Appl, 27 (1) (2004), pp. 133-142

[10] S. Sahan, K. Polat, H. Kodaz, S. Günes; A new hybrid method based on the fuzzy-artificial immune system and K-NN algorithm for breast cancer diagnosis
Comput Biol Med, 37 (3) (2007), pp. 415-423

[11] F.A. Spanhol, L.S. Oliveira, C. Petitjean, L. Heutte; A dataset for breast cancer histopathological image classification. IEEE Trans Biomed Eng, 63 (7) (2016), pp. 1455-1462.

[13] A. Chon, N. Balachandra, P. Lu, Deep convolutional neural networks for lung cancer detection, Standford University.

[14] A.A. Cruz-Roa, J.E.A. Ovalle, A. Madabhushi, F.A.G. Osorio; A deep learning architecture for image representation, visual interpretability and automated basal-cell carcinoma cancer detection. Medical image computing and computer-assisted intervention - (MICCAI) 2013 - 16th international conference, Nagoya, Japan, September 22-26, 2013, proceedings, Part II (2013), pp. 403-410.

[15] M. Veta, P.J. van Diest, S.M. Willems, H. Wang, A. Madabhushi, A. Cruz-Roa, F.A. González, A.B.L. Larsen, J.S. Vestergaard, A.B. Dahl, D.C. Ciresan, J. Schmidhuber, A. Giusti, L.M. Gambardella, F.B. Tek, T. Walter, C. Wang, S. Kondo, B.J. Matuszewski, F. Prec ioso, V. Snell, J. Kittler, T.E. de Campos, A.M. Khan, N.M. Rajpoot, E. Arkoumani, M.M. Lacle, M.A. Viergever, J.P.W. Pluim; Assessment of algorithms for mitosis detection in breast cancer histopathology images. Med Image Anal, 20 (1) (2015), pp. 237-248.

[16] R. Kumar, R. Srivastava, S. Srivastava; Detection and classification of cancer from microscopic biopsy images using clinically significant and biologically interpretable features. Journal of medical engineering (2015)

[17] E. Andre, K. Brett, A Roberto et al., "Dermatologist-level classification of skin cancer with deep neural networks," *Nature*, vol. 542, no. 7639, pp. 115–118, 2017.

[18] M. Grewal, M. M. Srivastava, P. Kumar, and S. Varadarajan, "Radiologist level accuracy using deep learning for haemorrhage detection in CT scans," 2017.

[19] R. Pranav, Y. H. Awni, H. Masoumeh, B. Codie, and Y. N. Andrew, "Cardiologist-level arrhythmia detection with convolutional neural networks," 2017.

[20] G. Varun, P. Lily, C. Marc et al., "Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs," *JAMA*, vol. 316, no. 22, pp. 2402–2410, 2017.

[21] P. Huang, S. Park, R. Yan et al., "Added value of computer-aided CT image features for early lung cancer diagnosis with small pulmonary nodules: a matched case-control study," *Radiology*, vol. 286, no. 1, pp. 286–295, 2017.

[22] P. Lakhani and B. Sundaram, "Deep learning at chest radiography: automated classification of pulmonary tuberculosis by using convolutional neural networks," *Radiology*, vol. 284, no. 2, pp. 574–582, 2017.

[23] F. D. Demner, M. D. Kohli, M. B. Rosenman et al., "Preparing a collection of radiology examinations for distribution and retrieval," *Journal of the American Medical Informatics Association*, vol. 23, no. 2, pp. 304–310, 2015.

[24] T. I. Mohammad, A. A. Md, T. M. Ahmed, and A. Khalid, "Abnormality detection and localization in chest x-rays using deep convolutional neural networks," 2017.

[25] Li. Yao, E. Poblenz, D. Dagunts, B. Covington, D. Bernard, and K. Lyman, "Learning to diagnose from scratch by exploiting dependencies among labels," 2017.

[26] W. Xiaosong, P. Yifan, L. Le, L. Zhiyong, B. Mohammadhadi, and M. S. Ronald, "Chest X-ray8: Hospital-scale chest X-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases," 2017.

[27] H. C. Shin, L. Lu, L. Kim, A. Seff, J. Yao, and R. M. Summers, "Interleaved text/image deep mining on a very large- scale radiology database," in *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, June 2015.

[28] H. C. Shin, L. Lu, L. Kim, A. Seff, J. Yao, and R. M. Summers, "Interleaved text/image deep mining on a large-scale radiology database for automated image interpretation," *Journal of Machine Learning Research*, vol. 17, no. 107, pp. 1–31, 2016.

[29] H. Boussaid and I. Kokkinos, "Fast and exact: ADMM-based discriminative shape segmentation with loopy part models," in *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, Columbus, OH, USA, June 2014.

[30] U. Avni, H. Greenspan, E. Konen, M. Sharon, and J. Goldberger, "X-ray categorization and retrieval on the organ and pathology level, using patch-based visual words," *Med Imaging, IEEE Transactions*, vol. 30, no. 3, 2011.

[31] J. Melendez, G. B. Van, P. Maduskar et al., "A novel multiple-instance learning-based approach to computer-aided detection of tuberculosis on chest x-ray," *IEEE Transactions on Medical Imaging*, vol. 34, no. 1, pp. 179–192, 2015.

[32] S. Jaeger, A. Karargyris, S. Candemir et al., "Automatic tuberculosis screening using chest radiographs," *IEEE Transactions on Medical Imaging*, vol. 33, no. 2, pp. 233–245, 2014.

[33] Z. Xue, D. You, S. Candemir et al., "Chest x-ray image view classification," in *Proceedings of the Computer-Based Medical Systems IEEE 28th International Symposium*, São Paulo, Brazil, June 2015.

[34] S. Hermann, "Evaluation of scan-line optimization for 3d medical image registration," in *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, Columbus, OH, USA, June 2014.

[36] Nadjm, Behzad, and Ron H. Behrens. "Malaria: An update for physicians." Infectious Disease Clinics 26, no. 2 (2012): 243-259.

[37] Gollin, Douglas, and Christian Zimmermann. "Malaria: Disease impacts and long-run income differences." (2007).

[38] Star Health Desk, (2018, March 18 Published). Shortage of pathologists inhibits progress on UHC.

[39] Chaity, A. Z. (2017, December 13 Published). Bangladeshis flock to Indian, Thai hospitals in huge numbers. Dhaka Tribune

[40] Litjens, Geert, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I. Sánchez. "A survey on deep learning in medical image analysis." Medical image analysis 42 (2017): 60-88.

[41] Liang, Zhaohui, Andrew Powell, Ilker Ersoy, Mahdieh Poostchi, Kamolrat Silamut, Kannappan Palaniappan, Peng Guo et al. "CNN-based image analysis for malaria diagnosis." In 2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), pp. 493-496. IEEE, 2016.

[42] Krizhevsky, Alex, and Geoff Hinton. "Convolutional deep belief networks on cifar- 10." Unpublished manuscript 40, no. 7 (2010).

[43] Dong, Yuhang, Zhuocheng Jiang, Hongda Shen, W. David Pan, Lance A. Williams, Vishnu VB Reddy, William H. Benjamin, and Allen W. Bryan. "Evaluations of deep convolutional neural networks for automatic identification of malaria-infected cells." In 2017 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI), pp. 101-104. IEEE, 2017.

[44] LeCun, Yann. "LeNet-5, convolutional neural networks." (2015)

[45] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." In Advances in neural information processing systems, pp. 1097-1105. 2012.

[46] Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1-9. 2015.

[47] Hung, Jane, and Anne Carpenter. "Applying faster R-CNN for object detection on malaria images." In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 56-61. 2017.

[48] Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "Imagenet: A large-scale hierarchical image database." In 2009 IEEE conference on computer vision and pattern recognition, pp. 248-255. Ieee, 2009.

[49] Bibin, Dhanya, Madhu S. Nair, and P. Punitha. "Malaria parasite detection from peripheral blood smear images using deep belief networks." IEEE Access 5 (2017): 9099- 9108.

[50] Lee, Honglak, Roger Grosse, Rajesh Ranganath, and Andrew Y. Ng. "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations." In Proceedings of the 26th annual international conference on machine learning, pp. 609-616. ACM, 2009.

[51] Salakhutdinov, Ruslan, and Geoffrey Hinton. "Deep Boltzmann machines." In Artificial intelligence and statistics, pp. 448-455. 2009.

[52] Carreira-Perpinan, Miguel A., and Geoffrey E. Hinton. "On contrastive divergence learning." In Aistats, vol. 10, pp. 33-40. 2005.

[53] Razzak, Muhammad Imran, and Saeeda Naz. "Microscopic blood smear segmentation and classification using deep contour aware CNN and extreme machine learning." In 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 801-807. IEEE, 2017.

[54] Kantorov, Vadim, Maxime Oquab, Minsu Cho, and Ivan Laptev. "Contextlocnet: Context-aware deep network models for weakly supervised localization." In European Conference on Computer Vision, pp. 350-365. Springer, Cham, 2016.

[55] Huang, Guang-Bin, Qin-Yu Zhu, and Chee-Kheong Siew. "Extreme learning machine: theory and applications." Neurocomputing 70, no. 1-3 (2006): 489-501.

[56] Mehanian, Courosh, Mayoore Jaiswal, Charles Delahunt, Clay Thompson, Matt Horning, Liming Hu, Travis Ostbye et al. "Computer-automated malaria diagnosis and quantitation using convolutional neural networks." In Proceedings of the IEEE International Conference on Computer Vision, pp. 116-125. 2017.

[57] Var, Esra, and F. Boray Tek. "Malaria Parasite Detection with Deep Transfer Learning." In 2018 3rd International Conference on Computer Science and Engineering (UBMK), pp. 298-302. IEEE, 2018.

[58] Rajaraman, Sivaramakrishnan, Sameer K. Antani, Mahdieh Poostchi, Kamolrat Silamut, Md A. Hossain, Richard J. Maude, Stefan Jaeger, and George R. Thoma. "Pretrained convolutional neural networks as feature extractors toward improved malaria parasite detection in thin blood smear images." PeerJ 6 (2018): e4568.

[59] Shen, Hongda, W. David Pan, Yuhang Dong, and Mohammad Alim. "Lossless compression of curated erythrocyte images using deep autoencoders for malaria infection diagnosis." In 2016 Picture Coding Symposium (PCS), pp. 1-5. IEEE, 2016.

[60]Mohanty, Itishree, P. A. Pattanaik, and Tripti Swarnkar. "Automatic Detection of Malaria Parasites Using Unsupervised Techniques." In International Conference on ISMAC in Computational Vision and Bio-Engineering, pp. 41-49. Springer, Cham, 2018.

[62] Park, Han Sang, Matthew T. Rinehart, Katelyn A. Walzer, Jen-Tsan Ashley Chi, and Adam Wax. "Automated detection of P. falciparum using machine learning algorithms with quantitative phase images of unstained cells." PloS one 11, no. 9 (2016): e0163045.

[63] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097-1105.

[64] L. Yu, H. Chen, Q. Dou, J. Qin, and P.-A. J. I. t. o. m. i. Heng, "Automated melanoma recognition in dermoscopy images via very deep residual networks," vol. 36, no. 4, pp. 994-1004, 2017.

[65] H. Haenssle *et al.*, "Man against machine: diagnostic performance of a deep learning convolutional neural network for dermoscopic melanoma recognition in comparison to 58 dermatologists," vol. 29, no. 8, pp. 1836-1842, 2018.

[66] S. S. Han, M. S. Kim, W. Lim, G. H. Park, I. Park, and S. E. J. J. o. I. D. Chang, "Classification of the Clinical Images for Benign and Malignant Cutaneous Tumors Using a Deep Learning Algorithm," 2018.

[67] U.-O. Dorj, K.-K. Lee, J.-Y. Choi, M. J. M. T. Lee, and Applications, "The skin cancer classification using deep convolutional neural network," pp. 1-16, 2018.

[68] A. Esteva *et al.*, "Dermatologist-level classification of skin cancer with deep neural networks," vol. 542, no. 7639, p. 115, 2017.

[69] www.google.com - The world's information

[70] www.kaggle.com - The world's largest data science community

[71] www.tensorflow.org - open-source machine-learning platform

INDIVIDUAL CONTRIBUTION REPORT

HEALTH DISCERNMENT SYSTEM

ANISH KUMAR
1706200

Abstract: With time there has been a remarkable development in the medical field. Furthermore, people are getting diagnosed with severe complex diseases. Diagnosis is the main part of medical science. Thus, in our project, we have developed an application using deep learning, named- Health Discernment System, that detects diseases based on medical test images. This will not only discard any chances of the wrong diagnosis but will also save much time of the medical professionals as well as the patients.

Individual contribution and findings: Responsible for developing the model for skin cancer detection. Originally, the dataset was collected from Kaggle to create the model, then independent and dependent objects were categorized and the entire dataset was split into train set and test set. Google Colab did the model training and testing. The model's initial accuracy was 75.6 percent, but after passing it through some medium and reducing over-fitting using data science articles, the model's performance greatly improved. Finally, the custom CNN model was able to detect the disease with 84.69 per cent accuracy from a resized image.

Individual contribution to project report preparation: For the project documentation, I wrote chapter 1 (Introduction), chapter 10 (Future Scope) and chapter 9 (Screenshots of the project). In addition I have also provided some code snippets.

Individual contribution for project presentation and demonstration: For the project presentation, I made the introduction slide and the side for the future scope and I will be demonstrating on the same.

Full Signature of Supervisor:
…………………………

Full signature of the student:
…………………………
Anish Kumar

INDIVIDUAL CONTRIBUTION REPORT

HEALTH DISCERNMENT SYSTEM

INDRILA BASAK
1706224

Abstract: With time there has been a remarkable development in the medical field. Furthermore, people are getting diagnosed with severe complex diseases. Diagnosis is the main part of medical science. Thus, in our project, we have developed an application using deep learning, named- Health Discernment System, that detects diseases based on medical test images. This will not only discard any chances of the wrong diagnosis but will also save much time of the medical professionals as well as the patients.

Individual contribution and findings: I have done research work based on the topics related to the project and the papers holding immense importance for the project. I analyzed the research papers and compared their outcomes. In the initial days, I tried to understand what are the relevant materials for this project such that it can be implemented as a web application. The mind behind the idea of implementing the MVC architecture while designing the system- as I thought it would be very time efficient.

Individual contribution to project report preparation: In the project report, I have written chapter 5 and 7 (System Design and Project Planning). In chapter 5 I have mentioned about how MVC architecture has been implemented and in chapter 7, I have written about the planning that was done before the project was started. I also made the flowchart of the proposed method (figure 5.1.1).

Individual contribution for project presentation and demonstration: For the project presentation, I made the result, preview of the application and the conclusion slides. I will demonstrate on what accuracies we acquired. I will also talk about how the application will be visible and used by the users.

Full Signature of Supervisor: Full signature of the student:

…………………………… ………………………………..

Indrila Basak

INDIVIDUAL CONTRIBUTION REPORT

HEALTH DISCERNMENT SYSTEM

PRANAV JAIN
1706243

Abstract: With time there has been a remarkable development in the medical field. Furthermore, people are getting diagnosed with severe complex diseases. Diagnosis is the main part of medical science. Thus, in our project, we have developed an application using deep learning, named- Health Discernment System, that detects diseases based on medical test images. This will not only discard any chances of the wrong diagnosis but will also save much time of the medical professionals as well as the patients.

Individual contribution and findings: In this project my contribution was in designing the Pneumonia detection model. The image dataset for this particular part of the work was collected from Kaggle and it was already divided into train and test folder. These folders were further segregated into two distinct classes of Normal and Pneumonia. I have divided the available test dataset in the ratio of 50:50 to generate separate validation data. The reason behind this is to validate the model in an entirely new data that has not been previously used by the model. Initially the model was built with fewer number of layers but the accuracy was not satisfactory, so the model was modified accordingly by adding more convolutional layers to enhance the performance and increase the accuracy. The final model consisted of 10 convolutional layers and 2 fully connected layers along with the activation functions- 'relu' in the hidden layers and 'sigmoid' in the outer layer. In the image augmentation part, the images were transformed both manually and automatically to not only avoid overfitting but also to compare the performance of the model by giving both kind of augmented images seperately.
I have also tried using different optimizers for compiling the model and found 'adam' and 'rmsprop' to be the most suitable ones for this work. The loss function used in this work was 'binary_crossentropy' as there were two output class. As the classes (Normal and Pneumonia) were imbalanced i.e. one of the classes had more images than the other, so the 'class_weight' technique was used to assign respective weights to the corresponding two classes. The final model achieved the highest accuracy of 90.06% in classifying between normal and pneumonia images. Some of the key findings of my part of the work were- manual image augmentation and class-weight balancing technique.

Individual contribution to project report preparation: My role in the project documentation was to write chapter 3, 4 and 10. In chapter 3, I have written the software requirement analysis (SRS), where I have mentioned about the scope of the product, it's functionalities and requirements and in chapter 4, I have written about the functional and behavioral requirements of our product. For chapter 10 I have written the conclusion part. I also made the use case view of the system (figure 4.1.3).

Individual contribution for project presentation and demonstration: For the presentation, I made the slides on motivation, software requirement specifications and requirement analysis. There I wrote about what motivated us to take up the proposed project, the scope and functionality of the product, it's characteristics, operating environment etc. I will also be demonstrating on the mentioned topics.

Full Signature of Supervisor:
…………………………

Full signature of the student:
……………………………..
Pranav Jain

INDIVIDUAL CONTRIBUTION REPORT

HEALTH DISCERNMENT SYSTEM

RITAM BARIK
1706254

Abstract: With time there has been a remarkable development in the medical field. Furthermore, people are getting diagnosed with severe complex diseases. Diagnosis is the main part of medical science. Thus, in our project, we have developed an application using deep learning, named- Health Discernment System, that detects diseases based on medical test images. This will not only discard any chances of the wrong diagnosis but will also save much time of the medical professionals as well as the patients.

Individual contribution and findings: In this particular project I have trained a neural network with thousands of histology images where there was 11697 images in train class and 2584 images in test class which further was divided into cancer and non_cancer classes. In this model I have used 6 convolutional layers followed by 1 fully connected layer. After that 'adam' was used as optimizer and 'binary cross entropy' was used as loss function in the compilation.I had trouble fixing the hyper parameters-that was goal size, batch size, activation type, etc. I've done a lot of cross-validation to get the hyper parameters optimum. The proposed technique obtained an accuracy of 87 percent and was jointly rewarded for the role of classifying breast cancer from histology images as one of the highest performing algorithms.

Individual contribution to project report preparation: I have written the abstract of the project report to describe the purpose of the study, investigation and major aspects of the entire project in a prescribed sequence. Furthermore I have gone through some research papers of related work and written literature review of the project for giving comprehensive summary of previous research on this project.

Individual contribution for project presentation and demonstration: Responsible for writing the related works, reference and about the diseases. I will be demonstrating about the diseases and the related works.

Full Signature of Supervisor: ……………………………

Full signature of the student: ……………………………..
Ritam Barik

INDIVIDUAL CONTRIBUTION REPORT

HEALTH DISCERNMENT SYSTEM

RITURAJ SAHA
1706256

Abstract: With time there has been a remarkable development in the medical field. Furthermore, people are getting diagnosed with severe complex diseases. Diagnosis is the main part of medical science. Thus, in our project, we have developed an application using deep learning, named- Health Discernment System, that detects diseases based on medical test images. This will not only discard any chances of the wrong diagnosis but will also save much time of the medical professionals as well as the patients.

Individual contribution and findings: Responsible for developing the malaria detection model and front-end of the application. To develop the model, initially, the dataset was collected from Kaggle, then independent and dependent objects were identified then the whole dataset was divided into train set and test set. Model training and testing were done on Google Colab. The initial accuracy of the model was 88.67%, after going through some medium and towards data science articles overfitting was reduced, the performance of the model improved. Finally, the custom CNN model was capable to detect the disease from a resized image with an accuracy of 95.22%.
The whole front-end was developed using tkinter. At first, a single GUI based window was created for a single disease, then a separate GUI menu-based window was developed to select any disease detection program and then both were connected. The whole program is converted to an executable file(.exe) to optimize the disease detection time. Moreover, a requirements.exe file is also created to check if the system satisfies all the requirements to run the application and the whole application along with its required files is also compressed into a single executable file.

Individual contribution to project report preparation: Responsible for preparing chapter 6: System Testing and chapter 8: Implementation of the Minor project thesis.

Individual contribution for project presentation and demonstration: Responsible for preparing the system design in the project presentation and will be demonstrating on the same.

Full Signature of Supervisor: ……………………………….

Full signature of the student: ………………………………..
Rituraj Saha