Non-linear Dimensionality Reduction-based Intrusion Detection using Deep Autoencoder

S. Sreenivasa Chakravarthi[1]
Asst. Prof. of CSE, Center for Intelligent Computing
Sree Vidyanikethan Engineering College &
Research Scholar, SCSE, VIT, Chennai, India

R. Jagadeesh Kannan[2]
Professor and Dean
SCSE, VIT
Chennai, India

***Abstract*—The intrusion detection has become core part of any network of computers due to increasing amount of digital content available. In parallel, the data breaches and malware attacks have also grown in large numbers which makes the role of intrusion detection more essential. Even though many existing techniques are successfully used for detecting intruders but new variants of malware and attacks are being released every day. To counterfeit these new types of attacks, intrusion detection must be designed with state of art techniques such as Deep learning. At present the Deep learning techniques have a strong role in Natural Language Processing, Computer Vision and Speech Processing. This paper is focused on reviewing the role of deep learning techniques for intrusion detection and proposing an efficient deep Auto Encoder (AE) based intrusion detection technique. The intrusion detection is implemented in two stages with a binary classifier and multiclass classification algorithm (dense neural network). The performance of the proposed approach is presented and compared with parallel methods used for intrusion detection. The reconstruction error of the AE model is compared with the PCA and the performance of both anomaly detection and the multiclass classification is analyzed using metrics such as accuracy and false alarm rate. The compressed representation of the AE model helps to lessen the false alarm rate of both anomaly detection and attack classification using SVM and dense NN model respectively.**

***Keywords*—Autoencoder; deep learning; principal component analysis; dense neural network; false alarm rate**

I. Introduction

For detecting various security attacks and breaches with a network Intrusion Detection Systems (IDS) are essential tools. An ID system monitors the traffic in the network (both incoming and outgoing traffic bounds) and performs analysis to raise an alarm if there is anomaly being detected. Based on the approach used for intrusion detection, it can be classified either as signature based or anomaly based method [1]. The signature based intrusion detection method work by matching predefined rules against the pattern in the current network traffic and classifies it as intrusion if the pattern deviates from the normal pattern. It is effective for identifying known type of attacks and yields high accuracy in detecting and low false alarm rate. This approach cannot be used for detecting the new category of attacks as the predefined rules cannot match the unknown patterns in them. The anomaly based intrusion detection systems are capable of classifying even unknown or new category of attacks [2]. As per the theoretical proofs given in many literature the anomaly based detection are more accurate but in practice they suffer from high false alarm rate. Among the several challenges in detecting the intrusions two primary challenges include selection of optimal feature from the network traffic dataset for classification, and unavailability of supervised (labeled) dataset for training the machine learning models [3]. As the attacks patterns are changing over a period of time the set of features used for discriminating the attacks cannot be suitable for effective discrimination of new category of attacks [4].

It is inferred from various literature that machine learning techniques such as Artificial neural Networks, Support Vector Machine (SVM), Naive Byes Classifier, Random Forest (RF), and Self-Organizing Maps (SOM) have been utilized for developing an anomaly based intrusion detection approach. In general the anomaly based classifiers are trained to discriminate between the normal and anomalous traffic. In apart from the training process most of the anomaly based methods adopts a feature selection task to select an optimal set of features to better discriminate the anomalous traffic. During the feature selection process the high dimensional training dataset is reduced in to low dimensional representation and the redundant features are eliminated. For selecting an optimal set of features various methods including Genetic Algorithm, meta-heuristic algorithms, and Principal Component Analysis (PCA) are used [5]. This paper utilizes a deep auto-encoder for finding a compact representation of the input data and a dense neural network for classification of anomalous traffic.

The estimation of compact representation of the network traffic data is a preprocessing task before classifying it. Training a classifier on low dimensional input is much faster than the original input data. This process can also be considered as a dimensionality reduction step which has a regularization effect and prevents over-fitting. The other dimensionality reduction strategies are not effective when compared to AE based approach. When PCA or Zero Component Analysis (ZCA) is used for dimensionality reduction the memory requirements will be high if the feature space is having high dimension. The AE maps the original input data to a compact representation after sending data through two un-supervised training stages. The AE is a generative model and is able to learn and discover the semantic similarity and correlation among the input features [6, 7].

The remaining sections of this paper are structures as follows. Section II of this paper briefs some of the important works relevant to machine learning or deep learning based

network intrusion detection approaches. In Section III, the architecture of the AE model and the procedure for its pre-training and training are presented. The efficiency of the proposed methodology for dimensionality reduction and classification (binary and multiclass classification) are presented in Section IV. Sub-section IV-A elaborates the modeling setup, and Subsection IV-B presents the analysis of the results. Finally, Section V concludes the paper.

II. Related Works

At present the role of Deep Learning can be found in many real time applications including intrusion detection and malware analysis. This section will present a detailed review of deep learning and its application in network intrusion detection. In [8] the authors have conducted a literary and experimental comparison between conventional network intrusion detection methods and deep learning based intrusion detection methods. Their experimental results proved that deep learning based detection techniques offered an improved performance when compared to their traditional counterparts. The problem of imbalanced dataset available for training the detection models can be overcome by using the oversampling technique SMOTE - Synthetic Minority Oversampling Technique [9]. For detecting the malware a stacked denoising AE was used in [10]. Also the same AE based classifier was used to classify executable files [11]. This model used API calls as the features for discriminating the malware from the normal codes. In another approach AE and Restricted Boltzmann Machine was combined used to detect malware using static and dynamic features [12]. In [13] Recurrent Neural Network based AE was developed to extract the features to best describe the malware from raw API calls. The RNN was trained with compact representation from the AE. The deep learning research community has also addressed the problem of outlier detection. The models developed vary on the structure, the application context, and motivation behind the opted strategy [14]. The work proposed in [16] introduced the usage of variation AE for detecting intruders. It was observed that the variation AE performed well in intrusion detection and also in outlier detection. A combination of hybrid AE along with Density Estimation (DE) model was used for detecting anomalies. The model is developed based on the density estimation of the compressed hidden layer representation of AE.

The model was constructed based on estimation of the density among the zipped hidden-layer notation of the applied AE. In another similar research work [17] the authors have used hybrid de-noising AEs in a stacked architecture. Their model utilized hex-based representation of portable executable files, without disassembling. The model training process does not include feature selection or extraction before processing the data which may induce the efficiency of the model. The proposed methodology provided a better discrimination between a legitimate network flow and an anomalous flow. The model output a fixed length vector for both detecting malware and classifying the attacks.

In [15] a three layered RNN architecture was used which utilizes 41 features as inputs and was able to detect intrusion out of four categories. The authors have not studied the performance of the model for binary classification. The nodes of the hidden layers are connected partially and hence it does not have the ability to model high dimensional features.

III. Background Concepts

This section presents some of the background concepts related to the techniques and algorithms used in this study. Deep Autoencoders follows unsupervised learning methodology for learning representation by using dense neural network architecture. The neural network is designed in such a way that the network yields a compressed representation of the input given to it. In a highly non-correlated input data the reconstruction of the input data from the compressed representation is a complex task. If there is a pattern exist in between the data then the pattern can be learned by passing the data through the layers of the neural architecture. Thus the network accepts an unlabeled set of samples and gives xˆ as output which is a reconstructed from the compressed representation of the given input. The network is trained to reduce the error in reconstruction denoted as $L(x, \hat{x})$. The last layer of the encoder decides the size of the data which pass through the decoder layer. The Principal Component Analysis (PCA) achieves a linear dimensionality reduction which is similar to the compressed representation of AE with linear activation function at each layer [18]. For detecting the intrusion patterns in the network data the model must be sensitive to the input for building an accurate reconstruction and at the same time it should be insensitive to the inputs that over-fit the model.

For achieving the above mentioned constraints the network is designed with a loss function with two terms. The first terms makes model to be sensitive to the inputs and the second term avoid the model from overfitting the training data. The alpha scaling parameter controls the trade-off between the two stated objectives. The AE can be considered as a nonlinear generalization of PCA and it is capable of learning non-linear relationship between the input data. Fig. 1 presents the schematic view of the PCA and AE based dimensionality reduction.

$$L(x, \hat{x}) + (\alpha * regularizer) \quad (1)$$

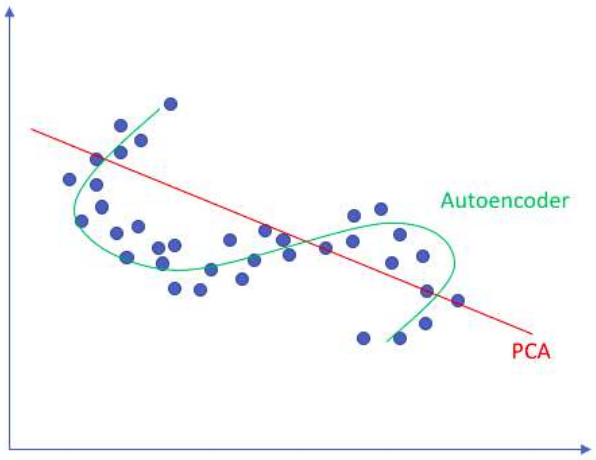

Fig. 1. Linear Versus Non-Linear Dimensionality Reduction.

When a higher dimensional data has to be classified then the AE can be used to estimate a lower dimensional representation of the data and the same can be decoded in to the original input data. The AE is used as a non-linear feature extractor which can be considered as non-linear generalization of Principal Component Analysis (PCA). It performs both encoding of the given input and decoding of the encoded i.e. the compressed feature representation. The encoded data are the dimensionally reduced features. In this study the encoded features are used for identifying anomaly in the network and as well the category of the attack.

The sparsity constraint can be imposed in the network by adding L1 regularization term to the loss function for penalizing the value of activation in the hidden layer along with a tuning parameter λ.

$$L(x, \hat{x}) + \lambda \sum_i \left| a_i^{(h)} \right| \quad (2)$$

IV. Proposed Methodology

The AE is constructed using encoder and decoder which do the transformation of input from the high dimensional space in to a compresses representation in a low dimensional space and reconstruction of the supplied input to the encoder respectively. The architecture of the AE model is presented in Fig. 2. During the training process the weights of the layers in the decoder and encoder are tuned sequentially. The objective is to minimize the reconstruction error. For training the AE model the back propagation learning algorithm is used and the layer weights are assigned randomly and from various experimental results presented in literature it is obvious that the selection of hyper-parameters play an important role in meeting the above stated objective.

For a given set of training examples $X = \{x_1, x_2, x_3, x_4, \ldots\ldots, x_m\}$ where x_i is a d - dimensional feature vector. The encoding layers of the AE maps the d - dimensional input vector to a hidden vector representation h_i and the mapping function is denoted as f_θ and it is defined as in Eq. 3

$$h_i = f_\theta(x_i) = s(Wx_i + b) \quad (3)$$

Where 's' is a sigmoid activation denoted as

$$s(t) = \frac{1}{1 + exp^{-t}}$$

The decoding layers reconstruct the input data represented as vector y_i in d-dimensional space.

$$y_i = g_{\hat{\theta}}(x_i) = s(\acute{W} h_i + \acute{b}) \quad (4)$$

The aim of the training is to minimize the error between input and reconstructed output. The output of the training process is the optimal parameter θ and $\acute{\theta}$. The error is estimated using a loss function and it is denoted as

$$L(x, y) = \frac{1}{m} \sum_{i=1}^{m} \|x_i - y_i\|^2 \quad (5)$$

The number of nodes in the code layer is a *hyper-parameter* that is initialized set before training the AE. The other hyper-parameters include number of layers, number of nodes per layer, and loss function, batch size and dropout rate. Either *mean squared error (MSE)* or *binary cross-entropy can be used depending on the input*. If the input values are in the range [0, 1] then typically cross-entropy can be used, otherwise mean squared error. As the value of hyper-parameters has a direct influence on the success of the training process, an optimal set of hyper-parameters is estimated using evolutionary optimization approach i.e. using Genetic Algorithm. During the hyper-parameter optimization the space of hyper-parameter is searched and an optimal set of parameters are found. Initially a random tuple of hyper-parameters are generated and the fitness value is estimated. Then the hyper-parameters tuples are ranked by their respective fitness values. The hyper-parameter tuples with low fitness values are replace with new hyper-parameter tuples generated by crossover and mutation functions. This process is repeated iteratively until there is no change in the fitness value for a set of consecutive iterations.

While optimizing the hyper-parameters using genetic algorithm; K-fold cross validation error is considered as the fitness function. As per K-fold cross validation for assessing the performance of the AE over a set of selected hyper-parameters a part available dataset is used for training the model and the remaining for testing it. The cross validation error can be estimated as follows.

$$CV(\hat{f}) = \frac{i}{N} \sum_{i=1}^{N} L(y_i, \hat{f}^{-k(i)}(x_i)) \quad (6)$$

where $\hat{f}^{-k(i)}$ denotes the model fitted with k^{th} part of the data removed.

After training an AE model the network data from the data set is applied to it and compressed feature vector is used for training both a binary classifier and multi-classifier for anomaly and specific intrusion type detection as presented in Fig. 3.

For binary classification the Support Vector Machine (SVM) model with a Radial Basis Function (RBF) kernel function was utilized and a Dense Neural Network was used for Multi-Classification.

$$K(X, X') = \exp[-\gamma \|x - x'\|^2] \quad (7)$$

The RBF kernel helps estimating the similarity between vectors by measuring the squared Euclidean distance between them. When the two vectors are close together then the value $\|x - x'\|$ will be small. As the value of $\gamma > 0$, it is obvious that $-\gamma \|x - x'\|^2$ will also be larger. Hence vectors appearing closer will have a large RBF kernel value when compared to vectors which are separated by a nominal distance. For evaluating the performance of the multi-classifier, various models including Naïve Bayes & Random Forest are utilized for classifying the category of attack. The SVM and the dense NN are trained with the compressed representation of the input.

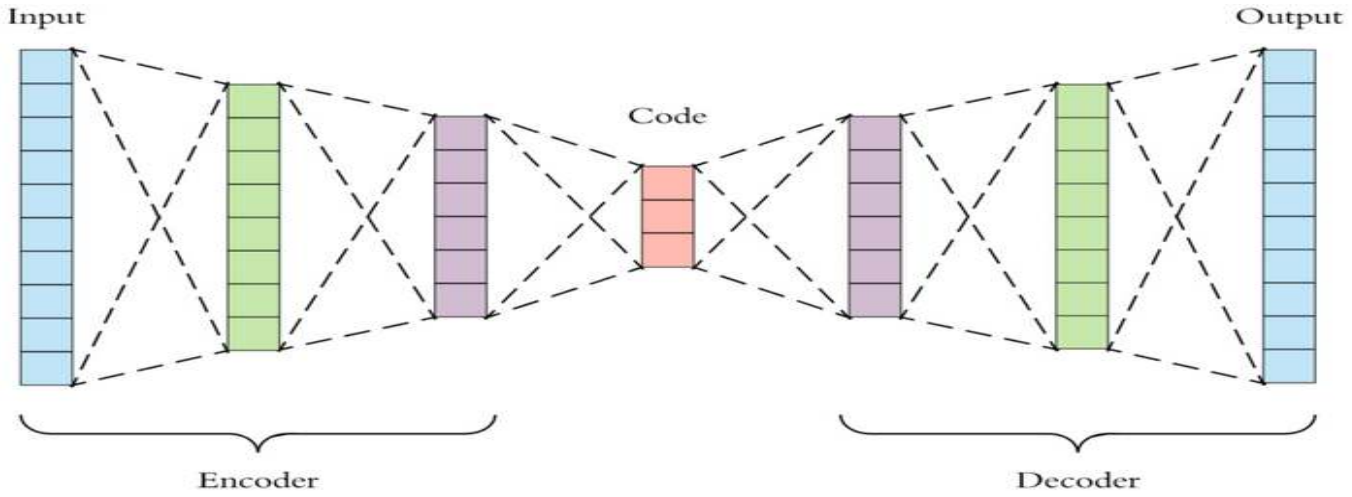

Fig. 2. Schematic View of Autoencoder Architecture.

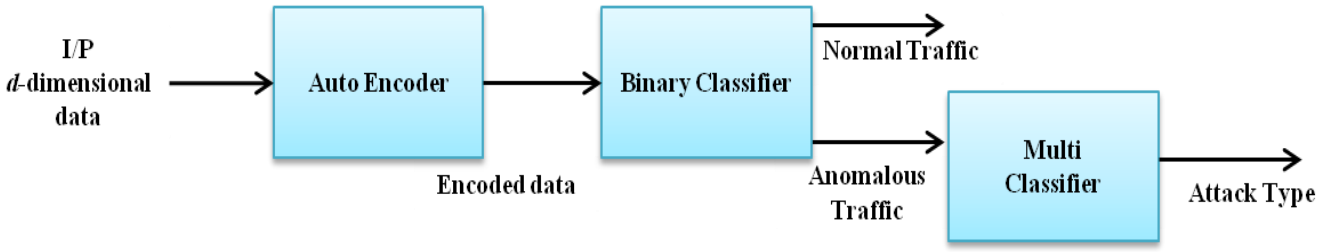

Fig. 3. Block Diagram of Proposed Intrusion Detection.

V. Experiments and Results

A. Dataset

The various attack type detected using signature based approach using a deep neural network is discussed in this section. The dataset used for training and evaluation of the intrusion detection model is obtained from UNSW dataset [19]. The dataset is constructed using the packets generated from the IXIA Perfect Storm tool for both normal and various attack categories possible in the cloud network. The TCP logs were used to extract the relevant attributes which better represents the attacks. Using k-fold cross validation the dataset is divided for training and testing the detection model. Description about the various attacks along with normal data is summarized in Table I.

As discussed in previous section the AE is trained in unsupervised mode and the trained model is used for constructing a compact representation of the input. The Compressed data is used for training the SVM and other multi classifier models. While training the AE the hyper-parameters are estimated using Genetic Algorithm. The error in the reconstruction of the given input to the AE model is analyzed to measure the performance of the AE model.

The numeric features are normalized for removing the effect of original feature value scales. Each feature is normalized and the normalized data is represented as

$$z_i = \frac{x_i - min(x)}{max(x) - min(x)} \quad (8)$$

TABLE. I. Dataset Records Distribution

Type	No. of Records	Description
Normal	2,218,761	Normal legitimate transaction data.
Fuzzers	24,246	Cause a service suspension by sending in a sequence of randomly generated data.
Analysis	2,677	This includes various attacks of scanning the communication ports, spam and penetration of html files in to cloud nodes.
Backdoors	2,329	Unauthorized access to virtual machine and the data stored is gained bypassing the security mechanism by stealth attack.
DoS	16,353	The service is made unavailable to authorized users by disrupting the host connected via Internet.
Exploits	44,525	Exploits the knowledge and information on a known security issue in an operating system of a virtual machine.
Generic	215,481	Irrespective of the structure of the block cipher a common technique against all block ciphers are used with a given block and key length.
Reconnaissance	13,987	Includes all strikes that are capable of collecting information by simulating the attacks.
Shellcode	1,511	The vulnerability in the software is exploited by a small piece of code as the pay-load.
Worms	174	The attack spread itself by replicating to other virtual machines in the cloud environment. Often, it uses a computer network to spread itself, relying on security failures on the target computer to access it.

After normalization all numeric features will be ranged between 0 and 1. The objective of the AE is to minimize the reconstruction error. The training process tunes the weights connecting the layer of the AE model. If the values of the weights are tuned to higher values then it make the generated features more dependent on the structure of the network; but not on the input. In order to avoid this dependency of features on the network structure weight decay regularization is imposed to maintain the weights of the network smaller. Thus the objective function can be defined as

$$L(x,y) = \frac{1}{m}\sum_{i=1}^{m}\|x_i - y_i\|^2 + \lambda \|W\|^2 \quad (9)$$

where $\|W\|^2$ is the weight decay regularization term to ensure a smaller value of weights tuned during the training process. The parameter λ scales the regularization term and can be considered as a hyperparameter of the AE model.

B. Results

Fig. 4 presents the performance of the AE model with a plot of average reconstruction error of PCA and AE for different encoding dimension.

The AE model performs better than PCA at reconstructing the input data set when the number of dimension in encoded output is small, but the error converges as the dimension increases. For very large data sets this difference will be larger and means a smaller data set could be used for the same error as PCA. When dealing with big data this is an important property.

Based on the previous results and comparison with the PCA encoding dimension size is selected as 15 and the value of λ – the weight regularization term is fixed as 3×10^{-3}.There are no guidelines to choose the size of the bottleneck layer in the AE unlike PCA. With PCA, the top $\boldsymbol{k}$ components can be chosen to factor. PCA is used as a guide to choose the value of k - the encoding dimension size.

The performance of the anomaly detection is analyzed based on the following metrics Detection Rate (DR), Accuracy (ACC), and False Alarm Rate (FA).

$$DR = \frac{TP}{(TP+FN)} \quad (10)$$

$$ACC = \frac{TP+TN}{No.of\ samples} \quad (11)$$

$$FA = \frac{FP}{FP+TN} \quad (12)$$

where TP, TN, FP, and FN are True Positive, True Negative, False Positive, and False Negative values of the anomaly detection using the SVM. The training loss trends as depicted in Fig. 5 and the plot in Fig. 6 exhibits the sync between the training and validation loss which proves that the model is not overfitting.

To avoid overfitting of the model is trained with a dropout rate of 0.3 for regularization. The dropout rate is chosen using the genetic algorithm based hyper-parameter optimization technique along with other hyper-parameters.

After completing the training and validation of the AE model then the encoded output is used for training SVM model used as binary classifier for detecting the anomalies in the network. As the accuracy measure alone is not sufficient for analyzing the performance, the Receiver Operating Characteristics curve is plotted to analyze the trade-off between the false positive rate and the true positive rate of the binary classifier and the RoC curve, as presented in Fig. 7, is preferred when the training samples are balanced between the normal and anomalous samples. There are a number of methods available to oversample a dataset used in a typical classification problem. The most common technique for oversampling the imbalanced class in a training set and the technique in the experiments is SMOTE: Synthetic Minority Over-sampling Technique. For a set of training set with ***m*** samples, and ***n*** features in the feature space of the data. To then oversample, take a sample from the dataset, and consider its *k* nearest neighbors (in feature space). To create a synthetic data point, take the vector between one of those *k* neighbors, and the current data point. Multiply this vector by a random number *x* which lies between 0, and 1. Add this to the current data point to create the new, synthetic data point. The area under the curve is equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one (assuming 'positive' ranks higher than 'negative'). The area under curve is given by.

$$A = \int_{x=0}^{1} TPR\left(FPR^{-1}(x)\right)dx \quad (13)$$

The area under the roc curve is estimated as 0.82 which proves that the accuracy of the anomalous detection is high. The Random Forest classifier and Naïve Bayes classifiers were also trained with the dimensionality data generated by the AE model and their results were compared with the results of the deep NN. Fig. 8 presents the plot results obtained from other classifiers and deep NN.

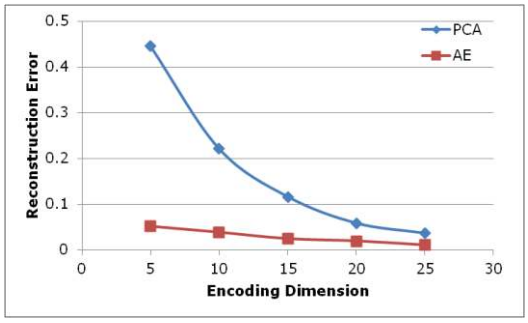

Fig. 4. Comparison of Reconstruction Error between PCA and AE.

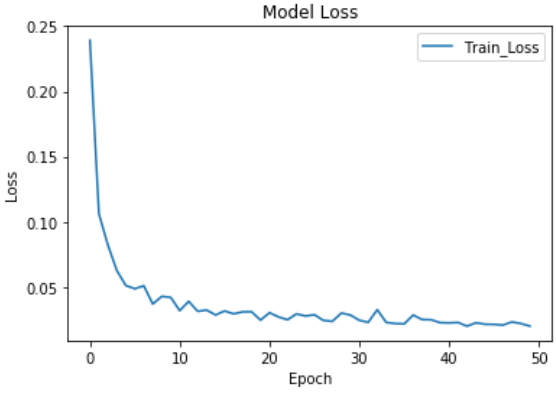

Fig. 5. Autoencoder training loss curve ($\lambda = 0.003$).

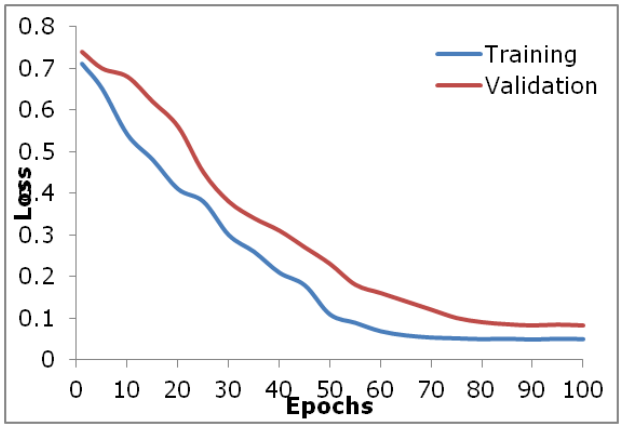

Fig. 6. Autoencoder Training History.

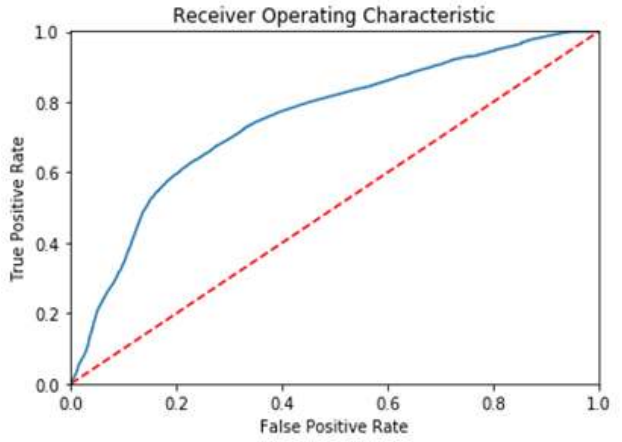

Fig. 7. RoC Curve for Anomaly Detection.

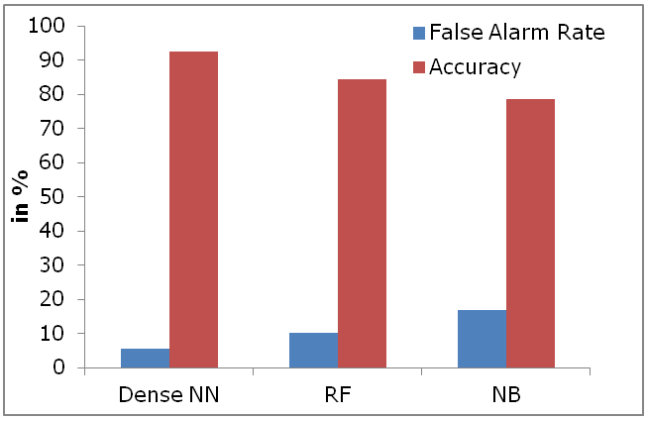

Fig. 8. Performance Comparison of Attack Classification.

VI. Conclusion

This paper presented an intrusion detection using autoencoder trained to generate a compact representation of the input data. The strength of the AE model complements the anomaly detection and attack classification process. The AE model is trained with a set of optimal hyper-parameters estimated using an optimization technique. During training the AE model is prevented from overfitting by using a regularization term and dropout at the hidden layers.

The performance of the AE model is measured using the reconstruction error and compared with the PCA. The accuracy of the intrusion detection process is implemented in a two level. The reduced feature set is applied to the SVM model for detecting the anomaly in the network and subsequently the same data sample is supplied through a dense network if there is an anomaly detected in the network. This methodology is quite simple to implement in a real time environment provided ample of data available for training the binary and multiclass classifiers.

The feasibility and effectiveness of the proposed anomaly detection model was evaluated using ROC curve. Future work will focus on designing a middleware to detect the anomalous traffic in the network in real time using deep learning techniques. This work validated the ability of the deep AE model in reconstructing the data points drawn from different distributions. The overall performance of the intrusion detection on the raw input data, the PCA transformed data, and the autoencoder generated data are compared and the AE seems to be the reason for improvement in accuracy of the attack detection (multi-class classification) especially.

References

[1] Casas, Pedro, Johan Mazel, and Philippe Owezarski. "Unsupervised network intrusion detection systems: Detecting the unknown without knowledge." Computer Communications vol.35, no.7, pp. 772-783, April 2012.

[2] Liao, Hung-Jen, et al. "Intrusion detection system: A comprehensive review." Journal of Network and Computer Applications, vol.36 no.1, pp. 16-24, January 2013.

[3] Bhuyan, Monowar H., Dhruba Kumar Bhattacharyya, and Jugal K. Kalita. "Network anomaly detection: methods, systems and tools." IEEE Communications Surveys & Tutorials vol.16, no.1, pp. 303-336, 2014.

[4] Heba, F. Eid, et al. "Principle components analysis and support vector machine based intrusion detection system." 10th International Conference on Intelligent Systems Design and Applications. IEEE, December 2010.

[5] George, Annie, and A. V. Vidyapeetham. "Anomaly detection based on machine learning: dimensionality reduction using PCA and classification using SVM." International Journal of Computer Applications, vol.47, no.21, pp. 5-8, June 2012.

[6] Shone, Nathan, et al. "A deep learning approach to network intrusion detection." IEEE Transactions on Emerging Topics in Computational Intelligence vol.2, no.1, pp. 41-50, February 2012.

[7] Mahmood Yousefi-Azar ; Vijay Varadharajan ; Len Hamey ; Uday Tupakula, "Autoencoder-based feature learning for cyber security applications." International Joint Conference on Neural Networks (IJCNN). IEEE, May 2017.

[8] Yin, Chuanlong, et al. "A deep learning approach for intrusion detection using recurrent neural networks." IEEE Access vol.5, pp.21954-21961, February 2017.

[9] Zheng, Zhuoyuan, YunpengCai, and Ye Li. "Oversampling method for imbalanced classification." Computing and Informatics, vol.34, no..5, pp. 1017-1037, 2016.

[10] Yuxin, Ding, and Zhu Siyi. "Malware detection based on deep learning algorithm." Neural Computing and Applications vol. 31, no.2, pp.461-472,February 2019.

[11] David, Omid E., and Nathan S. Netanyahu. "Deepsign: Deep learning for Automatic Malware Signature Generation and Classification." International Joint Conference on Neural Networks. IEEE, July 2015.

[12] Li, Yuancheng, Rong Ma, and Runhai Jiao. "A hybrid malicious code detection method based on deep learning." International Journal of Security and Its Applications vol.9, no.5, pp. 205-216, 2015.

[13] De Paola, Alessandra, et al. "Malware Detection through Low-level Features and Stacked DenoisingAutoencoders." Italian Conference on Cyber Security, ITASEC 2018, vol. 2058. CEUR-WS, February 2018.

[14] Patel, Ahmed, et al. "An intrusion detection and prevention system in cloud computing: A systematic review." Journal of Network and Computer Applications, vol.36 no.1, pp. 25-41, January 2013.

[15] Fiore, Ugo, et al. "Network anomaly detection with the restricted Boltzmann machine." Neurocomputing vol.122, pp. 13-23, 2013.

[16] Osada G., Omote K., Nishide T. "Network Intrusion Detection Based on Semi-supervised Variational Auto-Encoder", Foley S., Gollmann D., Snekkenes E. (eds) Computer Security – ESORICS 2017. ESORICS 2017. Springer, Cham Lecture Notes in Computer Science, vol 10493, August 2017.

[17] Zhang, Baoan, Yanhua Yu, and Jie Li. "Network Intrusion Detection Based on Stacked Sparse Autoencoder and Binary Tree Ensemble Method." 2018 IEEE International Conference on Communications Workshops (ICC Workshops). IEEE, May 2018.

[18] Jolliffe, Ian T., and Jorge Cadima. "Principal component analysis: a review and recent developments." Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences vol. 374, no.2065, April 2016.

[19] Moustafa, Nour, and Jill Slay. "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)." 2015 Military Communications and Information Systems Conference (MilCIS). IEEE, pp. 1-6, November 2015.

AUTHOR'S PROFILE

Mr. S. SREENIVASA CHAKRAVARTHI, works as Assistant Professor in Center for Intelligent Computing, Sree Vidyanikethan Research Center, Sree Vidyanikethan Engineering College, Tirupati. He was awarded with B.E from Visveswaraiah Technological University, Belaguam, and M.Tech from Jawaharlal Nehru Technological University, Anantapur, Anantapuram. He is Part-Time Research Scholar in Vellore Institute of Technology, Chennai. He holds more than 16+ years of industry and academic experience. His research interest includes, Design of Algorithms, Machine Learning, Computational Intelligence, Cloud Computing, Network Security, Internet of Things, and Software Quality and Testing.

Dr. R. JAGADEESH KANNAN is senior professor in Vellore Institute of Technology, Chennai with 17+ years of teaching and industrial experience in the reputed organizations. He is Director - Innovation and Entrepreneurship Development Centre, and Chair - Computational Intelligence Research Group in VIT Chennai. He was an active player having a Liaison with various Institutions, R&D organizations and Industries, to promote various technical activities and an efficient Team Leader with outstanding organizational and excellent interpersonal skills to conduct any events such as seminars, symposiums, conferences, workshops, training programs, etc.