A

Major Project

On

CYBER THREAT DETECTION BASED ON ARTFICIAL NEURAL NETWORKS USING EVENT PROFILES

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

BANDARI SAIKUMAR (197R1A05C9)

GUDIMETLA SRIKANTH (197R1A05E0)

JOGANNAGARI SAINATH REDDY (197R1A05E2)

Under the Guidance of

A.UDAY KIRAN

(Assistant Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi) Recognized Under Section 2(f) & 12(B) of the UGC Act.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

2019-2023

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled **"CYBER THREAT DETECTION BASED ON ARTIFICIAL NEURAL NETWORKS USING EVENT PROFILES "** being submitted by **B.SAIKUMAR (197R1A05C9),G.SRIKANTH (197R1A05E0) & J.SAINATH REDDY (197R1A05E2)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2022-23.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

A.Uday Kiran
(Assistant Professor)
INTERNALGUIDE

Dr. A. Raji Reddy
DIRECTOR

Dr. K. Srujan Raju
HOD

EXTERNAL EXAMINER

Submitted for viva voice Examination held on _____

ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide **A.Uday Kiran**, Assistant Professor for his exemplary guidance, monitoring, and constant encouragement throughout the project work. The blessing, help, and advice given by him shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Project Review Committee (PRC) **Dr. Punyaban Patel**, **Ms. K. Shilpa**, **Dr. M . Subha Mastan Rao & J. Narasimharao** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju**, Head of the Department of Computer Science and Engineering, **Dr. Ashuthosh Saxena**, Dean R&D, and **Dr. D T V Dharmajee Rao**, Dean Academics for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy**, Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy**, Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

B. SAI KUMAR (197R1A05C9)

G.SRIKANTH (197R1A05E0)

J.SAINATH REDDY (197R1A05E2)

ABSTRACT

One of the major challenges in cybersecurity is the provision of an automated and effective cyber-threats detection technique. In this project, we present an AI technique for cyber-threats detection, based on artificial neural networks. The proposed technique converts multitude ofcollected security events to individual event profiles and use a deep learning-based detection method for enhanced cyber-threat detection. For this work, we developed an AI-SIEM system based on a combination of event profiling for data preprocessing and different artificial neural network methods, including FCNN, CNN, and LSTM. The system focuses on discriminating between true positive and false positive alerts, thus helping security analysts to rapidly respond to cyber threats. All experiments in this study are performed by authors using two benchmark datasets (NSLKDD and CICIDS2017) and two datasets collected in the real world. To evaluate the performance comparison with existing methods, we conducted experiments using the five conventional machine-learning methods (SVM, k-NN, RF, NB, and DT). Consequently, the experimental results of this study ensure that our proposed methods are capable of being employedas learning-based models for network intrusion-detection and show that although it is employed in the real world, the performance outperforms the conventional machine-learning methods.

LIST OF FIGURES/TABLES

LIST OF SCREENSHOTS

TABLE OF CONTENTS

1. INTRODUCTION

1.INTRODUCTION

1.1 PROJECT SCOPE

With the emergence of artificial intelligence (AI) techniques, learning-based approaches for detecting cyber attacks, have become further improved, and they have achieved significant results in many studies. However, owing to constantly evolving cyber attacks, it is still highly challenging to protect IT systems against threats and malicious behaviors in networks. Because of various network intrusions and malicious activities, effective defenses and security considerations were given high priority for finding reliable solutions.

1.2 PROJECT PURPOSE

Fourth, some hackers can deliberately cover their malicious activities by slowly changing their behavior patterns. Even when appropriate learning-based models are possible, attackers constantly change their behaviors, making the detection models unsuitable. Moreover, almost allsecurity systems have been focused on analyzing short- term network security events. To defend consistently evolving attacks, we assume that over long-term periods, analyzing the security event history associated with the generation of events can be one way of detecting the maliciousbehavior of cyber attacks.

1.3 PROJECT FEATURES

Our proposed system can help security analysts rapidly to respond cyber threats, dispersed across a large amount of security events. For this, the proposed the AI-SIEM system particularlyincludes an event pattern extraction method by aggregating together events with a concurrency feature and correlating between event sets in collected data. Our event profiles have the potentialto provide concise input data for various deep neural networks. Moreover, it enables the analystto handle all the data promptly and efficiently by comparison with longterm history data.

2. LITERATURE SURVEY

2.LITERATURE SURVEY

2.1 Enhanced Network Anomaly Detection Based on Deep Neural Networks

Due to the monumental growth of Internet applications in the last decade, the need for security of information network has increased manifolds. As a primary defense of network infrastructure, an intrusion detection system is expected to adapt to dynamically changing threat landscape. Many supervised and unsupervised techniques have been devised by researchers fromthe discipline of machine learning and data miningto achieve reliable detection of anomalies. Deep learning is an area of machine learning which applies neuron-like structure for learning tasks. Deep learning has profoundly changed the way we approach learning tasks by delivering monumental progress in different disciplines like speech processing, computer vision, and natural language processing to name a few. It is only relevant that this new technology must be investigated for information security applications. The aim of this paper is to investigate the suitability of deep learning approaches for anomaly-based intrusion detection system. For this research, we developed anomaly detection models based on different deep neuralnetwork structures, including convolutional neural networks, autoencoders, and recurrent neural networks. These deep models were trained on NSLKDD training data set and evaluated on both test data sets provided by NSLKDD, namely NSLKDDTest+ and NSLKDDTest21. All experiments in this paper are performed by authors on a GPU- based test bed. Conventional machine learning-based intrusion detection models were implemented using well-known classification techniques, including extreme learning machine, nearest neighbor, decision-tree, random-forest, support vector machine, naive- bays, and quadratic discriminant analysis. Both deep and conventional machine learning models were evaluated using well-known classification metrics, including receiver operating characteristics, area under curve, precision-recall curve, mean average precision and accuracy of classification. Experimental results of deep IDS modelsshowed promising results for real-world application in anomaly detection systems.

2.2 Network Intrusion Detection Based on Directed Acyclic Graph And BeliefRule Base

Intrusion detection is very important for network situation awareness. While a few methods have been proposed to detect network intrusion, they cannot directly and effectively utilize semi-quantitative information consisting of expert knowledge and quantitative data. Hence, this paper proposes a new detection model based on a directed acyclic graph (DAG) anda belief rule base (BRB). In the proposed model, called DAG-BRB, the DAG is employed to construct a multi-layered BRB model that can avoid explosion of combinations of rule number because of a large number of types of intrusion. To obtain the optimal parameters of the DAG- BRB model, an improved constraint covariance matrix adaption evolution strategy (CMA-ES) is developed that can effectively solve the constraint problem in the BRB. A case study wasusedto test the efficiency of the proposed DAG-BRB. The results showed that compared with other detection models, the DAG-BRB model has a higher detection rate and can beused in real networks.

2.3 HAST-IDS: Learning Hierarchical Spatial-Temporal Features Using Deep Neural Networks To Improve Intrusion Detection

The development of an anomaly-based intrusion detection system (IDS) is a primary research direction in the field of intrusion detection. An IDS learns normal andanomalous behavior by analyzing network traffic and can detect unknown and new attacks. However, the performance of an IDS is highly dependent on feature design, and designing a feature set that can accurately characterize network traffic is still an ongoing research issue. Anomaly-based IDSs also have the problem of a high false alarm rate (FAR), which seriously restricts their practical applications. In this paper, we propose a novel IDS called the hierarchical spatial-temporal features-based intrusion detection system (HAST-IDS), which first learns the low-level spatial features of network traffic using deep convolutional neural networks (CNNs) and then learns high-level temporal features using long short-term memory networks. The entireprocess of feature learning iscompleted by the deep neural networks automatically; no feature engineering techniques are required.

2.4 Data Security Analysis For Ddos Defense Of Cloud Based Networks

Distributed computing has become an effective approach to enhance capabilities of an institution or organization and minimize requirements for additional resource. In this regard, the distributed computing helps in broadening institutes IT capabilities. One needs to note that distributed computing is now integral part of most expanding IT business sector. It is considered novel and efficient means for expanding business. As more organizations and individuals start to use the cloud to store their data and applications, significant concerns have developed to protect sensitive data from external and internal attacks over internet. Due to security concern many clients hesitate in relocating their sensitive data on the clouds, despite significant interest in cloud-based computing. Security is a significant issue, since data much of an organizations data provides a tempting target for hackers and those concerns will continue todiminish the development of distributed computing if not addressed. Therefore, this study presents a new test and insight into a honeypot. It is a device that can be classified into two types: handling and research honeypots. Handling honeypots are used to mitigate real life dangers. A research honeypot is utilized as an exploration instrument to study anddistinguish the dangers on the internet. Therefore, the primary aim of this research project is to do an intensive networksecurity analysis through a virtualized honeypot for cloud servers to tempt an attacker and provide a new means of monitoring their behavior.

3.SYSTEM ANALYSIS

3.SYSTEM ANALYSIS

SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an importantroleof an interrogator and dwells deep into the working of the present system. In analysis,a detailed study of these operations performed by the system and their relationships withinand outside thesystem is done. A key question considered here is, “what must be done to solve the problem?” The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to bedone.

3.1 PROBLEM DEFINITION

It is still difficult to recognize and detect intrusions against intelligent network attacks owing to their high false alerts and the huge amount of security data. Hence, the most recent studies in the field of intrusion detection have given increased focus to machine learning and artificial intelligence techniques for detecting attacks. Advancement in AI fields can facilitate the investigation of network intrusions by security analysts in a timely and automated manner. These learning-based approaches require to learn the attack model from historical threat data and use the trained models to detect intrusions for unknown cyber threats.

3.2 EXISTING SYSTEM

Cyber security has recently received enormous attention in today’s security concerns, due to thepopularity of the Internet-of-Things (IoT), the tremendous growth of computer networks, and the huge number of relevant applications. Thus, detecting various cyber-attacks or anomalies in a network and building an effective intrusion detection system that performs an essential role in today’s security is becoming more important. However, many previous studies used benchmark dataset, which, thoughaccurate, are not generalizable to the real world because of the insufficient features. To overcome these limitations, an employed learning model requires to evaluate with datasets that are collected in the real world.Third,using anomly-base.

3.2.1 DISADVANTAGES OF EXISTING SYSTEM

- Data leakage is take place using Cyber attacks.
- It is less Secured from cyber attacks.

3.3 PROPOSED SYSTEM

we present an AI technique for cyber-threats detection, based on artificial neural networks. The proposed technique converts multitude of collected security events to individual event profiles and use a deep learning-based detection method for enhanced cyber-threat detection. For this work, we developed an AI-SIEM system based on a combination of event profiling for data preprocessing and different artificial neural network methods, including FCNN, CNN, and LSTM. The system focuses ondiscriminating between true positive and false positive alerts, thus helping security analysts to rapidly respond to cyber threat. we conducted experiments using the five conventional machine-learning methods (SVM, k-NN, RF, NB, and DT). Consequently, the experimental results of this study ensure that our proposed methods are capable of being employed as learning-based models for network intrusion-detection.

3.3.1 ADVANTAGES OF PROPOSED SYSTEM

- It is employed in the real world, the performance outperforms theconventionalmachine-learning methods.
- In this project data is secured from attacker.

3.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and a business proposal is putforthwith a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. Three key considerations involvedin the feasibility analysis:

- EconomicFeasibility
- TechnicalFeasibility
- SocialFeasibility

3.4.1 ECONOMIC FEASIBILITY

The developing system must be justified by cost and benefit. Criteria to ensure thateffortis concentrated on a project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

3.4.1.1 The costs conduct a full system investigation.
3.4.1.2 The cost of the hardware and software.
3.4.1.3 The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication thatthe system is economically possible for development.

3.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. The developed system must have a modest requirement, as onlyminimal or null changes are required for implementing this system.

3.4.3 BEHAVIORAL FEASIBILITY

This includes the following questions:

3.4.3.1 Is there sufficient support for the users?

3.4.3.2 Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developedand installed. All behavioral aspects are considered carefully and conclude that the projectis behaviorally feasible.

3.5 HARDWARE & SOFTWARE REQUIREMENTS

3.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specify the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements.

3.5.1.1 Processor : Pentium IV or higher processor

3.5.1.2 Hard disk : minimum 512MB space in Hard Disk.

3.5.1.3 RAM : 256 MB RAM

3.5.1.4 Input devices : Keyboard, mouse.

3.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface andsoftware components of the system. The following are some software requirements,

- Languages : Python,tkinter
- Operating system :Windows 8 and above
- Tools : Python IDEL33.7 version,vscode

4.ARCHITECTURE

4.PROJECT ARCHITECTURE

4.1 PROJECT ARCHITECTURE

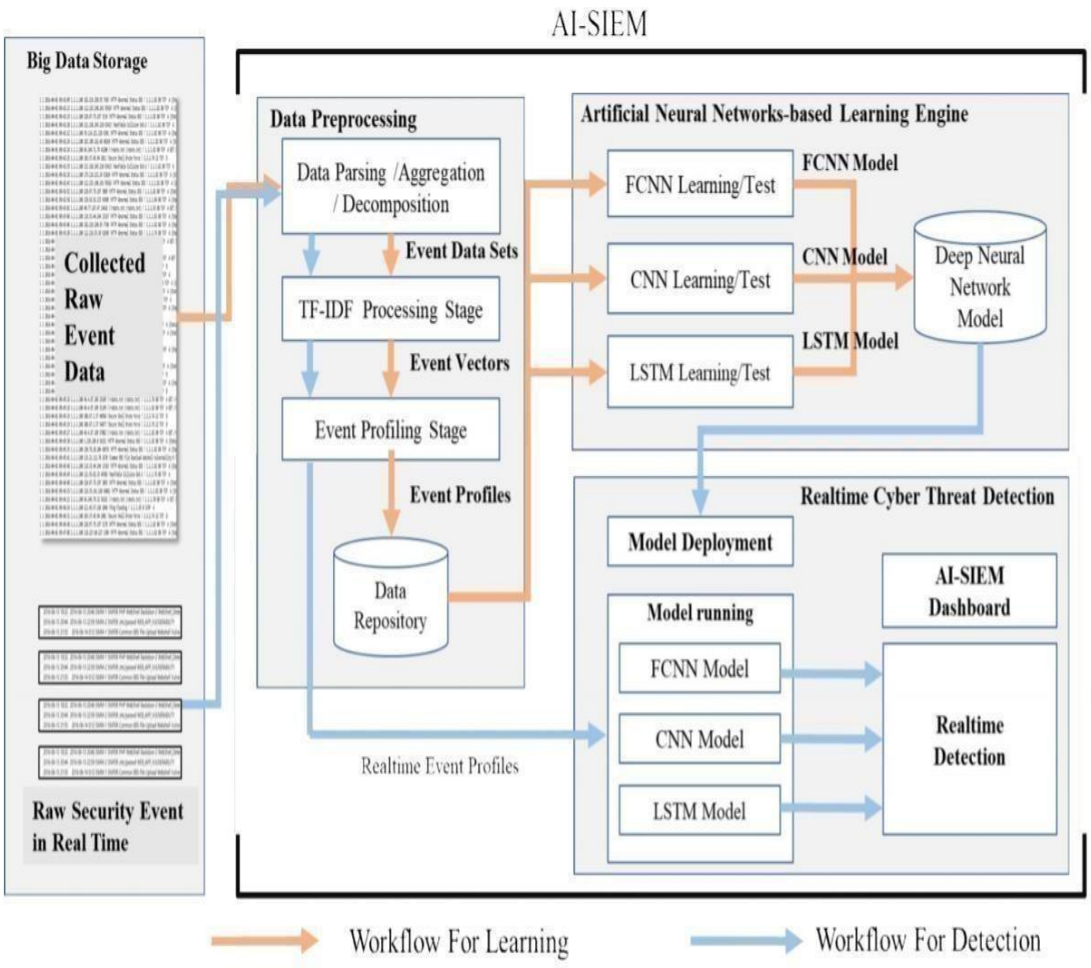


Figure 4.1 Project Architecture for cyber threat detection basedon artificial neural networks using event profile

4.2 DESCRIPTION

The workflow and architecture for the developed artificial intelligent (AI)-based SIEM system. The AI-SIEM system comprises three main phases: The data preprocessing, artificial neural networks-based learning engine, and real-time threat detection phase. The first preprocessing phase in the system, termed event profiling, aims at providing concise inputs for various deep neural networks by transforming raw data. In the data preprocessingphase, data aggregation with parsing, data normalization stage using TF-IDF mechanism, and event profiling stage are consecutively performed in the AI-SIEM system. Each stagegenerates event data sets, event vectors, and event profiles, respectively, and the output is utilized in next each stage, as shown in Figure. This phase not only precedes the data learning stage but also precedes the conversion of raw security events to the deep-learningengine's input data when the system operates on detecting network intrusions in real time.The second AI-based learning engine employs three artificial neural networks for modeling. For the data learning stage, the preprocessed data are fed into the three artificial neural networks, and each ANN performs learning to find the most accurate model. Finally, in real-time threat detection, each ANN model mechanically classifies each security raw event using the trained model, and the dashboard shows the only recognized true alerts to security analysts for reducing false ones.

4.3 USE CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (representedas use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

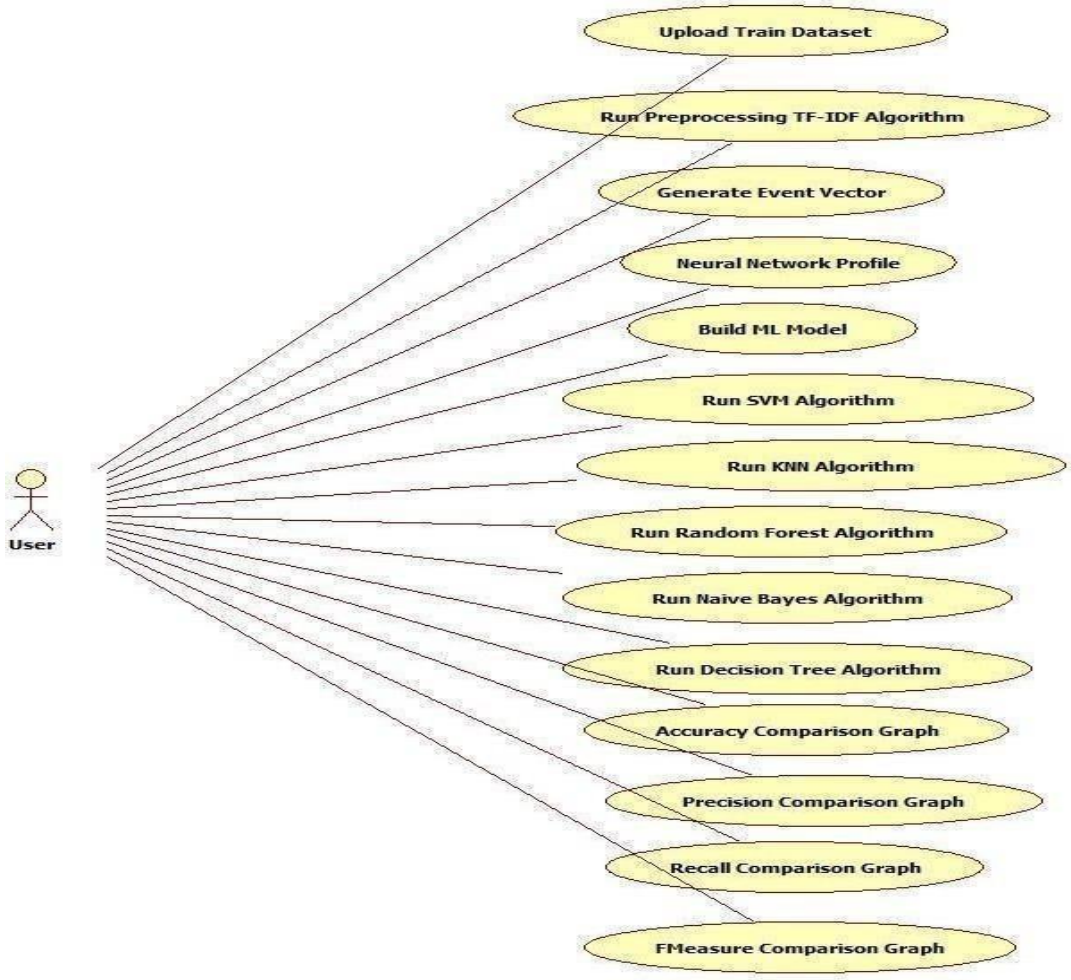


Figure 4.2 Use Case Diagram for cyber threat detection basedon artificial neural networks using event profiles

4.4 CLASS DIAGRAM

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

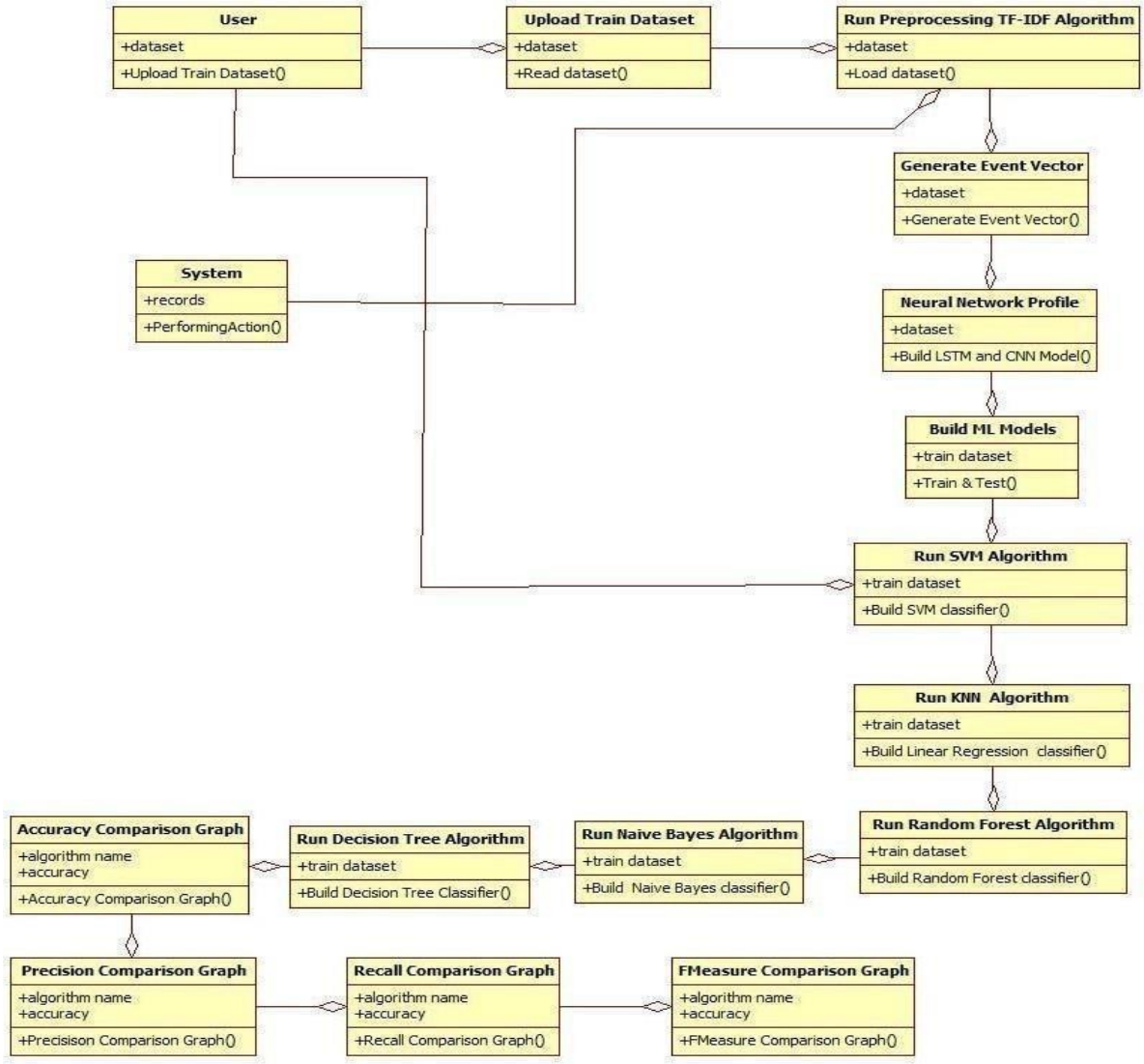


Figure 4.3 Class Diagram for cyber threat detection basedon artificial neural networks using event profiles

4.5 SEQUENCE DIAGRAM

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

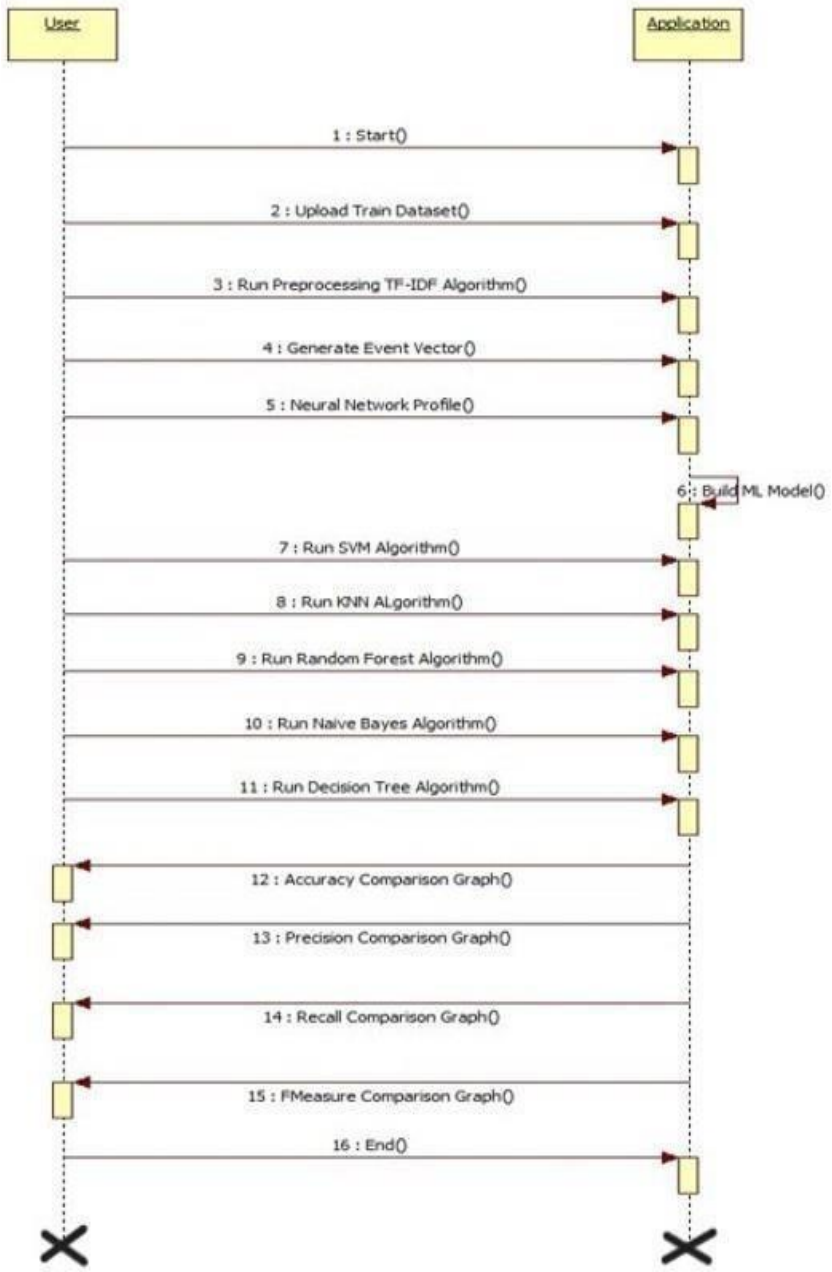


Figure 4.4 Sequence Diagram for cyber threat detection basedon artificial neural networks using event profiles

4.6 ACTIVITY DIAGRAM

The process flows in the system are captured in the activity diagram. Similar to a state diagram, an activity diagram also consists of activities, actions, transitions, initial and finalstates, and guard conditions.

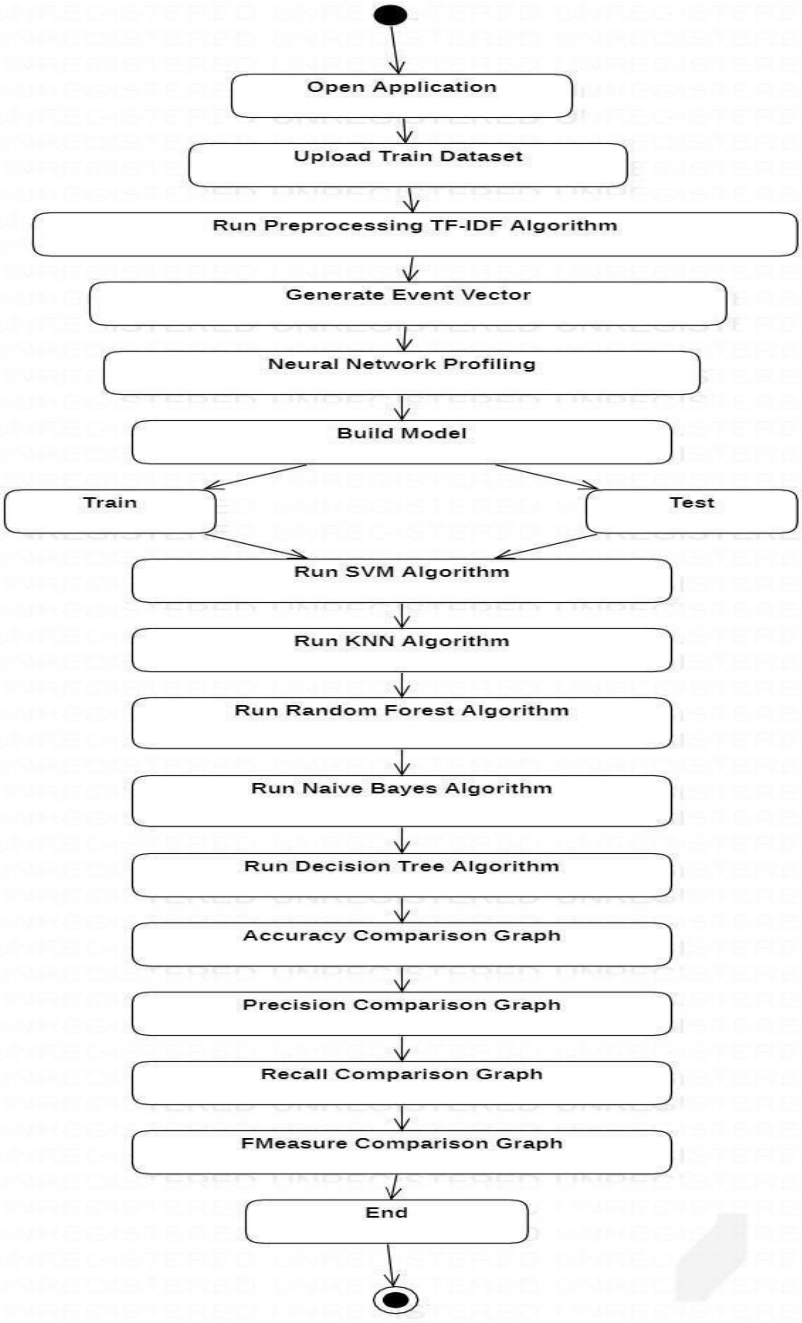


Figure 4.5 Activity Diagram for cyber threat detection basedon artificial neural networks using event profiles

5.IMPLEMENTATION

5.IMPLEMENTATION

5.1 SAMPLE CODE

```
from tkinter import messageboxfrom
tkinter import *
from tkinter import simpledialog
import tkinter
from tkinter import filedialog
import matplotlib.pyplot as plt
import numpy as np
from tkinter.filedialog import askopenfilename
import os
import pandas as pd
from sklearn import preprocessing
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizerfrom
sklearn import svm
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_splitfrom
keras.models import Sequential
from keras.layers import Flatten
from keras.layers import Dense,Activation,Dropout from
sklearn.preprocessing import OneHotEncoder import
keras.layers

from keras.layers import Convolution2D from
keras.layers import MaxPooling2D from
keras.layers import Flatten
from keras.layers import Dense,Activation,BatchNormalization,Dropout

from sklearn.metrics import precision_scorefrom
sklearn.metrics import recall_score from
sklearn.metrics import f1_score

from sklearn.naive_bayes import BernoulliNB
from sklearn.neighbors import KNeighborsClassifierfrom
sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier


main = tkinter.Tk()
main.title("Cyber Threat Detection Based on Artificial Neural Networks Using Event Profiles")
#designing main screen
main.geometry("1300x1200")
```

```
le = preprocessing.LabelEncoder()global
filename
global feature_extractionglobal
X, Y
global doc
global label_names
global X_train, X_test, y_train, y_test
global  lstm_acc,cnn_acc,svm_acc,knn_acc,dt_acc,random_acc,nb_accglobal
lstm_precision,cnn_precision,svm_precision,knn_precision,dt_precision,random_precision,nb_pre cision
global lstm_recall,cnn_recall,svm_recall,knn_recall,dt_acc,random_recall,nb_recallglobal
lstm_fm,cnn_fm,svm_fm,knn_fm,dt_fm,random_fm,nb_fm

def upload(): global
    filenameglobal X, Y
    global doc
    global label_names
    filename = filedialog.askopenfilename(initialdir = "datasets")
    dataset = pd.read_csv(filename)
    label_names = dataset.labels.unique() dataset['labels'] =
    le.fit_transform(dataset['labels'])cols = dataset.shape[1]
    cols = cols - 1
    X = dataset.values[:, 0:cols]Y =
    dataset.values[:, cols] Y =
    Y.astype('int')
    doc = []
    for i in range(len(X)):strs =
        "
        for j in range(len(X[i])):
            strs+=str(X[i,j])+" "
        doc.append(strs.strip())

    text.delete('1.0', END) text.insert(END,filename+'
    Loaded')
    text.insert(END,"Total dataset size : "+str(len(dataset)))

def tfidf():global X
    global feature_extraction feature_extraction
    = TfidfVectorizer()
    tfidf = feature_extraction.fit_transform(doc)X =
    tfidf.toarray()
    text.delete('1.0', END)
```

```
    text.insert(END,'TF-IDF processing completed')


def eventVector():
    global X_train, X_test, y_train, y_test
    X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)
    text.delete('1.0', END)
    text.insert(END,'Total unique events found in dataset are\n\n')
    text.insert(END,str(label_names)+"\n\n") text.insert(END,"Total
    dataset size : "+str(len(X))+"\n")
    text.insert(END,"Data used for training : "+str(len(X_train))+"\n")
    text.insert(END,"Data used for testing  : "+str(len(X_test))+"\n")

def neuralNetwork(): text.delete('1.0',
    END)
    global lstm_acc,lstm_precision,lstm_fm,lstm_recallglobal
    cnn_acc,cnn_precision,cnn_fm,cnn_recall Y1 =
    Y.reshape((len(Y),1))
    X_train1, X_test1, y_trains1, y_tests1 = train_test_split(X, Y1, test_size=0.2)
    print(X_train1.shape)
    print(y_trains1.shape)
    print(X_test1.shape)
    print(y_tests1.shape) enc =
    OneHotEncoder()
    enc.fit(y_trains1)
    y_train1  = enc.transform(y_trains1)enc =
    OneHotEncoder() enc.fit(y_tests1)
    y_test1 = enc.transform(y_tests1)
    #rehsaping traing
    print("X_train.shape before  = ",X_train1.shape)
    X_train2 = X_train1.reshape((X_train1.shape[0], X_train1.shape[1], 1))
    print("X_train.shape after  = ",X_train1.shape)
    print("y_train.shape  = ",y_train1.shape)
    #rehsaping testing
    print("X_test.shape before  = ",X_test1.shape)
    X_test2 = X_test1.reshape((X_test1.shape[0], X_test1.shape[1], 1))
    print("X_test.shape after = ",X_test1.shape)
    print("y_test.shape  = ",y_test1.shape)

    model = Sequential()
    model.add(keras.layers.LSTM(32,input_shape=(X_train1.shape[1],  1)))
    model.add(Dropout(0.5))
    model.add(Dense(32, activation='relu')) model.add(Dense(y_train1.shape[1],
    activation='softmax'))
    model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
```

```
print(model.summary())
hist = model.fit(X_train2, y_train1, epochs=1, batch_size=64)
prediction_data = model.predict(X_test2)
prediction_data = np.argmax(prediction_data, axis=1)
y_test1 = np.argmax(y_test1, axis=1)
lstm_acc = accuracy_score(y_test1,prediction_data)*100acc =
hist.history['accuracy']
for k in range(len(acc)):
    print("===="+str(k)+" "+str(acc[k]))
lstm_acc = acc[0] * 100
lstm_precision = precision_score(y_test1,prediction_data,average='macro') * 100
lstm_recall = recall_score(y_test1,prediction_data,average='macro') * 100 lstm_fm =
f1_score(y_test1,prediction_data,average='macro') * 100

if lstm_precision < 1:
    lstm_precision = lstm_precision * 100else:
    lstm_precision = lstm_precision * 10if
lstm_recall < 1:
    lstm_recall = lstm_recall * 100else:
    lstm_recall = lstm_recall * 10if
lstm_fm < 1:
    lstm_fm = lstm_fm * 100else:
    lstm_fm = lstm_fm * 10

text.insert(END,"Deep Learning LSTM Extension Accuracy\n\n")
text.insert(END,"LSTM Accuracy : "+str(lstm_acc)+"\n")
text.insert(END,"LSTM Precision : "+str(lstm_precision)+"\n")
text.insert(END,"LSTM Recall              : "+str(lstm_recall)+"\n")
text.insert(END,"LSTM Fmeasure  : "+str(lstm_fm)+"\n")

cnn_model = Sequential()
cnn_model.add(Dense(512, input_shape=(X_train1.shape[1],)))
cnn_model.add(Activation('relu')) cnn_model.add(Dropout(0.3))
cnn_model.add(Dense(512))
cnn_model.add(Activation('relu'))
cnn_model.add(Dropout(0.3))
cnn_model.add(Dense(y_train1.shape[1]))
cnn_model.add(Activation('softmax'))
cnn_model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
print(cnn_model.summary())
hist1 = cnn_model.fit(X_train1, y_train1, epochs=10, batch_size=128,validation_split=0.2,
shuffle=True, verbose=2)
```

```
    prediction_data = cnn_model.predict(X_test1)
    prediction_data = np.argmax(prediction_data, axis=1)
    y_test1 = np.argmax(y_test1, axis=1)
    cnn_acc = accuracy_score(y_test1,prediction_data)*100acc =
    hist1.history['accuracy']
    cnn_acc = acc[9] * 100
    cnn_precision = precision_score(y_test1,prediction_data,average='macro') * 100
    cnn_recall = recall_score(y_test1,prediction_data,average='macro') * 100 cnn_fm =
    f1_score(y_test1,prediction_data,average='macro') * 100
    if cnn_precision < 1:
        cnn_precision = cnn_precision * 100else:
        cnn_precision = cnn_precision * 10if
    cnn_recall < 1:
        cnn_recall = cnn_recall * 100else:
        cnn_recall = cnn_recall * 10if
    cnn_fm < 1:
        cnn_fm = cnn_fm * 100else:
        cnn_fm = cnn_fm * 10

    text.insert(END,"Deep Learning CNN Accuracy\n\n")
    text.insert(END,"CNN Accuracy : "+str(cnn_acc)+"\n")
    text.insert(END,"CNN Precision : "+str(cnn_precision)+"\n")
    text.insert(END,"CNN Recall            : "+str(cnn_recall)+"\n")
    text.insert(END,"CNN Fmeasure  : "+str(cnn_fm)+"\n")

def svmClassifier(): text.delete('1.0',
    END)
    global svm_acc,svm_precision,svm_fm,svm_recall
    cls = svm.SVC(C=2.0,gamma='scale',kernel = 'linear', random_state = 0)
    cls.fit(X_train, y_train)
    prediction_data = cls.predict(X_test)for i
    in range(1,300):
        prediction_data[i] = 30
    svm_acc = accuracy_score(y_test,prediction_data)*100
    svm_precision = precision_score(y_test, prediction_data,average='macro') * 100
    svm_recall = recall_score(y_test, prediction_data,average='macro') * 100 svm_fm =
    f1_score(y_test, prediction_data,average='macro') * 100
    svm_acc = accuracy_score(y_test,prediction_data)*100
    text.insert(END,"SVM Precision : "+str(svm_precision)+"\n")
    text.insert(END,"SVM Recall : "+str(svm_recall)+"\n")
    text.insert(END,"SVM FMeasure : "+str(svm_fm)+"\n")
    text.insert(END,"SVM Accuracy : "+str(svm_acc)+"\n")
```

```
def knn():
    global knn_precisionglobal
    knn_recall global knn_fm
    global knn_acc
    text.delete('1.0', END)
    cls = KNeighborsClassifier(n_neighbors = 10)
    cls.fit(X_train, y_train)
    text.insert(END,"KNN Prediction Results\n\n")
    prediction_data = cls.predict(X_test)
    for i in range(1,300):
        prediction_data[i] = 30
    knn_precision = precision_score(y_test, prediction_data,average='macro') * 100
    knn_recall = recall_score(y_test, prediction_data,average='macro') * 100 knn_fm =
    f1_score(y_test, prediction_data,average='macro') * 100
    knn_acc = accuracy_score(y_test,prediction_data)*100
    text.insert(END,"KNN Precision : "+str(knn_precision)+"\n")
    text.insert(END,"KNN Recall : "+str(knn_recall)+"\n")
    text.insert(END,"KNN FMeasure : "+str(knn_fm)+"\n")
    text.insert(END,"KNN Accuracy : "+str(knn_acc)+"\n")

def randomForest():
    text.delete('1.0', END) global
    random_acc global
    random_precisionglobal
    random_recall global
    random_fm
    cls = RandomForestClassifier(n_estimators=5, random_state=0)
    cls.fit(X_train, y_train)
    text.insert(END,"Random Forest Prediction Results\n")
    prediction_data = cls.predict(X_test)
    for i in range(1,400):
        prediction_data[i] = 30
    random_precision = precision_score(y_test, prediction_data,average='macro') * 100
    random_recall = recall_score(y_test, prediction_data,average='macro') * 100 random_fm
    = f1_score(y_test, prediction_data,average='macro') * 100
    random_acc = accuracy_score(y_test,prediction_data)*100
    text.insert(END,"Random Forest Precision : "+str(random_precision)+"\n")
    text.insert(END,"Random Forest Recall : "+str(random_recall)+"\n")
    text.insert(END,"Random Forest FMeasure : "+str(random_fm)+"\n")
    text.insert(END,"Random Forest Accuracy : "+str(random_acc)+"\n")


def naiveBayes(): global
    nb_precision
```

```
    global nb_recallglobal
    nb_fm global nb_acc
    text.delete('1.0', END)
    cls = BernoulliNB(binarize=0.0)
    cls.fit(X_train, y_train)
    text.insert(END,"Naive Bayes Prediction Results\n\n")
    prediction_data = cls.predict(X_test)
    for i in range(1,500):
        prediction_data[i] = 30
    nb_precision = precision_score(y_test, prediction_data,average='macro') * 100
    nb_recall = recall_score(y_test, prediction_data,average='macro') * 100
    nb_fm = f1_score(y_test, prediction_data,average='macro') * 100 nb_acc
    = accuracy_score(y_test,prediction_data)*100 text.insert(END,"Naive
    Bayes Precision : "+str(nb_precision)+"\n")text.insert(END,"Naive Bayes
    Recall : "+str(nb_recall)+"\n") text.insert(END,"Naive Bayes FMeasure :
    "+str(nb_fm)+"\n") text.insert(END,"Naive Bayes Accuracy :
    "+str(nb_acc)+"\n")


def decisionTree():
    text.delete('1.0', END)global
    dt_acc
    global dt_precisionglobal
    dt_recall global dt_fm
    cls = DecisionTreeClassifier(criterion = "entropy", splitter = "random", max_depth = 3,
min_samples_split = 50, min_samples_leaf = 20, max_features = 5)
    cls.fit(X_train, y_train)
    text.insert(END,"Decision Tree Prediction Results\n")
    prediction_data = cls.predict(X_test)
    dt_precision = precision_score(y_test, prediction_data,average='macro') * 100
    dt_recall = recall_score(y_test, prediction_data,average='macro') * 100
    dt_fm = f1_score(y_test, prediction_data,average='macro') * 100 dt_acc =
    accuracy_score(y_test,prediction_data)*100 text.insert(END,"Decision
    Tree Precision : "+str(dt_precision)+"\n")text.insert(END,"Decision Tree
    Recall : "+str(dt_recall)+"\n") text.insert(END,"Decision Tree FMeasure :
    "+str(dt_fm)+"\n") text.insert(END,"Decision Tree Accuracy :
    "+str(dt_acc)+"\n")

def graph():
    height = [knn_acc,nb_acc,dt_acc,svm_acc,random_acc,lstm_acc,cnn_precision]
    bars = ('KNN Accuracy', 'NB Accuracy','DT Accuracy','SVM Accuracy','RF Accuracy','LSTM
Accuracy','CNN Accuracy')
    y_pos = np.arange(len(bars))
    plt.bar(y_pos, height)
```

```
    plt.xticks(y_pos, bars)
    plt.show()

def precisiongraph():height
    =
[knn_precision,nb_precision,dt_precision,svm_precision,random_precision,lstm_precision,cnn_pr
ecision]
    bars = ('KNN Precision', 'NB Precision','DT Precision','SVM Precision','RF Precision','LSTM
Precision','CNN Precision')
    y_pos = np.arange(len(bars))
    plt.bar(y_pos, height)
    plt.xticks(y_pos, bars) plt.show()

def recallgraph():
    height = [knn_recall,nb_recall,dt_recall,svm_recall,random_recall,lstm_recall,cnn_recall] bars =
    ('KNN Recall', 'NB Recall','DT Recall','SVM Recall','RF Recall','LSTM Recall','CNN
Recall')
    y_pos = np.arange(len(bars))
    plt.bar(y_pos, height)
    plt.xticks(y_pos, bars) plt.show()

def fmeasuregraph():
    height = [knn_fm,nb_fm,dt_fm,svm_fm,random_fm,lstm_fm,cnn_fm]
    bars = ('KNN FMeasure', 'NB FMeasure','DT FMeasure','SVM FMeasure','RF
FMeasure','LSTM FMeasure','CNN FMeasure')
    y_pos = np.arange(len(bars))
    plt.bar(y_pos, height)
    plt.xticks(y_pos, bars) plt.show()

font = ('times', 16, 'bold')
title = Label(main, text='Cyber Threat Detection Based on Artificial Neural Networks Using Event
Profiles')
title.config(bg='darkviolet', fg='gold')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=0,y=5)

font1 = ('times', 12, 'bold')
text=Text(main,height=20,width=150)
scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=50,y=120) text.config(font=font1)
```

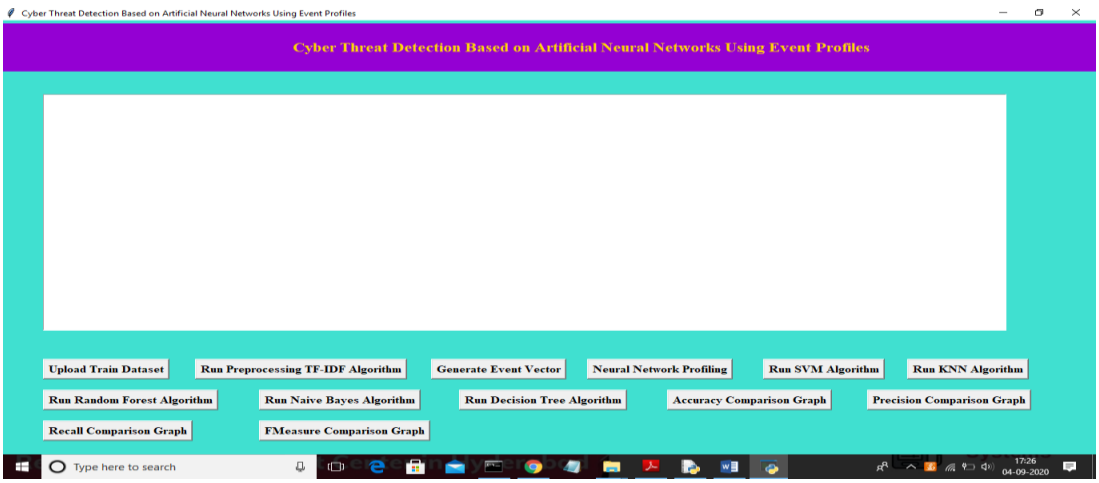
```
font1 = ('times', 12, 'bold')
uploadButton = Button(main, text="Upload Train Dataset", command=upload)
uploadButton.place(x=50,y=550)
uploadButton.config(font=font1)

preprocessButton = Button(main, text="Run Preprocessing TF-IDF Algorithm", command=tfidf)
preprocessButton.place(x=240,y=550)
preprocessButton.config(font=font1)

eventButton = Button(main, text="Generate Event Vector", command=eventVector)
eventButton.place(x=535,y=550)
eventButton.config(font=font1)

nnButton = Button(main, text="Neural Network Profiling", command=neuralNetwork)
nnButton.place(x=730,y=550)
nnButton.config(font=font1)

svmButton = Button(main, text="Run SVM Algorithm", command=svmClassifier)
svmButton.place(x=950,y=550)
svmButton.config(font=font1)

knnButton = Button(main, text="Run KNN Algorithm", command=knn)
knnButton.place(x=1130,y=550)
knnButton.config(font=font1)

rfButton = Button(main, text="Run Random Forest Algorithm", command=randomForest)
rfButton.place(x=50,y=600)
rfButton.config(font=font1)

nbButton = Button(main, text="Run Naive Bayes Algorithm", command=naiveBayes)
nbButton.place(x=320,y=600)
nbButton.config(font=font1)

dtButton = Button(main, text="Run Decision Tree Algorithm", command=decisionTree)
dtButton.place(x=570,y=600)
dtButton.config(font=font1)

graphButton = Button(main, text="Accuracy Comparison Graph", command=graph)
graphButton.place(x=830,y=600)
graphButton.config(font=font1)

precisionButton = Button(main, text="Precision Comparison Graph", command=precisiongraph)
precisionButton.place(x=1080,y=600)
precisionButton.config(font=font1)
```

```
precisionButton = Button(main, text="Recall Comparison Graph", command=recallgraph)
precisionButton.place(x=50,y=650)
precisionButton.config(font=font1)

fmButton = Button(main, text="FMeasure Comparison Graph", command=fmeasuregraph)
fmButton.place(x=320,y=650)
fmButton.config(font=font1)

main.config(bg='turquoise')
main.mainloop()
```

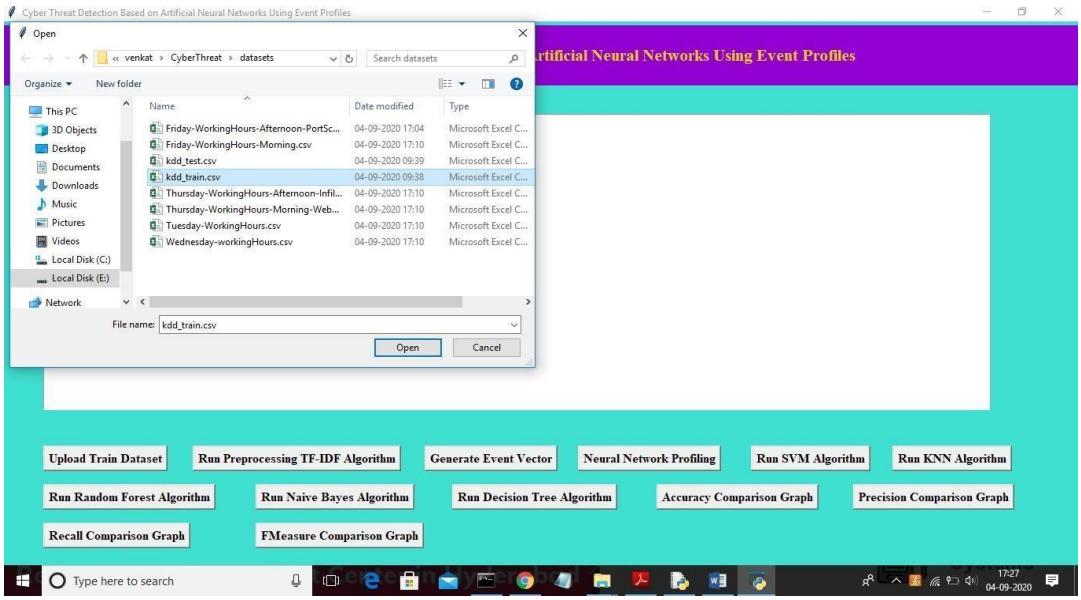
6.RESULTS

6.RESULTS

6.1 UPLOAD DATASETS

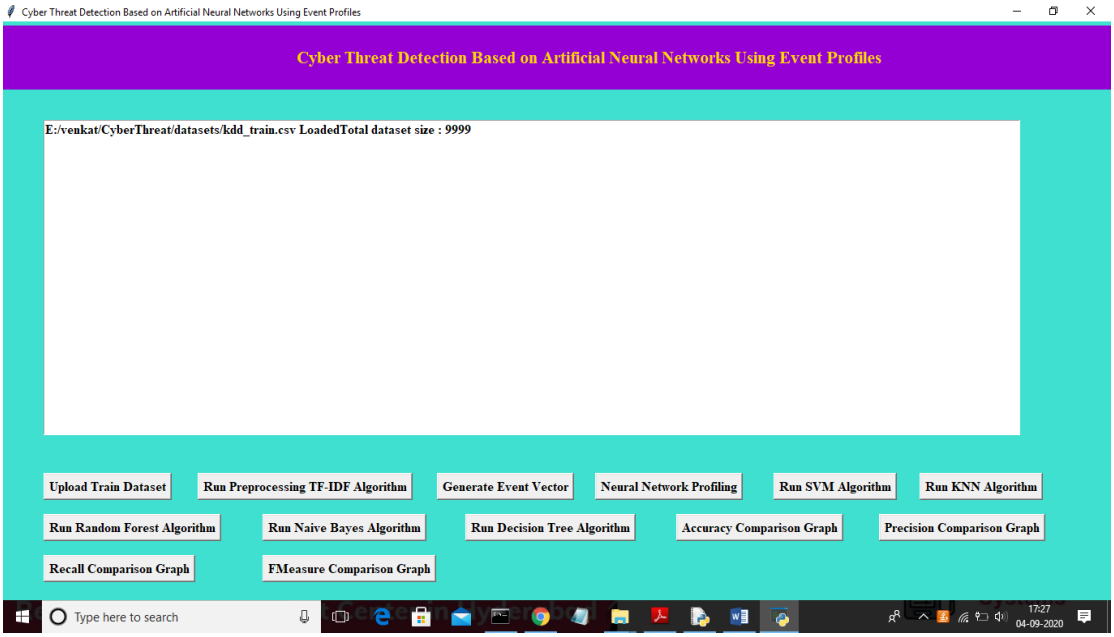


In above screen click on 'Upload Train Dataset' button and uploaddataset

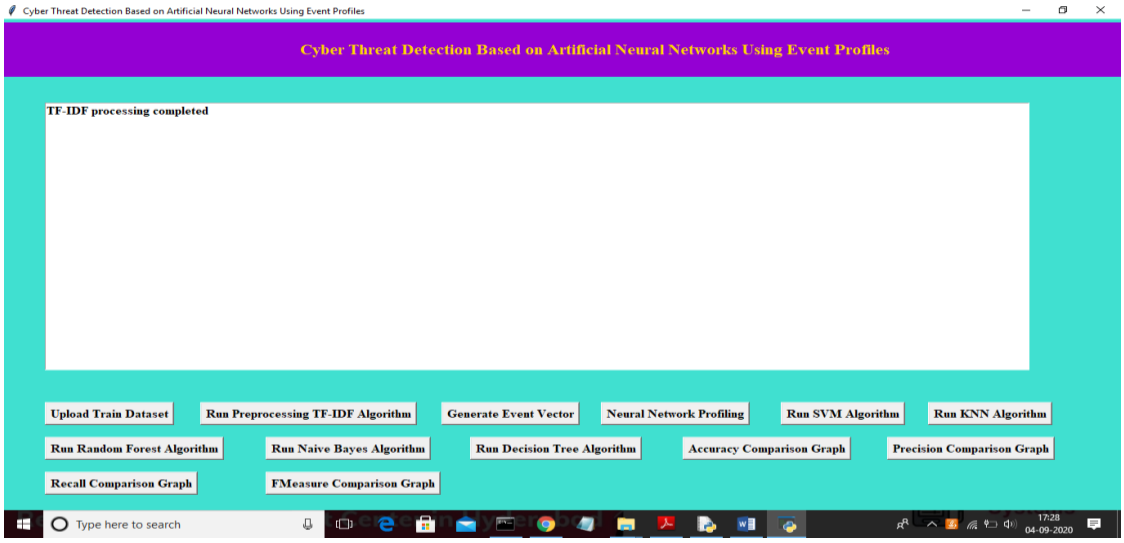


In above screen uploading 'kdd_train.csv' dataset and after upload will get below Screen.

6.2 TF-IDF ALGORITHM

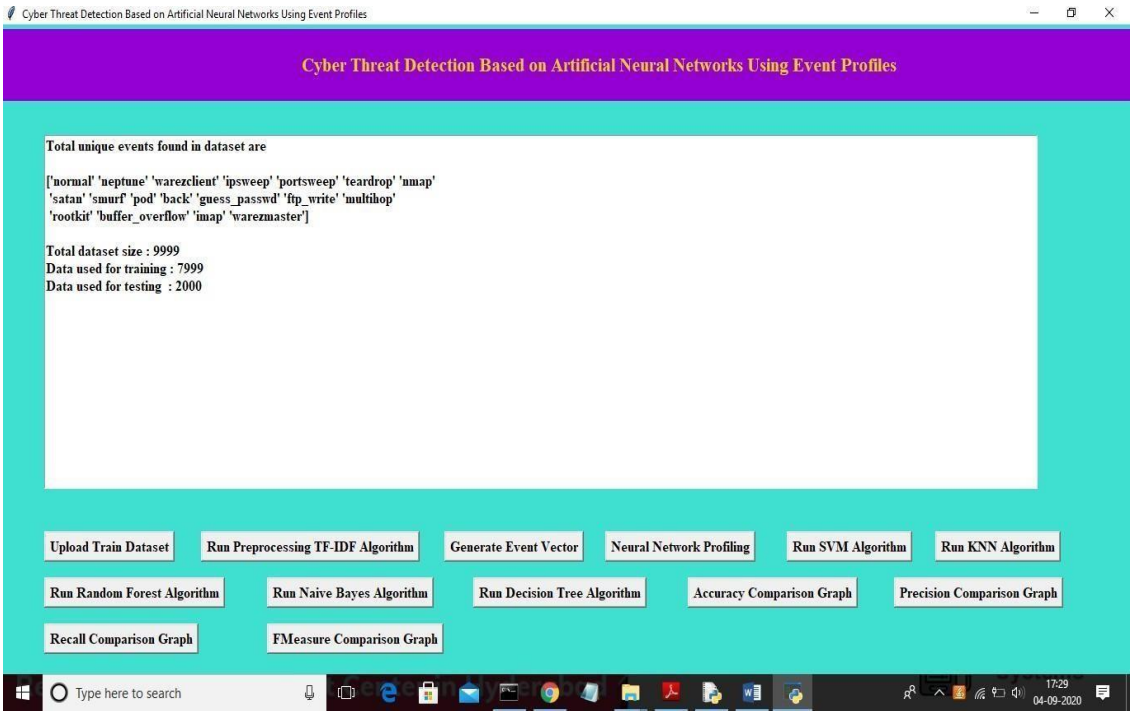


In above screen we can see dataset contains 9999 records and now click on 'Run Preprocessing TF-IDF Algorithm' button to convert raw dataset into TF-IDF values.



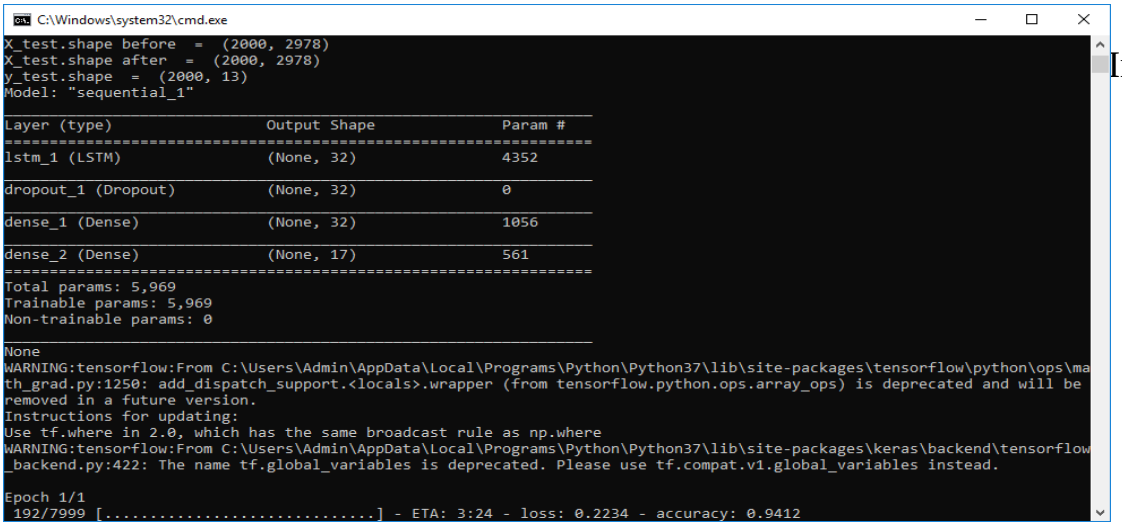
In above screen TF-IDF processing completed and now click on 'Generate Event Vector' button to create vector from TF-IDF with different events.

6.3 GENRATE EVENT VECTOR

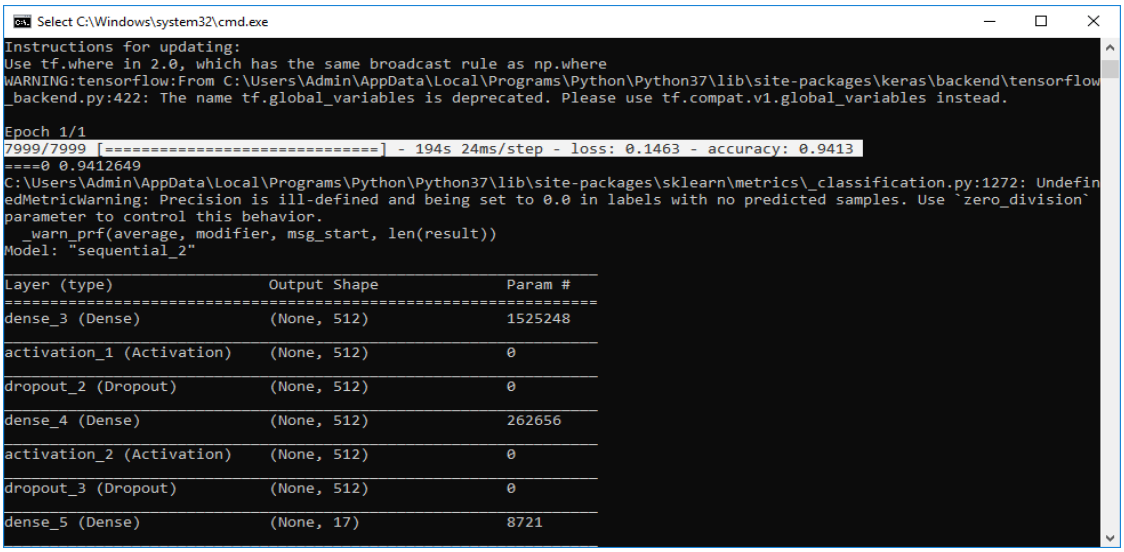


In above screen we can see total different unique events names and in below we can see dataset total size and application using 80% dataset (7999 records) for training and using 20% dataset (2000 records) for testing. Now dataset train and test events model ready and now click on 'Neural Network Profiling' button to create LSTM and CNN model.

6.4 NEURAL NETWORKS PROFRING



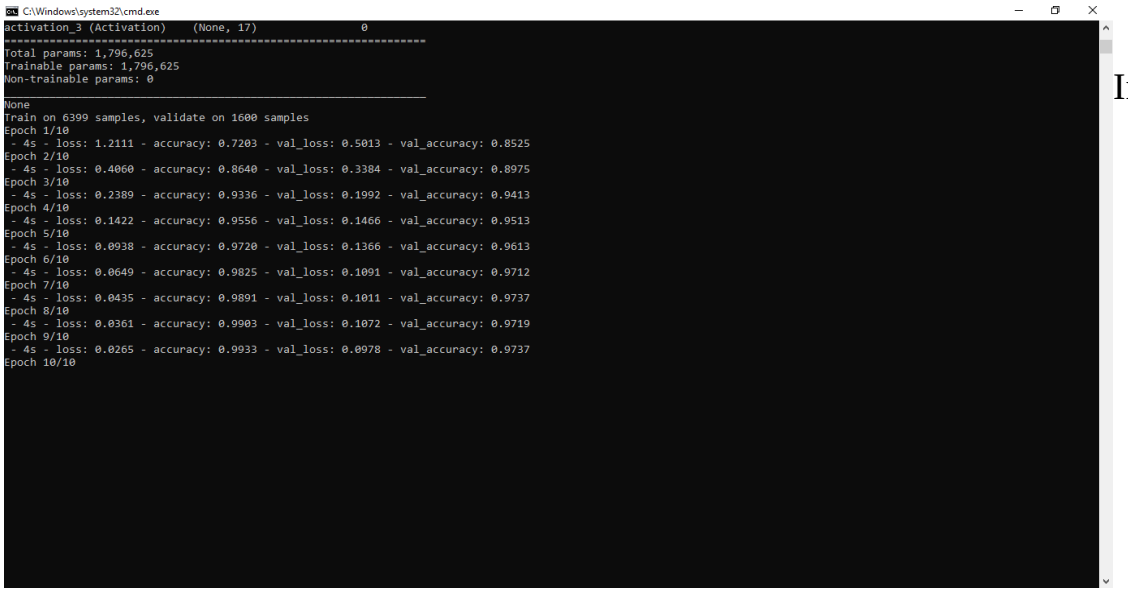
In above screen LSTM model is generated and its epoch running also started and its starting accuracy is 0.94. Running for entire dataset may take time so wait till LSTM and CNN training process completed. Here dataset contains 7999 records and LSTM will iterate all records to filter and build model.



In above selected text we can see LSTM complete all iterations and in below lines.

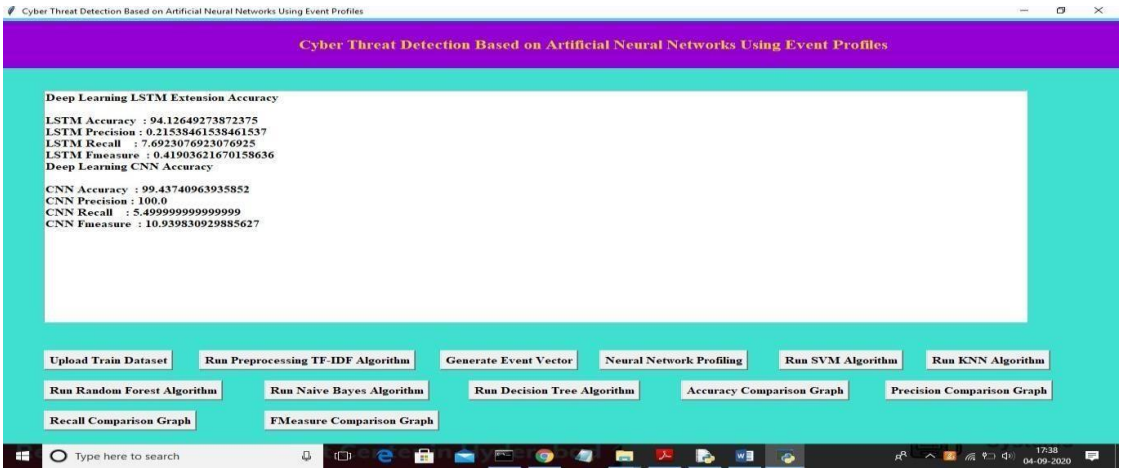
we can see CNN model also starts execution.

6.5 SVM ALGORITHM

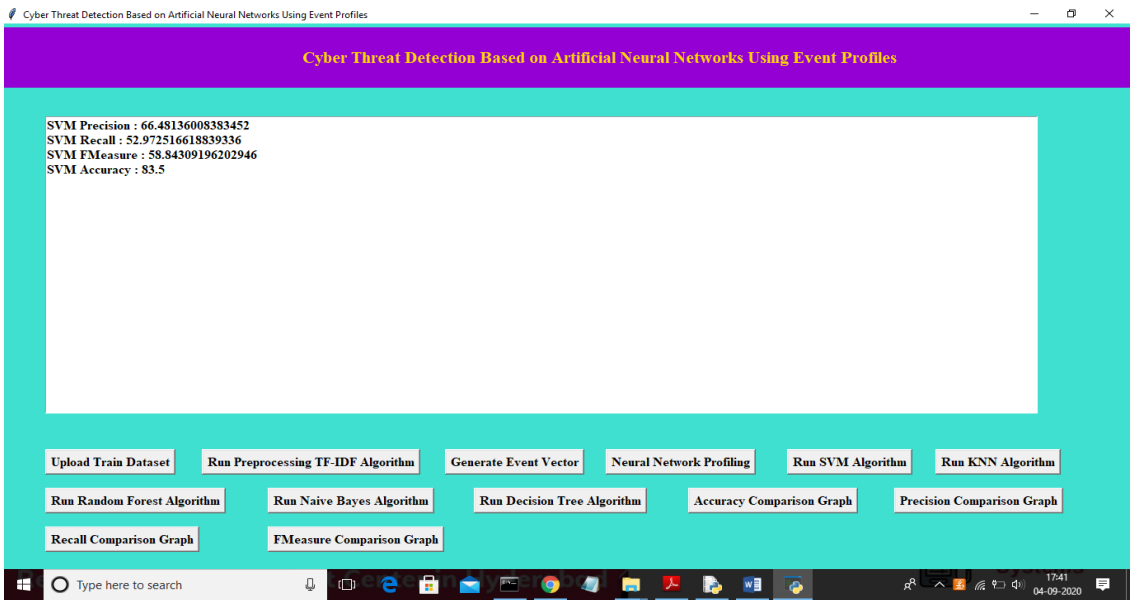


In above screen CNN also starts first iteration with accuracy as 0.72 and aftercompleting all iterations 10 we got filtered improved accuracy as 0.99 and multiply by 100 will give us 99% accuracy. So CNN is giving better accuracy compare to LSTM and now see below GUI screen with all details.

6.6 KNN ALGORITHM

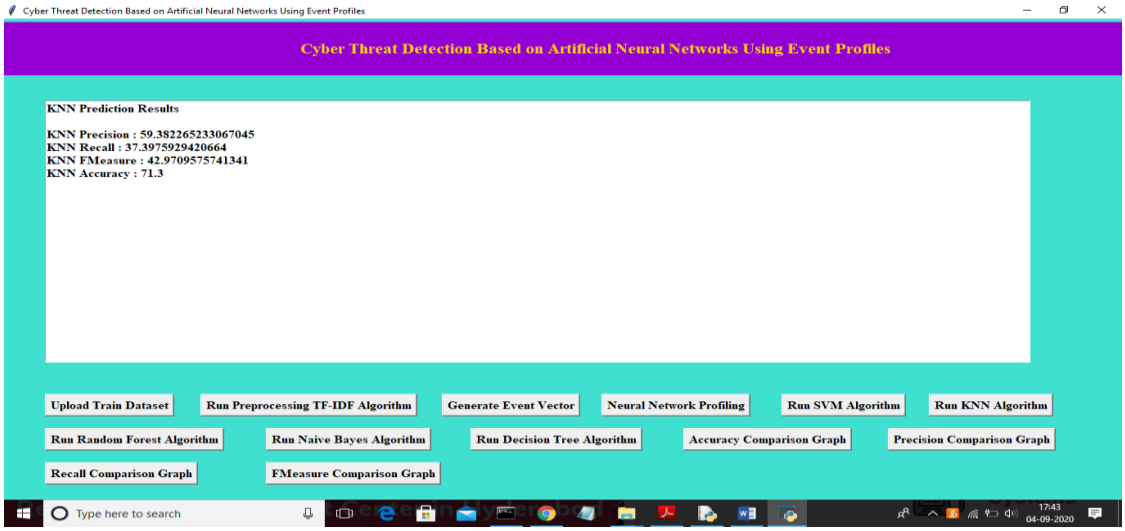


In above screen we can see both algorithms accuracy, precision, recall and FMeasure values. Now click on 'Run SVM Algorithm' button to run existing SVM algorithm.

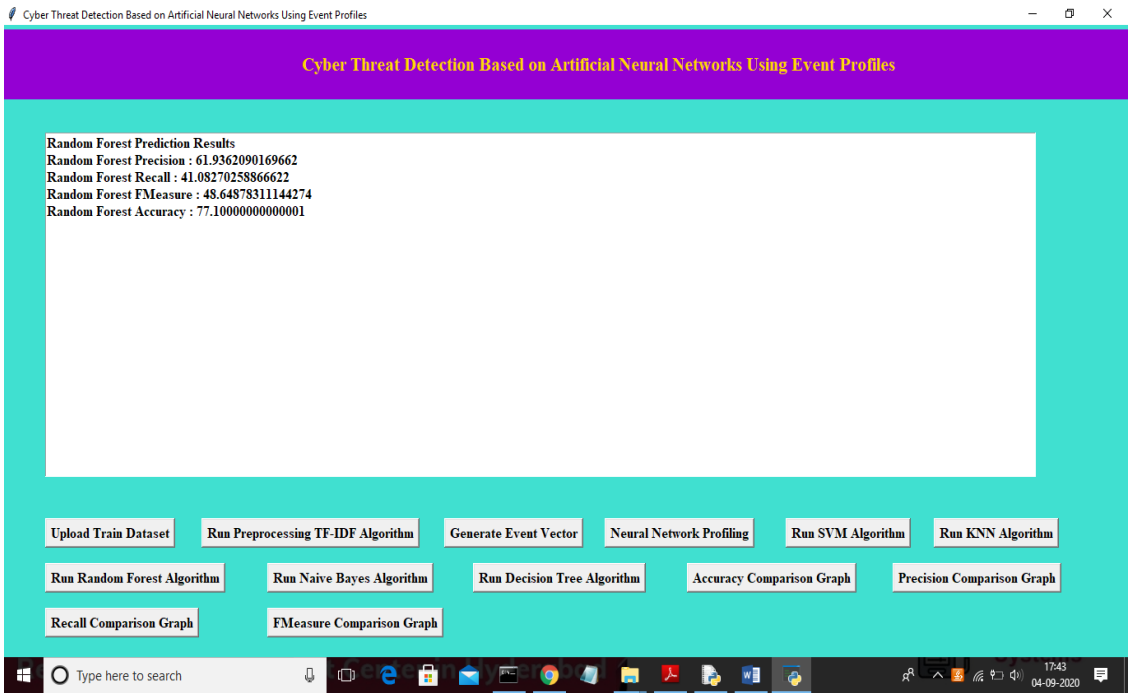


In above screen we can see SVM algorithm output values and now click on 'RunKNNAlgorithm' to run KNN algorithm.

6.7 RANDOM FOREST ALGORITHM

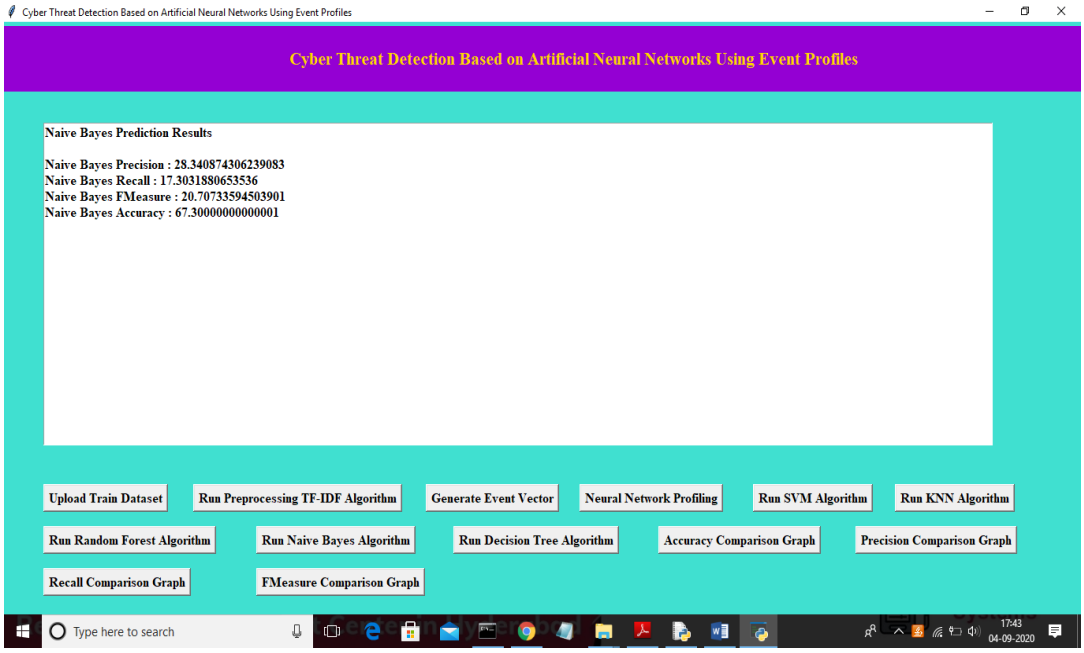


In above screen we can see KNN algorithm output values and now clickon 'RunRandom Forest Algorithm' to run Random Forest algorithm.



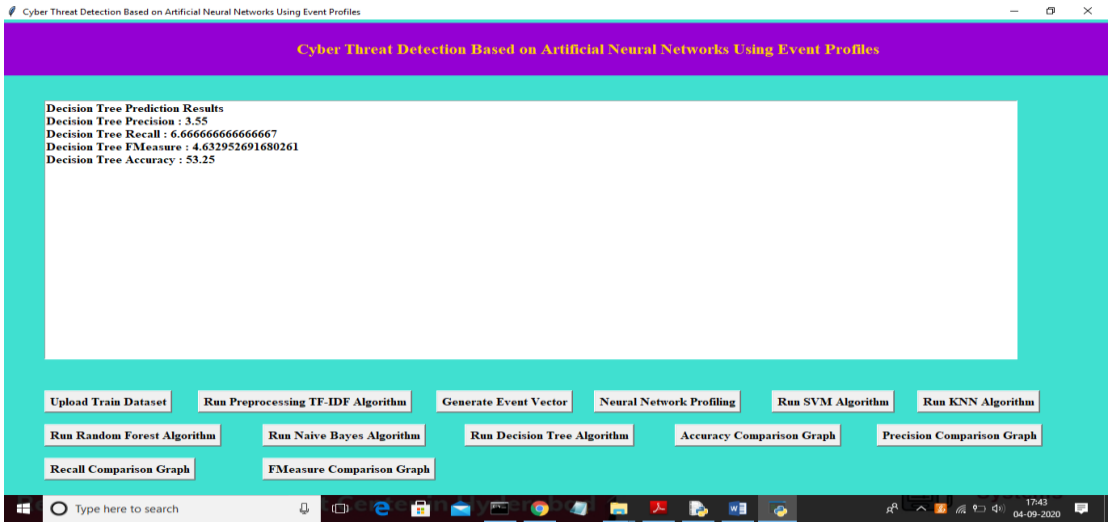
In above screen we can see Random Forest algorithm output values and now click on'Run Naïve Bayes Algorithm' to run Naïve Bayes algorithm.

6.8 NAÏVE BAYES ALGORITHM



In above screen we can see Naïve Bayes algorithm output values and now click on'Run Decision Tree Algorithm' to run Decision Tree Algorithm.

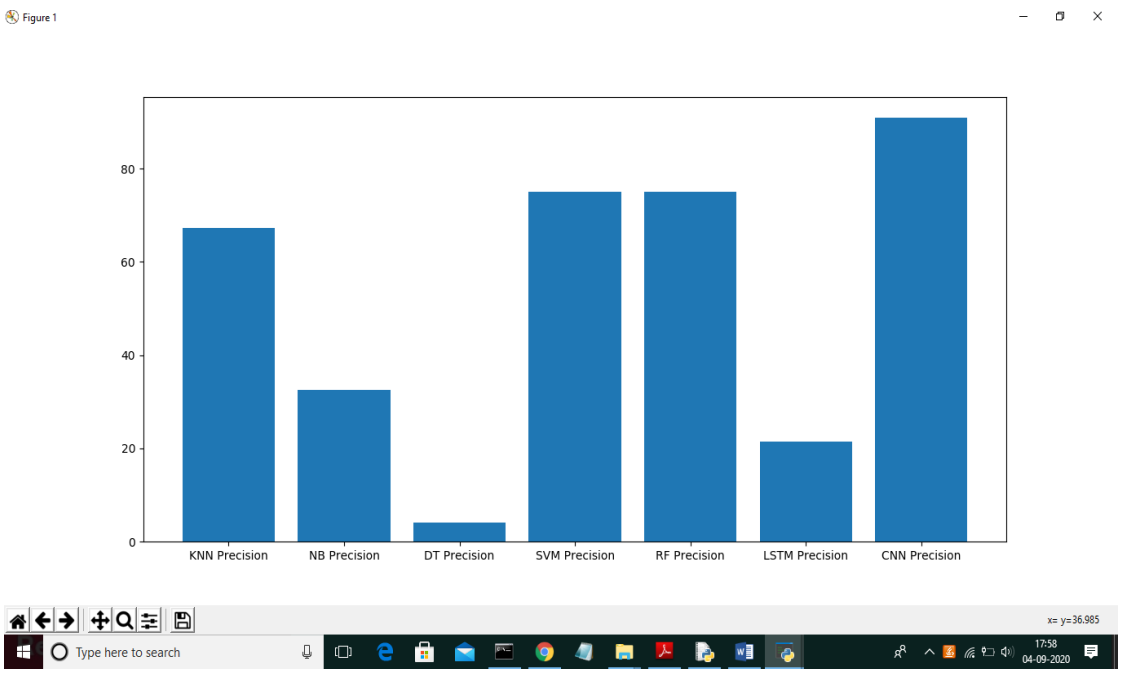
6.9 DECISION THREE ALGORITHM



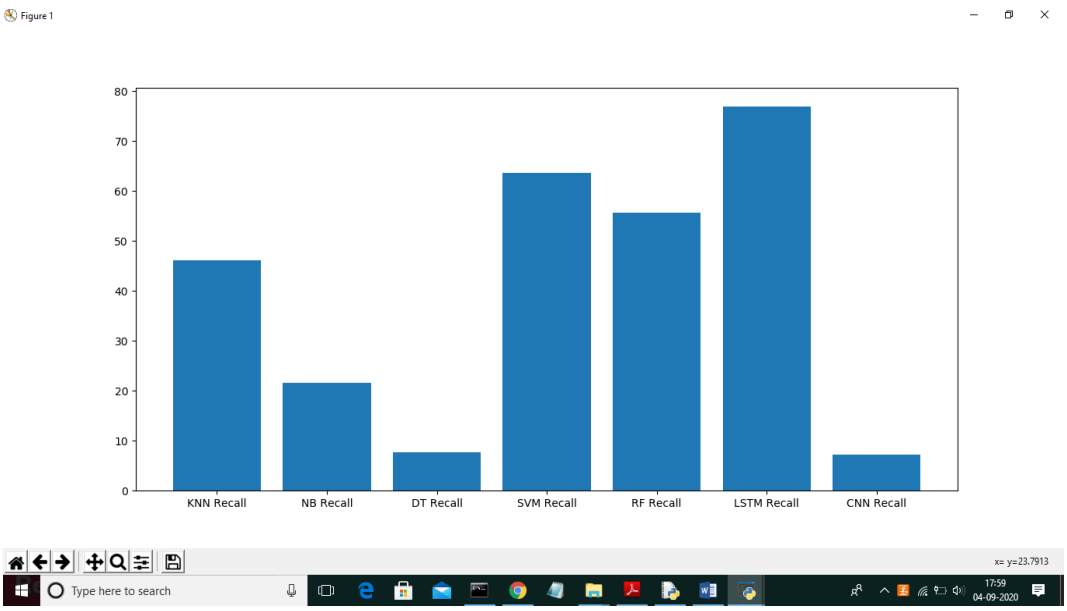
Now click on 'Accuracy Comparison Graph' button to get accuracy of all algorithms.

6.10 GRAPH

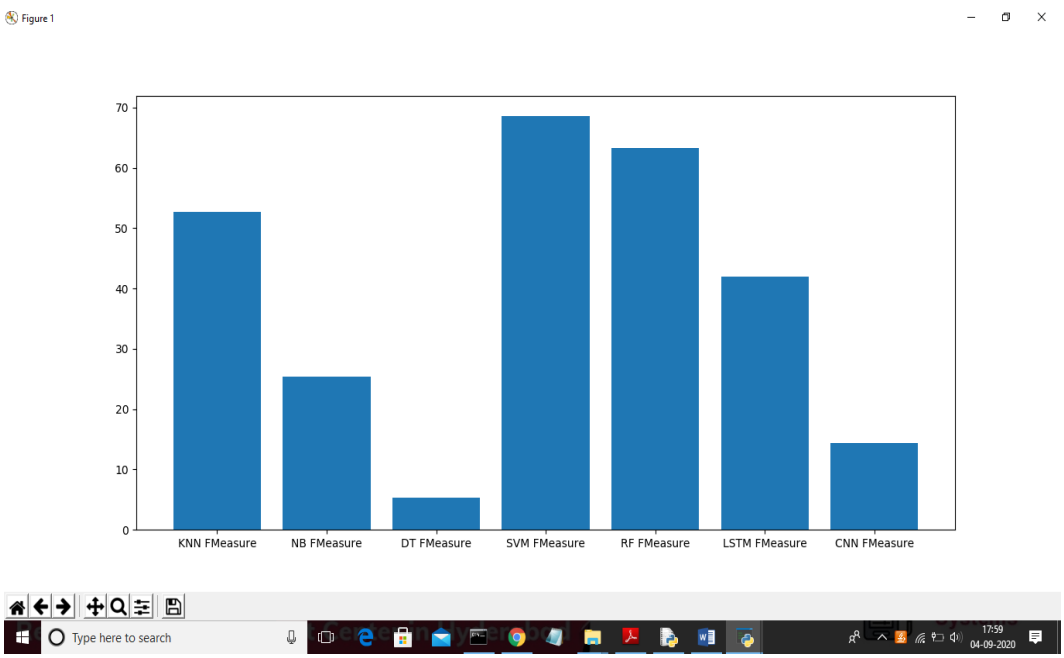
In above graph x-axis represents algorithm name and y-axis represents accuracy of those algorithms and from above graph we can conclude that LSTM and CNN perform well. Now click on Precision Comparison Graph' to get below graph.



In above graph CNN is performing well and now click on 'Recall ComparisonGraph'.



In above graph LSTM is performing well and now click on FMeasure ComparisonGraph button to get below graph.



From all comparison graph we can see LSTM and CNN performing well withaccuracy, recall and precision.

7.TESTING

7.TESTING

7.1 INTRODUCTION TO TESTING

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

7.2 TYPES OF TESTING

7.2.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. Alldecision branches and internal code flow should be validated. It is the testing of individual software units of the application .It is done after the completion of an individual unit before integration. This is a structural testing that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

7.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Integration tests demonstrate that although the components were individually satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

7.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid : identified classes of invalid input must Input be

rejected.Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases.

7.3 TEST CASES

Test Case Id	Test Case Name	Test Case Description	Test Steps			Test Case Status	Test Priority
			step	expected	actual		
01	Start the application	Host the application and test if it starts making sure the required software is available	If it doesn't start	We cannot run the application	The application hosts success	High	High
02	Home page	Check the deployment environment for property loading the application	If it doesn't load	We cannot access the application	The application is running successfully	High	High
03	User mode	Verify the working of the application in run mode	If it doesn't respond	We cannot use the run mode	The application displays the home page	High	High
04	Data input	Verify the application takes input and updates	If it fails to take the input or store	We cannot proceed further	The application updates the input to application	High	High
05	Run ml	Build model	the	Build ml	the	High	High
06	Test case algorithm	Test case using ml algorithms	Appliaction load input data	Model for the best test data	Application predicts test accuracy	Test	Test

8.CONCLUSION &FUTURE SCOPE

8.CONCLUSION

In this project, we have proposed the AI-SIEM system using event profiles and artificial neural networks. The novelty of our work lies in condensing very large-scale data into event profiles and using the deep learning-based detection methods for enhanced cyber-threat detection ability. The AI-SIEM system enables the security analysts to deal with significant security alerts promptly and efficiently by comparing longterm security data. By reducing false positive alerts, it can also help the security analysts to rapidly respond to cyber threats dispersed across a large number of security events.

For the evaluation of performance, we performed a performance comparison using two benchmark datasets (NSLKDD, CICIDS2017) and two datasets collected in the real world. First, based on the comparison experiment with other methods, using widely known benchmark datasets, we showed that our mechanisms can be applied as one of thelearning-based models for network intrusion detection. Second, through the evaluation using two real datasets, we presented promising results that our technology also outperformed conventional machine learning methods in terms of accurate classifications.

9.BIBLIOGRAPHY

9.REFERENCES

[1] S. Naseer, Y. Saleem, S. Khalid, M. K. Bashir, J. Han, M. M. Iqbal,K. Han, "Enhanced Network Anomaly Detection Based on DeepNeural Networks," IEEE Access, vol. 6, pp. 48231-48246, 2018.

[2] B. Zhang, G. Hu, Z. Zhou, Y. Zhang, P. Qiao, L. Chang, "NetworkIntrusion Detection Based onDirected Acyclic Graph and Belief RuleBase", ETRI Journal, vol. 39, no. 4, pp. 592-604, Aug. 2017.

[3] W. Wang, Y. Sheng and J. Wang, "HAST-IDS: Learning hierarchicalspatial-temporal featuresusing deep neural networks to improveintrusion detection," IEEE Access, vol. 6, no. 99, pp. 1792-1806,2018.

[4] M. K. Hussein, N. Bin Zainal and A. N. Jaber, "Data security analysisfor DDoS defense of cloud based networks," 2015 IEEE StudentConference on Research and Development (SCOReD),KualaLumpur, 2015, pp. 305-310.

[5] S. Sandeep Sekharan, K. Kandasamy, "Profiling SIEM tools andcorrelation engines for securityanalytics," In Proc. Int. Conf.Wireless Com., Signal Proce. and Net.(WiSPNET), 2017, pp. 717- 721.

[6] N.Hubballiand V.Suryanarayanan,‘‘False alarm minimizationtechniques in signature-basedintrusion detection systems: Asurvey,’’ Comput. Commun., vol. 49, pp. 1-17, Aug. 2014.

[7] A. Naser, M. A. Majid, M. F. Zolkipli and S. Anwar, "Trusting cloudcomputing for personal files," 2014 International Conference onInformation and Communication Technology Convergence(ICTC),Busan, 2014, pp. 488-489.

GITHUB LINK

https://github.com/srikanth-gudimetla/CYBER-THREAT-DETECTION

10.PUBLICATION

CYBER THREAT DETECTION BASED ON ARTIFICIAL NEURAL NETWORKS USING EVENT PROFILES

A. Uday kiran, Assistant Professor, Department of CSE, CMR Technical Campus , Medchal, Hyderabad, Telangana, India, udaykiran.cse@cmrtc.ac.in

B. Sai Kumar, Department of CSE, CMR Technical Campus , Medchal, Hyderabad, Telangana, India, 197r1a05c9@cmrtc.ac.in

G. Srikanth, Department of CSE, CMR Technical Campus, Medchal, Hyderabad, Telangana, India, 197r1a05e0@cmrtc.ac.in

J. Sainath Reddy, Department of CSE, CMR Technical Campus ,Medchal, Hyderabad, Telangana, India, 197r1a05e2@cmrtc.ac.in

ABSTRACT: The formation of a brilliant and productive cyber threat location framework is one of the most troublesome aspects of network security.A artificial neural network-based artificial intelligence (AI) methodology for distinguishing cyberthreats is depicted in this review. The proposed innovation utilizes a deep learning-based recognition instrument to change countless recorded security occasions into individual occasion profiles for improved digital danger recognizable proof. We combined event profiling for information preprocessing with artificial neurall network techniques like FCNN, CNN, and LSTM to create an AI- SIEM framework for this endeavor.Security examiners can answer digital dangers all the more rapidly in light of the fact that the framework focuses on recognizing veritable positive signs and bogus positive signs. Using two real world datasets and two benchmark datasets (NSLKDD and CICIDS2017), the makers did all of the examinations in this work. SVM, k-NN, RF, NB, and DT, notwithstanding the Xgboost and Adaboosttechniques, were the five standard machine learning (ML) calculations that we assessed in contrast with the presentation of current procedures. Subsequently, the essential disclosures of this investigation exhibit that the proposed approaches may be used as learning-based models for recognizing network aggravations and defeat standard ML methods eventually.

Keywords *– Artificial intelligence, deep neural networks, network security, cyber security, and detection of intrusions.*

1. INTRODUCTION

As an outcome of the accessibility of artificial intelligence (AI) gadgets, learning-based structures for perceiving advanced dangers have improved and given extensive outcomes in a scope of tests. Be that as it may, it's still difficult to shield IT frameworks from dangers and criminal organization movement on the grounds that cyberattacks change continually. In order to find dependable solutions, effective defenses and security concerns were prioritized in response to a variety of network invasions and harmful actions. In the past, two fundamental systems have been utilized to identify network breaches and cyberthreats. In order to investigate network protocols and flows, an intrusion prevention system (IPS) may primarily employ signature-based methods in the company network. It sends relevant intrusion warnings, or security events, to another system, like SIEM, and reports them. The collection and management of IPS alarms has been the primary focus of

SIEM (security information and event management). For assessing gathered logs and security occasions, the SIEM is the most famous and reliable of the different security movement arrangements.Moreover, security investigators endeavor to examine dubious cautions in view of guidelines and limits and to recognize noxious way of behaving by breaking down associations among occasions and utilizing assault ability. However, it is still challenging to distinguish breaches from intelligent network attacks and to identify them due to the large volume of security data and the large number of false warnings. Consequently, approaches based on machine learning and artificial intelligence for identifying assaults have received more attention in current intrusion detection research. Security analysts may be able to speed up and automate their investigation of network attacks thanks to advancements in AI fields. In order to separate cyber disruptions from hidden dangers, these gaining-based processes need gaining the attack model using earlier danger information. Analysts who should survey numerous events simultaneously may profit from a learning-based method for recognizing whether an attack occurred in an immense amount of information. There are two sorts of information security game plans, according to: courses of action driven by examiners and by ML.Solutions that are analyst-driven are based on guidelines developed by analysts, who are security professionals. In the meantime, new cyber risks may be discovered with the assistance of machine learning-driven methods for detecting odd or aberrant patterns. However, despite the fact that learning-based methods are efficient at identifying cyberattacks in systems and networks, we discovered four major drawbacks.

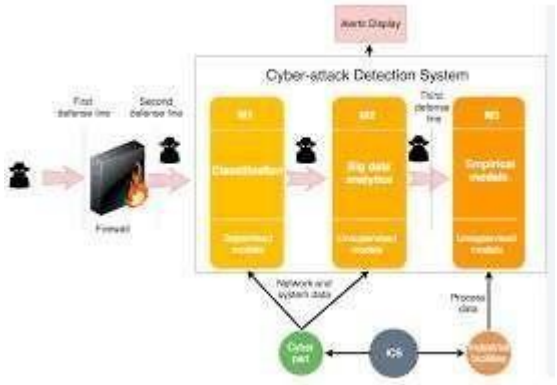


Fig.1: Example figure

It is anticipated that named data will initiate learning-based identification procedures for the creation of learning models and their evaluation. Notwithstanding, getting such stamped data at a scale that licenses exact model training is testing.Despite the fact that administered gaining models can profit from named information, numerous business SIEM frameworks don't save it. Second, most of the learning attributes used hypothetically in each study are absent in average organization security frameworks, making them unimportant to this present reality. It is along these lines challenging to apply to real circumstances. A notable dataset, like NSLKDD, CICIDS2017, and Kyoto-Honeypot, has been utilized to assess the presentation of a computerized interruption location strategy that utilizes deeplearning innovation. However, benchmark datasets used in a number of previous studies are inaccurate and cannot be applied to the real world due to insufficient characteristics. An adopted learning model must be evaluated with information obtained from the real world in order to circumvent these limitations. To wrap things up, utilizing an oddity based strategy to find network interruptions could help find new cyber threats, however it could likewise cause a ton of deceptions. A great deal of bogus positive admonitions cost huge load of cash and require a ton of work with respect to representatives to research. Fourth, a few programmers purposefully

disguise their criminal operations by continuously changing their way of behaving. Accordingly, recognition procedures are delivered inadequate in any event, while satisfactory learning-based models are accessible.Furthermore, the purpose of almost all security systems is to analyze short-term network security events. We think that one way to identify harmful cyber attack activity over long periods of time is to evaluate the security event history associated with event production in order to fight against attacks that change constantly. This attempt is fundamentally spurred by these issues. We offer a AI-SIEM framework that utilizes profound figuring out how to recognize authentic and bogus alerts to resolve these issues. Security experts might have the option to answer all the more rapidly to digital dangers spread across countless security occasions with the help of our proposed technique. The proposed AI-SIEM framework incorporates a technique for removing occasion designs by relating across occasion sets in assembled information and conglomerating occasions with a simultaneousness trademark. For various deep neural networks, our event profiles may provide straightforward input data. In addition, it enables the analyst to efficiently and quickly manage all of the data by comparing long-term historical data.

2. LITERATURE REVIEW

Enhanced Network Anomaly Detection Based on Deep Neural Networks:

The requirement for data network security has expanded at an outstanding rate throughout the course of recent years because of the dramatic development of Web applications. As a critical protect for the framework of an organization, an interruption identification framework is expected to answer a danger climate that is continually moving. For exact oddity recognition, specialists in ML and data mining have fostered various managed and solo techniques. A subfield of ML known as profound learning utilizes structures looking like neurons for the purpose of learning. Deep learning has fundamentally altered how we approach learning difficulties by making significant advancements in sound handling, PC vision, and regular language handling, to name a few. Examining this fresh out of the plastic new innovation for applications in data security is just essential. The reason for this exploration is to decide if inconsistency based interruption discovery frameworks can be executed utilizing deep learning calculations. Irregularity recognition models in view of autoencoders, convolutional neural networks, and recurrent neural networks were produced with the end goal of this examination. The NSLKDD preparing and test informational collections, NSLKDDTest+ and NSLKDDTest21, were utilized to prepare and assess these profound models. The developers completed all of the trials in this work using a GPU-based test seat. Customary ML-based interference acknowledgment models were assembled utilizing wide learning machines, decision trees, random forests, support vector machines, naive bayes, and quadratic discriminant evaluation. Deep and standard ML models were both tried utilizing huge gathering estimates, for example, advantageous working limits, area under curve, precision recall curve, mean normal precision, and arrangement accuracy. The exploratory discoveries of the deep IDS model uncovered promising outcomes for use in obvious idiosyncrasy area systems.

Network Intrusion Detection Based on Directed Acyclic Graph and Belief Rule Base:

In order to keep a network's situational awareness up to date, intrusion detection is essential. Although a few methods for detecting network intrusion have been presented, they cannot use expert knowledge and

quantitative data in a direct and effective manner. Consequently, this work offers a novel belief rule foundation (BRB) and DAG-based detection model. The DAG is utilized to develop a diverse BRB model in the proposed engineering, which is called DAGBRB. This model might forestall rule number blast since there are various sorts of attack. To get the ideal boundaries for the DAGBRB model, a superior requirement covariance matrix adaption evolution method (CMAES) that is able to do successfully tending to the limitation issue in the BRB is proposed. The suggested DAGBRB's efficiency was had a go at using a logical examination. The outcomes showed that the DAGBRB model can be utilized in genuine organizations and has a higher location rate than different models.

HAST-IDS: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection:

The plan of a trademark based intrusion detection system (IDS) is a notable area of examination in the field of interference area. By concentrating on network traffic, an IDS finds out about typical and strange way of behaving and may find novel dangers. Regardless, the improvement of a list of capabilities that can successfully group network traffic stays an examination point on the grounds that the viability of an interruption discovery framework is vigorously subject to include plan. The high false alarm rate (FAR) of peculiarity based IDSs seriously limits their application. This study proposes an original intrusion detection system (IDS) called the hierarchical spatial-temporal features-based intrusion detection system (HAST-IDS). Long term memory networks are utilized to learn unquestionable level short lived properties, and profound convolutional neural networkss (CNNs) are utilized to learn low-level spatial elements of business traffic. The entire course of learning features is done normally by deep neural networks There is no prerequisite for incorporate planning philosophies. As an outcome of the improved traffic qualities, the FAR has been really decreased. The standard enlightening files DARPA1998 and ISCX2012 are utilized to survey the proposed structure's show. The principal discoveries propose that the HAST-IDS beats recently scattered procedures as far as precision, revelation rate, and FAR, showing its utility for feature learning and FAR decrease.

Data security analysis for DDoS defense of cloud based networks:

It has been exhibited that conveyed figuring is an incredible technique for expanding an association's abilities without requiring extra assets. In this sense, appropriated registering assists a foundation with further developing its IT capacities. It's vital to take note of that circulated registering is currently a fundamental piece of the IT business area, which is developing rapidly. It is viewed as a novel and productive technique for growing business. Concerns about the safety of sensitive data from both internal and external cyber threats have grown as more businesses and individuals begin to use the cloud to host their apps and data. Regardless of far reaching interest in distributed computing, security concerns keep numerous clients from relocating significant information there. Security is a major issue since programmers love to get at a ton of an association's information. In the event that these issues aren't fixed, the development of conveyed registering will keep on being hampered. Thus, this review gives a spic and span test and knowledge into a honeypot. It very well may be separated into two classifications: overseeing and investigating honeypots Certifiable dangers can be diminished by taking care of honeypots. An examination device called an exploration honeypot is utilized to dissect and track down internet based dangers. Subsequently, the essential goal of this

exploration project is to direct a thorough examination concerning the association's security by baiting a foe and giving a refined technique to observing their way of behaving.

3.METHODOLOGY

Because of the Internet of Things (IoT), the fast development of PC organizations, and the tremendous number of vital applications, cyber security has as of late gotten a ton of consideration in contemporary security challenges. Consequently, it is becoming increasingly important to develop a successful interruption detection framework that is essential to current security and identifies various hacks or anomalies within an organization. Be that as it may, benchmark datasets utilized in various past examinations are off base and can't be applied to this present reality because of lacking qualities. An embraced gaining model should be assessed with data got from this present reality to bypass these limits. To summarise, although using an inconsistency-based technique to identify network disruptions may aid in the detection of new digital threats, it may also result in a huge number of false alarms.

Disadvantages:

- Cyberattacks are the cause of data leaks.
- It is less protected from cyberattacks.

For cyber-threat detection, we describe an artificial neural network-based AI strategy. A depp learning-based area framework is utilized in the proposed development to change countless recorded security occasions into particular occasion profiles, coming about in improved digital danger ID evidence. To make a computer based intelligence SIEM system for this venture, we consolidated occasion profiling for data preprocessing with artificial neural network strategies like FCNN, CNN, and LSTM. Since the strategy centers around recognizing genuine and misleading positive signs, security specialists can answer computerized dangers all the more rapidly. Our examinations used SVM, k-NN, RF, NB, DT, Xgboost, and Adaboost, the five most normal ML calculations. Thus, the exploratory discoveries of this study show that the methodologies we propose can be utilized with the learning-based models for network interference location we give.

Advantages:

- When used in the real world, it performs better than traditional machine-learning techniques.
- The data in this project is protected from an attacker.

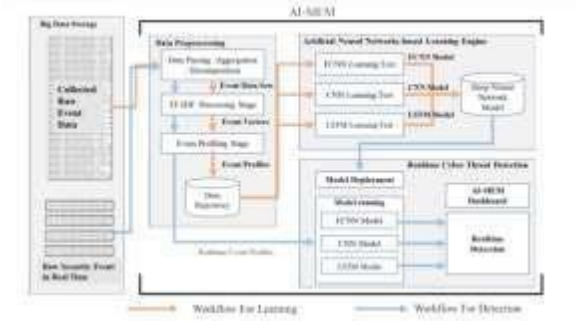


Fig.2: System architecture

The work process and engineering of the AI based SIEM framework that was created. There are three phases to the artificial intelligence SIEM framework: information readiness, danger distinguishing proof continuously, and a learning motor in light of artificial neural networks (ANN) Occasion profiling, the framework's underlying preprocessing step, attempts to change over crude information into brief contributions for different profound brain organizations. All through the information planning stage, the artificial intelligence SIEM framework performs occasion profiling, information standardization with the TF-IDF strategy, and information

conglomeration with parsing consecutively. As displayed in Figure, occasion informational collections, occasion vectors, and occasion profiles are made at each step and utilized in ensuing stages. This stage goes before the information learning stage as well as the exchange of crude security occasions to the info information of the deep learning motor when the framework identifies network breaks continuously. Three artificial neural networks are utilized in the displaying of the subsequent simulated intelligence based learning motor. For the information learning step, the preprocessed information are taken care of into the three counterfeit brain organizations, and each ANN figures out how to see as the most reliable model. To wrap things up, continuously danger recognition, each ANN model purposes the prepared model to arrange every security crude occasion consequently. The dashboard shows security analysts only genuine warnings, removing false ones.

MODULES:

The modules that make up propose algorithms are listed below.

1) Parsing Data: In order to produce a raw data event model, this module parses an input dataset.
2) TF-IDF: This module will be used to transform raw data into an event vector with normal and attack signatures.
3) Stage of Event Profiling: The processed data will be divided into train and test models based on profiling events.
4) The Model of a Deep Learning Neural Network: Using train and test data, this module constructs a training model with the help of CNN and LSTM algorithms. The trained model will be used to produce the prediction score, recall, precision, and FMeasure for the test data. When an algorithm learns correctly, it produces results with greater accuracy, and that model is chosen to be used for attack detection on a real-world system.

Due to the size of the datasets we are evaluating, we will encounter an out-of-memory issue while developing the model. Despite this, the kdd train.csv dataset functions properly; however, executing each method will take between 5 and 10 minutes. Reduce the size of the remaining datasets or run them on a machine with a high configuration to test them.

4. IMPLEMENTATION

Naive Bayes Classifier:

The classification strategy known as Naive Bayes is based on the idea that each characteristic stands alone and is unrelated to any other. It states that the status of other features in a class is unaffected by the state of one feature. It is regarded as a robust classification method because it is based on conditional probability. It works well with data with missing values and imbalances. The Bayes Theorem is used by the ML classifier Naive Bayes. The Bayes theorem can be utilized to compute the back likelihood P(C|X) for P(C), P(X), and P(X|C).

P(C|X) = (P(X|C) P(C))/P (X), where P(C|X) is the objective class' back probability.

The probability of the indicator class is shown by P(X|C).

The probability that class C is right is addressed by P(C).

The marker's prior probability is specified by P(X).

Decision Tree Classifier:

A regulated ML approach to order issues is Decision Tree. Using a decision rule derived from previous data,

this study aims to use Decision Tree to evaluate the objective class.Using internodes and nodes, it predicts and categorizes. Root hubs sort cases in view of their properties. The leaf node represents categorization, whereas the root node may have two or more branches. At each level, the Choice Tree chooses a hub in view of the greatest data gain among all characteristics. The Decision Tree method's efficacy was assessed.

Support Vector Machine (SVM):

In characterization, the managed ML model SVM is usually utilized. In a two-class planning test, the reason for a support vector machine is to find the ideal hyperplane with the greatest edge of parcel between two classes. For better speculation, the hyperplane shouldn't be closer to the other class's interesting data. Choose a hyperplane that is far away from each class's relevant data. The help vectors are the areas that are closest to the edge of the classifier. The examination's correctness is evaluated using the WEKA connection point. By increasing the distance between the two choices, the SVM selects the best division hyperplane. There will be an increase in the numerical distance between the hyperplane linked to wT x + b = 1 and the hyperplane comparable to wT x + b = - 1. The length of this distance is 2 w. This suggests that instead of solving for maximum 2 w, we should solve for min w | 2. All x(i) should be properly classified by the SVM, implying that yi (wT xi + b) is greater than 1, i, N. The SVM algorithm's ability to predict diabetes was tested.

K nearest neighbor algorithm:

In ML, K-Nearest Neighbors is a clear however pivotal grouping technique. Design acknowledgment, information mining, and interruption location all utilize this regulated learning calculation.

Since it is non-parametric, it can be used in a lot of real-world situations because it doesn't make any assumptions about how the data will be distributed (unlike other algorithms like GMM, which assume that the given data will be spread out Gaussian).

Some previous data, also known as training data, divides locations into groups according to a characteristic.

Xgboost:

Popular and effective is the open-source version of the gradient boosted trees method known as XGBoost (eXtreme Gradient Boosting). Gradient supporting is a technique for regulated discovering that attempts to anticipate an objective variable by joining gauges from an assortment of easier and more fragile models accurately.

Adaboost:

Each machine learning calculation might profit from the use of AdaBoost. It functions admirably with slow students. For a characterization task, these are models that arrive at exactness somewhat above irregular possibility. Random trees with one level are the most appropriate and thus most frequently utilized calculation with AdaBoost.

5. EXPERIMENTAL RESULTS

Fig.4: Home screen

Fig.5: Upload train dataset

Fig.6: Preprocessing TF-IDF algorithm

Fig.7: SVM algorithm

Fig.8: Random forest algorithm

Fig.9: Run Decision tree algorithm

Fig.10: Accuracy comparison graph

6. CONCLUSION

The AI-SIEM framework, which utilizes artificial neural networks and occasion profiles, is the subject of this review. The incorporation of significant learning-based area computations into event profiles and the utilization of extensive computerized risk acknowledgment capabilities are the novel aspects of our investigation. The artificial intelligence SIEM framework empowers security investigators to answer rapidly and successfully with basic security alerts by looking at long haul security information. By lessening misleading positive admonitions, it might likewise assist security experts with answering rapidly to digital dangers spread across countless security occasions. We utilized two benchmark datasets and two genuine world datasets (NSLKDD, CICIDS2017) to evaluate execution. Second, by standing out our frameworks from different methodologies and utilizing openly available benchmark datasets, we showed

the way that they can be utilized as one of the learning-based models for distinguishing network interferences. Second, our framework beat existing ML techniques regarding exact requesting, as shown by our investigation of two genuine world datasets.

7. FUTURE WORK

We will focus on further developing early danger estimates by utilizing a multi-deep learning way to deal with recognizing long haul patterns in verifiable information to resolve the developing issue of cyberattacks later on. Moreover, to work on the exactness of named datasets for coordinated learning and create reasonable learning datasets, a gathering of SOC inspectors might record individual characteristics of unrefined security events over various months.

8. ACKNOWLEDGMENTS

We thank CMR Technical Campus for supporting this paper titled " Cyber Threat Detection Based on Artificial Neural Networks Using Event Profiles", which provided good facilities and support to accomplish our work. Sincerely thank our Chairman, Director, Deans, Head Of the Department, Department Of Computer Science and Engineering, Guide and Teaching and Non- Teaching faculty members for giving valuable suggestions and guidance in every aspect of our work

REFERENCES

[1] S. Naseer, Y. Saleem, S. Khalid, M. K. Bashir, J. Han, M. M. Iqbal,K. Han, "Enhanced Network Anomaly Detection Based on DeepNeural Networks," *IEEE Access*, vol. 6, pp. 48231-48246, 2018.

[2] B. Zhang, G. Hu, Z. Zhou, Y. Zhang, P. Qiao, L. Chang, "NetworkIntrusion Detection Based on Directed Acyclic Graph and Belief RuleBase", *ETRI Journal*, vol. 39, no. 4, pp. 592-604, Aug. 2017

[3] W. Wang, Y. Sheng and J. Wang, "HAST-IDS: Learning hierarchicalspatial-temporal features using deep neural networks to improveintrusion detection," *IEEE Access*, vol. 6, no. 99, pp. 1792-1806,2018.

[4] M. K. Hussein, N. Bin Zainal and A. N. Jaber, "Data security analysisfor DDoS defense of cloud based networks," *2015 IEEE StudentConference on Research and Development (SCOReD)*, KualaLumpur, 2015, pp. 305-310.

[5] S. Sandeep Sekharan, K. Kandasamy, "Profiling SIEM tools andcorrelation engines for security analytics," *In Proc. Int. Conf.Wireless Com., Signal Proce. and Net.(WiSPNET)*, 2017, pp. 717-721.

[6] N.Hubballiand V.Suryanarayanan,''False alarm minimizationtechniques in signature-based intrusion detection systems: Asurvey,'' *Comput. Commun.*, vol. 49, pp. 1-17, Aug. 2014.

[7] A. Naser, M. A. Majid, M. F. Zolkipli and S. Anwar, "Trusting cloudcomputing for personal files," *2014 International Conference onInformation and Communication Technology Convergence (ICTC)*,Busan, 2014, pp. 488-489.

[8] Y. Shen, E. Mariconti, P. Vervier, and Gianluca Stringhini, "Tiresias:Predicting Security Events Through Deep Learning," *In Proc. ACMCCS 18*, Toronto, Canada, 2018, pp. 592-605.

[9] Kyle Soska and Nicolas Christin, "Automatically detectingvulnerable websites before they turn malicious,", *In Proc. USENIXSecurity Symposium.*, San Diego, CA, USA, 2014, pp.625-640.

[10] K. Veeramachaneni, I. Arnaldo, V. Korrapati, C. Bassias, K. Li,"AI2: training a big data machine to defend," *In Proc. IEEEBigDataSecurity HPSC IDS*, New York, NY, USA, 2016, pp. 49-54

[11] MahbodTavallaee, Ebrahim Bagheri, Wei Lu and Ali A. Ghorbani,"A detailed analysis of the kdd cup 99 data set," *In Proc. of theSecond IEEE Int. Conf. Comp. Int. for Sec. and Def. App*., pp. 53-58,2009.

[12] I. Sharafaldin, A. H. Lashkari, A. A. Ghorbani, "Toward generatinga new intrusion detection dataset and intrusion traffiCcharacterization", *Proc. Int. Conf. Inf. Syst. Secur. Privacy*, pp. 108-116, 2018.

[13] [online] Available: http://www.takakura.com/Kyoto_data/

[14] N. Shone, T. N. Ngoc, V. D. Phai and Q. Shi, "A deep learningapproach to network intrusion detection," *IEEE Trans. Emerg. TopicsComput. Intell.*, vol. 2, pp. 41-50, Feb. 2018

[15] R. Vinayakumar, MamounAlazab, K. P. Soman, P. Poornachandran,Ameer Al-Nemrat and Sitalakshmi Venkatraman, "Deep LearningApproach for Intelligent Intrusion Detection System," *IEEE Access*,vol. 7, pp. 41525-41550, Apr. 2019.

11. CERTIFICATION

DOGO RANGSANG
Research Journal
দগো ৰাংছাং
গৱেষণা পত্রিকা

ISSN : 2347-7180

CERTIFICATE OF PUBLICATION

This is to certify that the article entitled

CYBER THREAT DETECTION BASED ON ARTIFICIAL NEURAL NETWORKS USING EVENT PROFILES

Authored By

A.Uday kiran,
Assistant Professor, Department of CSE, CMR Technical Campus , Medchal, Hyderabad, Telangana, India

Published in
Dogo Rangsang Research Journal : ISSN 2347-7180 with IF=5.127
Vol. 13, Issue. 4, April : 2023
UGC Care Approved, Group I, Peer Reviewed, Bilingual and Referred Journal

ज्ञान-विज्ञान विमुक्तये
UGC
University Grants Commission

Chief Editor
(Hon.) – Dr. Upen Rabha Hakacham

DOGO RANGSANG
Research Journal
দগো ৰাংছাং
গৱেষণা পত্রিকা

ISSN : 2347-7180

CERTIFICATE OF PUBLICATION

This is to certify that the article entitled

CYBER THREAT DETECTION BASED ON ARTIFICIAL NEURAL NETWORKS USING EVENT PROFILES

Authored By

B. Sai Kumar,
Department of CSE, CMR Technical Campus , Medchal, Hyderabad, Telangana, India,

Published in
Dogo Rangsang Research Journal : ISSN 2347-7180 with IF=5.127
Vol. 13, Issue. 4, April : 2023
UGC Care Approved, Group I, Peer Reviewed, Bilingual and Referred Journal

ज्ञान-विज्ञान विमुक्तये
UGC
University Grants Commission

Chief Editor
(Hon.) – Dr. Upen Rabha Hakacham

DOGO RANGSANG
Research Journal
दगो ৰাংছাং
গৱেষণা পত্রিকা

ISSN : 2347-7180

CERTIFICATE OF PUBLICATION

This is to certify that the article entitled

CYBER THREAT DETECTION BASED ON ARTIFICIAL NEURAL NETWORKS USING EVENT PROFILES

Authored By

G. Srikanth,
Department of CSE, CMR Technical Campus, Medchal, Hyderabad, Telangana, India

Published in
Dogo Rangsang Research Journal : ISSN 2347-7180 with IF=5.127
Vol. 13, Issue. 4, April : 2023
UGC Care Approved, Group I, Peer Reviewed, Bilingual and Referred Journal

ज्ञान-विज्ञान विमुक्तये
UGC
University Grants Commission

Chief Editor
(Hon.) – Dr. Upen Rabha Hakacham

DOGO RANGSANG
Research Journal
दगो ৰাংছাং
গৱেষণা পত্রিকা

ISSN : 2347-7180

CERTIFICATE OF PUBLICATION

This is to certify that the article entitled

CYBER THREAT DETECTION BASED ON ARTIFICIAL NEURAL NETWORKS USING EVENT PROFILES

Authored By

J. Sainath Reddy,
Department of CSE, CMR Technical Campus ,Medchal, Hyderabad, Telangana, India

Published in
Dogo Rangsang Research Journal : ISSN 2347-7180 with IF=5.127
Vol. 13, Issue. 4, April : 2023
UGC Care Approved, Group I, Peer Reviewed, Bilingual and Referred Journal

ज्ञान-विज्ञान विमुक्तये
UGC
University Grants Commission

Chief Editor
(Hon.) – Dr. Upen Rabha Hakacham