

ADVANCING CYBERSECURITY USING AI
SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering Degree in
Computer Science and Engineering

By

PUVVADA GOMANTH (REG.NO 37110608)

R. SHANMUKH SAI (37110612)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING
SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI – 600119.

MARCH 2021



SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY

(Established under Section 3 of UGC Act, 1956)

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai -600 119

www.sathyabama.ac.in



SCHOOL OF COMPUTING

BONAFIDE CERTIFICATE

This is to certify that this Project report is the Bonafide work of **PUVVADA GOMANTH (Reg. No 37110608)**, **R. SHANMUKH SAI (37110612)** who carried out the project entitled **“ADVANCING CYBERSECURITY USING AI”** under our supervision from NOVEMBER 2020 to MARCH 2021.

Internal Guide Name

Dr. G. MATHIVANAN, M.E., Ph.D.,

Head of the Department

Dr. S. VIGNESHWARI, M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **P. GOMANTH (Reg. No. 37110608), R. SHANMUKH SAI (37110612)** hereby declare that the Project Report entitled “**ADVANCING CYBERSECURITY USING AI**” done by me under the guidance of **Dr. G. MATHIVANAN, M.E., Ph.D.**, is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of management of SATHYABAMA for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. SASIKALA, M.E., Ph.D., Dean, School of Computing** and **Dr. S. VIGNESHWARI, M.E., Ph.D., and Dr. L. LAKSHMAN, M.E., Ph.D., Heads of the Department, Department of Computer Science and Engineering** for providing me the necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my project guide **Dr. G. MATHIVANAN, M.E., Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department of Computer Science and Engineering who were helpful in many ways for the completion of the project.

Training Certificate



ABSTRACT

In this project, we have chosen to focus on predicting the accuracy of various modes of cyberattacks possible and that have occurred and used previously.

This project is about classifying a variety of random failures and cyber-attacks exist in cyber-physical systems, which greatly restrict their growth. Fortunately, an intrusion detection mechanism could take effect for protecting cyber-physical systems. When misbehaviour is found by the intrusion detector, the appropriate action can be taken immediately so that any harm to the system will be minimized.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
1	INTRODUCTION	1
	1.1 OVERVIEW	1
	1.2 SCOPE OF THE PROJECT	1
	1.3 DOMAIN OVERVIEW	1
	1.3.1 ARTIFICIAL INTELLIGENCE	2
2	LITERATURE SURVEY	5
	2.1 LITERATURE SURVEY	5
3	AIM AND SCOPE OF THE PRESENT INVESTIGATION	7
	3.1 AIM	7
	3.2 SCOPE	7
4	EXPERIMENTAL MATERIALS AND METHODS	8
	4.1 ALGORITHM	8
	4.1.1 PYTHON	8
	4.1.2 NUMPY	14
	4.3 ARCHITECTURE	16
5	RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS	23
	5.1 PERFORMANCE ANALYSIS	23
	5.2 DISCUSSION	23

	5.3 RESULTS	24
6	SUMMARY AND CONCLUSION	26
	6.1 SUMMARY	26
	6.2 CONCLUSION	26
	REFERENCES	26

LIST OF FIGURES

NAME	DESCRIPTION	PAGE NO
FIG 4.1	ARCHITECTURE DIAGRAM	17
FIG 5.1	MODULES	24
FIG 5.2	ATTACK TYPE IN DATASET	24
FIG 5.3	HEAT GRAPH IN DATASET	25
FIG 5.4	ACCURACY & LOSS	25

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

As Cyber-physical systems are yet to be defined universally, the application of the intrusion detection mechanism remain open presently. As a result, the effort will be made to discuss how to appropriately apply the intrusion detection mechanism to Cyber-physical systems in this paper. By examining the unique properties of Cyber-physical systems, it intends to define the specific requirements first. Then, the design outline of the intrusion detection mechanism in Cyber-physical systems is introduced in terms of the layers of system and specific detection techniques. Finally, some significant research problems are identified for enlightening the subsequent studies.

1.2 SCOPE OF PROJECT

The main contributions of this project therefore are:

- ▶ Data Analysis
- ▶ Dataset Preprocessing
- ▶ Training the Model
- ▶ Testing of Dataset

1.3 DOMAIN OVERVIEW

Artificial intelligence (AI) is wide-ranging branch of computer science concerned with building smart machines capable of performing tasks that typically require human intelligence. AI is an interdisciplinary science with multiple approaches, but advancements in machine learning and deep learning are creating a paradigm shift in virtually every sector of the tech industry.

Narrow AI: Sometimes referred to as "Weak AI," this kind of artificial intelligence operates within a limited context and is a simulation of human intelligence. Narrow AI is often focused on performing a single task extremely well and while these machines may seem intelligent, they are operating under far more constraints and limitations than even the most basic human intelligence.

Artificial General Intelligence (AGI): AGI, sometimes referred to as "Strong AI," is the kind of artificial intelligence we see in the movies, like the robots from Westworld or Data from Star

Trek: The Next Generation. AGI is a machine with general intelligence and, much like a human being, it can apply that intelligence to solve any problem.

1.3.1 ARTIFICIAL INTELLIGENCE

Artificial intelligence (AI) makes it possible for machines to learn from experience, adjust to new inputs and perform human-like tasks. Most AI examples that you hear about today – from chess-playing computers to self-driving cars – rely heavily on deep learning and natural language processing. Using these technologies, computers can be trained to accomplish specific tasks by processing large amounts of data and recognizing patterns in the data.

The AI is programmed to do something devastating: Autonomous weapons are artificial intelligence systems that are programmed to kill. In the hands of the wrong person, these weapons could easily cause mass casualties. Moreover, an AI arms race could inadvertently lead to an AI war that also results in mass casualties. To avoid being thwarted by the enemy, these weapons would be designed to be extremely difficult to simply “turn off,” so humans could plausibly lose control of such a situation. This risk is one that’s present even with narrow AI, but grows as levels of AI intelligence and autonomy increase.

The AI is programmed to do something beneficial, but it develops a destructive method for achieving its goal: This can happen whenever we fail to fully align the AI’s goals with ours, which is strikingly difficult. If you ask an obedient intelligent car to take you to the airport as fast as possible, it might get you there chased by helicopters and covered in vomit, doing not what you wanted but literally what you asked for. If a superintelligent system is tasked with a ambitious geoengineering project, it might wreak havoc with our ecosystem as a side effect, and view human attempts to stop it as a threat to be met.

AI applications can provide personalized medicine and X-ray readings. Personal health care assistants can act as life coaches, reminding you to take your pills, exercise or eat healthier.

AI can analyze factory IoT data as it streams from connected equipment to forecast expected load and demand using recurrent networks, a specific type of deep learning network used with sequence data.

AI provides virtual shopping capabilities that offer personalized recommendations and discuss purchase options with the consumer. Stock management and site layout technologies will also be improved with AI.

Artificial Intelligence enhances the speed, precision and effectiveness of human efforts. In financial institutions, AI techniques can be used to identify which transactions are likely to be fraudulent, adopt fast and accurate credit scoring, as well as automate manually intense data management tasks.

Artificial intelligence is going to change every industry, but we have to understand its limits. The principle limitation of AI is that it learns from the data. There is no other way in which knowledge can be incorporated. That means any inaccuracies in the data will be reflected in the results. And any additional layers of prediction or analysis have to be added separately. Today's AI systems are trained to do a clearly defined task. The system that plays poker cannot play solitaire or chess. The system that detects fraud cannot drive a car or give you legal advice. In fact, an AI system that detects health care fraud cannot accurately detect tax fraud or warranty claims fraud.

In other words, these systems are very, very specialized. They are focused on a single task and are far from behaving like humans.

Likewise, self-learning systems are not autonomous systems. The imagined AI technologies that you see in movies and TV are still science fiction. But computers that can probe complex data to learn and perfect specific tasks are becoming quite common.

AI automates repetitive learning and discovery through data. But AI is different from hardware-driven, robotic automation. Instead of automating manual tasks, AI performs frequent, high-volume, computerized tasks reliably and without fatigue. For this type of automation, human inquiry is still essential to set up the system and ask the right questions.

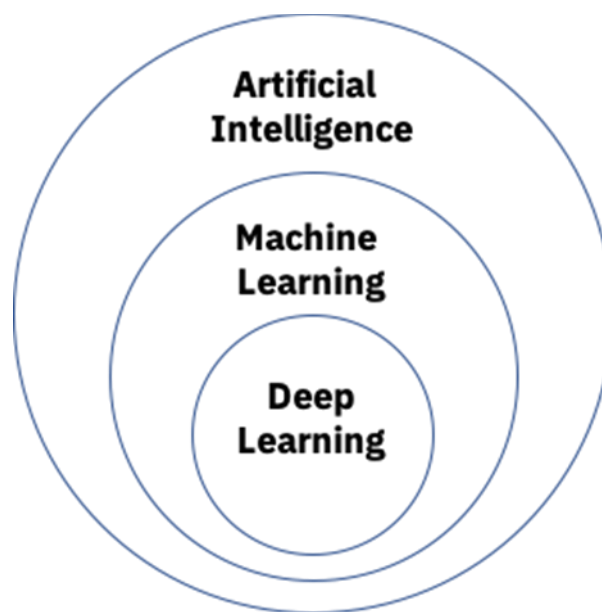
AI adds intelligence to existing products. In most cases, AI will not be sold as an individual application. Rather, products you already use will be improved with AI capabilities, much like Siri was added as a feature to a new generation of Apple products. Automation, conversational platforms, bots and smart machines can be combined with large amounts of data to improve many technologies at home and in the workplace, from security intelligence to investment analysis.

AI adapts through progressive learning algorithms to let the data do the programming. AI finds structure and regularities in data so that the algorithm acquires a skill: The algorithm becomes a classifier or a predictor. So, just as the algorithm can teach itself how to play chess, it can teach itself what product to recommend next online. And the models adapt when given new data. Back propagation is an AI technique that allows the model to adjust, through training and added data, when the first answer is not quite right.

Artificial intelligence algorithms are designed to make decisions, often using real-time data. They are unlike passive machines that are capable only of mechanical or predetermined responses. Using sensors, digital data, or remote inputs, they combine information from a variety of different sources, analyze the material instantly, and act on the insights derived from those data. With massive improvements in storage systems, processing speeds, and analytic techniques, they are capable of tremendous sophistication in analysis and decisionmaking.

AI systems have the ability to learn and adapt as they make decisions. In the transportation

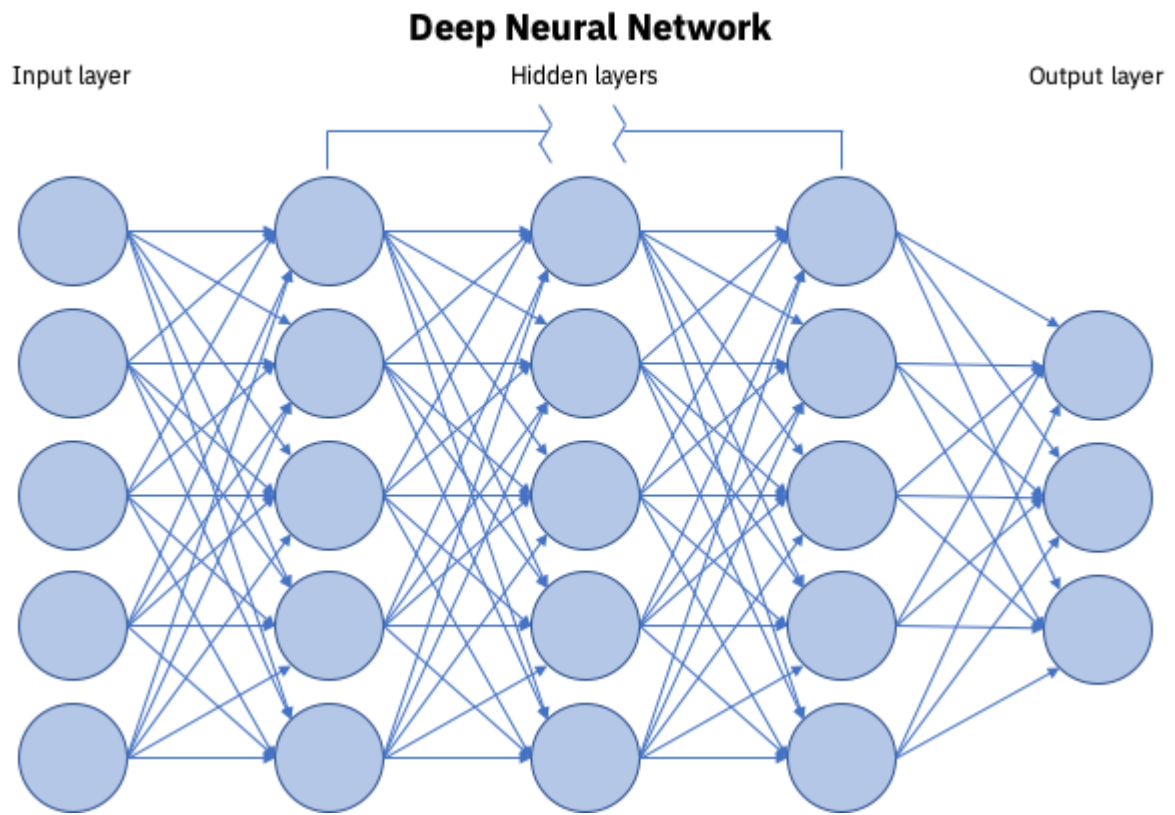
area, for example, semi-autonomous vehicles have tools that let drivers and vehicles know about upcoming congestion, potholes, highway construction, or other possible traffic impediments. Vehicles can take advantage of the experience of other vehicles on the road, without human involvement, and the entire corpus of their achieved “experience” is immediately and fully transferable to other similarly configured vehicles. Their advanced algorithms, sensors, and cameras incorporate experience in current operations, and use dashboards and visual displays to present information in real time so human drivers are able to make sense of ongoing traffic and vehicular conditions. And in the case of fully autonomous vehicles, advanced systems can completely control the car or truck, and make all the navigational decisions.



Think of artificial intelligence as the entire universe of computing technology that exhibits anything remotely resembling human intelligence. AI systems can include anything from an expert system—a problem-solving application that makes decisions based on complex rules or if/then logic—to something like the equivalent of the fictional Pixar character Wall-E, a computer that develops the intelligence, free will, and emotions of a human being.

Machine learning is a subset of AI application that learns by itself. It actually reprograms itself, as it digests more data, to perform the specific task it's designed to perform with increasingly greater accuracy.

Deep learning is a subset of machine learning application that teaches itself to perform a specific task with increasingly greater accuracy, without human intervention.



CHAPTER 2

LITERATURE SURVEY

2.1 LITERATURE SURVEY

Literature Survey 1	
Title	Network Intrusion Detection: Based on Deep Hierarchical Network and Original Flow Data
Authors	YONG ZHANG , XU CHEN , LEI JIN, XIAOJUAN WANG, AND DA GUO
Published Year	2019
Efficiency	✓ Integrates the improved LeNet-5 and LSTM neural network structures

	✓ Can train our proposed hierarchical network at the same time instead of training two networks separately ✓ Can achieve the best detection accuracy
Drawbacks	↓ Have the problem of a high false alarm rate ↓ Seriously restricts their practical applications ↓ Intruders have not been investigated thoroughly
Description	Network intrusion detection plays a very important role in protecting computer network security. The abnormal traffic detection and analysis by extracting the statistical features of flow is the main analysis method in the field of network intrusion detection. However, these features need to be designed and extracted manually, which often loses the original information of the flow and leads to poor detection efficiency. In this paper, we do not manually design the features of the flow but directly extract the raw data information of the flow for analysis. In addition, we first proposed a new network intrusion detection model named the deep hierarchical network, which integrates the improved LeNet-5 and LSTM neural network structures, while learning the spatial and temporal features of flow. By designing a reasonable network cascading method, we can train our proposed hierarchical network at the same time instead of training two networks separately. In this paper, we use the CICIDS2017 dataset and the CTU dataset. These types of attacks are no longer sufficient for today's network intrusion detection research. However, we found that the previously mentioned methods use different flow features, and the datasets used have been released for a long time without including some recent new attack types. In addition, most researchers use a shallow

	classification model, which can achieve better classification results when the feature dimension is small, but when the amount of data used is large and the feature dimension is large, the classification effect will be poor. we designed an anomaly traffic detection model named deep hierarchical network. The deep hierarchical network consists of two layers of the neural network algorithms model. The first layer is based on the improved LetNet-5 convolutional neural network to extract the spatial features of the flow, and the second layer uses the LSTM network to extract the temporal features of the flow. The two networks are simultaneously trained by cascading into a hybrid network to enable the network to automatically extract the spatial and temporal features of the flow. Before introducing the deep hierarchical network, we will first introduce the composition of the traffic data used by the training model.
--	---

CHAPTER 3

AIM AND SCOPE OF THE PRESENT INVESTIGATION

3.1 AIM

The aim of the project is to predict the accuracy of the cyber-attacks in order to understand the possible counter measures.

3.2 SCOPE

It can be achieved through machine learning a proposed topology for a wireless networked control system is studied under several cyber-attack scenarios, and a distributed intrusion detection system is designed to identify the existence of attacks.

More specifically, the paper presents a modelling framework for the closed-loop control system with the intrusion detection system, and a computational procedure to design and compute the intrusion detection system.

The computational procedure delivers a stable closed-loop control system with the intrusion detection system being sensitive to cyber-attacks. Also, a simulation example is used to illustrate the application of the proposed procedure as well as its effectiveness.

In this paper, three types of cyber-attacks are addressed:

- (i) a DoS attack, where the attacker blocks the transmission of information between the transmitting and receiving nodes;
- (ii) a replay attack, where the attacker eavesdrops and records transmitted information, and replays the information at a later time; and
- (iii) a bias injection attack, where the attacker injects values into the transmitted information to manipulate the information.

CHAPTER 4

EXPERIMENTAL MATERIALS AND METHODS

4.1 ALGORITHMS USED

Backend Technologies

- ✓ Python
- ✓ Numpy
- ✓ Sci-learn
- ✓ Eclipse IDE

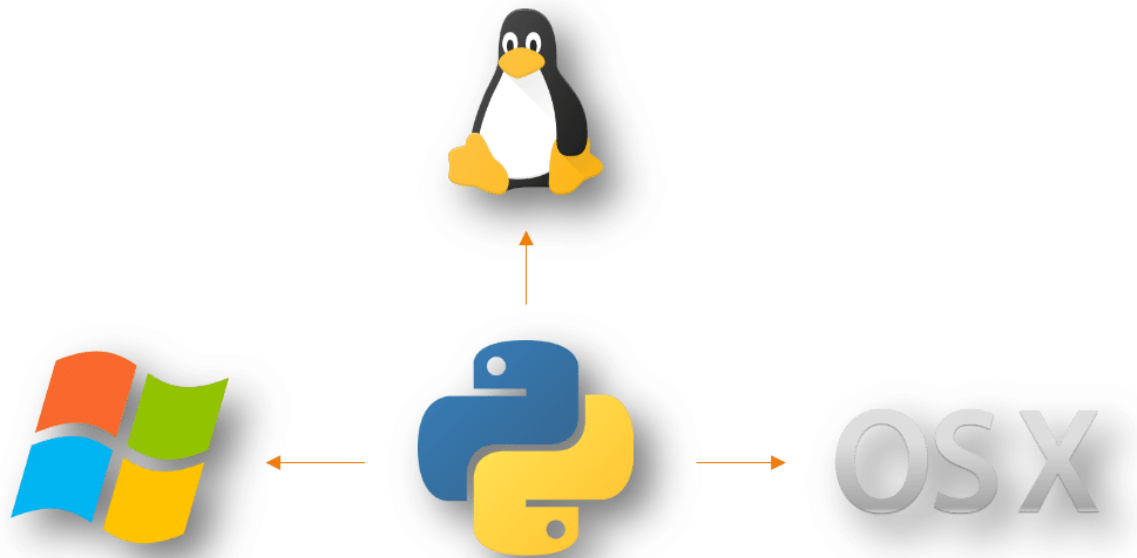
Frontend Technologies

- ✓ Web Technologies
- ✓ Bootstrap

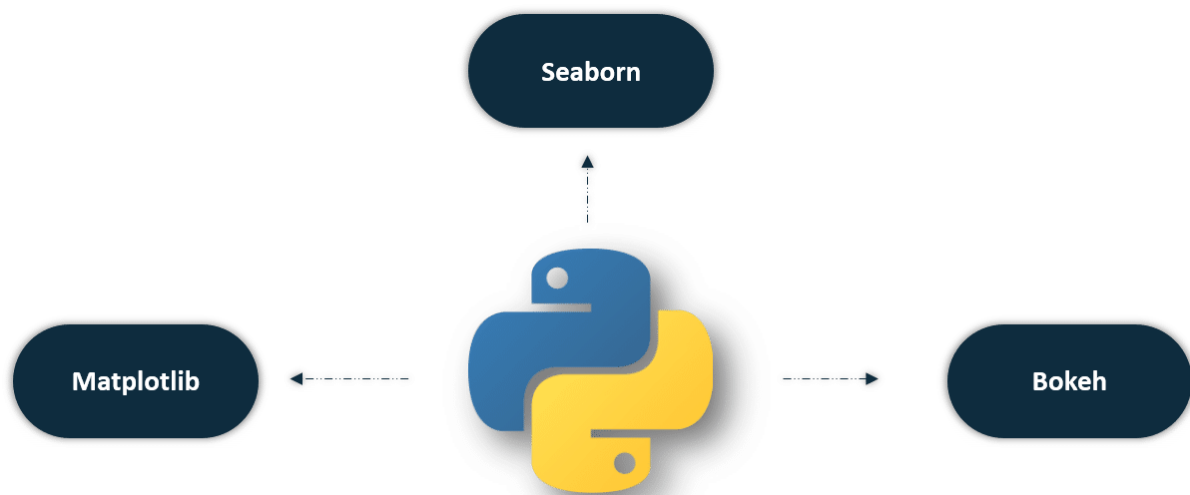
4.1.1 Python

Python is a free, open-source programming language. Therefore, all you have to do is install Python once, and you can start working with it. Not to mention that you can contribute your own code to the community. Python is also a cross-platform compatible language. So,

what does this mean? Well, you can install and run Python on several operating systems. Whether you have a Windows, Mac or Linux, you can rest assure that Python will work on all these operating systems.



Python is also a great visualization tool. It provides libraries such as Matplotlib, seaborn and bokeh to create stunning visualizations.



In addition, Python is the most popular language for machine learning and deep learning. As a matter of fact, today, all top organizations are investing in Python to implement machine learning in the back-end.

Python Line Structure

Python coding style comprises physical lines as well as logical lines or statements. A physical line in a Python program is a sequence of characters, and the end of the line terminates the line sequence as opposed to some other languages, such as C and C++

where a semi-colon is used to mark the end of the statement. A logical line, on the other hand, is composed of one or more physical lines. The use of a semi-colon is not prohibited in Python, although it's not mandatory. The NEWLINE token denotes the end of the logical line. A logical line that only contains spaces, comments, or tabs are called blank lines and they are ignored by the interpreter.

As we saw that in Python, a new line simply means that a new statement has started.

Although, Python does provide a way to split a statement into a multiline statement or to join multiple statements into one logical line. This can be helpful to increase the readability of the statement. Following are the two ways to split a line into two or more lines:

Explicit Line Joining

In explicit line joining, we use a backward slash to split a statement into a multiline statement.

Implicit Line Joining

Statements that reside inside [], {}, or () parentheses can be broken down into two or more physical lines without using a back slash.

Multiple Statements on a Single Line

In Python, it is possible to club multiple statements in the same line using a semi-colon; however, most programmers do not consider this to be a good practice as it reduces the readability of the code.

Whitespaces and Indentation

Unlike most of the programming languages, Python uses indentation to mark a block of code. According to Python coding style guideline or PEP8, we should keep an indent size of four.

Most of the programming languages provide indentation for better code formatting and don't enforce to have it. But in Python it is mandatory. This is why indentation is so crucial in Python.

Comments in any programming language are used to increase the readability of the code. Similarly, in Python, when the program starts getting complicated, one of the best ways to maintain the readability of the code is to use Python comments. It is considered a good practice to include documentations and notes in the python syntax since it makes the code way more readable and understandable to other programmers as well, which comes in handy when multiple programmers are simultaneously working on the same project.

The code can only explain how it does something and not why it does that, but Python comments can do that. With Python comments, we can make documentations for various explanations in our code itself. Comments are nothing but tagged lines of codes which increase the readability of a code and make it self-explanatory. There are different ways of

creating comments depending on the type of comment we want to include in our code.

Following are different kinds of comments that can be included in our Python program:

1. Single Line Comments
2. Docstring Comments
3. Multiline Comments

Single line Python comments are marked with # character. These comments end at the end of the physical line, which means that all characters starting after the # character (and lasts till the end of the line) are part of the comment.

Python has the documentation strings (or docstrings) feature which is usually the first statement included in functions and modules. Rather than being ignored by the Python Interpreter like regular comments, docstrings can actually be accessed at the run time using the dot operator.

It gives programmers an easy way of adding quick notes with every Python module, function, class, and method. To use this feature, we use triple quotes in the beginning of the documentation string or comment and the closing triple quotes at the end of the documentation comment. Docstrings can be one-liners as well as multi-liners.

Unlike some programming languages that support multiline comments, such as C, Java, and more, there is no specific feature for multiline comments in Python. But that does not mean that it is totally impossible to make multiline comments in Python. There are two ways we can include comments that can span across multiple lines in our Python code.

Python Block Comments: We can use several single line comments for a whole block. This type of comment is usually created to explain the block of code that follows the Block comment. Python Block comment is the only way of writing a real comment that can span across multiple lines. It is supported and preferred by Python's PEP8 style guide since Block comments are ignored by Python interpreter or parser. However, nothing is stopping programmers from using the second 'non-real' way of writing multiline comments in Python which is explained below.

Using Docstrings: Docstrings are largely used as multiline comments in Python by many programmers since it is the closest thing to having a multiline comment feature in Python. While it is not wrong to use docstrings when we need to make multiline comments, it is important to keep in mind that there is a significant difference between docstrings and comments. Comments in Python are totally ignored by the Python Interpreter, while docstrings, when used inside the Python function, can be accessed at the run time.

Python Data Types

One of the most crucial part of learning any programming language is to understand how data is stored and manipulated in that language. Users are often inclined toward Python

because of its ease of use and the number of versatile features it provides. One of those features is dynamic typing.

In Python, unlike statically typed languages like C or Java, there is no need to specifically declare the data type of the variable. In dynamically typed languages such as Python, the interpreter itself predicts the data type of the [Python Variable](#) based on the type of value assigned to that variable.

Advantages of Python

- Universal Language Construct
- Support both High Level and Low Level Programming
- Language Interoperability
- Fastest Development life cycle therefore more productive coding environment
- Less memory used because a single container hold multiple data types and each type doesn't require its own function
- Learning Ease and open source development
- Speed and user-friendly data structure
- Extensive and extensible libraries.
- Simple & support IoT
- And many more

Based on the 3 languages that are described above, we decided to use Python as programming language for developing this e-Voting web based application. The important motives are

- Easy to learn, even non experienced programmers can use it. Ex: spacing and tabbing instead of extra syntax
- Interactive mode
- Large and comprehensive standard libraries
- Python programs resemble to that of pseudo-code. This makes it a basis and a must have for beginner programmers due to its extreme ease a difficulty when compared to C++ Java, Perl, and so forth

About Pandas

Pandas is a popular Python package for data science, and with good reason: it offers powerful, expressive and flexible data structures that make data manipulation and analysis easy, among many other things. The DataFrame is one of these structures.

Pandas is a high-level data manipulation tool developed by Wes McKinney. It is built on the Numpy package and its key data structure is called the DataFrame. DataFrames allow you to store and manipulate tabular data in rows of observations and columns of variables.

Pandas is built on top of the NumPy package, meaning a lot of the structure of NumPy is used or replicated in Pandas. Data in pandas is often used to feed statistical analysis in SciPy, plotting functions from Matplotlib, and machine learning algorithms in Scikit-learn. Jupyter Notebooks offer a good environment for using pandas to do data exploration and modeling, but pandas can also be used in text editors just as easily.

Jupyter Notebooks give us the ability to execute code in a particular cell as opposed to running the entire file. This saves a lot of time when working with large datasets and complex transformations. Notebooks also provide an easy way to visualize pandas' DataFrames and plots. As a matter of fact, this article was created entirely in a Jupyter Notebook.

There are two types of data structures in pandas: Series and DataFrames.

1. Series: a pandas Series is a one dimensional data structure ("a one dimensional ndarray") that can store values — and for every value it holds a unique index, too.
2. DataFrame: a pandas DataFrame is a two (or more) dimensional data structure — basically a table with rows and columns. The columns have names and the rows have indexes.

Those who are familiar with R know the data frame as a way to store data in rectangular grids that can easily be overviewed. Each row of these grids corresponds to measurements or values of an instance, while each column is a vector containing data for a specific variable. This means that a data frame's rows do not need to contain, but can contain, the same type of values: they can be numeric, character, logical, etc.

Now, DataFrames in Python are very similar: they come with the Pandas library, and they are defined as two-dimensional labeled data structures with columns of potentially different types.

In general, you could say that the Pandas DataFrame consists of three main components: the data, the index, and the columns.

Firstly, the DataFrame can contain data that is:

- a Pandas DataFrame
- a Pandas Series: a one-dimensional labeled array capable of holding any data type with axis labels or index. An example of a Series object is one column from a DataFrame.
- a NumPy ndarray, which can be a record or structured
- a two-dimensional ndarray
- dictionaries of one-dimensional ndarray's, lists, dictionaries or Series.

Some of the key features of Python Pandas are as follows:

It provides DataFrame objects with default and customized indexing which is very fast and efficient.

- There are tools available for loading data of different file formats into in-memory data

objects.

- It is easy to perform data alignment and integrated handling of missing data in Python Pandas.
- It is very simple to perform pivoting and reshaping of data sets in Pandas.
- It also provides indexing, label-based slicing, and sub-setting of large data sets.
- We can easily insert and delete columns from a data structure.
- Data aggregation and transformations can be done using group by.
- High-performance merging and joining of data can be done using Pandas.
- It also provides time series functionality.
- Inserting and deleting columns in data structures.
- Merging and joining data sets.
- Reshaping and pivoting data sets.
- Aligning data and dealing with missing data.
- Manipulating data using integrated indexing for DataFrame objects.
- Performing split-apply-combine on data sets using the group by engine.
- Manipulating high-dimensional data in a data structure with a lower dimension using hierarchical axis indexing.
- Subsetting, fancy indexing, and label-based slicing data sets that are large in size.
- Generating data range, converting frequency, date shifting, lagging, and other time-series functionality.
- Reading from files with CSV, XLSX, TXT, among other formats.
- Arranging data in an order ascending or descending.
- Filtering data around a condition.
- Analyzing time series.
- Iterating over a data set.

4.1.2 Numpy

Numpy is the core library for scientific computing in Python. It provides a high-performance multidimensional array object, and tools for working with these arrays. If you are already familiar with MATLAB, you might find this tutorial useful to get started with Numpy.

A numpy array is a grid of values, all of the same type, and is indexed by a tuple of nonnegative integers. The number of dimensions is the rank of the array; the shape of an array is a tuple of integers giving the size of the array along each dimension.

NumPy is, just like SciPy, Scikit-Learn, Pandas, etc. one of the packages that you just can't miss when you're learning data science, mainly because this library provides you with an

array data structure that holds some benefits over Python lists, such as: being more compact, faster access in reading and writing items, being more convenient and more efficient.

NumPy is a Python library that is the core library for scientific computing in Python. It contains a collection of tools and techniques that can be used to solve on a computer mathematical models of problems in Science and Engineering. One of these tools is a high-performance multidimensional array object that is a powerful data structure for efficient computation of arrays and matrices. To work with these arrays, there's a vast amount of high-level mathematical functions operate on these matrices and arrays.

an array is basically nothing but pointers. It's a combination of a memory address, a data type, a shape, and strides:

- The data pointer indicates the memory address of the first byte in the array,
- The data type or dtype pointer describes the kind of elements that are contained within the array,
- The shape indicates the shape of the array, and
- The strides are the number of bytes that should be skipped in memory to go to the next element. If your strides are (10,1), you need to proceed one byte to get to the next column and 10 bytes to locate the next row.

Or, in other words, an array contains information about the raw data, how to locate an element and how to interpret an element.

With NumPy, we work with multidimensional arrays. We'll dive into all of the possible types of multidimensional arrays later on, but for now, we'll focus on 2-dimensional arrays. A 2-dimensional array is also known as a matrix, and is something you should be familiar with. In fact, it's just a different way of thinking about a list of lists. A matrix has rows and columns. By specifying a row number and a column number, we're able to extract an element from a matrix.

We can create a NumPy array using the `numpy.array` function. If we pass in a list of lists, it will automatically create a NumPy array with the same number of rows and columns.

Because we want all of the elements in the array to be float elements for easy computation, we'll leave off the header row, which contains strings. One of the limitations of NumPy is that all the elements in an array have to be of the same type, so if we include the header row, all the elements in the array will be read in as strings. Because we want to be able to do computations like find the average quality of the wines, we need the elements to all be floats. NumPy has several advantages over using core Python mathematical functions, a few of which are outlined here:

1. NumPy is extremely fast when compared to core Python thanks to its heavy use of

C extensions.

2. Many advanced Python libraries, such as Scikit-Learn, Scipy, and Keras, make extensive use of the NumPy library. Therefore, if you plan to pursue a career in data science or machine learning, NumPy is a very good tool to master.

NumPy comes with a variety of built-in functionalities, which in core Python would take a fair bit of custom code.

Matplotlib

Plotting of data can be extensively made possible in an interactive way by Matplotlib, which is a plotting library that can be demonstrated in Python scripts. Plotting of graphs is a part of data visualization, and this property can be achieved by making use of Matplotlib.

Matplotlib makes use of many general-purpose GUI toolkits, such as wxPython, Tkinter, QT, etc., in order to provide object-oriented APIs for embedding plots into applications. John D. Hunter was the person who originally wrote Matplotlib, and its lead developer was Michael Droettboom. One of the free and open-source Python library which is basically used for technical and scientific computing is Python SciPy. Matplotlib is widely used in SciPy as most scientific calculations require plotting of graphs and diagrams.

Matplotlib is a plotting library like GNUplot. The main advantage towards GNUplot is the fact that Matplotlib is a Python module. Due to the growing interest in python the popularity of matplotlib is continually rising as well.

Another reason for the attractiveness of Matplotlib lies in the fact that it is widely considered to be a perfect alternative to MATLAB, if it is used in combination with Numpy and Scipy. Whereas MATLAB is expensive and closed source, Matplotlib is free and open source code. It is also object-oriented and can be used in an object oriented way. Furthermore it can be used with general-purpose GUI toolkits like wxPython, Qt, and GTK+. There is also a procedural "pylab", which designed to closely resemble that of MATLAB. This can make it extremely easy for MATLAB users to migrate to matplotlib.

Matplotlib can be used to create publication quality figures in a variety of hardcopy formats and interactive environments across platforms.

4.3 ARCHITECTURE

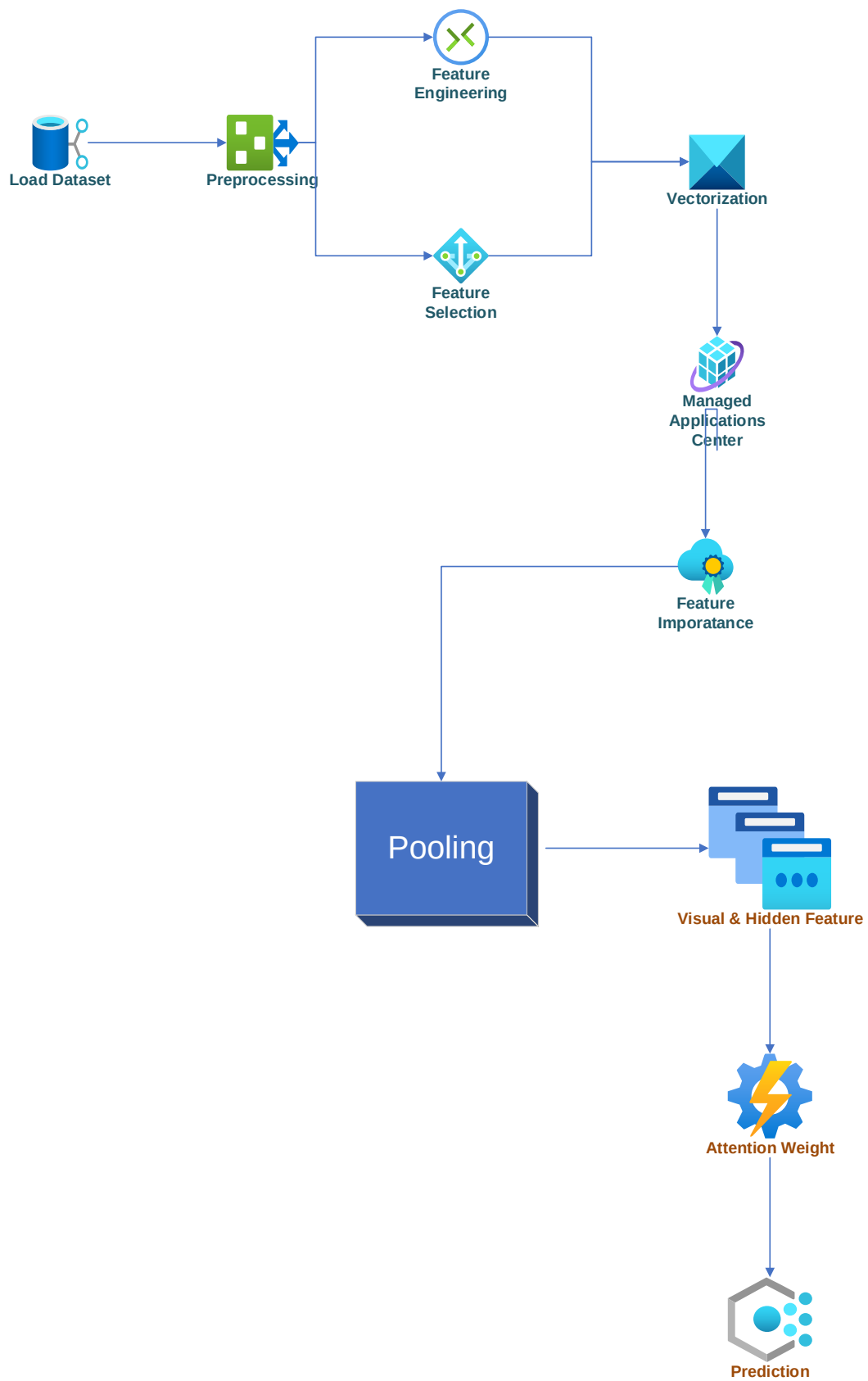


Fig 4.1 Architecture Diagram

An architecture diagram is a graphical representation of a set of concepts, that are part of an architecture, including their principles, elements and components.

What is a diagram? What are the types of diagrams for architecture? The Dragon1 open EA Method makes it very clear: if a diagram does not show a concept, principle or part of a principle, it is NOT an architecture diagram, because it does not show (a part of) the architecture.

There are many kinds of architecture diagrams, like a software architecture diagram, system architecture diagram, application architecture diagram, security architecture diagram, etc.

For system developers, they need system architecture diagrams to understand, clarify, and communicate ideas about the system structure and the user requirements that the system must support. It's a basic framework can be used at the system planning phase helping partners understand the architecture, discuss changes, and communicate intentions clearly.

SYSTEM DESIGN

A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

Mainly, UML has been used as a general-purpose modeling language in the field of software engineering. However, it has now found its way into the documentation of several business processes or workflows. For example, activity diagrams, a type of UML diagram, can be used as a replacement for flowcharts. They provide both a more standardized way of modeling workflows as well as a wider range of features to improve readability and efficiency.

Test Driven Development

Test Driven Development, or TDD, is a code design technique where the programmer writes a test before any production code, and then writes the code that will make that test pass. The idea is that with a tiny bit of assurance from that initial test, the programmer can feel free to refactor and refactor some more to get the cleanest code they know how to write. The idea is simple, but like most simple things, the execution is hard. TDD requires a completely

different mind set from what most people are used to and the tenacity to deal with a learning curve that may slow you down at first.

Functional Testing types include:

1. Unit Testing
2. Integration Testing
3. System Testing
4. Sanity Testing
5. Smoke Testing
6. Interface Testing
7. Regression Testing
8. Beta/Acceptance Testing
9. Non-functional Testing types include:
Performance Testing
 - Load Testing
 - Stress Testing
 - Volume Testing
 - Security Testing
 - Compatibility Testing
 - Install Testing
 - Recovery Testing
 - Reliability Testing
 - Usability Testing
 - Compliance Testing
 - Localization Testing

Unit Testing

UNIT TESTING is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output. In procedural programming, a unit may be an individual program, function, procedure, etc. In object-oriented programming, the smallest unit is a method, which may belong to a base/ super class, abstract class or derived/ child class. (Some treat a module of an application as a unit. This is to be discouraged as there will probably be many individual units within that module.) Unit testing frameworks, drivers, stubs, and mock/ fake objects are used to assist in unit testing.

A unit can be almost anything you want it to be -- a line of code, a method, or a class. Generally though, smaller is better. Smaller tests give you a much more granular view of how your code is performing. There is also the practical aspect that when you test very small units, your tests can be run fast; like a thousand tests in a second fast.

Black Box testers don't care about Unit Testing. Their main goal is to validate the application against the requirements without going into the implementation details.

Unit Testing is not a new concept. It's been there since the early days of programming. Usually, developers and sometimes White box testers write Unit tests to improve code quality by verifying each and every unit of the code used to implement functional requirements (aka test drove development TDD or test-first development).

Most of us might know the classic definition of Unit Testing –

“Unit Testing is the method of verifying the smallest piece of testable code against its purpose.”

If the purpose or requirement failed then the unit test has failed.

In simple words, Unit Testing means – writing a piece of code (unit test) to verify the code (unit) written for implementing requirements.

Black Box Testing

During functional testing, testers verify the app features against the user specifications. This is completely different from testing done by developers which is unit testing. It checks whether the code works as expected. Because unit testing focuses on the internal structure of the code, it is called the white box testing. On the other hand, functional testing checks app's functionalities without looking at the internal structure of the code, hence it is called black box testing. Despite how flawless the various individual code components may be, it is

essential to check that the app is functioning as expected, when all components are combined. Here you can find a detailed comparison between functional testing vs unit testing.

Integration Testing

INTEGRATION TESTING is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing.

integration testing: Testing performed to expose defects in the interfaces and in the interactions between integrated components or systems. See also component integration testing, system integration testing.

component integration testing: Testing performed to expose defects in the interfaces and interaction between integrated components. System integration testing: Testing the integration of systems and packages; testing interfaces to external organizations (e.g. Electronic Data Interchange, Internet).

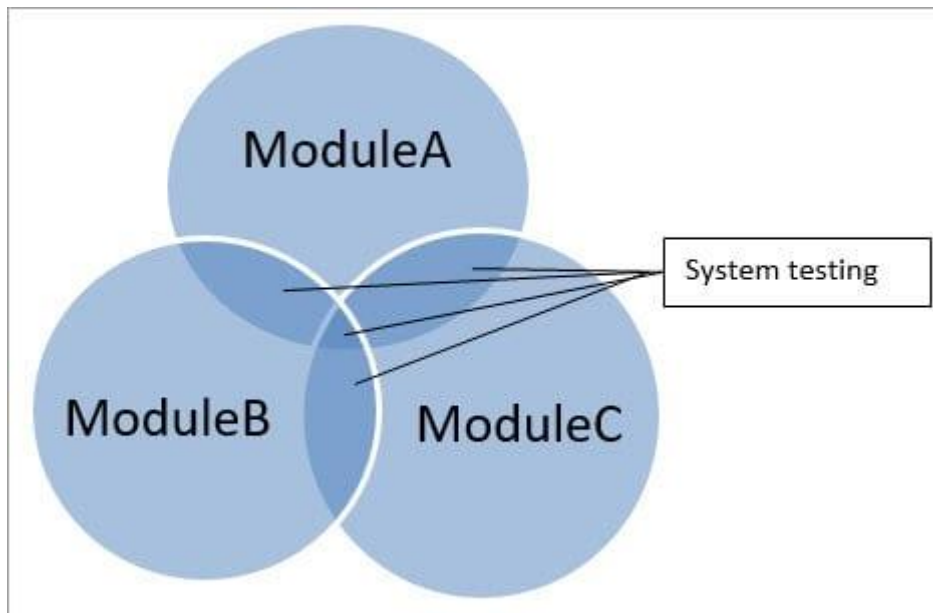
Integration tests determine if independently developed units of software work correctly when they are connected to each other. The term has become blurred even by the diffuse standards of the software industry, so I've been wary of using it in my writing. In particular, many people assume integration tests are necessarily broad in scope, while they can be more effectively done with a narrower scope.

As often with these things, it's best to start with a bit of history. When I first learned about integration testing, it was in the 1980's and the waterfall was the dominant influence of software development thinking. In a larger project, we would have a design phase that would specify the interface and behavior of the various modules in the system. Modules would then be assigned to developers to program. It was not unusual for one programmer to be responsible for a single module, but this would be big enough that it could take months to build it. All this work was done in isolation, and when the programmer believed it was finished they would hand it over to QA for testing.

Integration testing tests integration or interfaces between components, interactions to different parts of the system such as an operating system, file system and hardware or interfaces between systems. Integration testing is a key aspect of software testing.

System Testing

SYSTEM TESTING is a level of software testing where a complete and integrated software is tested. The purpose of this test is to evaluate the system's compliance with the specified requirements. System Testing means testing the system as a whole. All the modules/components are integrated in order to verify if the system works as expected or not. System Testing is done after Integration Testing. This plays an important role in delivering a high-quality product.



System testing is a process of testing the entire system that is fully functional, in order to ensure the system is bound to all the requirements provided by the client in the form of the functional specification or system specification documentation. In most cases, it is done next to the Integration testing, as this testing should be covering the end-to-end system's actual routine. This type of testing requires a dedicated Test Plan and other test documentation derived from the system specification document that should cover both software and hardware requirements. By this test, we uncover the errors. It ensures that all the system works as expected. We check System performance and functionality to get a quality product. System testing is nothing but testing the system as a whole. This testing checks complete end-to-end scenario as per the customer's point of view. Functional and Non-Functional tests also done by System testing. All things are done to maintain trust within the development that the system is defect-free and bug-free. System testing is also intended to test hardware/software requirements specifications.

CHAPTER 5

RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS

5.1 PERFORMANCE ANALYSIS

Intrusion Detection is a major focus of research in network security. This paper analyzes Neural Network (NN) methods being used in IDS. An intrusion detection system with LSTM network is presented. This paper proposed a new anomaly-based approach to detect and prevent wormhole and rushing attacks in ad hoc networks. The proposed solution is based on hierarchical architecture and uses a statistical approach. The intrusion detection and prevention system is implemented in cluster heads.

5.2 DISCUSSION

Implementing this technology on a large scale could evidently bring about several improvements in the future of cyber-threat detection. By utilizing the accuracy we can further predict the possible ways of increasing our safety and preventing many cyber related attacks. The accuracy of these attacks helps us in understanding the most vulnerable modes of attacks and the future is limitless as new modes of attacks are prevention methods are being invented and implemented everyday.

5.3 RESULTS

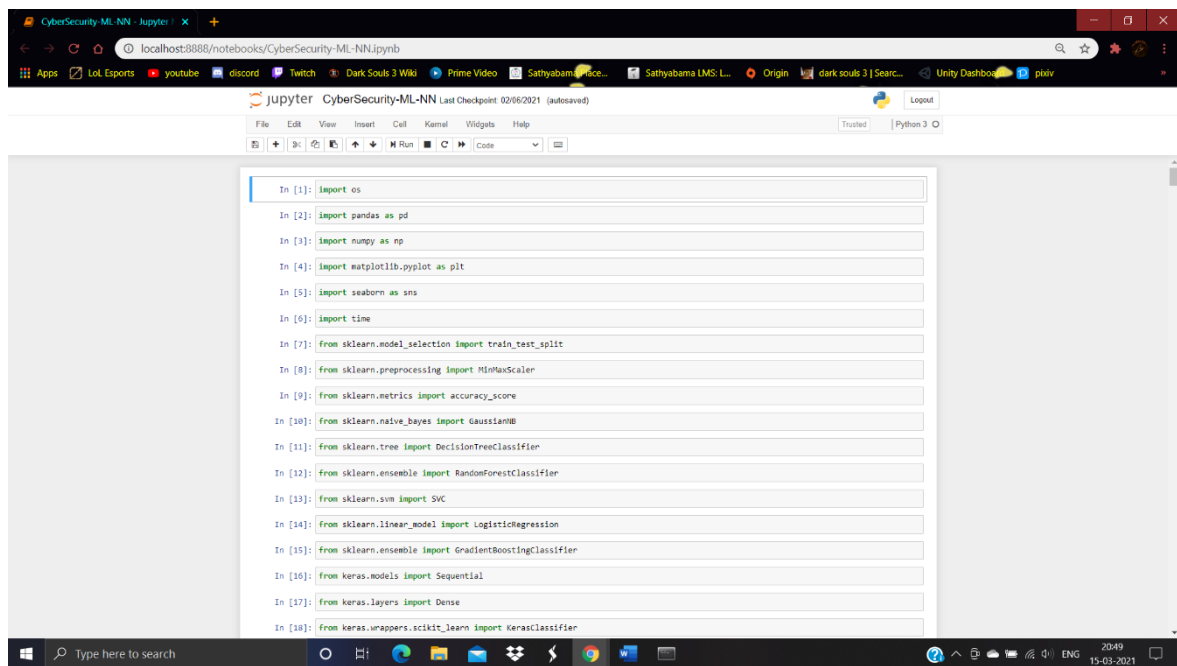


Fig 5.1 Modules

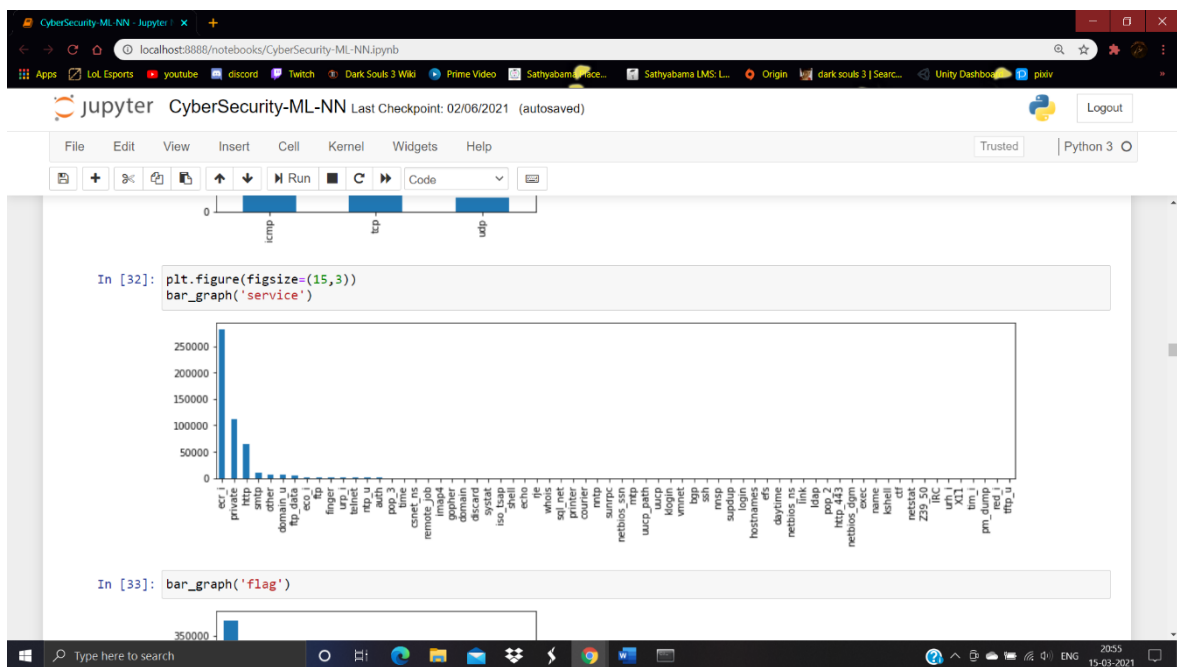


Fig 5.2 Attack types in dataset

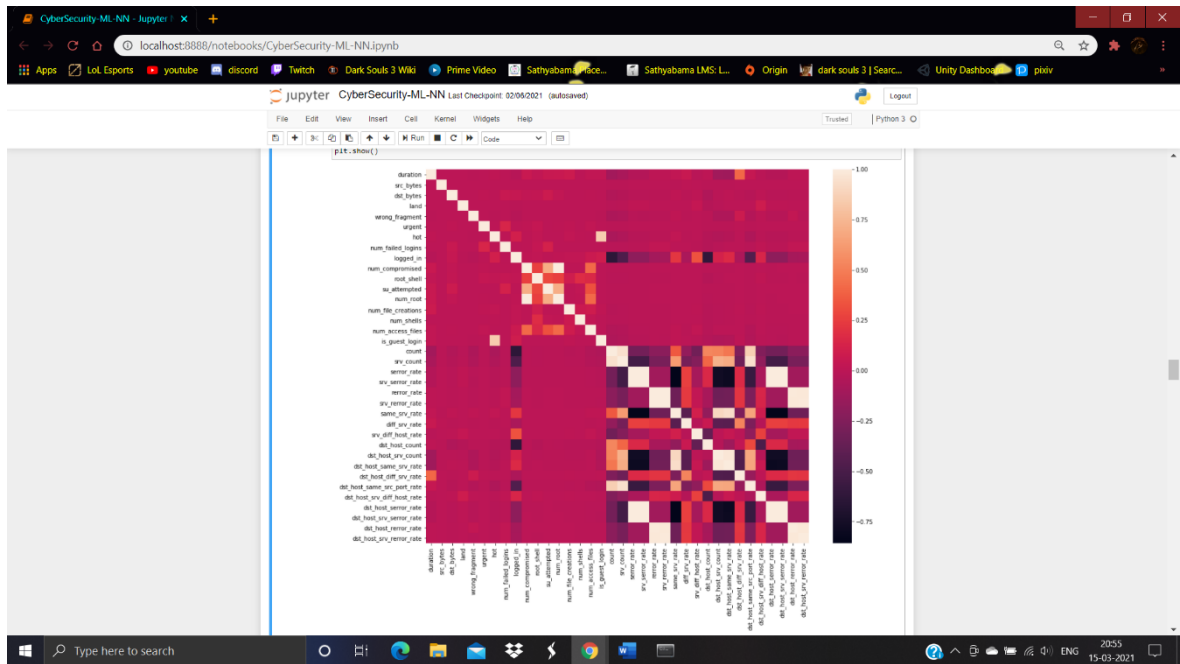


Fig 5.3 Heat Graph of dataset

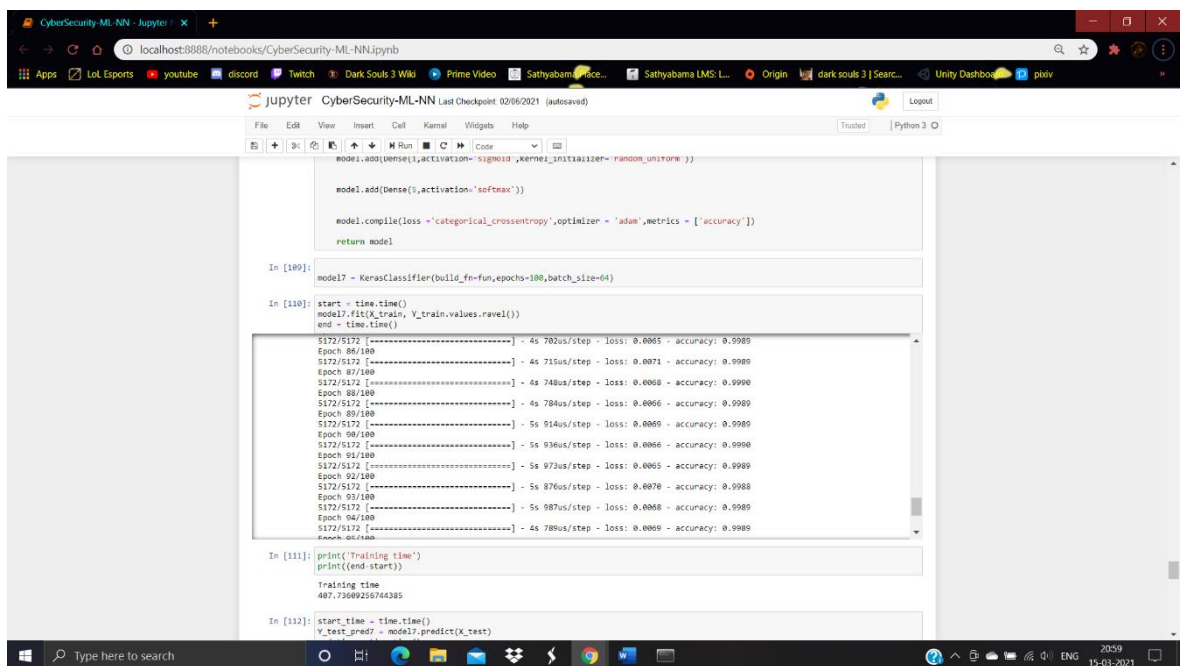


Fig 5.4 Accuracy & loss

This figure shows accuracy and loss percentage.

CHAPTER 6

SUMMARY AND CONCLUSION

6.1 SUMMARY

Let us now discuss about the summary in this section, the major problems I figured out while making this project and I could see that this project can be used for good as well as bad. Its usage depends on person using it. The main idea of developing this project is providing an accurate value for numerous cyber-attacks which can be used to identify and prevent future attacks by developing anti-intrusion software using this accuracy.

6.2 CONCLUSION

Intrusion Detection is a major focus of research in network security. This paper analyzes Neural Network (NN) methods being used in IDS. An intrusion detection system with LSTM network is presented. This paper proposed a new anomaly-based approach to detect and prevent wormhole and rushing attacks in ad hoc networks. The proposed solution is based on hierarchical architecture and uses a statistical approach. The intrusion detection and prevention system is implemented in cluster heads.

REFERENCES

- ▶ B. Wang, J. Y. Chen, W. Y. Liu, and L. T. Yang, "Minimum Cost Placement of Bistatic Radar Sensors for Belt Barrier Coverage," *IEEE Trans. on Comput.*, vol. 65, no. 2, pp. 577-588, Feb. 2016.
- ▶ D. Shen, G. Wu, and H.-I. Suk, "Deep learning in medical image analysis," *Annu. Rev. Biomed. Eng.*, vol. 19, no. 1, pp. 221–248, 2017.
- ▶ F. J. Parrado-Garcia, J. Vales-Alonso, and J. Alcaraz, "Optimal planning of wsn deployments for in situ lunar surveys," *IEEE Trans. Aerosp. Electron. Syst.*, 2017 online, doi:10.1109/TAES.2017.2674258.
- ▶ H. Al-Hamadi and I.R. Chen, "Integrated Intrusion Detection and Tolerance in Homogeneous Clustered Sensor Networks," *ACM Transactions on Sensor Networks*, vol. 11, no. 3, March 2015, article no. 47.

- ▶ H. Mahboubi and A. G. Aghdam, "Distributed deployment algorithms for coverage improvement in a network of wireless mobile sensors: Relocation by virtual force," *IEEE Transactions on Control of Network Systems*, 2016 online, doi:10.1109/TCNS.2016.2547579
- ▶ H. Park, et al., "PASSAGES: Preserving Anonymity of Sources and Sinks against Global Eavesdroppers," *IEEE INFOCOM*, April 2013.
- ▶ I.R. Chen, F. Bao, M. Chang, and J.H. Cho, "Dynamic Trust Management for Delay Tolerant Networks and Its Application to Secure Routing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 5, 2014, pp. 1200-1210.
- ▶ K. Nie, Q. L. Zhou, W. J. Zhu, and Z. Y. Zhang, "Three kinds of network attack modeling based on sequential logic," *Comput. Sci.*, vol. 45, no. 2, pp. 209–214, 2018.
- ▶ L. Zhang, Y. Cui, J. Liu, Y. Jiang, and J. P. Wu, "Application of machine learning in network space security research," *J. Comput.*, vol. 41, no. 9, pp. 1943–1975, 2018.
- ▶ M. Al-Qatf, Y. Lasheng, M. Al-Habib, and K. Al-Sabahi, "Deep learning approach combining sparse auto-encoder with SVM for network intrusion detection," *IEEE Access*, vol. 6, pp. 52843–52856, 2018.
- ▶ R. Mitchell and I.R. Chen, "Behavior Rule Specification-based Intrusion Detection for Safety Critical Medical Cyber Physical Systems," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 1, 2015, pp. 16-30.
- ▶ X. W. Gong, J. S. Zhang, D. Cochran, and K. Xing, "Optimal placement for barrier coverage in bistatic radar sensor networks," *IEEE/ACM Trans. on Networking*, vol. 24, no.1, pp.259-271, Jan. 2016.
- ▶ Y. Li and G. Q. Huang, "Attack intention detection method based on possible graph," *Comput. Eng. Sci.*, vol. 39, no. 4, pp. 698–707, 2017.