Mathematical Statistician and Engineering Applications
ISSN: 2094-0343
2326-9865

Cyber Threat Detection based on Artificial Neural Networks using Event Profile

K.N.V.P.B. Ramesh, Asst Prof. CSE:BVCE, ramesh.kb17@gmail.com
Shubhashish Jena, OUTR:BBSR, sjena1998@gmail.com
P. Ramesh, Asst Prof. CSE:BVCE
Veernala Lakshmi Satya Sri, CSE:BVCE
Villa Rajesh, CSE:BVCE
Yandagandi V. K. Pavan, CSE:BVCE
Pati Vara Prasad, CSE:BVCE

Article Info

Page Number: 5300 - 5319

Publication Issue:

Vol 71 No. 4 (2022)

Article History

Article Received*: 25 March 2022*

Revised*: 30 April 2022*

Accepted*: 15 June 2022*

Publication*: 19 August 2022*

Abstract

Since cyber threats are getting worse, companies are looking for better ways to analyse security logs and make sure that cyber threats are found quickly and automatically. In this work, we want to use Deep Learning to make a cyber-threat detection framework that is both automated and effective (DL). DL is a promising way to find unknown network intrusions by using self-taught learning. It learns normal and dangerous patterns from the data it collects, taking into account how often they happen and reducing the number of false positive alerts in cyber security. It makes it easier for security analysts to respond quickly to a wide range of cyber threats. PSO was used to improve the accuracy of classification. It ranked all the attributes and chose the features. The SVM algorithm is used to classify the data in these chosen features. The proposed datamodel works well, as shown by the results of experiments on datasets of different sizes.
Search Terms: PSO, SVM, and Deep Learning (DL). CNN and FCNN

I. INTRODUCTION:

With the Internet becoming more and more a part of people's social lives, it changes how people learn and work, but it also makes us more vulnerable to security threats. Where to find that information and how to recognise different network attacks, most of which have not been seen before. It is an important issue to be solved immediately. There are two primary systems for detecting cyber-threats and network intrusions. It is still hard to know and detect intrusions against intelligent network attacks owing to their high false alerts and the huge amount of security data In this present work our objective is to achieve an automated and effective cyber- threats detection framework using Deep Learning (DL) (DL). It learns normal and threat patterns from collected data, considering the frequency of their occurrence. It helps security analysts respond quickly to cyber threats by cutting down on the number of false positive alerts. The goal is to improve the performance of detection by getting rid of wrong, noisy, or unimportant features. The class of an object can be predicted with the help of classification algorithms. In a dataset, the large number of data, some of which may be irrelevant or duplicate features, has a big effect on how well classification methods work. These can increase the complexity of classifiers. The experimental results with datasets of different sizes show that the proposed data model works well.

In this, I suggested that the PSO algorithm would work better. It ranks all of the attributes, chooses the features, and then uses SVM to classify the data. With the PSO algorithm, the goal is to improve the accuracy of classification.

In this, I use the NDLKKD CUP dataset. It was made as a variation of the well-known KDD-99 dataset, which is used to find outliers. It comes with 41 features and 5 classes and has a training set with 125,973 examples and a testing set with 22,543 samples. This set of data's features can be broken down into three types of data: nominal, binary, and numeric.

1.1 Problem Statement: Using the features of the given dataset, build a deep learning model that can find possible threats.

1.2 Goals: • To look at the data and find the different features that can help find cyber threats the most.
• To find the most accurate classifier by comparing different neural network models.

1.3 The work to be done:
With 10,000 records, this study will help improve how Machine Learning and Deep Learning are used to find cyber threats. The data was gathered from the internet and is a mix of two different sets of data. Here are the datasets that have been put together:
i. "NSL-KDD Dataset" ii. "CICIDS2017 Dataset"

1.4 How to Use
It is the process of looking at the whole security ecosystem to find any bad behaviour that could put the network at risk.
If a threat is found, mitigation steps must be taken to stop it before it can take advantage of any weaknesses that are already there.
It can find out what's wrong with data and tell if it's a real threat or a false alarm.
For this, they use a new encoding method that makes it easier to identify anonymous events when using the CNN structure. This CNN structure may or may not work better sometimes. To find out, they look at how other CNN

2.2 Contribution to Research:
The unique thing about our work is that we turn very large amounts of data into event profiles and use deep learning-based detection methods to make it easier to find cyber threats. By comparing long-term security data, the AI-SIEM system lets the security analysts deal with important security alerts quickly and effectively. It can also help security analysts respond quickly to cyber threats that are spread across a large number of security events by reducing the number of false positive alerts.

I. System Planned
To find threats using AI-SIEM (Artificial Intelligence-Security Information and Event Management), which is a combination of deep learning algorithms like FCNN, CNN, and LSTM (long short term memory). This technique works by profiling events, such as attack signatures. Author using standard algorithms like SVM, Decision Tree, Random Forest, KNN, and Naive

Bayes to evaluate how well the proposed work works. Here, we're putting the CNN and LSTM algorithms to work.

Paper	Application	Algorithm
Tu Chung-Jui, Li-Yeh Chuang	Feature Selection	PSO-SVM
W.SIEDLECKI, J.SKLANSKY	Finding optimal subsets	GENETIC
M. Tavallaee E. Bagheri, W. Lu, A.A. Ghorban	Analysis and solving issues of old dataset	KDDCUP-99
G.Wang, J. Hao, J. Ma, L.Huang	To solve imblanced attacks	ANN
Thiyagarajan Parama sivan	Security Mechanisms	ML AND DL
Jeng-Fung Chen ,Q Hung Do ,OrcID and Ho-Nien Hsieh	To reduce fast convergence rate	PDO-CS
MujahidH. Khalifa; Marwa Ammar; Wael Ouarda	Pattern recognition	PSO
Mohammed Harun Babu R, Vinayakumar R, Soman KP	Review on DL applications	DL
T. Kim, S. C. Suh, H. Kim	Identification of network anomalies	CNN
Jongboon Lee, Kim, Ikkyun Kim; Kijun Han	Cyber threat detection	SVM,CNN, LSTM

3.1 Algorithms:

3.1.1 SVM: The Support Vector Machine (SVM) is a machine learning algorithm that learns from being watched. This model is better at making predictions in the short and medium term than it is in the long term. Every algorithm has its own way of recognising patterns and making predictions based on them. The SVM model was used to look at the weekly trend of the NIKKEI 225 index to see how well it could predict financial trends. SVM is a boundary that uses a line or hyperplane to separate two classes in the best way. Equation shows how to find the decision boundary. SVMs use kernel functions like linear, non-linear, sigmoid, radial basis function (RBF), and polynomial to turn classes that can't be split into ones that can.

The slope and the intercepted constant c are the two things that can be changed in a sigmoid function.

RBF: $K(xi,xj)= \exp(-\gamma\|xi-xj\|2)$

Polynomial: $K(xi,xj) = xi\cdot xj+1)d$

Sigmoid:$K(xi,xj)= \tanh(\alpha xTiy+c)$

These work well in high-dimensional spaces and situations where the number of dimensions is larger than the number of samples. However, to avoid over-fitting when choosing regularisation terms and kernel functions, the number of features should be much larger than the number of samples.

3.1.2 KNN algorithm (K-Nearest Neighbor):

KNN is usually thought to have two properties: lazy learning and a non-parametric algorithm. This is because KNN makes no assumptions about how the underlying data is distributed. To find targets, the method has a few steps:

Step 1: start

Step 2: Separating the data into training data and test data

Step 3: Choose a value for K and figure out which distance function to use.

Step 4: Choose a test data sample.

Step 5: stop

The goal of the 3.1.3 Decision Tree algorithm is to make a model that can predict a target value by learning simple rules for making decisions from the data features. This method has some benefits, like being easy to understand and explain or being able to solve problems with more than one output. A decision tree is a common way to use supervised learning for both regression and classification problems. The goal of a technique is to predict a goal by making simple rules for making decisions based on a dataset and related features. Two benefits of using this model are that it is easy to understand and can solve problems with different outcomes. On the other hand, one disadvantage is that it can lead to overfitting by building trees that are too complicated.

3.1.4 Random forest algorithm: This model has three random parts: picking training data at random when making trees, picking some subsets of features when splitting nodes, and only looking at a subset of all features when splitting each node in each simple decision tree. In a random forest, each tree learns from a random sample of the data points when it is being trained. A random forest model is made up of a large number of decision trees. The model basically takes the average of what the trees predict, which is why it is called a forest. Also, the algorithm includes three random ideas: picking training data at random when making trees, picking some subsets of variables at random when splitting nodes, and deciding that only a subset of all variables should be used to split every node in each basic decision tree. During the training process of a random forest, each basic tree learns from a random sample of the dataset. Naive Bayes: Compared to more complex algorithms, the Naive Bayes classifier can be very fast. Because the class distributions aren't mixed up with

each other, each one can be evaluated on its own as a one-dimensional distribution. In turn, this helps get rid of some of the problems caused by the "dimensionality curse."

The Nave Bayes classifier is a type of probabilistic classifier that is based on Bayes' theorem and assumes that the features are very independent from each other based on the value of the class variable. This method is a group of algorithms for learning with help.

Equation by Bayes' theorem shows the following relationship: where y is a class variable and x1 through xn are dependent feature vectors.

$P(y|x1,\ldots,xn)=P(y)\prod i=1nP(xi|y)/P(x1,\ldots,xn)$

3.1.6 Algorithms for Deep Learning:

LSTM: LSTM is a type of RNN that can be used for a wide range of tasks, such as classifying documents, analysing time series, and recognising voice and speech. In contrast to feedforward networks, RNNs use estimates from the past to make predictions. In experimental work, RNNs aren't used very often because they have a few flaws that make their estimates useless.

LSTM solves the problems by using gates to let old information go and let new information in. A typical LSTM unit consists of a cell, an output gate, and a forget gate. The cell's main job is to recognise values at random points in time. The gates' job is to control how information flows into and out of the cell.

Structure of LSTM: LSTM is set up like a chain, with four neural networks and different memory blocks called cells at the end of each chain. A new part of LSTM is called a memory cell. The memory cell decides what information to store and when to let the information be read, written, or forgotten. There are three main gates in a memory cell:

o Input gate: a new value enters the memory cell through this gate.

o Forget gate: a value stays in the memory cell.

o Output gate: The output is based on the value in the memory cell.

CNN, or Convolution Neural Network:

It is one of the main categories that neural networks use to sort images into groups and recognise images. Convolutional neural networks are often used to label scenes, find objects, recognise faces, and do other things like that.

This algorithm takes an image as input, which is then put into a certain category, like "dog," "cat," "lion," "tiger," etc., and then processed. A computer sees an image as a set of pixels, and the quality of the image depends on how many pixels it has. Depending on the image resolution, it will look like h * w * d, where h is the height, w is the width, and d is the dimension. For example, the array of an RGB image is 6 * 6 * 3, and the array of a grayscale image is 4 * 4 * 1.

Each image that is fed into CNN will go through a series of convolution layers, pooling layers, fully connected layers, and filters (Also known as kernels). Then, we'll use the Soft-max function to put an object with probabilistic values of 0 and 1 into a certain category.

3.3 Architecture/Framework:

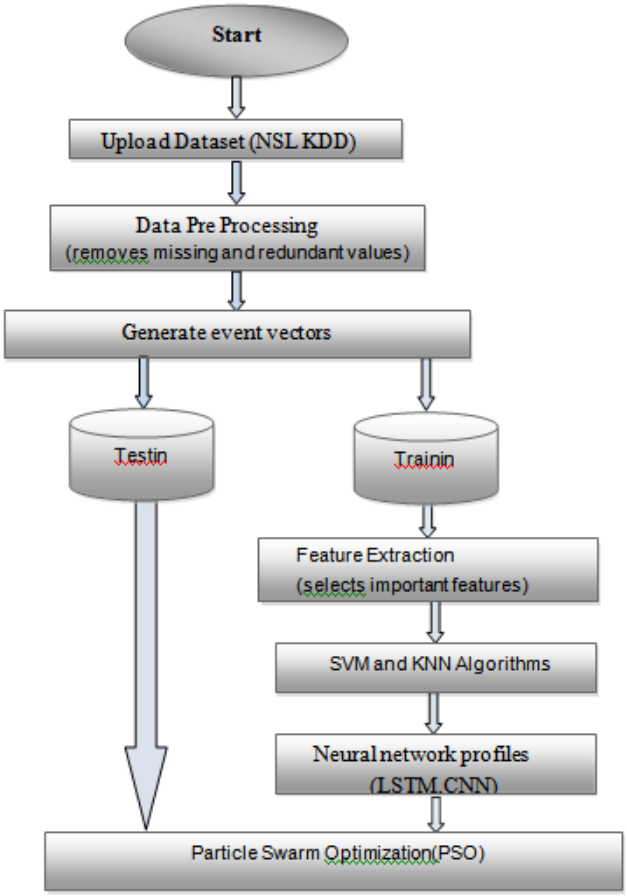

Fig.1. Architecture

The goal of PSO (Particle Swarm Optimization) is to find values for the variables that either minimise or maximise the functions while still meeting the constraints. It is a type of artificial intelligence that can be used to solve problems with numbers that are hard to solve. It uses the idea of social interaction to help solve problems.

Data Pre-processing: It gets rid of all missing values, duplicate values, and data that is already there.

Make TF-IDF (Text Frequency Inverse Document Frequency) and event vectors: It makes a list of all the unique events in the dataset and uses tf-idf to calculate each word and find the most important ones.

Training and Testing: 80% of the data is used for training, and 20% is used for testing. In this case, training is used to make a model, and testing is used to check the model's accuracy.

The number of features in a dataset is cut down by feature extraction, which creates new features. There are two ways to pull out features: one is supervised and the other is not. The main goal of this is to capture the same information with fewer features. It uses a method called "object-based."

3.4 Algorithm and Process Design:

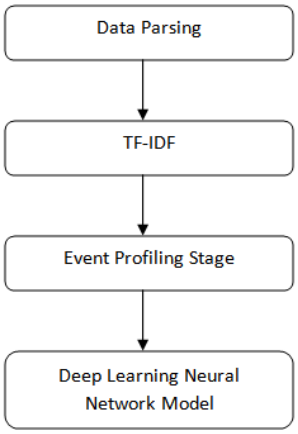

Fig.2.Process Design

1) Data Parsing: This module takes an input dataset and parses it to make a raw data event model.
2) TF-IDF: With this module, we'll turn raw data into an event vector that has signatures for both normal and attack situations.
3) Event Profiling Stage: Based on profiling events, the processed data will be split into train and test models.
4) Deep Learning Neural Network Model: This module runs CNN and LSTM algorithms on train and test data to make a training model. The prediction score, Recall, Precision, and FMeasure will be calculated by applying the trained model that was made to test data. The algorithm will learn perfectly, which will lead to more accurate results, and that model will be chosen to use on the real system to find attacks.

IV. How it was done and what happened

4.1 Concerning the data:
In this, I use NSL KDD, which is a benchmark dataset from Kaggle. It was made as a well-known version of the KDDCUP-99 dataset, which is used to find outliers. It comes with 41 features and 5 classes and has a training set with 1,25,973 examples and a testing set with 22,543 samples. There are three types of data that can be used to describe this dataset's features: nominal, binary, and numeric.
There are four kinds of feature types in this dataset.
• 6Binary(Features:7,12,14,20,21,22)\s• 4Categorical(Features:2,3,4,42)
• 23Discrete(Features:8,9,15,23-41,43)\s• 10Continuous(Features:1,5,6,10,11,13,16,17,18,19)
This dataset has the following data files: KDD Train+ARFF and KDD Test+ARFF, KDD Train+TXT and KDD Test+TXT, and KDD Train+-20%.
ARFF
• KDDTest-21.ARFF
4.2 Metrics for Evaluating:

Score, Accuracy, and Receiver Characteristics of Operation-Area We use the Area Under the Curve (ROC-AUC) metric to measure how well our models work. FPR = False Positive Rate; TPR = True Positive Rate; F1-score = Accuracy, Precision, and Recall; FPR = False Positive Rate; TPR = True Positive Rate; Accuracy, Precision, and Recall; and F1-score.
For this, values are calculated based on: • True positive (TP) = number of events for which the correct answer was given.
• False negative (FN): The number of events that were wrongly predicted and did not happen.
• False-positive (FP) = the number of wrongly predicted events.
• True negative (TN): The number of events that could have happened but didn't.
False Positive Rate (FPR): This is a way to measure how accurate machine learning is. What it means is: FPR=FP/(FP+TN)
True Positive Rate (TPR): This is the same thing as recall, so its definition is TPR = FP/(FP+TN).
Accuracy is the most important way to measure performance, and it's easy to do with a ratio of the number of correct predictions to the total number of observations.
Accuracy=(TN+TP)/(TP+FP+TN+FN)
Remember that it is the number of positive observations that can be predicted correctly out of all the observations in the original data.
Recall = (TP+FN)/(TP+TP)
Precision: It is used to figure out which values are correct. This means to take the total number of software's that were predicted as positive and figure out how many of them were right. Precision = TP / (TP + FP) says what it means.
F-Measure is the average of Precision and Recall based on how much weight each has. So, both false positives and false negatives are taken into account in this score. It is not as easy to understand as accuracy, but F Measure is usually more useful than accuracy, especially if you have a rough idea of how the class is distributed. The best way for accuracy to work is if both false positives and false negatives cost the same. If the prices for Precision and Recall are different, it's best to look at both.
F1 Score = 2(Precision Recall/Precision + Recall)

For this, values are calculated based on: • True positive (TP) = number of events for which the correct answer was given.
• False negative (FN): The number of events that were wrongly predicted and did not happen.
• False-positive (FP) = the number of wrongly predicted events.
• True negative (TN): The number of events that could have happened but didn't.

4.3 Outcome: To find threats using AI-SIEM (Artificial Intelligence-Security Information and Event Management), which is a combination of deep learning algorithms like FCNN, CNN, and LSTM (long short term memory). This technique works by profiling events, such as attack signatures. Author using standard algorithms like SVM, Decision Tree, Random Forest, KNN, and Naive Bayes to evaluate how well the proposed work works. Here, I'm putting the CNN and LSTM algorithms to work.

Fig.**3.** 'Upload Train Dataset'

In above figure click on 'Upload Train Dataset' button and upload dataset

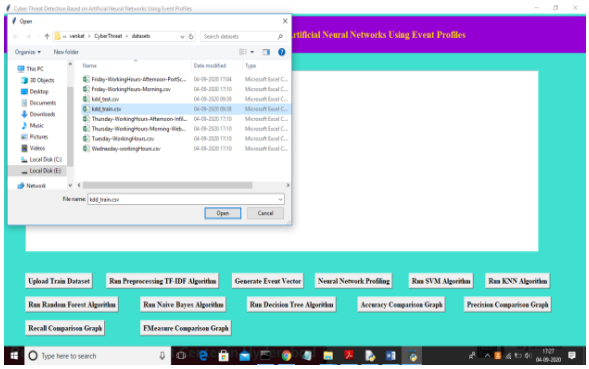

Fig.4. uploading 'kdd_train.csv'

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

Fig.5. Total record in dataset

In above figure 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

Fig.6. TF-IDF processing

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

Fig.7. total different unique events

In above figure 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

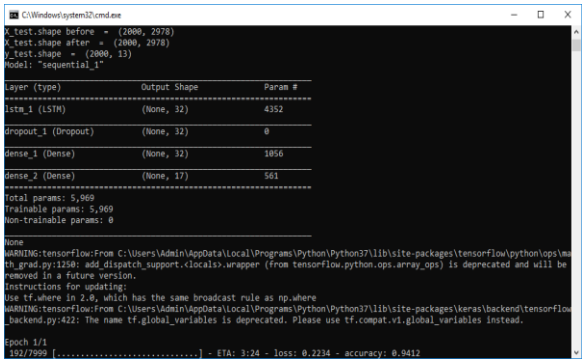

Fig.8. LSTM model

In above figure 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.

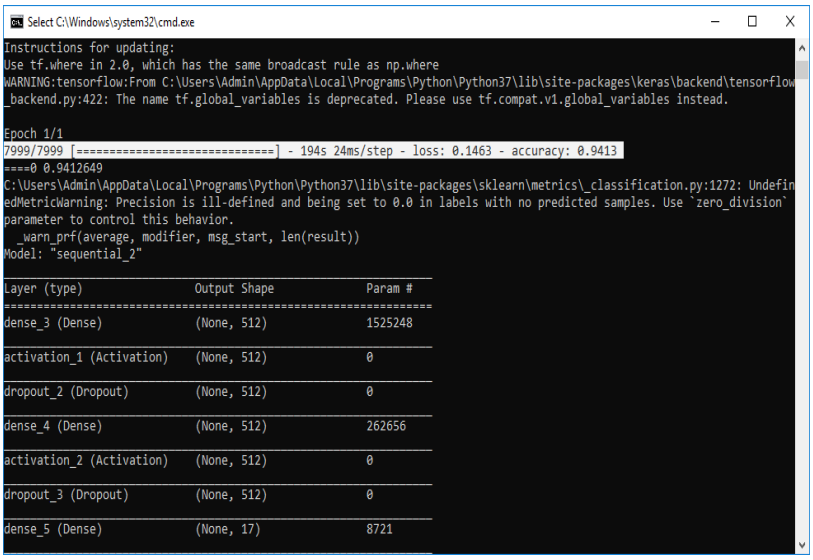

Fig.9. LSTM complete all iterations

In above selected text we can see LSTM complete all iterations and in below lines we can see CNN model also starts execution

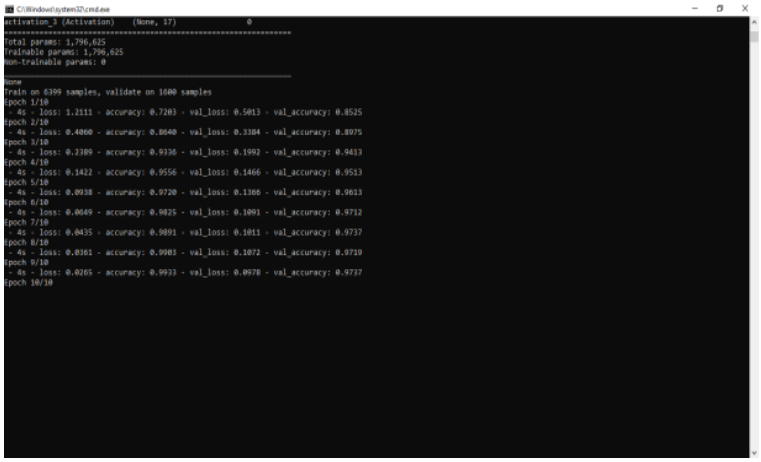

Fig.10.CNN starts first iteration with accuracy as 0.72

In above figure CNN also starts first iteration with accuracy as 0.72 and after completing 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 figure with all details

Fig.11. accuracy, precision, recall

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

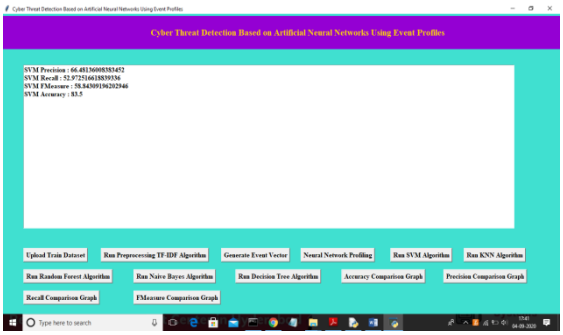

Fig.12. SVM algorithm output

In above figure we can see SVM algorithm output values and now click on 'Run KNNAlgorithm' to run KNN algorithm

Fig.13. KNN algorithm output

In above figure we can see KNN algorithm output values and now click on 'Run Random Forest Algorithm' to run Random Forest algorithm

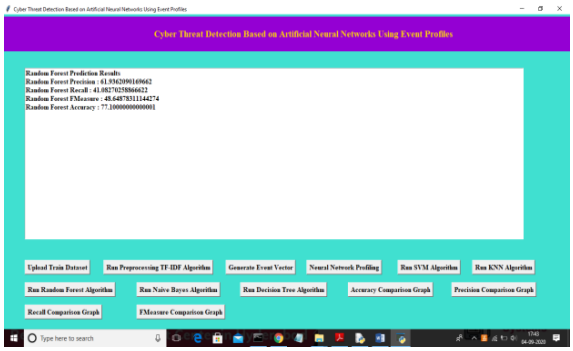

Fig.14. Random Forest algorithm output

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

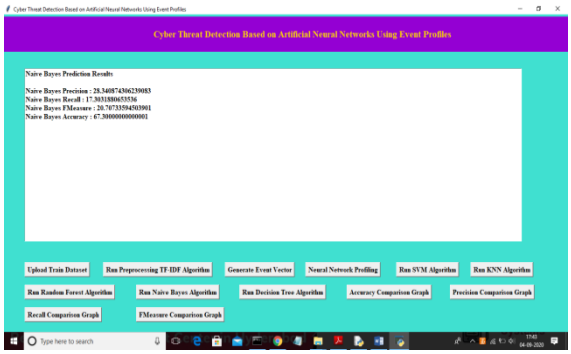

Fig.15. Naïve Bayes algorithm output

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

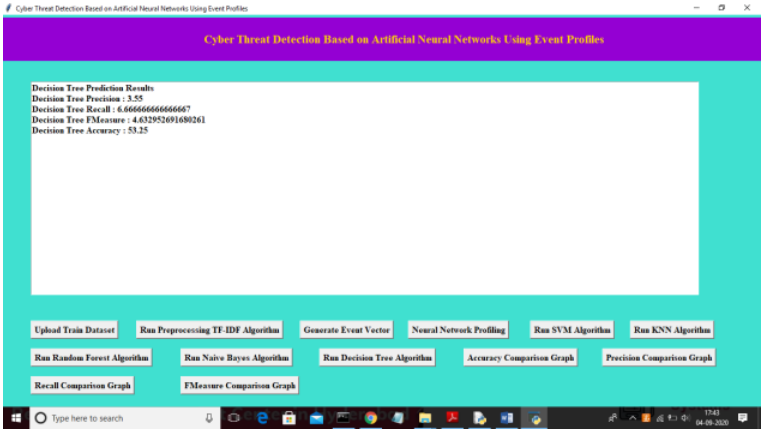

Fig.16.Accuracy Comparison Graph

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

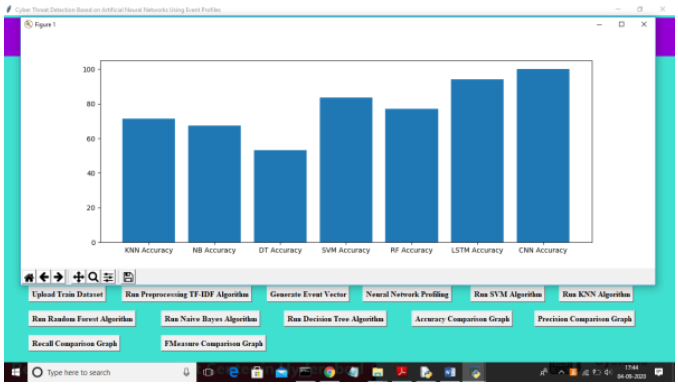

Fig.17.algorithmvc accuracy

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

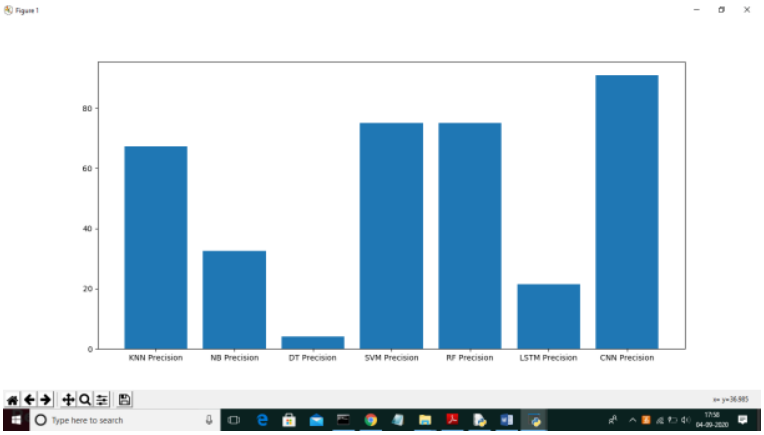

Fig.18. CNN is performing well

In above graph CNN is performing well and now click on 'Recall Comparison Graph'

Fig.19. LSTM is performing well

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

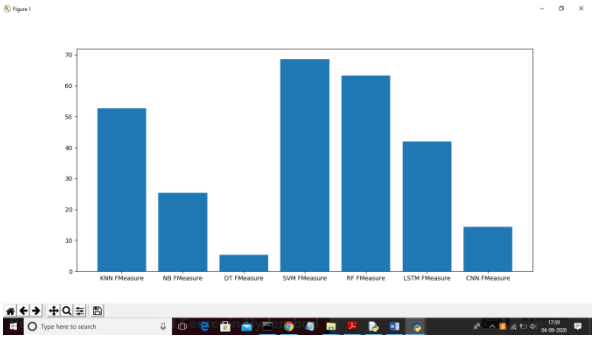

Fig.20. LSTM and CNN performing well

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

Extension Outcomes:

In this project as extension I added PSO algorithm which selected relevant features from dataset and then employing SVM on PSO selected features and the calculating and comparing its accuracy with rest of the algorithms. Below figure shots code with comments in red colour explaining about PSO implementation with SVM

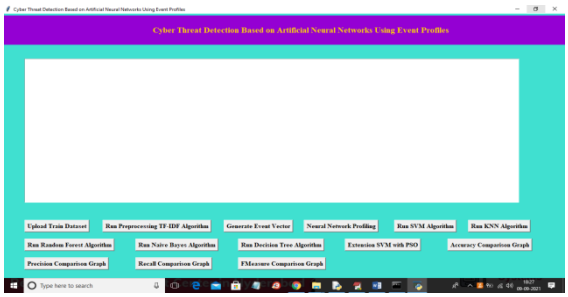

Fig.21. Extension SVM with PSO'

In above figure run all buttons as previous execution and I just added extra module called 'Extension SVM with PSO'. So click each button one by one to get output

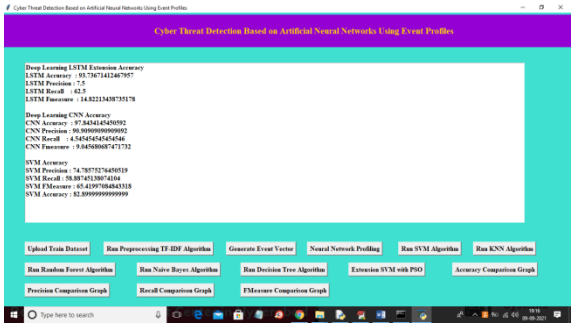

Fig.22. accuracy and other metrics

In above figure we can see accuracy and other metrics for LSTM, CNN and SVM and in below figure we can see other algorithms output

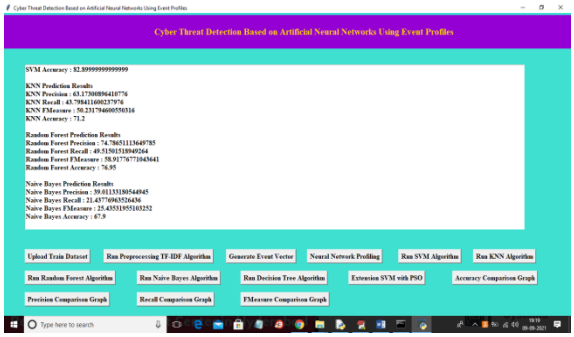

Fig.23. Random Forest, Naïve Bayes and Decision Tree

In above figure we can see result for Random Forest, Naïve Bayes and Decision Tree and now click on ‘Extension SVM with PSO’ button to get below output

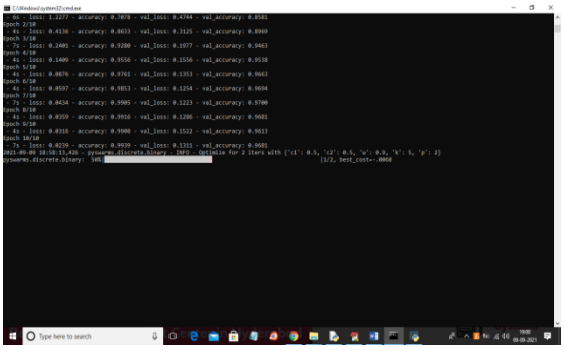

Fig.24.PSO starts selecting important features

In above figure we can see PSO starts selecting important features from dataset and after that will get below output

Fig.25.result for decision tree

In above figure we can see result for decision tree and then we can see SVM with PSO accuracy as 99%. Now below are the performance graph

Accuracy graph

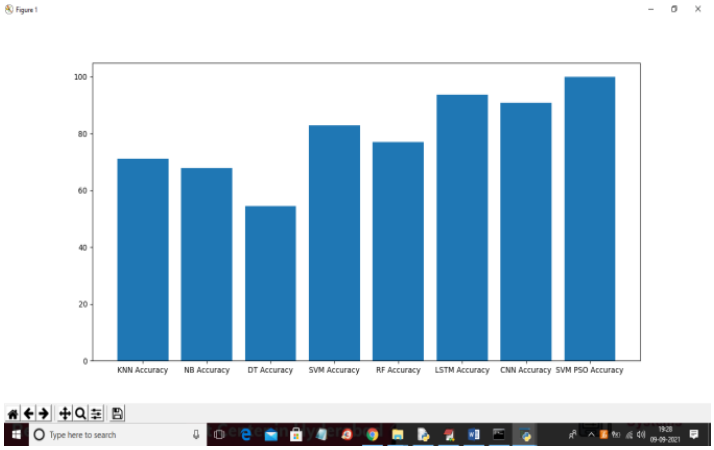

Fig.27. Accuracy graph

Precision graph

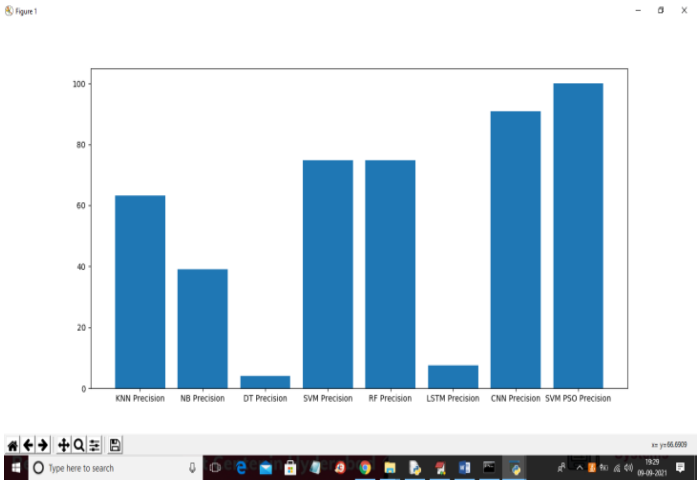

Fig.28. Precision graph

Recall graph

Fig.29.Recall graph

FSCORE Graph

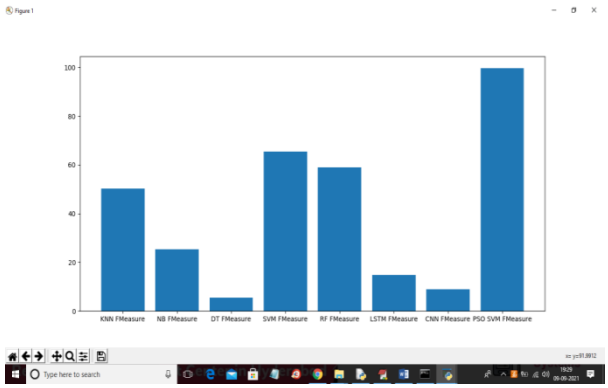

Fig.30. FSCORE Graph

Note: due to huge dataset size execution may take long time and to complete all modules it may take 20 minutes of time

CONCLUSIONS

To sum up, this paper proposes that the PSO algorithm be used with a deep learning mechanism for better classification accuracy. This lets us put all the attributes in order. Accurately selecting and taking out features can also be done. By doing feature sub selection, it gets rid of all the features that aren't needed. Using the PSO algorithm, we can find the best place.

Here, a benchmark dataset (NSLKDD) is used to measure how well something works. This dataset comes from Kaggle. First, an experiment was done to compare known datasets with other methods. Second, by evaluating our method with real datasets, we can show that it has the potential to classify things more accurately than other methods.

Bibliography

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

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

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

[4] M. K. Hussein, N. Bin Zainal and A. N. Jaber, "Data security analysis for DDoS defense of cloud based networks," 2015 IEEE Student Conference on Research and Development (SCOReD), Kuala Lumpur, 2015, pp. 305-310.

[5] S. Sandeep Sekharan, K. Kandasamy, "Profiling SIEM tools and correlation 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 minimization techniques in signature-based intrusion detection systems: A survey,’’ Comput. Commun., vol. 49, pp. 1-17, Aug. 2014.

[7] A. Naser, M. A. Majid, M. F. Zolkipli and S. Anwar, "Trusting cloud computing for personal files," 2014 International Conference on Information 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. ACM CCS 18, Toronto, Canada, 2018, pp. 592-605.

[9] Kyle Soska and Nicolas Christin, "Automatically detecting vulnerable websites before they turn malicious,", In Proc. USENIX Security 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. IEEE BigDataSecurity 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 the Second IEEE Int. Conf. Comp. Int. for Sec. and Def. App., pp. 53-58, 2009.

[12] I. Sharafaldin, A. H. Lashkari, A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization", 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 learning approach to network intrusion detection," IEEE Trans. Emerg. Topics Comput. Intell., vol. 2, pp. 41-50, Feb. 2018

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

[16] W. Hu, W. Hu, S. Maybank, "Adaboost-based algorithm for network intrusion detection," IEEE Trans. Syst. Man B Cybern., vol. 38, no. 2, pp. 577-583, Feb. 2008.

[17] T.-F. Yen et al., "Beehive: Large-scale log analysis for detecting suspicious activity in enterprise networks", Proc. 29th Annu. Comput. Security Appl. Conf., New York, NY, USA, 2013, pp. 199- 208.

[18] K.-O. Detken, T Rix, C Kleiner, B Hellmann, L. Renners, "Siem approach for a higher level of it security in enterprise networks", In Proc. IDAACS, Warsaw, Poland, 2015, pp. 322-327.

[19] en.wikipedia.org,“Security information and event management,” 2016 [Online] Available: https://en.wikipedia.org/wiki/Security- _information_and_event_management.

[20] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proc. IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998.