A Major Project

On

Fake Social Media Profile detection and reporting

Submitted to **CMR ENGINEERING COLLEGE**

BACHELOR OF TECHNOLOGY IN CSE-Cybersecurity

Submitted By

K. JAGADISH (208R1A6237)

Under the Esteemed guidance of

Mr. M. Ramasamy
Assistant Professor, Department of CSE-Cyber Security

Department of CSE-Cybersecurity

CMR ENGINEERING COLLEGE

UGC AUTONOMOUS

(Accredited by NBA Approved by AICTE,NEW DELHI, Affiliated to JNTU Hyderabad)(Kandlakoya, Medchal Road, Medchal Malkajgiri Dist. Hyderabad-501 401)

2023-2024

CMR ENGINEERING COLLEGE

(Accredited by NBA,Approved by AICTE NEW DELHI, Affiliated to JNTU, Hyderabad)

Kandlakoya, Medchal Road, Hyderabad-501 401

DEPARTMENT OF CSE-CYBER SECURITY

CERTIFICATE

This is to certify that the project entitled "**Fake social media profile detection and reporting**" is a bonafide work carried out by

K.JAGADISH **(208R1A6237)**

in partial fullfilment of the requirement for the award of the degree of **BACHELOR OF TECHNOLOGY** in **CSE-CYBER SECURITY** from CMR Engineering College, affiliated to JNTU, Hyderabad, under our guidance and supervision.

The results presented in this project have been verified and are found to be satisfactory. The results embodied in this project based Learning has not been submitted to any other university for the award of any other degree or diploma.

Internal Guide	**Major Project Coordinator**	**Head of the Department**
Mr.M.Ramasamy	**Ms.T.Usha**	**Dr.V.SathiyaSuntharam**
Assistant Professor	Assistant Professor	Professor & H.O.D
CSE-CS Department	CSE-CS Department	CSE-CS Department
CMREC	CMREC	CMREC

DECLARATION

This is to certify that the work reported in the present project entitled **“**Fake Social Media Profile detection and reporting**”** is a record of bonafide work done by us in the Department of Computer Science and Engineering-Cyber Security, CMR Engineering College, JNTU Hyderabad. The reports are based on the project work done entirely by us and not copied from any other source. We submit our project for further development by any interested students who share similar interests to improve the project in the future.

The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma to the best of our knowledge and belief.

K.JAGADISH (208R1A6237)

ACKNOWLEDGMENT

I am extremely grateful to **Dr. A. Srinivasula Reddy**, Principal and **Dr. V. Sathiya Suntharam**, HOD, Department of CSE-CYBER SECURITY, **CMR Engineering College** for their constant support.

We are extremely thankful to **Mr.M.Ramasamy**, Assistant Professor, Internal Guide, Department of CSE-CYBER SECURITY, for his/ her constant guidance, encouragement and moral support throughout the project.

We will be failing in duty if I do not acknowledge with grateful thanks to the authors of the references and other literatures referred in this Project.

We thank **Ms.T.usha & Ms.A.Anusha** Major Project Coordinator for his constant support in carrying out the project activities and reviews.

We express our thanks to all staff members and friends for all the help and co-ordination extended in bringing out this project successfully in time.

Finally, we are very much thankful to our parents who guided us in every step.

K.JAGADISH (208R1A6237)

ABSTRACT

In seeing the present condition, online social networks are engaging with the majority of the people. From child to adult, all are spending a considerable time on these platforms either by exchanging information or making efficient communication with others. But nowadays, these social networking sites are suffering from a lot of fake accounts in taking advantage of vulnerabilities, either taking the benefits or targeting accounts attempting cybercrimes. Platforms for social media like Facebook, Twitter, Instagram, and others have a big impact on our lives. All across the world, people are actively engaged in it. But,it also needs to address the problem of false profiles. Fake accounts are regularly made by people, software, or machines. They are employed in the spread of rumors and illegal actions like phishing and identity theft. This project uses machine learning techniques to discriminate between fake and authentic social profiles based on characteristics such as follower and friend counts, status changes, and more. In this section, we talk about Aritifical neural networks. The important traits are picked to judge the veracity of a social media page. The architecture and hyperparameters are also discussed. Lastly, after the models have been trained, results are generated. As a result, the output is 0 for true profiles and 1 for fake profiles. It is possible to disable or delete a fake profile when it is found, preventing cyber security issues.

Keywords: Social Media - World - Rumours - Fake Profiles – Detection

CONTENTS

ABSTRACT

LIST OF FIGURES

CHAPTER-01

INTRODUCTION

In the current generation, everyone's social life is now entwined with online social networks. It has been simpler to add new friends and stay in touch with them and their updates. Online social networks influence a variety of fields, including research, education, community activism, employment, and business. These online social networks have been the subject of research to see how they affect people. Instructors may quickly contact their students using this, creating a welcoming atmosphere for them to learn. Teachers are becoming more familiar with these sites and using them to provide online classroom pages, assign assignments, hold conversations, and other activities that greatly enhance learning. Employers may utilize these social networking sites to find and hire skilled individuals who are enthusiastic about their jobs. In addition to all of this, propaganda spread via social media is a problem. False accounts sharing incorrect and unsuitable information cause conflicts. The following are the main goals of this research project: These false profiles are likewise created to get followers. More people are harmed by phony profiles than by other internet crimes. Thus, now that the user is aware, it's crucial to recognize a fake profile. False accounts on the website are mostly used to spread spam, information, and other misleading information. The technological work that has been done and what is being done right now to identify false profiles is examined in this paper. The majority of fraudulent profiles are created with the intention of spamming, phishing, and getting more followers. Fake accounts have all the tools necessary to conduct online crimes. Identity theft and data breaches are substantial risks posed by fake accounts. All user information is transferred to faraway servers when users view the URLs supplied by these fake accounts, where it may be utilized against them. False accounts that claim to have been made on behalf of organizations or individuals can also harm their reputation and reduce the number of followers and likes they receive.

CHAPTER-02

LITERATURE SURVEY

1.Deep Profile: Finding fake profile in online social network using dynamic CNN

Online Social Networks (OSN) are popular applications for sharing various data, including text, photos, and videos. However, fake account problems are one of the obstacles in the current OSN systems. Attacker exploits fake accounts to distribute misleading information such as malware, virus, or malicious URLs. Inspired by the big successes of deep learning in computer vision, mainly in automatic feature extraction and representation, we propose Deep Profile, a deep neural network (DNN) algorithm to deal with fake account issues. Instead of using standard machine learning, we construct a dynamic CNN to train a learning model in fake profile classification. Notably, we propose a novel pooling layer to optimize the neural network performance in the training process. Demonstrated by the experiments, we harvest a promising result with better accuracy and small loss than common learning algorithms in a malicious account classification task.

2.Fake detector: Effective fake news detection with deep diffusive neural network.

In recent years, due to the booming development of online social networks, fake news for various commercial and political purposes has been appearing in large numbers and widespread in the online world. With deceptive words, online social network users can get infected by these online fake news easily, which has brought about tremendous effects on the offline society already. An important goal in improving the trustworthiness of information in online social networks is to identify the fake news timely. This paper aims at investigating the principles, methodologies and algorithms for detecting fake news articles, creators and subjects from online social networks and evaluating the corresponding performance. This paper addresses the challenges introduced by the unknown characteristics of fake news and diverse connections among news articles, creators and subjects. This paper introduces a novel gated graph neural network, namely FAKEDETECTOR. Based on a set of explicit and latent features extracted from the textual information, FAKEDETECTOR builds a deep diffusive network model to learn the representations of news articles, creators and subjects simultaneously.

3.Fake Profile Detection on Social Networking Websites.

These days, social media has a significant impact on everyone’s life. Most people frequently utilize social media platforms. Each of these social media platforms offers benefits and drawbacks, as well as security risks for our information. To determine who poses threats on these platforms, it is necessary to distinguish between the real and fake social media profiles. There are traditionally used various methods for identifying fake social media accounts. But these platforms need to be better at identifying phoney accounts. The accuracy rate of identifying fake accounts utilising timestamp data types is improved in this proposed work employing high gradient boosting algorithms and Natural Language Processing. In order to investigate the relationship between various machine learning methods and multi-features in time series, this study employs a variety of machine learning techniques.

CHAPTER-03

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM:

In existing system, SVM algorithm and Random forest algorithm are used.

Disadvantages:

- SVM performance can be sensitive to the choice of kernel function and regularization parameters. Selecting appropriate values through cross-validation can be computationally expensive.
- Training an SVM on large datasets can be time-consuming, especially for non-linear kernels, as the optimization problem scales quadratically with the number of samples.
- While Random Forest models generally perform well, they can be challenging to interpret due to their ensemble nature. Understanding the decision-making process at the individual tree level may not be straightforward.
- Random Forests may produce biased results when dealing with imbalanced datasets, where one class is significantly more prevalent than the others. Techniques such as class weighting or resampling strategies may be needed to address this issue.

3.2 PROPOSED SYSTEM:

Proposed system is used to detect fake social media profiles by using ANN algorithm which gave better accuracy.

Proposed system considering various advantages :

Non-linearity: ANNs can model complex non-linear relationships between input and output variables, allowing them to capture intricate patterns and correlations within the data that linear models may overlook.

Flexibility: ANNs are highly flexible and can be applied to a wide range of tasks, including classification, regression, clustering, and pattern recognition. They can handle various types of data, such as numerical, categorical, and textual data.

Parallel Processing: ANNs can perform computations in parallel, enabling them to process large volumes of data efficiently. This parallel processing capability makes ANNs suitable for tasks requiring real-time or high-speed processing, such as image and speech recognition.

Adaptability: ANNs can adapt and learn from data iteratively, continuously improving their performance over time. Through techniques like backpropagation and gradient descent, ANNs adjust their internal parameters to minimize prediction errors and optimize model performance.

Feature Learning: ANNs can automatically learn relevant features from raw data, eliminating the need for manual feature extraction. This ability to learn hierarchical representations of data can be particularly advantageous in tasks involving high-dimensional or unstructured data, such as image and text processing.

Robustness to Noise: ANNs are inherently robust to noisy input data and can generalize well to unseen examples. They can learn to ignore irrelevant or noisy features, focusing on the most informative aspects of the data to make accurate predictions.

3.3 FUNCTIONAL REQUIREMENTS

It provides the users a clear statement of the functions required for the system in order to solve the project information problem it contains a complete set of requirements for the applications.A requirement is condition that the application must meet for the customer to find the application satisfactory. A requirement has the following characteristics

1. It provides a benefit to the origination.
2. It describes the capabilities the application must provide in business terms.
3. It does not describe how the application provides that capability.
4. It is stated in unambiguous words. Its meaning is clear and understandable. 5. It is verifiable.

3.4 NON-FUNCTIONAL REQUIREMENTS

Career recommendation non-functional requirements, like interests he has, how hours he can work likewise, with today's IT projects, to determine non-functional requirements, like availability, the approach requires that the designer 1st determine the scope: does the whole solution or only part of it need to be architected to meet minimum levels?

1. This is done through 4 steps:
2. Identify the critical areas of solutions
3. Identify the critical components within each critical area.
4. Determine each components availability and risk.
5. Model worst-case failure scenarios.

3.5 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-04

SOFTWARE AND HARDWARE REQUIREMENTS

SOFTWARE REQUIREMENTS:

• Operating System	: Windows
• Coding Language	: Python 3.7

HARDWARE REQUIREMENTS:

• System	: Pentium IV 2.4 GHz.
• Hard Disk	: 40 GB.
• Floppy Drive	: 1.44 Mb.
• Monitor	: 15 VGA Colour.
• Mouse	: Logitech.
• Ram	: 512 Mb.

CHAPTER-05

SYSTEM DESIGN

UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

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

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

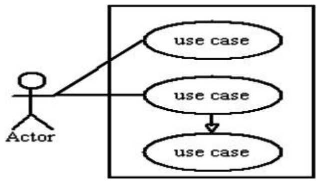
1. Use Case Diagram

What is a UML Use Case Diagram?

Use case diagrams model the functionality of a system using actors and use cases. Use cases are services or functions provide by the system to its users.

Basic Use Case Diagram Symbols and Notations System

Draw your system's boundaries using a rectangle that contains use cases. Place actors outside the system's boundaries.

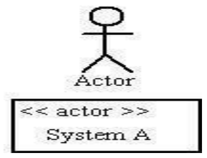


Use Case

Draw the use case using ovals. Label with verbs that are represents the systems functions.

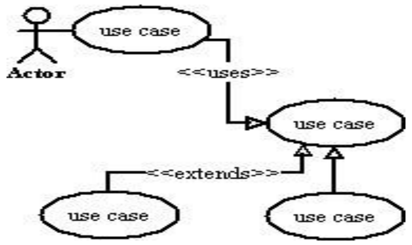
Actors

Actors are the users of a system. When one system is the actor of another system, label the actor system with the actor stereotype.



Relationships

Illustrate relationships between an actor and a use case with a simple line. For relationships among use cases, use arrows labelled either "uses" or "extends." A "uses" relationship indicates that one use case is needed by another in order to perform a task.



2. Activity Diagram

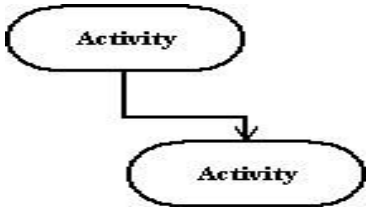
An activity diagram illustrates the dynamic nature of a system by modelling the flow of control from activity to activity. An activity represents an operation on some class in the system that results in a change in the state of the system. Typically, activity diagrams are used to model workflow or business processes and internal operation. Because an activity diagram is a special king of state chart diagram, it uses some of the same modelling conventions.

Basic Activity Diagram Symbols and Notations Action states

Action states represent the non-interruptible actions of objects. You can grow an action state in Smart Draw using a rectangle with rounded corners.

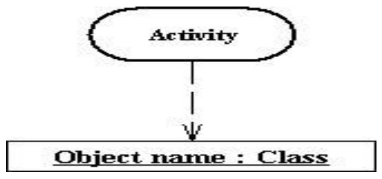
Action Flow

Action flow arrows illustrate the relationships among action states.



Object Flow

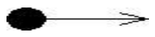
Object flow refers to the creation and modification of objects by activities. An object flow arrow from an action to an object means that the action creates or influences the object. An object flow arrow from an object to an action indicates that the action state uses the



object.

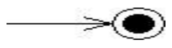
Initial State

A filled circle followed by an arrow represents the initial action state.



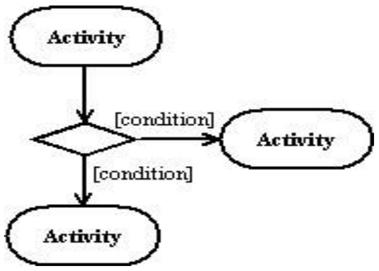
Final State

An arrow pointing to a filled circle nested inside another circle represents the final state.



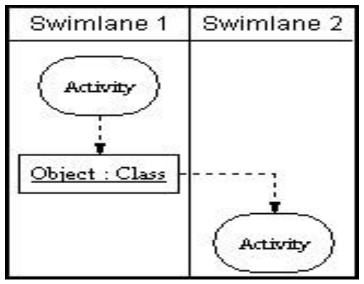
Branching

A gaining represents a decision with alternate paths. The outgoing alternates should be labelled with a cognition or guard expression.



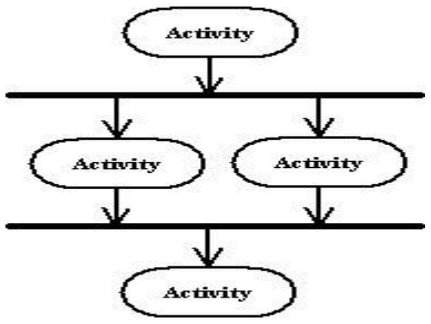
Swim lanes

Swim lanes group related activities into one column.



Synchronization

A synchronization bar helps illustrate parallel transitions. Synchronization is also called forking ang joining.

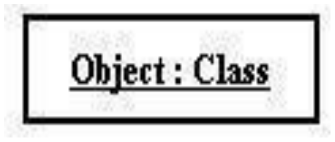


3. Sequence Diagram

Sequence diagrams describe interactions among classes in terms of an exchange of messages over time.

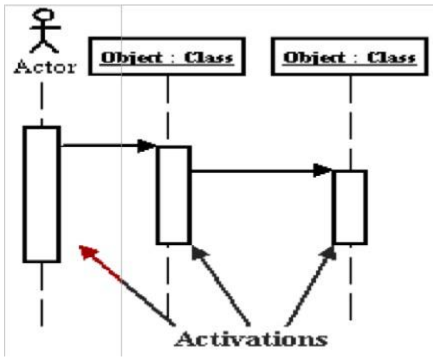
Basic Sequence Diagram Symbols and Notations Class roles

Class roles describe the way an object will behave in context. Use the UML object symbol to illustrate class roles, but don't list object attributes.



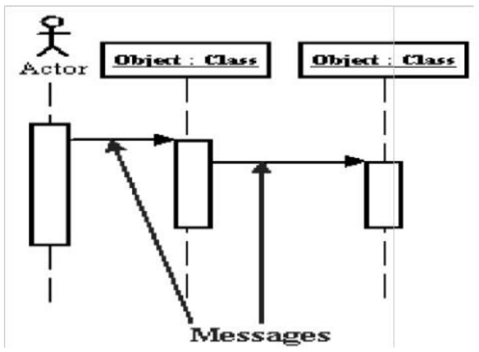
Activation

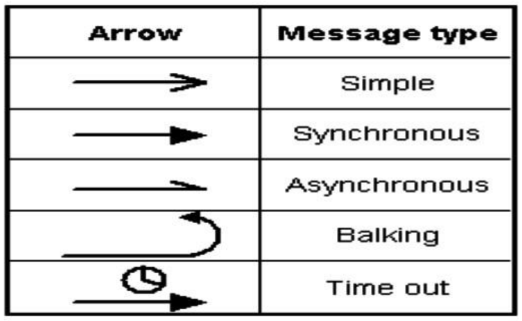
Activation boxes represent the time an object needs to complete a task.



Messages

Messages are arrows that represent communication between objects. Use half-arrowed lines to represent asynchronous messages. Asynchronous messages are sent from an object that will not wait for a response from the receiver before continuing its tasks.

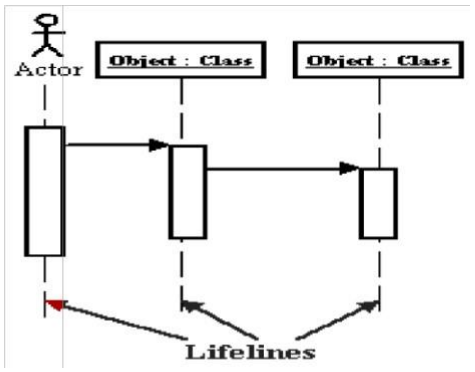




Arrow	Message type
→	Simple
➡	Synchronous
→	Asynchronous
↩	Balking
🕒➡	Time out

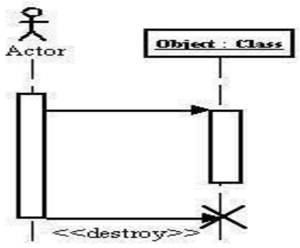
Lifelines

Lifelines are vertical gashes lines that indicate the object's presence over time



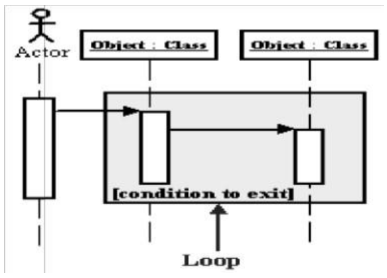
Destroying Objects

Objects can be terminated early using an arrow labelled "<< destroy >>" that points to an x.



Loops

A repetition or loop within a sequence diagram is depicted as a rectangle. Place the cognition for exiting the loop at the bottom left corner in square brackets.



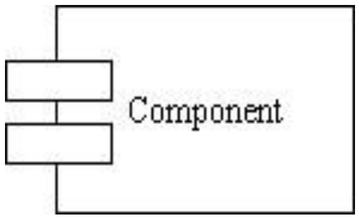
4. Components Diagram

What is a UML Component Diagram?

A component diagram describes the organization of the physical components in a system.

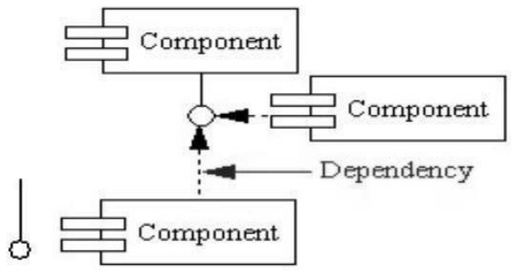
Component

A component is a physical building block of the system Learn how to resize grouped objects like components.



Interface

An interface describes a group of operations used or created by components**.**

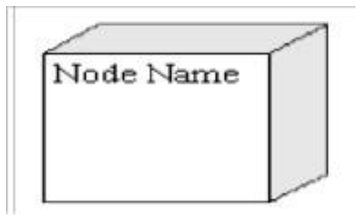


Dependencies

Draw dependencies among components using gashes arrows. Learn about line styles in Smart Draw.

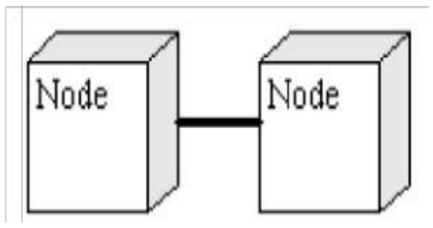
Component

A node is a physical resource that executes code components. Learn how to resize grouped objects like nodes.



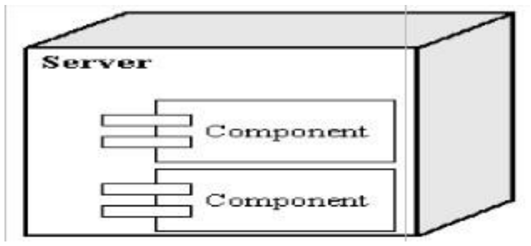
Association

Association refers to a physical connection between nodes, such as Ethernet. Learn how to connect two nodes.



Components and Nodes

Place components inside the node that deploys them.



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.

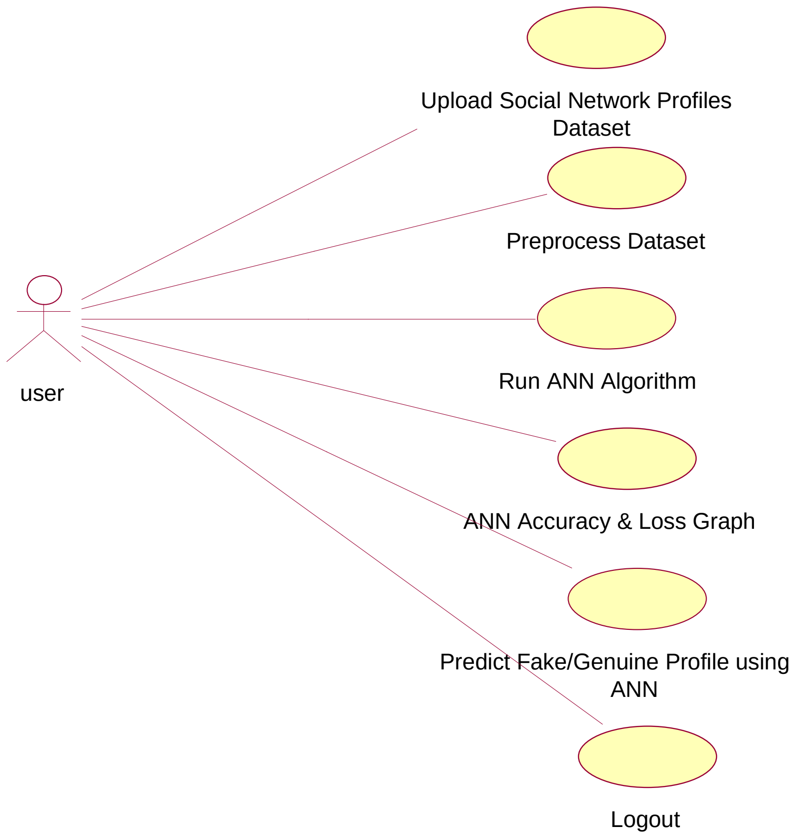

Upload Social Network Profiles Dataset
Preprocess Dataset
user
Run ANN Algorithm
ANN Accuracy & Loss Graph
Predict Fake/Genuine Profile using ANN
Logout

Figure: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.

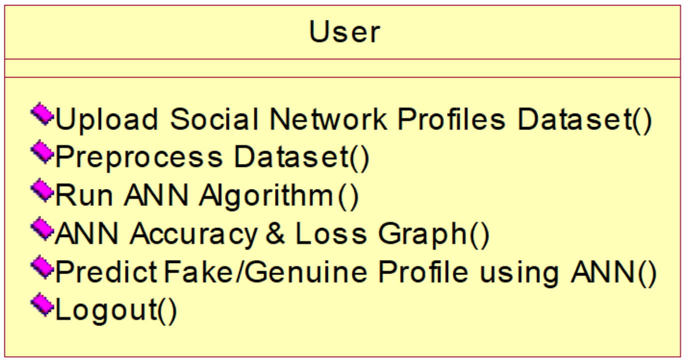

User
Upload Social Network Profiles Dataset()
Preprocess Dataset()
Run ANN Algorithm()
ANN Accuracy & Loss Graph()
Predict Fake/Genuine Profile using ANN()
Logout()

Figure: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.

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

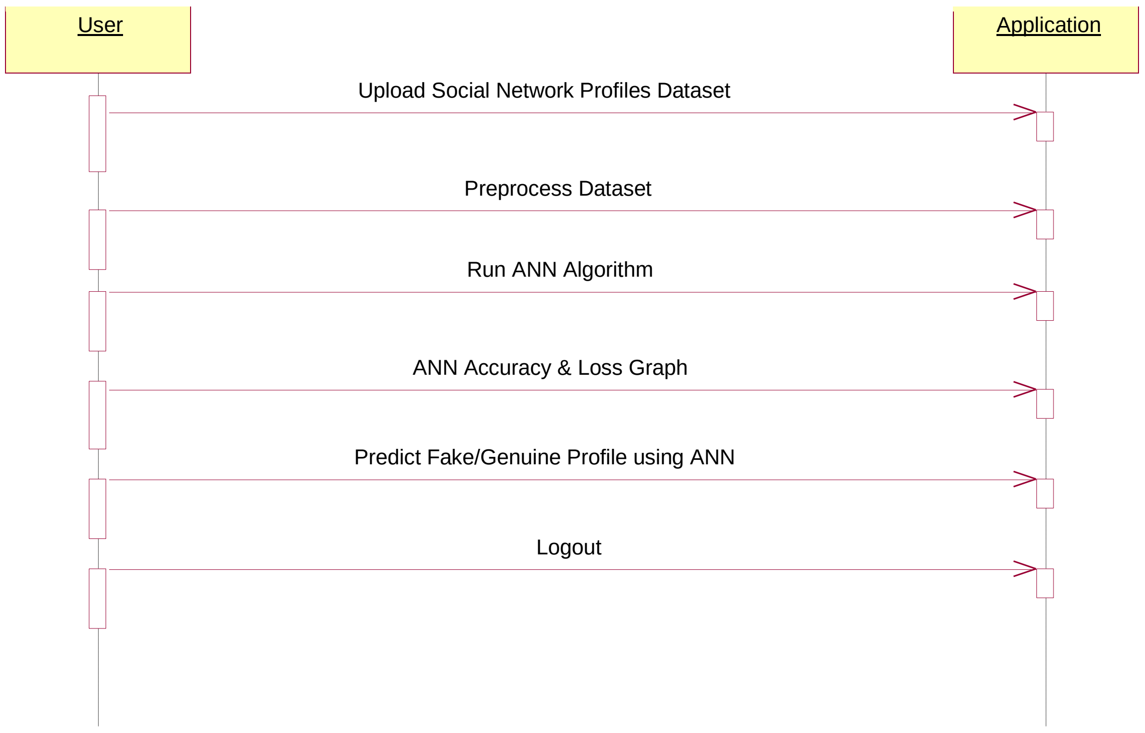

User
Application
Upload Social Network Profiles Dataset
Preprocess Dataset
Run ANN Algorithm
ANN Accuracy & Loss Graph
Predict Fake/Genuine Profile using ANN
Logout

Figure: 5.3 Sequence diagram

COLLABORATION DIAGRAM:

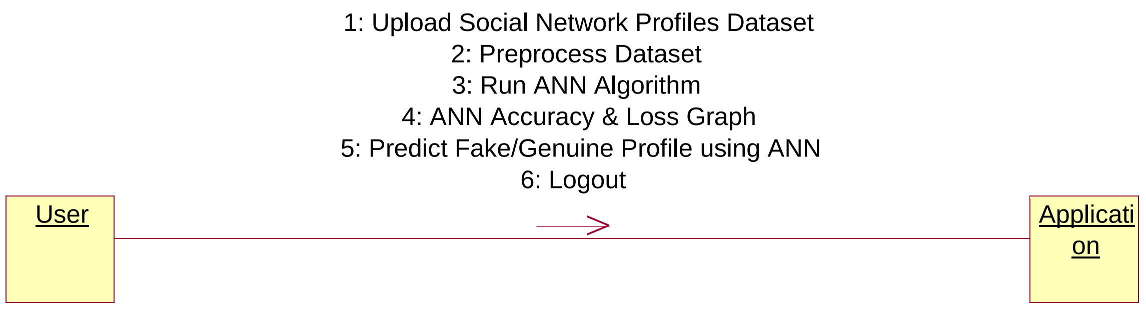

1: Upload Social Network Profiles Dataset
2: Preprocess Dataset
3: Run ANN Algorithm
4: ANN Accuracy & Loss Graph
5: Predict Fake/Genuine Profile using ANN
6: Logout
User
Application

Figure:5.4 collaboration diagram

CHAPTER-06

CODING

6.1 SOURCE CODE

```
from tkinter import messagebox

from tkinter import * from

tkinter import simpledialog

import tkinter import

matplotlib.pyplot as plt import

numpy as np from tkinter import

ttk from tkinter import filedialog

import pandas as pd

from sklearn.model_selection import train_test_split

from keras.models import Sequential from

keras.layers.core import Dense,Activation,Dropout

from keras.callbacks import EarlyStopping from

sklearn.preprocessing import OneHotEncoder from

keras.optimizers import Adam from keras.utils.np_utils

import to_categorical main = Tk()

main.title("IDENTIFING OF FAKE PROFILES ACROSS ONLINE SOCIAL NETWORKS BY

USING NEURAL NETWORK") main.geometry("1300x1200") main.config(bg="lightgreen")

global filename global X, Y

global X_train, X_test, y_train, y_test

global accuracy global dataset

global model def

loadProfileDataset():
```

```
global filename
global dataset
outputarea.delete('1.0', END)
filename = filedialog.askopenfilename(initialdir="Dataset")
outputarea.insert(END,filename+" loaded\n\n")
dataset = pd.read_csv(filename)
outputarea.insert(END,str(dataset.head()))
  def preprocessDataset():    global X, Y   global dataset
 globalX_train,X_test,y_train, y_test
outputarea.delete('1.0', END)
 X = dataset.values[:, 0:8]
 Y = dataset.values[:, 8]
 indices = np.arange(X.shape[0])
np.random.shuffle(indices)
X = X[indices]
Y = Y[indices]
Y = to_categorical(Y)
X_train,X_test,y_train,y_test=train_test_split(X,Y,test_size=0.2)
outputarea.insert(END,"\n\nDataset contains total profiles : "+str(len(X))+"\n")
outputarea.insert(END,"Total profiles used to train ANN algorithm : "+str(len(X_train))+"\n")
outputarea.insert(END,"Total profiles used to test ANN algorithm  : "+str(len(X_test))+"\n")
 def executeANN():    global model
outputarea.delete('1.0', END)
global X_train, X_test, y_train, y_test
global accuracy  model = Sequential()
model.add(Dense(200,input_shape=(8,)
, activation='relu', name='fc1'))
```

```
model.add(Dense(200, activation='relu',
name='fc2'))     model.add(Dense(2,
activation='softmax', name='output'))
optimizer = Adam(lr=0.001)
model.compile(optimizer, loss='categorical_crossentropy', metrics=['accuracy'])
print('ANN Neural Network Model Summary: ')
 print(model.summary())
 hist = model.fit(X_train, y_train, verbose=2, batch_size=5, epochs=200)
results = model.evaluate(X_test, y_test)
 ann_acc = results[1] * 100     print(ann_acc)
accuracy = hist.history
acc = accuracy['acc']
acc = acc[199] * 100
outputarea.insert(END,"ANN model generated and its prediction accuracy is : "+str(acc)+"\n")
def graph():
global accuracy
acc = accuracy['acc']
loss = accuracy['loss']
plt.figure(figsize=(10,6))
plt.grid(True)
plt.xlabel('Iterations')
plt.ylabel('Accuracy/Loss')
plt.plot(acc, 'ro-', color = 'green')
 plt.plot(loss, 'ro-', color = 'blue')
 plt.legend(['Accuracy', 'Loss'], loc='upper left')
 #plt.xticks(wordloss.index)
```

```
plt.title('ANN Iteration Wise Accuracy & Loss Graph')
plt.show()
def predictProfile():
outputarea.delete('1.0', END)
 global model
filename = filedialog.askopenfilename(initialdir="Dataset")
test = pd.read_csv(filename)
test = test.values[:, 0:8]
predict = model.predict_classes(test)
print(predict)
for i in range(len(test)):
 msg = ''
if str(predict[i]) == '0':
msg = "Given Account Details Predicted As Genuine"
if str(predict[i]) == '1':
msg = "Given Account Details Predicted As Fake"
outputarea.insert(END,str(test[i])+" "+msg+"\n\n")
def close():     main.destroy() font = ('times', 15, 'bold')
title = Label(main, text='IDENTIFING OF FAKE PROFILES ACROSS ONLINE SOCIAL
NETWORKS BY USING NEURAL NETWORK')
#title.config(bg='powder blue', fg='olive drab')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=0,y=5) font1 = ('times', 13,
'bold') ff = ('times', 12, 'bold')
```

```
uploadButton = Button(main, text="Upload Social Network Profiles Dataset",
command=loadProfileDataset) uploadButton.place(x=20,y=100)
uploadButton.config(font=ff)
processButton = Button(main, text="Preprocess Dataset", command=preprocessDataset)
processButton.place(x=20,y=150) processButton.config(font=ff)
annButton = Button(main, text="Run ANN Algorithm",
command=executeANN) annButton.place(x=20,y=200)
annButton.config(font=ff)
graphButton = Button(main, text="ANN Accuracy & Loss Graph",
command=graph) graphButton.place(x=20,y=250) graphButton.config(font=ff)
PredictButton = Button(main, text="Predict Fake/Genuine Profile using
ANN", command=predictProfile) predictButton.place(x=20,y=300)
predictButton.config(font=ff)
exitButton = Button(main, text="Logout", command=close)
exitButton.place(x=20,y=350) exitButton.config(font=ff)
font1 = ('times', 12, 'bold') outputarea =
Text(main,height=30,width=85) scroll =
Scrollbar(outputarea)
outputarea.configure(yscrollcommand=scroll.set)
outputarea.place(x=400,y=100)
outputarea.config(font=font1) main.config()
main.mainloop()
```

CHAPTER-07

IMPLEMENTATION

7.1 MODULES:

Module Details:

1.Upload Social Network Profiles Dataset:

Using this module we will upload dataset to application

2.Preprocess Dataset:

Using this module we will apply processing technique such as removing missing values and then split dataset into train and test where application use 80% dataset to train ANN and 20% dataset to test ANN prediction accuracy

3.Run ANN Algorithm:

Using this module we will train ANN algorithm with train and test data and then train model will be generated and we can use this train model to predict fake accounts from new dataset.

4.ANN Accuracy & Loss Graph:

To train ANN model we are taking 200 epoch/iterations and then in graph we will plot accuracy/loss performance of ANN at each epoch/iteration.

5.Predict Fake/Genuine Profile using ANN: using this module we will upload new test data and then apply ANN train model to predict whether test data is genuine or fake.

7.2 PYTHON

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 –

- Machine Learning
- 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 :-

1. Extensive Libraries

Python downloads with an extensive library and it contain 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. Also, the fact that you need to write less and get more things done.

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 World'. But in Python, just a print statement will do. It is also quite easy to learn**,** understand**,** and code**.** This is why when 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.
Advantages of Python Over Other Languages

1. Less Coding

Almost all of the tasks done in Python requires 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.

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.

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 Venners[1], 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."

7.3 GUI:

Python provides the standard library Tkinter for creating the graphical user interface for desktop based applications. Developing desktop based applications with python Tkinter is not a complex task. An empty Tkinter top-level window can be created by using the following steps.

1. import the Tkinter module.
2. Create the main application window.
3. Add the widgets like labels, buttons, frames, etc. to the window.
4. Call the main event loop so that the actions can take place on the user's computer screen.

The first thing you need to do is import the Python GUI Tkinter module:

```
>>> import tkinter as tk
```

A **window** is an instance of Tkinter’s Tk class. Go ahead and create a new window and assign it to the variable window:

```
>>> window = tk.Tk()
```

When you execute the above code, a new window pops up on your screen. How it looks depends on your operating system:

Adding a Widget

Now that you have a window, you can add a widget. Use the tk.Label class to add some text to a window. Create a Label widget with the text "Hello, Tkinter" and assign it to a variable called greeting:

```
>>> greeting = tk.Label(text="Hello, Tkinter")
```

The window you created earlier doesn't change. You just created a Label widget, but you haven't added it to the window yet. There are several ways to add widgets to a window. Right now, you can use the Label widget's .pack() method:

```
>>> greeting.pack()
```

When you pack a widget into a window, Tkinter sizes the window as small as it can be while still fully encompassing the widget. Now execute the following:

Displaying Text and Images With Label Widgets

Label widgets are used to display **text** or **images**. The text displayed by a Label widget can't be edited by the user. It's for display purposes only. As you saw in the example at the beginning of this tutorial, you can create a Label widget by instantiating the Label class and passing a string to the text parameter: label = tk.Label(text="Hello, Tkinter")

Label widgets display text with the default system text color and the default system text background color. These are typically black and white, respectively, but you may see different colors if you've changed these settings in your operating system.

You can control Label text and background colors using the foreground and background parameters:

```
label = tk.Label(     text="Hello, Tkinter",
foreground="white", # Set the text color to white
background="black" # Set the background color to black
)
```

There are numerous valid color names, including:

- "red"
- "orange"
- "yellow"
- "green"
- "blue"
- "purple"

```
label = tk.Label(text="Hello, Tkinter", background="#34A2FE")
```

This sets the label background to a nice, light blue color. Hexadecimal RGB values are more cryptic than named colors, but they're also more flexible. Fortunately, there are tools available that make getting hexadecimal color codes relatively painless.

If you don't feel like typing out foreground and background all the time, then you can use the shorthand fg and bg parameters to set the foreground and background colors:

```
label = tk.Label(text="Hello, Tkinter", fg="white", bg="black")
```

You can also control the width and height of a label with the width and height parameters:

```
label = tk.Label(
text="Hello, Tkinter",
fg="white",
bg="black",
width=10,     height=10
)
```

There are many similarities between Button and Label widgets. In many ways, a button is just a label that you can click! The same keyword arguments that you use to create and style a Label will work with Button widgets. For example, the following code creates a button with a blue background and yellow text. It also sets the width and height to 25 and 5 text units, respectively:

```
button = tk.Button(
text="Click me!",
width=25,
height=5,
bg="blue",
fg="yellow",)
```

When you need to get a little bit of text from a user, like a name or an email address, use an Entry widget. It'll display a **small text box** that the user can type some text into. Creating and styling an Entry widget works pretty much exactly like with Label and Button widgets. For example, the following code creates a widget with a blue background, some yellow text, and a width of 50 text units: entry = tk.Entry(fg="yellow", bg="blue", width=50)

The interesting bit about Entry widgets isn't how to style them, though. It's how to use them to get **input from a user**. There are three main operations that you can perform with Entry widgets:

1. Retrieving text with.get ()
2. Deleting text with .delete()
3. Inserting text with .insert()

The best way to get an understanding of Entry widgets is to create one and interact with it. Open up a Python shell and follow along with the examples in this section. First, import tkinter and create a new window:

```
>>> import tkinter as tk
>>> window = tk.Tk()
```

Now create a Label and an Entry widget:

```
>>> label = tk.Label(text="Name")
>>> entry = tk.Entry()
```

The Label describes what sort of text should go in the Entry widget. It doesn't enforce any sort of requirements on the Entry, but it tells the user what your program expects them to put there. You need to .pack() the widgets into the window so that they're visible:

```
>>> label.pack()
>>> entry.pack()
```

What is Machine Learning: -

Before we take a look at the details of various machine learning methods, let's start by looking at what machine learning is, and what it isn't. Machine learning is often categorized as a subfield of artificial intelligence, but I find that categorization can often be misleading at first brush. The study of machine learning certainly arose from research in this context, but in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of building models of data.

Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models tunable parameters that can be adapted to observed data; in this way the program can be considered to be "learning" from the data. Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. I'll leave to the reader the more philosophical digression regarding the extent to which this type of mathematical, model-based "learning" is similar to the "learning" exhibited by the human brain. Understanding the problem setting in machine learning is essential to using these tools effectively, and so we will start with some broad categorizations of the types of approaches we'll discuss here.

Categories Of Machine Learning:-

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised learning. Supervised learning involves somehow modeling the relationship between measured features of data and some label associated with the data; once this model is determined, it can be used to apply labels to new, unknown data. This is further subdivided into classification tasks and regression tasks: in classification, the labels are discrete categories, while in regression, the labels are continuous quantities. We will see examples of both types of supervised learning in the following section. Unsupervised learning involves modeling the features of a dataset without reference to any label, and is often described as "letting the dataset speak for itself." These models include tasks such as clustering *and* dimensionality reduction. Clustering algorithms identify distinct groups of data, while dimensionality reduction algorithms search for more succinct representations of the data. We will see examples of both types of unsupervised learning in the following section.

Need for Machine Learning

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate and solve complex problems. On the other side, AI is still in its initial stage and haven't surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, "to make decisions, based on data, with efficiency and scale".

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programing logic, in the problems that cannot be programmed inherently. The fact is that we can't do without human intelligence, but other aspect is that we all need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

Challenges in Machines Learning:-

While Machine Learning is rapidly evolving, making significant strides with cybersecurity and autonomous cars, this segment of AI as whole still has a long way to go. The reason behind is that ML has not been able to overcome number of challenges. The challenges that ML is facing currently are −

Quality of data − Having good-quality data for ML algorithms is one of the biggest challenges. Use of low-quality data leads to the problems related to data preprocessing and feature extraction.

Time-Consuming task − Another challenge faced by ML models is the consumption of time especially for data acquisition, feature extraction and retrieval.

Lack of specialist persons − As ML technology is still in its infancy stage, availability of expert resources is a tough job.

No clear objective for formulating business problems − Having no clear objective and welldefined goal for business problems is another key challenge for ML because this technology is not that mature yet.

Issue of overfitting & underfitting − If the model is overfitting or underfitting, it cannot be represented well for the problem.

Curse of dimensionality − Another challenge ML model faces is too many features of data points. This can be a real hindrance.

Difficulty in deployment − Complexity of the ML model makes it quite difficult to be deployed in real life.

Applications of Machines Learning:-

Machine Learning is the most rapidly growing technology and according to researchers we are in the golden year of AI and ML. It is used to solve many real-world complex problems which cannot be solved with traditional approach. Following are some real-world applications of ML

- Emotion analysis
- Sentiment analysis
- Error detection and prevention
- Weather forecasting and prediction
- Stock market analysis and forecasting
- Speech synthesis
- Speech recognition
- Customer segmentation
- Object recognition
- Fraud detection
- Fraud prevention
- Recommendation of products to customer in online shopping How to start learning ML?

Step 1 – Understand the Prerequisites

In case you are a genius, you could start ML directly but normally, there are some prerequisites that you need to know which include Linear Algebra, Multivariate Calculus, Statistics, and Python. And if you don't know these, never fear! You don't need a Ph.D. degree in these topics to get started but you do need a basic understanding.
(a) Learn Linear Algebra and Multivariate Calculus

Both Linear Algebra and Multivariate Calculus are important in Machine Learning. However, the extent to which you need them depends on your role as a data scientist. If you are more focused on application heavy machine learning, then you will not be that heavily focused on maths as there are many common libraries available. But if you want to focus on R&D in Machine Learning, then mastery of Linear Algebra and Multivariate Calculus is very important as you will have to implement many ML algorithms from scratch.

(b) Learn Statistics

Data plays a huge role in Machine Learning. In fact, around 80% of your time as an ML expert will be spent collecting and cleaning data. And statistics is a field that handles the collection, analysis, and presentation of data. So it is no surprise that you need to learn it!!! Some of the key concepts in statistics that are important are Statistical Significance, Probability Distributions, Hypothesis Testing, Regression, etc. Also, Bayesian Thinking is also a very important part of ML which deals with various concepts like Conditional Probability, Priors, and Posteriors, Maximum Likelihood, etc.

(c) Learn Python

Some people prefer to skip Linear Algebra, Multivariate Calculus and Statistics and learn them as they go along with trial and error. But the one thing that you absolutely cannot skip is Python! While there are other languages you can use for Machine Learning like R, Scala, etc. Python is currently the most popular language for ML. In fact, there are many Python libraries that are specifically useful for Artificial Intelligence and Machine Learning such as Keras, TensorFlow, Scikit-learn, etc.

Step 2 – Learn Various ML Concepts

Now that you are done with the prerequisites, you can move on to actually learning ML (Which is the fun part!!!) It’s best to start with the basics and then move on to the more complicated stuff. Some of the basic concepts in ML are:

(a) Terminologies of Machine Learning

- **Model –** A model is a specific representation learned from data by applying some machine learning algorithm. A model is also called a hypothesis.

- **Feature –** A feature is an individual measurable property of the data. A set of numeric features can be conveniently described by a feature vector. Feature vectors are fed as input to the model. For example, in order to predict a fruit, there may be features like color, smell, taste, etc.
- **Target (Label) –** A target variable or label is the value to be predicted by our model. For the fruit example discussed in the feature section, the label with each set of input would be the name of the fruit like apple, orange, banana, etc.
- **Training –** The idea is to give a set of inputs(features) and it's expected outputs(labels), so after training, we will have a model (hypothesis) that will then map new data to one of the categories trained on.
- **Prediction –** Once our model is ready, it can be fed a set of inputs to which it will provide a predicted output(label).

(b) Types of Machine Learning

- **Supervised Learning –** This involves learning from a training dataset with labeled data using classification and regression models. This learning process continues until the required level of performance is achieved.
- **Unsupervised Learning –** This involves using unlabelled data and then finding the underlying structure in the data in order to learn more and more about the data itself using factor and cluster analysis models.
- **Semi-supervised Learning –** This involves using unlabelled data like Unsupervised Learning with a small amount of labeled data. Using labeled data vastly increases the learning accuracy and is also more cost-effective than Supervised Learning.
- **Reinforcement Learning –** This involves learning optimal actions through trial and error. So the next action is decided by learning behaviors that are based on the current state and that will maximize the reward in the future.

Advantages of Machine learning :-

1. Easily identifies trends and patterns -

Machine Learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans. For instance, for an e-commerce website like Amazon, it serves to understand the browsing behaviors and purchase histories of its users to help cater to the right products, deals, and reminders relevant to them. It uses the results to reveal relevant advertisements to them.

2. No human intervention needed (automation)

With ML, you don't need to babysit your project every step of the way. Since it means giving machines the ability to learn, it lets them make predictions and also improve the algorithms on their own. A common example of this is anti-virus softwares; they learn to filter new threats as they are recognized. ML is also good at recognizing spam.

3. Continuous Improvement

As ML algorithms gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. Say you need to make a weather forecast model. As the amount of data you have keeps growing, your algorithms learn to make more accurate predictions faster.

4. Handling multi-dimensional and multi-variety data

Machine Learning algorithms are good at handling data that are multi-dimensional and multivariety, and they can do this in dynamic or uncertain environments.

5. Wide Applications

You could be an e-tailer or a healthcare provider and make ML work for you. Where it does apply, it holds the capability to help deliver a much more personal experience to customers while also targeting the right customers.

Purpose:-

We demonstrated that our approach enables successful segmentation of intra-retinal layers—even with low-quality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

Tensor flow

TensorFlow is a 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.

Scikit – learn

Scikit-learn provides 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.

7.4 ALGORIHTMS:

ARTIFICIAL NEURAL NETWORK:

The term "Artificial neural network" refers to a biologically inspired sub-field of artificial intelligence modeled after the brain. An Artificial neural network is usually a computational network based on biological neural networks that construct the structure of the human brain. Similar to a human brain has neurons interconnected to each other, artificial neural networks also have neurons that are linked to each other in various layers of the networks. These neurons are known as nodes. Artificial neural network tutorial covers all the aspects related to the artificial neural network. To understand the concept of the architecture of an artificial neural network, we have to understand what a neural network consists of. In order to define a neural network that consists of a large number of artificial neurons, which are termed units arranged in a sequence of layers. Lets us look at various types of layers available in an artificial neural network.

Artificial Neural Network primarily consists of three layers:

Input Layer:

As the name suggests, it accepts inputs in several different formats provided by the programmer.

Hidden Layer:

The hidden layer presents in-between input and output layers. It performs all the calculations to find hidden features and patterns.

Output Layer: The input goes through a series of transformations using the hidden layer, which finally results in output that is conveyed using this layer. The artificial neural network takes input and computes the weighted sum of the inputs and includes a bias. This computation is represented in the form of a transfer function. It determines weighted total is passed as an input to an activation function to produce the output. Activation functions choose whether a node should fire or not. Only those who are fired make it to the output layer. There are distinctive activation functions available that can be applied upon the sort of task we are performing.

ADVANTAGES:

Parallel processing capability:

Artificial neural networks have a numerical value that can perform more than one task simultaneously.

Storing data on the entire network:

Data that is used in traditional programming is stored on the whole network, not on a database. The disappearance of a couple of pieces of data in one place doesn't prevent the network from working.

Capability to work with incomplete knowledge:

After ANN training, the information may produce output even with inadequate data. The loss of performance here relies upon the significance of missing data.

Having a memory distribution:

For ANN is to be able to adapt, it is important to determine the examples and to encourage the network according to the desired output by demonstrating these examples to the network. The succession of the network is directly proportional to the chosen instances, and if the event can't appear to the network in all its aspects, it can produce false output.

Having fault tolerance:

Extortion of one or more cells of ANN does not prohibit it from generating output, and this feature makes the network fault-tolerance.

Working of ANN Algorithm:

Artificial Neural Network can be best represented as a weighted directed graph, where the artificial neurons form the nodes. The association between the neurons outputs and neuron inputs can be viewed as the directed edges with weights. The Artificial Neural Network receives the input signal from the external source in the form of a pattern and image in the form of a vector. These inputs are then mathematically assigned by the notations x(n) for every n number of inputs.

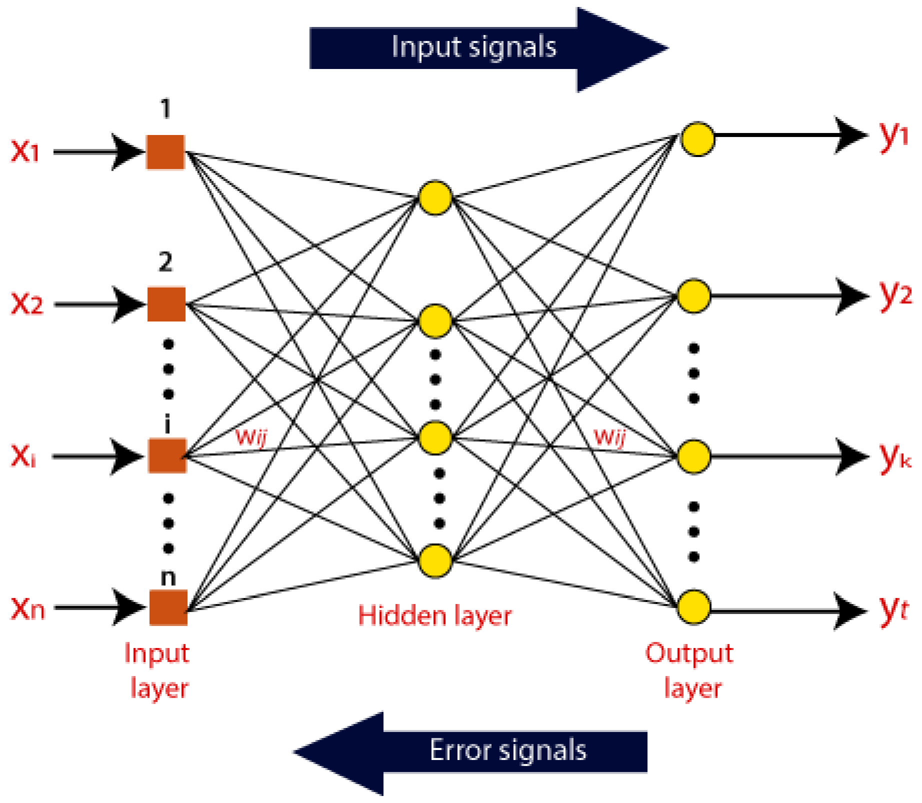


Figure:6.3.1 ANN working

Afterward, each of the input is multiplied by its corresponding weights (these weights are the details utilized by the artificial neural networks to solve a specific problem). In general terms, these weights normally represent the strength of the interconnection between neurons inside the artificial neural network. All the weighted inputs are summarized inside the computing unit.

If the weighted sum is equal to zero, then bias is added to make the output non-zero or something else to scale up to the system's response. Bias has the same input, and weight equals to 1. Here the total of weighted inputs can be in the range of 0 to positive infinity. Here, to keep the response in the limits of the desired value, a certain maximum value is benchmarked, and the total of weighted inputs is passed through the activation function.

The activation function refers to the set of transfer functions used to achieve the desired output. There is a different kind of the activation function, but primarily either linear or non-linear sets of functions. Some of the commonly used sets of activation functions are the Binary, linear, and Tan hyperbolic sigmoidal activation functions.

Types of Artificial Neural Network:

There are various types of Artificial Neural Networks (ANN) depending upon the human brain neuron and network functions, an artificial neural network similarly performs tasks. The majority of the artificial neural networks will have some similarities with a more complex biological partner and are very effective at their expected tasks. For example, segmentation or classification.

Feedback ANN:

In this type of ANN, the output returns into the network to accomplish the best-evolved results internally. As per the **University of Massachusetts**, Lowell Centre for Atmospheric Research. The feedback networks feed information back into itself and are well suited to solve optimization issues. The Internal system error corrections utilize feedback ANNs.

Feed-Forward ANN:

A feed-forward network is a basic neural network comprising of an input layer, an output layer, and at least one layer of a neuron. Through assessment of its output by reviewing its input, the intensity of the network can be noticed based on group behavior of the associated neurons, and the output is decided. The primary advantage of this network is that it figures out how to evaluate and recognize input patterns.

CHAPTER-08

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

1.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.

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.

2.Integration testing

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

3.Functional test

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.

4.System Test

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

5.White Box Testing

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

6.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.

OTHER TESTING METHODOLOGIES

1.User Acceptance Testing

User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

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

2.Output Testing

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the system under consideration. Hence the output format is considered in 2 ways – one is on screen and another in printed format.

3.Validation Checking

Validation checks are performed on the following fields.

Text Field: The text field can contain only the number of characters lesser than or equal to its size. The text fields are alphanumeric in some tables and alphabetic in other tables. Incorrect entry always flashes and error message.

Numeric Field: The numeric field can contain only numbers from 0 to 9. An entry of any character flashes an error messages. The individual modules are checked for accuracy and what it has to perform. Each module is subjected to test run along with sample data. The individually tested modules are integrated into a single system. Testing involves executing the real data information is used in the program the existence of any program defect is inferred from the output. The testing should be planned so that all the requirements are individually tested. A successful test is one that gives out the defects for the inappropriate data and produces and output revealing the errors in the system.

4.Preparation of Test Data

Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using that test data. While testing the system by using test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

5.Using Live Test Data:

Live test data are those that are actually extracted from organization files. After a system is partially constructed, programmers or analysts often ask users to key in a set of data from their normal activities. Then, the systems person uses this data as a way to partially test the system. In other instances, programmers or analysts extract a set of live data from the files and have them entered themselves.

It is difficult to obtain live data in sufficient amounts to conduct extensive testing. And, although it is realistic data that will show how the system will perform for the typical processing requirement, assuming that the live data entered are in fact typical, such data generally will not test all combinations or formats that can enter the system. This bias toward typical values then does not provide a true systems test and in fact ignores the cases most likely to cause system failure.

6.Using Artificial Test Data:

Artificial test data are created solely for test purposes, since they can be generated to test all combinations of formats and values. In other words, the artificial data, which can quickly be prepared by a data generating utility program in the information systems department, make possible the testing of all login and control paths through the program.

USER TRAINING

Whenever a new system is developed, user training is required to educate them about the working of the system so that it can be put to efficient use by those for whom the system has been primarily designed. For this purpose the normal working of the project was demonstrated to the prospective users. Its working is easily understandable and since the expected users are people who have good knowledge of computers, the use of this system is very easy.

MAINTAINENCE

This covers a wide range of activities including correcting code and design errors. To reduce the need for maintenance in the long run, we have more accurately defined the user’s requirements during the process of system development. Depending on the requirements, this system has been developed to satisfy the needs to the largest possible extent. With development in technology, it may be possible to add many more features based on the requirements in future. The coding and designing is simple and easy to understand which will make maintenance easier.

CHAPTER-09

OUTPUT SCREEN

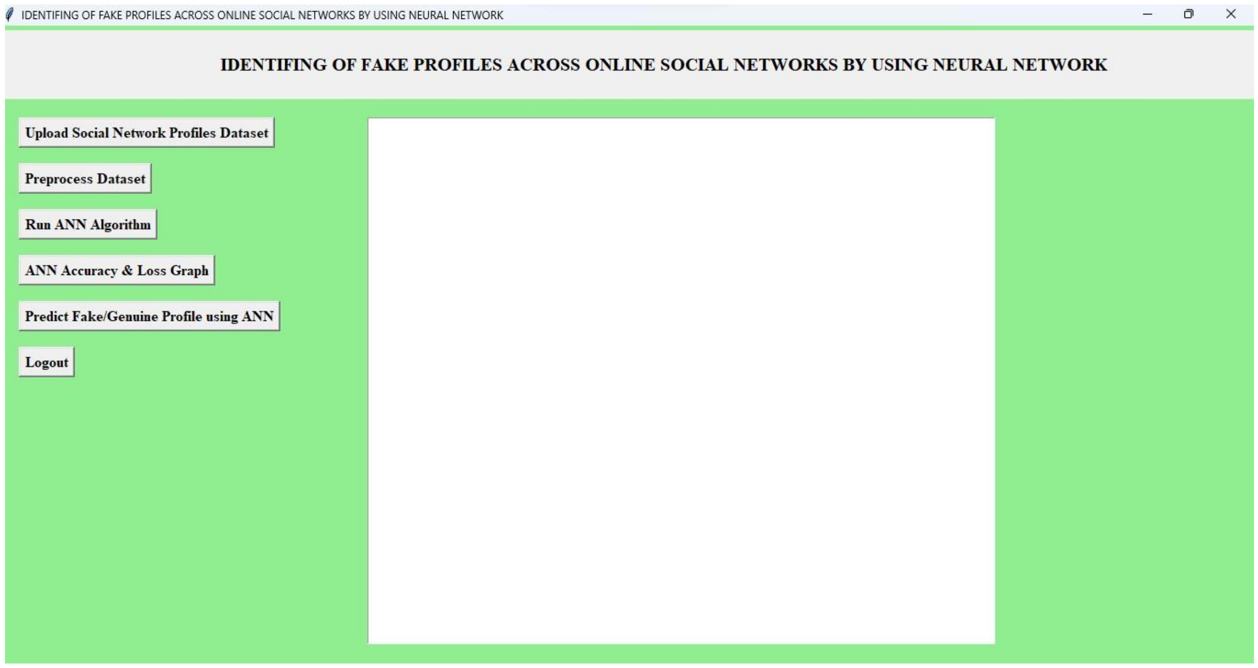
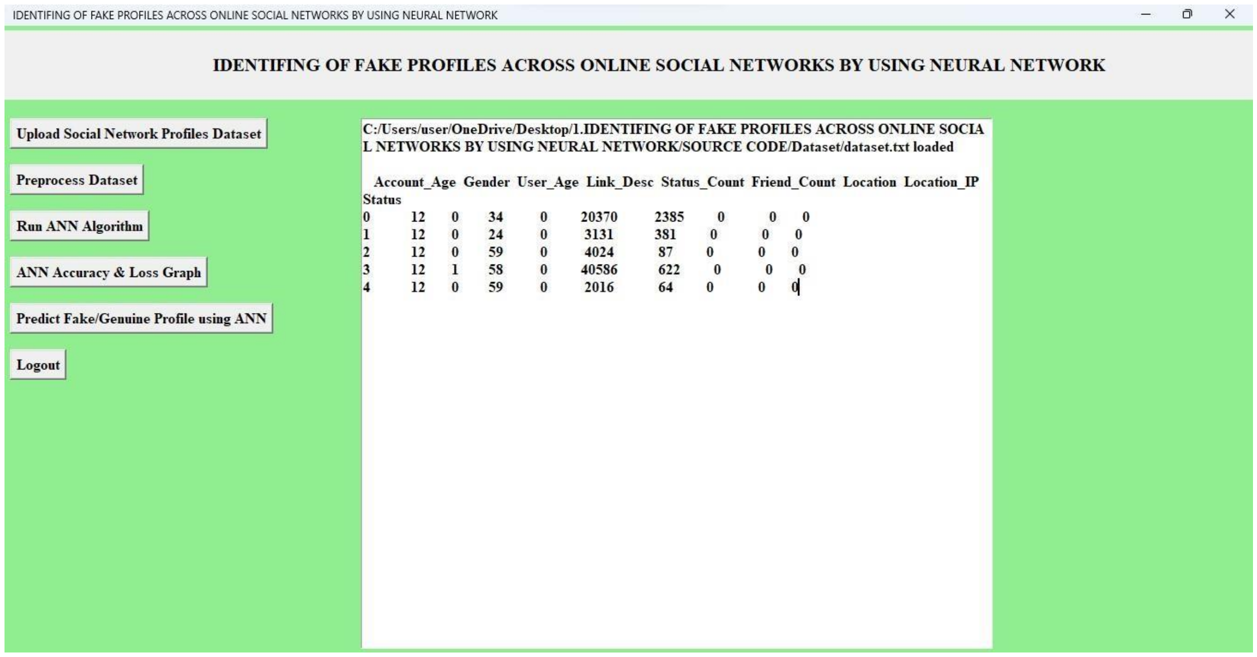
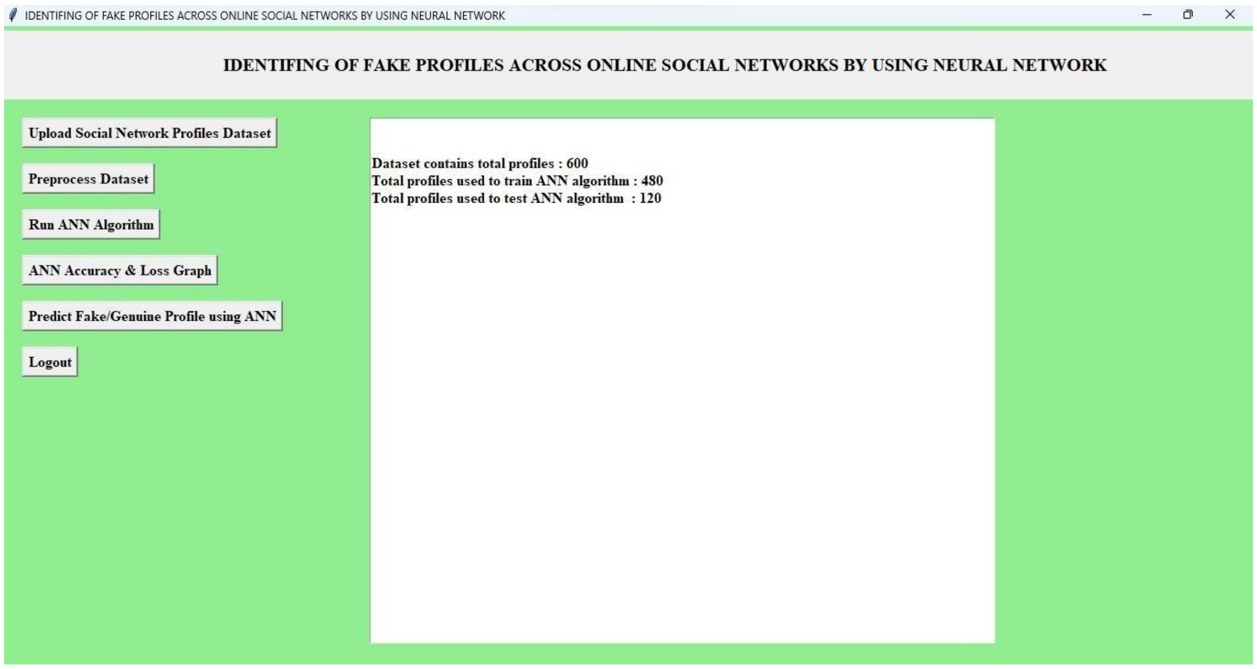

IDENTIFING OF FAKE PROFILES ACROSS ONLINE SOCIAL NETWORKS BY USING NEURAL NETWORK
IDENTIFING OF FAKE PROFILES ACROSS ONLINE SOCIAL NETWORKS BY USING NEURAL NETWORK
Upload Social Network Profiles Dataset
Preprocess Dataset
Run ANN Algorithm
ANN Accuracy & Loss Graph
Predict Fake/Genuine Profile using ANN
Logout

Figure:9.1 Upload dataset


IDENTIFING OF FAKE PROFILES ACROSS ONLINE SOCIAL NETWORKS BY USING NEURAL NETWORK
IDENTIFING OF FAKE PROFILES ACROSS ONLINE SOCIAL NETWORKS BY USING NEURAL NETWORK
Upload Social Network Profiles Dataset
Preprocess Dataset
Run ANN Algorithm
ANN Accuracy & Loss Graph
Predict Fake/Genuine Profile using ANN
Logout

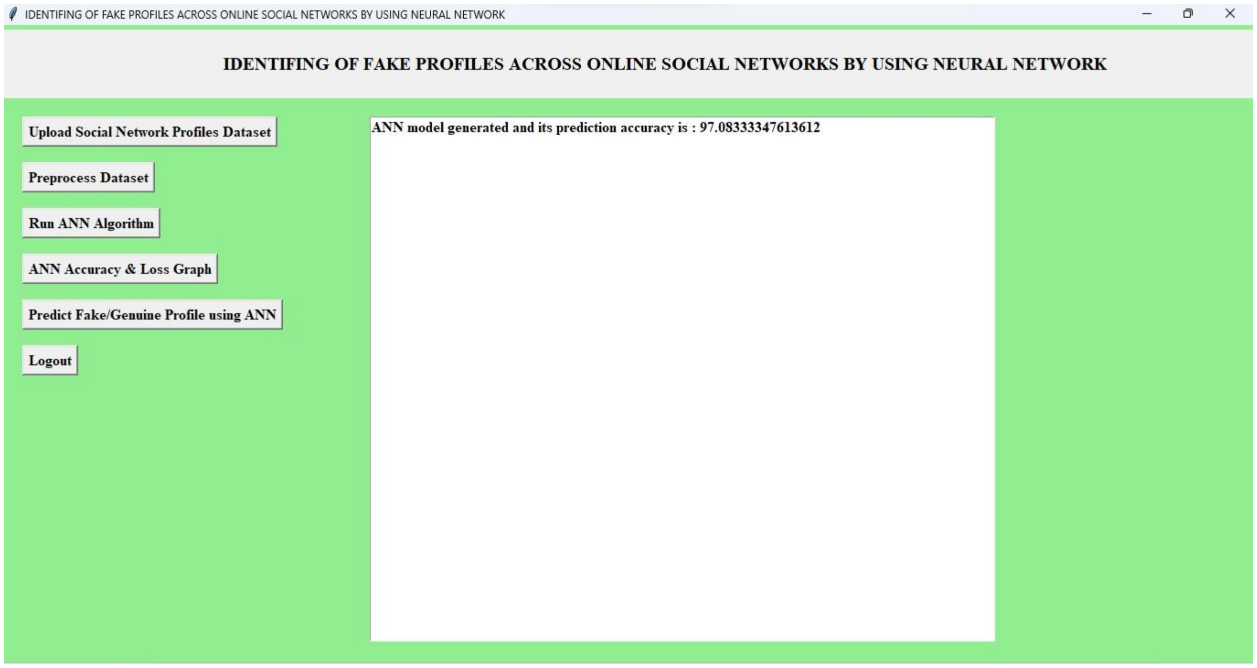
```
C:/Users/user/OneDrive/Desktop/1.IDENTIFING OF FAKE PROFILES ACROSS ONLINE SOCIA
L NETWORKS BY USING NEURAL NETWORK/SOURCE CODE/Dataset/dataset.txt loaded

  Account_Age  Gender  User_Age  Link_Desc  Status_Count  Friend_Count  Location  Location_IP
Status
0        12      0       34        0       20370        2385        0          0     0
1        12      0       24        0        3131         381        0          0     0
2        12      0       59        0        4024          87        0          0     0
3        12      1       58        0       40586         622        0          0     0
4        12      0       59        0        2016          64        0          0     0
```

Figure:9.2 Preprocess data


IDENTIFING OF FAKE PROFILES ACROSS ONLINE SOCIAL NETWORKS BY USING NEURAL NETWORK
IDENTIFING OF FAKE PROFILES ACROSS ONLINE SOCIAL NETWORKS BY USING NEURAL NETWORK
Upload Social Network Profiles Dataset
Preprocess Dataset
Run ANN Algorithm
ANN Accuracy & Loss Graph
Predict Fake/Genuine Profile using ANN
Logout

```
Dataset contains total profiles : 600
Total profiles used to train ANN algorithm : 480
Total profiles used to test ANN algorithm  : 120
```

Figure:9.3 Run ANN algorithm


IDENTIFING OF FAKE PROFILES ACROSS ONLINE SOCIAL NETWORKS BY USING NEURAL NETWORK
IDENTIFING OF FAKE PROFILES ACROSS ONLINE SOCIAL NETWORKS BY USING NEURAL NETWORK
Upload Social Network Profiles Dataset
Preprocess Dataset
Run ANN Algorithm
ANN Accuracy & Loss Graph
Predict Fake/Genuine Profile using ANN
Logout

```
ANN model generated and its prediction accuracy is : 97.08333347613612
```

Figure:9.4 ANN Accuracy

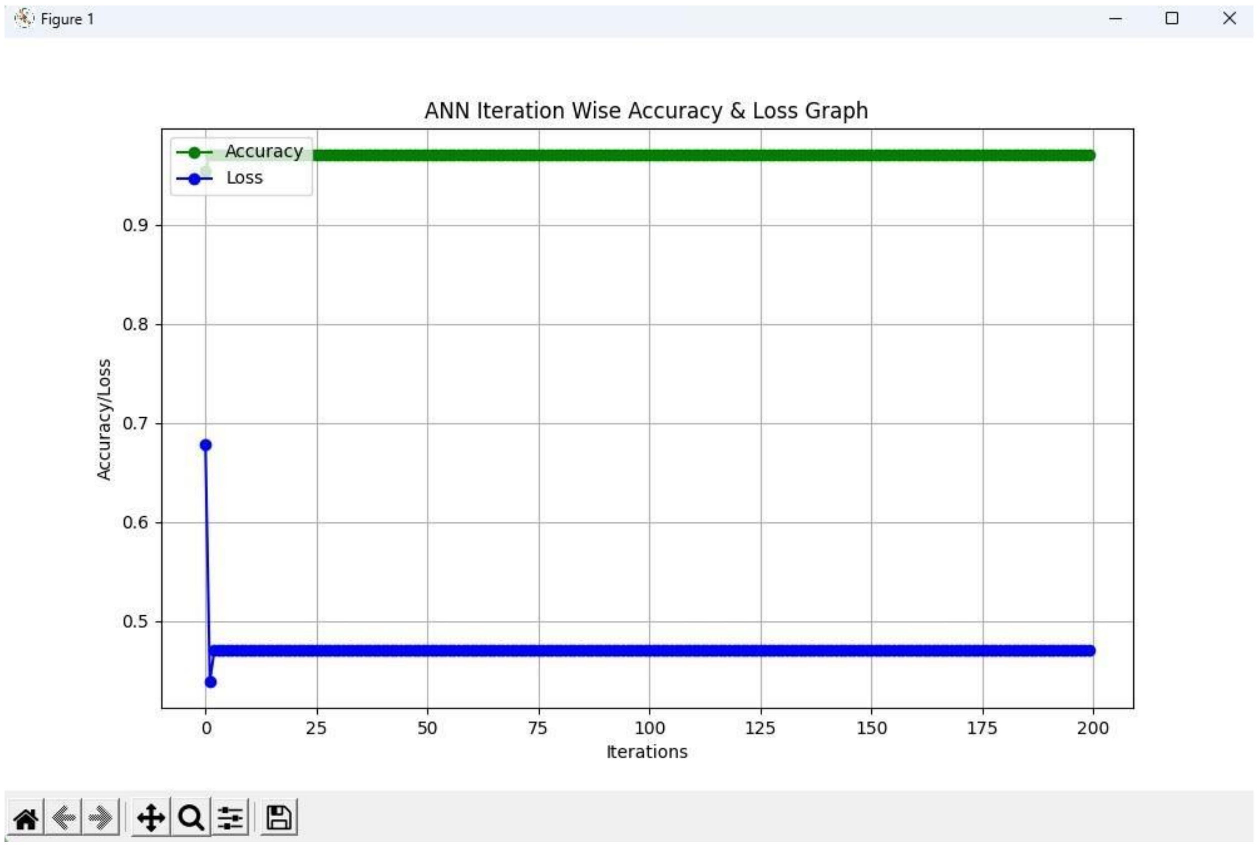
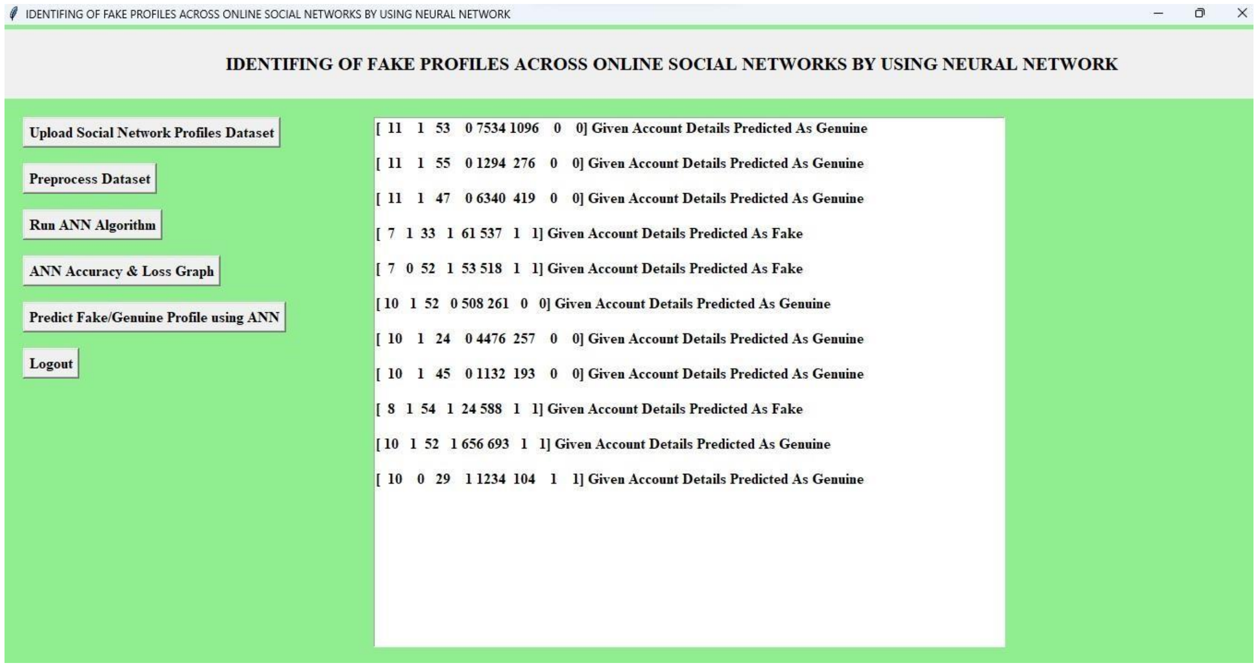


Figure:9.5 Accuracy and loss Graph


IDENTIFING OF FAKE PROFILES ACROSS ONLINE SOCIAL NETWORKS BY USING NEURAL NETWORK
Upload Social Network Profiles Dataset
Preprocess Dataset
Run ANN Algorithm
ANN Accuracy & Loss Graph
Predict Fake/Genuine Profile using ANN
Logout

```
[ 11   1  53   0 7534 1096   0   0] Given Account Details Predicted As Genuine
[ 11   1  55   0 1294  276   0   0] Given Account Details Predicted As Genuine
[ 11   1  47   0 6340  419   0   0] Given Account Details Predicted As Genuine
[  7   1  33   1 61 537   1   1] Given Account Details Predicted As Fake
[  7   0  52   1 53 518   1   1] Given Account Details Predicted As Fake
[10   1  52   0 508 261   0   0] Given Account Details Predicted As Genuine
[ 10   1  24   0 4476  257   0   0] Given Account Details Predicted As Genuine
[ 10   1  45   0 1132  193   0   0] Given Account Details Predicted As Genuine
[  8   1  54   1 24 588   1   1] Given Account Details Predicted As Fake
[10   1  52   1 656 693   1   1] Given Account Details Predicted As Genuine
[ 10   0  29   1 1234  104   1   1] Given Account Details Predicted As Genuine
```

Figure:9.6 Predict Fake Genuine profile

CHAPTER-10
CONCLUSION

False identities in the form of compromised or fake email accounts, accounts in social media, fake or cracked websites, fake domain names, and malicious Tor nodes, are heavily used in APT attacks, especially in their initial phases, and in other malicious activities. Using these fake identities, the attacker(s) aim at establishing trust with the target and at crafting and mounting a spear phishing or another attack. Based on research evidence, information gathering for a spear phishing attack heavily relies on the use of social media and fake accounts therein. It is therefore important to detect, as early as possible, the presence of a fake social media account. A number of recent research works have focused on detecting such fake accounts, either by analyzing the characteristics of individual profiles and their connections, or in case of coordinated activities, by multiple fake social media accounts, such as in the case of crowdturfing – by analyzing the commonality of these activities, too.

CHAPTER-11
FUTURE ENHANCEMENT

We have used a small dataset to train the classifier, in future we can implement it on a large dataset and, we can implement it on mobile applications and desktops separately, so users can use this system on a mobile phone more efficiently for detecting fake profile. The accuracy of the proposed technique can also be improved using different feature selection techniques. Better outcomes could also occur from the addition of new parameters, the fusion of different models, and the creation of a real-time model. Depending on their size or specific significance in the identification process, the areas in the model and data may be given varying degrees of prominence.

CHAPTER-12

REFERENCES

1. Fake Social Media Profile Detection, Joshi, U.D., Singh, A.P., Pahuja, T.R., Naval, S., and Singal, G., 2021.Machine Learning Algorithms and Applications, edited by Srinivas, Sucharitha, G., Matta, and P., Scrivener Publishing LLC, Beverly,

2. Evaluation of Eye-ball Movement and Head Movement Detection Based on Reading, Sayeed, S., Sultana, F., Chakraborty, P., and Yousuf, M.A. 2021. [11].

3. Awasthi, S., Shanmugam, R., Jena, S.R. and Srivastava, A., 2020. Review of Techniques to Prevent Fake Accounts on SocialMedia.

4. Wanda, P. and Jie, H.J., 2020. Deep Profile: Finding fake profile in online social network using dynamic CNN. *Journal of Information Security and Applications*, *52*, p.102465.

5. Zhang, J., Dong, B. and Philip, S.Y., 2020, April. Fake detector: Effective fake news detection with deep diffusive neural network. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)* (pp. 1826-1829). IEEE.

6. Jie, H.J., and Wanda, P. (2020) DeepProfile: Utilizing Dynamic CNN to Detect Fake Profiles in Internet Social Networks.52, Article ID 102465 in Journal of Information Security and Applications.

7. Awasthi, S., Shanmugam, R., Jena, S.R. and Srivastava, A., 2020. Review of Techniques to Prevent Fake Accounts on Social Media.

8. Zhang, J., Dong, B. and Philip, S.Y., 2020, April. Fakedetector: Effective fake news detection with deep diffusive neural network. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)* (pp. 1826-1829). IEEE.

9. Wanda, P. and Jie, H.J., 2020. DeepProfile: Finding fake profile in online social network using dynamic CNN. *Journal of Information Security and Applications*, *52*, p.102465.

10. Fake Profile Detection on Social Networking Websites: A Complete Study, Roy, P.K. and Chahar, S. 2020. pp. 271-285 of IEEE Transactions on Artificial Intelligence.

11. Utilizing face recognition, Chakraborty, P., Muzammel, C.S., Khatun, M., Islam, S.F., and Rahman, S. (2020) developed an automatic student attendance system. IJEAT, 9, 93–99. https://doi.org/10.35940/ijeat.B4207.029320 [10].

12. Hajdu, G., Minoso, Y., Lopez, R., Acosta, M. and Elleithy, A., 2019, May. Use of Artificial Neural Networks to Identify Fake Profiles. In *2019 IEEE Long Island Systems, Applications and Technology Conference (LISAT)* (pp. 1-4). IEEE.

13. Hajdu, G., Minoso, Y., Lopez, R., Acosta, M. and Elleithy, A., 2019, May. Use of Artificial Neural Networks to Identify Fake Profiles. In *2019 IEEE Long Island Systems, Applications and Technology Conference (LISAT)* (pp. 1-4). IEEE.

14. Kaur, J. and Sabharwal, M., 2018. Spam detection in online social networks using feed forward neural network. In *RSRI conference on recent trends in science and engineering* (Vol. 2, pp. 69-78).

15. Khaled, S., El-Tazi, N. and Mokhtar, H.M., 2018, December. Detecting fake accounts on social media. In *2018 IEEE International Conference on Big Data (Big Data)* (pp. 3672-3681). IEEE.

16. Ramalingam, D. and Chinnaiah, V., 2018. Fake profile detection techniques in large-scale online social networks: A comprehensive review. *Computers & Electrical Engineering*, *65*, pp.165-177.

17. Using Machine Learning to Identify False Identities: Bots vs. Humans. Van Der Walt, E. and Eloff, J. 2018.6540–6549 in IEEE Access. https://doi.org/10.1109/ACCESS.2018.2796018

18. Ramalingam, D. and Chinnaiah, V., 2018. Fake profile detection techniques in large-scale online social networks: A comprehensive review. *Computers & Electrical Engineering*, *65*, pp.165-177.

19. Ferrara and Kudugunta, S. (2018) Bot detection using deep neural networks. 312–322 in Information Sciences, 467. https://doi.org/10.1016/j.ins.2018.08.019 [3].

20. Fake Profile Detection Methods in Large-Scale Online Social Networks: A Complete Study, D. Ramalingam and V. Chinnaiah, 2018.165–177 in Computers & Electrical Engineering, vol. 65

21. Using a black-list, Swe, M.M., and Myo, N.N. (2018) detected fake accounts on Twitter. Singapore, 6–8 June 2018, IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS), 562–566.

22. Kaur, J. and Sabharwal, M., 2018. Spam detection in online social networks using feed forward neural network. In *RSRI conference on recent trends in science and engineering* (Vol. 2, pp. 69-78).

23. Khaled, S., El-Tazi, N. and Mokhtar, H.M., 2018, December. Detecting fake accounts on social media. In *2018 IEEE International Conference on Big Data (Big Data)* (pp. 3672-3681). IEEE.

24. Meligy, A.M., Ibrahim, H.M. and Torky, M.F., 2017. Identity verification mechanism for detecting fake profiles in online social networks. *Int. J. Comput. Netw. Inf. Secur.(IJCNIS)*, *9*(1), pp.31-39.

25. Meligy, A.M., Ibrahim, H.M. and Torky, M.F., 2017. Identity verification mechanism for detecting fake profiles in online social networks. *Int. J. Comput. Netw. Inf. Secur.(IJCNIS)*, *9*(1), pp.31-39.