CHAPTER 1

INTRODUCTION

The major impact of problem to the cyber security in not basically from the outsider malicious malwares or spywares, the insider can harm the organization by leaking the information to the outside world. The problem is getting worst every day as information is growing bigger and bigger. According to the survey from the survey testimony i.e. Insider Report 2019, it is proven that 92% of the organizations are prone to insider threat from their own resources. As the insider has the access to the resources and organizational assets, there might be the chances that they can demoralize the data availability, Confidentiality and Integrity of confidential information than exterior attackers.

There may be many reasons behind the motivation to the insider to leak the organizational data such as Organizational political affairs, Pressure, Greediness , Anger , Commercial Gain, Betrayal, Jealousy , Dissatisfaction over work pressure and other parameters which affect the system as shown in Figure 1.1.

Figure 1.1. Motivations behind the Insider Outbreaks

The malicious employee is first who can breach the security and harm the company security system by theft and defacement of records. Almost 65% of the damage happens due to carelessness and negligent employees (Inadvertent Insiders, Malicious Insiders, and Discontented Insiders, External third parties, Contractors and IT Employees and software suppliers as shown in Figure 2.1.

Figure 1. 2. Types of Insiders.

According to the survey, the chief hazards are personnel workers (50%), Restricted IT Employees (59%) and other contractors (52%) as shown in Figure 1.3. Every organization having many employees , among them some of the employees use unnecessary access right by damaging confidentiality of data, complexities in technology usage , lack of knowledge in knowing the system prototyping and harm the system as well as thoughtful data .

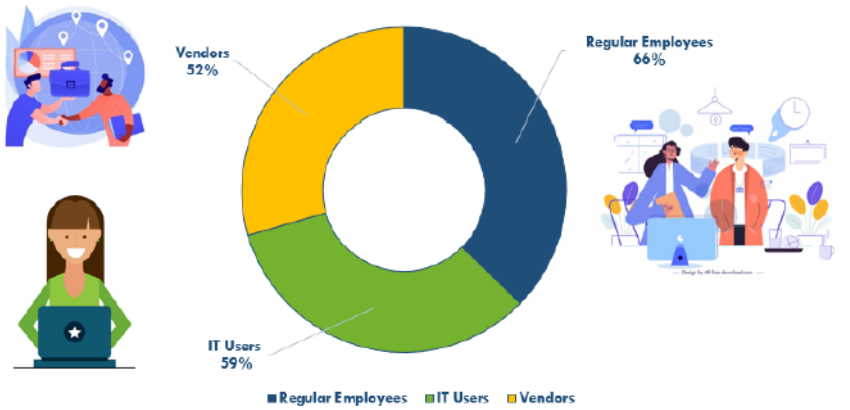


Figure 1.3. Chief Insider Threat Players

Machine Learning, Deep Learning, Artificial Intelligence are the trending focused areas to define the effective system or architecture deployment to minimize the damage from the insider by proving more security features to the existing system.

Machine Learning provides the solution to highest extent, but deep learning also into the competition to detect the insider threat activities and present the improvised results. This results in outstanding security system that can be robust and available to adapt easily by all the organizations to save themselves from information leakage.

This project mainly focuses on the user behavior to detect the insider attack within the organization. Based on the above considerations, we have come up with some solutions where we focus on the behaviour of individual user, User activities and analysis on the access rights and usage to check to outcome as whether they are Normal user with no harm or Abnormal (Malicious). Feature Engineering is the process where we select the fixed set of procedures to identify behaviour of the employee effectively. The implementation is done by applying various machine learning and deep learning algorithms to get high classification accuracy of the model. The trained data is feed to the Model engine to gain the experience about the user activities and test data is used to find the accuracy of the model and defining the behaviour of the user a normal or malicious.

1.1 Perseverance

Consider the all types of organizations (small, medium and large scale industries) are always vulnerable to insider threats starting from the tiny business to Fortune corporates, governing councils, agencies and many defence federal sectors. To win the battle against the insider threat, administrations must ponder a practical and preclusion engrossed program to mitigate the threat.

1.2 Effective Organizational Insider Threat Alleviation procedures

- Train the employees to learn the data importance, culture, asset knowledge and serious landscapes of organization.
- Awareness program on organization’s investment, a culture, Operational cost, Employee salaries, Maintenance cost and overall pliability.

- Adopt the defensive and sympathetic cultural and business values and protect public freedoms.
- Maintain the non- hazardous office environment, value individually and individual human values to all the employees.
- Conduct the sessions within the organization to know the value of financial loss, IP Theft, Unauthorized disclosure, Disruption of timely services, Infrastructure damage.

1.3 Components of Effective implementation of Insider Threat Detection Framework

Figure 1.4: Planning and Execution of effective Framework platform

The initial goal and purpose of the organization must be very much clear to all the organizational members to know the severity of information leakage to outside globe. Set the standard procedures and policies satisfy the business needs. Form the organizational committee who can lead the employees towards the success and leadership sprit.

Physical and Intellectual property rights should be clearly defined at workplace by saving the critical assets and resources within the company. Clarifying that how important one's position and workforce matter and what are the potential risk factors. Form the team of

brainy leaders to mitigate and detect the insider threat issues and preventions.

1.4 Insider Overview

An insider (Person) who is having the access rights and authorized resource user can leak the organizations property, data, knowledge, personnel portfolios, resources, Hardware and Communication system and support the foreign competitor to grow their organization.

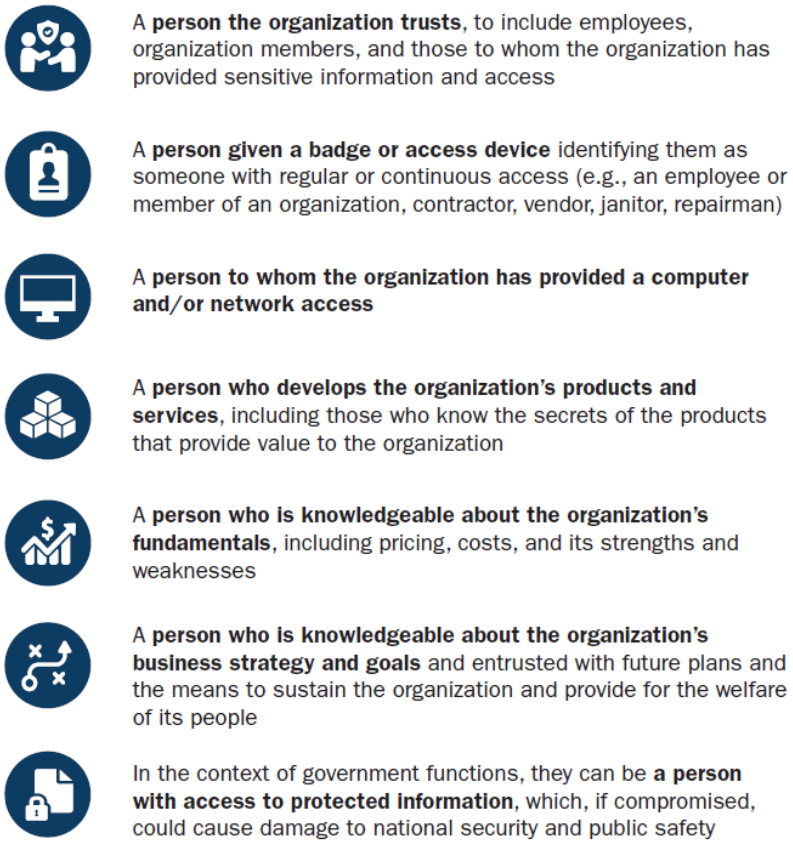


Figure 1.5: Insider Impressions

1.5 Deep Learning: Ideology and Overview

Any complex problem needs a solution, many learning methods are evolved to understand the complexity and computational approach of the problem, in that the methodologies towards Deep Learning plays a very crucial and satisfactory towards to finding of solutions. Here in this context we have used many concepts of Deep learning into our study. Let us start the terminologies used throughout the research process.

- **Deep Learning**

Deep learning (DL) also refereed as Deep Structured Learning or Large Neural Networks, which is also as subfield of Machine Learning (ML) related to the Algorithms, Functions, Units of Neural Networks, Modeling, Transformations and Simulations.

We have various DL Frameworks to deploy and implement the model based approaches and strategies. In which most common and powerful models are Convolutional Neural Networks (CNN), Depp Neural Networks (DNN), Recurrent Neural Networks (RNN), Deep Belief Networks (DBN), Deep Reinforcement Learning (DRL), Self-Thought Learning (STL), Artificial Neural Networks (ANN) , Representational Learning (RL) , Computer Vision (CV) and Natural Language processing (NLP).

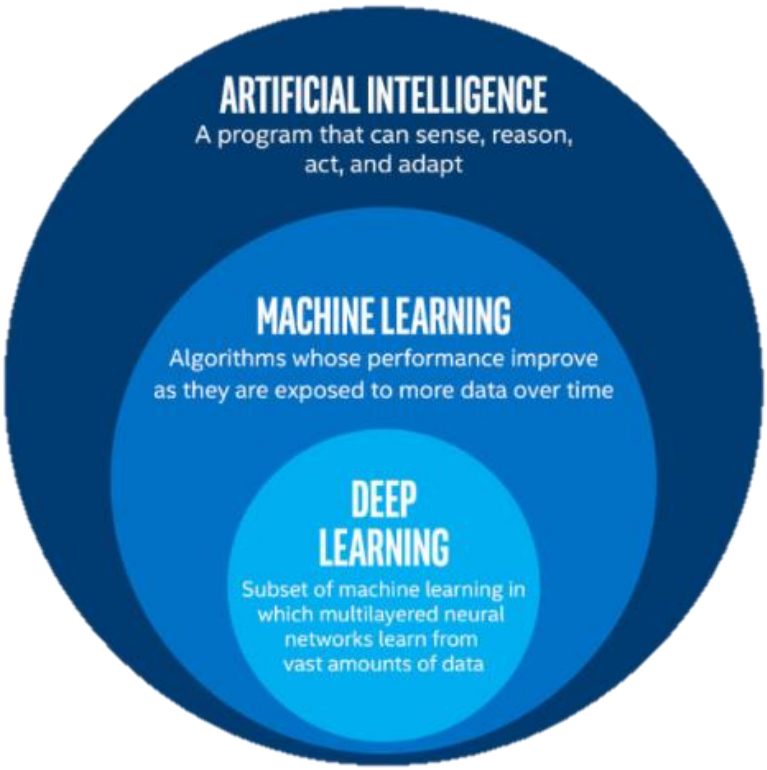


Figure 1.6: Deep Learning Existence

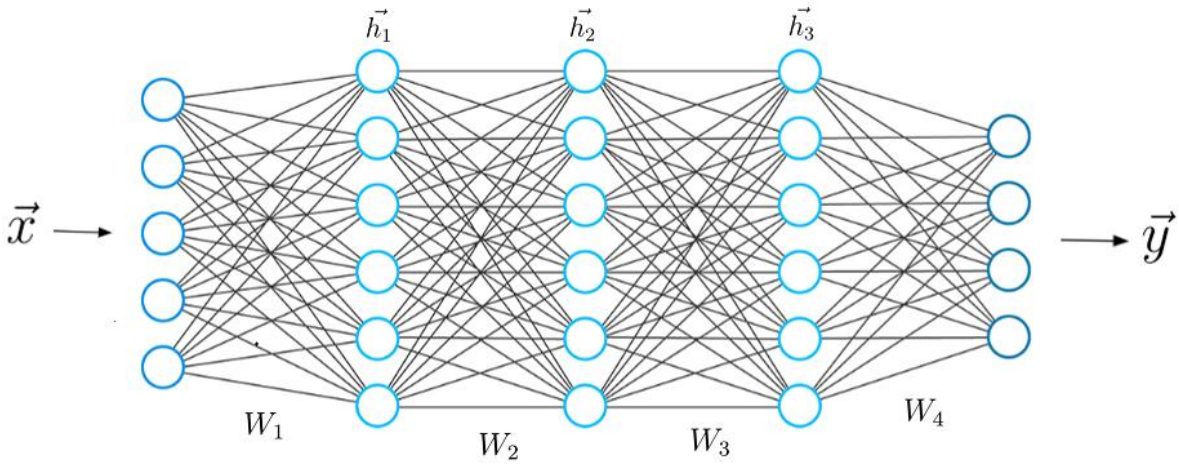
- **Neural Networks (NN)**

  Neural Network is the invention from the biological Neuron System of Human Body by studying knowledge, Information Processing, Computing Speed and distributed communication mechanism of Human Neural System and brain activities. The term ANN is also the part of NN, Which is the collection of Algorithms and models designed especially to recognize the input, process it and find the patterns. NN are helpful in training the model and also to test and validate the input data. The recognized patterns are

numeric or categorical data that comprises in vectors, into which data transform to video, image, audio, text and any static or dynamic time series data.

Neural Networks helps in Classification and Clustering of data using train and test input dataset. It helps in clustering unlabeled data according to the similarities extracted from the input and classifies the data when we have trained input data that works as an experience to the machine to develop the model.

The multilayer model view of structured algorithm of deep learning model can be depicted as:



The above figure shows the Input, Hidden and Output layers.

X: Set of Inputs to the system

W: Weights assigned to the input values

H: Hidden layers

Y: Predicting values at output layer

CHAPTER 2

LITERATURE SURVEY

Insider threat detection can be solved by means of ML & DL approaches, in which the problem can be easily defines, execute and deploy the model, finally the predictions. Many authors have contributed their research on the above trending technologies and problem. The

survey helps in enhancing the knowledge towards the field of cyber security. Many theories, Novel Approaches and strategies are deliberated to stop the unintentional entertainments that undesirably affect the privacy, secrecy, confidentiality and organizational assets.

The survey can be discussed as a case study to enhance our research direction and as follows:

Title: Insider Threat Detection using Deep Learning

Published in: ICCS 2018: Computational Science – ICCS 2018

Authors: Fangfang Yuan1, Yanan Cao1, Yanmin Shang, Yanbing Liu1, Jianlong Tan1, and Binxing Fang

Year: 2018

The authors have given the description about the current scenario over the detection of insider threat using machine learning techniques. The proposed system uses Feature Engineering algorithm i.e. Convolutional Neural Networks (CNN) algorithm to resolve the above stated problem. The user behaviour is the central point of study and used Deep Neural Network (DNN) approach and has produced the novel methodology to detect the insider threat and analyze the user behaviour. First they presented the architectural to work on Malicious Behaviour. Long-Short T Memory (LSTM-CNN) to excerpt the chronological Topographies. Later the CNN model found the fixed sized feature matrix classification. The outcome of the article is to find the probability of the inconsistent behaviour of individual user. The experimental study proved the the value of Area under Curve (AUC) is 0.9449 Receiver Operating Characteristics Curves (ROC) with regards to the best case value.

Merits:

- Analyze the user's workstation based activities
- Findings of AUC and ROC on unbalanced dataset.
- ADAM Gradient Boosting optimizer for training the model.

Title: Deep Learning based Attribute Classification Insider Threat Detection for Data Security

Published in: IEEE Third International Conference on Data Science in Cyberspace

Authors: Fanzhi Meng, Fang Lou, Yunsheng Fu, Zhihong Tian

Year: 2018

The author proposed the experimental study to detect the mischievous insiders using the Recurrent Neural Networks (LSTM-RNN) model. Framework design includes various

components such as Feature Engineering, Aggregator for occurrence of events, different attribute-based classifiers anomaly detection calculator application, all together integrated to form the final insider threat detection system. The models are trained using Support Vector Machine (SVM), K-Nearest Neighbor (KNN), Isolation Forest (IF), and Principal Component Analysis (PCA) machine learning models for the threat detection. Results are comparing with the deployed algorithms based on the attributes such as True_Positive_Rate (TPR), False_ Positive_Rate (FPR), Precision value and finally and accuracy of the model. Anomaly Detection Calculator is used to find the abnormal behaviour on the results of numerous attribute classifiers. The results of the experimental study says the proposed approach is achieving the highest accuracy of 93.85 % compared the Isolation Forest of 89.54%.

Merits

- Comparison study of above algorithms
- Faster prediction Rate
- Take less memory is classification of features

Title: A Graph Based Framework for Malicious Insider Threat Detection

Published in: Proceedings of the 50th Hawaii International Conference on System Sciences (HICSS)

Authors: Anagi Gamachchi, Li Sun and Serdar Boztas

Year: 2018

The hybrid model has been used to mitigating the insider threat problem, which is founded on the Graphical Investigation scheme and anomaly detection strategies to solve the cyber threat of insider employees. The Graphical Processing unit (GPU) contains of Induced Sub Graphs, Graph/Sub graph Attribute Extraction, Generate Original Graph and Graph/Sub graph attributes. Anomaly Detection Unit consists of implementation of Isolation Forest Machine learning Algorithm to find the Anomaly scores for individual users. The CERT Carnegie Mellon University dataset version of r4.2 used in the study of the article. The conversation is founded on the “Research and Engineering / Engineering” work part, and finishing results for all three work roles measured for the comparative study. The authors tried to concentration on integrating as numerous as conceivable input constraints to advance the efficiency.

Merits

- Useful in email and instant messenger communication
- Extract the graphical parameters
- Clustering Mechanism used is more effective.

Title: Insider threat detection based on deep belief network feature representation

Published in: International Conference on Green Informatics, 2018

Authors: Lingli Lin , Shangping Zhong , Cunmin Jia , Kaizhi

Year: 2018

The authors explained the Hybrid Framework Architecture on Deep Belief Network approach on insider threat issue and mitigations. The hidden features are extracted by using the Machine learning unsupervised model such as audit logins, after that they used the Once-Class SVM model to train the features. They have taken the dataset of Audit Logs which detects the actions performed by the users, the feature extracted are Login and Logout information, Directory access for file transfer, Usage of resources such as Universal Serial Bus Drives, Printers and mobile phones. HTTP and mailing services are also one of the parameter to know the network behavior of the users. At training phase, the attributes are selected, Features are extracted and converted as Feature vectors as input unit to the Deep Belief Network block, which creates a model using SVM and finds the predicted outcome.

Merits

- DBN is consider as the best classifier compared to others
- Helps in finding the Multiple-Domain Feature Engineering Process.
- More suitable strategy to find the insider threat.

Title: Insider Threat Detection Using Supervised Machine Learning Algorithms on an Extremely Imbalanced Dataset

Published in: International Journal of Cyber Warfare and Terrorism

Authors: Naghmeh Moradpoor Sheykhkanloo

Year: April-June 2020

The author explained the detailed explanation about the Data mining technique to identify the insider threat using unbalanced CERT Dataset. They have mainly concentrated the reducing the time to Train, Test and validate the data rather than the findings of accuracy of the Machine Learning Models. The impact on analysis is more on the imbalanced dataset. They have made extensive survey of machine learning algorithms taking the parameters such as Framework design, System configurations, Proposed Environment, Experimental Outcomes, Parameters study during the evaluation and implemented in Real-time or not. Next section explains are Data Analysis where they performed all sorts of analysis methods on different demo's scenarios. The data mining steps performed on the input data are Data Preporcessing and Outlier Identification, Data Decomposition and Conversion, Data Reduction and segmentation and finally the data classification. The results are produced based on balanced and imbalanced dataset.

Merit

- Concentrate on Data Analysis Procedures
- Usage of Machine Learning algorithms
- Comparison study help to derive the next level research perspectives.

Title: Exploring anomalous behaviour detection and classification for insider threat identification.

Published in: International Journal of Network Management

Authors: Duc C. Le, Nur Zincir-Heywood

Year: 2020

The author has done an excellent job on Multiple Data Granularity stages to find the threat detection in organization and Anomaly-based detection on ML algorithms. The Evaluation metrics are calculated on individual and group of users. Here the supervised Machine Learning (ML) algorithms and Unsupervised Anomaly-Based detection creates a better impact on the data high precision and low recall rate. The proposed algorithms consists of 5

stages of design interfaces namely Data collection (CERT), Data Acquisition (Reasoning, User Data, Probability, and Frequency Distributions), Detection Unit (Anomaly-Based Detection Stage and ML-Based Detection Stage) and the final phase is performance evaluation and predictions (Normal or Malicious).

Merits

- Outlier detection results are pretty useful.
- Autoencoder help is minimizing the testing time
- Can save the training time up to 20%

CHAPTER 3

SYSTEM REQUIREMENT SPECIFICATIONS

Prerequisites exam is simple for undertaking improvement. Prerequisites need to be archived,

vast, quantifiable, and testable and characterised to some extent of detail adequate for framework plan. Necessities can be engineering, underlying, social, realistic, and beneficial. Software Requirements Specifications (SRS), product requirements specific is a faraway attaining depiction of the planned reason and the weather for programming being worked on.

3.1 Functional Requirements

The tools to execute the Python programs can be many, among that we can go with Visual Studio, Anaconda Navigator (Jupyter Notebook) or any IDLE based on Python. The online tool from Google can be an effective solution towards the execution of Python coding.

3.1.1Approach 1: Microsoft Visual Studio

This is an Integrated Development Environment (IDE) from the US-Microsoft Organization which is basically used in the development and execution of the programs. More efficient and powerful applications such as Website Development, Mobile Application Development and other Web-based Apps can be designed very effectively and easily. It support for productive design, Development of Cross-platform Application and (Artificial Intelligence) AI based power tools.

The major contributions from this product are:

- Project Scaling ability and support for the complexity
- .NET and C++ Platform to work with any code integrity
- Real-time coding experience
- Automatic Code Writing tool (IntelliCode)
- Sharing Multiple Screen on Single Platform
- Unified cloud Support

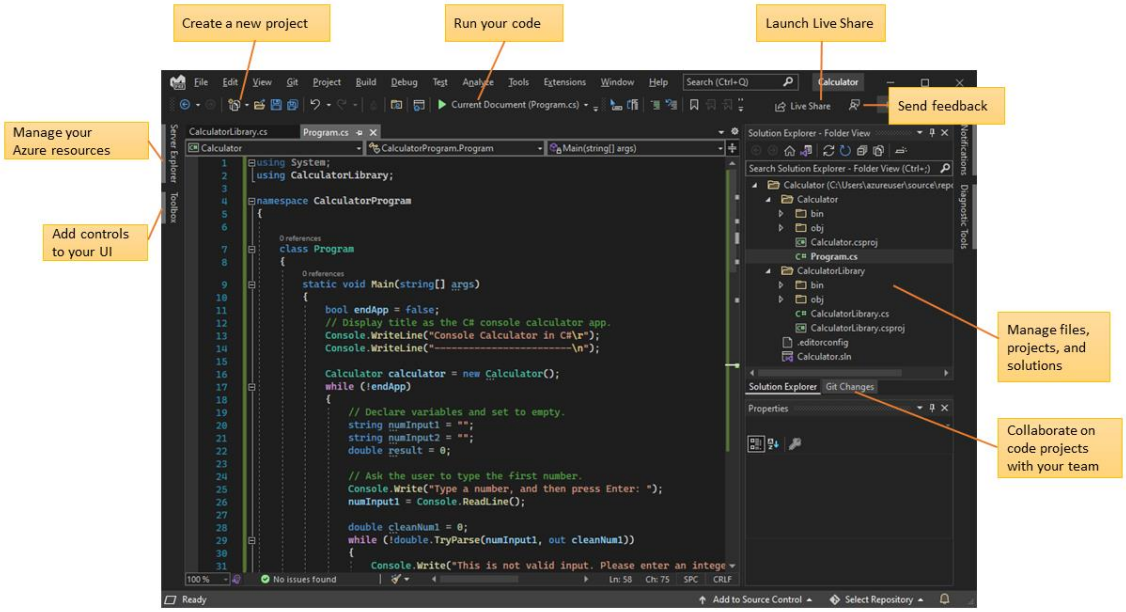


Figure 3.1 Microsoft Visual Studio 2022 IDE, with callouts indicating the location of significant structures and functions.

3.1.2. Approach 2: Jupyter Notebook (Anaconda Navigator)

This tool is also known as IPython Notebook, and it is Open-Source Distribution Software and provides the platform for development of web applications, computational interactive and specific environment for the users to create notebook documentations. It support for individual code execution , browser based interoperability, can plot various graphs using python libraries and also support for many open source libraries like Bootstrap, JQuery, Tornado, Matplotlib , Seaborn and others.

The features of Jupyter Notebook can be listed as:

- Flexible Notebook Interface
- Useful tool in Machine learning, Deep learning and Ai based Application and model Design.
- Creating and sharing the computational Documents.

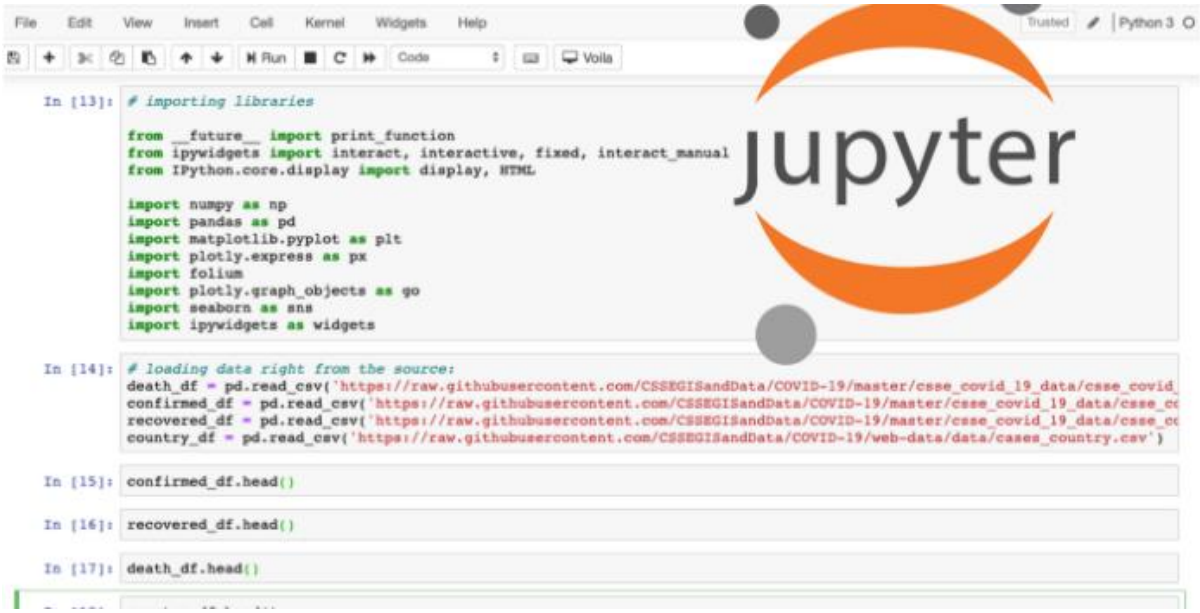


Figure 3.2 Jupyter Notebook Dashboards

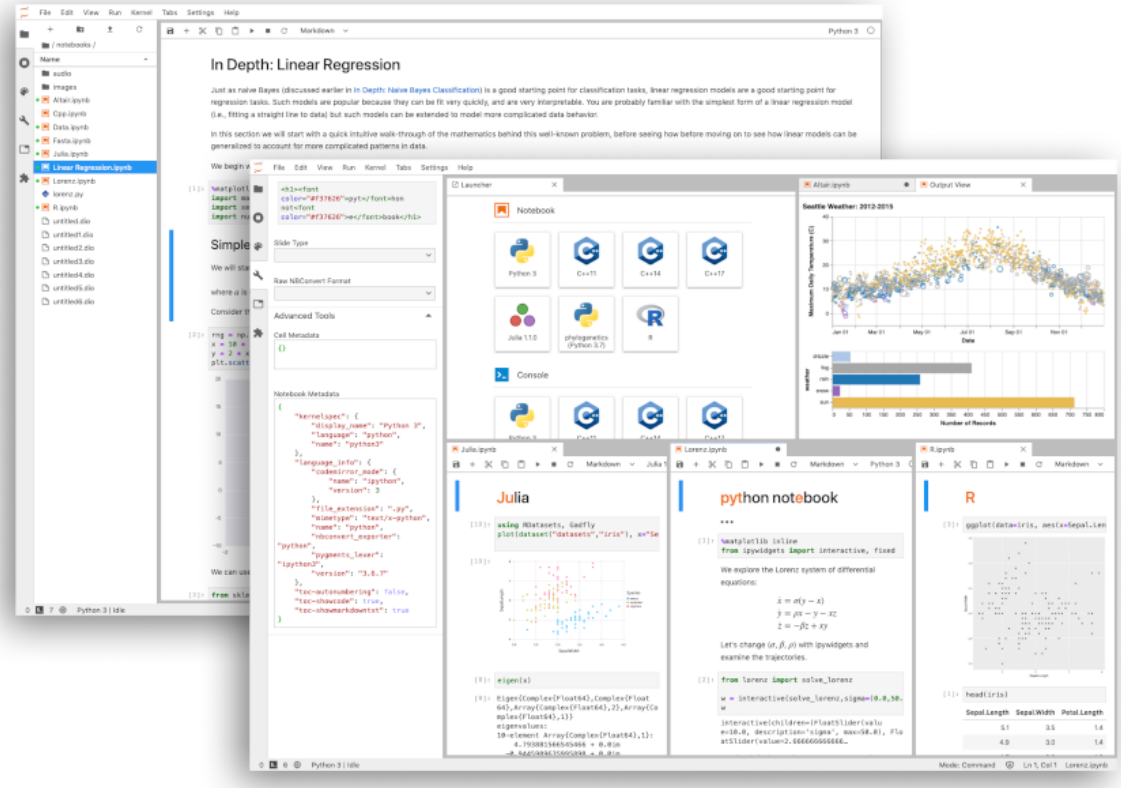


Figure 3.3: Notebook support for plotting

3.1.3 Approach 3: Python IDLE

Python IDLE (Python Integrated Development and Learning Environment) help is writing the code very effectively and efficiently and helpful tool to the Python learning who wants to start from the scratch and beginners can have an advantage to execute the code easily. This is a powerful interpreter and compiler to run the code.

It's an Interactive Interpreter also known as shell, which executes the python written code, reads the input, evaluate the statements and print the output on the standard output screen provided.

File Editor Help to edit the code, save the program in text files and store as .py file.

Figure 3.4: Python IDLE Download Page

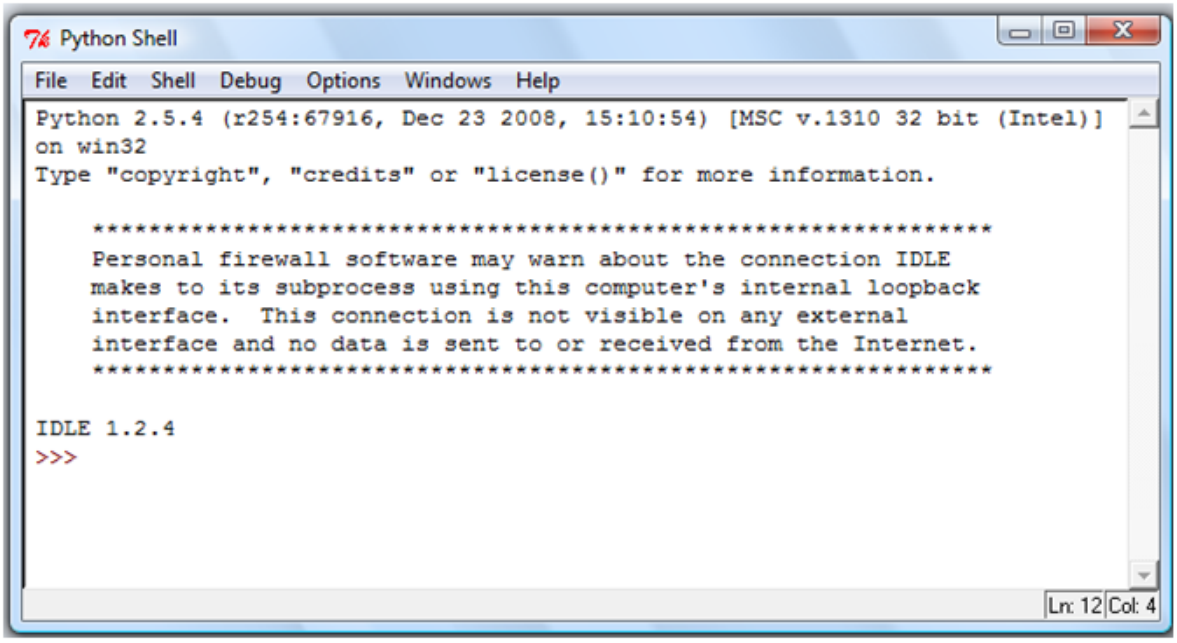


Figure 3.4: Python IDLE prompt to write and execute code.

3.1.4 Approach 4: Google Colab

Google Colab, Also called as Colab in short is a powerful Machine Learning, Deep Learning and Data Analysis Tool that allows mixing the Python script along with text document. Rich support for Plotting the graphs, Diagram, Charts, Import Images, HTML Tags Support and LATEX format API conversions. Additional functional is it works on cloud model where document can be accessed and run on any platform independent of framework design and operating system. The runtime support for Virtual Hard Disk space and 12GB of RAM to execute the application is very excited feature of Colab. The uploading of files is very easy in this application so that it connects to the runtime.

Some of the important feature is:

- Remote Desktop Connection
- Runtime Environment
- Dataset Upload Features
- I/O operations and Operating System API Support
- General Processing Unit (GPU) availability

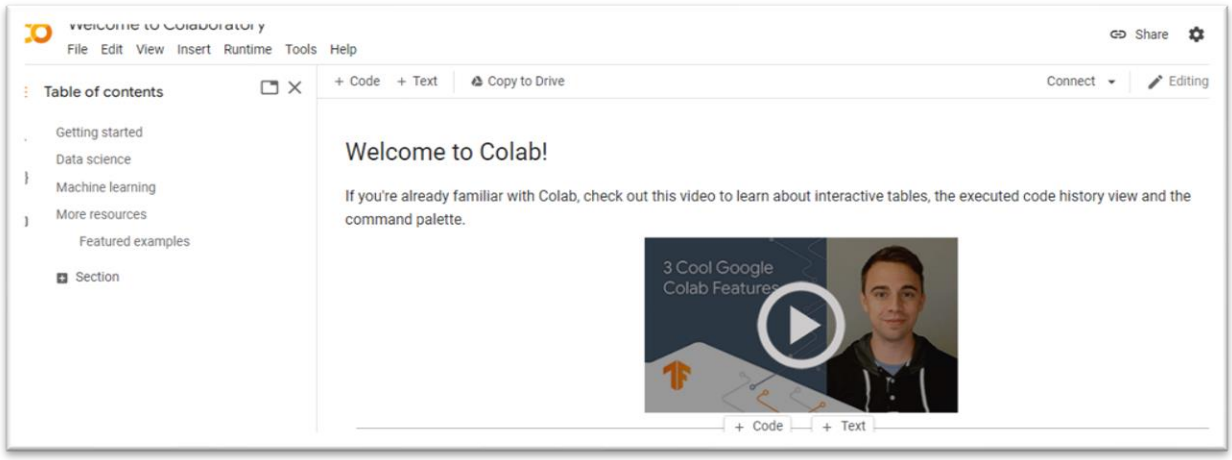


Figure 3.5: Welcome page of Google Colab

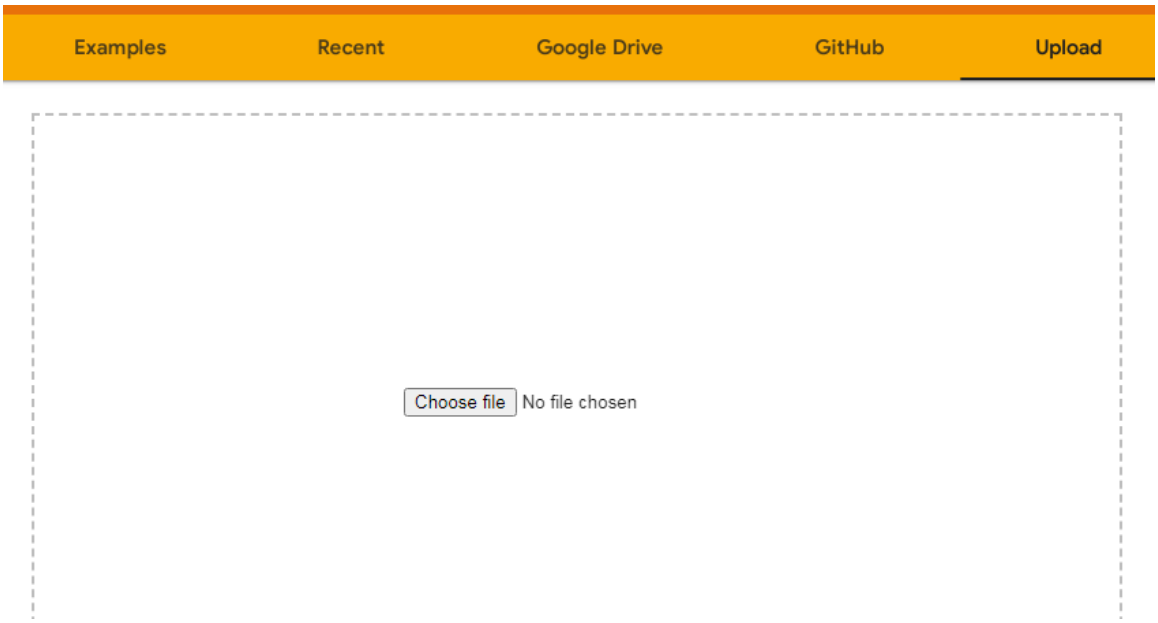


Figure 3.6: Upload the Notebook File

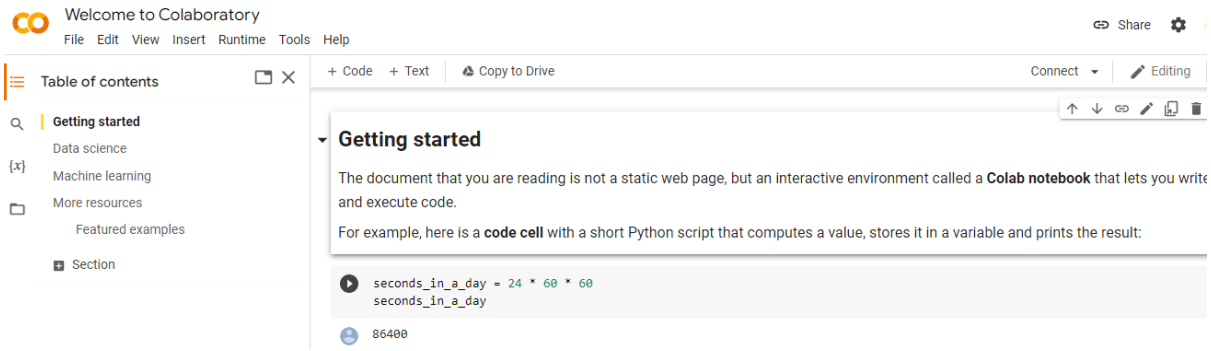


Figure 3.7 Start the Application Page

3.2 Non-Functional Requirements

3.2.1 Hardware Requirements

Processor	:	3.0 GHz and Above
Output Devices	:	Monitor (LCD)
Input Devices	:	Keyboard
Hard Disk	:	1 TB
RAM	:	8GB or Above

3.2.2 Software Requirements

Scripting language	: Python Programming
Scripting Tool	: Anaconda Navigator (Jupyter Notebook) & Google Colab
Operating System	: Microsoft Windows 7, 8 or 10
Dataset	: UCI Machine Learning Repository
Machine Learning Packages etc.	: Numpy, Pandas, Matplotlib, Seaborn Packages

CHAPTER 4

SYSTEM ANALYSIS

Across the globe, there are lot of research is going on to address the different issues pertaining to the Threat from the insiders. Here in our proposed system, we have mentioned the various methods to resolve the issue. The process of detecting the insider and misbehaviour der threat traces the fake alarms to the administrator whenever it finds the malicious activity of many employees in the organization.

Aim

To confrontation the aforesaid explained, we have tried to focus on the detection of malicious or non-malicious using deep learning model and Standard CERT CMU r5.2 Dataset and predicting the possibility of Malicious or Normal (Non-Malicious).

Existing System

- Server based log system
- Behavioural Risk Indicators using Antivirus software
- Consolidate and Analyse user behaviour by creating daily log reports
- Discover and understand privileged access through traffic monitoring system
- activities of firewall and access monitoring
- Proactively assess insider threat processes

Limitations of Existing System

- The methods mentioned above are based on the User behaviour and approaches are based on ML & DL. As mentioned in in of the research, the accuracy of the model using deep learning is 90%. We need to enhance the accuracy of the model.
- Secondly the Dataset used is CERT CMU r4.2, which is having the limitations over labelling the dataset values.

Proposed System

The proposed system consists of the procedure to detect the Insider Threat and finding the accuracy of the models. The proposed system consists of various stages of executions.

Initially the input, Dataset is collected from the CERT CMU (Computer Emergency Readiness Team Carnegie Mellon University) by version r5.2. The dataset are namely File.csv, logon.csv, http.csv, device.csv, email.csv. The second stage of the proposed system is importing the python APIs. Next stage is to pre-process the input dataset. As the dataset is not labelled, we need to apply the methodologies to convert the categorical data into similar to integer format and assign the label to the dataset. Once the labelling is done, we can train the deep learning model and dataset is divided into training and testing data based on some split ratios. The developed Models such as LSTM-CNN and Bidirectional LSTM-CNN are applied to get the highest accuracy of the model. The prediction can be shown as Malicious and Non-Malicious data values.

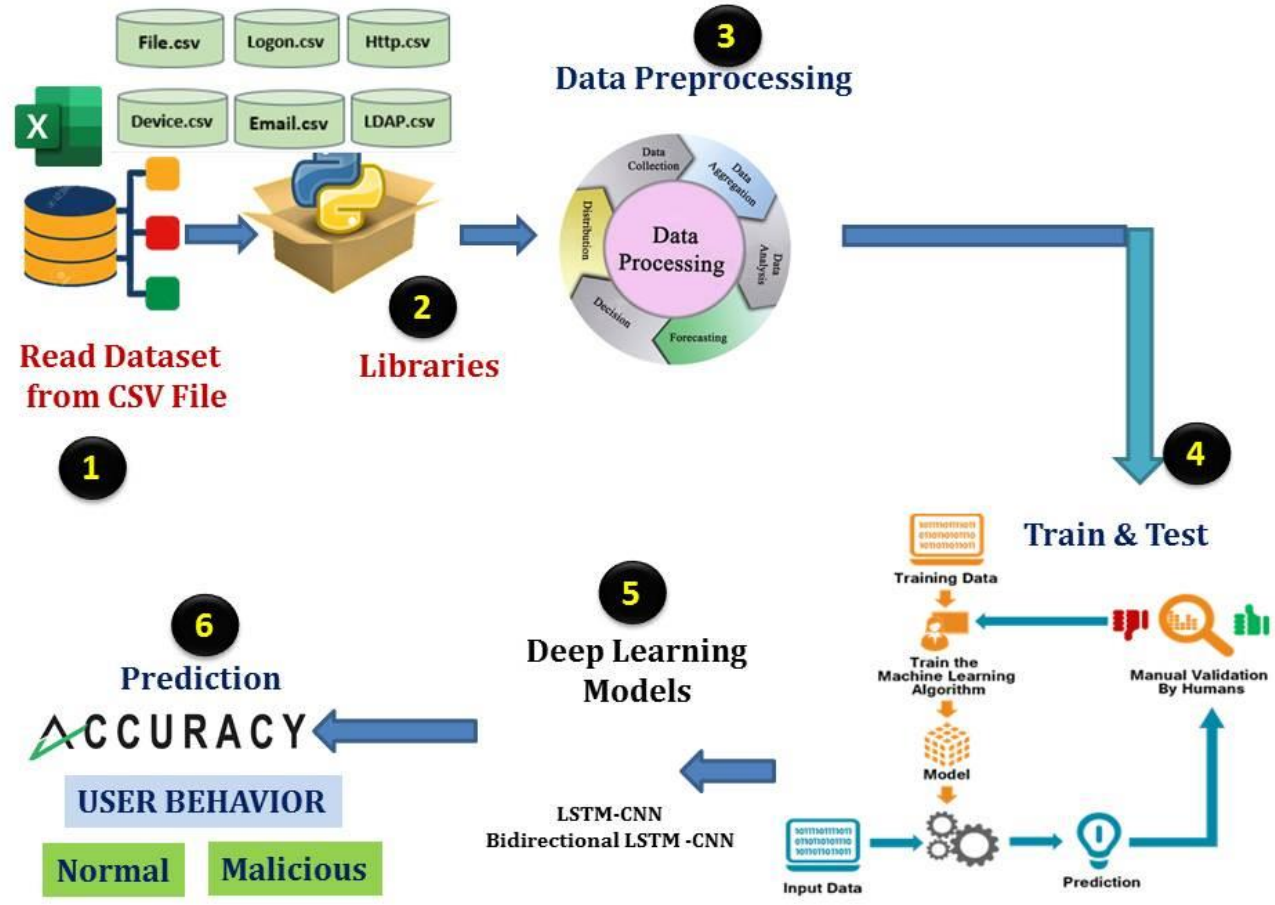


Figure 4.1. The overall structure of the Proposed System

Objectives

As per the above mentioned Aim of the project, herewith we have framed the objectives restricted within the organization and find the user behaviour within the organization.
The objectives of the project are categorized as:

1. Data Gathering or the collection of dataset for the study from the genuine source.
2. Data Preprocessing and Feature Extraction of the attributes which are important part of data analysis.
3. Building the Deep Learning models (LSTM-CNN and Bidirectional LSTM-CNN)
4. Finding the accuracy and loss of the model
5. Classification and Predictions
6. Exploratory Analysis on the Dataset and finding the various scenarios of data presentations.

Motivations of the Project

Cyber security issues are around the globe, where data security is the major issue. Most of the organizations are vulnerable to one or the other information leakage issues. The organizations are feeling safe from outside world in perspective of data security, but most important problem is insiders. They can harm the system at any point of time. So there are lot of motivation towards this project.

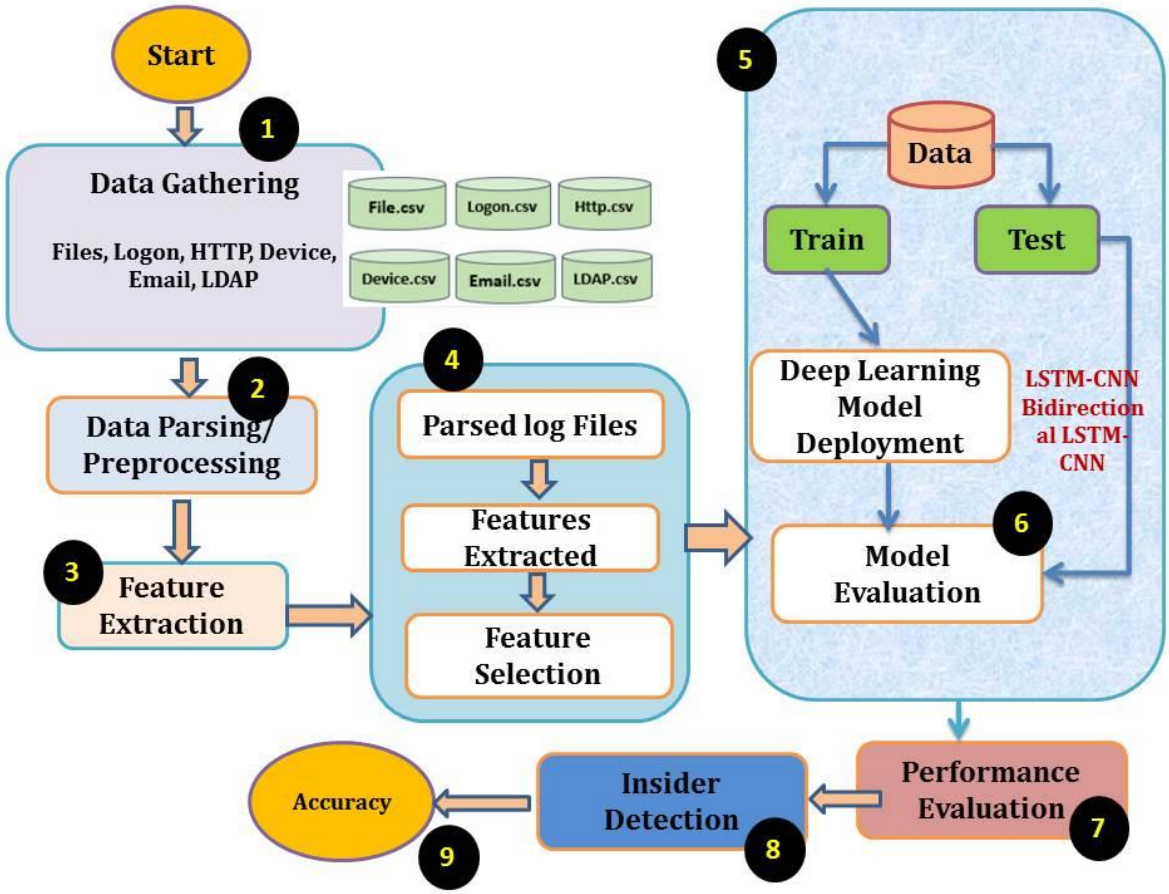
There are many approaches towards insider thread detection methodologies, such as Signature based user behaviour system, Machine Learning Algorithms, Anomaly detection, Classification, Graph based and sequence based learning system and others.

CHAPTER 5

SYSTEM DESIGN

5.1 Flow Diagram

The system design consists of the various stages as per the proposed system mentioned earlier.



1. Data Collection

The data acquisition is the first phase of every system design. Here we have used the existing Dataset from the Carnegie Mellon University CMU-CERT with various versions of version r1 to version r6.Basically the information present in the dataset comprises of Malicious and Non-Malicious User's Activities. We have collected the data and converted data into .csv files (Comma Separated Value). The next step is Data extraction before producing the data to the model development. We have taken 1000 Instances of data from each dataset, according

to the survey collected, among 1000 users, 70 users are malicious insiders.

- logon.csv: Information about the User's Login and Logout
- device.csv: The users connecting and disconnecting to the External USB devices
- http.csv: The complete browsing history information is recorded here.
- Email.csv: Email information like sent, received, cc, Bcc etc.
- file.csv: Information pertaining to the file transfer (Sent and Received from the system)
- Psychometric.csv: Users personal features.
- LDAP (Lightweight Directory Access Protocol): Directory service & Job roles and files related to users.

2. Data Preprocessing

Before feeding data to the model, the information or the input need to be preprocessed, we are applying various schemes of preprocessing like cleaning the data, removing irrelevant rows and columns, data abstraction and final data aggregation. We are parsing all the above mentioned .csv files, aggregate files and convert into preprocessed_filename.csv format. Finally we are creating the master preprocessed csv files. The feature engineering process can be applied on both textual and numerical type of data.

3. Feature Extraction

Feature extraction is a process of renovating the fresh input data into the integral attributes and the main thing is we must preserve the data during the extraction phase. The data integrity, data confidentiality and storage are important and retain the original information of dataset. The raw data is processed at each stage and cleaned attributes which required for the study are extracted.

First stage in feature extraction refers to the parsing of data; the important attributes are extracted, which are useful in predicting the behaviour of the users. Usually the working hours for the first shift for the employee are around 8:00AM to 7: 00PM. If any user login with the credentials after the office hour and login from the different computer, that activity can be logged and considered as malicious activity. Some of the features like Day, Time,

Personal Computer, User identification Number, User roles, Functional Units; Departments are extracted for the study of our project.

Some of the Feature Extraction table are mentioned below:

Features	Values
Day	0-6
Time	1-24
Activity	1-7
User_id	1-1000
User_role	1-42
User_functional_unit	1-6
User_department	1-7
PC	Unique number

Table 5.1 Feature value

Activity	Label
Logon	1
Logoff	2
Connect	3
Disconnect	4
E-mail	5
File	6
Http	7

Table 5.2 Activity labels

User_Functinal_Unit	Label
Administration	1
Research And Engineering	2
Manufacturing	3
Finance	4
Sales And Marketing	5
Purchasing And Contracts	6

Table 5.3 User functional Unit encoding

4. Model Building

The deep learning models we are implementing here are, Namely LSTM-CNN and Bidirectional LSTM-CNN. We have used CNN layers for feature extraction and LSTM module for the prediction. CNN LSTM is a powerful category of deep learning model that is used for deep data learning for both spatially and temporally dataset and performs all the input and output. CNN uses a process called Convolution in finding the relationship between

the two variable or functions. The function shape can be modified

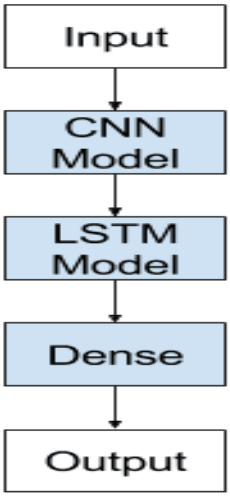


Figure 5.1 LSTM-CNN Model Structure

5. Dataflow Diagram of CNN-LSTM Module

The flow diagram shows the different units like Employee information, Data Processing Unit, Feature Extraction module, LSTM Module, Decision maker and prediction unit. The Employee module contains all the csv files of device, email, login/ logout. Websites or urls visited, file transfer information and psychometric data. The second stage is data preprocessing, where we apply various schemes starting from data collection to the data segmentation. Next stage is Feature extraction, which is also an important unit in data processing that extract the important features required for the CNN-LSTM model as an input. LSTM can have many layers (Hidden Layers) LSTM1, LSTM2 till LSTMn. Decision maker gets the output from each layer, combine it and send as input to the prediction unit. The prediction Unit shows the outcome as Malicious or Normal.

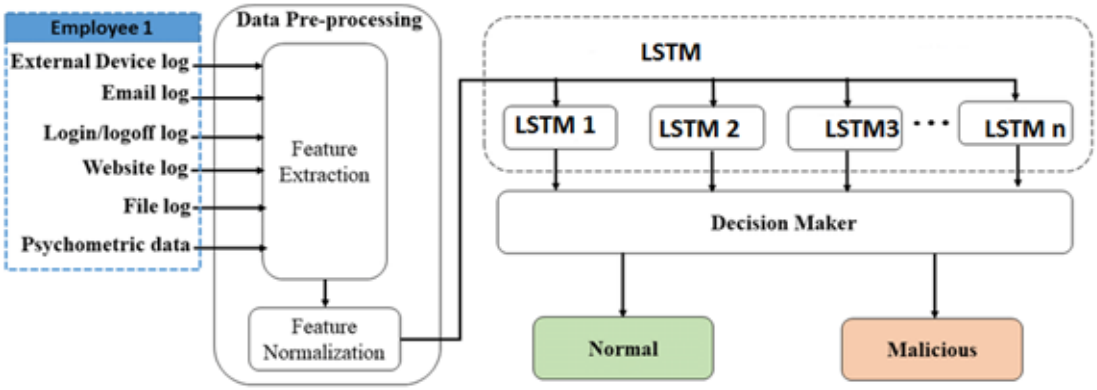


Figure 5.2 LSTM class diagram

6. Prediction

The two algorithms Namely CNN-LSTM and Bidirectional LSTM-CNN are used and model is deployed. Accuracy of the both models is 100%. Also we used traditional classifier called Support Vector Machine (SVM). The prediction shows that whether the users are malicious or Normal in behaviour.

Sequence Diagram

The sequence diagram is a systematic approach to show the relationship and interactions between objects and every object or entity is executed sequentially. Sequence of occurrence of events can also be depicted by using the sequence diagram. Sequence diagram is mostly used in Software Development models, Business Models and making the SRS of the project.

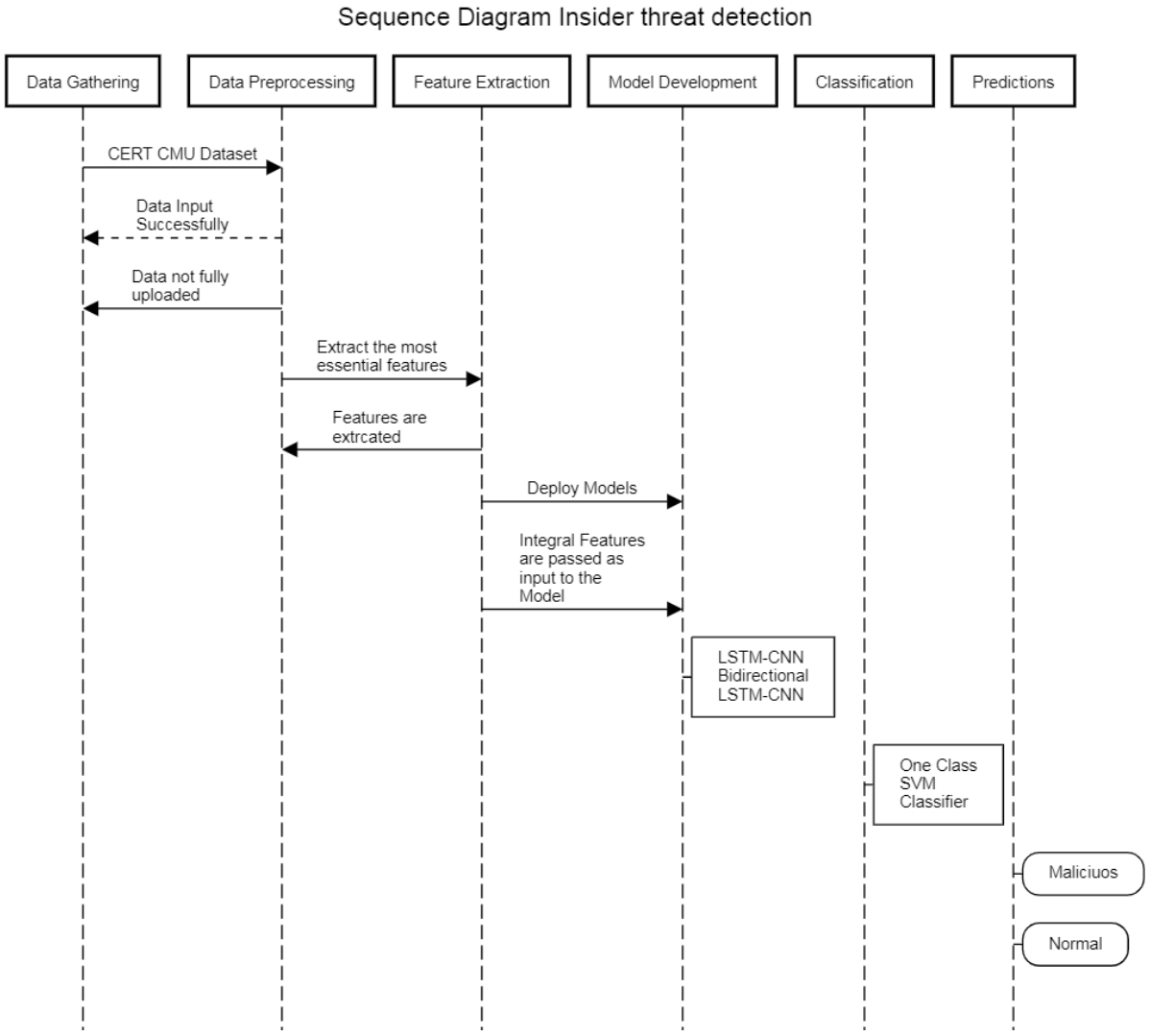


Figure 5.3 Sequence diagram

- The various components involved in the process of sequence data processing
- Data Gathering or Data Acquisition
- Data Preprocessing & Procedure
- Feature Extraction / Feature Extraction
- Model Deployment using CNN-LSTM and Bidirectional CNN_LSTM
- Model Classification using SVM
- Predictions

CHAPTER 6

SYSTEM IMPLEMENTATION

System Implementation is basically taken the input from system definition, the system definitions are the actual requirements of the project. System definitions consist of System requirements (Functional and Non-Functional requirements), the architectural Design, Design Features, Specifications and complete blueprint of development.

The System Implementation is the process where user has to implement the application with the help of all components, defined at design phase of the project. The implementation phase is the final phase of project development where the system is executed in a Real-time environment.

The ideology of system development follows many questions as:

- Need Existing System or need to design new Proposed System
- How to build the system / Application.
- What are the Frontend, Backend and Datasets Design tools can be used.
- How all the components can be interconnected?
- What are the interfaces to build the application
- Which are models need to be selected
- Testing tools required to test the application.
- How to maintain the application
- What is the total costing of the project etc?

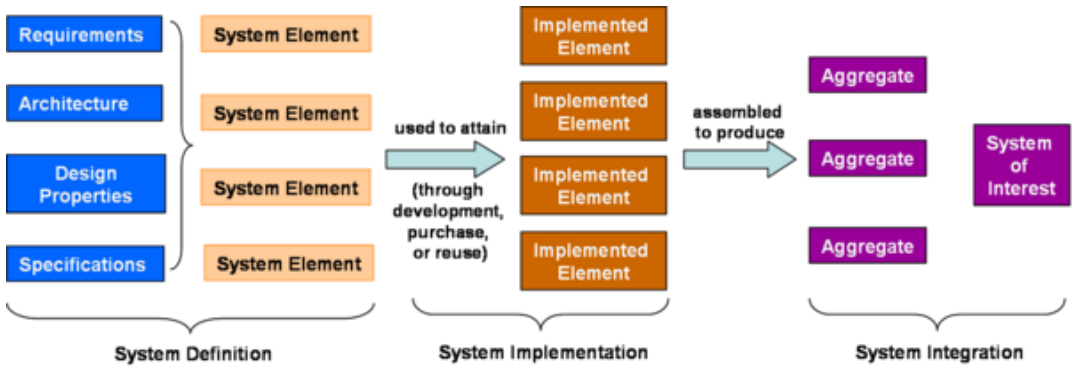


Figure 6.1 System Implementation Phases

The implementation can classified into different modules of project and are listed as:

Module 1: Data Gathering or Data Acquisition

The first phase of system implementation is Information processing or data gathering. We have downloaded the required data from CMU-CERT, Carnegie Mellon University.

The dataset can be downloaded from the resource:

: https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=508099

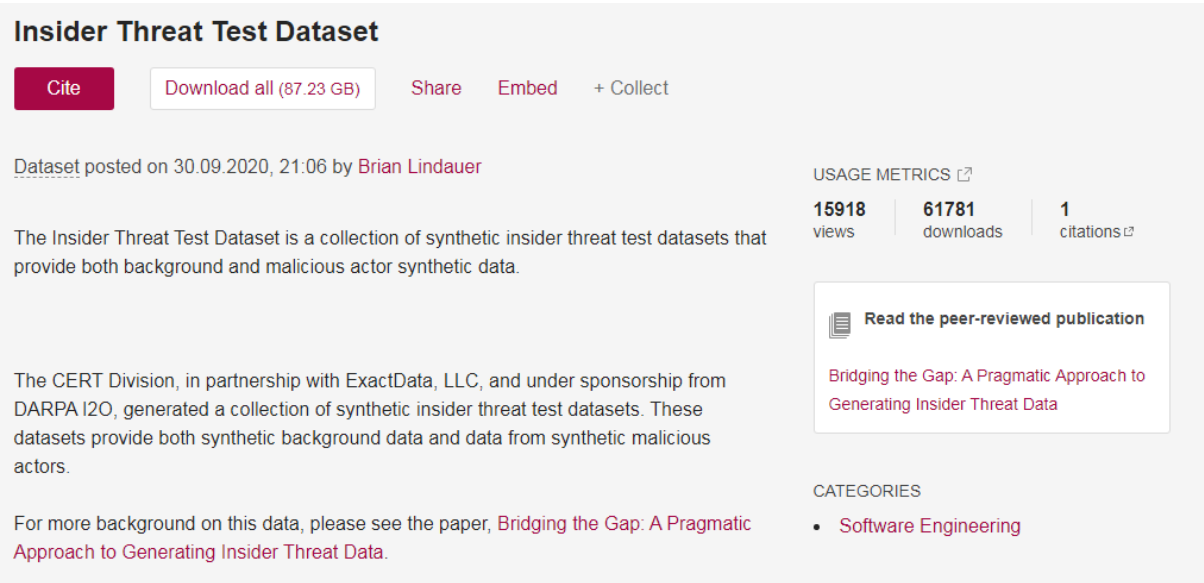


Figure 6.2 CERT Dataset download

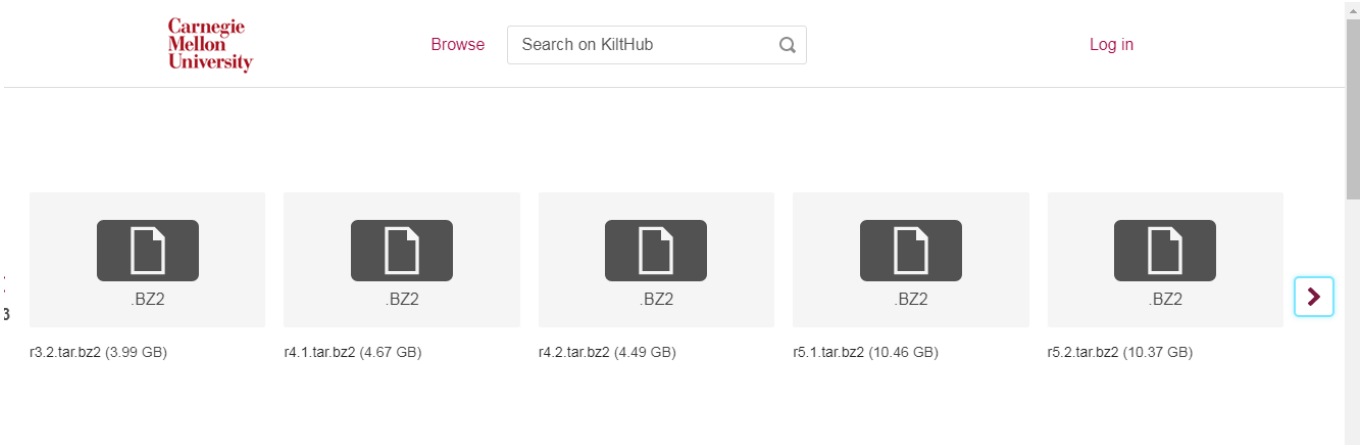


Figure 6.3 r5.2 Dataset

The CERT division is expert team and frontrunner in Cryptography and Cyber Security solutions. They are the backend support of IT industries, Government Agencies, Academic Excellence, Law Enforcement and many sectors where they are working on the integrity and data availability of systems and communication networks.

The dataset has various versions for finding the insider threat detection i.e. r1 to r6. Here in our research problem, we have used the version r5.2. The dataset contains the 1000 instances or rows. Each .csv file contains 1000 user information. The details about the dataset attributes are shown in below figures.

.

record_id	user_name	user_id	email	role
103	Kirk Dustin Lancaster	KDL1901	Kirk.Dustin.Lancaster@dtaa.com	ProductionLineWorker
104	Louis Kirk Logan	LKL1895	Louis.Kirk.Logan@dtaa.com	ProductionLineWorker
105	Baker Grant Rodriguez	BGR0917	Baker.Grant.Rodriguez@dtaa.com	ITAdmin
106	Madonna Shellie Shannon	MSS0799	Madonna.Shellie.Shannon@dtaa.com	TestEngineer
107	Patience Heather Yates	PHY0730	Patience.Heather.Yates@dtaa.com	HardwareEngineer
108	Martina Macey England	MME1034	Martina.Macey.England@dtaa.com	ProductionLineWorker
109	Nelle Regan Roman	NRR1835	Nelle.Regan.Roman@dtaa.com	Salesman

Figure 6.4: User Personal Data

record_id	datetime	user_id	pc	activity
310	02/01/2010 10:02:11	GTC0614	PC-6556	Logon
311	02/01/2010 10:07:04	AES1373	PC-6944	Logon
312	02/01/2010 10:10:53	GTP1369	PC-6531	Logon
313	02/01/2010 10:11:55	GTP1369	PC-6531	Logoff
314	02/01/2010 10:14:27	AYG1697	PC-6531	Logon
315	02/01/2010 10:17:26	AYG1697	PC-6531	Logoff
316	02/01/2010 10:18:01	ARB0626	PC-6531	Logon

Figure 6.5 Login and Logout Information

record_id	datetime	user_id	pc	activity
795	02/01/2010 11:27:20	ASR0150	PC-8653	Connect
796	02/01/2010 11:27:27	SMK1323	PC-4219	Disconnect
797	02/01/2010 11:27:28	MEB1743	PC-4130	Connect
798	02/01/2010 11:27:39	LPL0877	PC-9216	Disconnect
799	02/01/2010 11:28:09	JVH1575	PC-5682	Connect
800	02/01/2010 11:28:15	KAJ1413	PC-4500	Connect
801	02/01/2010 11:28:29	EKS1182	PC-4175	Disconnect

Figure 6.6 Device Activities / Device Logs

record_id	datetime	user_id	pc	url
327655	01/04/2011 20:26:41	WJM1922	PC-3793	http://mylife.com/
327656	01/04/2011 20:27:40	SYH1902	PC-4421	http://tmz.com/
327657	02/04/2011 06:00:51	ELT1370	PC-1929	http://wikileaks.org/
327658	02/04/2011 07:19:19	IGG1571	PC-3866	http://lockerz.com/
327659	02/04/2011 07:20:47	ANH1583	PC-7133	http://networksolutions.com/
327660	02/04/2011 07:21:03	JIG1593	PC-3301	http://zendesk.com/
327661	02/04/2011 07:24:06	BIS1598	PC-8485	http://digitalpoint.com/

Figure 6.7 Browsing History / http Logs

record_id	datetime	user_id	pc	file_extension
56599	02/02/2010 17:36:57	PMM1117	PC-9096	.doc
56600	02/02/2010 17:37:53	WTC0699	PC-9950	.pdf
56601	02/02/2010 17:38:32	CWR0696	PC-0613	.zip
56602	02/02/2010 17:40:16	PMM1117	PC-9096	.doc
56603	02/02/2010 17:40:20	TLB0894	PC-9738	.doc
56604	02/02/2010 17:40:44	OCW1127	PC-7876	.doc
56605	02/02/2010 17:41:26	TLB0894	PC-9738	.doc

Figure 6.8: Files Transferred Logs

record_id	datetime	user_id	pc	from	to	cc	bcc	size	attachments
992691	28/12/2010 09:29:53	CEM1385	PC-9104	Carlos.Elijah.	Gabriel.Joseph.	Leah_S_Michael	Carlos.Elijah.	310724	possible.exe
992692	28/12/2010 09:29:56	AGW1389	PC-6552	Amir.Giacom	Charles.Cullen.I		Cody.Lyle.Sal	1725276	listing.doc
992693	28/12/2010 09:29:58	AGW1389	PC-6552	Amir.Giacom	Gabriel.Joseph.	Cody.Lyle.Salina		29368	
992694	28/12/2010 09:29:58	BCB1715	PC-9816	Brenden.Cod	Blair.Hiram.Mid	Patience.Lesley.I		30913	
992695	28/12/2010 09:29:58	CBC1607	PC-3209	Colleen.Belle	Troy.Fulton.Sal			2700988	72.zip;curls.txt;
992696	28/12/2010 09:30:00	BCB1715	PC-9816	Brenden.Cod	Adele.Margaret	Lucius.Seth.Flyn		36603	
992697	28/12/2010 09:30:04	HDH1384	PC-6758	Helen.Darrel	Ivana_V_Sheph			393697	travelled.pdf

Figure 6.9 Email Logs

Module 2: Data Preprocessing and Feature Extraction

The Data preprocessing steps are as follows:

- **Import the packages required for data preprocessing and extraction**

```
import pandas as pd
import numpy as np
import sys
import os
from pathlib import Path
import re
from gensim.models import TfidfModel, nmf
from gensim.corpora import Dictionary as Dict
from gensim.models.ldamulticore import LdaModel
from multiprocessing import Pool
from functools import partial
from gensim.models.nmf_pgd import solve_h
```

Figure 6.10: Importing the libraries

- **Import the files: Dataset .csv files**

```
ALLFILES = ['logon.csv','device.csv', 'email.csv', 'file.csv', 'http.csv']
CONTENT_FILES = ['email.csv', 'file.csv', 'http.csv']
```

- **Create the Preprocessed Files**

```
def pre_process_logon(path):
    df = pd.read_csv(path)
    df['date'] = pd.to_datetime(df.date, format='%m/%d/%Y %H:%M:%S').dt.floor('D')
    df['day']=df['date'].dt.dayofweek

    self_pc = df \
        .groupby(['user', 'day', 'pc']).size().to_frame('count') \
        .reset_index().sort_values('count', ascending=False) \
        .drop_duplicates(subset=['user', 'day']) \
        .drop(columns=['count']).sort_values(['user', 'day']) \
        .groupby('user').pc.agg(pd.Series.mode).rename('self_pc')
    df = df.merge(self_pc.to_frame(), left_on='user', right_on='user')
    print("Done")
    print(df.head())
    #df['is_usual_pc'] = df['self_pc'] == df['pc']

    is_work_time = (8 <= df.date.dt.hour) & (df.date.dt.hour < 17)
    df['is_work_time'] = is_work_time

    df['subtype'] = df['activity']
    #df[['id', 'date', 'user', 'is_usual_pc', 'is_work_time', 'subtype']].to_csv(output_dir / 'logon_preprocessed.csv')
    df[['id', 'date', 'user', 'is_work_time', 'subtype']].to_csv(output_dir / 'logon_preprocessed.csv')
    return self_pc.to_frame()
```

Figure 6.11: Preporcessing logon_Preprocessed.csv

```
def pre_process_file(path):
    df = pd.read_csv(path, usecols=['id', 'date', 'user', 'pc', 'filename'])
    df['date'] = pd.to_datetime(df.date, format='%m/%d/%Y %H:%M:%S')

    df = df.merge(self_pc, left_on='user', right_on='user', )
    #df['is_usual_pc'] = df['self_pc'] == df['pc']

    is_work_time = (8 <= df.date.dt.hour) & (df.date.dt.hour < 17)
    df['is_work_time'] = is_work_time

    file_extensions = df.filename.str[-4:]
    df['subtype'] = file_extensions
    #df[['id', 'date', 'user', 'is_usual_pc', 'is_work_time', 'subtype']].to_csv(
    #    output_dir / f'file_preprocessed.csv')
    df[['id', 'date', 'user', 'is_work_time', 'subtype']].to_csv(
        output_dir / f'file_preprocessed.csv')
```

Figure 6.12: Preporcessing file_preprocessed.csv

```
def pre_process_email(path):
    df = pd.read_csv(path, usecols=['id', 'date', 'user', 'pc', 'to', 'cc', 'bcc', 'from'])
    df = df.fillna('')
    to_concated = df[['to', 'cc', 'bcc']].apply(lambda x: ';'.join([x.to, x.cc, x.bcc]), axis=1)
    is_external_to = to_concated.apply(
        lambda x: any([re.match('^.+@(.+$)', e).group(1) != 'dtaa.com' for e in x.split(';') if e != '']))
    is_external = is_external_to | is_external_to
    df['date'] = pd.to_datetime(df.date, format='%m/%d/%Y %H:%M:%S')

    df = df.merge(self_pc, left_on='user', right_on='user', )
    df['is_usual_pc'] = df['self_pc'] == df['pc']

    is_work_time = (8 <= df.date.dt.hour) & (df.date.dt.hour < 17)
    df['is_work_time'] = is_work_time

    df['subtype'] = is_external
    df[['id', 'date', 'user', 'is_usual_pc', 'is_work_time', 'subtype']].to_csv(
        output_dir / f'email_preprocessed.csv')
```

Figure 6.13: Preporcessing email_preprocessed.csv

```
def pre_process_http(path):

    # scenario 1
    scenario_1_http = [
        'actualkeylogger.com',
        'best-spy-soft.com',
        'dailykeylogger.com',
        'keylogpc.com',
        'refog.com',
        'relytec.com',
        'softactivity.com',
        'spectorsoft.com',
        'webwatchernow.com',
        'wellresearchedreviews.com',
        'wikileaks.org'
    ]
```

Figure 6.13: Preporcessing http_preprocessed.csv (Scenario 1)

```
    # scenario 2
    scenario_2_http = [
        'careerbuilder.com',
        'craiglist.org',
        'indeed.com',
        'job-hunt.org',
        'jobhuntersbible.com',
        'linkedin.com',
        'monster.com',
        'simplyhired.com',
    ]


    #scenario 3
    scenario_3_http = [
        '4shared.com'
        'dropbox.com',
        'fileserve.com',
        'filefreak.com',
        'filestube.com',
        'megaupload.com',
        'thepiratebay.org'
    ]
```

Figure 6.13: Preporcessing http_preprocessed.csv (Scenario 2 & 3)

- **Merging all contents from the .csv processed files**

```
def merge_all_content():
    df_dict = Dict(chunk_iterator(dataset_dir / 'email.csv'))
    df_dict.add_documents(chunk_iterator(dataset_dir / 'file.csv'))
    df_dict.add_documents(chunk_iterator(dataset_dir / 'http.csv'))

    df_dict.save((output_dir / 'dict.pkl').as_posix())
```

Figure 6.13: Merging Contents and converting to .pkl file

- **Making Text Classification Models**

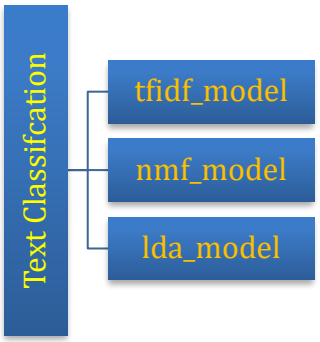


Figure 6.14: Model for Textual Classification

```
def make_tfidf_model():
    tfidf_model = TfidfModel(
        tfidf_iterator(CONTENT_FILES, Dict.load((output_dir / 'dict.pkl').as_posix())))

    tfidf_model.save((output_dir / 'tfidf_model.pkl').as_posix())


def make_nmf_model():
    tfidf_model = TfidfModel.load((output_dir / 'tfidf_model.pkl').as_posix())
    nmf_model = nmf.Nmf(
        nmf_iterator(CONTENT_FILES, Dict.load((output_dir / 'dict.pkl').as_posix()),
                     tfidf_model), num_topics=TOPIC_NUM)
    nmf_model.save((output_dir / 'nmf_model.pkl').as_posix())


def make_lda_model():
    tfidf_model = TfidfModel.load((output_dir / 'tfidf_model.pkl').as_posix())
    lda_model = LdaModel(
        nmf_iterator(CONTENT_FILES, Dict.load((output_dir / 'dict.pkl').as_posix()),
                     tfidf_model), num_topics=TOPIC_NUM)
```

Figure 6.15: Three Models: tfidf model, nmf model and lda model

The dataset contains raw data from the repository r5.2 and contains complete information about the users. All the activities are stored since from 18 months by the users. Various activities take an account of: Login / Logout, External Device connections, Browsing websites logs, Communication of files over the email and other information. The figure 6.16

depicts the overall framework of data repository and arrangements.

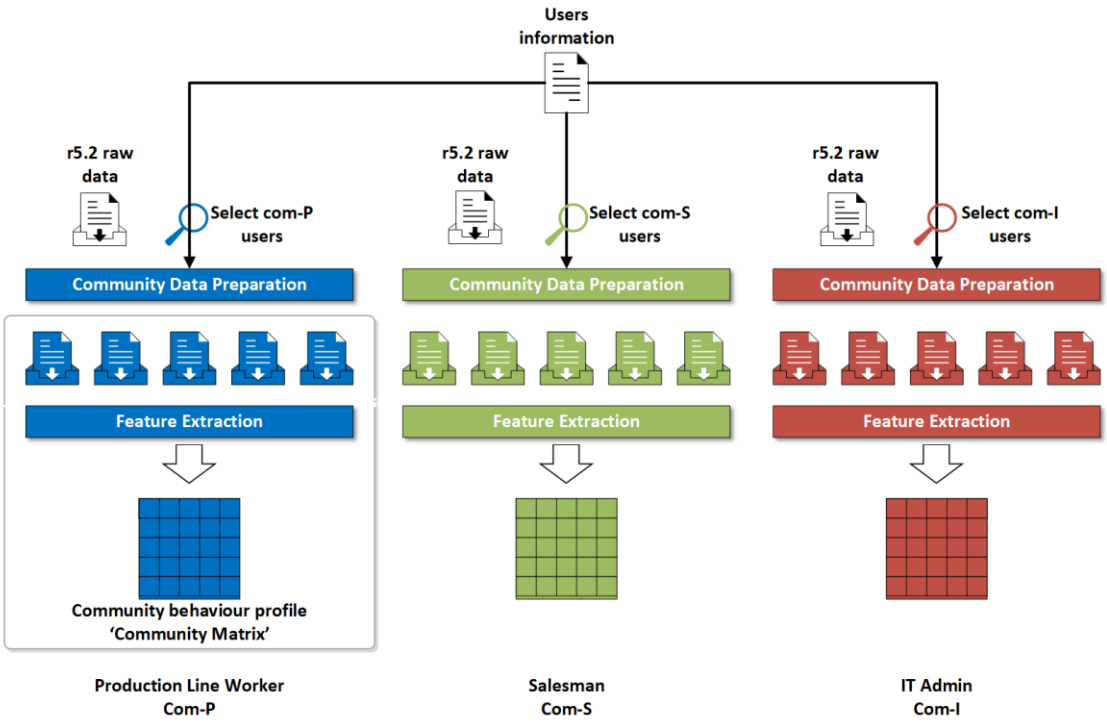


Figure 6.16 Preprocessing and Feature Extraction process

Module 3: Building the Deep Learning Models

1. Long Short-Term Memory networks (LSTMs)

LSTM-CNN is a variant of Recurrent Neural Network (RNN) mostly used in time series and gradient problems and long term convolutions. LSTM –CNN can be implemented on three basic layers Input Layer, Hidden Layer and Output Layer. When there is an autocorrelation in the input data, time series forecasting is very easily adopted by using LSTM. Here we are using LSTM stateful architecture for real time prediction and finding the accuracy of the model. The below figure shoes the cell arrangements and Gated Arrangements.

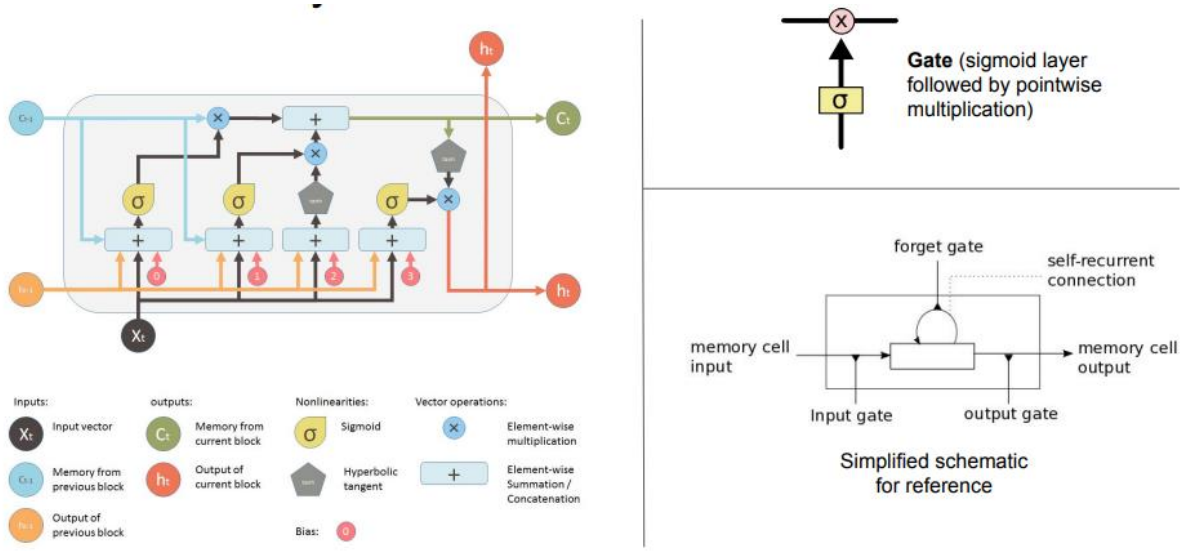


Figure 6.17 Memory Cell of LSTM

The model development and configurations of layer architecture can be deployed as follows:

```
# making the deep learning function
def model():
    model = models.Sequential()
    model.add(layers.Dense(256, activation='relu', input_shape=(X_train.shape[1],)))
    model.add(layers.Dense(128, activation='relu'))
    model.add(layers.Dense(64, activation='relu'))
    model.add(layers.Dense(2, activation='softmax'))

    model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

    regressor = Sequential()
    regressor.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train.shape[1], 1)))
    regressor.add(Dropout(0.2))
    regressor.add(LSTM(units = 50, return_sequences = True))
    regressor.add(Dropout(0.2))
    regressor.add(LSTM(units = 50, return_sequences = True))
    regressor.add(Dropout(0.2))
    regressor.add(LSTM(units = 50))
    regressor.add(Dropout(0.2))
    regressor.add(Dense(units = 1))
```

Figure 6.18 LSTM-CNN Model Configurations

2. Bidirectional LSTM-CNN

Whenever we need to design a model to run the sequence of information in either directions i.e. Forward Engineering and Backward Engineering, we use Bidirectional LSTM-CNN model. Here we can save the information of Past and Future values.

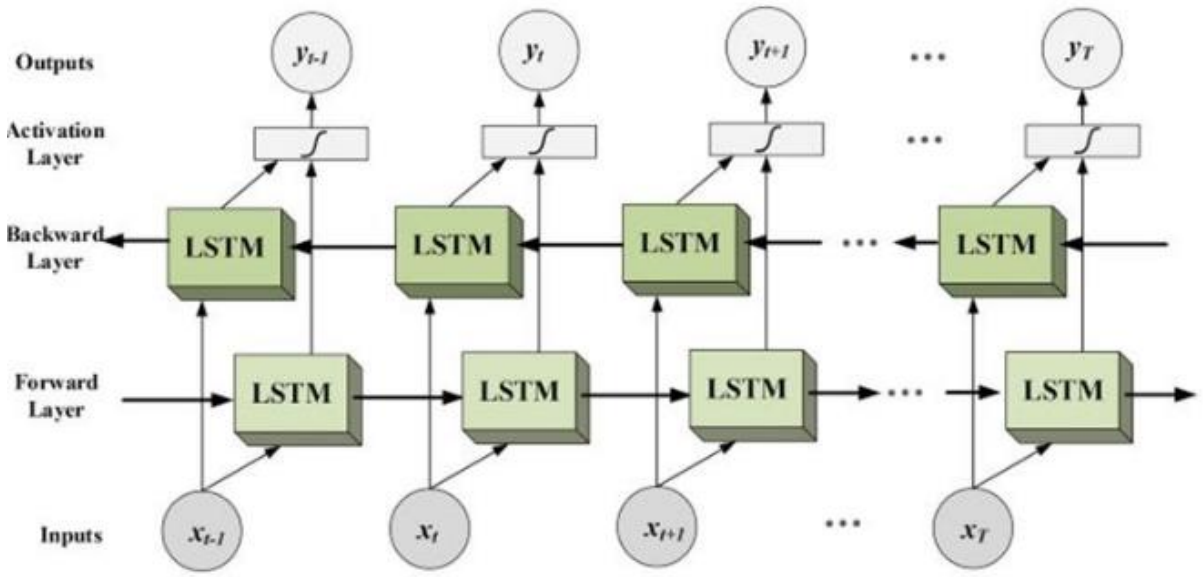


Figure 6.19: Bidirectional LSTM Model

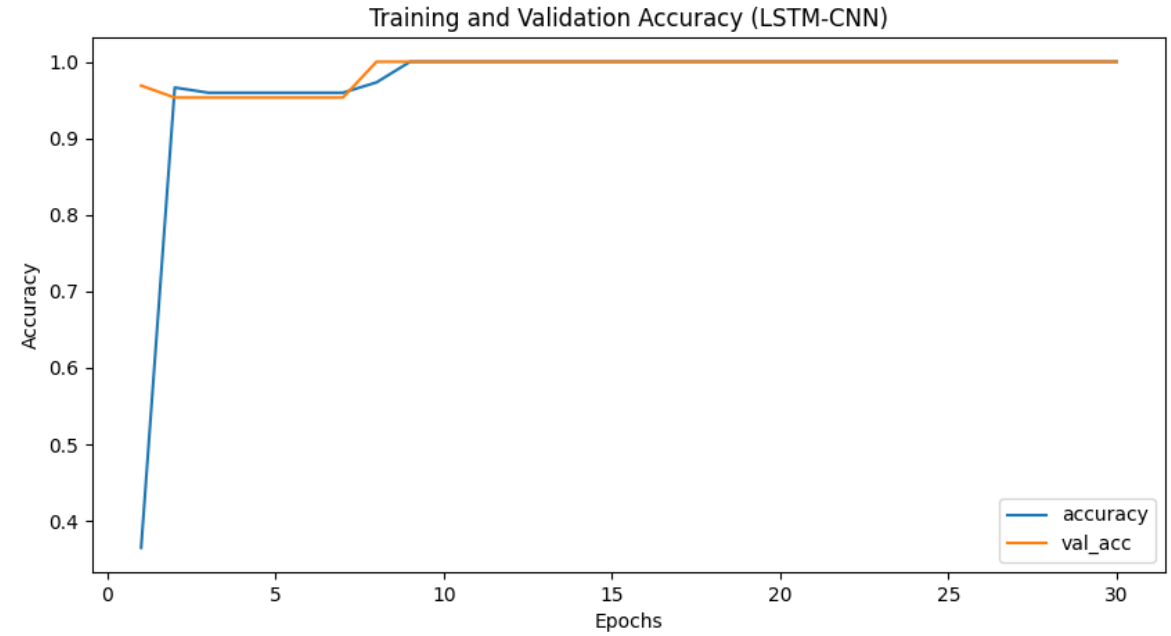
```
# making the deep learning function
def model2():
    model = models.Sequential()
    model.add(layers.Dense(256, activation='relu', input_shape=(X_train.shape[1],)))
    model.add(layers.Dense(128, activation='relu'))
    model.add(layers.Dense(64, activation='relu'))
    model.add(layers.Dense(2, activation='softmax'))

    model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])


    regressor = Sequential()
    regressor.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train.shape[1], 1)))
    regressor.add(Dropout(0.2))
    regressor.add(LSTM(units = 50, return_sequences = True))
    regressor.add(Dropout(0.2))
    regressor.add(LSTM(units = 50, return_sequences = True))
    regressor.add(Dropout(0.2))
    regressor.add(Bidirectional(LSTM(units = 50)))
    regressor.add(Dropout(0.2))
    regressor.add(Dense(units = 1))
```

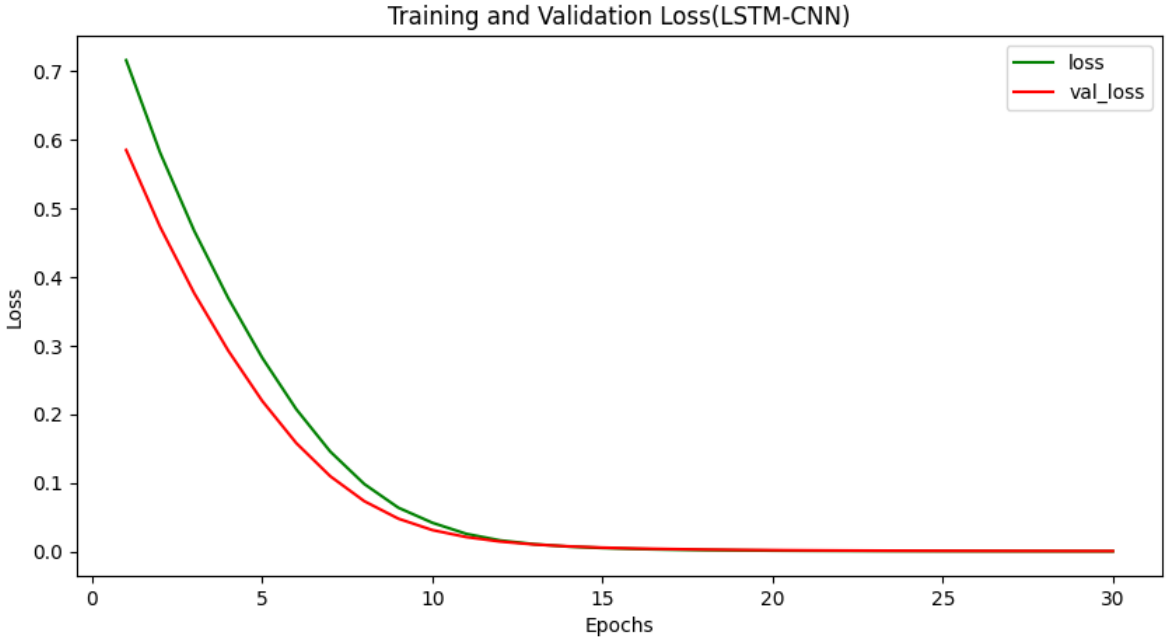
Figure 6.20: Bidirectional LSTM-CNN Model Configurations

Module 4: Prediction

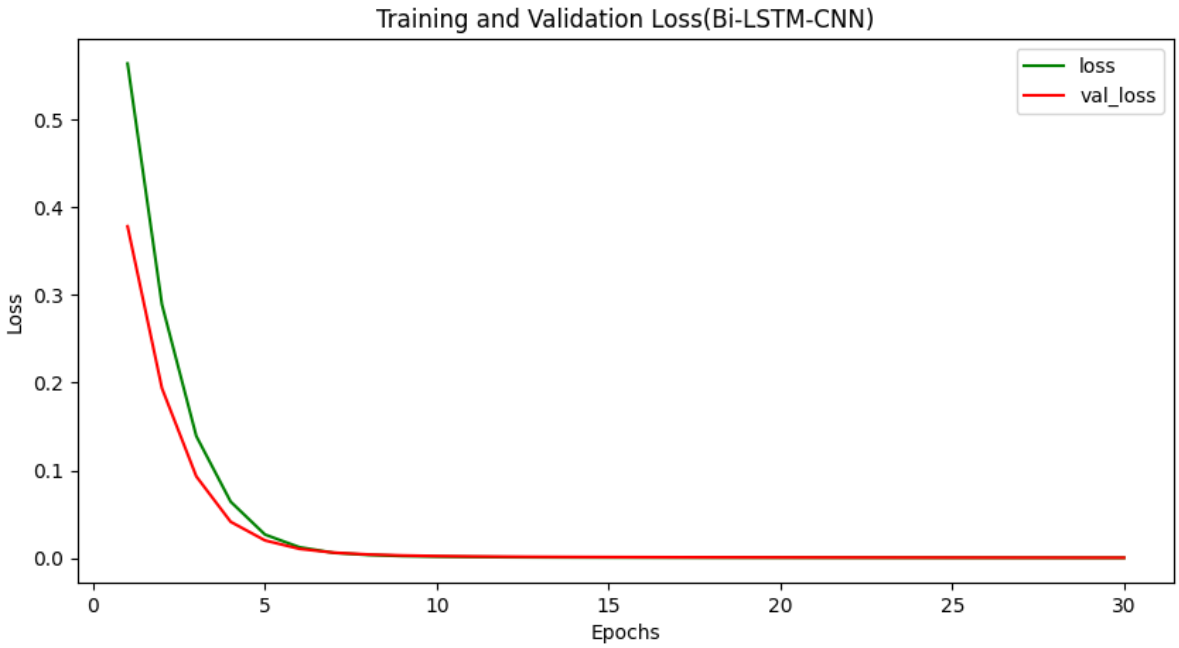
The prediction can be done on insider threat data and we trained the model using LSTM-CNN and Bidirectional LSTM-CNN. The outcome can be either malicious or non-malicious. The accuracy and loss for both the models are shown here.

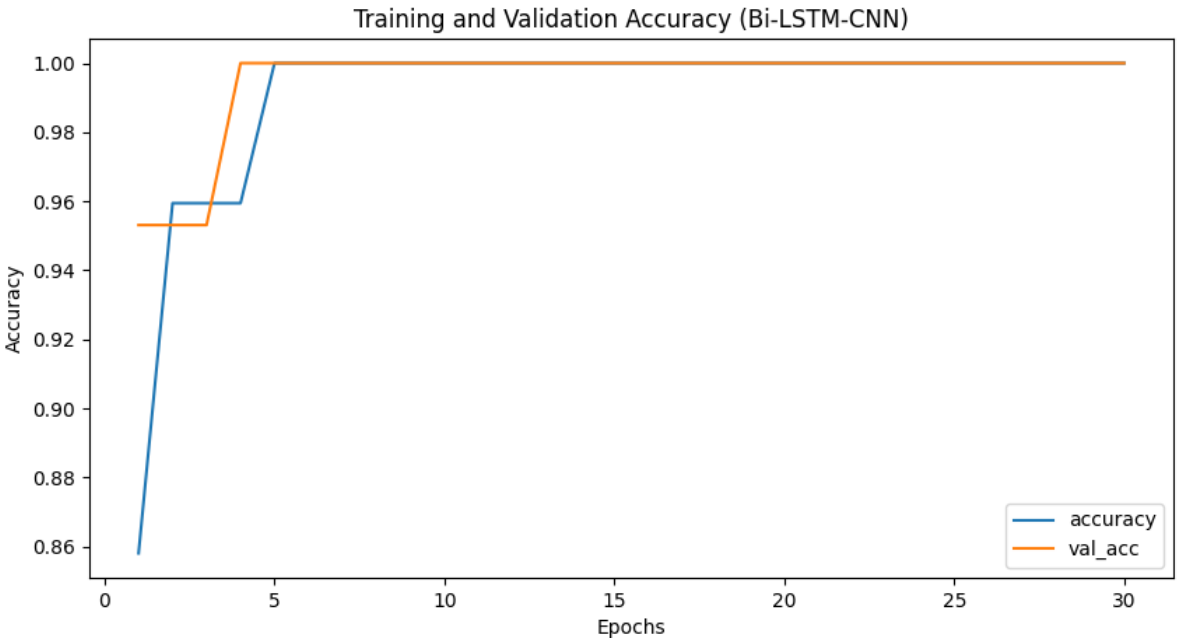
Results of LSTM-CNN Model





Results of Bidirectional LSTM-CNN Model


Training and Validation Loss(Bi-LSTM-CNN)
loss
val_loss
Loss
0.0
0.1
0.2
0.3
0.4
0.5
0
5
10
15
20
25
30
Epochs


Training and Validation Accuracy (Bi-LSTM-CNN)
accuracy
val_acc
Accuracy
0.86
0.88
0.90
0.92
0.94
0.96
0.98
1.00
0
5
10
15
20
25
30
Epochs

CHAPTER 7

SYSTEM TESTING

Testing is extremely important for quality assurance and ensuring the products reliability. The success of testing for programme flaws in largely determined by the experience. Testing might be a crucial component in ensuring the proposed systems quality and efficiency in achieving its goal. Testing is carried out at various phases of testing process with the goal of creating a system that is visible, adaptable and secure and tested the models. Testing is an important element of the software development process. The testing procedure verifies whether the generated product meets the requirements for which it was intended.

7.1 Test objectives

• Testing may be a defined as a process of running a programme with the goal of detecting a flaw.

• An honest case is one in which there is a good chance of discovering a mistake that hasn't been detected yet.

• A successful test is one that uncovers previously unknown flaw. If testing is done correctly, problems in the programme will be discovered. Testing cannot reveal whether or not flaws are present. It can only reveal the presence of software flaws.

7.2 Testing principles

A programmer must first grasp the fundamental idea that governs software testing before applying the methodologies to create successful test cases. All testing must be able to be tracked back to the customer's specification.

7.3 Testing design

Any engineering product is frequently put to the test in one of two ways:

7.3.1 White Box Testing

Glass container checking out is every other call for this kind of checking out. By understanding the necessary characteristic that the product has been supposed to do, checking out is regularly accomplished that proves every characteristic is absolutely operational at the same time as additionally checking for faults in every characteristic. The take a look at case layout technique that leverages the manage shape of the procedural layout to create take a

look at instances is used on this take a look at case.

7.3.2 Black Box Testing

Tests are regularly finished on this checking out via way of means of understanding the indoors operation of a product to make certain that each one gears mesh, that the indoors operation operates reliably in step with specification, and that each one inner additives had been nicely exercised. It is in most cases worried with the software's practical needs.

7.4 Testing Strategies

Testing might be a collection of actions that are prepared ahead of time and carried out in a methodical manner. As a result, a software testing template should be established as a set of stages in which particular test suit design techniques are defined for the software engineering process. The following characteristics should be included in every software testing strategy:

- Testing begins with the modules and extends to the mixing of the full computer-based system.
- At different periods in time, different testing approaches are applicable.
- Testing is carried out by the software's developer and an independent test group.

A software developer can use a software testing strategy as a route map. Testing might be a collection of actions that are prepared ahead of time and carried out in a methodical manner. As a result a software testing template should be established as a set of stages in which particular test suit design techniques are defined for the software engineering process. The following characteristics should be included in every software testing strategy:

Testing begins at the module level and progresses to entire computer based system are mixing.

- At different periods in time different testing approaches are applicable.
- Testing is carried out by the software's developer and a separate test group.

7.5 Levels of Testing

Testing is frequently omitted at various stages of the SDLC. They are as follows:

7.5.1 Unit Testing

Unit testing checks the tiny piece of software that makes up the module. The white box orientation of the unit test is maintained throughout. Different modules are tested alongside the requirements created throughout the module design process. The aim of unit testing is to inspect the inner logic of the modules, and it is used to verify the code created during development phase. It is usually done by the module's developer. The coding phase is sometimes referred to as coding and unit testing because of its tight association with coding. Unit tests for many modules are frequently run in simultaneously.

7.5.2 Integration Testing

Integration testing is a method of building a program's structure while running tests to find interface issues. Many tested modules are combined into subsystems and tested as a result of this. The purpose of this test is to see if all of the modules are properly integrated. Integration testing may be divided into three categories:

- **Top-Down Integration:** Top-Down integration is a method of gradually constructing a Programme structures. Modules are connected by working their way down the control Hierarchy, starting with the module having the most control.
- **Bottom-Up Integration:** Construction and testing using autonomous modules begin with Bottom-up integration, as the name suggests.
- **Regression Testing:** it is a subset of previously executed tests to ensure that Modifications have not propagated unexpected side effects during this competition of an Integration test strategy.

7.5.3 Functional Testing

The business and technical requirements, system documentation, and user guides all specify that functional tests must be conducted to ensure that the functions being tested are available. The following items are the focus of functional testing:

7.5.4 Validation Testing

Validation may be characterized in a lot of ways, however one easy definition is that validation is a hit whilst software program plays in a manner that clients may fairly expect. The affordable expectation is said with inside the software program requirement specification that is a record that lists all the software program's user-seen attributes. Validation standards are a segment of the specification. The statistics on this component serves as the premise for the validation trying out strategy.

7.5.5 Alpha Testing

Software developer can't know how a customer will utilise a programme ahead of time. Instructions to be utilised could be misconstrued, a peculiar combination of knowledge could be employed on a regular basis, and results that was clear to the tester could be unclear to a field user. It's impractical to conduct a formal acceptance test with all users if the programme is designed as a product that will be used by many people. Most software developers utilise alpha and beta testing to detect bugs that only the most experienced users seem to be aware of. At the developer's premises, a customer does the trial.

CONCLUSION

The project aim was to detect the insider threat detection using r5.2 CERT CMU dataset. We have taken various .csv log files to carry out the experiment. In the initial stage the Dataset used are not having the labels, so we applied the process of data mining techniques such as Pre-processing and Feature Extractions. We processed all the files and merged to form the new dataset for training, Testing and validation. The preprocessed dataset is used here to develop the models. The two Models LSTM-CNN and Bidirectional LSTM-CNN models are developed and found the accuracy of the models 100% respectively. The classification algorithm used is Support vector machine to show the classification and validation of dataset. We experimented the model and calculated the Accuracy and loss.