

*A Project report  
on*

# **FACE ANTI SPOOFING USING DEEP LEARNING**

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

## **BACHELOR OF TECHNOLOGY**

*in*

## **Computer Science & Engineering**

*By*

**P. ANKITHA** (184G1A0502)

**M. KOMALA** (184G1A0531)

**P. NAVEEN KUMAR** (184G1A0553)

**S. JASWANTH KUMAR** (184G1A0524)



**Department of Computer Science & Engineering**

**SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY**

**(Affiliated to JNTUA & Approved by AICTE)**

**(Accredited by NAAC with 'A' Grade & Accredited by NBA(EEE, ECE & CSE))**

**Rotarypuram Village, B K Samudram Mandal, Ananthapuramu-515701.**

**2021-2022**

# SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

(Affiliated to JNTUA & Approved by AICTE)  
(Accredited by NAAC with 'A' Grade & Accredited by NBA (EEE, ECE & CSE)  
Rotarypuram Village, B K Samudram Mandal, Ananthapuramu-515701.

## DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



## Certificate

This is to certify that the Project report entitled **Face Anti Spoofing Using Deep Learning** is the bonafide work carried out by **P. Ankitha** bearing Roll Number **184G1A0502**, **M. Komala** bearing Roll Number **184G1A0531**, **P. Naveen Kumar** bearing Roll Number **184G1A0553** and **S. Jashwanth Kumar** bearing Roll Number **184G1A0524** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering** during the academic year 2021 - 2022.

### Signature of the Guide

Dr. G. K. Venkata Narasimha Reddy M.Tech., Ph.D.  
Professor

### Head of the Department

Mr. P. Veera Prakash M.Tech., (Ph.D.)  
Assistant Professor & HOD

Date:

Place: Rotarypuram

**EXTERNAL EXAMINER**

## ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that I have now the opportunity to express my gratitude for all of them.

It is with immense pleasure that we would like to express my indebted gratitude to my Guide **Dr.G.K.Venkata Narasimha Reddy**, M.Tech., Ph.D., **Professor, Computer Science & Engineering**, who has guided me a lot and encouraged me in every step of the project work. I thank him for the stimulating guidance, constant encouragement and constructive criticism which have made possible to bring out this technical seminar.

We are very much thankful to **Mr. P. Veera Prakash**, M.Tech., (Ph.D.), **Assistant Professor & Head of the Department, Computer Science & Engineering**, for his kind support and for providing necessary facilities to carry out the work.

We wish to convey my special thanks to **Dr. G. Bala Krishna**, Ph.D., **Principal of Srinivasa Ramanujan Institute of Technology** for giving the required information in doing my project work. Not to forget, we thank all other faculty and non-teaching staff, and my friends who had directly or indirectly helped and supported me in completing my project in time.

We also express our sincere thanks to the Management for providing excellent facilities.

Finally, we wish to convey my gratitude to my family who fostered all the requirements and facilities that we need.

**184G1A0502**

**184G1A0531**

**184G1A0553**

**184G1A0524**

## **Declaration**

We, Ms P.Ankitha bearing reg no: 184G1A0502 , Ms M.Komala bearing reg no: 184G1A0531, Mr P. Naveen Kumar bearing reg no: 184G1A0553 , Mr S. Jaswanth Kumar bearing reg no: 184G1A0524 students of SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY , Rotarypuram , hereby declare that the dissertation entitled “FACE ANTI SPOOFING USING DEEP LEARNING” embodies the report of our project work carried out by us during IV year Bachelor of Technology under the guidance of Dr.G.K.Venkata Narasimha Reddy M.Tech.,Ph.D. Department of CSE, SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY, and this work has been submitted for the partial fulfillment of the requirements for the award of the Bachelor of Technology degree.

The results embodied in this project have not been submitted to any other University of Institute for the award of any Degree or Diploma.

P. ANKITHA

Reg no: 184G1A0502

M. KOMALA

Reg no: 184G1A0531

P. NAVEEN KUMAR

Reg no: 184G1A0553

S. JASWANTH KUMAR

Reg no: 184G1A0524

<b><u>CONTENTS</u></b>		<b>Page No.</b>
<b>List of Figures</b>		<b>VI</b>
<b>Abbreviations</b>		<b>VII</b>
<b>Abstract</b>		<b>VIII</b>
<b>Chapter 1</b>	<b>Introduction</b>	<b>1</b>
	1.1 Existing System	2
	1.2 Proposed System	3
<b>Chapter 2</b>	<b>Literature Survey</b>	<b>4</b>
<b>Chapter 3</b>	<b>Analysis</b>	<b>10</b>
	3.1 Feasibility study	10
	3.2 System Requirement Specification	11
	3.2.1 Functional and Non- Functional Requirement	11
	3.2.2 Software and Hardware Requirement	12
	3.3 Software Installation	12
<b>Chapter 4</b>	<b>Software Development Life Cycle</b>	<b>17</b>
<b>Chapter 5</b>	<b>Design</b>	<b>19</b>
	5.1 System Design	19
	5.2 UML Introduction	21
	5.2.1 Usage of UML in project	21
	5.3 UML Diagrams	21
	5.4 Architecture	28
	5.5 Block Diagram	29
<b>Chapter 6</b>	<b>Implementation</b>	<b>30</b>
	6.1 Data Analysis	30
	6.2 Data Preprocessing	30
	6.2.1 Data Cleaning	31
	6.2.2 Data Integration	32
	6.2.3 Data Transformation	33
	6.2.4 Data Reduction	34
	6.3 Methodology And Algorithms	34
<b>Chapter 7</b>	<b>Result and Performance Analysis</b>	<b>43</b>
<b>Chapter 8</b>	<b>Testing</b>	<b>47</b>
<b>Conclusion</b>		<b>50</b>
<b>Reference</b>		<b>51</b>

## **LIST OF FIGURES**

<b>Fig.No</b>	<b>Description</b>	<b>Page No</b>
Fig.4.1	Waterfall Model	17
Fig.5.1	Use Case Diagram	23
Fig.5.2	Class Diagram	23
Fig.5.3	Sequence Diagram	24
Fig.5.4	Collaboration Diagram	25
Fig.5.5	Deployment Diagram	25
Fig.5.6	Activity Diagram	26
Fig.5.7	Component Diagram	26
Fig.5.8	ER Diagram	27
Fig.5.9	Data Flow Diagram	28
Fig.6.1	Data Preprocessing	31
Fig.6.2	Data Integration	32
Fig.6.3	CNN Model	36
Fig.7.1	Accuracy Graph	46

## **LIST OF ABBREVIATIONS**

PA	Presentation Attack
SVM	Support Vector Machine
RPPG	Remote Photo Plethysmo Graphy
FAS	Face Anti-Spoofing
LBP	Local Binary Pattern
CNN	Convolution Neural Network
UML	Unified Modeling Language

## **ABSTRACT**

Face recognition has been widely researched and achieved great success in a variety of applications because the human face saves the most extravagant data for perceiving people. Face spoofing attacks continue to pose a threat to modern face recognition systems. Recent researchers had proposed their research in this area but many existing methods for face anti spoofing have been degraded by illuminations.

We propose a novel framework based on the Convolutional Neural Network for the face anti-spoofing problem, which is inspired by the philosophy used by humans to determine whether a presented face example is genuine or not, namely, to look at the example globally first and then carefully observe the local regions to gain more discriminative information (CNN). Specifically, we use deep learning to mimic the behavior of discovering face-spoofing-related information from image sub-patches.

**Keywords**— *face recognition, face anti spoofing, convolution neural network, deep learning.*



# CHAPTER 1

## INTRODUCTION

Biometrics utilize physiological, such as fingerprint, face, and iris, or behavioural characteristics, such as typing rhythm and gait, to uniquely identify or authenticate an individual. As biometric systems are widely used in real-world applications including mobile phone authentication and access control, biometricspoof, or Presentation Attack (PA) are becoming a larger threat, where a spoofed biometric sample is presented to the biometric system and attempted to be authenticated. Since face is the most accessible biometric modality, there have been many different types of PAs for faces including print attack, replay attack, 3D masks, etc. As a result, conventional face recognition systems can be very vulnerable to such PAs. In order to develop a face recognition system that is invulnerable to various types of PAs, there is an increasing demand on designing a robust face anti-spoofing (or PA detection) system to classify a face sample as live or spoof before recognizing its identity.

Previous approaches to tackle face anti-spoofing can be categorized in three groups. The first is the texture based methods, which discover discriminative texture characteristics unique to various attack mediums. Due to a lack of an explicit correlation between pixel intensities and different types of attacks, extracting robust texture features is challenging. The second is the motion based methods that aim at classifying face videos based on detecting movements of facial parts, e.g., eye blinking and lip movements. These methods are suitable for static attacks, but not dynamic attacks such as replay or mask attacks. The third is image quality and reflectance-based methods, which design features to capture the superimposed illumination and noise information to the spoof images. The historically influential works in anti-spoofing area contains four major approaches. One, is the texture-based methods which incorporate some handcrafted features such as Hog, and LBP followedby traditional classifiers such as SVM to perform the task.

The temporal-based methods, on the other hand, either use the facial motion patterns (e.g., eye blinking) or involve the movements between face and the background and employ methods such as the optical flow to track the movement of face in order to discriminate real faces from the fake ones. Some 3D structure-based

methods have also been developed which either extract depth information from 2D images, or they analyze the 3D shape information being recorded with 3D sensors and then compare the 3D model of the input sample with that of a genuine face. This method, however requires specific 3D devices which are not easily available and should be costly. Finally, the rPPG (Remote Photoplethysmography) methods extract pulse signal from facial videos without contacting any skin. Nevertheless, all these systems are highly vulnerable against the fake face attacks and masks, and may not cope with these attacks without the auxiliary data assistance, such as depth information, IR. In recent years, the deep learning based methods have been pervasively used for many detection and recognition tasks, as well as anti-spoofing.

### **1.1. Existing system:-**

Face recognition biometrics is now widely employed. Face recognition software should be able to recognize not only people's faces, but also spoofing attempts using printed faces or digital presentations. By presenting a spoofing face of a client to the system's camera, attackers can simply hack a facial recognition system. A spoofing face could be a face mask and a face picture exhibited by a printed photo or a digital display. As a result, effective Face Anti-Spoofing (FAS) approaches are greatly desired and required for the development of safe face recognition systems. The past few years have witnessed much progress in the FAS problem.

Traditionally, in either the Spatial or Fourier space, various techniques have been proposed to extract handcrafted features with image descriptors as representations. These features are usually used to train a Support Vector Machine (SVM) to classify genuine or spoofing examples. However, these features are insufficiently discriminative because those descriptors (e.g., Local Binary Pattern) are not originally designed for the FAS problem. Deep-learning-based strategies, which aim to learn discriminative representations in an end-to-end way, have recently demonstrated to be more effective than standard methods in countering spoofing assaults.

**Disadvantages:-**

1. Traditional Algorithms like SVM are not made for face spoofing
2. Doesn't work efficiently.
3. It gives more accuracy.

**1.2. Proposed System:-**

We propose a novel framework based on the Convolutional Neural Network for the face anti-spoofing problem, which is inspired by the philosophy used by humans to determine whether a presented face example is genuine or not, namely, to look at the example globally first and then carefully observe the local regions to gain more discriminative information (CNN). Specifically, we use deep learning to mimic the behaviour of discovering face-spoofing-related information from image sub-patches. The dataset used is CelebA Spoof dataset from Kaggle.

**Advantages:-**

1. Made for dealing with images.
2. Works efficiently.
3. Have more accuracy.

## CHAPTER 2

### LITERATURE SURVEY

**[1] H. Li, W. Li, H. Cao, S. Wang, F. Huang, and A. C. Kot, “Unsupervised domain adaptation for face anti-spoofing:**

Face anti-spoofing ( presentation attack detection) has recently emerged as an active topic with great significance for both academia and industry due to the rapidly increasing demand in user authentication on mobile phones, PCs, tablets, and so on. Recently, numerous face spoofing detection schemes have been proposed based on the assumption that training and testing samples are in the same domain in terms of the feature space and marginal probability distribution. However, due to unlimited variations of the dominant conditions (illumination, facial appearance, camera quality, and so on) in face acquisition, such single domain methods lack generalization capability, which further prevents them from being applied in practical applications. In light of this, we introduce an unsupervised domain adaptation face anti-spoofing scheme to address the real-world scenario that learns the classifier for the target domain based on training samples in a different source domain. In particular, an embedding function is first imposed based on source and target domain data, which maps the data to a new space where the distribution similarity can be measured. Subsequently, the Maximum Mean Discrepancy between the latent features in source and target domains is minimized such that a more generalized classifier can be learned. State-of-the art representations including both hand-crafted and deep neural network learned features are further adopted into the framework to quest the capability of them in domain adaptation. Moreover, we introduce a new database for face spoofing detection, which contains more than 4000 face samples with a large variety of spoofing types, capture devices, illuminations, and so on. Extensive experiments on existing benchmark databases and the new database verify that the proposed approach can gain significantly better generalization capability in cross- domain scenarios by providing consistently better anti-spoofing performance

**Summary:** We propose a novel framework utilizing advanced unsupervised domain adaptation algorithms for face anti-spoofing. The novelty of this framework lies in transferring the feature space of face samples from the labeled source domain to the

unlabeled target domain, such that reliable model can be learned for spoofing detection. State-of-the-art hand-crafted and deep learning based features are incorporated into the domain adaptation framework and their classification accuracies are further evaluated. Extensive experiments have been conducted based on the available databases and our new database. The results show that we can achieve clearly improved generalization ability with an average of 20% improvement by domain adaptation as compared with the straightforward learning approach without domain adaptation.

**[2]R. Nosaka, Y. Ohkawa, and K. Fukui, “Feature extraction based on co-occurrence of adjacent local binary patterns,” in *Advances in Image and Video Technology*:**

In this paper, we propose a new image feature based on spa-tial co-occurrence among micro patterns, where each micro pattern is rep-resented by a Local Binary Pattern (LBP). In conventional LBP-based features such as LBP histograms, all the LBPs of micro patterns in the image are packed into a single histogram. Doing so discards important information concerning spatial relations among the LBPs, even though they may contain information about the image’s global structure. To consider such spatial relations, we measure their co-occurrence among multiple LBPs. The proposed feature is robust against variations in illumination, a feature inherited from the original LBP, and simultaneously retains more detail of image. The significant advantage of the pro-posed method versus conventional LBP-based features is demonstrated through experimental results of face and texture recognition using public databases.

**Summary:** We have proposed a novel image feature based on the spatial co-occurrence of micro patterns, which are represented by Local Binary Pattern (LBP). The conventional LBP-based features as represented by the LBP histogram still have room for performance improvements. In particular, expression ability for a given image can be improved, since all LBPs are simply summed into a single histogram, thereby discarding spatial relations among the LBPs and the rich image information they contain. To improve their performance, we introduced the extension of original LBP by considering the co-occurrence of adjacent LBPs, measuring co-occurrence with an auto-correlation matrix generated from multiple LBPs. The proposed feature

is robust against variations in illumination, because it only depends on the magnitude relation between a center pixel and its surrounding pixels. Experimental results of face and texture recognition tasks using public Feature Extraction Based on Co-occurrence of Adjacent LBPs 91databases have demonstrated a significant advantage of the proposed feature against conventional LBP-based features.

**[3] I. Chingovska, A. Anjos, and S. Marcel, “On the effectiveness of local binary patterns in face anti-spoofing:**

Spoofing attacks are one of the security traits that biometric recognition systems are proven to be vulnerable to. When spoofed, a biometric recognition systems bypassed by presenting a copy of the biometric evidence of a valid user. Among all biometric modalities, spoofing a face recognition system is particularly easy to perform: all that is needed is a simple photograph of the user. In this paper, we address the problem of detecting face spoofing attacks. In particular, we inspect the potential of texture features based on Local Binary Patterns (LBP) and their variations on three types of attacks: printed photographs, and photos and videos displayed on electronic screens of different sizes. For this purpose, we introduce REPLAY- ATTACK, a novel publicly available face spoofing database which contains all the mentioned types of attacks. We conclude that LBP, with  $\sim 15\%$  Half Total Error Rate, show moderate discriminability when confronted with a wide set of attack types.

**Summary:** Spoofing and anti-spoofing has become a prevalent topic in the biometrics community. Regardless of the sophistication of a particular face recognition system, it should not be completely trusted if it does not have a protection against spoofing attacks. The contributions of this paper can be summarized as follows. Firstly, it introduces REPLAY-ATTACK, a novel spoofing attack database containing three types of possible attacks using three different media and two different recording conditions. The database includes a protocol for training, development and testing purposes, and also proves the vulnerability of a baseline face recognition system to its attacks. Secondly, it proposes simple and easily reproducible LBP based face spoofing counter-measure and explores its efficiency against a variety of attacks. Variants of LBP were also investigated, but the regular  $LBP_{u23 \times 3}$  shows the best performance/complexity tradeoff. The simple setup and low-dimensional features manage to reach reasonable performance even without using complex non-linear

classifiers whose size can be inconvenient for fast computation. In support of reproducible research, the database, its protocols, as well as the source code will be made publicly available. The LBP based anti-spoofing method guarantees different levels of certainty for different types of attacks and different databases. Some attacks can deceive this counterfeit more easily than others. There is no consistency in the results with regards to the types of attacks, nor the attacks from different databases. Our belief is that this is not valid only for texture-based methods, but also for methods that approach the problem from a different aspect. The various face spoofing attacks differ from the real accesses in their own particular manner: the devices that are used introduce different artifacts and the amount and type of movement they possess is different. In other words, the cues that distinguish two different types of face spoofing attacks from real accesses differ in their essence and should be grasped in their own unique way. There is not a single notion which describes all the types of attacks. Hence, we believe that the future work in the field of anti-spoofing should focus on addressing as more spoofing attacks as possible with separate attack-specific approaches. Another option is to congregate the characteristics of all the real accesses into a single model to which none of the spoofing attacks will relate in any sense.

**[4] Z. Boulkenafet, J. Komulainen, and A. Hadid, "Face spoofing detection using colour texture analysis,:"**

Research on non-intrusive software based facespoofing detection schemes has mainly been focused on the analysis of the luminance information of the face images, hence discarding the Chroma component which can be very useful for discriminating fake faces from genuine ones. This work introduces a novel and appealing approach for detecting facespoofing using color texture analysis. We exploit the joint color-texture information from the luminance and the chrominance channels by extracting complementary low-level feature descriptions from different color spaces. More specifically, the feature histograms are computed over each image band separately. Extensive experiments on the three most challenging benchmark datasets, namely the CASIA Face Anti-Spoofing Database, the Replay-Attack Database and MSU Mobile Face Spoof Database, showed excellent results compared to the state of the art. More importantly, unlike most of the methods proposed in the literature, our proposed approach is able to achieve stable performance across all the three benchmark datasets. The promising results of our cross-database evaluation suggest that facial

color texture representation is more stable in unknown conditions compared to its gray-scale counterparts

**Summary:**In this article, we proposed to approach the problem of fascinate-spoofing from the colour texture analysis point of view. We investigated how well different colour image representations (RGB, HSV and YCbCr) can be used for describing the intrinsic disparities in the colour texture between genuine faces and fake ones and if they provide complementary presentations. The effectiveness of the different facial colour texture representations was studied by extracting different local descriptors from the individual image channels in the different colour spaces.

**[5] X. Tan, Y. Li, J. Liu, and L. Jiang, “Face liveness detection from a single image with sparse low rank bilinear discriminative model**

Spoofing with photograph or video is one of the most common manner to circumvent a face recognition system. In this paper, we present a real-time and non-intrusive method to address this based on individual images from a generic webcam. The task is formulated as a binary classification problem, in which, however, the distribution of positive and negative are largely overlapping in the input space, and a suitable representation space is hence of importance. Using the Lambertian model, we propose two strategies to extract the essential information about different surface properties of a live human face or a photograph, in terms of latent samples. Based on these, we develop two new extensions to the sparse logistic regression model which allow quick and accurate spoof detection. Primary experiments on a large photo imposter database show that the proposed method gives preferable detection performance compared to others.

**Summary:**

Spoofing with photograph or video is one of the most common manner to circumvent a face recognition system. In this paper, we present a real-time and non-intrusive method to address this based on individual images from a generic webcam. The task is formulated as a binary classification problem, in which, however, the distribution of positive and negative are largely overlapping in the input space, and a suitable representation space is hence of importance. Using the Lambertian model, we propose two strategies to extract the essential information about different surface



properties of a live human face or a photograph, in terms of latent samples. Based on these, we develop two new extensions to the sparse logistic regression model which allow quick and accurate spoof detection. Primary experiments on a large photo imposter database show that the proposed method gives preferable detection performance compared to others.

## CHAPTER-3

### ANALYSIS

#### 3.1. 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

##### **Economic 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

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.

## 3.2. System Requirements Specification

### 3.2.1. Functional and non-functional requirements:

Requirement's analysis is very critical process that enables the success of a system or software project to be assessed. Requirements are generally split into two types: Functional and non-functional requirements.

**Functional Requirements:** These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

Examples of functional requirements:

- 1) Authentication of user whenever he/she logs into the system
- 2) System shutdown in case of a cyber-attack
- 3) A verification email is sent to user whenever he/she register for the first time on some software system.

**Non-functional requirements:** These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements.

They basically deal with issues like:

- Portability
- Security
- Maintainability
- Reliability

- Scalability
- Performance
- Reusability
- Flexibility

Examples of non-functional requirements:

- 1) Emails should be sent with a latency of no greater than 12 hours from such an activity.
- 2) The processing of each request should be done within 10 seconds
- 3) The site should load in 3 seconds whenever of simultaneous users are > 10000

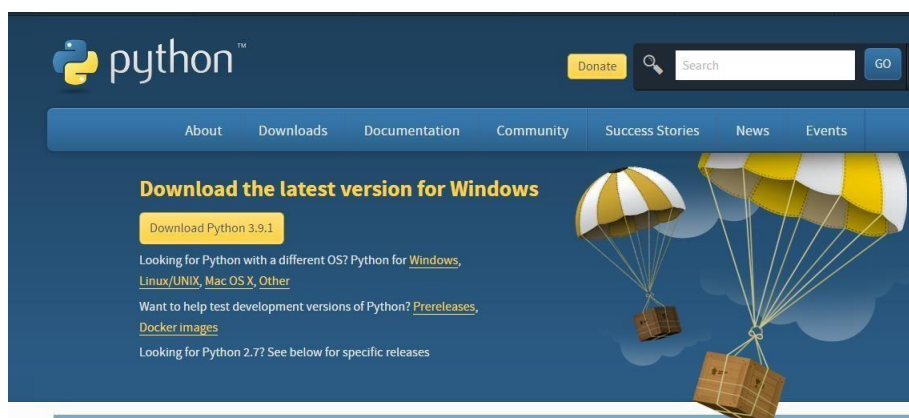
### 3.2.2. Software and Hardware Requirements:

Operating system	: Windows 7 or 7+
Ram	: 8 GB
Hard disc or SSD	: More than 500 GB
Processor	: Intel 3rd generation or high or Ryzen with 8 GB Ram
Software's	: Python 3.6 or high version, Visual studio, PyCharm.

### 3.3. Software Installation

#### 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".

### **Libraries Used :**

#### **NumPy:**

NumPy stands for 'Numerical Python' or 'Numeric Python'. It is an open source module of Python which provides fast mathematical computation on arrays and matrices. Since arrays and matrices are an essential part of the Machine Learning ecosystem, NumPy along with Machine Learning modules like Scikit-learn, Pandas, Matplotlib, TensorFlow, etc. complete the Python Machine Learning Ecosystem. NumPy provides the essential multi-dimensional array-oriented computing functionalities designed for high-level mathematical functions and scientific computation. NumPy can be imported into the notebook using

**import numpy as np**

#### **Pandas:**

Similar to NumPy, Pandas is one of the most widely used python libraries in data science. It provides high-performance, easy to use structures and data analysis tools. Pandas provides an in-memory 2d table object called Data frame. It is like a spreadsheet with column names and row labels. Hence, with 2d tables, pandas are capable of providing many additional functionalities like creating pivot tables, computing columns based on other columns and plotting graphs. Pandas can be imported into Python using:

**import pandas as pd**

**Matplotlib:**

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy One of the greatest benefits of visualization is Face spoofing using Deep Learning that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc. Matplotlib comes with a wide variety of plots. Plots help to understand trends, patterns, and to make correlations. They're typically instruments for reasoning about quantitative information. Matplotlib can be imported into Python using:

```
import matplotlib.pyplot as plt
```

**Seaborn:**

Seaborn is a data visualization library built on top of matplotlib and closely integrated with pandas data structures in Python. Visualization is the central part of Seaborn which helps in exploration and understanding of data. Seaborn offers the functionalities like Dataset oriented API to determine the relationship between variables, Automatic estimation and plotting of linear regression plots, It supports high-level abstractions for multi-plot grids and Visualizing univariate and bivariate distribution. It can be imported into Python using `import seaborn as sns`.

**Sklearn:**

Scikit-learn is a free software machine library for Python programming language. It features various classification, regression and clustering algorithms including Linear regression, Polynomial Regression and XGBoost Regression. In our project we have used different features of sklearn library like:

```
from sklearn.model_selection import train_test_split
```

**Installing PyCharm:**

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

## Download PyCharm

[Windows](#)[Mac](#)[Linux](#)

### Professional

For both Scientific and Web Python development. With HTML, JS, and SQL support.

[Download](#)

Free trial

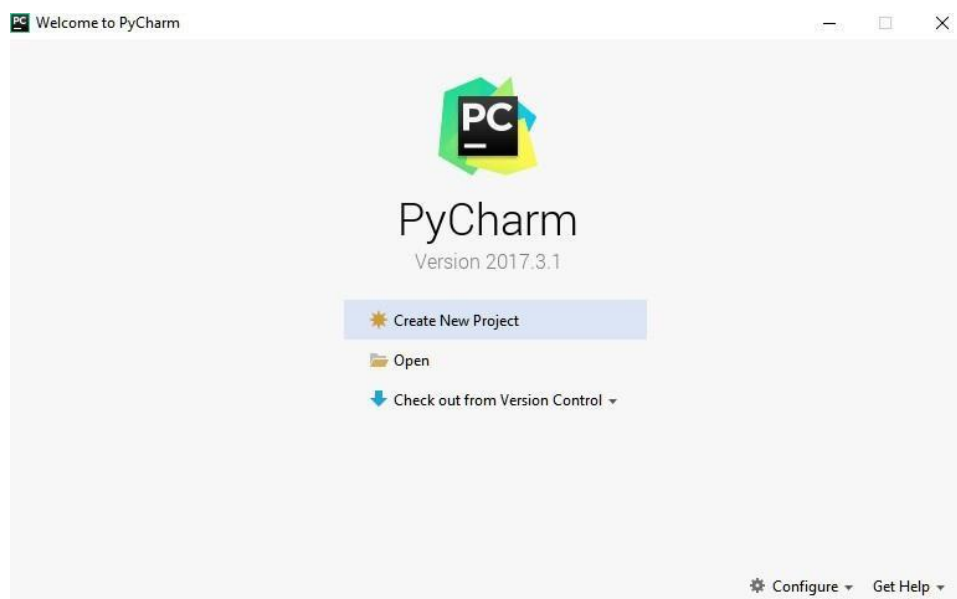
### Community

For pure Python development

[Download](#)

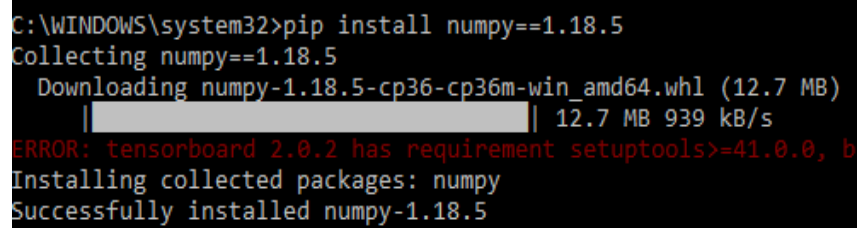
Free, open-source

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 JetBrains 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, sea born, scikit-learn, Matplotlib, Pyplot)

Ex: Pip install NumPy



```
C:\WINDOWS\system32>pip install numpy==1.18.5
Collecting numpy==1.18.5
  Downloading numpy-1.18.5-cp36-cp36m-win_amd64.whl (12.7 MB)
    |████████████████████| 12.7 MB 939 kB/s
ERROR: tensorboard 2.0.2 has requirement setuptools>=41.0.0, but you have setuptools 40.6.0
Installing collected packages: numpy
Successfully installed numpy-1.18.5
```

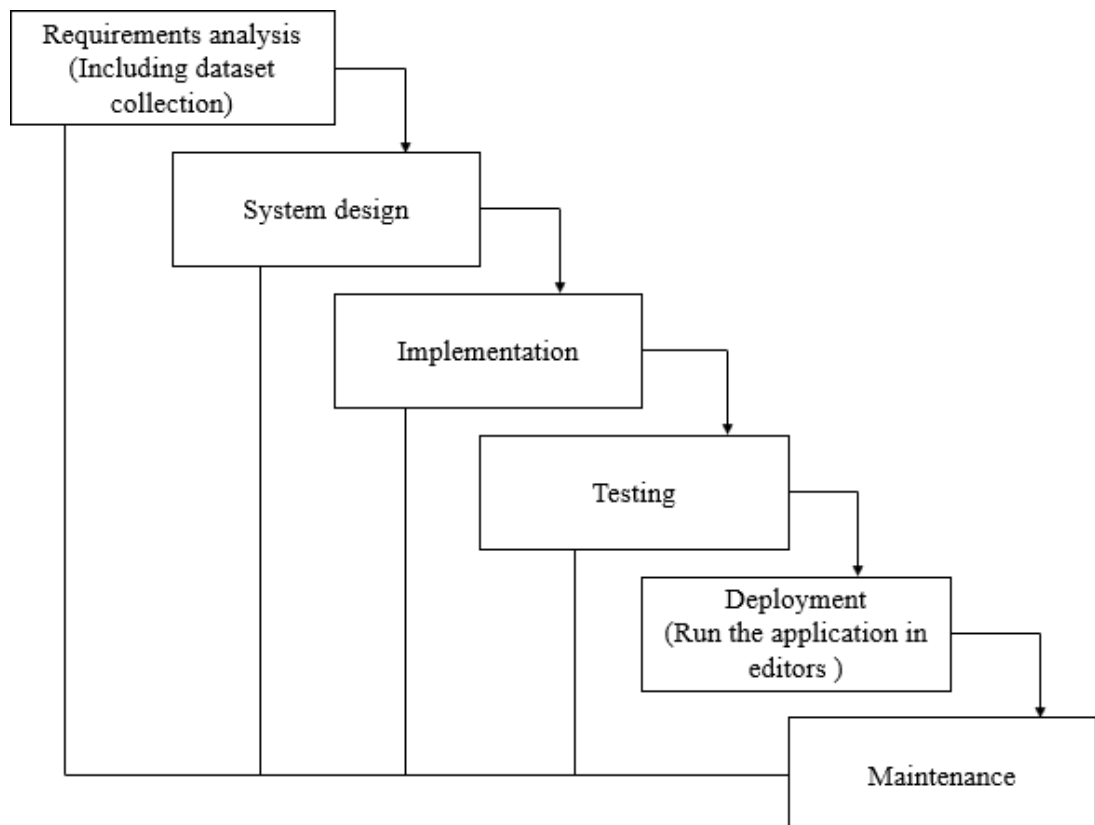


## CHAPTER-4

### SOFTWARE DEVELOPMENT LIFE CYCLE

#### Waterfall Model:

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



**Fig.4.1: Waterfall Model**

- **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.

## **CHAPTER-5**

### **DESIGN**

#### **5.1. System Design:**

##### **Input Design:**

In an information system, input is the raw data that is processed to produce output. During the input design, the developers must consider the input devices such as PC, MICR, OMR, etc.

Therefore, the quality of system input determines the quality of system output. Well-designed input forms and screens have following properties –

- It should serve specific purpose effectively such as storing, recording, and retrieving the information.
- It ensures proper completion with accuracy.
- It should be easy to fill and straightforward.
- It should focus on user's attention, consistency, and simplicity.
- All these objectives are obtained using the knowledge of basic design principles regarding –
  - What are the inputs needed for the system?
  - How end users respond to different elements of forms and screens.

##### **Objectives for Input Design:**

The objectives of input design are –

- To design data entry and input procedures
- To reduce input volume
- To design source documents for data capture or devise other data capture methods
- To design input data records, data entry screens, user interface screens, etc.
- To use validation checks and develop effective input controls.

**Output Design:**

The design of output is the most important task of any system. During output design, developers identify the type of outputs needed, and consider the necessary output controls and prototype report layouts.

**Objectives of Output Design:**

The objectives of input design are:

1. To develop output design that serves the intended purpose and eliminates the production of unwanted output.
2. To develop the output design that meets the end user's requirements.
3. To deliver the appropriate quantity of output.
4. To form the output in appropriate format and direct it to the right person.
5. To make the output available on time for making good decisions.

**Modules:****1. User:****Data gathering:**

Needs to gather the information or data from the open source, this will be use in the train the models.

**Pre-processing:**

Data need to be pre-processed according the models it helps to increase the accuracy of the model and better information about the data.

**Feature Engineering:**

In this step features are selected based on the priority of the column data, by this we can reduce the time investing on many columns.

**Model Building**

To get the final result model building for the dataset is an important step.

Based on the dataset we build the model for classification and regression.

**View Results**

User view's the generated results from the model.

**2. System****Model Checking**

System checks model accuracy and it takes of the necessary for the model building

### **Generate Results**

System takes the input data from the users and produces the output.

## **5.2. UML Introduction:**

The unified modeling language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic, semantic and pragmatic rules. A UML system is represented using five different views that describe the system from a distinctly different perspective. UML is specifically constructed through two different domains, they are:

- UML Analysis modeling, this focuses on the user model and structural model views of the systems.
- UML Design modeling, which focuses on the behavioral modeling, implementation modeling and environmental model views.

### **5.2.1. Usage of UML in Project:**

As the strategic value of software increases for many companies, the industry looks for techniques to automate the production of software and to improve quality and reduce cost and time to the market. These techniques include component technology, visual programming, patterns and frameworks. Additionally, the development for the World Wide Web, while making some things simpler, has exacerbated these architectural problems. The UML was designed to respond to these needs. Simply, systems design refers to the process of defining the architecture, components, modules, interfaces and data for a system to satisfy specified requirements which can be done easily through UML diagrams

## **5.3. UML Diagrams**

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling 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 Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modelling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modelling 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 modelling 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 modelling 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.

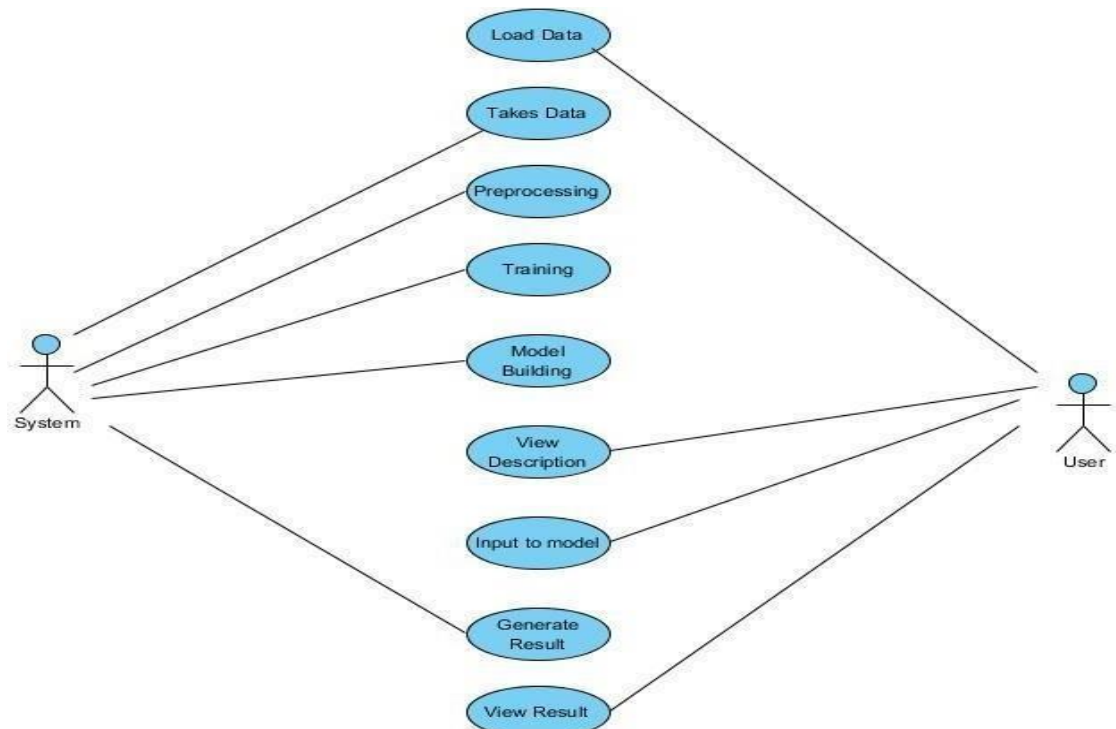


Fig 5.1: Use Case Diagram

### 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 5.2: 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 timing diagrams

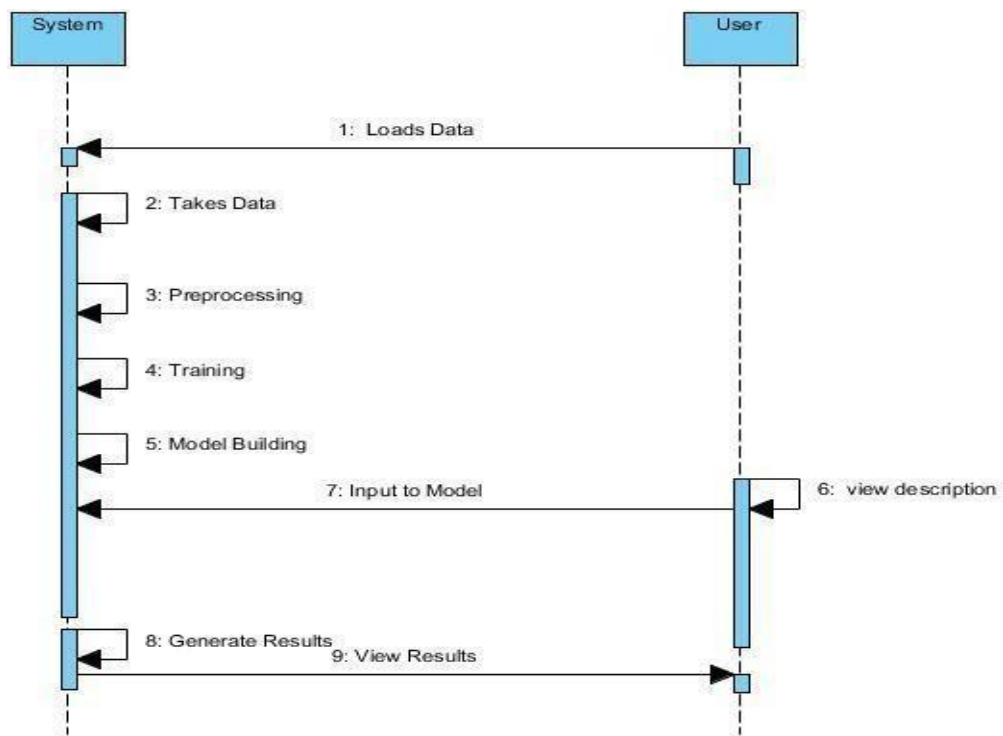
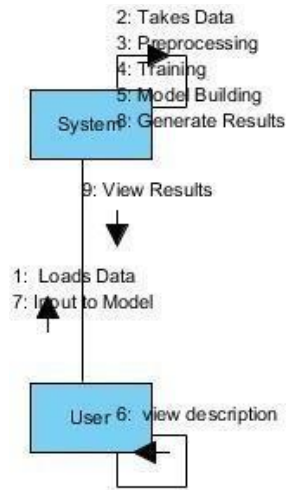


Fig 5.3: Sequence Diagram

### 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.

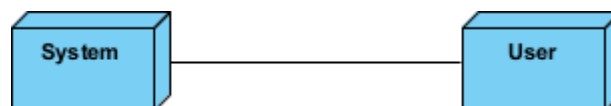




**Fig 5.4: Collaboration Diagram**

### 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 5.5: Deployment 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 Modelling 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.

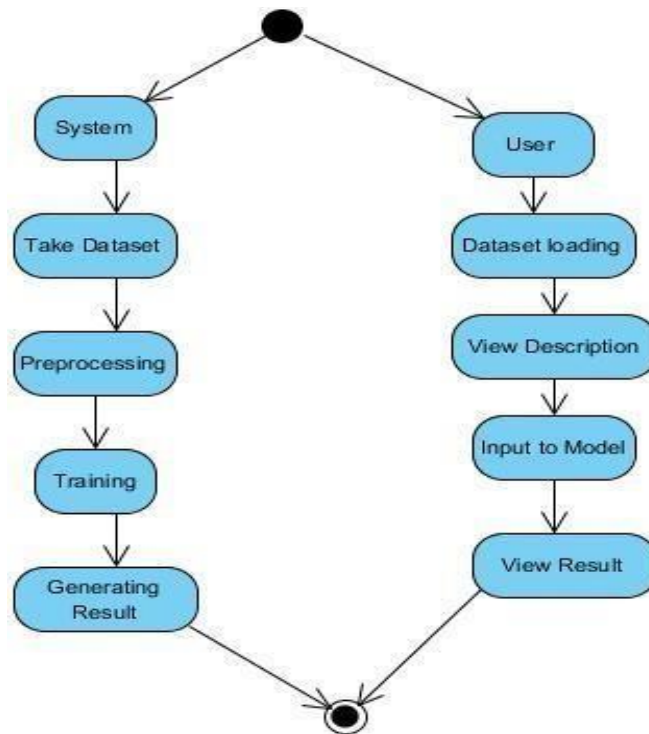


Fig 5.6: Activity Diagram

**Component Diagram:**

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

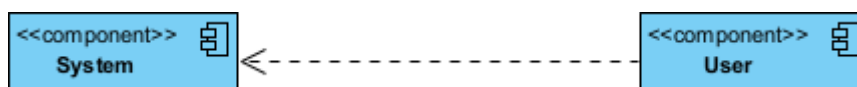


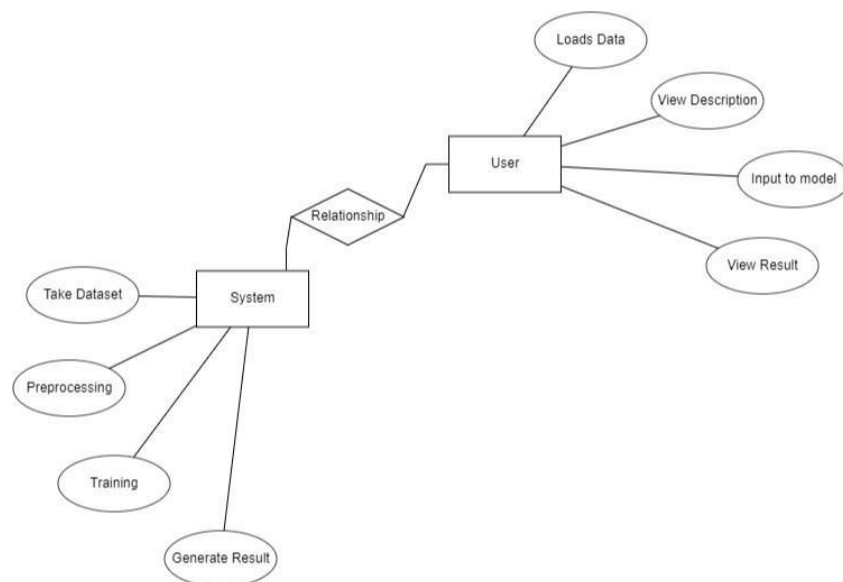
Fig 5.7: Component Diagram

**Entity Relationship 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 5.8: ER Diagram**

### **Data Flow 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.

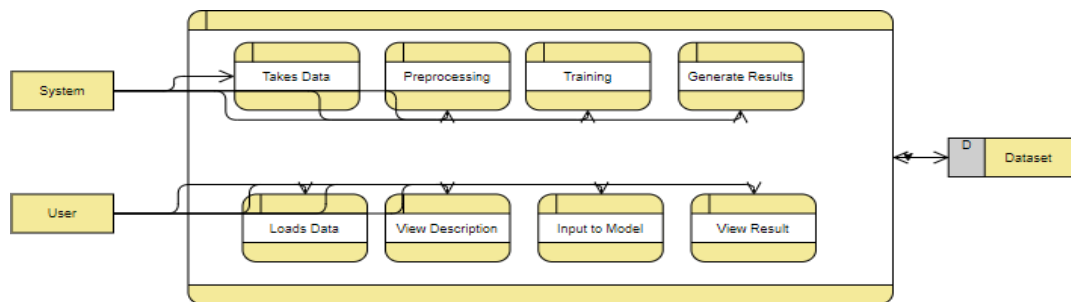
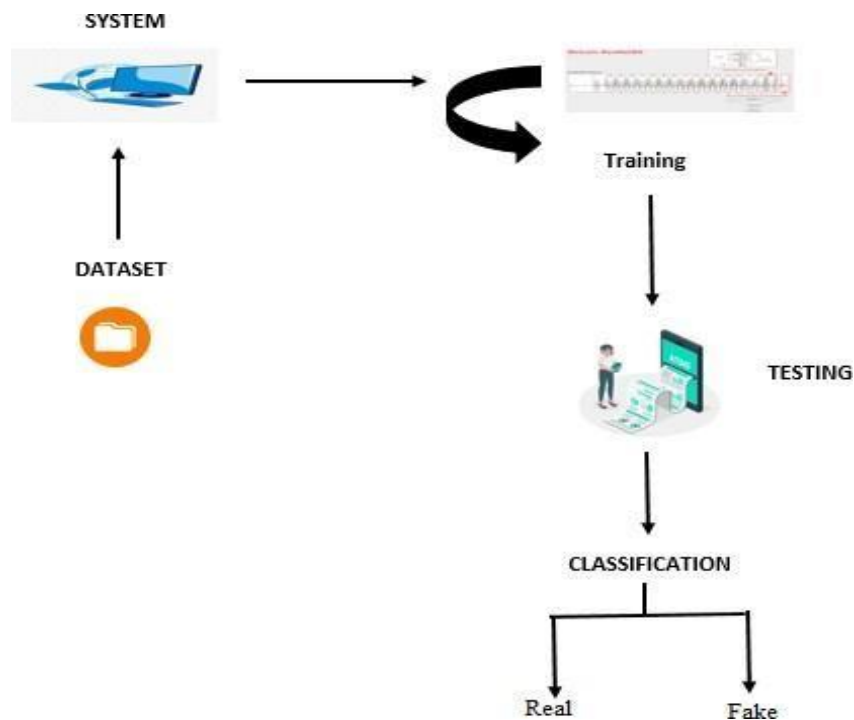
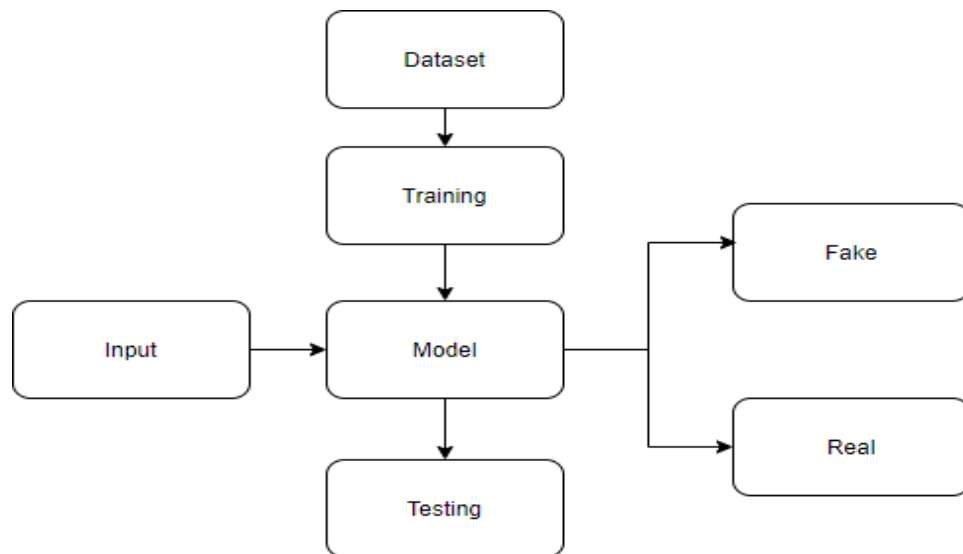


Fig 5.9: Data Flow Diagram

## 5.4. Architecture



## 5.5. Block Diagram



## **CHAPTER-6**

### **IMPLEMENTATION**

#### **6.1. Data Analysis**

**Data Analysis** One of the first steps we perform during implementation is an analysis of the data. The image datasets are collected from various rice plants.

##### **Acquisition of Training Dataset:**

The accuracy of any deep learning algorithm depends on the correctness of the training dataset. We in this project analyzed multiple datasets collected from Kaggle that would give the best results. Many works done in this field have considered image datasets to predict whether the rice leaf has been affected by disease or not.

#### **6.2. Data Pre-Processing**

Data Pre-Processing is a Data Mining method that entails converting raw data into a format that can be understood. Real-world data is frequently inadequate, inconsistent, and/or lacking in specific activities or trends, as well as including numerous inaccuracies. This might result in low-quality data collection and, as a result, low-quality models based on that data. Preprocessing data is a method of resolving such problems.

Machines do not comprehend free text, image, or video data; instead, they comprehend 1s and 0s. So putting on a slideshow of all our photographs and expecting our machine learning model to learn from it is probably not going to be adequate.

Data Pre-processing is the step in any Machine Learning process in which the data is changed, or encoded, to make it easier for the machine to parse it. In other words, the algorithm can now easily interpret the data's features.

Data Pre-processing can be done in four different ways. Data cleaning/cleaning, data integration, data transformation, and data reduction are the four categories.



**Fig 6.1: Data Pre-processing**

### 6.2.1. Data Cleaning

Data in the real world is frequently incomplete, noisy, and inconsistent. Many bits of the data may be irrelevant or missing. Data cleaning is carried out to handle this aspect. Data cleaning methods aim to fill in missing values, smooth out noise while identifying outliers, and fix data discrepancies. Unclean data can confuse data and the model. Therefore, running the data through various Data Cleaning/Cleansing methods is an important Data Pre-processing step.



#### (a) **Missing Data:**

It's fairly common for your dataset to contain missing values. It could have happened during data collection or as a result of a data validation rule, but missing values must be considered anyway.

- 1. Dropping rows/columns:** If the complete row is having Nan values, then it doesn't make any value out of it. So, such rows/columns are to be dropped immediately. Or if the % of row/column is mostly missing say about more than 65% then also one can choose to drop.

**2. Checking for duplicates:** If the same row or column is repeated then also you can drop it by keeping the first instance. So that while running machine learning algorithms, so as not to offer that particular data object an advantage or bias.

**3. Estimate missing values:** If only a small percentage of the values are missing, basic interpolation methods can be used to fill in the gaps. However, the most typical approach of dealing with missing data is to fill them in with the feature's mean, median, or mode value.

**(b) Noisy Data:**

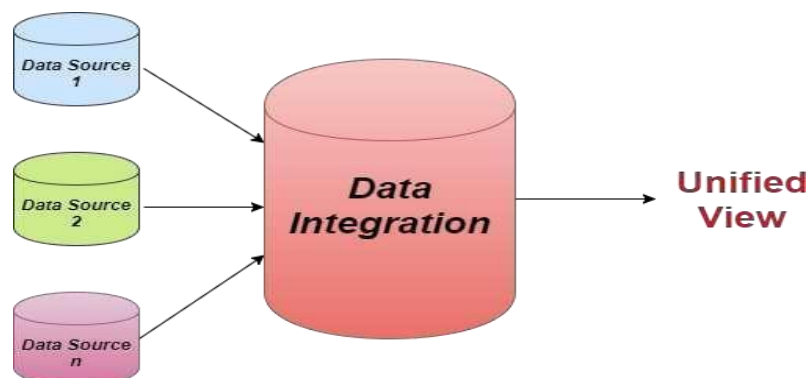
Noisy data is meaningless data that machines cannot interpret. It can be caused by poor data collecting, data input problems, and so on. It can be dealt with in the following ways:

1. **Binning Method:** This method smooths data that has been sorted. The data is divided into equal-sized parts, and the process is completed using a variety of approaches. Each segment is dealt with independently. All data in a segment can be replaced by its mean, or boundary values can be used to complete the task.

2. **Clustering:** In this method, related data is grouped in a cluster. Outliers may go unnoticed, or they may fall outside of clusters.

3. **Regression:** By fitting data to a regression function, data can be smoothed out. The regression model employed may be linear (with only one independent variable) or multiple (with numerous independent variables) (having multiple independent variables).

### 6.2.2. Data Integration



**Fig 6.2: Data Integration**



It is involved in a data analysis task that combines data from multiple sources into a coherent datastore. These sources may include multiple databases. Do you think how data can be matched up ?? For a data analyst in one database, he finds Customer\_ID and in another he finds cust\_id, How can he be sure about them and say these two belong to the same entity. Databases and Data warehouses have Metadata (It is the data about data) it helps in avoiding errors.

### **6.2.3. Data Transformation**

This stage is used to convert the data into a format that can be used in the mining process. This is done in the following ways:

#### **1. Normalization:**

It is done to scale the data values in a specified range (-1.0 to 1.0 or 0.0 to 1.0)

#### **2. Concept Hierarchy Generation:**

Using concept hierarchies, low-level or primitive/raw data is substituted with higher-level concepts in data generalization. Categorical qualities, for example, are generalized to higher-level notions such as street, city, and nation. Similarly, numeric attribute values can be translated to higher-level concepts like age, such as youthful, middle-aged, or elderly.

#### **3. Smoothing:**

Smoothing works to remove the noise from the data. Such techniques include binning, clustering, and regression.

#### **4. Aggregation:**

Aggregation is the process of applying summary or aggregation operations on data. Daily sales data, for example, might be combined to calculate monthly and annual totals. Feature Aggregation — If the features are highly correlated or if the features can be aggregated into another single feature then it is worth doing it. For example in the dataset you have the height and width of an object then they can be featured into a single feature area. This decreases dimensionality. These types of features are highly correlated in nature as a result it also decreases multi collinearity.

#### **6.2.4. Data Reduction:**

Because data mining is a methodology for dealing with large amounts of data. When dealing with large amounts of data, analysis becomes more difficult. We employ a data reduction technique to get rid of this. Its goal is to improve storage efficiency while lowering data storage and analysis expenses.

##### **1.Dimensionality Reduction :**

A huge number of features may be found in most real-world datasets. Consider an image processing problem: there could be hundreds of features, also known as dimensions, to deal with. As the name suggests, dimensionality reduction seeks to minimize the number of features — but not just by selecting a sample of features from the feature set, which is something else entirely — Feature Subset Selection or feature selection.

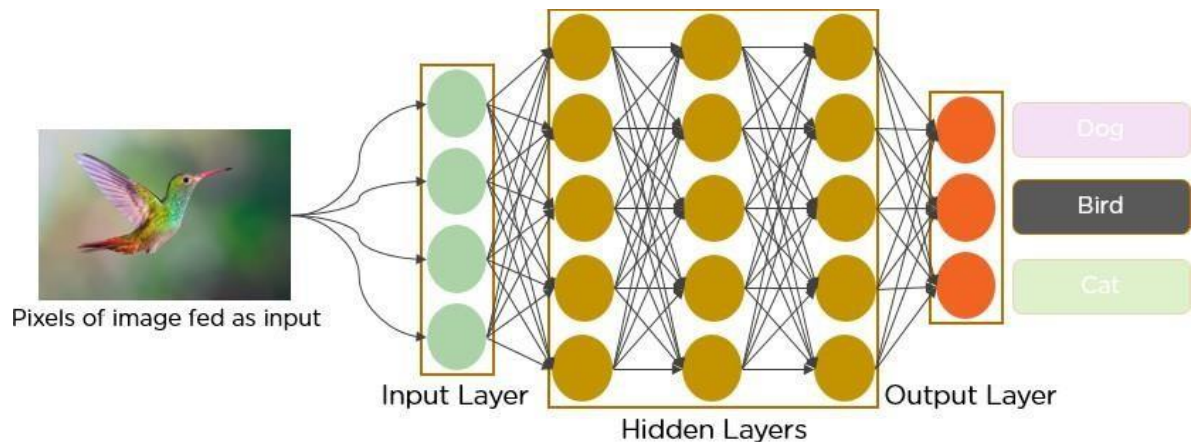
##### **2.Numerosity Reduction:**

Data is replaced or estimated using alternative and smaller data representations such as parametric models (which store only the model parameters rather than the actual data, such as Regression and Log-Linear Models) or non-parametric approaches (e.g. Clustering, Sampling, and the use of histograms).

### **Methodology and Algorithms:**

#### **1.CNN**

In the past few decades, Deep Learning has proved to be a very powerful tool because of its ability to handle large amounts of data. The interest to use hidden layers has surpassed traditional techniques, especially in pattern recognition. One of the most popular deep neural networks is Convolutional Neural Networks.



Since the 1950s, the early days of AI, researchers have struggled to make a system that can understand visual data. In the following years, this field came to be known as Computer Vision. In 2012, computer vision took a quantum leap when a group of researchers from the University of Toronto developed an AI model that surpassed the best image recognition algorithms and that too by a large margin.

The AI system, which became known as AlexNet (named after its main creator, Alex Krizhevsky), won the 2012 ImageNet computer vision contest with an amazing 85 per cent accuracy. The runner-up scored a modest 74 per cent on the test.

At the heart of AlexNet was Convolutional Neural Networks a special type of neural network that roughly imitates human vision. Over the years CNNs have become a very important part of many Computer Vision applications and hence a part of any computer vision course online. So let's take a look at the workings of CNNs.

### **Background of CNNs**

CNN's were first developed and used around the 1980s. The most that a CNN could do at that time was recognizing handwritten digits. It was mostly used in the postal sectors to read zip codes, pin codes, etc. The important thing to remember about any deep learning model is that it requires a large amount of data to train and also requires a lot of computing resources. This was a major drawback for CNNs at that period and hence CNNs were only limited to the postal sectors and it failed to enter the world of machine learning.

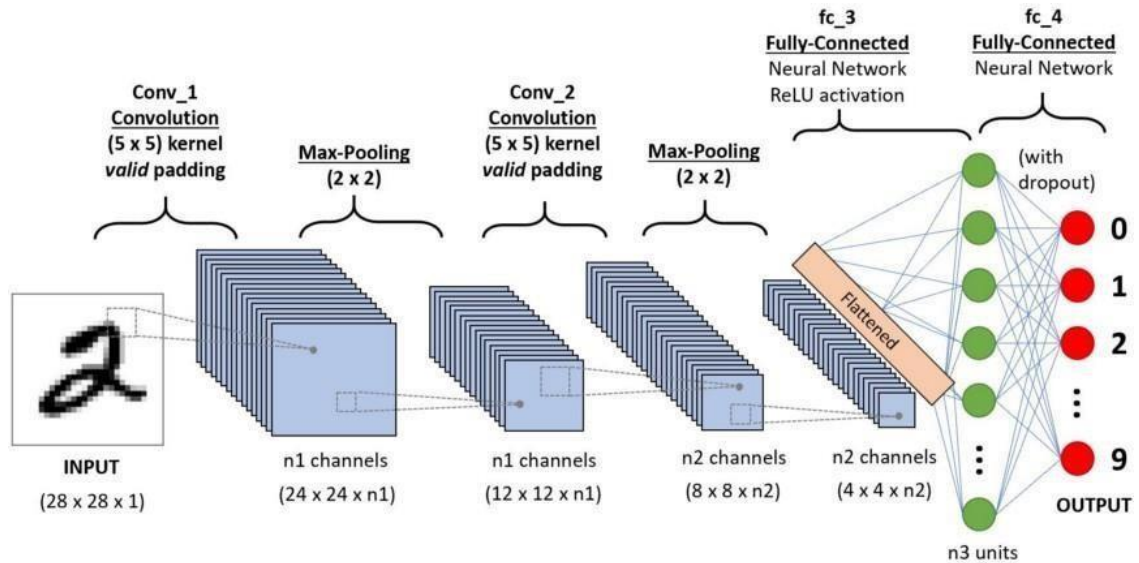
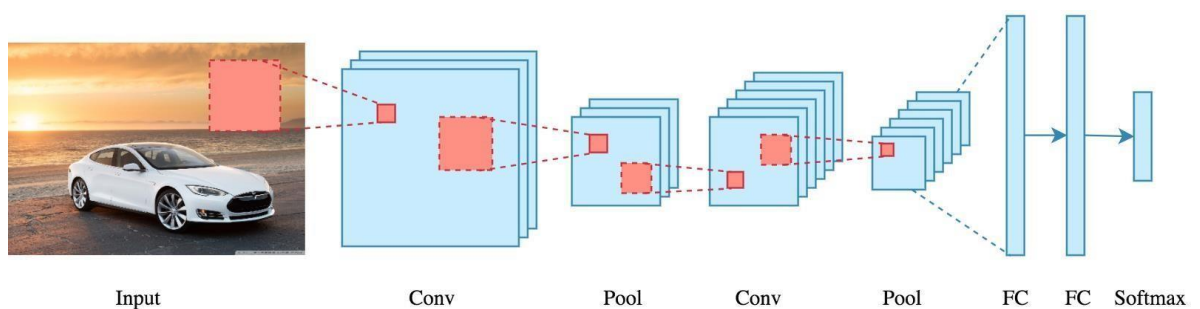


Fig 6.3: CNN Model

In 2012 Alex Krizhevsky realized that it was time to bring back the branch of deep learning that uses multi-layered neural networks. The availability of large sets of data, to be more specific ImageNet datasets with millions of labelled images and an abundance of computing resources enabled researchers to revive CNNs.

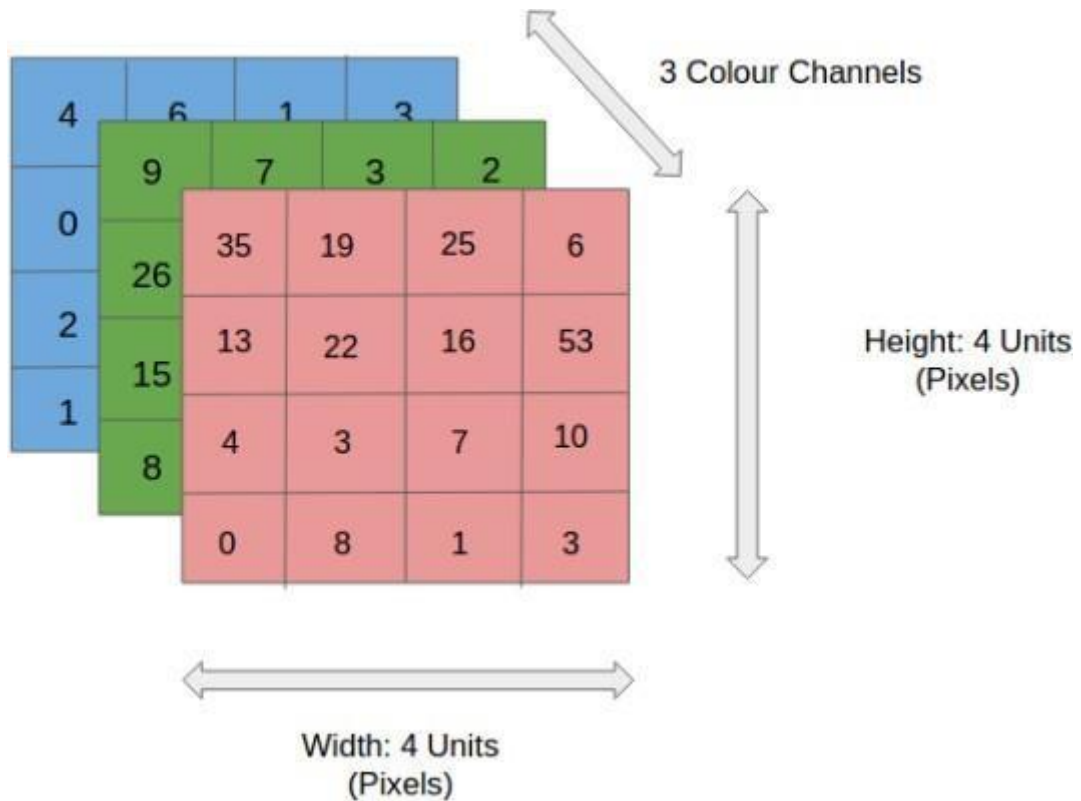
### What exactly is a CNN?

In deep learning, a **convolutional neural network (CNN/ConvNet)** is a class of deep neural networks, most commonly applied to analyse visual imagery. Now when we think of a neural network we think about matrix multiplications but that is not the case with ConvNet. It uses a special technique called Convolution. Now in mathematics **convolution** is a mathematical operation on two functions that produces a third function that expresses how the shape of one is modified by the other.

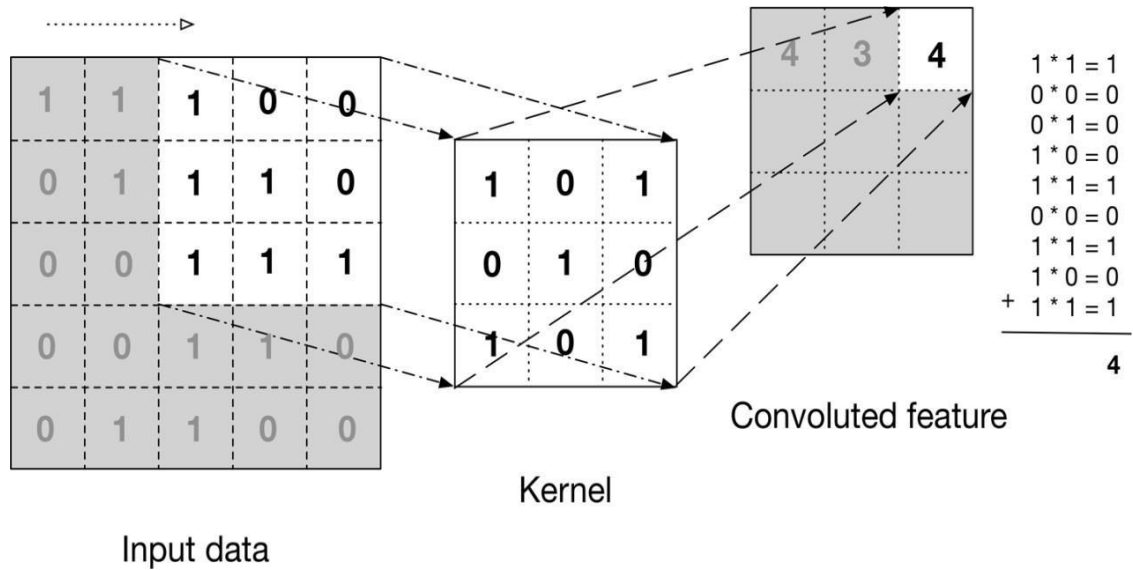


### How does it work?

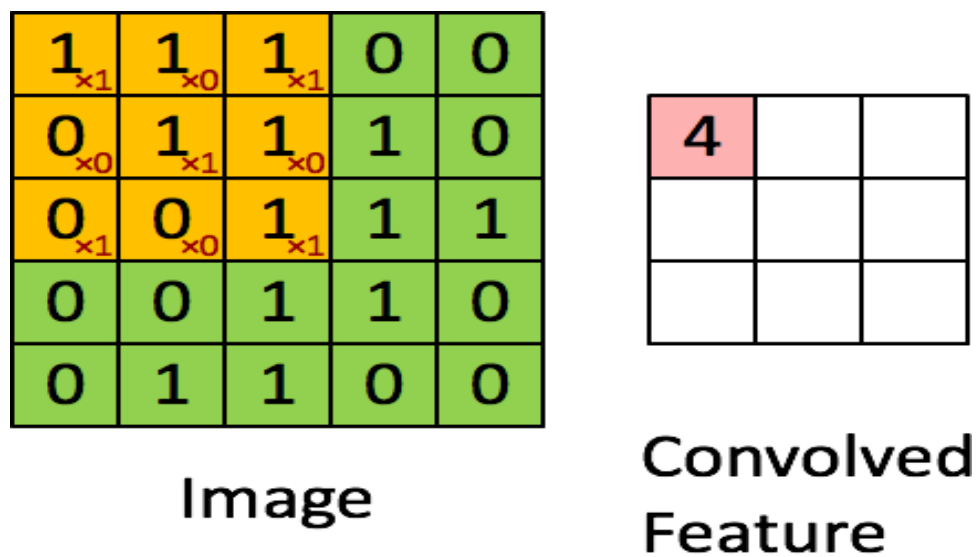
Before we go to the working of CNN's let's cover the basics such as what is an image and how is it represented. An RGB image is nothing but a matrix of pixel values having three planes whereas a gray scale image is the same but it has a single plane. Take a look at this image to understand more.



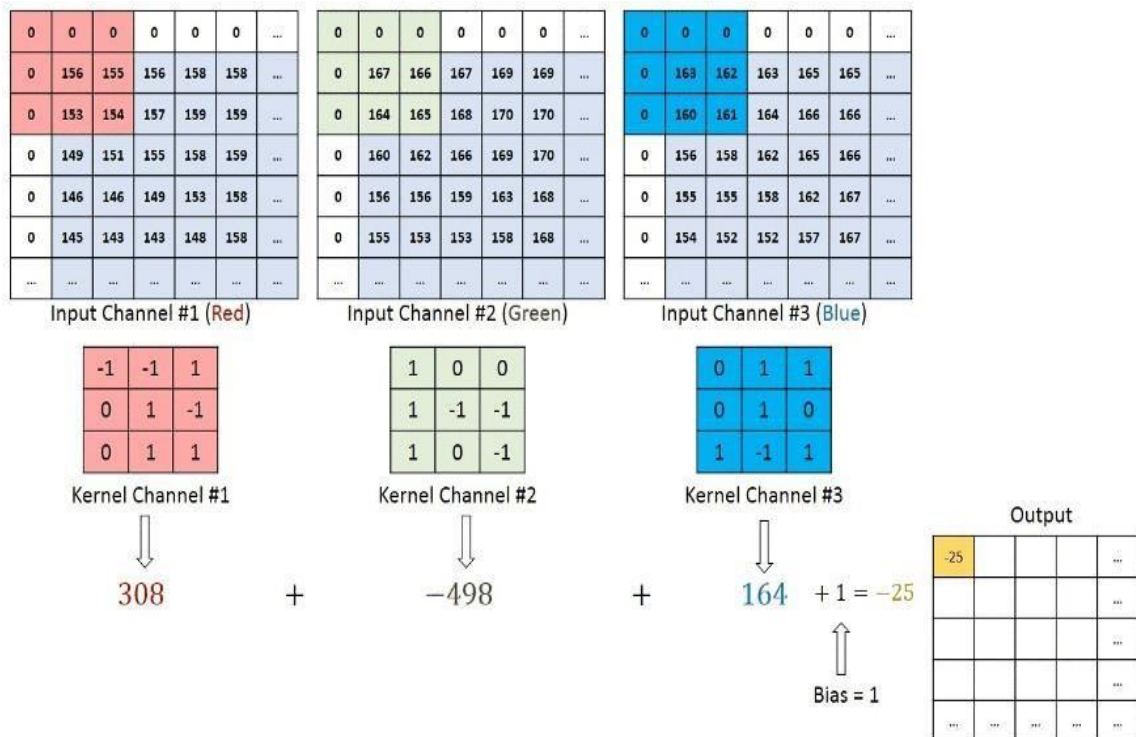
For simplicity, let's stick with gray scale images as we try to understand how CNNs work.



The above image shows what a convolution is. We take a filter/kernel (3×3 matrix) and apply it to the input image to get the convolved feature. This convolved feature is passed on to the next layer.



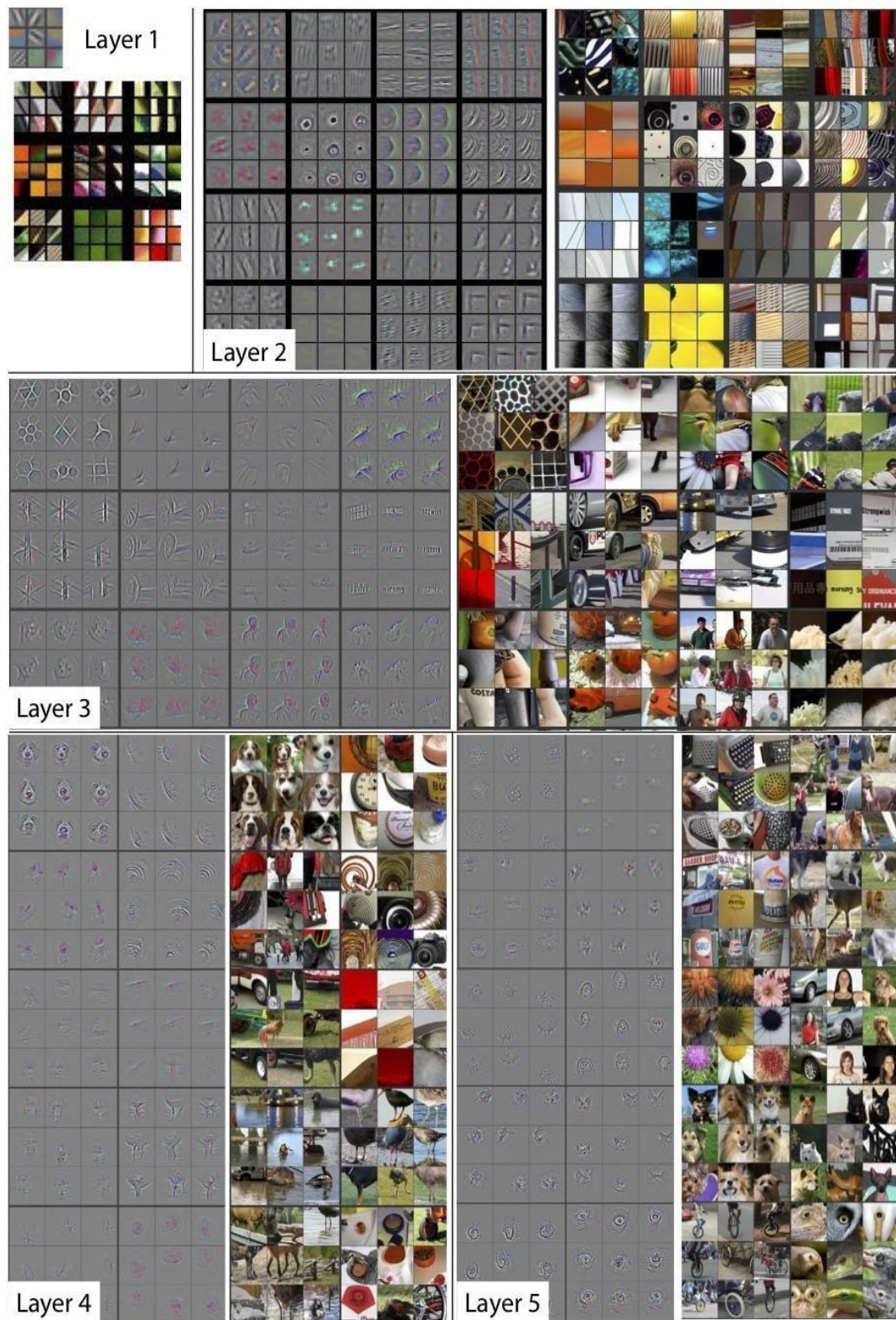
In the case of RGB colour, channel take a look at this animation to understand its working



Convolutional neural networks are composed of multiple layers of artificial neurons. Artificial neurons, a rough imitation of their biological counterparts, are mathematical functions that calculate the weighted sum of multiple inputs and output an activation value. When you input an image in a ConvNet, each layer generates several activation functions that are passed on to the next layer.

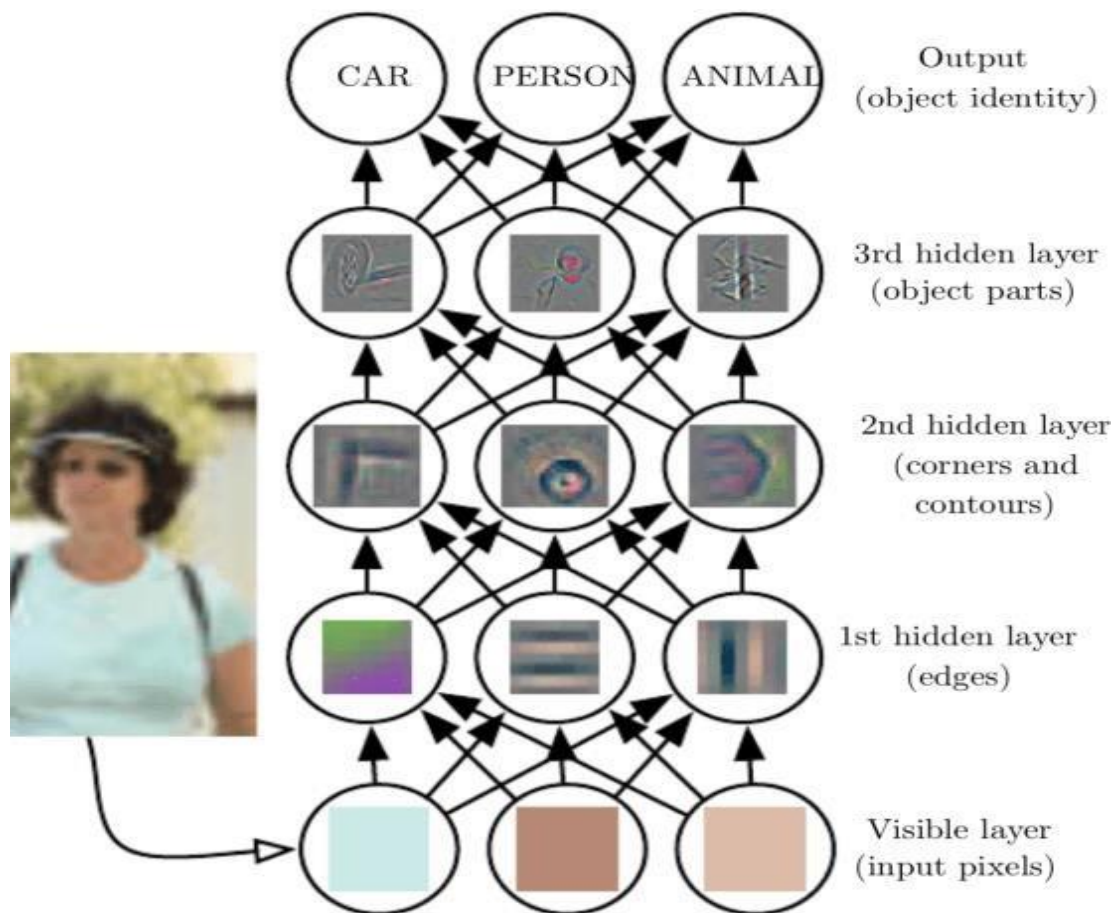
The first layer usually extracts basic features such as horizontal or diagonal edges. This output is passed on to the next layer which detects more complex features such as corners or combinational edges. As we move deeper into the network it can identify even more complex features such as objects, faces, etc.







Based on the activation map of the final convolution layer, the classification layer outputs a set of confidence scores (values between 0 and 1) that specify how likely the image is to belong to a “class.” For instance, if you have a ConvNet that detects cats, dogs, and horses, the output of the final layer is the possibility that the input image contains any of those animals.



### What's a pooling layer?

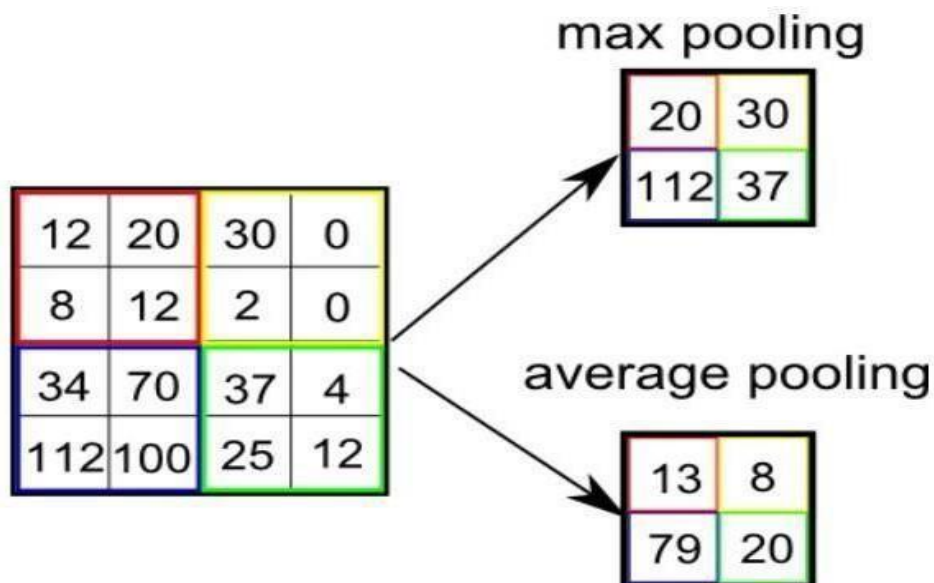
Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to **decrease the computational power required to process the data** by reducing the dimensions. There are two types of pooling average pooling and max pooling. I've only had experience with Max Pooling so far I haven't faced any difficulties.

3.0	3.0	3.0
3.0	3.0	3.0
3.0	2.0	3.0

3	3	2	1	0
0	0	1	3	1
3	1	2	2	3
2	0	0	2	2
2	0	0	0	1

So what we do in Max Pooling is we find the maximum value of a pixel from a portion of the image covered by the kernel. Max Pooling also performs as a **Noise Suppressant**. It discards the noisy activations altogether and also performs de-noising along with dimensionality reduction.

On the other hand, **Average Pooling** returns the **average of all the values** from the portion of the image covered by the Kernel. Average Pooling simply performs dimensionality reduction as a noise suppressing mechanism. Hence, we can say that **Max Pooling performs a lot better than Average Pooling**.

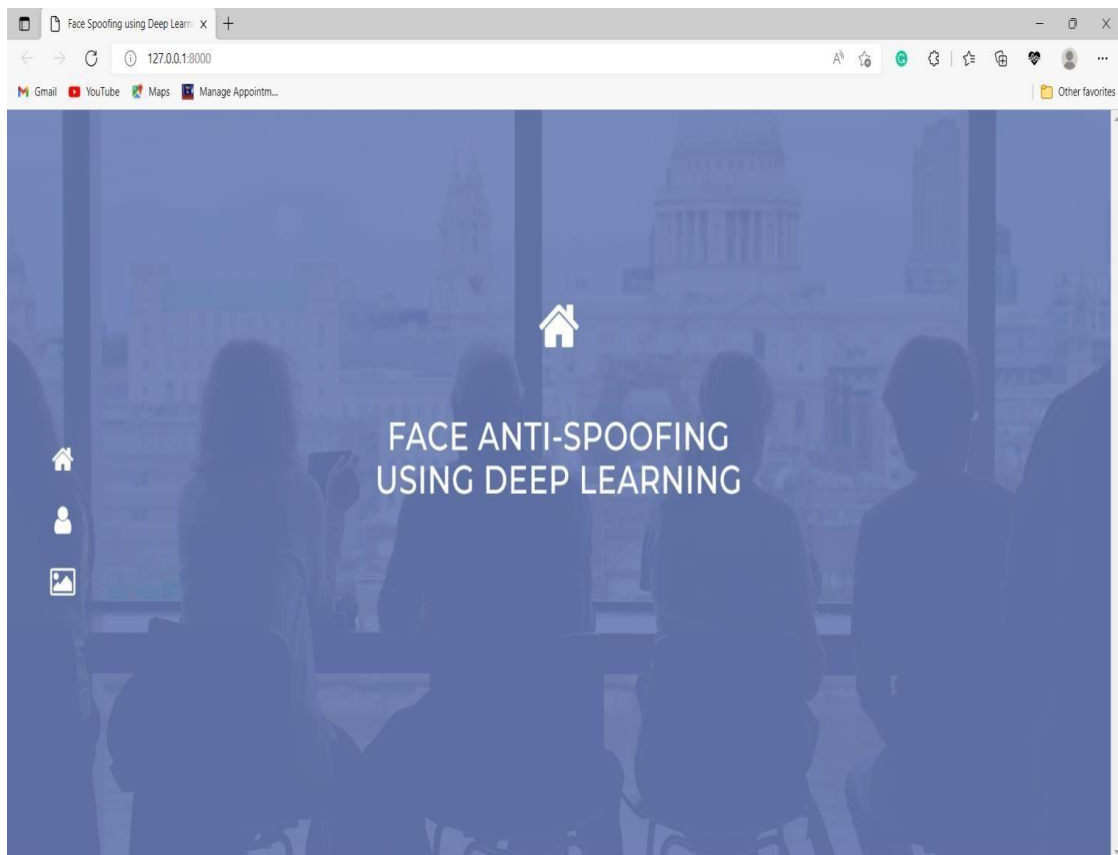


## CHAPTER 7

# RESULT AND PERFORMANCE ANALYSIS

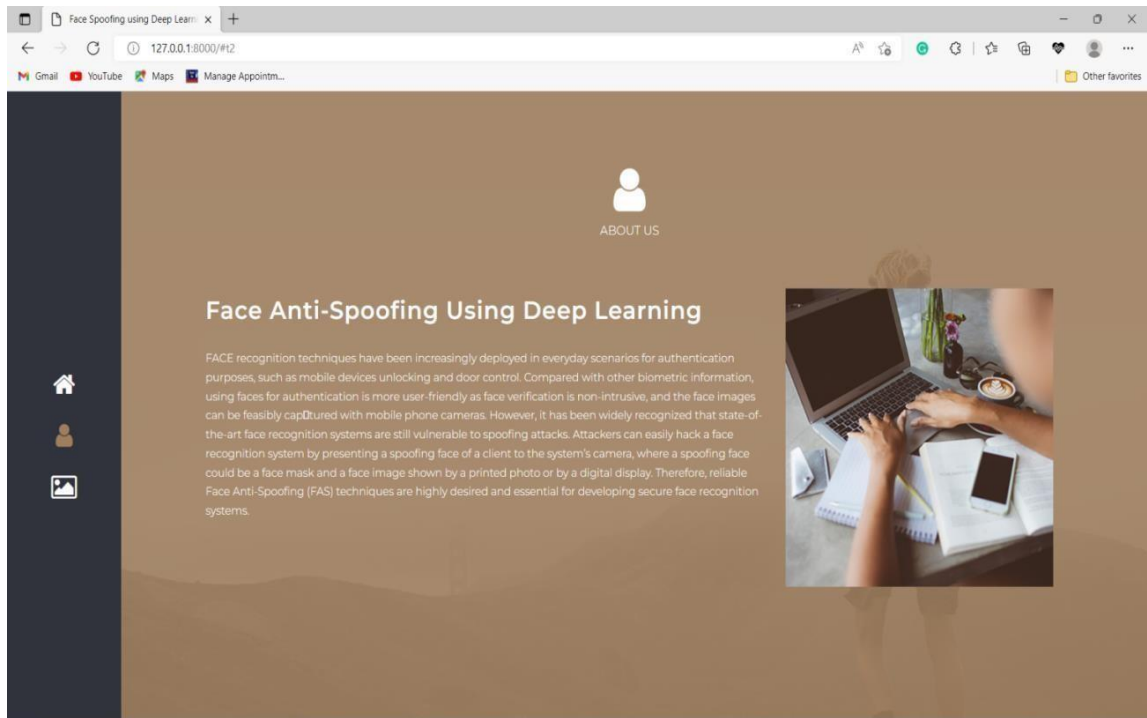
### Home page:

This is the home page of the project.



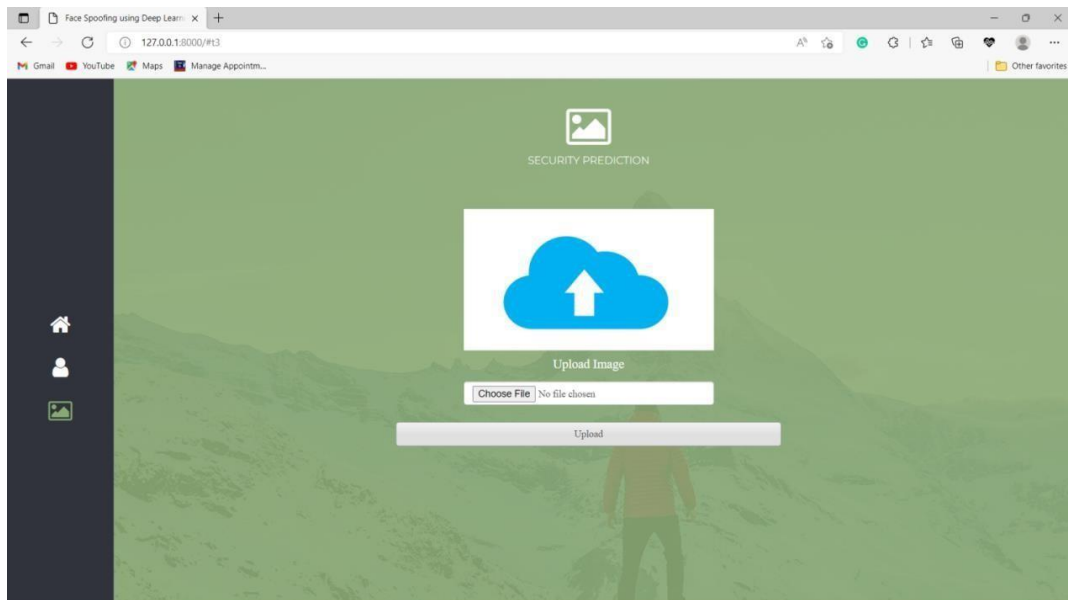
## Description Page:

This page describes the project briefly.



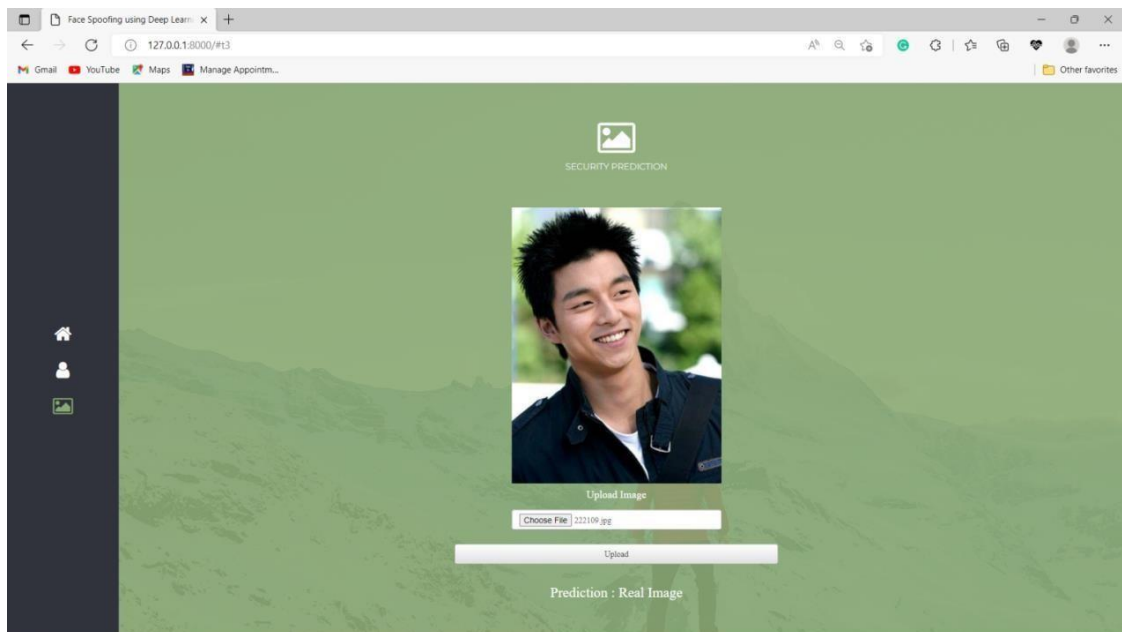
## Analysis Page:

This is the prediction page where u can run the algorithm.

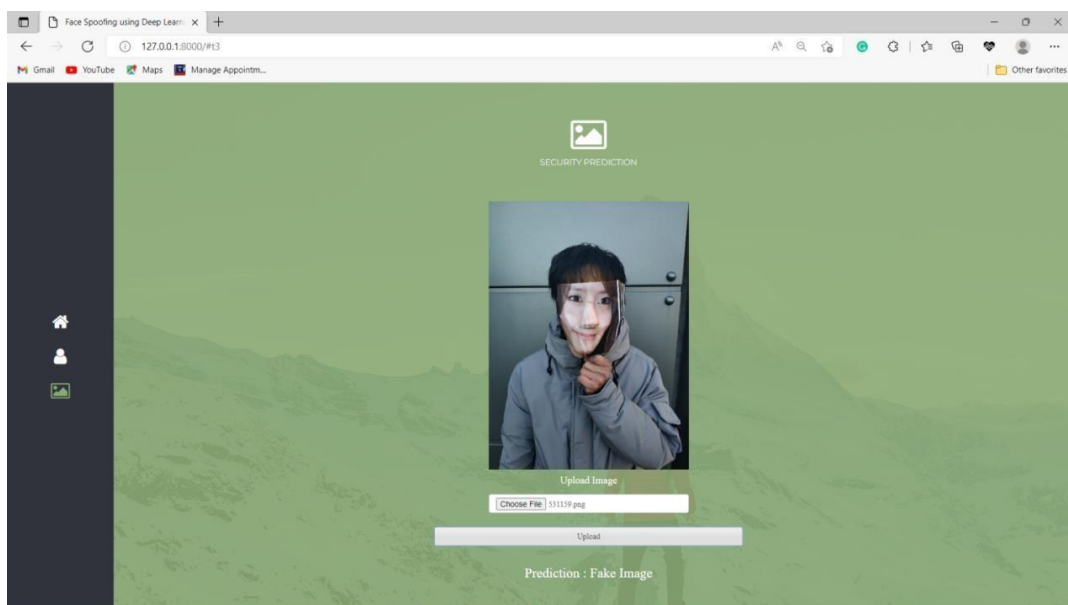


## Output pictures:

The page describes the output of real image.



The page describes the output of spoof image.



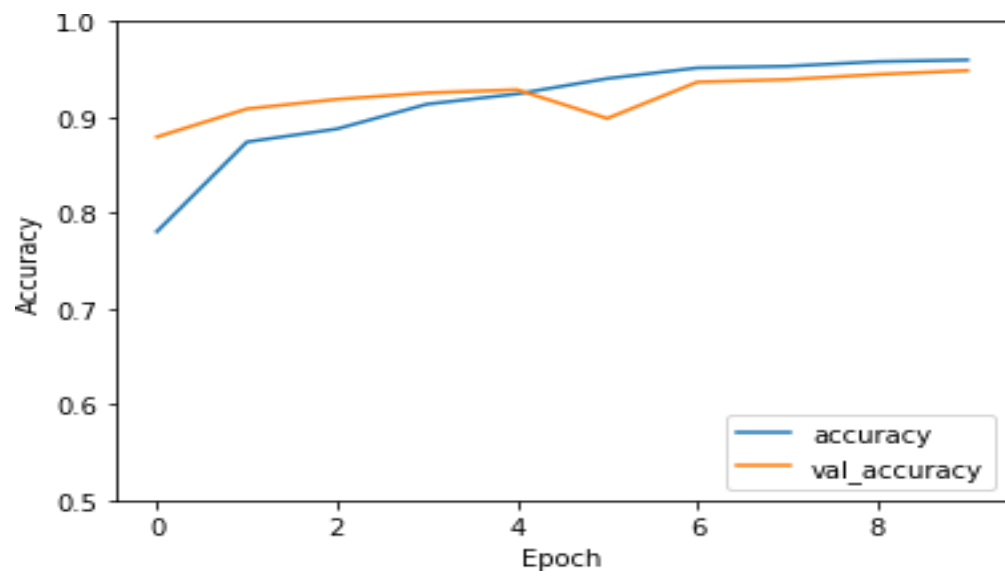


Fig 7.1: Accuracy Graph

## CHAPTER 8

### TESTING

#### 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 test. Each test type addresses a specific testing requirement.

Types Of Tests:

➤ **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.

➤ **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 satisfactory, 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.

➤ **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 centered 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.

## **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.

➤ **White Box Testing**

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

➤ **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.

## **CONCLUSION**

In this paper, we proposed a Convolution Neural Network framework for face anti-spoofing problem, which is inspired by the philosophy used by humans to determine whether a presented face example is genuine or not, namely, to look at the example globally first and then carefully observe the local regions to gain more discriminative information (CNN). An end-to-end learning approach, where the input to a CNN is a face image and the output is class probability has been proposed.

Specifically, we use deep learning to mimic the behaviour of discovering face-spoofing-related information from image sub-patches. Experimental results clearly demonstrate the effectiveness of the proposed CNN architecture with 95% percent accuracy.

## REFERENCES

- [1] R. Nosaka, Y. Ohkawa, and K. Fukui, "Feature extraction based on co-occurrence of adjacent local binary patterns," in *Advances in Image and Video Technology*, Y.-S. Ho, ed. Berlin, Germany: Springer, 2012, pp. 82–91.
- [2] I. Chingovska, A. Anjos, and S. Marcel, "On the effectiveness of local binary patterns in face anti-spoofing," in *Proc. Biometrics Special Interest Group*, 2012, pp. 1–7.
- [3] Z. Boulkenafet, J. Komulainen, and A. Hadid, "Face spoofing detection using colour texture analysis," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 8, pp. 1818–1830, Aug. 2016.
- [4] X. Tan, Y. Li, J. Liu, and L. Jiang, "Face liveness detection from a single image with sparse low rank bilinear discriminative model," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 504–517.
- [5] D. Gragnaniello, G. Poggi, C. Sansone, and L. Verdoliva, "An investigation of local descriptors for biometric spoofing detection," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 4, pp. 849–863, Apr. 2015.
- [6] J. Yang, Z. Lei, and S. Z. Li, "Learn convolutional neural network for face anti-spoofing," *Comput. Sci.*, vol. 9218, pp. 373–384, Aug. 2014.
- [7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
- [8] Y. Liu, A. Jourabloo, and X. Liu, "Learning deep models for face antispoofing: Binary or auxiliary supervision," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, Jun. 2018, pp. 389–398.