**A PROJECT REPORT**

**ON**

# MEDICAL IMAGE CLASSIFICATON USING CNN

**Submitted in partial fulfillment of requirements**
**for the award of the degree of**

## MASTER OF COMPUTER APPLICATIONS

**By**

**N. SRINATH REDDY**
**(21091F0050)**

**Under the esteemed guidance of**
**Mrs. BANA SWETHA M. Tech, (Ph.D.)**

**Assistant Professor, Dept. of CSE**


**MASTER OF COMPUTER APPLICATIONS**

**RAJEEV GANDHI MEMORIAL COLLEGE OF ENGINEERING & TECHNOLOGY**

**(AUTONOMOUS)**

AFFILIATED TO J.N.T UNIVERSITY ANANTAPUR. ACCREDITED BY NBA (TIER-1) & NAAC OF UGC. NEW DELHI, WITH A+ GRADE

RECOGNIZED UGC-DDU KAUSHAL KENDRA

NANDYAL-518501, (Estd-1995)

**YEAR: 2022-2023**

# Rajeev Gandhi Memorial College of Engineering &Technology

**(AUTONOMOUS)**

AFFILIATED TO J.N.T UNIVERSITY ANANTAPUR. ACCREDITED BY NBA (TIER-1) & NAAC OF UGC. NEW DELHI, WITH A+ GRADE
RECOGNIZED UGC-DDU KAUSHAL KENDRA
NANDYAL-518501, (Estd-1995)

**(ESTD – 1995)**

## MASTER OF COMPUTER APPLICATIONS

## CERTIFICATE

This is to certify that NARALA SRINATH REDDY (*21091F0050*), of MCA IV semester, has carried out the main project work entitled "MEDICAL IMAGE CLASSIFICATION USING CNN" under the supervision and guidance of BANA SWETHA, Assistant Professor, CSE Department, in partial fulfillment of the requirements for the award of Degree of Master of Computer Applications from Rajeev Gandhi Memorial College of Engineering & Technology (Autonomous), Nandyal is a bonafied record of the work done by them during 2022-2023.

**Project Guide**

**Mrs. BANA SWETHA M. Tech, (Ph.D.)**

**Assistant Professor,**

**Dept. of CSE**

**Head of the department**

**Dr. K. SUBBA REDDY, M. Tech, Ph.D.**

**Professor, Dept. of CSE**

**External Examiner**

**Place:** Nandyal
**Date:**

# PROJECT COMPLETION CERTIFICATE

This is to certify that, **Mr. Narala Srinath Reddy** studying **MCA** bearing the **Reg. No: 21091F0050** from "**Rajeev Gandhi Memorial College of Engineering and Technology, NANDYAL**" has successfully completed his project entitled "**Medical Image Classification Using CNN**" as part of his academic Project curriculum in our organization.

He has done this project using **Python** during the period from **March 2023 to August 2023.** He has done project in **Young Minds Technology Solutions Pvt Ltd., Tirupati.**

He has completed the project well with in the time frame. He is sincere hardworking and his conduct during the project is commendable.

We wish him all the best in his future endeavors.

Young Minds Technology Solutions Pvt Ltd.

**Authorized Signature.**

# ACKNOWLEDGEMENT

The satisfaction that accomplishes completion of any task would be incomplete without the mention of people who made it possible and whose encouragement and guidance has been a source of inspiration throughout the course of the project I want to express my sincere gratitude to my Project Guide **Mrs. B. SWETHA M. Tech**. Asst. prof. in CSE Department Under his guidance, I have learned a lot about different aspects in research, including finding a good research problem.

Once again, I am extremely thankful to my project guide **Mrs. B. SWETH M.Tech**. Head of the Department who was always a motivational power to me.

I express my profound sense of gratitude to **Dr. K. SUBBA REDDY, Head of the Department of CSE** for inspiring and guiding us throughout the project.

I express my profound sense of gratitude to **Dr. T. JAYA CHANDRA PRASAD, Principal, and Rajeev Gandhi Memorial (autonomous) College of Engineering and Technology.**

**N. SRINATH REDDY**

**(21091F0050)**

# DECLARATION

I hereby declare that the project report entitled**" Medical Image classification Using CNN"**, Under the guidance of **Mrs. B. Swetha, M. TECH.,** is submitted for the award of degree of bachelor of technology in **MASTER OF COMPUTER APPLICATIONS**.

This is a record of bonafide work carried out by us and the results embodied in this project have not been reproduced or copied from any source. The results embodied in this project have not been submitted to any other university or institute for the award of any other Degree.

**N. Srinath Reddy**

**(21091F0050)**

# CONTENTS

## LIST OF FIGURES

# ABSTRACT

Deep learning is one of the most unexpected machine learning techniques which is being used in many applications like image classification, image analysis, clinical archives and object recognition. With an extensive utilization of digital images as information in the hospitals, the archives of medical images are growing exponentially. Digital images play a vigorous role in predicting the patient disease intensity and there are vast applications of medical images in diagnosis and investigation. Due to recent developments in imaging technology, classifying medical images in an automatic way is an open research problem for researchers of computer vision. For classifying the medical images according to their relevant classes, a most suitable classifier is most important. Where we are proposing our model where the algorithm is trained for classifying medical images by deep learning technique. A pre-trained deep convolution neural network (GoogleNet) is used that which can classifies the various medical images for various body organs. This method of image classification is beneficial to predict the appropriate class or category of unknown images. The results of the experiment exhibit that our method is best suited to classify various medical images.



**Keywords:** Medical image classification, pre-trained DCNN, convolution neural network, deep learning

# 1. INTRODUCTION

Image classification is the primary domain, in which deep neural networks play the most important role of medical image analysis. The image classification accepts the given input images and produces output classification for identifying whether the disease is present or not.

In our model we mainly classify the different types of organs and predict the accuracy of the most imperative problems faced in the domain area of image recognition is the classification of medical images. The major intention of medical image classification is to classify medical images into several elements to assist medical practitioners or physicists in diagnosing disease. Hence, medical image classification is split into two steps. The first and foremost step of medical image classification is to extract the essential features from the acquired input image. The second step in medical image classification is utilizing the features to construct models that classify the image data set. In the recent past, medical practitioners customarily utilized their specialized experience to extract features so that classification of medical images could be performed into several classes. However, this manual medical image classification was found to be highly cumbersome and time consuming.

Medical image classification involves the process of segregating medical-related information into a useful form. Classification of medical images is based on placing image pixels with similar values into groups. With the placement of similar values into groups, common pixels are identified and are denoted by these pixels. Hence, a correctly classified image usually denotes the areas on the ground that share specific features as specified in the classification scheme.

Image classification is where a computer can analyse an image and identify the ‘class’ the image falls under. (Or a probability of the image being part of a ‘class’.) A class is essentially a label, for instance, ‘car’, ‘animal’, ‘building’ and so on. For example, you input an image of a sheep. Image classification is the process of the computer analysing the image and telling you it’s a sheep. (Or the probability that it’s a sheep.) For us, classifying images is no big deal. But it’s a perfect example of Moravec’s paradox when it comes to machines. (That is, the things we find easy are difficult for AI.)

Early image classification relied on raw pixel data. This meant that computers would break down images into individual pixels. The problem is that two pictures of the same thing can look very different. They can have different backgrounds, angles, poses, and etcetera. This made it quite the challenge for computers to correctly 'see' and categorise images.

Deep learning is a type of machine learning; a subset of artificial intelligence (AI) that allows machines to learn from data. Deep learning involves the use of computer systems known as neural networks. In neural networks, the input filters through hidden layers of nodes. These nodes each process the input and communicate their results to the next layer of nodes. This repeats until it reaches an output layer, and the machine provides its answer.

There are different types of neural networks based on how the hidden layers work.

Image classification with deep learning most often involves convolutional neural networks, or CNNs. In CNNs, the nodes in the hidden layers don't always share their output with every node in the next layer (known as convolutional layers). Deep learning allows machines to identify and extract features from images. This means they can learn the features to look for in images by analysing lots of pictures. So, programmers don't need to enter these filters by hand.

Image classification has a few uses — and vast potential as it grows in reliability. Here are just a few examples of what makes it useful. Self-driving cars use image classification to identify what's around them. I.e., trees, people, traffic lights and so on. Image classification can also help in healthcare. For instance, it could analyse medical images and suggest whether they classify as depicting a symptom of illness. Or, for example, image classification could help people organise their photo collections.

# 2. LITERATURE REVIEW

**Q. Zhu, B. Du, and P. Yan:** Accurate segmentation of the prostate from magnetic resonance (MR) images provides useful information for prostate cancer diagnosis and treatment. However, automated prostate segmentation from 3D MR images still faces several challenges. The complex background texture and large variation in size, shape and intensity distribution of the prostate itself make segmentation even further complicated. Since large-scale dataset is one of the critical components for the success of deep learning, lack of sufficient training data makes it difficult to fully train complex CNNs. To tackle the above challenges, in this paper boundary-weighted domain adaptive neural network (BOWDA-Net) is proposed. To make the network more sensitive to the boundaries during segmentation, a boundary-weighted segmentation loss (BWL) is proposed.

**Summary:** In this paper boundary-weighted domain adaptive neural network (BOWDA-Net) is proposed. To make the network more sensitive to the boundaries during segmentation, a boundary-weighted segmentation loss (BWL) is proposed. . Furthermore, an advanced boundary-weighted transfer leaning approach is introduced to address the problem of small medical imaging datasets. We evaluate our proposed model on the publicly available MICCAI 2012 Prostate MR Image Segmentation (PROMISE12) challenge dataset.

**Q. Zhu, B. Du, P. Yan, H. Lu, and L. Zhang:** Bladder wall segmentation from Magnetic Resonance (MR) images plays an important role in diagnosis. Since the thickness of the bladder wall is a key indication of bladder cancer. There are several methods that have been used for bladder wall segmentation, such as level sets and Active Shape Model (ASM). However, the weak boundaries, the artifacts inside bladder lumen and the complex background outside the bladder wall make the bladder wall segmentation very challenging. To overcome these difficulties and obtain accurate bladder walls, in this paper, a shape prior constrained particle swarm optimization (SPC-PSO) model is proposed to segment the inner and outer boundaries of the bladder wall. The bladder walls are divided into two categories: strong boundaries and weak boundaries by the proposed model.

**Summary**: In this paper, a shape prior constrained particle swarm optimization (SPC-PSO) model is proposed to segment the inner and outer boundaries of the bladder wall. The bladder walls are divided into two categories: strong boundaries and weak boundaries by the proposed

model. For the strong boundaries, the proposed model can reserve it. For the weak boundaries, the model applies the shape prior to guide the process of segmentation**.**

**Q. Zhu, B. Du, B. Turkbey, P. Choyke, and P. Yan:** Segmentation of the prostate from Magnetic Resonance Imaging (MRI) plays an important role in prostate cancer diagnosis. However, the lack of clear boundary and significant variation of prostate shapes and appearances make the automatic segmentation very challenging. In the past several years, approaches based on deep learning technology have made significant progress on prostate segmentation. However, those approaches mainly paid attention to features and contexts within each single slice of a 3D volume. As a result, this kind of approaches faces many difficulties when segmenting the base and apex of the prostate due to the limited slice boundary information. To tackle this problem, in this paper, we propose a deep neural network with bidirectional convolutional recurrent layers for MRI prostate image segmentation.

**Summary:** A deep neural network is proposed with bidirectional convolutional recurrent layers for MRI prostate image segmentation. In addition to utilizing the interslice contexts and features, the proposed model also treats prostate slices as a data sequence and utilizes the interslice contexts to assist segmentation. The experimental results show that the proposed approach achieved significant segmentation improvement compared to other reported methods.

**K. Kranthi Kumar and T.V. Gopal:** Content Based Image Retrieval (CBIR) is a prominent research area in effective retrieval and management process for large image databases. Which was a bottleneck in reducing semantic gap issue to solve, many approaches have been proposed. Among them, Relevance Feedback (RF) is a technique absorbed into CBIR systems to improve retrieval accuracy using user given feedback. One of the traditional methods to enact relevance feedback is Feature Reweighting (FRW), it is useful technique to enhance retrieval performance based on the acquired feedback from user. The assumption for previous FRW approaches is that the length of feature vectors for images are fixed and use only the information from the set of images send back in the early query result for feature reweighting.

**Summary:** In this article, examined systematically the proposed system with various weight update strategies and compared output retrieval results and proposed a new self-order feature reweighting approach in CBIR to reduce semantic gap using relevance feedback which we experimented with

COREL database with 25 different categories and each category containing 100 number of relevant images.

**R. Ashraf, K.B. Bajwa, and T. Mahmood:** Segmentation of the images is considered as a solution but there isn't any technique which can guarantee the object extraction in a robust way. Another limitation of the segmentation is that, most of the image segmentation techniques are very slow and still their results are not reliable. To overcome these problems a Bandelets transform based image representation technique is presented in this paper, which reliably returns the information about the major objects found in an image. For image retrieval purposes Support Vector Machine are applied and the performance of the system is evaluated on three standard data sets used in the domain of content-based image retrieval.

**Summary:** To overcome the problems which are in the existing method, a Bandelets transform based image representation technique is presented in this paper, which reliably returns the information about the major objects found in an image. For image retrieval purposes Support Vector Machine are applied and the performance of the system is evaluated on three standard data sets used in the domain of content-based image retrieval.

## 2.1 Existing System

This model emphasizes an existing method that which is designed using the machine learning architecture which is used to classify the various medical images. With an extensive utilization of digital images as information in the hospitals, the archives of medical images are growing exponentially. Digital images play a vigorous role in predicting the patient disease intensity and there are vast applications of medical images in diagnosis and investigation. To make this in the easier way Support Vector Machine (SVM) is used that which can classifies the various medical images for various body organs. The block diagram of the existing method is shown in the below figure.

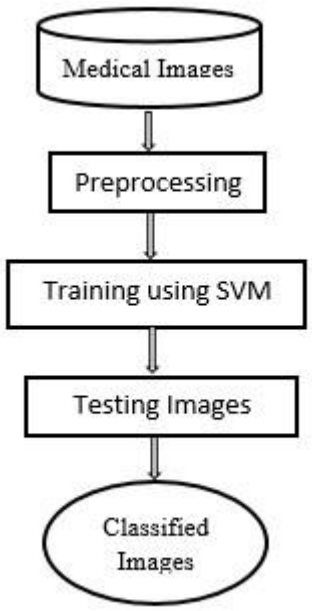



**Fig 1. Block Diagram of Existing Method**

### 2.2.1 Disadvantages:

- Low efficiency.
- Time consuming.
- High complexities.
- No accurate classification

## 2.2 Proposed system.

The proposed model emphasizes a deep network architecture which is used to classify the various medical images. With an extensive utilization of digital images as information in the hospitals, the archives of medical images are growing exponentially. Digital images play a vigorous role in predicting the patient disease intensity and there are vast applications of medical images in diagnosis and investigation. Hence, we are proposing our model where the algorithm is trained for classifying medical images by deep learning technique. A pre-trained deep convolution neural network (GoogleNet) is used that which can classifies the various medical images for various body organs. The block diagram of the proposed model is shown in the below figure.

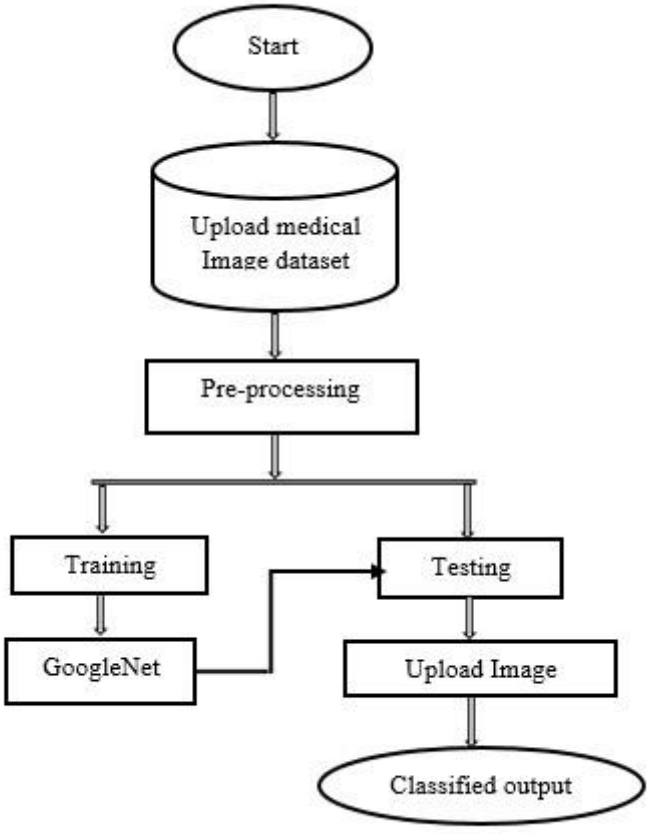



**Fig 2. Block Diagram of Proposed Method**

### 2.2.1 Advantages

- High efficiency.
- Time Saving.
- Inexpensive.
- Low complexities.

### 2.2.2 APPLICATIONS

Useful for the hospitals to detect brain strokes using this application.

# 3. SYSTEM MODEL

## 3.1 Architecture

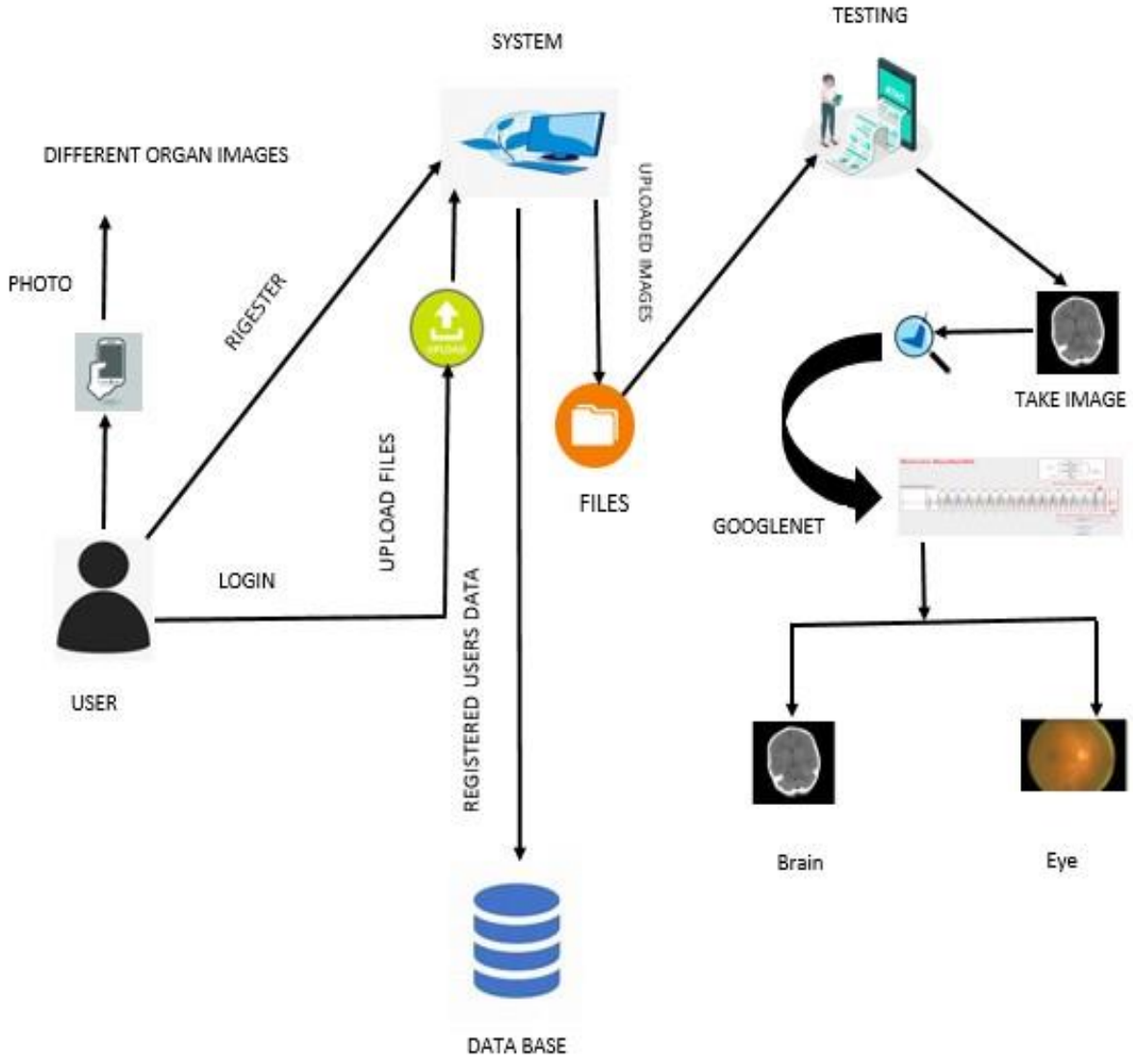



Fig 3. ARCHITECTURE

## 3.2 MODULES

## System User:

### 2.5.1. System:

**Create Dataset:**

The dataset containing images of the desired objects to be recognize is split into training and testing dataset with the test size of 20-30%.

**Pre-processing:**

Resizing and reshaping the images into appropriate format to train our model.

**Training:**

Use the pre-processed training dataset is used to train our model using CNN algorithm.

### 2.5.2. User:

**Register**

The user needs to register and the data stored in MySQL database.

**Login**

A registered user can login using the valid credentials to the website to use a application.
About-Project

In this application, we have successfully created an application which takes to classify the images.

**Upload Image**

The user has to upload an image which needs to be classify the images.

**Prediction**

The results of our model is displayed as either Rice Blast, Leaf Blight, Healthy & Brown Spot.

**Logout**

Once the prediction is over, the user can logout of the application.

## 3.3 SYSTEM NARRATION

## Software Development Life Cycle – SDLC:

In our project we use waterfall model as our software development cycle because of its step-by-step procedure while implementing.

- **Requirement Gathering and analysis** − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** − The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** − With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.

- **Maintenance** − There are some issues which come up in the client environment. To fix those issues, patches are released. Also, to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

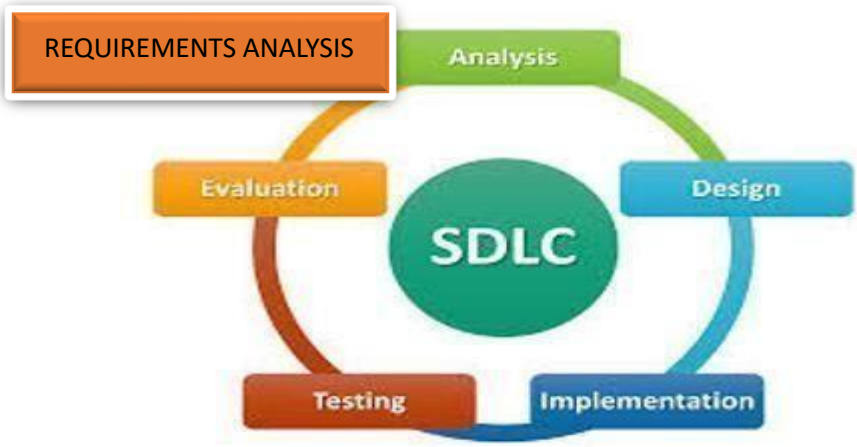


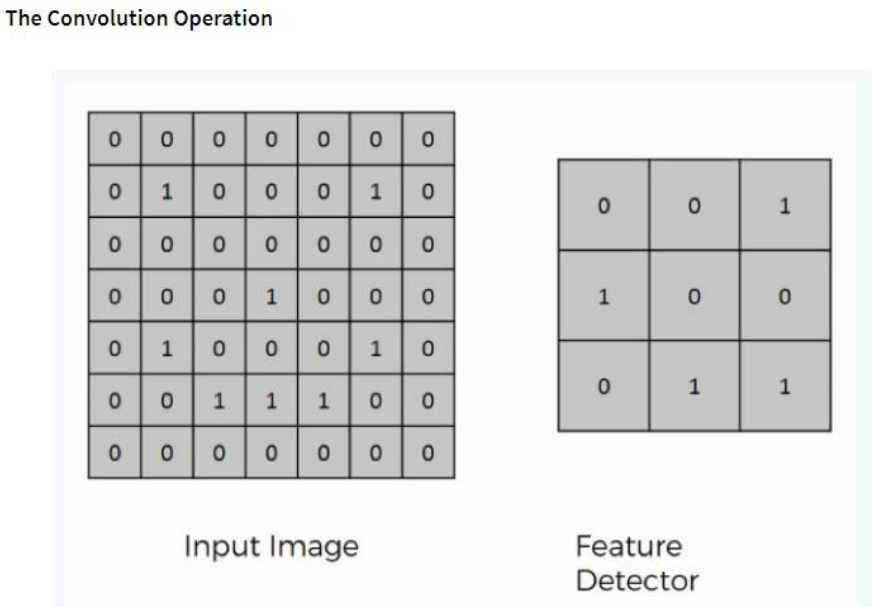

Fig 4. Life Cycle of SDLC Spiral Model

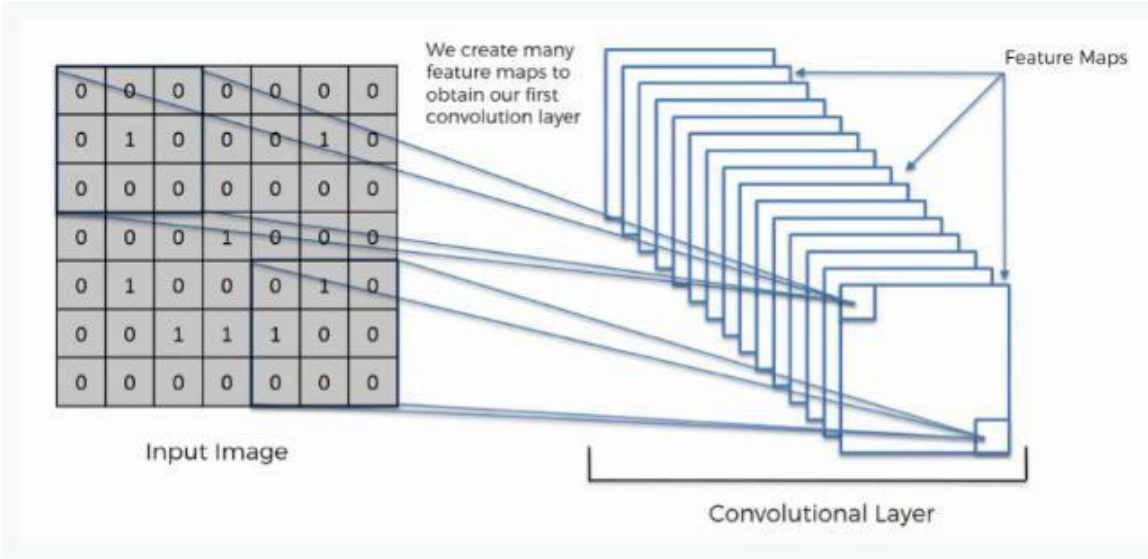
## 3.4 ALGORITHMS

### 3.4.1 Convolutional Neural Network

#### Step1: Convolutional Operation

The first building block in our plan of attack is convolution operation. In this step, we will touch on feature detectors, which basically serve as the neural network's filters. We will also discuss feature maps, learning the parameters of such maps, how patterns are detected, the layers of detection, and how the findings are mapped out.

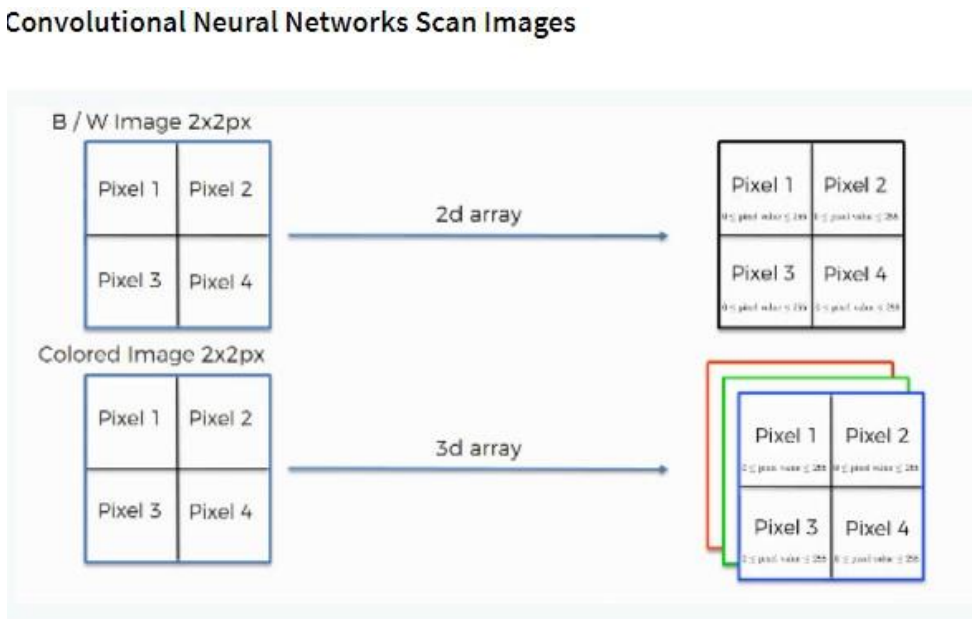





**Step (1b): ReLU Layer**

The second part of this step will involve the Rectified Linear Unit or Relook. We will cover Relook layers and explore how linearity functions in the context of Convolutional Neural Networks.

Not necessary for understanding CNNs, but there's no harm in a quick lesson to improve your skills.



**Step 2: Pooling Layer**

In this part, we'll cover pooling and will get to understand exactly how it generally works. Our nexus here, however, will be a specific type of pooling; max pooling. We'll cover various approaches, though, including mean (or sum) pooling. This part will end with a demonstration made using a visual interactive tool that will definitely sort the whole concept out for you.

**Step 3: Flattening**

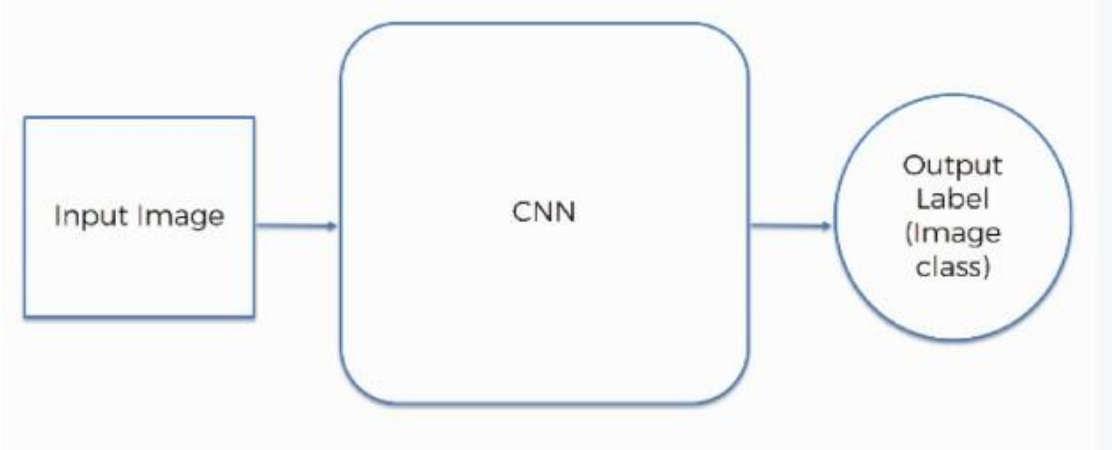
This will be a brief breakdown of the flattening process and how we move from pooled to flattened layers when working with Convolutional Neural Networks.

**Step 4: Full Connection**

In this part, everything that we covered throughout the section will be merged together. By learning this, you'll get to envision a fuller picture of how Convolutional Neural Networks operate and how the "neurons" that are finally produced learn the classification of images.

**Summary**

In the end, we'll wrap everything up and give a quick recap of the concept covered in the section. If you feel like it will do you any benefit (and it probably will), you should check out the extra tutorial in which Soft as and Cross-Entropy are covered. It's not mandatory for the course, but you will likely come across these concepts when working with Convolutional Neural Networks and it will do you a lot of good to be familiar with them.



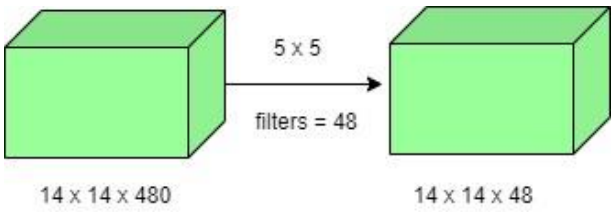

### 3.4.2 Google Net:

Google Net (or Inception V1) was proposed by research at Google (with the collaboration of various universities) in 2014 in the research paper titled "Going Deeper with Convolutions". This architecture was the winner at the ILSVRC 2014 image classification challenge. It has provided a significant decrease in error rate as compared to previous winners AlexNet (Winner of ILSVRC 2012) and ZF-Net (Winner of ILSVRC 2013) and significantly less error rate than VGG (2014 runner up). This architecture uses techniques such as 1×1 convolutions in the middle of the architecture and global average pooling.

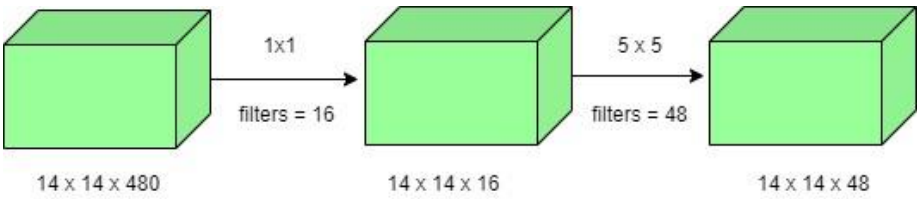
### Features of GoogleNet:

The GoogleNet architecture is very different from previous state-of-the-art architectures such as AlexNet and ZF-Net. It uses many different kinds of methods such as 1×1 convolution and global average pooling that enables it to create deeper architecture. In the architecture, we will discuss some of these methods:

- **1×1 convolution:** The inception architecture uses 1×1 convolution in its architecture. These convolutions used to decrease the number of parameters (weights and biases) of the architecture. By reducing the parameters, we also increase the depth of the architecture. Let's look at an example of a 1×1 convolution below:
- For Example, If we want to perform 5×5 convolution having 48 filters without using 1×1 convolution as intermediate:




- Total Number of operations: (14 x 14 x 48) x (5 x 5 x 480) = 112.9 M • With 1×1 convolution:




- (14 x 14 x 16) x (1 x 1 x 480) + (14 x 14 x 48) x (5 x 5 x 16) = 1.5M + 3.8M = 5.3M which is much smaller than 112.9M.

**Global Average Pooling:**

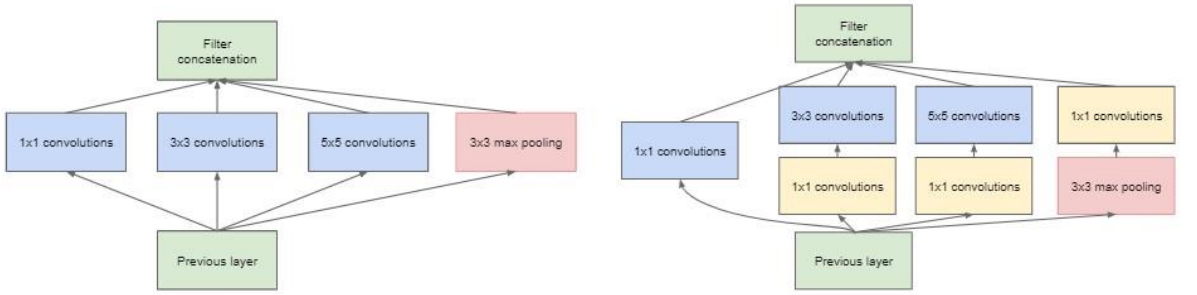
In the previous architecture such as AlexNet, the fully connected layers are used at the end of the network. These fully connected layers contain the majority of parameters of many architectures that causes an increase in computation cost.

In GoogleNet architecture, there is a method called global average pooling is used at the end of the network. This layer takes a feature map of 7×7 and averages it to 1×1. This also decreases the number of trainable parameters to 0 and improves the top-1 accuracy by 0.6%.

**Inception Module:**

The inception module is different from previous architectures such as AlexNet, ZF-Net. In this architecture, there is a fixed convolution size for each layer.

In the Inception module 1×1, 3×3, 5×5 convolution and 3×3 max pooling performed in a parallel way at the input and the output of these are stacked together to generated final output. The idea behind that convolution filters of different sizes will handle objects at multiple scale better.




(a) Inception module, naïve version (b) Inception module with dimension reductions

**Auxiliary Classifier for Training:**

Inception architecture used some intermediate classifier branches in the middle of the architecture, these branches are used during training only. These branches consist of a 5×5 average pooling layer with a stride of 3, a 1×1 convolutions with 128 filters, two fully connected layers of 1024 outputs and 1000 outputs and a SoftMax classification layer. The generated loss of these layers added to total loss with a weight of 0.3. These layers help in combating gradient vanishing problem and also provide regularization.

## 3.5 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful insists the modeling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

## Goals:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

**Use Case Diagram:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

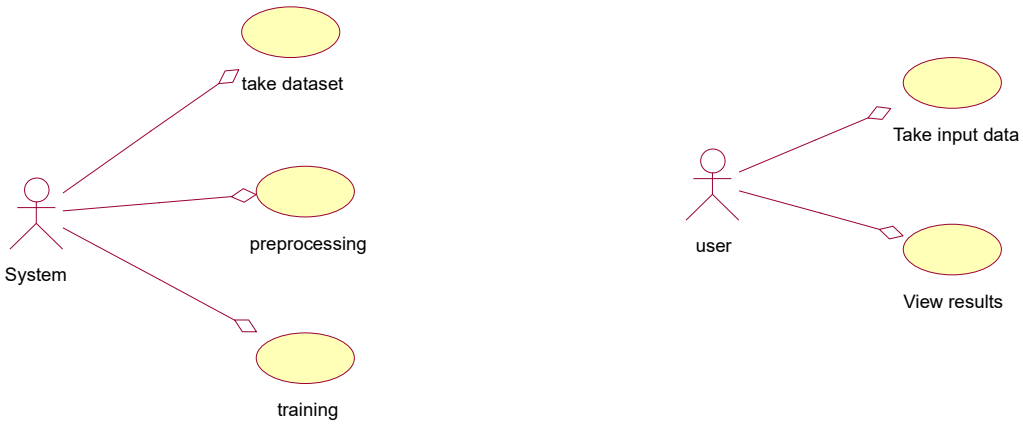



**Fig 5. Use Case Diagram**

## Class Diagram:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

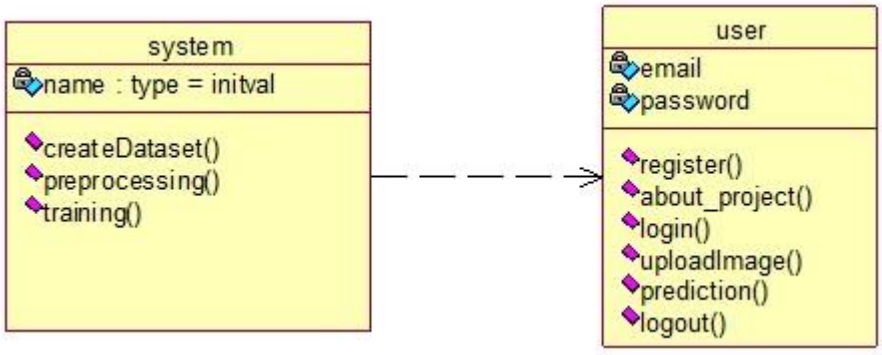



**Fig 6. Class Diagram**

## Sequence Diagram:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and time diagrams.

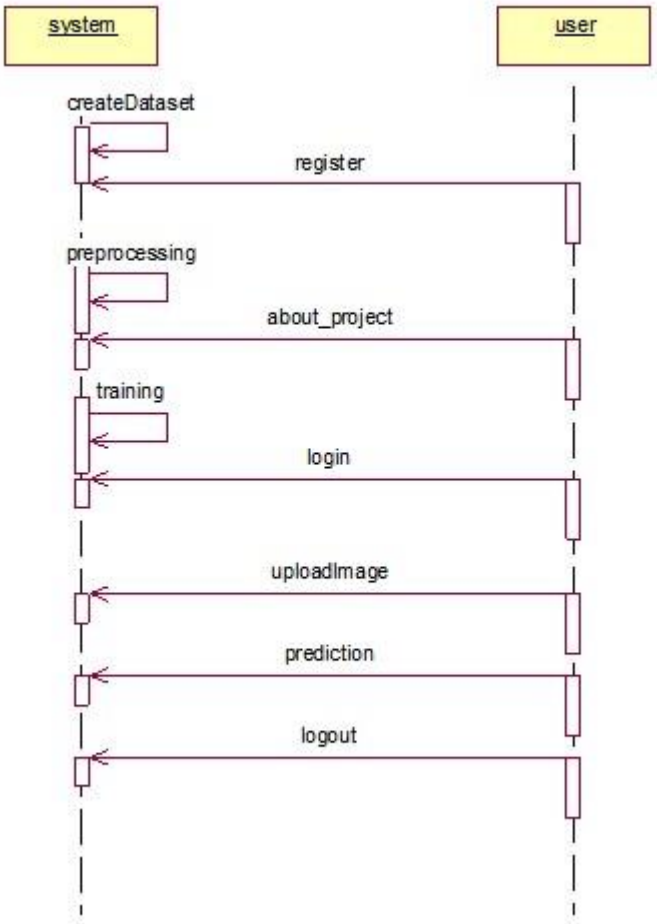



**Fig 7 Sequence Diagram**

## Collaboration Diagram:

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.

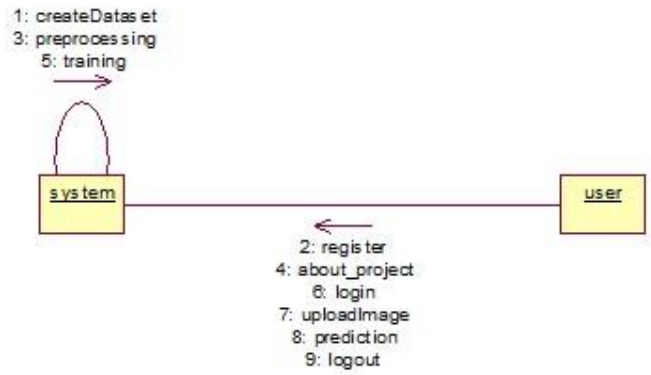



**Fig 8 Collaboration Diagram**

## Activity Diagram:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

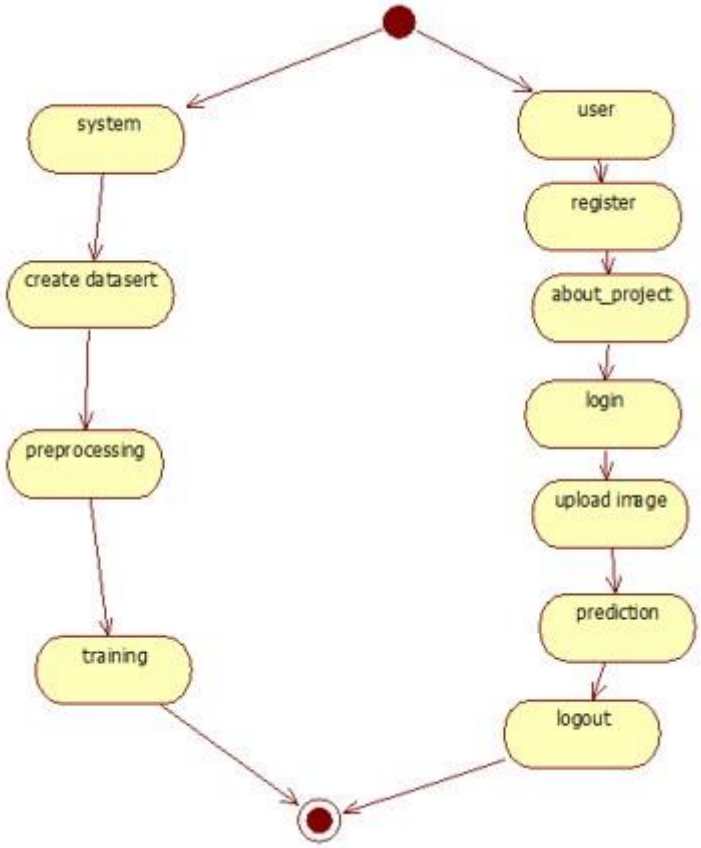



**Fig 9. Activity Diagram**

## Component diagram

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical **c**omponents in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required functions is covered by planned development.

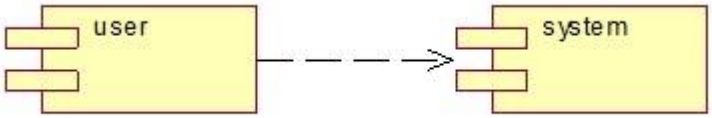



**Fig 10. Component Diagram**

## ER Diagram:

An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let's have a look at a simple ER diagram to understand this concept.

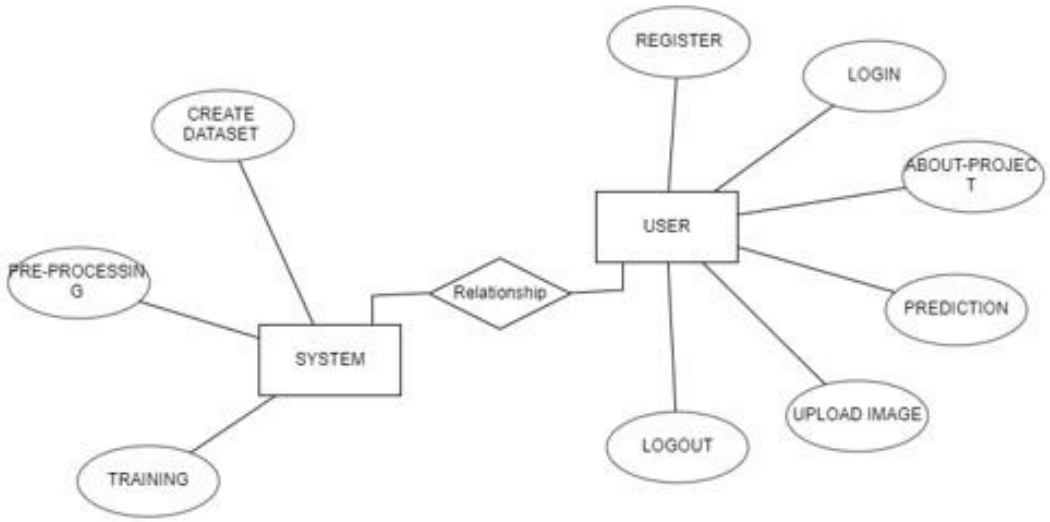
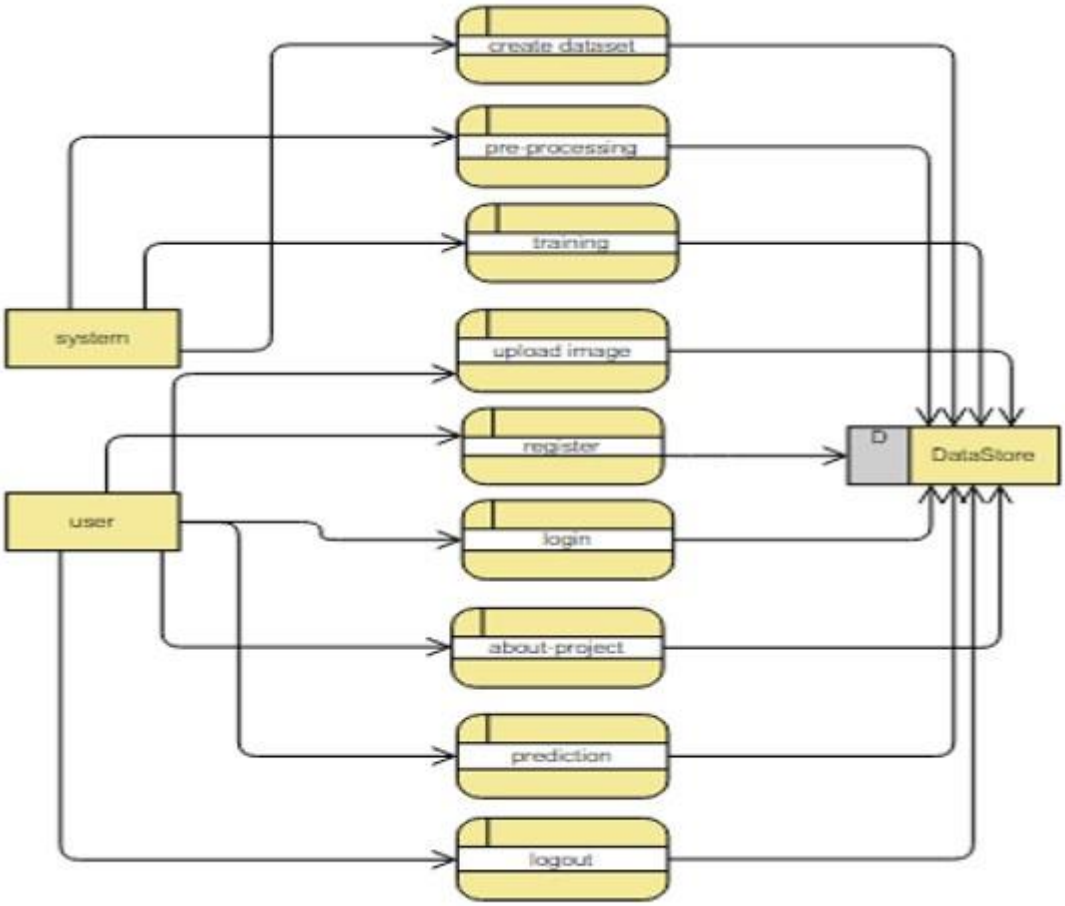



**Fig 11. ER Diagram**

## DFD Diagram:

A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.



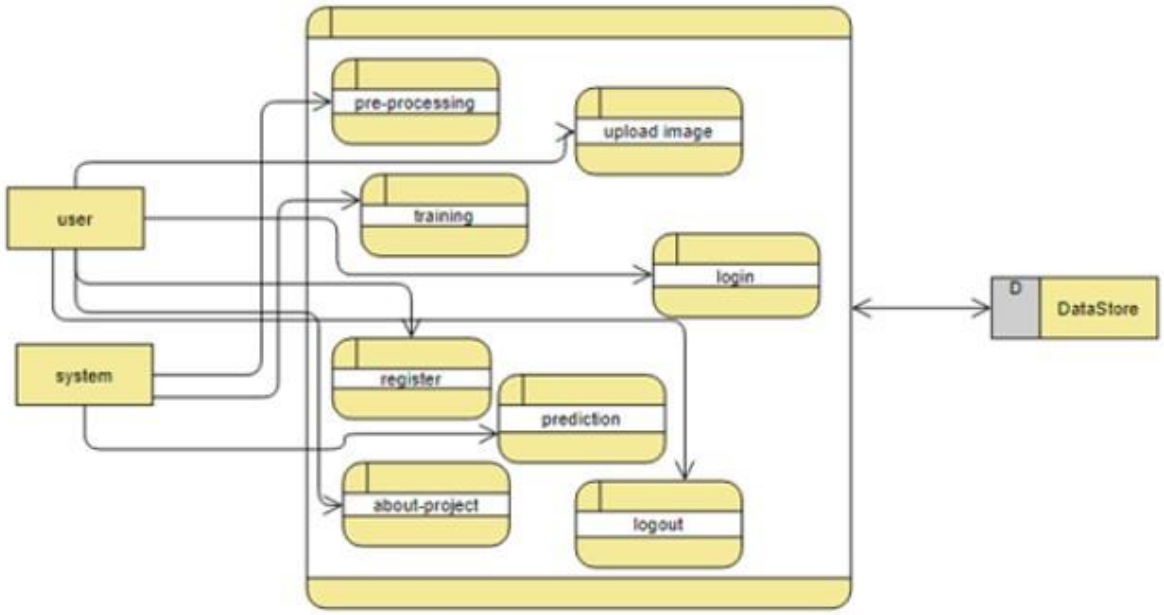



**Fig 12. DFD Diagrams**

## Deployment diagram:

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware's used to deploy the application.


**Fig 13. Deployment diagram**

## 3.6 FEASIBILITY STUDY

The feasibility of the project is analysed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- Economical Feasibility
- Technical Feasibility
- Social Feasibility

### Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

### Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the

users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

## 3.7 HARDWARE & SOFTWARE REQUIREMENTS

### HARDWARE REQUIREMENTS

- Processor : I3/Intel Processor
- Hard Disk : 128GB
- RAM : 8Gb

### SOFTWARE REQUIREMENTS

- Operating System : Windows 7/8/10 .
- Server side Script : HTML, CSS & JS.
- IDE : PyCharm.
- Libraries Used : NumPy, IO, OS, Flask, keras.
- Technology : Python 3.6+.

# 4. IMPLEMENTATION

## Technologies Used:

## 4.1 Python:

## What is a Script?

Up to this point, I have concentrated on the interactive programming capability of Python. This is a very useful capability that allows you to type in a program and to have it executed immediately in an interactive mode

## Scripts are reusable

Basically, a script is a text file containing the statements that comprise a Python program. Once you have created the script, you can execute it over and over without having to retype it each time.

## Scripts are editable

Perhaps, more importantly, you can make different versions of the script by modifying the statements from one file to the next using a text editor. Then you can execute each of the individual versions. In this way, it is easy to create different programs with a minimum amount of typing.

You will need a text editor

Just about any text editor will suffice for creating Python script files.

You can use Microsoft Notepad, Microsoft WordPad, Microsoft Word, or just about any word processor if you want to.

## Difference between a script and a program

## Script:

Scripts are distinct from the core code of the application, which is usually written in a different language, and are often created or at least modified by the end-user. Scripts are often interpreted from source code or byte code, whereas the applications they control are traditionally compiled to native machine code.

## Program:

The program has an executable form that the computer can use directly to execute the instructions. The same program in its human-readable source code form, from which executable programs are derived (e.g., compiled)

## Python:

What is Python? Chances you are asking yourself this. You may have found this book because you want to learn to program but don't know anything about programming languages. Or you may have heard of programming languages like C, C++, C#, or Java and want to know what Python is and how it compares to "big name" languages. Hopefully I can explain it for you.

## Python concepts

If you're not interested in the how's and whys of Python, feel free to skip to the next chapter. In this chapter I will try to explain to the reader why I think Python is one of the best languages available and why it's a great one to start programming with.

• Open-source general-purpose language.

• Object Oriented, Procedural, Functional

• Easy to interface with C/ObjC/Java/Fortran

• Easy-is to interface with C++ (via SWIG)

• Great interactive environment

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

• Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

• Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

• Python is Object-Oriented − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

• Python is a Beginner's Language − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

## 4.2 History of Python:

- Python was developed by Guido van Possum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.
- Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, Smalltalk, and UNIX shell and other scripting languages.
- Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).
- Python is now maintained by a core development team at the institute, although Guido van Possum still holds a vital role in directing its progress.

### 4.3 Python Features:

Python's features include −

- Easy-to-learn − Python has few keywords, simple structure, and a clearly defined Syntax. This allows the student to pick up the language quickly.
- Easy-to-read − Python code is more clearly defined and visible to the eyes.
- Easy-to-maintain − Python's source code is fairly easy-to-maintain.
- A broad standard library − Python's bulk of the library is very portable and Cross platform compatible on UNIX, Windows, and Macintosh.
- Interactive Mode − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- Portable − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- Extendable − you can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- Databases − Python provides interfaces to all major commercial databases.
- GUI Programming − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

## Dynamic vs. Static

Types Python is a dynamic-typed language. Many other languages are static typed, such as C/C++ and Java. A static typed language requires the programmer to explicitly tell the computer what type of “thing” each data value is. For example, in C if you had a variable that was to contain the price of something, you would have to declare the variable as a “float” type.

This tells the compiler that the only data that can be used for that variable must be a floating-point number, i.e., a number with a decimal point. If any other data value was assigned to that variable, the compiler would give an error when trying to compile the program.

Python, however, doesn't require this. You simply give your variables names and assign values to them. The interpreter takes care of keeping track of what kinds of objects your program is using. This also means that you can change the size of the values as you develop the program. Say you have another decimal number (a.k.a. a floating-point number) you need in your program.

With a static typed language, you have to decide the memory size the variable can take when you first initialize that variable. A double is a floating-point value that can handle a much larger number than a normal float (the actual memory sizes depend on the operating environment).

If you declare a variable to be a float but later on assign a value that is too big to it, your program will fail; you will have to go back and change that variable to be a double. With Python, it doesn't matter. You simply give it whatever number you want and Python will take care of manipulating it as needed. It even works for derived values.

For example, say you are dividing two numbers. One is a floating-point number and one is an integer. Python realizes that it's more accurate to keep track of decimals so it automatically calculates the result as a floating-point number

## 4.4 Variables:

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory. Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to variables, you can store integers, decimals or characters in these variables.

## 4.5 Standard Data Types:

The data stored in memory can be of many types. For example, a person's age is stored as a numeric value and his or her address is stored as alphanumeric characters. Python has various standard data types that are used to define the operations possible on them and the storage method for each of them.

## Python Numbers:

Number data types store numeric values. Number objects are created when you assign a value to them

## Python Strings:

Strings in Python are identified as a contiguous set of characters represented in the quotation marks. Python allows for either pairs of single or double quotes. Subsets of strings can be taken using the slice operator ([] and [:]) with indexes starting at 0 in the beginning of the string and working their way from -1 at the end.

## Python Lists:

Lists are the most versatile of Python's compound data types. A list contains items separated by commas and enclosed within square brackets ([]). To some extent, lists are similar to arrays in C. One difference between them is that all the items belonging to a list can be of different data type. The values stored in a list can be accessed using the slice operator ([] and [:]) with indexes starting at 0 in the beginning of the list and working their way to end -1. The plus (+) sign is the list concatenation operator, and the asterisk (*) is the repetition operator.

## Python Tuples:

A tuple is another sequence data type that is similar to the list. A tuple consists of a number of values separated by commas. Unlike lists, however, tuples are enclosed within parentheses. The main differences between lists and tuples are: Lists are enclosed in brackets ([]) and their elements and size can be changed, while tuples are enclosed in parentheses (()) and cannot be updated. Tuples can be thought of as read-only lists.

## Python Dictionary:

Python's dictionaries are kind of hash table type. They work like associative arrays or hashes found in Perl and consist of key-value pairs. A dictionary key can be almost any Python type, but are usually numbers or strings. Values, on the other hand, can be any arbitrary Python object. Dictionaries are enclosed by curly braces ({}) and values can be assigned and accessed using square braces ([]).

**Different modes in python:**

Python has two basic modes: normal and interactive.

The normal mode is the mode where the scripted and finished .pie files are run in the Python interpreter. Interactive mode is a command line shell which gives immediate feedback for each statement, while running previously fed statements in active memory. As new lines are fed into the interpreter, the fed program is evaluated both in part and in whole

## 4.6 Python libraries:

1. Requests. The most famous http library written by Kenneth remits. It's a must have for every python developer.
2. Scrappy. If you are involved in web scraping then this is a must have library for you. After using this library, you won't use any other.
3. Python. A guy toolkit for python. I have primarily used it in place of tinder. You will really love it.
4. Pillow. A friendly fork of PIL (Python Imaging Library). It is more user friendly than PIL and is a must have for anyone who works with images.
5. SQL Alchemy. A database library. Many love it and many hate it. The choice is yours.
6. Beautiful Soup. I know it's slow but this xml and html parsing library is very useful for beginners.
7. Twisted. The most important tool for any network application developer. It has a very beautiful ape and is used by a lot of famous python developers.
8. Numbly. How can we leave this very important library? It provides some advance math functionalities to python.
9. Skippy. When we talk about numbly then we have to talk about spicy. It is a library of algorithms and mathematical tools for python and has caused many scientists to switch from ruby to python.

10. Matplotlib. A numerical plotting library. It is very useful for any data scientist or any data analyser.
11. Pygmy. Which developer does not like to play games and develop them? This library will help you achieve your goal of 2d game development.
12. Piglet. A 3d animation and game creation engine. This is the engine in which the famous python port of mine craft was made
13. Pit. A GUI toolkit for python. It is my second choice after python for developing GUIs for my python scripts.
14. Pit. Another python GUI library. It is the same library in which the famous Bit torrent client is created.
15. Scaly. A packet sniffer and analyser for python made in python.
16. Pywin32. A python library which provides some useful methods and classes for interacting with windows.
17. Notch. Natural Language Toolkit – I realize most people won't be using this one, but it's generic enough. It is a very useful library if you want to manipulate strings. But its capacity is beyond that. Do check it out.
18. Nose. A testing framework for python. It is used by millions of python developers. It is a must have if you do test driven development.
19. Simply. Simply can-do algebraic evaluation, differentiation, expansion, complex numbers, etc. It is contained in a pure Python distribution.

I Python. I just can't stress enough how useful this tool is. It is a python prompt on steroids. It has completion, history, shell capabilities, and a lot more. Make sure that you take a look at it.

## 4.7 Python class and objects

These are the building blocks of OOP. Class creates a new object. This object can be anything, whether an abstract data concept or a model of a physical object, e.g., a chair. Each class has individual characteristics unique to that class, including variables and methods. Classes are very powerful and currently "the big thing" in most programming languages. Hence, there are several chapters dedicated to OOP later in the book.

The class is the most basic component of object-oriented programming. Previously, you learned how to use functions to make your program do something. Now will move into the big, scary world of Object-Oriented Programming (OOP). To be honest, it took me several months to get a handle on objects.

However, when my C++ course started talking about objects, classes, and all the new features of OOP, my grades definitely suffered. Once you learn OOP, you'll realize that it's actually a pretty powerful tool. Plus, many Python libraries and APIs use classes, so you should at least be able to understand what the code is doing.

One thing to note about Python and OOP: it's not mandatory to use objects in your code in a way that works best; maybe you don't need to have a full-blown class with initialization code and methods to just return a calculation. With Python, you can get as technical as you want.

As you've already seen, Python can do just fine with functions. Unlike languages such as Java, you aren't tied down to a single way of doing things; you can mix functions and classes as necessary in the same program. This lets you build the code

Objects are an encapsulation of variables and functions into a single entity. Objects get their variables and functions from classes. Classes are essentially a template to create your objects.

Here's a brief list of Python OOP ideas:

- The class statement creates a class object and gives it a name. This creates a new namespace.
- Assignments within the class create class attributes. These attributes are accessed by qualifying the name using dot syntax: ClassName.Attribute.

- Class attributes export the state of an object and its associated behaviour. These attributes are shared by all instances of a class.
- Calling a class (just like a function) creates a new instance of the class.
- This is where the multiple copy's part comes in.
- Each instance gets ("inherits") the default class attributes and gets its own namespace. This prevents instance objects from overlapping and confusing the program.
- Using the term self identifies a particular instance, allowing for per-instance attributes. This allows items such as variables to be associated with a particular instance.

## Inheritance

- First off, classes allow you to modify a program without really making changes to it.
- To elaborate, by sub classing a class, you can change the behaviour of the program by simply adding new components to it rather than rewriting the existing components.
- As we've seen, an instance of a class inherits the attributes of that class.
- However, classes can also inherit attributes from other classes. Hence, a subclass inherits from a superclass allowing you to make a generic superclass that is specialized via subclasses.
- The subclasses can override the logic in a superclass, allowing you to change the behaviour of your classes without changing the superclass at all.

## Operator Overloads

Operator overloading simply means that objects that you create from classes can respond to actions (operations) that are already defined within Python, such as addition, slicing, printing, etc.

Even though these actions can be implemented via class methods, using overloading ties the behaviour closer to Python's object model and the object interfaces are more consistent to Python's built-in objects, hence overloading is easier to learn and use. User-made classes can override nearly all of Python's built-in operation methods

**Exceptions**

I've talked about exceptions before but now I will talk about them in depth. Essentially, exceptions are events that modify program's flow, either intentionally or due to errors.

They are special events that can occur due to an error, e.g., trying to open a file that doesn't exist, or when the program reaches a marker, such as the completion of a loop.

Exceptions, by definition, don't occur very often; hence, they are the "exception to the rule" and a special class has been created for them. Exceptions are everywhere in Python.

Virtually every module in the standard Python library uses them, and Python itself will raise them in a lot of different circumstances.

Here are just a few examples:

- Accessing a non−existent dictionary key will raise a Key Error exception.
- Searching a list for a non−existent value will raise a Value Error exception
- Calling a non−existent method will raise an Attribute Error exception.
- Referencing a non−existent variable will raise a Name Error exception.
- Mixing data types without coercion will raise a Type Error exception.

**User-Defined Exceptions:** I won't spend too much time talking about this, but Python does allow for a programmer to create his own exceptions.

You probably won't have to do this very often but it's nice to have the option when necessary.

However, before making your own exceptions, make sure there isn't one of the built-in exceptions that will work for you. They have been "tested by fire" over the years and not only work effectively, they have been optimized for performance and are bug-free.

Making your own exceptions involves object-oriented programming, which will be covered in the next chapter. To make a custom exception, the programmer determines which base exception to use as the class to inherit from, e.g., making an exception for negative numbers or one for imaginary numbers would probably fall under the Arithmetic Error exception class.

To make a custom exception, simply inherit the base exception and define what it will do.

## Python Modules

Python allows us to store our code in files (also called modules). This is very useful for more serious programming, where we do not want to retype a long function definition from the very beginning just to change one mistake. In doing this, we are essentially defining our own modules, just like the modules defined already in the Python library.

To support this, Python has a way to put definitions in a file and use them in a script or in an interactive instance of the interpreter. Such a file is called a module; definitions from a module can be imported into other modules or into the main module.

## Functions in Python

It is possible, and very useful, to define our own functions in Python. Generally speaking, if you need to do a calculation only once, then use the interpreter. But when you or others have need to perform a certain type of calculation many times, then define a function.

You use functions in programming to bundle a set of instructions that you want to use repeatedly or that, because of their complexity, are better self-contained in a sub-program and called when needed. That means that a function is a piece of code written to carry out a specified task.

To carry out that specific task, the function might or might not need multiple inputs. When the task is carved out, the function can or cannot return one or more values.

There are three types of functions in python: Help (), min (), print ().

## Python Namespace

Generally speaking, a namespace (sometimes also called a context) is a naming system for making names unique to avoid ambiguity. Everybody knows a name spacing system from daily life, i.e., the naming of people in first name and family name (surname). An example is a network: each network device (workstation, server, printer,) needs a unique name and address. Yet another example is the directory structure of file systems.

The same file name can be used in different directories, the files can be uniquely accessed via the pathnames.

Many programming languages use namespaces or contexts for identifiers. An identifier defined in a namespace is associated with that namespace.

This way, the same identifier can be independently defined in multiple namespaces. (Like the same file names in different directories) Programming languages, which support namespaces, may have different rules that determine to which namespace an identifier belongs.

Namespaces in Python are implemented as Python dictionaries, this means it is a mapping from names (keys) to objects (values). The user doesn't have to know this to write a Python program and when using namespaces.

Some namespaces in Python:

- Global names of a module
- Local names in a function or method invocation
- Built-in names: this namespace contains built-in functions (e.g., abs (), camp (), ...) and built-in exception names

## Garbage Collection

Garbage Collector exposes the underlying memory management mechanism of Python, the automatic garbage collector. The module includes functions for controlling how the collector operates and to examine the objects known to the system, either pending collection or stuck in reference cycles and unable to be freed.

## Python XML Parser

XML is a portable, open-source language that allows programmers to develop applications that can be read by other applications, regardless of operating system and/or developmental language.

What is XML? The Extensible Markup Language XML is a markup language much like HTML or SGML. This is recommended by the World Wide Web Consortium and available as an open standard.

XML is extremely useful for keeping track of small to medium amounts of data without requiring a SQL-based backbone. XML Parser Architectures and APIs the Python standard library provides a minimal but useful set of interfaces to work with XML.

The two most basic and broadly used APIs to XML data are the SAX and DOM interfaces.

Simple API for XML SAX: Here, you register call-backs for events of interest and then let the parser proceed through the document. This is useful when your documents are large or you have memory limitations, it parses the file as it reads it from disk and the entire file is never stored in memory.

Document Object Model DOM API: This is a World Wide Web Consortium recommendation wherein the entire file is read into memory and stored in a hierarchical tree − based form to represent all the features of an XML document.

SAX obviously cannot process information as fast as DOM can when working with large files. On the other hand, using DOM exclusively can really kill your resources, especially if used on a lot of small files.

SAX is read-only, while DOM allows changes to the XML file. Since these two different APIs literally complement each other, there is no reason why you cannot use them both for large projects.

## Python Web Frameworks

A web framework is a code library that makes a developer's life easier when building reliable, scalable and maintainable web applications.

Why are web frameworks useful?

Web frameworks encapsulate what developers have learned over the past twenty years while programming sites and applications for the web. Frameworks make it easier to reuse code for common HTTP operations and to structure projects so other developers with knowledge of the framework can quickly build and maintain the application.

## Common web framework functionality

Frameworks provide functionality in their code or through extensions to perform common operations required to run web applications. These common operations include:

1. URL routing
2. HTML, XML, JSON, and other output format tinplating
3. Database manipulation

4. Security against Cross-site request forgery (CSRF) and other attacks
5. Session storage and retrieval

Not all web frameworks include code for all of the above functionality. Frameworks fall on the spectrum from executing a single use case to providing every known web framework feature to every developer. Some frameworks take the "batteries-included" approach where everything possibly comes bundled with the framework while others have a minimal core package that is amenable to extensions provided by other packages.

## Comparing web frameworks

There is also a repository called compare-python-web-frameworks where the same web application is being coded with varying Python web frameworks, tinplating engines and object.

## Web framework resources

- When you are learning how to use one or more web frameworks it's helpful to have an idea of what the code under the covers is doing.
- Frameworks is a really well-done short video that explains how to choose between web frameworks. The author has some particular opinions about what should be in a framework. For the most part I agree although I've found sessions and database ORMs to be a helpful part of a framework when done well.
- What is a web framework? Is an in-depth explanation of what web frameworks being and their relation to web servers?
- Jingo vs. Flash vs. Pyramid: Choosing a Python web framework contains background information and code comparisons for similar web applications built in these three big Python frameworks.
- This fascinating blog post takes a look at the code complexity of several Python web frameworks by providing visualizations based on their code bases.
- Python’s web frameworks benchmarks are a test of the responsiveness of a framework with encoding an object to JSON and returning it as a response as well as retrieving data from the database and rendering it in a template. There were no conclusive results but the output is fun to read about nonetheless.

### Web frameworks learning checklist

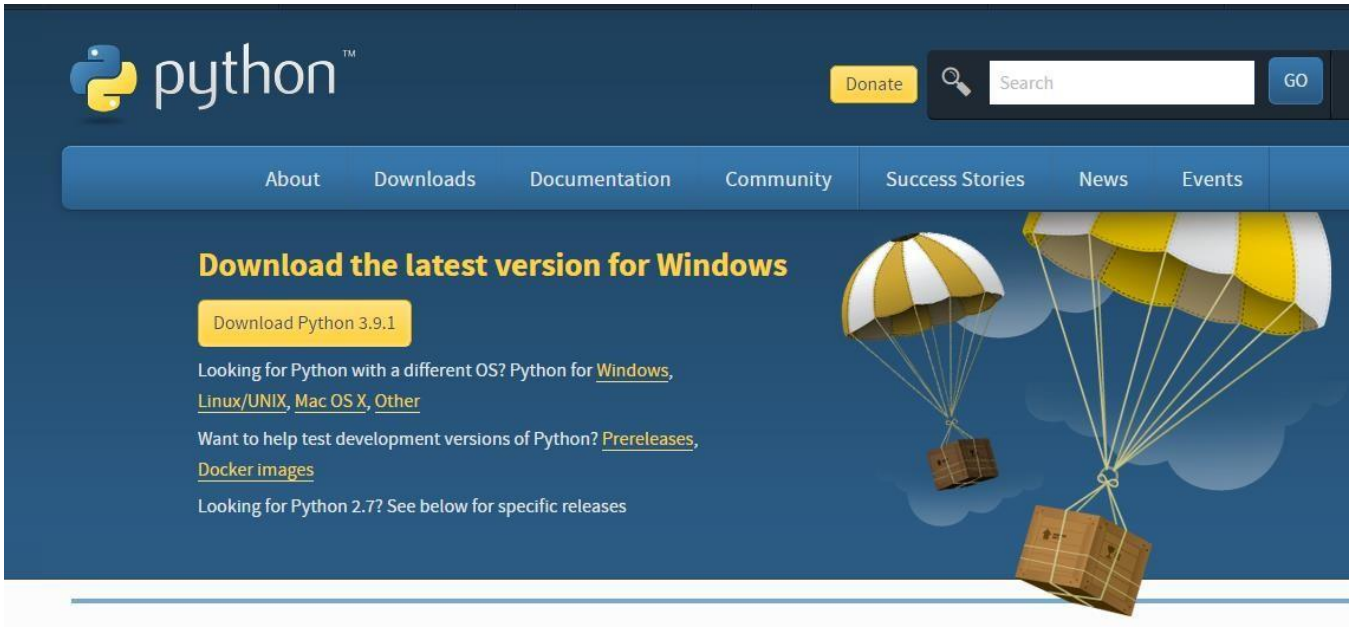
1. Choose a major Python web framework (Jingo or Flask are recommended) and stick with it. When you're just starting it's best to learn one framework first instead of bouncing around trying to understand every framework.
2. Work through a detailed tutorial found within the resource's links on the framework's page.
3. Study open-source examples built with your framework of choice so you can take parts of those projects and reuse the code in your application.
4. Build the first simple iteration of your web application then go to the deployment section to make it accessible on the web.

## 4.8 Implementation Setup

### Software Installation for Machine Learning Projects

Installing Python:

1. To download and install Python visit the official website of Python https://www.python.org/downloads/ and choose your version.




2. Once the download is complete, run the exe for install Python. Now click on Install Now.

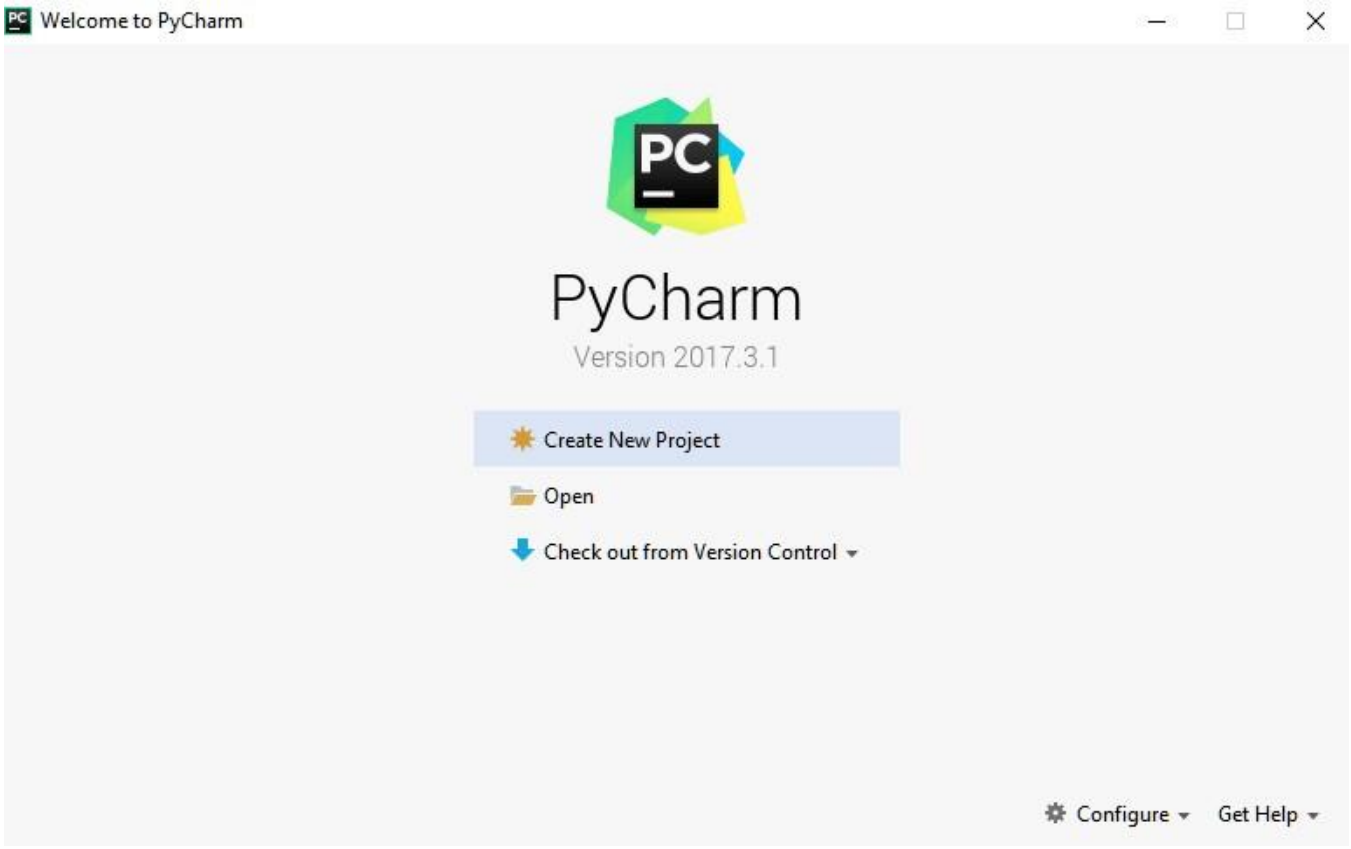
3. You can see Python installing at this point.
4. When it finishes, you can see a screen that says the Setup was successful. Now click on "Close".

Installing PyCharm:

1. To download PyCharm visit the website https://www.jetbrains.com/pycharm/download/ and Click the "DOWNLOAD" link under the Community Section.


2. Once the download is complete, run the exe for install PyCharm. The setup wizard should have started. Click “Next”.
3. On the next screen, Change the installation path if required. Click “Next”.
4. On the next screen, you can create a desktop shortcut if you want and click on “Next”.
5. Choose the start menu folder. Keep selected Jet Brains and click on “Install”.
6. Wait for the installation to finish.
7. Once installation finished, you should receive a message screen that PyCharm is installed. If you want to go ahead and run it, click the “Run PyCharm Community Edition” box first and click “Finish”.

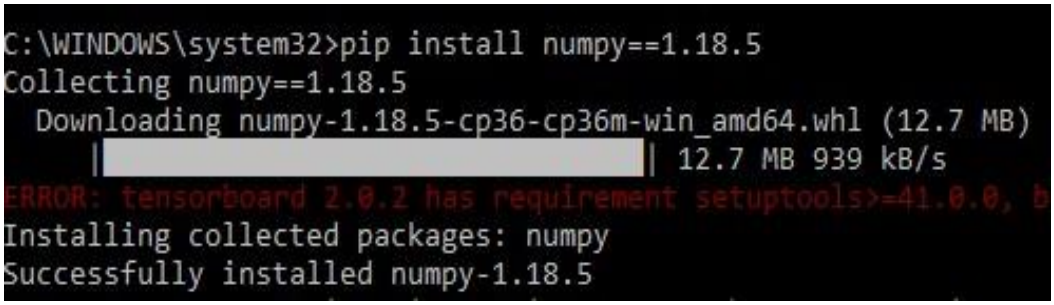
8. After you click on "Finish," the Following screen will appear.




9. You need to install some packages to execute your project in a proper way.

10. Open the command prompt/ anaconda prompt or terminal as administrator.

11. The prompt will get open, with specified path, type “pip install package name” which you want to install (like NumPy, pandas, seaborn, scikit-learn, matplotlib.pyplot) Ex: pip install NumPy



# 5. TESTING

## Types of Tests

## 5.1 System Testing

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

## 5.2 Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

## 5.3 Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

## 5.4 Functional Testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is cantered on the following items:

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

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

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

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

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

## 5.5 System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

## 5.6 White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

## 5.7 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

**Unit Testing:**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

**Features to be tested**

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

**Integration Testing**

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g., components in a software system or – one step up – software applications at the company level interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

**Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

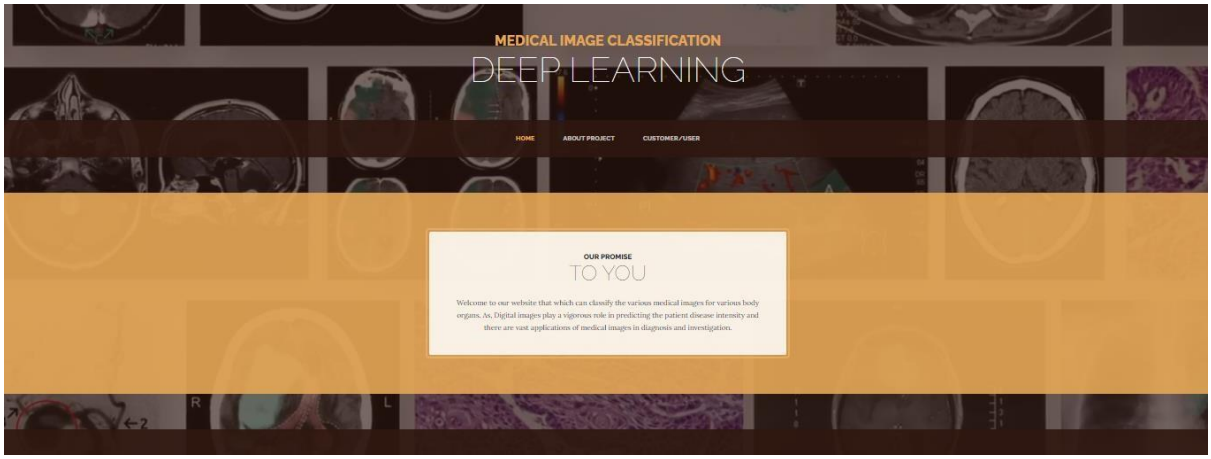
# 6. FUTURE ENHANCEMENT

In future, we aim to explore large-scale image datasets for medical image classification and detection problems. And we can go for another types of pre-trained algorithms that can perform well and gives high classification. By which, we can classify different types of organs and detect the problems easily

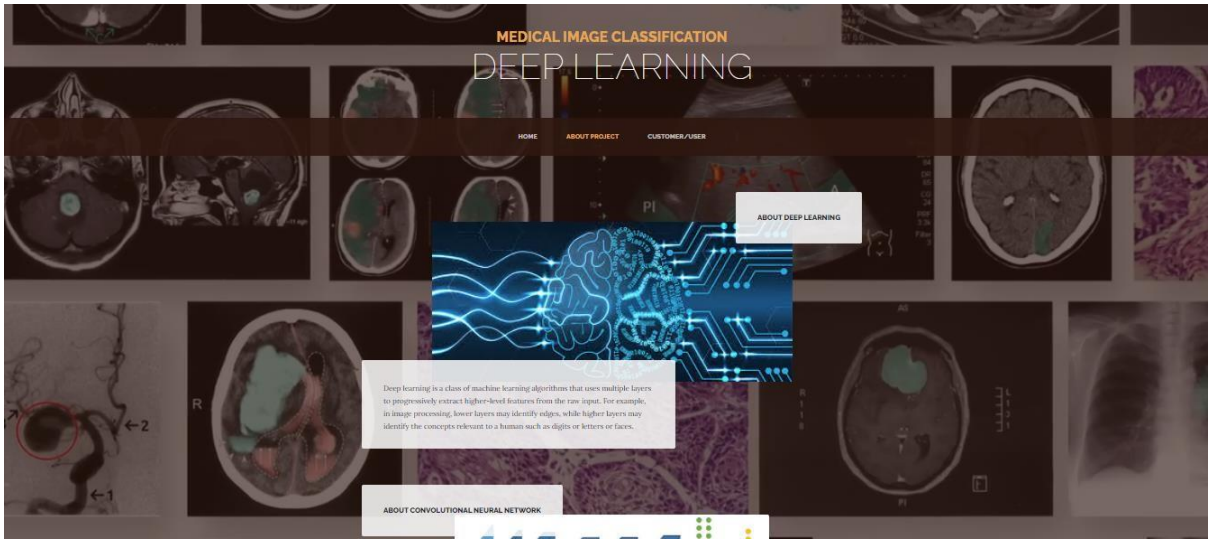
# 7. CONCLUSION

For classification purpose, deep learning-based framework for medical image classification by training the images is proposed. In this regard, diagnosis is one of the main requirements of the existing era and investigated or examine to specific diseases. Hence, we have proposed a novel deep convolution network-based approach that is assist of doctors and physicians in making reasonable decisions. The results obtained from the proposed method outperformed state-of the-art methods that is reported for the same dataset.

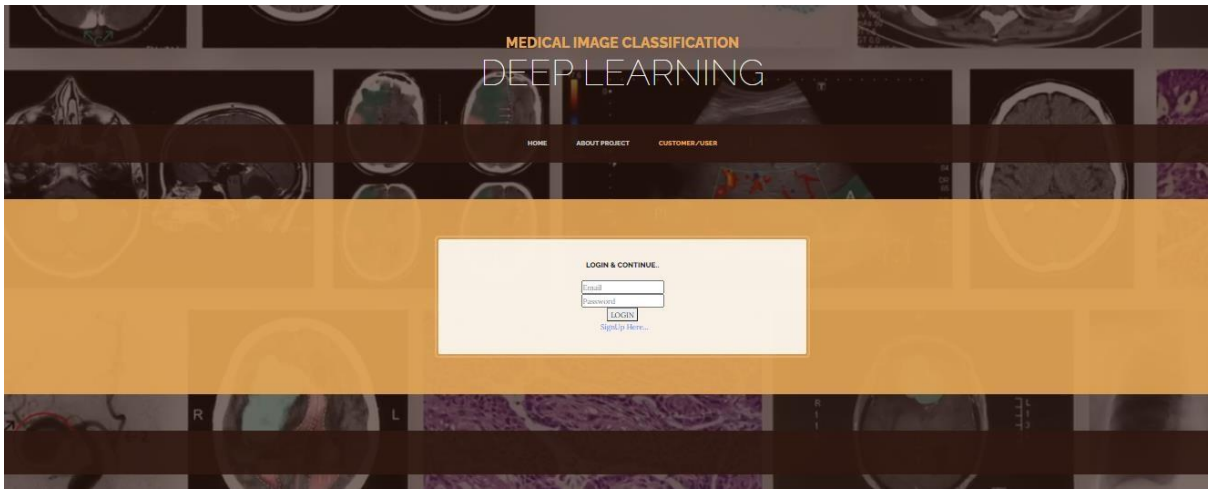
# Appendix: A Snap Shots:



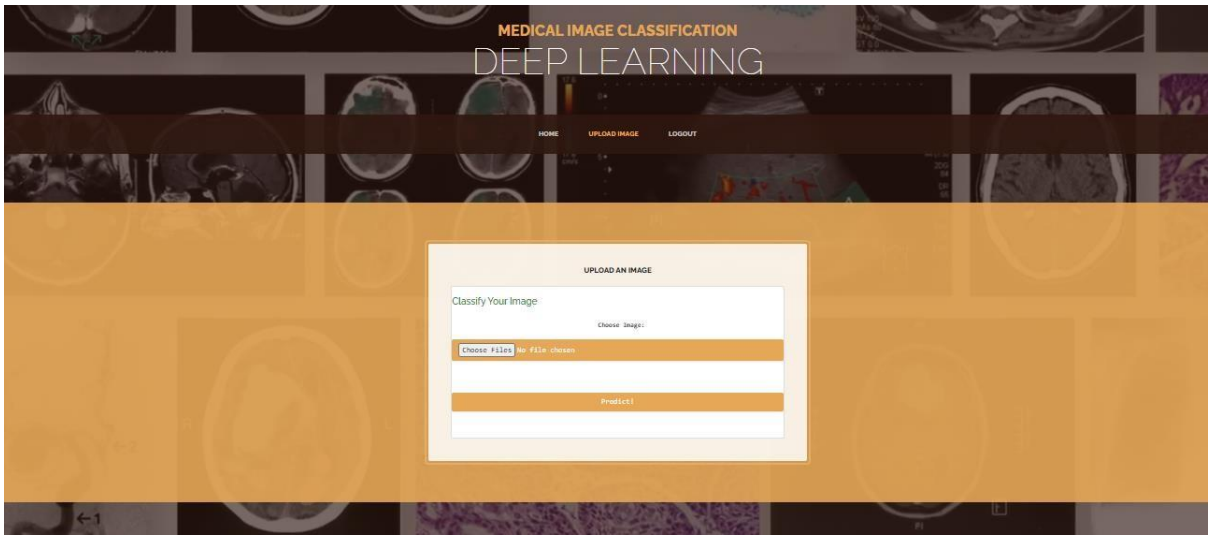

**1. Home Page**



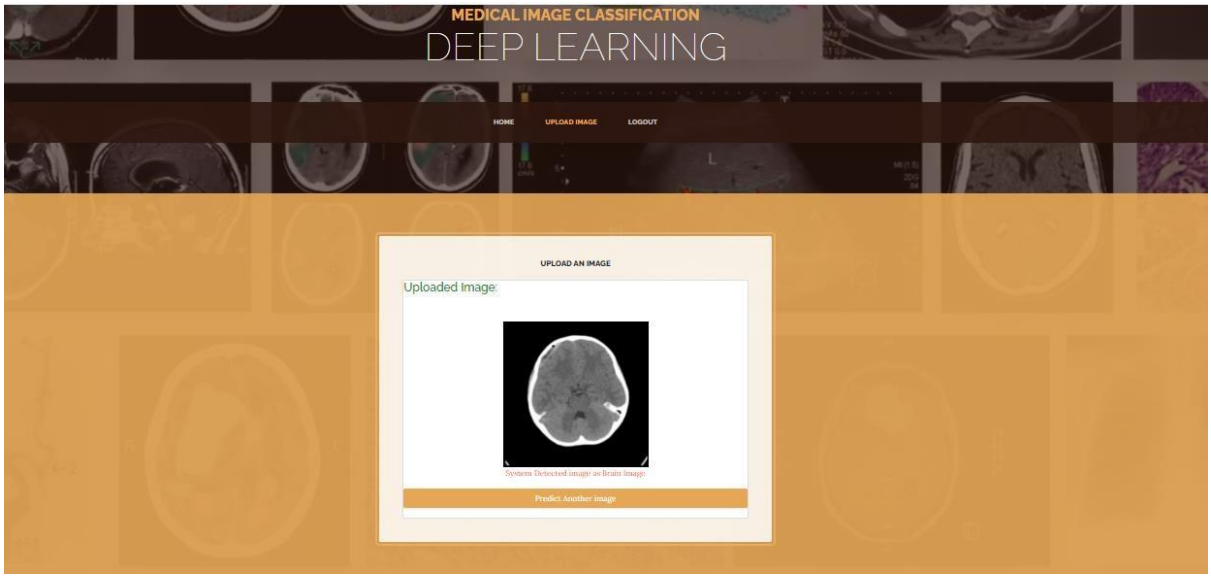

**2. About Project**



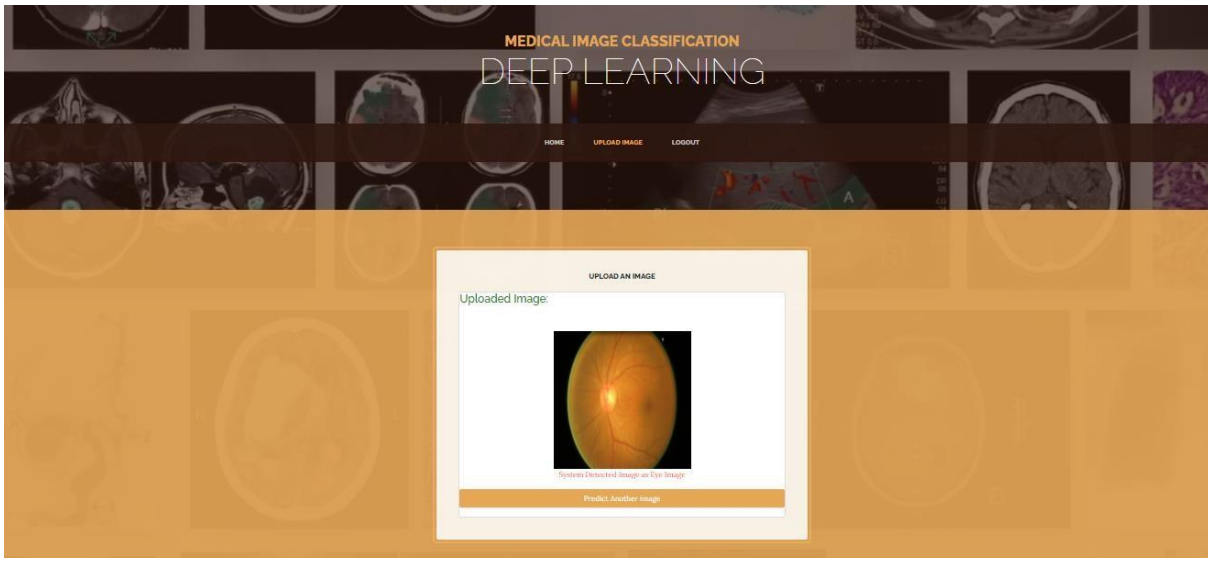

**3.Login Page**




**4.Uploading Image**




**5.Output image classified as Brain**




**6.Output image classified as Eye**

## Appendix: B Reference

[1] Q. Zhu, B. Du, and P. Yan, ‘‘Boundary-weighted domain adaptive neural network for prostate MR image segmentation,’’ IEEE Trans. Med. Imag., vol. 39, no. 3, pp. 753–763, Mar. 2020.

[2] Q.Zhu, B.Du, P.Yan, H.Lu, and L.Zhang, ‘‘Shape prior constrained PSO model for bladder wall MRI segmentation,’’ Neuro computing, vol. 294, pp. 19–28, Jun. 2018.

[3] Q. Zhu, B. Du, B. Turkbey, P. Choyke, and P. Yan, ‘‘Exploiting inter slice correlation for MRI prostate image segmentation, from recursive neural networks aspect,’’ Complexity, vol. 2018, pp. 1–10, Feb. 2018.

[4] K.Kranthi Kumar and T.V.Gopal, ‘‘A novel approach to self-order feature reweighting in CBIR to reduce semantic gap using relevance feedback,’’ in Proc. Int. Conf. Circuits, Power Comput. Technol. (ICCPCT), Mar. 2014, pp. 1437–1442.

[5] R.Ashraf, K.B.Bajwa, and T.Mahmood, ‘‘Content-based image retrieval by exploring bandletized regions through support vector machines. ’J.Inf. Sci. Eng., vol. 32, no. 2, pp. 245–269, 2016.

[6] J. Wan, D. Wang, S. C. H. Hoi, P. Wu, J. Zhu, Y. Zhang, and J. Li, ‘‘Deep learning for content-based image retrieval: A comprehensive study,’’ in Proc. ACM Int. Conf. Multimedia (MM), 2014, pp. 157–166.

[7] F. Shaukat, G. Raja, R. Ashraf, S. Khalid, M. Ahmad, and A. Ali, ‘‘Artificial neural network based classification of lung nodules in ct images using intensity, shape and texture features,’’ J. Ambient Intell. Humanized Comput. vol. 10, no. 10, pp. 4135–4149, 2019.

[8] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, ‘‘Large-scale video classification with convolutional neural networks, ’in Proc. IEEE Conf. Comput. Vis .Pattern Recognit. Jun.2014, pp. 1725–1732.

[9] G.Wu, W.Lu, G.Gao, C.Zhao, and J.Liu, ‘‘Regional deep learning model for visual tracking,’’ Neuro computing, vol. 175, pp. 310–323, Jan. 2016.

[10] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, ‘‘Deep neural networks for acoustic modeling in speech recognition,’’ IEEE Signal Process. Mag., vol. 29, no. 6, pp. 82–97, Nov. 2012.

[11] S. Zhou, Q. Chen, and X. Wang, ‘‘Active deep learning method for semi-supervised sentiment classification,’’ Neuro computing, vol. 120, pp. 536–546, Nov. 2013

[12] R.Girshick, J.Donahue, T.Darrell, and J.Malik, ‘‘Rich feature hierarchies for accurate object detection and semantic segmentation,’’ in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2014, pp. 580–587.

[13] H.-Y.Shi, ‘‘Pancreatic carcinosarcoma: First literature report on computed tomography imaging,’’ World J. Gastroenterol., vol. 21, no. 4, p. 1357, 2015.

[14] M. R. Zare, W. C. Seng, and A. Mueen, ‘‘Automatic classification of medical X-ray images using a bag of visual words,’’ IET Comput. Vis., vol. 7, no. 2, pp. 105–114, Apr. 2013.

[15] S. Khan, S.-P. Yong, and J. D. Deng, ‘‘Ensemble classification with modified SIFT descriptor for medical image modality,’’inProc.Int.Conf.Image Vis. Comput. New Zealand (IVCNZ), Nov. 2015, pp. 1–6.

[16] D. C. Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, ‘‘Mitosisdetectioninbreastcancerhistologyimageswithdeepneuralnetworks,’’ in Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent. Berlin, Germany: Springer, 2013, pp. 411–418.

[17] P. Kanerva, ‘‘Hyper dimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors,’’ Cognit. Comput. vol. 1, no. 2, pp. 139–159, Jun. 2009.

[18] O. Yilmaz, ‘‘Machine learning using cellular automata-based feature expansion and reservoir computing,’’ J. Cellular Automata, vol. 10, pp. 435–472, Sep. 2015.

[19] D. Cheng, G. Meng, G. Cheng, and C. Pan, ‘‘SeNet: Structured edge network for Sea–Land segmentation,’’ IEEE Geosci. Remote Sens. Lett. vol. 14, no. 2, pp. 247–251, Feb. 2017.

[20] P.Chen, Y.Song, D.Yuan, and Z.Liu, ‘‘Feature fusion adversarial learning network for liver lesion classification,’’ in Proc. ACM Multimedia Asia ZZZ, Dec. 2019, pp. 1–7.

[21] Y.Bengio, P.Lamblin, D.Popovici, and H.Larochelle, ‘‘Greedylayer-wise training of deep networks,’’ in Proc. Adv. Neural Inf. Process. Syst., 2007, pp. 153–160.

[22] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. W. M. van der Laak, B. van Ginneken, and C. I. Sánchez, ‘‘A survey on deep learning in medical image analysis,’’ Med. Image Anal., vol. 42, pp. 60–88, Dec. 2017

Certificate of Publication

# INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS | ISSN: 2320 - 2882

*An International Open Access, Peer-reviewed, Refereed Journal*

The Board of

International Journal of Creative Research Thoughts

Is hereby awarding this certificate to

**Mr. Narala Srinath Reddy**

In recognition of the publication of the paper entitled

**Medical Image Classification Using CNN**

Published In IJCRT ( www.ijcrt.org ) & 7.97 Impact Factor by Google Scholar

Volume 11 Issue 7 July 2023 , Date of Publication: 27-July-2023

UGC Approved Journal No: 49023 (18)

ISSN 2320-2882 IJCRT

EDITOR IN CHIEF

PAPER ID : IJCRT2307761

**Registration ID : 241756**

**Scholarly open access journals, Peer-reviewed, and Refereed Journals, Impact factor 7.97 (Calculate by google scholar and Semantic Scholar | AI-Powered Research Tool) , Multidisciplinary, Monthly Journal**

**INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS | IJCRT**

*An International Scholarly, Open Access, Multi-disciplinary, Indexed Journal*

**Website: www.ijcrt.org | Email id: editor@ijcrt.org | ESTD: 2013**

IJCRT | ISSN: 2320-2882 | IJCRT.ORG

Certificate of Publication

# INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS | ISSN: 2320 - 2882

*An International Open Access, Peer-reviewed, Refereed Journal*

The Board of

International Journal of Creative Research Thoughts

Is hereby awarding this certificate to

**Mrs. B. Swetha**

In recognition of the publication of the paper entitled

**Medical Image Classification Using CNN**

Published In IJCRT ( www.ijcrt.org ) & 7.97 Impact Factor by Google Scholar

Volume 11 Issue 7 July 2023 , Date of Publication: 27-July-2023

UGC Approved Journal No: 49023 (18)

PAPER ID : IJCRT2307761

**Registration ID : 241756**

ISSN 2320-2882 IJCRT

**EDITOR IN CHIEF**

**Scholarly open access journals, Peer-reviewed, and Refereed Journals, Impact factor 7.97 (Calculate by google scholar and Semantic Scholar | AI-Powered Research Tool) , Multidisciplinary, Monthly Journal**

**INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS | IJCRT**

*An International Scholarly, Open Access, Multi-disciplinary, Indexed Journal*

**Website: www.ijcrt.org | Email id: editor@ijcrt.org | ESTD: 2013**

IJCRT | ISSN: 2320-2882 | IJCRT.ORG

IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

# Medical Image Classification Using CNN

Mr. Narala Srinath Reddy (M.C.A). Rajeev Gandhi Memorial college Of Engineering and Technology, Nandyal

*Mrs. B. Swetha M. Tech, (Ph.D). Rajeev Gandhi Memorial college Of Engineering and Technology, Nandyal

## Abstract

Deep learning is one of the most unexpected machine learning techniques which is being used in many applications like image classification, image analysis, clinical archives and object recognition. With an extensive utilization of digital images as information in the hospitals, the archives of medical images are growing exponentially. Digital images play a vigorous role in predicting the patient disease intensity and there are vast applications of medical images in diagnosis and investigation. Due to recent developments in imaging technology, classifying medical images in an automatic way is an open research problem for researchers of computer vision. For classifying the medical images according to their relevant classes, a most suitable classifier is most important. Where we are proposing our model where the algorithm is trained for classifying medical images by deep learning technique. A pre-trained deep convolution neural network (GoogleNet) is used that which can classifies the various medical images for various body organs. This method of image classification is beneficial to predict the appropriate class or category of unknown images. The results of the experiment exhibit that our method is best suited to classify various medical images.

**Keywords:** Medical image classification, pre-trained DCNN, convolution neural network, deep learning

## 1. INTRODUCTION

### 1.1 Introduction

Image classification is the primary domain, in which deep neural networks play the most important role of medical image analysis. The image classification accepts the given input images and produces output classification for identifying whether the disease is present or not. In our model we mainly classify the different types of organs and predict the accuracy.

One of the most imperative problems faced in the domain area of image recognition is the classification of medical images. The major intention of medical image classification is to classify medical images into several elements to assist medical practitioners or physicists in diagnosing disease. Hence, medical image classification is split into two steps. The first and foremost step of medical image classification is to extract the essential features from the acquired input image. The second step in medical image classification is utilizing the features to construct models that classify the image data set. In the recent past, medical practitioners customarily utilized their specialized experience to extract features so that classification of medical images could be performed into several classes. However, this manual medical image classification was found to be highly cumbersome and time consuming.

Medical image classification involves the process of segregating medical-related information into a useful form. Classification of medical images is based on placing image pixels with similar values into groups. With the placement of similar values into groups, common pixels are identified and are denoted by these

pixels. Hence, a correctly classified image usually denotes the areas on the ground that share specific features as specified in the classification scheme.

Image classification is where a computer can analyse an image and identify the 'class' the image falls under. (Or a probability of the image being part of a 'class'.) A class is essentially a label, for instance, 'car', 'animal', 'building' and so on. For example, you input an image of a sheep. Image classification is the process of the computer analysing the image and telling you it's a sheep. (Or the probability that it's a sheep.) For us, classifying images is no big deal. But it's a perfect example of Moravec's paradox when it comes to machines. (That is, the things we find easy are difficult for AI.)

Early image classification relied on raw pixel data. This meant that computers would break down images into individual pixels. The problem is that two pictures of the same thing can look very different. They can have different backgrounds, angles, poses, and etcetera. This made it quite the challenge for computers to correctly 'see' and categories images.

## 2. Literature Survey

**[1] Q. Zhu, B. Du, and P. Yan:** Accurate segmentation of the prostate from magnetic resonance (MR) images provides useful information for prostate cancer diagnosis and treatment. However, automated prostate segmentation from 3D MR images still faces several challenges. The complex background texture and large variation in size, shape and intensity distribution of the prostate itself make segmentation even further complicated. Since large-scale dataset is one of the critical components for the success of deep learning, lack of sufficient training data makes it difficult to fully train complex CNNs. To tackle the above challenges, in this paper boundary-weighted domain adaptive neural network (BOWDA-Net) is proposed. To make the network more sensitive to the boundaries during segmentation, a boundary-weighted segmentation loss (BWL) is proposed.

**Summary:** In this paper boundary-weighted domain adaptive neural network (BOWDA-Net) is proposed. To make the network more sensitive to the boundaries during segmentation, a boundary-weighted segmentation loss (BWL) is proposed. . Furthermore, an advanced boundary-weighted transfer leaning approach is introduced to address the problem of small medical imaging datasets. We evaluate our proposed model on the publicly available MICCAI 2012 Prostate MR Image Segmentation (PROMISE12) challenge dataset.

**[2] Q.Zhu, B.Du, P.Yan, H.Lu, and L.Zhang:** Bladder wall segmentation from Magnetic Resonance (MR) images plays an important role in diagnosis. Since the thickness of the bladder wall is a key indication of bladder cancer. There are several methods that have been used for bladder wall segmentation, such as level sets and Active Shape Model (ASM). However, the weak boundaries, the artifacts inside bladder lumen and the complex background outside the bladder wall make the bladder wall segmentation very challenging. To overcome these difficulties and obtain accurate bladder walls, in this paper, a shape prior constrained particle swarm optimization (SPC-PSO) model is proposed to segment the inner and outer boundaries of the bladder wall. The bladder walls are divided into two categories: strong boundaries and weak boundaries by the proposed model.

## 3. OVERVIEW OF THESYSTEM

### 3.1 Existing System

This model emphasizes an existing method that which is designed using the machine learning architecture which is used to classify the various medical images. With an extensive utilization of digital images as information in the hospitals, the archives of medical images are growing exponentially. Digital images play a vigorous role in predicting the patient disease intensity and there are vast applications of medical images in diagnosis and investigation. To make this in the easier way Support Vector Machine (SVM) is used that which can classifies the various medical images for various body organs.

#### 3.1.1 Disadvantages of Existing System

- Low efficiency.
- Time consuming.
- High complexities.
- No accurate classification

### 3.2 Proposed System

The proposed model emphasizes a deep network architecture which is used to classify the various medical images. With an extensive utilization of digital images as information in the hospitals, the archives of medical images are growing exponentially. Digital images play a vigorous role in predicting the patient disease intensity and there are vast applications of medical images in diagnosis and investigation. Hence, we are proposing our model where the algorithm is trained for classifying medical images by deep learning technique. A pre-trained deep convolution neural network (GoogleNet) is used that which can classifies the various medical images for various body organs.

### 3.3 Methodology

**User**

**1. System:**

**1.1 Create Dataset:**

The dataset containing images of the desired objects to be recognize is split into training and testing dataset with the test size of 20-30%.

**1.2 Pre-processing:**

Resizing and reshaping the images into appropriate format to train our model. **1.3Training:**

Use the pre-processed training dataset is used to train our model using CNN algorithm.

**2. User:**

**2.1 Register**

The user needs to register and the data stored in MySQL database**.**

**2.2 Login**

A registered user can login using the valid credentials to the website to use a application.

**2.1 About-Project**

In this application, we have successfully created an application which takes to classify the images.

**2.2 Upload Image**

The user has to upload an image which needs to be classify the images.

**2.3 Prediction**

The results of our model is displayed as either Rice Blast, Leaf Blight, Healthy & Brown Spot.

**2.6 Logout**

Once the prediction is over, the user can logout of the application.

## ALGORITHM

**1. Convolutional Neural Network**

**Step1: convolutional operation**

The first building block in our plan of attack is convolution operation. In this step, we will touch on feature detectors, which basically serve as the neural network's filters. We will also discuss feature maps, learning the parameters of such maps, how patterns are detected, the layers of detection, and how the findings are mapped out.

**Step (1b): ReLU Layer**

The second part of this step will involve the Rectified Linear Unit or Relook. We will cover Relook layers and explore how linearity functions in the context of Convolutional Neural Networks.

Not necessary for understanding CNNs, but there's no harm in a quick lesson to improve your skills.

## 4 Architecture

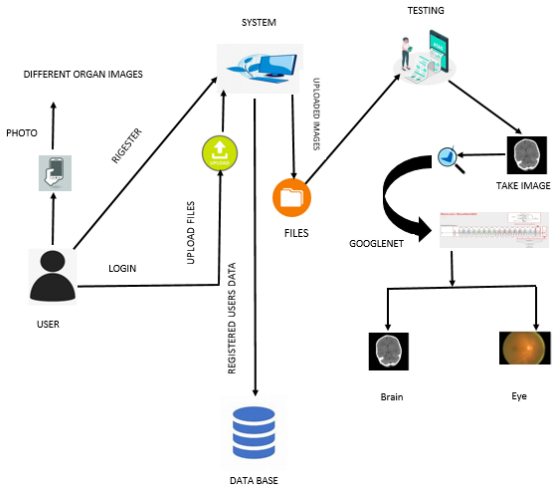


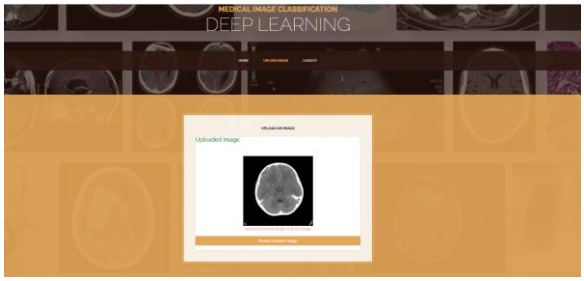

Fig 1: Frame work of proposed method

## 5 RESULTS SCREEN SHOTS

**Home Page:**


**Upload Dataset:**


**Choose options:**




**Predict Result:**


## 6. CONCLUSION

✓ For classification purpose, deep learning-based framework for medical image classification by training the images is proposed. In this regard, diagnosis is one of the main requirements of the existing era and investigated or examine to specific diseases. Hence, we have proposed a novel deep convolution network-based approach that is assist of doctors and physicians in making reasonable decisions. The results obtained from the proposed method outperformed state-of the-art methods that is reported for the same dataset.

## FUTURE WORK

In future, we aim to explore large-scale image datasets for medical image classification and detection problems. And we can go for another types of pre-trained tea that can perform well and gives high classification. By which, we can classify different types of organs and detect the problems easily.

**7. References**

[1] Q. Zhu, B. Du, and P. Yan, ‘‘Boundary-weighted domain adaptive neural network for prostate MR image segmentation,’’ IEEE Trans. Med. Imag., vol. 39, no. 3, pp. 753–763, Mar. 2020.

[2] Q.Zhu, B.Du, P.Yan, H.Lu, and L.Zhang, ‘‘Shape prior constrained PSO model for bladder wall MRI segmentation,’’ Neuro computing, vol. 294, pp. 19–28, Jun. 2018.

[3] Q. Zhu, B. Du, B. Turkbey, P. Choyke, and P. Yan, ‘‘Exploiting inter slice correlation for MRI prostate image segmentation, from recursive neural networks aspect,’’ Complexity, vol. 2018, pp. 1–10, Feb. 2018.

[4] K.Kranthi Kumar and T.V.Gopal, ‘‘A novel approach to self-order feature reweighting in CBIR to reduce semantic gap using relevance feedback,’’ in Proc. Int. Conf. Circuits, Power Comput. Technol. (ICCPCT), Mar. 2014, pp. 1437–1442.

[5] R.Ashraf, K.B.Bajwa, and T.Mahmood, ‘‘Content-based image retrieval by exploring bandletized regions through support vector machines. ’J.Inf. Sci. Eng., vol. 32, no. 2, pp. 245–269, 2016.

[6] J. Wan, D. Wang, S. C. H. Hoi, P. Wu, J. Zhu, Y. Zhang, and J. Li, ‘‘Deep learning for content-based image retrieval: A comprehensive study,’’ in Proc. ACM Int. Conf. Multimedia (MM), 2014, pp. 157–166.

[7] F. Shaukat, G. Raja, R. Ashraf, S. Khalid, M. Ahmad, and A. Ali, ‘‘Artificial neural network based classification of lung nodules in ct images using intensity, shape and texture features,’’ J. Ambient Intell. Humanized Comput. vol. 10, no. 10, pp. 4135–4149, 2019.

[8] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, ‘‘Large-scale video classification with convolutional neural networks, ’in Proc. IEEE Conf. Comput. Vis .Pattern Recognit. Jun.2014, pp. 1725–1732.

[9] G.Wu, W.Lu, G.Gao, C.Zhao, and J.Liu, ‘‘Regional deep learning model for visual tracking,’’ Neuro computing, vol. 175, pp. 310–323, Jan. 2016.

[10] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, ‘‘Deep neural networks for acoustic modeling in speech recognition,’’ IEEE Signal Process. Mag., vol. 29, no. 6, pp. 82–97, Nov. 2012.