

ABSTRACT

The one important asset of our country is Bank currency and to create discrepancies of money miscreants introduce the fake notes which resembles to original note in the financial market. During demonetization time it is seen that so much of fake currency is floating in market. In general, by a human being, it is very difficult to identify forged note from the genuine not instead of various parameters designed for identification as many features of forged note are similar to original one. To discriminate between fake bank currency and original note is a challenging task. So, there must be an automated system that will be available in banks or in ATM machines. To design such an automated system there is need to design an efficient algorithm which is able to predict whether the Currency note is genuine or forged bank currency as fake notes are designed with high precision.

In this project, a new approach of Convolution Neural Network towards identification of fake currency notes through their images. This method is based on Deep Learning, which has seen tremendous success in image classification tasks in recent times. This technique can help both people and machine in identifying a fake currency note in real time through an image of the same currency notes.

TABLE OF CONTENTS

CHAPTER NO.	CONTENTS	PAGE NO.
	ABSTRACT	i
	LIST OF CONTENTS	ii
	LIST OF FIGURES	iv
CHAPTER 1	INTRODUCTION	1
	1.1 ABOUT DEEP LEARNING	1
	1.1.1 CNN IN DEEP LEARNING	2
	1.2 ABOUT THE PROJECT	2
	1.3 OBJECTIVES OF THE PROJECT	3
	1.4 EXISTING SYSTEM	3
	1.5 DRAWBACKS OF EXISTING SYSTEM	3
	1.6 PROPOSED SYSTEM	3
	1.7 ADVANTAGES OF PROPOSED SYSTEM	4
CHAPTER 2	LITERATURE SURVEY	5
	2.1 INTRODUCTION	5
CHAPTER 3	SYSTEM ANALYSIS	8
	3.1 INTRODUCTION	8
	3.2 SOFTWARE REQUIREMENTS	8
	3.3 HARDWARE REQUIREMENTS	8
	3.4 FEASIBILITY STUDY	8

CHAPTER 4	SYSTEM DESIGN	10
	4.1 INTRODUCTION	10
	4.2 MODULE DESCRIPTION	10
	4.3 SYSTEM ARCHITECTURE	12
	4.4 INPUT AND OUTPUT REPRESENTATION	12
	4.4.1 INPUT REPRESENTATION	12
	4.4.2 OUTPUT REPRESENTATION	14
	4.5 UML DIAGRAMS	14
CHAPTER 5	SOFTWARE ENVIRONMENT	19
	5.1 INTRODUCTION	19
	5.2 TECHNOLOGY AND DESCRIPTION	19
	5.3 MODULE IMPLEMENTATION	33
	5.4 SCREENSHOTS	53
CHAPTER 6	TESTING AND VALIDATION	57
	6.1 TESTING STRATEGIES	57
CHAPTER 7	RESULTS AND DISCUSSIONS	60
CHAPTER 8	CONCLUSION AND FUTURE ENHANCEMENTS	63
	8.1 CONCLUSION	63
	8.2 FUTURE ENHANCEMENTS	63
CHAPTER 9	BIBLIOGRAPHY	65

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
Figure 1.1	Schematic Diagram of Basic Convolutional Neural Network(CNN) Architecture	2
Figure 1.2	Extraction of Features of a Currency Note	4
Figure 4.1	Internal Structure of the System	13
Figure 4.2	Use Case Diagram for Fake Currency Detection	15
Figure 4.3	Class Diagram for Fake Currency Detection	16
Figure 4.4	Data Flow Diagram for Fake Currency Detection	16
Figure 4.5	Sequence Diagram for Fake Currency Detection	17
Figure 4.6	Activity Diagram for Fake Currency Detection	18
Figure 5.1	Python Official Portal	25
Figure 5.2	Selection of Version	26
Figure 5.3	Available Versions	26
Figure 5.4	Available Versions for Different Operating Systems	27
Figure 5.5	Opening the Downloaded Python	28
Figure 5.6	Installing Python	28
Figure 5.7	Python Setup	29

Figure 5.8	Execution of Commands	30
Figure 5.9	Checking the Downloaded Version	30
Figure 5.10	Selecting Python Idle	31
Figure 5.11	Selecting the File	31
Figure 5.12	User Window	53
Figure 5.13	Selecting the Dataset Folder	54
Figure 5.14	Uploading Dataset	55
Figure 5.15	Detection of Real or Fake Currency	56
Figure 7.1	Performance Analysis	60
Figure 7.2	Accuracy of CNN Model	61
Figure 7.3	Loss of CNN Model	61

CHAPTER-1

INTRODUCTION

Financial activities are carrying out in every second by people in the world in which counterfeit currency is a big threat. Fake currency notes are introduced into the market to create discrepancies in the financial market which resembles the original note. Basically they are illegally created to complete various tasks. In 1990 forgery issue is not much of concern but from late 19th century forgery has been increasing drastically. In 20th century, due to increase in technology, the frauds can generate fake currency which resembles genuine currency note and it is very difficult to discriminate them. This will lead to duplication of currency notes on a very large scale. As a human being it is very difficult to identify genuine currency and forged currency.

The original currency can be identified by the watermarks, typography, micro-lettering, etc. provided on the currency note. But fraudulent are creating fake currency with accurate features that make it very difficult to identify genuine note. To determine the legitimacy of the currency note artificial intelligence and Machine learning (ML) can play a vital role to design such a system that can identify forged currency note from the genuine bank currency.

1.1 ABOUT DEEP LEARNING

Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans: learn by example. Deep learning is a key technology behind driverless cars, enabling them to recognize a stop sign, or to distinguish a pedestrian from a lamppost. It's achieving results that were not possible before.

In deep learning, a computer model learns to perform classification tasks directly from images, text, or sound. Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance. Models are trained by using a large set of labeled data and neural network architectures that contain many layers.

Deep learning is a specialized form of machine learning. A machine learning workflow starts with relevant features being manually extracted from images. The features are then used to create a model that categorizes the objects in the image. With a deep learning workflow, relevant features are automatically extracted from images. In addition, deep learning performs

“end-to-end learning” – where a network is given raw data and a task to perform, such as classification, and it learns how to do this automatically.

1.1.1 CNN IN DEEP LEARNING

One of the most popular types of deep neural networks is known as convolutional neural networks (**CNN** or **ConvNet**). A CNN convolves learned features with input data, and uses 2D convolutional layers, making this architecture well suited to processing 2D data, such as images.

CNNs eliminate the need for manual feature extraction, so you do not need to identify features used to classify images. The CNN works by extracting features directly from images. The relevant features are not pretrained; they are learned while the network trains on a collection of images. This automated feature extraction makes deep learning models highly accurate for computer vision tasks such as object classification.

CNNs learn to detect different features of an image using tens or hundreds of hidden layers. Every hidden layer increases the complexity of the learned image features. For example, the first hidden layer could learn how to detect edges, and the last learns how to detect more complex shapes specifically catered to the shape of the object we are trying to recognize.

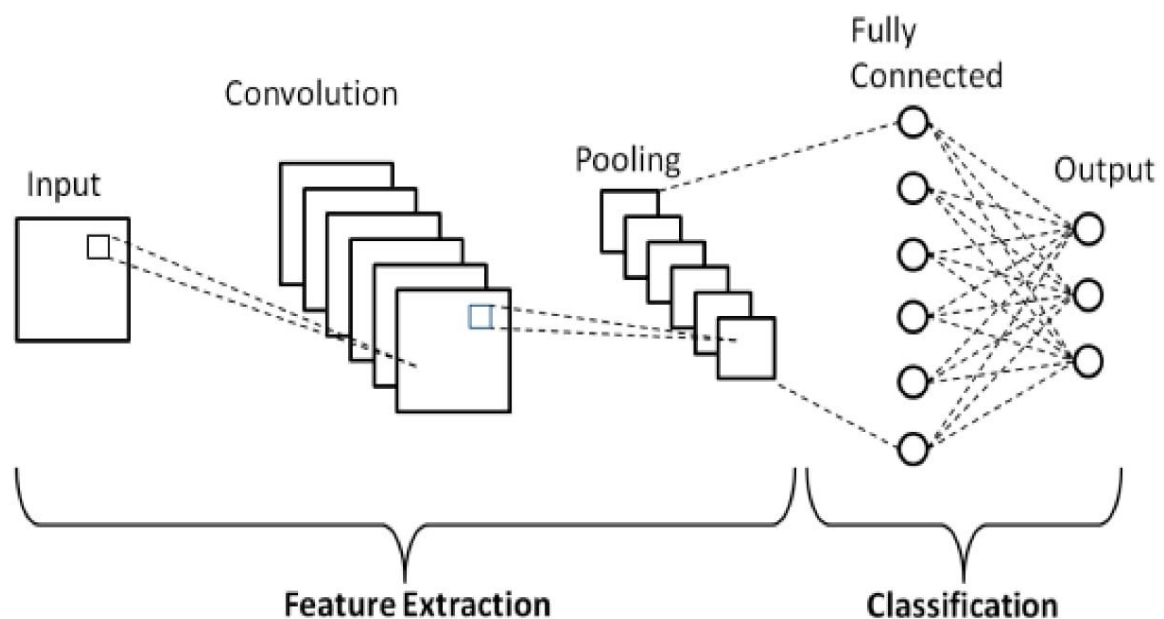


Figure 1.1 Schematic Diagram of Basic Convolutional Neural Network (CNN) Architecture.

1.2 ABOUT THE PROJECT

We are going to invent currency detection system over deep learning and CNN techniques which solves existing accuracy problem as well as reduce crime rates by genuine or fake notes. For future work, we can implement this technique on some more currency with rich dataset. Increasing the number of currency and dataset used for the process can improve the accuracy.

1.3 OBJECTIVES OF THE PROJECT

The main objective of the project is to identify the fake currency notes automatically using Morphological Algorithm. Although there were many methods in existence, this method was designed to overcome the drawbacks of the previous methods.

1.4 EXISTING SYSTEM

In previous project, machine learning approaches are used to classify whether currency note is original or not. Yeh et. al. implemented SVM based on multiple kernels to reduce false rate in detecting fake currency. To classify real and forged network, Author's Hassanpour et. al. used texture-based feature extraction method for the recognition of currency image and to model the texture, Markov chain concept is used. This method can able to recognize the currencies of many countries. To classify whether the currency note is forged or not, global optimization algorithms are applied in Artificial Neural Network (ANN) training phase; and they have observed good success in classification of currency note.

1.5 DRAWBACKS OF THE EXISTING SYSTEM

- 1) The process can only done for small datasets.
- 2) The performance of the system is not efficient which can cause inaccurate results.
- 3) The technology is increasing very vastly that will help the fradulents to generate fake notes whose resemblance is like genuine notes and it is very difficult to discriminate them.
- 4) Accuracy of success is low.

1.6 PROPOSED SYSTEM

The background of the project uses image processing technology and Convolutional Neural Network algorithm in deep learning which can be applied for the purpose of verifying valid currency notes. The software will detect the fake currency by extracting features of notes. The success rate of the software can be measured in terms of accuracy and speed. So our aim is to work on those parameters which will be impossible to implement on counterfeit notes. so we started working on parameters which will be enough to differentiate between fake and original notes.

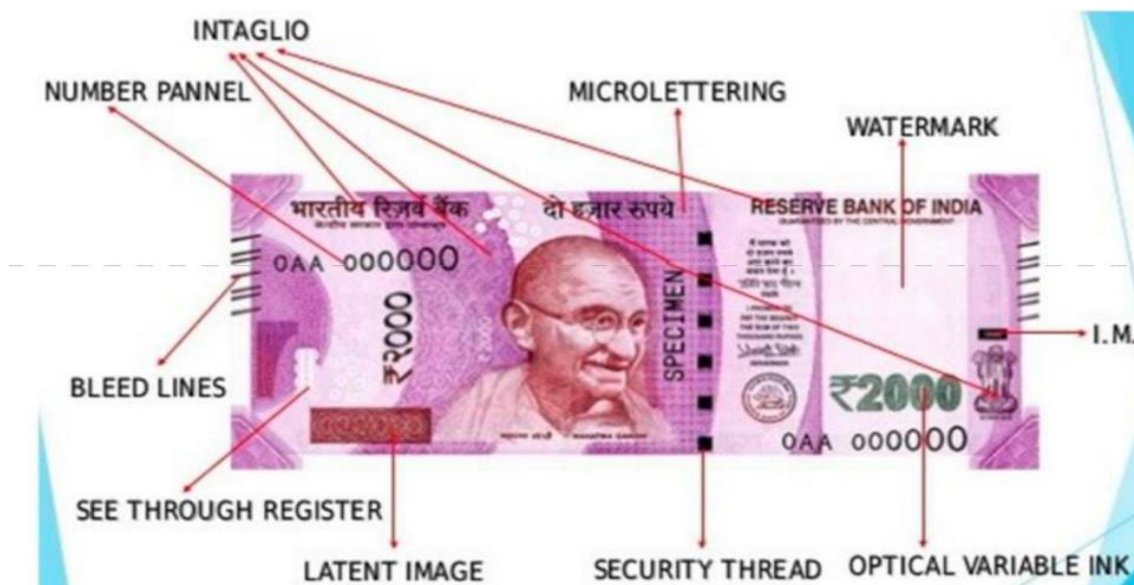


Figure 1.2 Extraction of features of a currency note.

Features of Currency Note:

- Watermark, Security thread, Latent image, Bleed lines, See through register, Micro lettering, Optical variable.
- Ink, I.M., Number panel, Intaglio.

Above features are extracted from the currency note by using computer vision. It is widely used to extract the features from an image. Those features are compared with the original collection of trained datasets. If the features of trained dataset and input image matched then it is a real note otherwise it is a fake note.

1.7 ADVANTAGES OF PROPOSED SYSTEM

- 1) Classification of fake and original notes is very easy.
- 2) On huge data sets, it functions admirably.

- 3) Detects characteristics without the need for human intervention.
- 4) Increase in the accuracy of determining if a note is genuine or counter.

CHAPTER -2

LITERATURE SURVEY

2.1 INTRODUCTION

Literature survey is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy and company strength. Once these things are satisfied, then the next step is to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system the above consideration are taken into account for developing the proposed system. The major part of the project development sector considers and fully survey all the required needs for developing the project. For every project Literature survey is the most important sector in software development process. Before developing the tools and the associated designing it is necessary to determine and survey the time factor, resource requirement, man power, economy, and company strength. Once these things are satisfied and fully surveyed, then the next step is to determine about the software specifications in the respective system such as what type of operating system the project would require, and what are all the necessary software are needed to proceed with the next step such as developing the tools, and the associated operations.

1. Tushar Agasti, Gajanan Burand, Pratik Wade and P Chitra, —Fake currency detection using image processing|| 14th ICSET-2017

Fake Currency has always been an issue which has created a lot of problems in the market. The increasing technological advancements have made the possibility for creating more counterfeit currency which are circulated in the market which reduces the overall economy of the country. There are machines present at banks and other commercial areas to check the authenticity of the currencies. But a common man does not have access to such systems and hence a need for software to detect fake currency arises, which can be used by common people. This proposed system uses Image Processing to detect whether the currency is genuine or counterfeit. The system is designed completely using Python programming

language. It consists of the steps such as gray scale conversion, edge detection, segmentation, etc. which are performed using suitable methods

2. Eshita Pilania, Bhavika Arora, —Recognition of Fake Currency Based on Security Thread Feature of Currency| International Journal Of Engineering And Computer Science, ISSN: 2319-7242

In the last few years a great technological advances in color printing, duplicating and scanning, counterfeiting problems have become more serious. In past only authorized printing house has the ability to make currency paper, but now a days it is possible for anyone to print fake bank note with the help of modern technology such as computer, laser printer. Fake notes are burning questions in almost every country. Counterfeit notes are a problem of almost every country but India has been hit really hard and has become a very acute problem. Fake Indian currency of 100, 500 and 1000 rupees seems to have flooded the whole system and there is no proper way to deal with them for a common person. There is a need to design a system that is helpful in recognition of paper currency notes with fast speed and in less time. Our system describes an approach for verification of Indian and other countries currency banknotes. The currency will be verified by using image processing techniques

3. Nayana Susan Jose, Shermin Siby, Juby Mathew, Mrudula Das, Android Based Currency Recognition System for Blind, International Journal of Engineering Research in Computer Science and Engineering (IJERCSE) Vol 2, Issue 4, April 2015.

In recent years, a lot of illegal counterfeiting rings manufacture and sell fake coins and at the same time fake note currency is printed as well which have caused great loss and damage to the society. Thus it is imperative to be able to detect fake currency, We propose a new approach to detect fake Indian notes using their images. Currency image is represented in the dissimilarity space, which is a vector space constructed by comparing the image with a set of prototypes. Each dimension measures the dissimilarity between the image under consideration and a prototype. In order to obtain the dissimilarity between two images, the local key points on each image are detected and described. Based on the characteristics of the currency, the matched key points between the two images can be identified in an efficient manner. A post processing procedure is further proposed to remove mismatched key points. Due to the limited number of fake currency in real life, SVM is conducted for fake currency detection, so only genuine currency are needed to train the classifier

4. Komal Vora, Ami Shah, Jay Mehta, A Review Paper on Currency Recognition System, International Journal of Computer Applications (0975 –8887) Volume 115 – No. 20, April 2015

In this paper, an algorithm based on the frequency domain feature extraction method is discussed for the detection of currency. This method efficiently utilizes the local spatial features in a currency image to recognize it. The entire system is pre-processed for the optimal and efficient implementation of two dimensional discrete wavelet transform (2D DWT) which is used to develop a currency recognition system. A set of coefficient statistical moments are then extracted from the approximate efficient matrix. The extracted features can be used for recognition, classification and retrieval of currency notes. The classification result will facilitate the recognition of fake currency mainly using serial number extraction by implementing OCR. It is found that the proposed method gives superior results.

CHAPTER -3

SYSTEM ANALYSIS

3.1 INTRODUCTION

The purpose of this work is detection of fake bank currency using machine learning algorithms. In detail, this document will provide a general description of our project, including user requirements, product perspective, and overview of requirements, general constraints. In addition, it will also provide the specific requirements and functionality needed for this project - such as interface, functional requirements and performance requirements.

3.2 SOFTWARE REQUIREMENTS

- **Operating System** : Windows
- **Coding Language** : Python 3.7

3.3 HARDWARE REQUIREMENTS

- **System** : Pentium IV 2.4 GHz
- **Hard Disk** : 40 GB
- **Floppy Drive** : 1.44 MB
- **Monitor** : 15 VGA Color
- **Mouse** : Logitech
- **Mouse** : Logitech

3.4 FEASIBILITY STUDY

The feasibility of the project is analyzed 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.

CHAPTER -4

SYSTEM DESIGN

4.1 INTRODUCTION

System design refers to the process of defining the architecture, modules, interfaces, data for a system to satisfy specified requirements. It is a multi-disciplinary field that involves trade-off analysis, balancing conflicting requirements, and making decisions about design choices that will impact the overall system.

4.2 MODULES DESCRIPTION

Modules Used in Project:-

Tensorflow

TensorFlow is free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

Numpy

Numpy is a general-purpose array-processing package. It provides a high performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object.
- Sophisticated (broadcasting) functions.
- Tools for integrating C/C++ and Fortran code.
- Useful linear algebra, Fourier transform, and random number capabilities.

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

Scikit – learn

Scikit-learn provide a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

4.3 SYSTEM ARCHITECTURE:

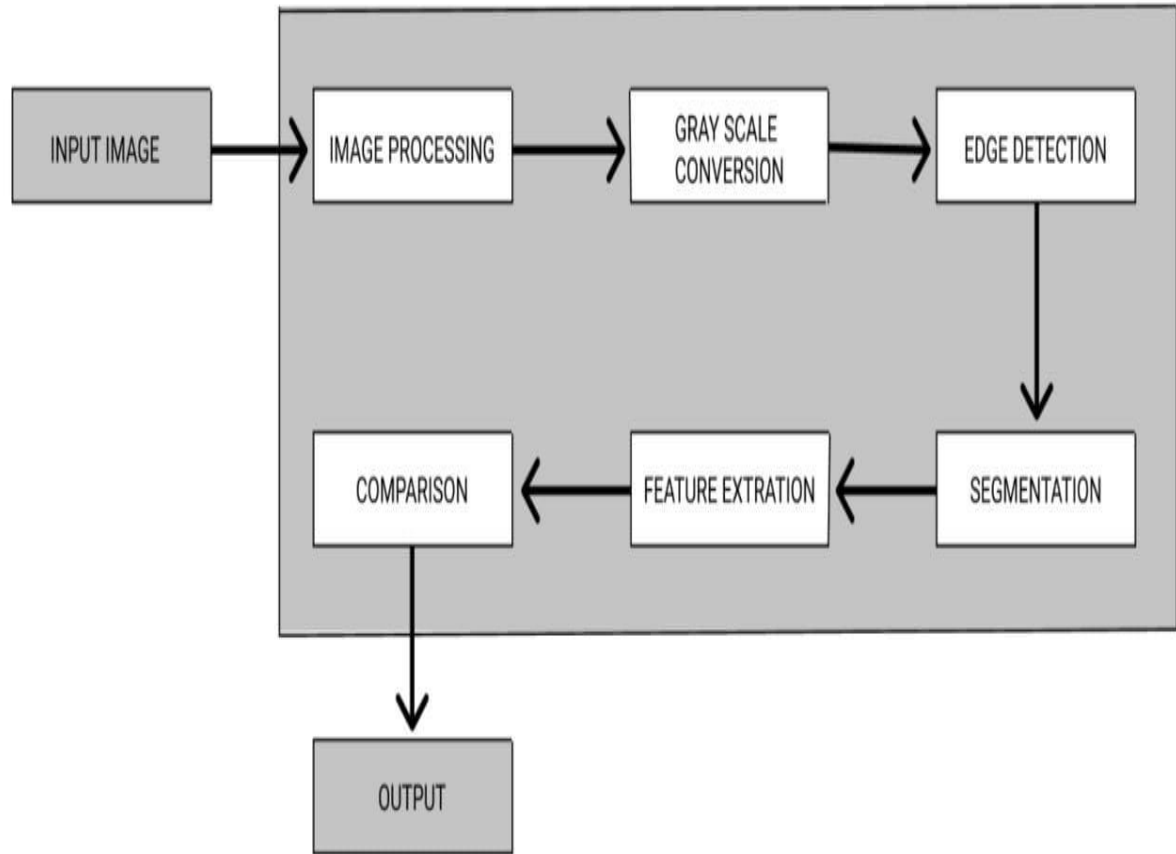


Figure 4.1 Internal Structure of the System.

4.4 INPUT AND OUTPUT REPRESENTATIONS

4.4.1 Input Representation

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

4.4.2 Output Representation

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making. The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

4.5 UML Diagram

The underlying premise of UML is that no one diagram can capture the different elements of a system in its entirety. Hence, UML is made up of nine diagrams that can be used to model a system at different points of time in the software life cycle of a system. The UML diagrams are:

Use Case Diagram:

The use case diagram is used to identify the primary elements and processes that form the system. The primary elements are termed as "actors" and the processes are called "use cases." The use case diagram shows which actors interact with each use case.

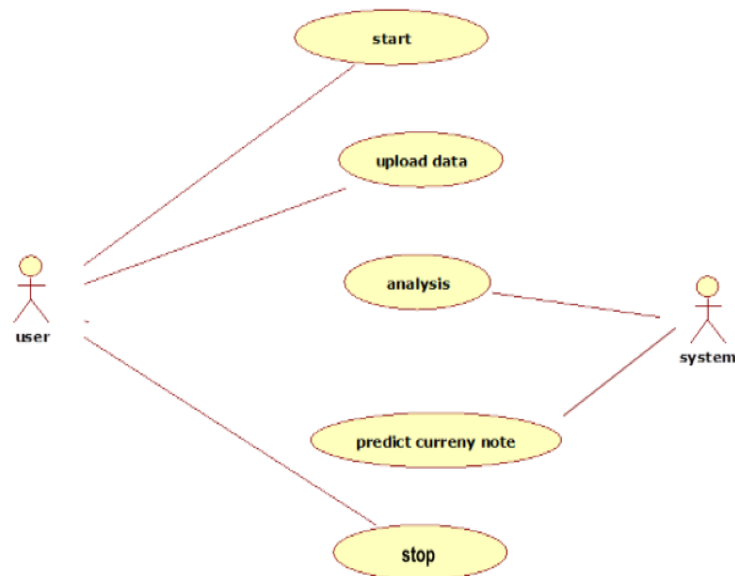


Figure 4.2 Use Case Diagram for Fake Currency Detection.

Class Diagram:

The class diagram is used to refine the use case diagram and define a detailed design of the system. The class diagram classifies the actors defined in the use case diagram into a set of interrelated classes. The relationship or association between the classes can be either an "is-a" or "has-a" relationship. Each class in the class diagram may be capable of providing certain functionalities. These functionalities provided by the class are termed "methods" of the class. Apart from this, each class may have certain "attributes" that uniquely identify the class.

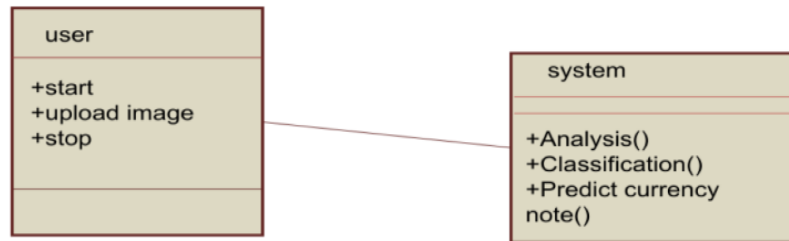


Figure 4.3 Class Diagram for Fake Currency Detection.

Data Flow Diagram:

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled.

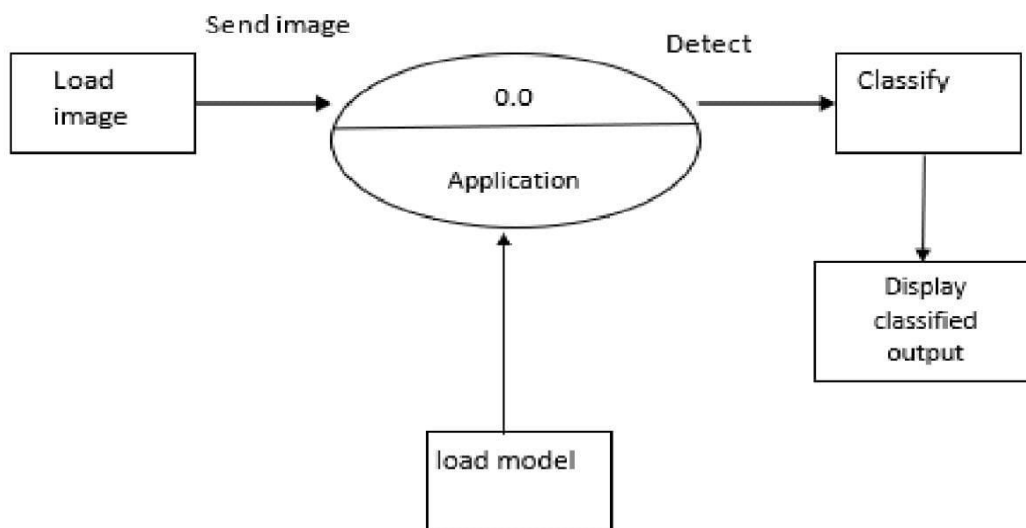


Figure 4.4 Data Flow Diagram for Fake Currency Detection.

Sequence Diagram:

A sequence diagram represents the interaction between different objects in the system. The important aspect of a sequence diagram is that it is time-ordered. This means that the exact sequence of the interactions between the objects is represented step by step. Different objects in the sequence diagram interact with each other by passing "messages".

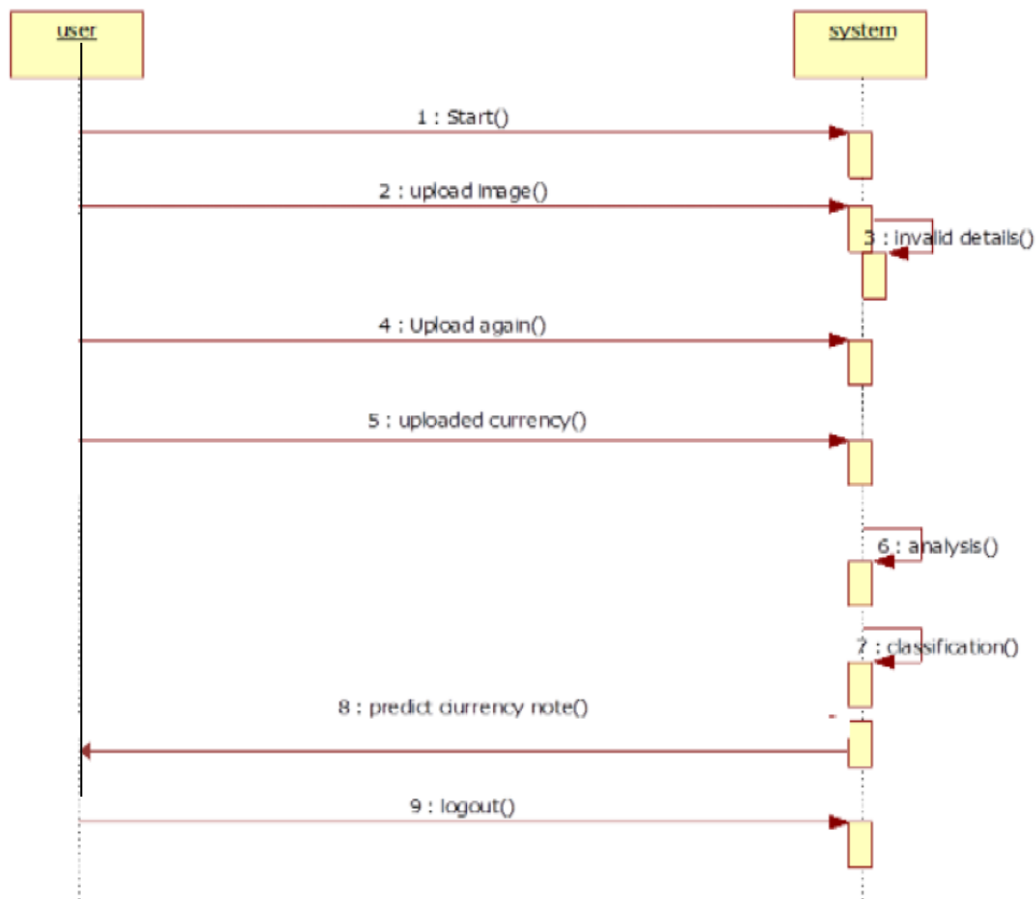


Figure 4.5 Sequence Diagram for Fake Currency Detection.

Activity Diagram:

We use Activity Diagrams to illustrate the flow of control in a system and refer to the steps involved in the execution of a use case. We model sequential and concurrent activities using activity diagrams. So, we basically depict workflows visually using an activity diagram. An activity diagram focuses on condition of flow and the sequence in which it happens.

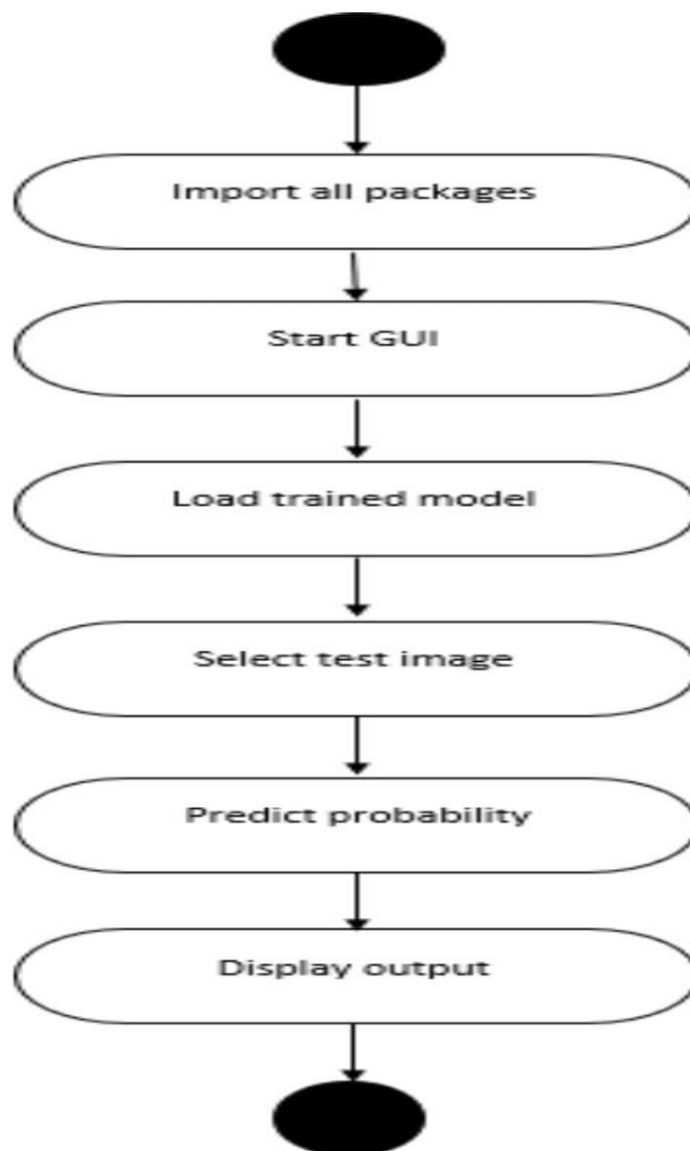


Figure 4.6 Activity Diagram for Fake Currency Detection.

CHAPTER -5

SOFTWARE ENVIRONMENT

5.1 INTRODUCTION

Here in the implementation part we implement the code by the defined modules that we are already discussed. We mention the module implementation and their screenshots.

5.2 TECHNOLOGY DESCRIPTION

Python is currently the most widely used multi-purpose, high-level programming language. Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java. Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time. Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber... etc. The biggest strength of Python is huge collection of standard library which can be used for the following:

- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like Opencv, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia

Advantages of Python:-

Let's see how Python dominates over other languages.

1. Extensive Libraries

Python downloads with an extensive library and it code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more. So, we don't have to write the complete code for that manually.

2. Extensible

As we have seen earlier, Python can be extended to other languages. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

3. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add scripting capabilities to our code in the other language.

4. Improved Productivity

The language's simplicity and extensive libraries render programmers more productive than languages like Java and C++ do.

5. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet Of Things. This is a way to connect the language with the real world.

6. Simple and Easy

When working with Java, you may have to create a class to print 'Hello easy to learn, understand, and code. This is why when World'. But in Python, just a print statement will do. It is also quite people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

7. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and indentation is mandatory. This further aids the readability of the code.

8. Object-Oriented

This language supports both the procedural and object-oriented programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the encapsulation of data and functions into one.

9. Free and Open-Source

Like we said earlier, Python is freely available. But not only can you download Python for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

10. Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn't the same with Python. Here, you need to code only once, and you can run it anywhere. This is called Write Once Run Anywhere (WORA). However, you need to be careful enough not to include any system-dependent features.

11. Interpreted

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, debugging **is easier** than in compiled languages. Any doubts till now in the advantages of Python? Mention in the comment section.

Advantages of Python Over Other Languages:

1. Less Coding

Almost all of the tasks done in Python require less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don't have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

2. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.

3. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and machine learning, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

Disadvantages of Python

So far, we've seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let's now see the downsides of choosing Python over another language.

1. Speed Limitations

We have seen that Python code is executed line by line. But since Python is interpreted, it often results in slow execution. This, however, isn't a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the client-side. Besides that, it is rarely ever used to implement Smartphone-based applications. One such application is called Carbonnelle.

The reason it is not so famous despite the existence of Brython is that it isn't that secure.

3. Design Restrictions

As you know, Python is dynamically-typed. This means that you don't need to declare the type of variable while writing the code. It uses duck-typing. But wait, what's that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can raise run-time errors.

4. Underdeveloped Database Access Layers

Compared to more widely used technologies like JDBC (Java DataBase Connectivity) and ODBC (Open DataBase Connectivity), Python's database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

5. Simple

No, we're not kidding. Python's simplicity can indeed be a problem. Take my example. I don't do Java, I'm more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

History of Python : -

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde & Informatica). The greatest achievement of ABC was to influence the design of Python. Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners¹, Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it." Later on in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems. So I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

Pandas

Panda is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data mugging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

Install Python Step-by-Step in Windows and Mac:

Python a versatile programming language doesn't come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

How to Install Python on Windows and Mac :

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your System Requirements. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a Windows 64-bit operating system. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. The steps on how to install Python on Windows 10, 8 and 7 are divided into 4 parts to help understand better.

Download the Correct version into the system

Step 1: Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: <https://www.python.org>



Figure 5.1 Python Official Portal.

Now, check for the latest and the correct version for your operating system.

Step 2: Click on the Download Tab.



Figure 5.2 Selection of Version.

Step 3: You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4.








Looking for a specific release?			
Python releases by version number:			
Release version	Release date	Click for more	
Python 3.7.4	July 8, 2019	 Download	Release Notes
Python 3.6.9	July 2, 2019	 Download	Release Notes
Python 3.7.3	March 25, 2019	 Download	Release Notes
Python 3.4.10	March 18, 2019	 Download	Release Notes
Python 3.5.7	March 18, 2019	 Download	Release Notes
Python 2.7.16	March 4, 2019	 Download	Release Notes
Python 3.7.2	Dec. 24, 2018	 Download	Release Notes

Figure 5.3 Available Versions.

Step 4: Scroll down the page until you find the Files option.

Step 5: Here you see a different version of python along with the operating system.

Files					
Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		68111671e5b2db4ae77b9ab01b709be	13017563	SG
XZ compressed source tarball	Source release		d33e4aae6097051c2eca45ee3604803	17133432	SG
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	6428b4fa7583daf11a42c8a1ce08e6	34898436	SG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	5dd605c38217a45773bf5e4a936b241f	20082845	SG
Windows help file	Windows		063999573a2c96b2ac56cadefb47cd2	8131761	SG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	9809c3cfd29c3bdfab683184a4072fa2	7504391	SG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	a702b4b0ad76d4bcb3543a383e563400	26680368	SG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	28cb1c608b6d73ae8e53a3bf051b4bd2	1362904	SG
Windows x86 embeddable zip file	Windows		9fab3b819841879fda94133574139d8	6741626	SG
Windows x86 executable installer	Windows		33cc802942a54446a3d8e451476394789	25663848	SG
Windows x86 web-based installer	Windows		1b670cfafcd317df82c30983ea371d87c	1324608	SG

Figure 5.4 Available Versions for Different Operating Systems.

- To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.
- To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation.

Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

Installation of Python:

Step 1: Go to Download and Open the downloaded python version to carry out the installation process.

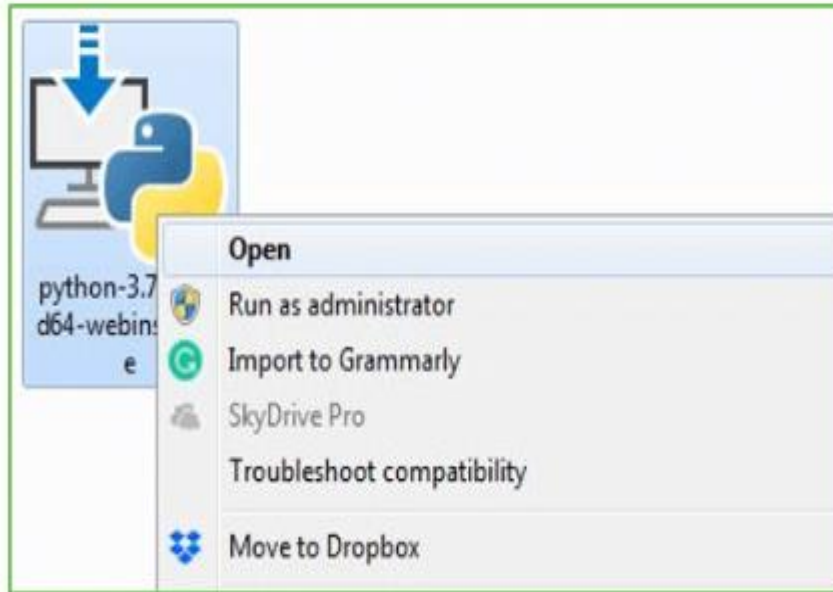


Figure 5.5 Opening the Downloaded Python.

Step 2: Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.



Figure 5.6 Installing Python.

Step 3: Click on Install NOW After the installation is successful. Click on Close.



Figure 5.7 Python Setup.

With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

Note: The installation process might take a couple of minutes.

Verify the Python Installation

Step 1: Click on Start.

Step 2: In the Windows Run Command, type “cmd”.

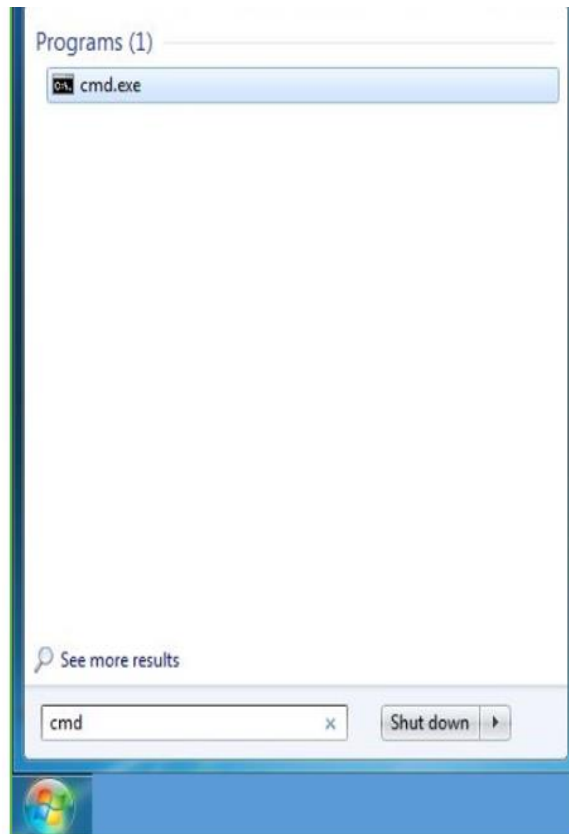


Figure 5.8 Execution of Commands.

Step 3: Open the Command prompt option.

Step 4: Let us test whether the python is correctly installed. Type **python -V** and press Enter.

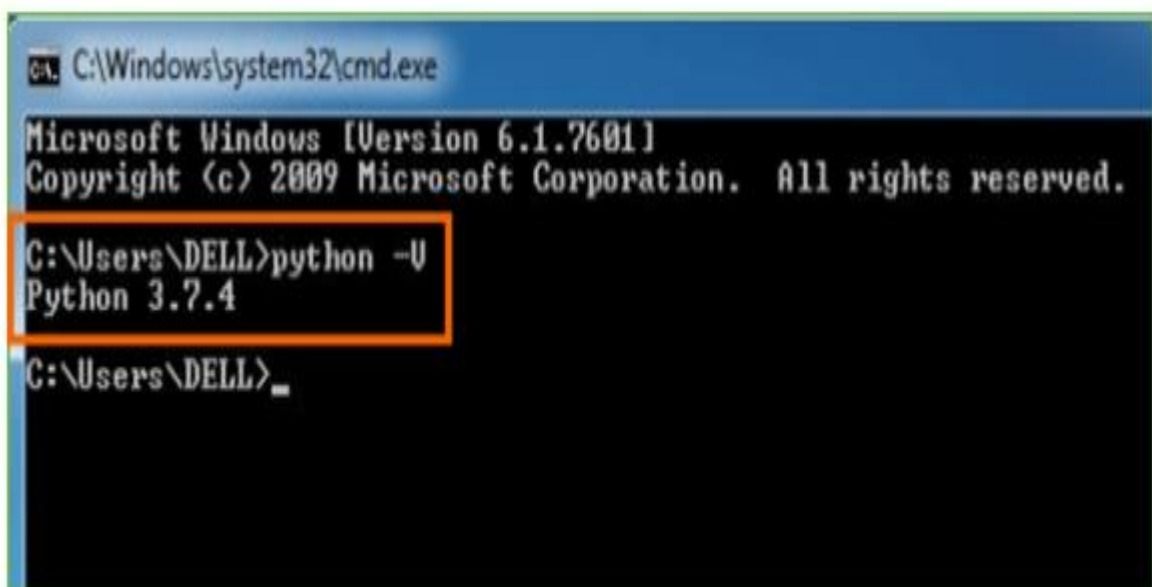


Figure 5.9 Checking the Downloaded Version.

Step 5: You will get the answer as 3.7.4

Note: If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

Check how the Python IDLE works:

Step 1: Click on Start.

Step 2: In the Windows Run command, type “python idle”.

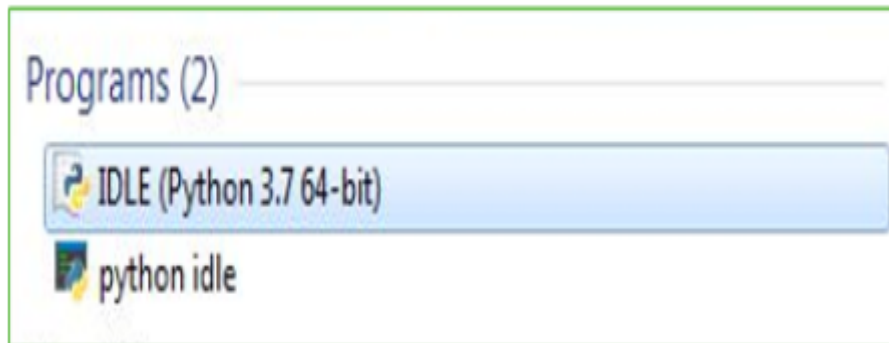


Figure 5.10 Selecting Python Idle.

Step 3: Click on IDLE (Python 3.7 64-bit) and launch the program.

Step 4: To go ahead with working in IDLE you must first save the file. Click on File > Click on Save

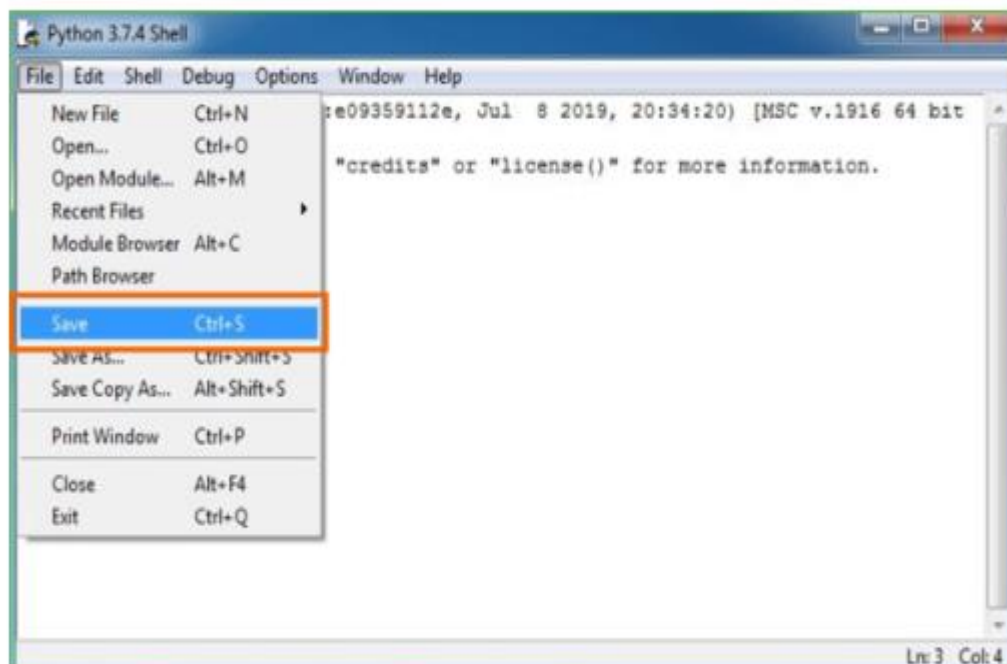


Figure 5.11 Selecting the File.

Step 5: Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

Step 6: Now for e.g. **enter print.**

5.3 MODULE IMPLEMENTATION

```
from tkinter import *

import tkinter

from tkinter import filedialog

import numpy as np

from tkinter.filedialog import askdirectory

from tkinter import simpledialog

import cv2

from keras.utils.np_utils import to_categorical

from keras.layers import Input

from keras.models import Model

from keras.layers import MaxPooling2D

from keras.layers import Dense, Dropout, Activation, Flatten

from keras.layers import Convolution2D

from keras.models import Sequential

import keras

import pickle

import matplotlib.pyplot as plt

import os

from keras.models import model_from_json

main = tkinter.Tk()#fake currency detection using image processing

main.title("fake currency detection using image processing")

#designing main screen
```

```

main.geometry("1000x700")

global filename

global classifier

def upload():

    global filename

    filename = filedialog.askdirectory(initialdir = ".")

    text.delete('1.0', END)

    text.insert(END,filename+' Loaded')

    text.insert(END,"Dataset Loaded")

def processImages():

    text.delete('1.0', END)

    X_train = np.load('model/features.txt.npy')

    Y_train = np.load('model/labels.txt.npy')

    text.insert(END,'Total images found in dataset for training  =' +str(X_train.shape[0])+"\n\n")

def generateModel():

    global classifier

    text.delete('1.0', END)

    if os.path.exists('model/model.json'):

        from tkinter import *

import tkinter

from tkinter import filedialog

import numpy as np

from tkinter.filedialog import askdirectory

```

```

from tkinter import simpledialog

import cv2

from keras.utils.np_utils import to_categorical

from keras.layers import Input

from keras.models import Model

from keras.layers import MaxPooling2D

from keras.layers import Dense, Dropout, Activation, Flatten

from keras.layers import Convolution2D

from keras.models import Sequential

import keras

import pickle

import matplotlib.pyplot as plt

import os

from keras.models import model_from_json

main = tkinter.Tk()#fake currency detection using image processing

main.title("fake currency detection using image processing") #designing main screen

main.geometry("1000x700")

global filename

global classifier

def upload():

    global filename

    filename = filedialog.askdirectory(initialdir = ".")

    text.delete('1.0', END)

```



```

text.insert(END,filename+' Loaded')

text.insert(END,"Dataset Loaded")

def processImages():

    text.delete('1.0', END)

    X_train = np.load('model/features.txt.npy')

    Y_train = np.load('model/labels.txt.npy')

    text.insert(END,'Total images found in dataset for training = '+str(X_train.shape[0])+"\n\n")

def generateModel():

    global classifier

    text.delete('1.0', END)

    if os.path.exists('model/model.json'):

        with open('model/model.json', "r") as json_file:

            loaded_model_json = json_file.read()

            classifier = model_from_json(loaded_model_json)

            classifier.load_weights("model/model_weights.h5")

            classifier._make_predict_function()

            print(classifier.summary())

            f = open('model/history.pkl', 'rb')

            data = pickle.load(f)

            f.close()

            acc = data['accuracy']

            accuracy = acc[9] * 100

            text.insert(END,"CNN Training Model Accuracy = "+str(accuracy)+"\n")

```

else:

```
    classifier = Sequential()

    classifier.add(Convolution2D(32, 3, 3, input_shape = (64, 64, 1), activation = 'relu'))

    classifier.add(MaxPooling2D(pool_size = (2, 2)))

    classifier.add(Convolution2D(32, 3, 3, activation = 'relu'))

    classifier.add(MaxPooling2D(pool_size = (2, 2)))

    classifier.add(Flatten())

    classifier.add(Dense(output_dim = 256, activation = 'relu'))

    classifier.add(Dense(output_dim = 1, activation = 'softmax'))

    print(classifier.summary())

    classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics =
['accuracy'])

    hist = classifier.fit(X_train, Y_train, batch_size=16, epochs=10, shuffle=True, verbose=2)

    classifier.save_weights('model/model_weights.h5')

    model_json = classifier.to_json()

    with open("model/model.json", "w") as json_file:

        json_file.write(model_json)

    f = open('model/history.pkl', 'wb')

    pickle.dump(hist.history, f)

    f.close()

f = open('model/history.pkl', 'rb')

data = pickle.load(f)

f.close()
```

```

acc = data['accuracy']

accuracy = acc[9] * 100

text.insert(END,"CNN Training Model Accuracy = "+str(accuracy)+"\n")

def predict():

    name = filedialog.askopenfilename(initialdir="testImages")

    img = cv2.imread(name)

    img = cv2.resize(img, (64,64))

    im2arr = np.array(img)

    im2arr = im2arr.reshape(1,64,64,3)

    XX = np.asarray(im2arr)

    XX = XX.astype('float32')

    XX = XX/255

    preds = classifier.predict(XX)

    print(str(preds)+" "+str(np.argmax(preds)))

    predict = np.argmax(preds)

    print(predict)

    img = cv2.imread(name)

    img = cv2.resize(img,(450,450))

    msg = "Real"

    if predict == 0:

        cv2.putText(img, 'Fake', (10, 25), cv2.FONT_HERSHEY_SIMPLEX,0.6, (0, 255, 255), 2)

        msg = 'Fake'

    else:from tkinter import *
```

```
import tkinter

from tkinter import filedialog

import numpy as np

from tkinter.filedialog import askdirectory

from tkinter import simpledialog

import cv2

from keras.utils.np_utils import to_categorical

from keras.layers import Input

from keras.models import Model

from keras.layers import MaxPooling2D

from keras.layers import Dense, Dropout, Activation, Flatten

from keras.layers import Convolution2D

from keras.models import Sequential

import keras

import pickle

import matplotlib.pyplot as plt

import os

from keras.models import model_from_json


main = tkinter.Tk()#fake currency detection using image processing

main.title("fake currency detection using image processing") #designing main screen

main.geometry("1000x700")

global filename
```

```

global classifier

def upload():

    global filename

    filename = filedialog.askdirectory(initialdir = ".")

    text.delete('1.0', END)

    text.insert(END,filename+' Loaded')

    text.insert(END,"Dataset Loaded")

def processImages():

    text.delete('1.0', END)

    X_train = np.load('model/features.txt.npy')

    Y_train = np.load('model/labels.txt.npy')

    text.insert(END,'Total images found in dataset for training = '+str(X_train.shape[0])+"\n\n")

def generateModel():

    global classifier

    text.delete('1.0', END)

    if os.path.exists('model/model.json'):

        with open('model/model.json', "r") as json_file:

            loaded_model_json = json_file.read()

            classifier = model_from_json(loaded_model_json)

            classifier.load_weights("model/model_weights.h5")

            classifier._make_predict_function()

            print(classifier.summary())

            f = open('model/history.pkl', 'rb')

```

```

data = pickle.load(f)

f.close()

acc = data['accuracy']

accuracy = acc[9] * 100

text.insert(END, "CNN Training Model Accuracy = "+str(accuracy)+"\n")

else:

    classifier = Sequential()

    classifier.add(Convolution2D(32, 3, 3, input_shape = (64, 64, 1), activation = 'relu'))

    classifier.add(MaxPooling2D(pool_size = (2, 2)))

    classifier.add(Convolution2D(32, 3, 3, activation = 'relu'))

    classifier.add(MaxPooling2D(pool_size = (2, 2)))

    classifier.add(Flatten())

    classifier.add(Dense(output_dim = 256, activation = 'relu'))

    classifier.add(Dense(output_dim = 1, activation = 'softmax'))

    print(classifier.summary())

    classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics =
['accuracy'])

    hist = classifier.fit(X_train, Y_train, batch_size=16, epochs=10, shuffle=True, verbose=2)

    classifier.save_weights('model/model_weights.h5')

    model_json = classifier.to_json()

    with open("model/model.json", "w") as json_file:

        json_file.write(model_json)

    f = open('model/history.pkl', 'wb')

```

```

pickle.dump(hist.history, f)

f.close()

f = open('model/history.pckl', 'rb')

data = pickle.load(f)

f.close()

acc = data['accuracy']

accuracy = acc[9] * 100

text.insert(END, "CNN Training Model Accuracy = "+str(accuracy)+"\n")

def predict():

    name = filedialog.askopenfilename(initialdir="testImages")

    img = cv2.imread(name)

    img = cv2.resize(img, (64,64))

    im2arr = np.array(img)

    im2arr = im2arr.reshape(1,64,64,3)

    XX = np.asarray(im2arr)

    XX = XX.astype('float32')

    XX = XX/255

    preds = classifier.predict(XX)

    print(str(preds)+" "+str(np.argmax(preds)))

    predict = np.argmax(preds)

    print(predict)

    img = cv2.imread(name)

    img = cv2.resize(img,(450,450))

```

```

msg = "

if predict == 0:

    cv2.putText(img, 'Fake', (10, 25), cv2.FONT_HERSHEY_SIMPLEX,0.6, (0, 255, 255), 2)

    msg = 'Fake'

else:

    cv2.putText(img, 'Real', (10, 25), cv2.FONT_HERSHEY_SIMPLEX,0.6, (0, 255, 255), 2)

    msg = 'Real'

cv2.imshow(msg,img)

cv2.waitKey(0)

def graph():

    f = open('model/history.pckl', 'rb')

    data = pickle.load(f)

    f.close()

    accuracy = data['accuracy']

    loss = data['loss']

    plt.figure(figsize=(10,6))

    plt.grid(True)

    plt.xlabel('Iterations')

    plt.ylabel('Accuracy/Loss')

    plt.plot(loss, 'ro-', color = 'red')

    plt.plot(accuracy, 'ro-', color = 'green')

    plt.legend(['Loss', 'Accuracy'], loc='upper left')

    plt.title('CNN Accuracy & Loss')

```



```

plt.show()

font = ('times', 16, 'bold')

title = Label(main, text='detection of fake currency ', justify=LEFT)

title.config(bg='deep skyblue', fg='white')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=100,y=5)

title.pack()

font1 = ('times', 13, 'bold')

uploadButton = Button(main, text="Upload Dataset", command=upload)

uploadButton.place(x=10,y=100)

uploadButton.config(font=font1)

processButton = Button(main, text="Image Preprocessing", command=processImages)

processButton.place(x=280,y=100)

processButton.config(font=font1)

cnnButton = Button(main, text="Generate CNN Model", command=generateModel)

cnnButton.place(x=10,y=150)

cnnButton.config(font=font1)

predictButton = Button(main, text="Upload Test Image", command=predict)

predictButton.place(x=280,y=150)

predictButton.config(font=font1)

graphButton = Button(main, text="Accuracy & Loss Graph", command=graph)

graphButton.place(x=10,y=200)

```

```

graphButton.config(font=font1)

font1 = ('times', 12, 'bold')

text=Text(main,height=20,width=120)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=10,y=250)

text.config(font=font1)

main.config(bg='LightSteelBlue3')

main.mainloop()

cv2.putText(img, 'Real', (10, 25), cv2.FONT_HERSHEY_SIMPLEX,0.6, (0, 255, 255), 2)

msg = 'Real'

cv2.imshow(msg,img)

cv2.waitKey(0)

def graph():

    f = open('model/history.pckl', 'rb')

    data = pickle.load(f)

    f.close()

    accuracy = data['accuracy']

    loss = data['loss']

    plt.figure(figsize=(10,6))

    plt.grid(True)

    plt.xlabel('Iterations')

    plt.ylabel('Accuracy/Loss')

```

```

plt.plot(loss, 'ro-', color = 'red')

plt.plot(accuracy, 'ro-', color = 'green')

plt.legend(['Loss', 'Accuracy'], loc='upper left')

plt.title('CNN Accuracy & Loss')

plt.show()

font = ('times', 16, 'bold')

title = Label(main, text='detection of fake currency ', justify=LEFT)

title.config(bg='deep skyblue', fg='white')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=100,y=5)

title.pack()

font1 = ('times', 13, 'bold')

uploadButton = Button(main, text="Upload Dataset", command=upload)

uploadButton.place(x=10,y=100)

uploadButton.config(font=font1)

processButton = Button(main, text="Image Preprocessing", command=processImages)

processButton.place(x=280,y=100)

processButton.config(font=font1)

cnnButton = Button(main, text="Generate CNN Model", command=generateModel)

cnnButton.place(x=10,y=150)

cnnButton.config(font=font1)

predictButton = Button(main, text="Upload Test Image", command=predict)

```

```

predictButton.place(x=280,y=150)

predictButton.config(font=font1)


graphButton = Button(main, text="Accuracy & Loss Graph", command=graph)

graphButton.place(x=10,y=200)

graphButton.config(font=font1)

font1 = ('times', 12, 'bold')

text=Text(main,height=20,width=120)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=10,y=250)

text.config(font=font1)

main.config(bg='LightSteelBlue3')

main.mainloop()with open('model/model.json', "r") as json_file:

    loaded_model_json = json_file.read()

    classifier = model_from_json(loaded_model_json)

    classifier.load_weights("model/model_weights.h5")

    classifier._make_predict_function()

    print(classifier.summary())

    f = open('model/history.pkl', 'rb')

    data = pickle.load(f)

    f.close()

    acc = data['accuracy']

```

```

accuracy = acc[9] * 100

text.insert(END,"CNN Training Model Accuracy = "+str(accuracy)+"\n")

else:

    classifier = Sequential()

    classifier.add(Convolution2D(32, 3, 3, input_shape = (64, 64, 1), activation = 'relu'))

    classifier.add(MaxPooling2D(pool_size = (2, 2)))

    classifier.add(Convolution2D(32, 3, 3, activation = 'relu'))

    classifier.add(MaxPooling2D(pool_size = (2, 2)))

    classifier.add(Flatten())

    classifier.add(Dense(output_dim = 256, activation = 'relu'))

    classifier.add(Dense(output_dim = 1, activation = 'softmax'))

    print(classifier.summary())

    classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics =
['accuracy'])

    hist = classifier.fit(X_train, Y_train, batch_size=16, epochs=10, shuffle=True, verbose=2)

    classifier.save_weights('model/model_weights.h5')

    model_json = classifier.to_json()

    with open("model/model.json", "w") as json_file:

        json_file.write(model_json)

    f = open('model/history.pkl', 'wb')

    pickle.dump(hist.history, f)

    f.close()

    f = open('model/history.pkl', 'rb')

```

```

data = pickle.load(f)

f.close()

acc = data['accuracy']

accuracy = acc[9] * 100

text.insert(END, "CNN Training Model Accuracy = "+str(accuracy)+"\n")

def predict():

    name = filedialog.askopenfilename(initialdir="testImages")

    img = cv2.imread(name)

    img = cv2.resize(img, (64,64))

    im2arr = np.array(img)

    im2arr = im2arr.reshape(1,64,64,3)

    XX = np.asarray(im2arr)

    XX = XX.astype('float32')

    XX = XX/255

    preds = classifier.predict(XX)

    print(str(preds)+" "+str(np.argmax(preds)))

    predict = np.argmax(preds)

    print(predict)

    img = cv2.imread(name)

    img = cv2.resize(img,(450,450))

    msg = "

    if predict == 0:

        cv2.putText(img, 'Fake', (10, 25), cv2.FONT_HERSHEY_SIMPLEX, 0.6, (0, 255, 255), 2)

```

```

    msg = 'Fake'

else:

    cv2.putText(img, 'Real', (10, 25), cv2.FONT_HERSHEY_SIMPLEX, 0.6, (0, 255, 255), 2)

    msg = 'Real'

cv2.imshow(msg, img)

cv2.waitKey(0)

def graph():

    f = open('model/history.pckl', 'rb')

    data = pickle.load(f)

    f.close()

    accuracy = data['accuracy']

    loss = data['loss']

    plt.figure(figsize=(10, 6))

    plt.grid(True)

    plt.xlabel('Iterations')

    plt.ylabel('Accuracy/Loss')

    plt.plot(loss, 'ro-', color='red')

    plt.plot(accuracy, 'ro-', color='green')

    plt.legend(['Loss', 'Accuracy'], loc='upper left')

    plt.title('CNN Accuracy & Loss')

    plt.show()

font = ('times', 16, 'bold')

```

```

title = Label(main, text='detection of fake currency ', justify=LEFT)

title.config(bg='deep skyblue', fg='white')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=100,y=5)

title.pack()

font1 = ('times', 13, 'bold')

uploadButton = Button(main, text="Upload Dataset", command=upload)

uploadButton.place(x=10,y=100)

uploadButton.config(font=font1)

processButton = Button(main, text="Image Preprocessing", command=processImages)

processButton.place(x=280,y=100)

processButton.config(font=font1)

cnnButton = Button(main, text="Generate CNN Model", command=generateModel)

cnnButton.place(x=10,y=150)

cnnButton.config(font=font1)

predictButton = Button(main, text="Upload Test Image", command=predict)

predictButton.place(x=280,y=150)

predictButton.config(font=font1)

graphButton = Button(main, text="Accuracy & Loss Graph", command=graph)

graphButton.place(x=10,y=200)

graphButton.config(font=font1)

font1 = ('times', 12, 'bold')

```



```
text=Text(main,height=20,width=120)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=10,y=250)

text.config(font=font1)

main.config(bg='LightSteelBlue3')

main.mainloop()
```

5.4 SCREENSHOTS

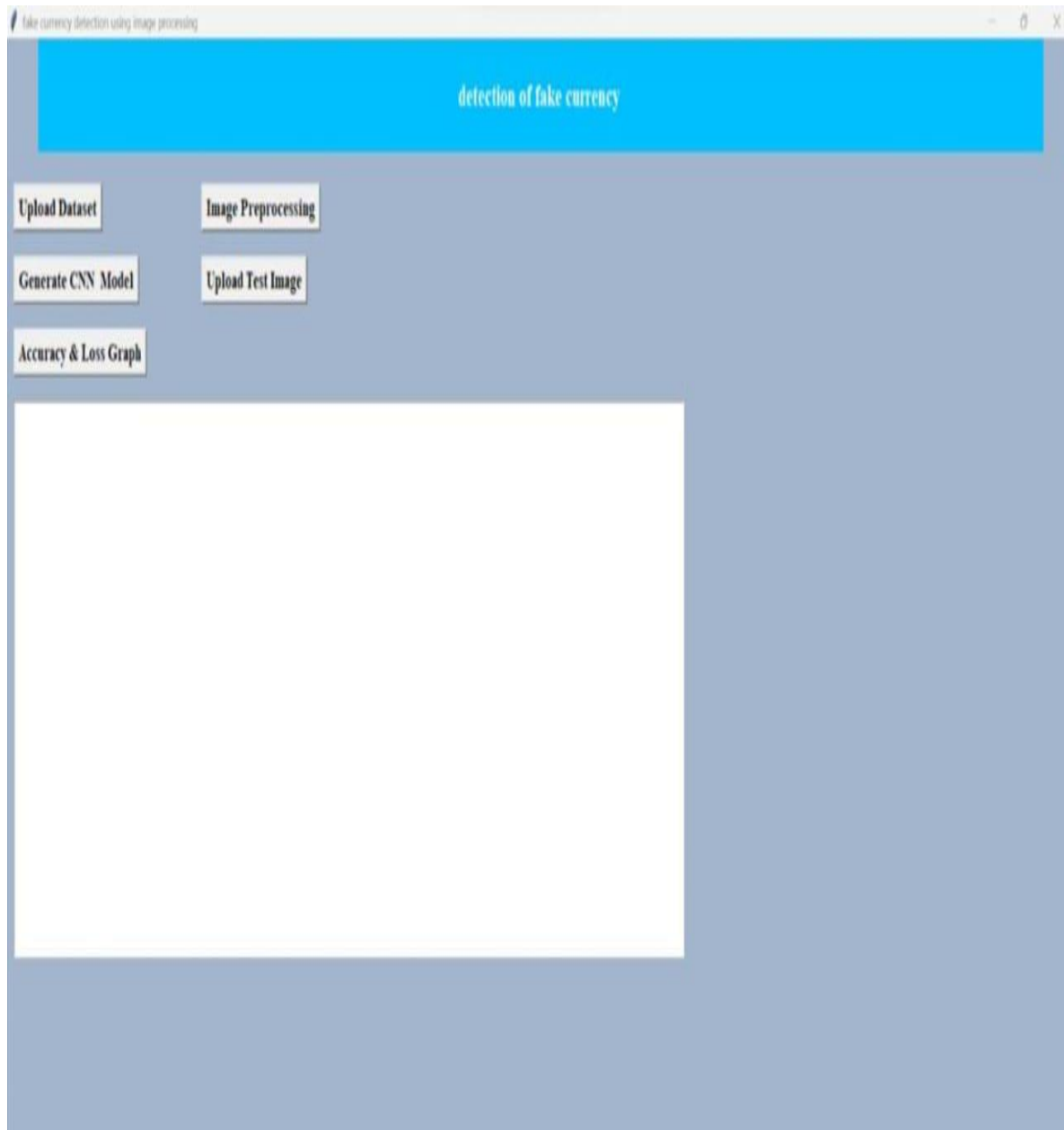


Figure 5.12 User Window.

The five buttons will be opened are:

- 1) Upload Dataset
- 2) Generate CNN Model
- 3) Accuracy & Loss Graph
- 4) Image Processing

5) Upload Test Image

1. Select dataset from the folders.

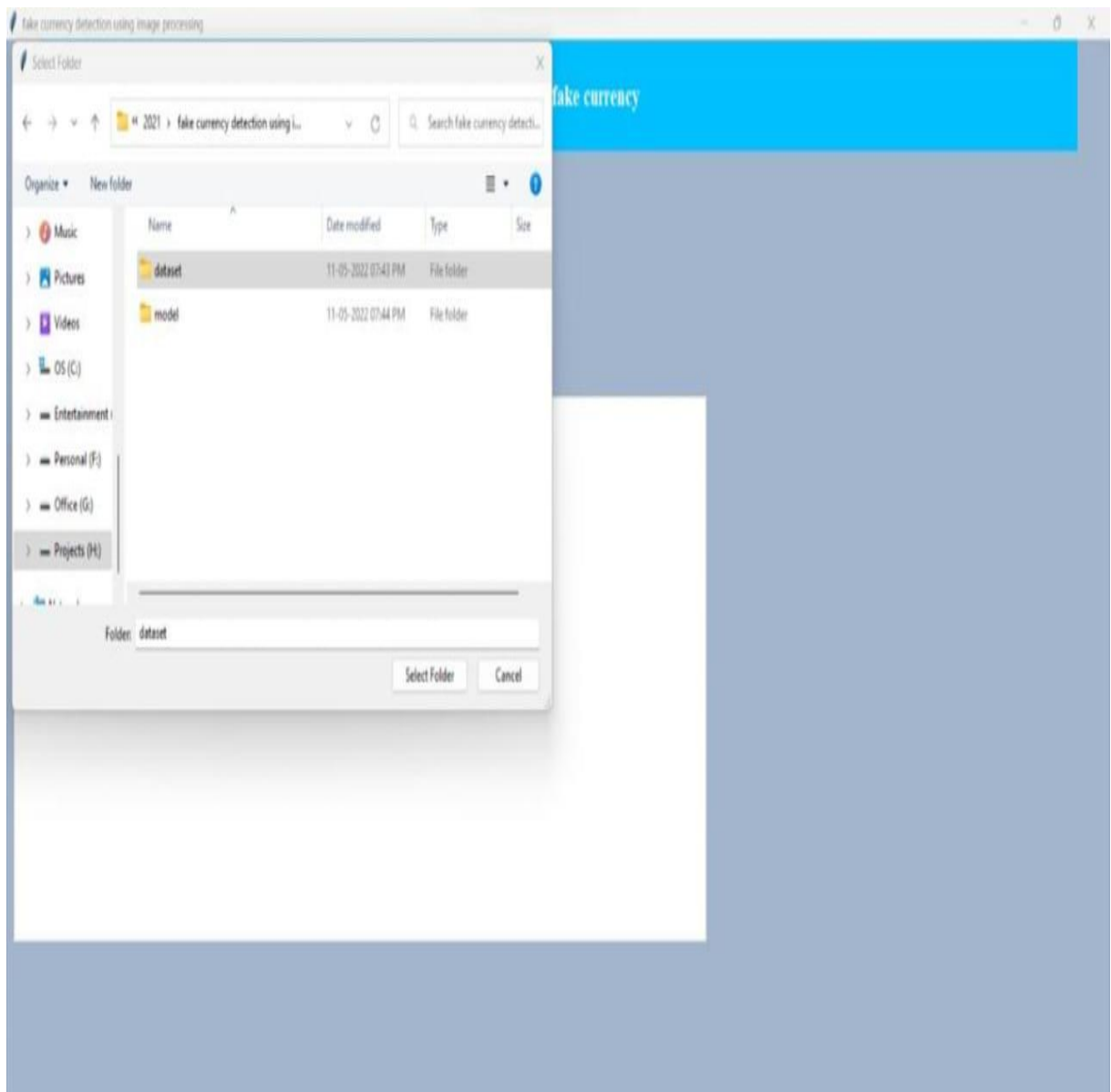


Figure 5.13 Selecting the Dataset Folder.

2. Uploading all the images of the dataset.

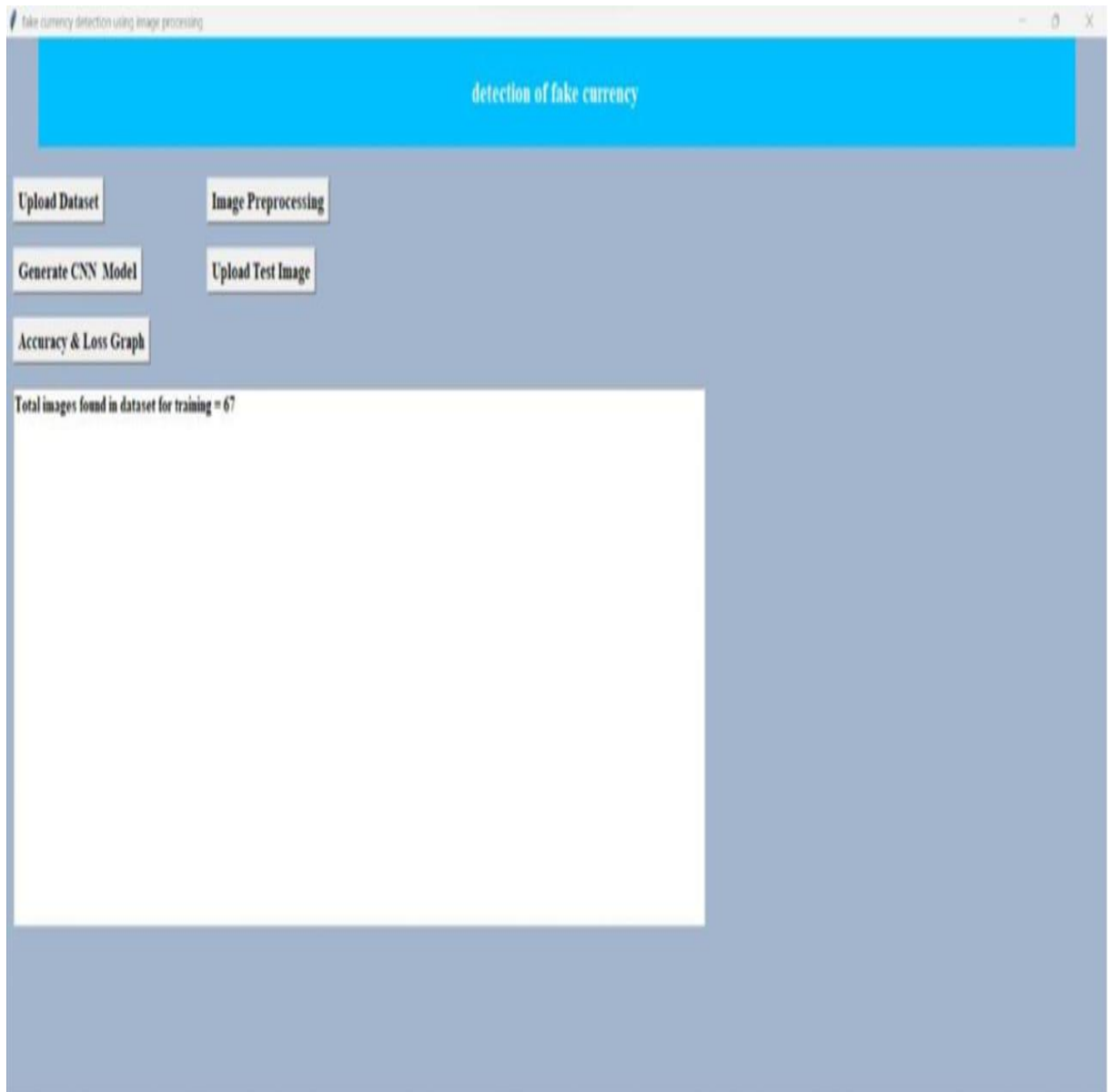


Figure 5.14 Uploading Dataset.

3. Detection of Fake or Real Currency Notes.

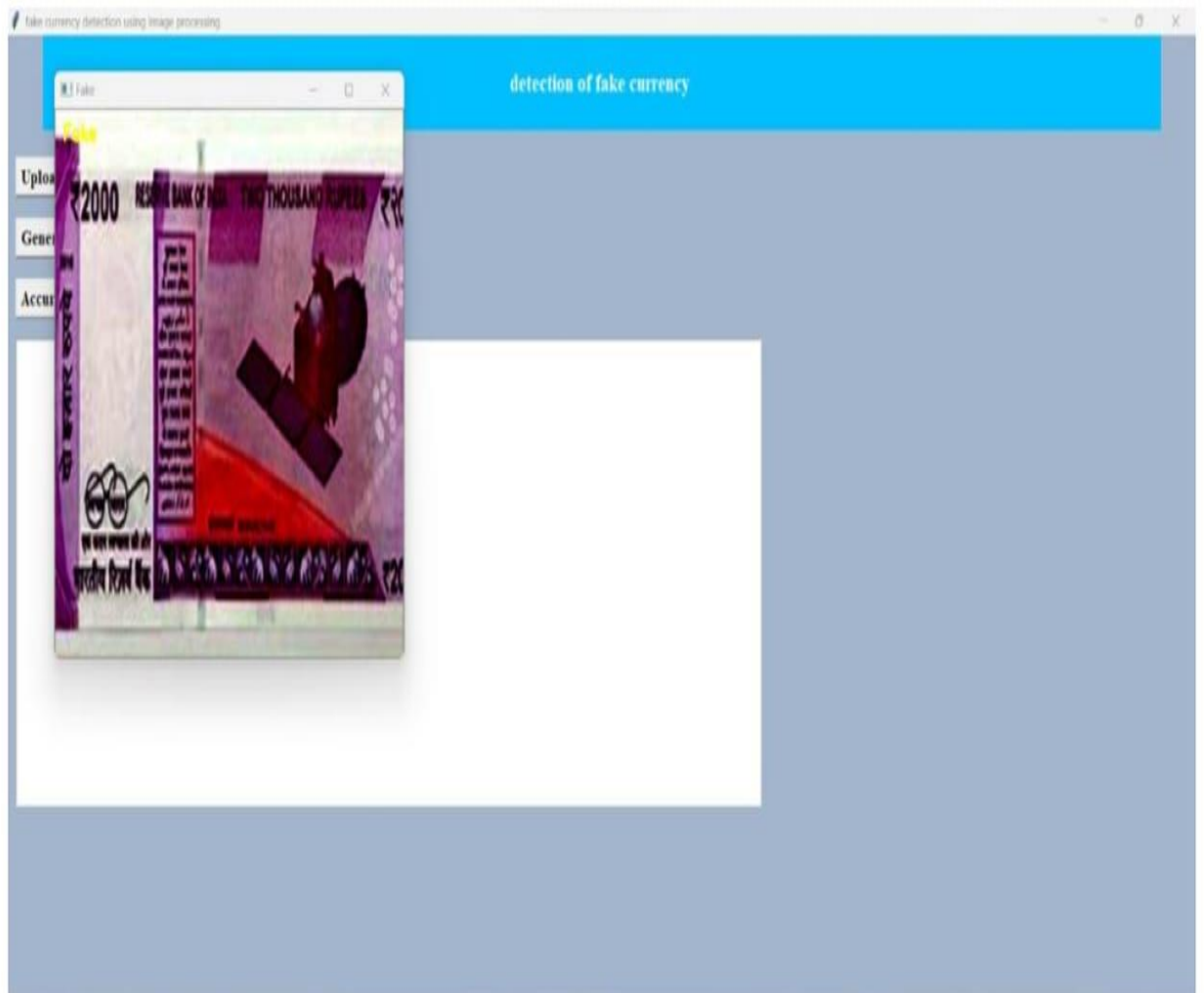


Figure 5.15 Detection of Real or Fake Currency.

CHAPTER -6

TESTING AND VALIDATION

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. The increasing visibility of software as a system element and attendant costs associated with a software failure are motivating factors for we planned, through testing. Testing is the process of executing a program with the intent of finding an error. The design of tests for software and other engineered products can be as challenging as the initial design of the product itself.

There of basically two types of testing approaches. One is Black-Box testing; the specified function that a product has been designed to perform, tests can be conducted that demonstrate each function is fully operated.

The other is White-Box testing – knowing the internal workings of the product, tests can be conducted to ensure that the internal operation of the product performs according to specifications and all internal components have been adequately exercised.

White box and Black box testing methods have been used to test this package. The entire loop constructs have been tested for their boundary and intermediate conditions. The test data was designed with a view to check for all the conditions and logical decisions. Error handling has been taken care of by the use of exception handlers.

6.1 TESTING STRATEGIES

Testing is a set of activities that can be planned in advanced and conducted systematically. A strategy for software testing must accommodation low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.

Software testing is one element of verification and validation. Verification refers to the set of activities that ensure that software correctly implements as specific function. Validation refers to a different set of activities that ensure that the software that has been built is traceable to customer requirements.

The main objective of software is testing to uncover errors. To fulfill this objective, a series of test steps unit, integration, validation and system tests are planned and executed. Each test step is accomplished through a series of systematic test technique that assist in the design of test cases. With each testing step, the level of abstraction with which software is considered is broadened.

Testing is the only way to assure the quality of software and it is an umbrella activity rather than a separate phase. This is an activity to be performed in parallel with the software effort and one that consists of its own phases of analysis, design, implementation, execution and maintenance.

Unit Testing:

This testing method considers a module as single unit and checks the unit at interfaces and communicates with other modules rather than getting into details at statement level. Here the, which will take some input and generate output. Outputs for a given set of input combination are pre-calculated and are generated by the module.

System Testing:

Here all the pre tested individual modules will be assembled to create the larger system and tests are carried out at system level to make sure that all modules are working in synchronous with each other. This testing methodology helps in making sure that all modules which are running perfectly when checked individually are also running in cohesion with other modules. For this testing, we create test cases to check all modules once and then a generated test combination of test paths throughout the system to make sure that no path is making its way into chaos.

Integrated Testing:

Testing is a major quality control measure employed during software development. Its basic function is to detect errors. Sub functions when combined may not produce than it is desired. Global data structures can represent the problems. Integrated testing is a systematic technique for constructing the program structure while conducting the tests. To uncover errors that are associated with interfacing the objective is to make unit test modules and built a program structure that has been detected by design. In a non-incremental integration all the modules are combined in advance and the program is tested as a whole. Here errors will appear in an endless loop function. In incremental testing the program is constructed and tested in small segments where the errors are isolated and corrected.

Different incremental integration strategies are top–down integration, bottom–up integration, regression testing.

Regression Testing:

Each time a new module is added as a part of integration as the software changes. Regression testing is an actually that helps to ensure changes that do not introduce unintended behavior as additional errors.

Regression testing maybe conducted manually by executing a subset of all test cases or using automated capture play back tools enables the software engineer to capture the test case and results for subsequent playback and compression. The regression suit contains different classes of test cases. A representative sample to tests that will exercise all software functions.

Additional tests that focus on software functions that are likely to be affected by the change.

One is **Black-Box testing**; the specified function that a product has been designed to perform, tests can be conducted that demonstrate each function is fully operated.

The other is **White-Box testing** – knowing the internal workings of the product, tests can be conducted to ensure that the internal operation of the product performs according to specifications and all internal components have been adequately exercised. White box and Black box testing methods have been used to test this package. The entire loop constructs have been tested for their boundary and intermediate conditions. The test data was designed with a view to check for all the conditions and logical decisions. Error handling has been taken care of by the use of exception handlers.

CHAPTER -7

RESULTS AND DISCUSSIONS

Performance: Several performance requirements were established, checking for inputs, outputs and working.

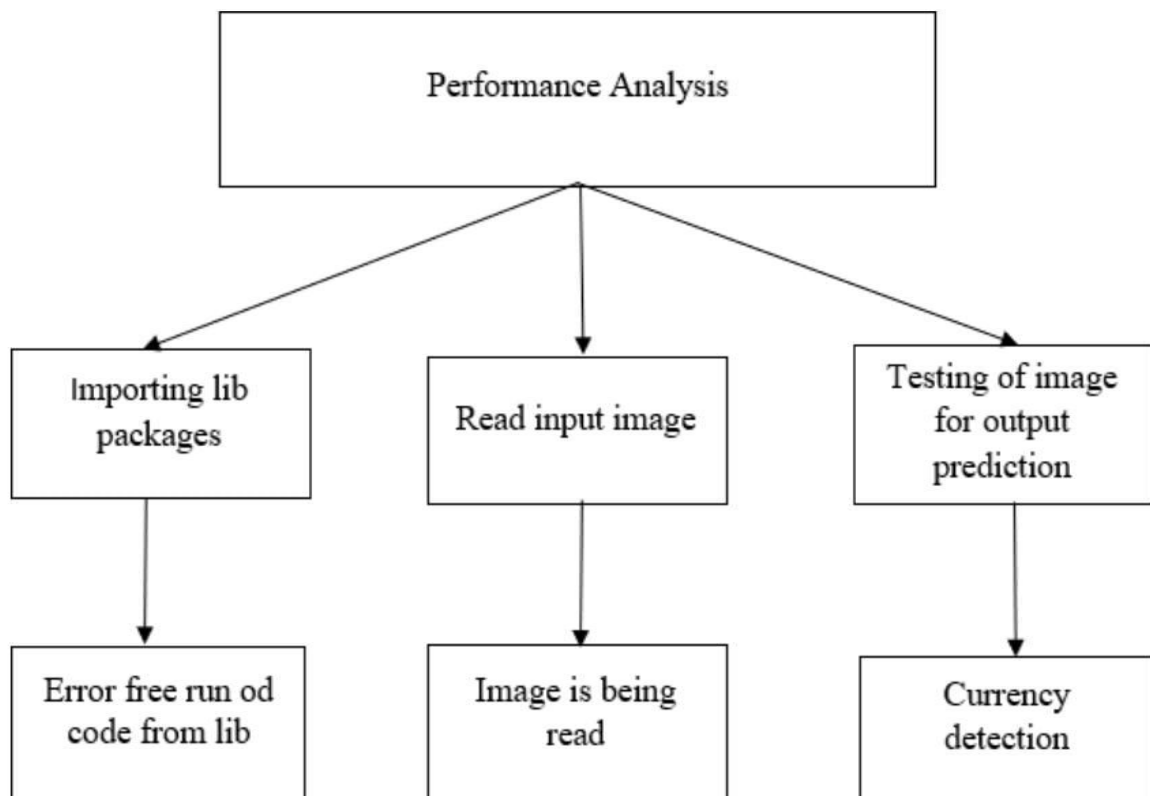


Figure 7.1 Performance Analysis.

Environmental: No harm for environmental parameters.

Social: Feasibility for everyone in day-to-day life.

Accuracy: In general, performance data obtained using sampling techniques are less accurate than data obtained by using counters or timers. In the case of timers, the accuracy of the clock must be taken into account.

Simplicity: User friendly.

Flexibility: A flexible can be extended easily to collect additional performance data or to provide different views of the same data. Flexibility and simplicity are often opposing requirements.

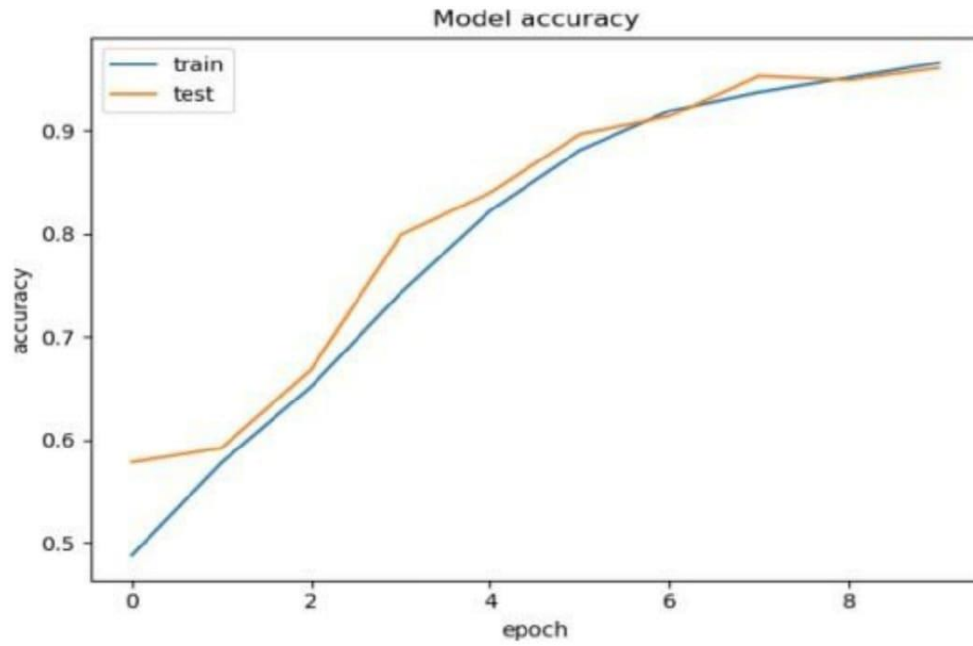


Figure 7.2 Accuracy of CNN Model.

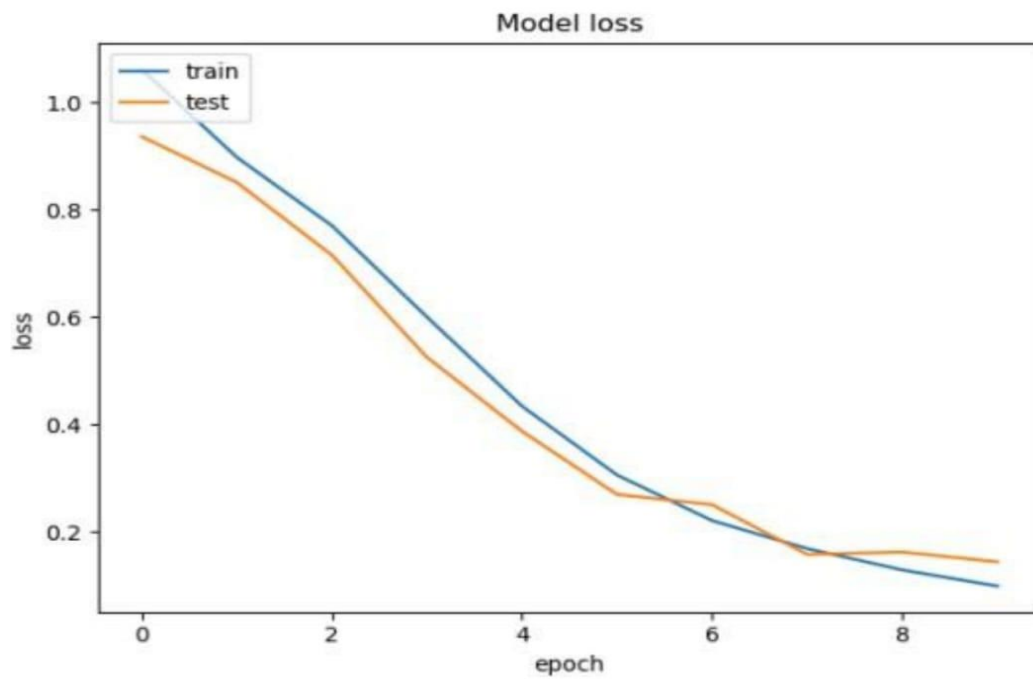


Figure 7.3 Loss of CNN Model.

Figure 7.2 shows the graph of the accuracy with respect to the epochs. It's observed that as the epochs increase, the accuracy of the model also increases.

Further Figure 7.3 shows the loss function and it can be clearly seen that the loss decreases with increase in the epochs.

CHAPTER -8

CONCLUSION AND FUTURE ENHANCEMENTS

8.1 CONCLUSION

In this work, we have discussed that how our proposed system detects the fake bank currency using machine learning algorithms. The proposed system is also scalable for detecting the whether the currency is fake or not by image processing. The system is not having complex process to detect the whether the data contains fake bank currency like the existing system. Proposed system gives genuine and fast result than existing system. Here in this system we use CNN algorithm to detect whether currency is fake or not.

8.2 FUTURE ENHANCEMENT

Technology is advancing at a rapid pace these days. The proposed technique can be used to detect coins as well as recognize phony currencies. Other countries' currencies can be added, and a comparison between them can be made. When a picture is loaded into the training folder from the outside, it does not provide 100 percent accuracy. By optimizing the system, we can solve this problem.

CHAPTER -9

BIBLIOGRAPHY

- [1] A. Patle and D. S. Chouhan, “SVM Kernel Functions for Classification”, ICATE 2013.
- [2] A. Roy, B. Halder, and U. Garain, “Authentication of currency notes through printing technique verification,” Proceedings of the Seventh Indian Conference on Computer Vision, Graphics and Image Processing -ICVGIP '10, pp. 383–390, 2010.
- [3] C. Gigliarano, S. Figini, P. Muliere, “Making classifier performance comparisons when ROC curves intersect”, Computational Statistics and Data Analysis 77 (2014) 300–312.
- [4] C. Kumar and A. K. Dudyala, “Banknote Authentication using Decision Tree rules and Machine Learning Techniques”, International Conference on Advances in Computer Engineering and Applications(ICACEA), 2015.
- [5] C.-Y. Yeh, W.-P. Su, and S.-J. Lee, “Employing multiple kernel support vector machines for counterfeit banknote recognition,” Applied Soft Computing, vol. 11, no. 1, pp. 1439–1447, Jan. 2011.
- [6] E. Gillich and V. Lohweg, “Banknote Authentication”, 2014.
- [7] F. Takeda, L. Sakoobunthu and H. Satou, “Thai Banknote Recognition Using Neural Network and Continues Learning by DSP Unit”, International Conference on Knowledge-Based and Intelligent Information and Engineering Systems, 2003.
- [8] H. Hassanpour and E. Hallajian, “Using Hidden Markov Models for Feature Extraction in Paper Currency Recognition.
- [9] <https://archive.ics.uci.edu/ml/datasets/banknote+authentication>.
- [10] M. Aoba, T. Kikuchi, and Y. Takefuji, “Euro Banknote Recognition System Using a Three-layered Perceptron and RBF Networks”, IPSJ Transactions on Mathematical Modeling and its Applications, May 2003.

- [11] M. Thirunavukkarasu, K. Dinakaran, E.N Satishkumar and S Gnanendra, "Comparison of support vector machine(svm) and Back propagation network (bpn) methods in predicting the protein Virulence factors", Jr. of Industrial Pollution Control 33(2)(2017)pp 11-19.
- [12] M. Lee and T. Chang, "Comparison of Support Vector Machine and Back Propagation Neural Network in Evaluating the Enterprise Financial Distress", International Journal of Artificial Intelligence & Applications 1.3 (2010) 31-43.
- [13] P. D. Shahare and R. N. Giri, "Comparative Analysis of Artificial Neural Network and Support Vector Machine Classification for Breast Cancer Detection", International Research Journal of Engineering and Technology, Dec-2015.
- [14] S. Desai, S. Kabade, A. Bakshi, A. Gunjal, M. Yeole, "Implementation of Multiple Kernel Support Vector Machine for Automatic Recognition and Classification of Counterfeit Notes", International Journal of Scientific & Engineering Research, October-2014.
- [15] S. Omatu, M. Yoshioka and Y. Kosaka, "Bankcurrency Classification Using Neural Networks", IEEE, 2007.
- [16] Verma, Vivek K., Anju Yadav, and Tarun Jain. "Key Feature Extraction and Machine Learning-Based Automatic Text Summarization." *Emerging Technologies in Data Mining and Information Security*. Springer, Singapore, 2019. 871-877.
- [17] Z. Huang, H. Chen, C. J. Hsu, W. H. Chen and S. Wuc, "Credit rating analysis with support vector machines and neural network: a market comparative study", 2004.