TL-DNN: Detection Method for Anomalous Flow Classification in Software Defined Network

[1]Reenu Batra
Ph.D. Scholar, Department of Computer Science and Engineering, SGT University, Gurugram, Haryana, India, reenubatra88@gmail.com

[2] Mamta Dahiya
Professor, Department of Computer Science and Engineering, SGT University, Gurugram, Haryana, India, malik.mamta@gmail.com

Abstract:

Software Defined Networking (SDN) is a rising network systemwhich offers theconcept of separationdata plane from the control plane.Main focus of the control plane is to govern traffic while Data Plane is used in traffic forwarding the based on the control plane's decisions. Also, the SDN unifies the control plane, allowing numerous data plane'scomponents to be controlled by a controllerapplication. The SDN's control plane exercises direct control over the components and the state of dataplane. Unfortunately, the SDN network architecture is complicated and poorly built to support fine-grained security policies. This is due to lack of the tight assimilationof the control and data planes. Another SDN's core challenges is figuring out how to efficiently handle network packet anomalies because SDN is logically centralized, an anomalous control plane attack can bring the entire network system down. The existing SDN anomaly detection methods, on the other hand, are known for their low accuracy and overall performance. They do not, however, facilitate gradual learning. As a result, improving and optimizing existing methods and establishing high-precision as well as real-time classification models for abnormal flows is critical. Also the flow detection mechanism must be able to support giga-bit level networks in real-time. The goal of this study is to develop such efficient anomaly detection algorithm for SDN networks.The work demonstrates this by utilizingDeep Neural Networks combined with Transductive learningfor identifying anomalous flows. An attempt is made to further improve the learning capabilities of deep neural networks andsimultaneously reducing the false positives. This is achieved by processing the input flows using transductive confidence machines and then enabling deep neural networks tomake better decisions during the learning process. Also a new transduction measure named skewness is added in this work to work alongside the exiting transduction measures. In comparison to previous techniques, the proposed TL-DNNalgorithm gives considerably reduced False Positives and greater Accuracy based on experimental findings on the KDDDataset. The work also demonstrated the superiority of the suggested method in terms of FPR, TPR, MSE, and Accuracy through various simulations.

Keywords: Software Defined Network (SDN), Transductive Confidence Machines, Deep learning, Densenet.

DOI Number: 10.14704/nq.2022.20.9.NQ44212 **Neuro Quantology 2022; 20(9):1824-1839**

1. Introduction

SDN is a rising technology that allows us to advance us in how networks design and manage. It is actually altering the tradition techniques to design and manage networks. It has two significant features [1, 2, and 3] First, SDN divides the control plane (CP) from the data plane (DP). CP is the one who manages traffic,while DP forwards the traffic based on the control plane's commands. Second, SDN unifies the control plane, allowing numerous data plane members (switches, routers, and other middle boxes) to be controlled by a single software control program. With the use of a well-defined API like OpenFlow, CP exerts direct control over the state of the network's DP elements. Unfortunately, existing network architecture is complicated and poorly built to support fine-grained and QoS-aware traffic engineering. Because of the tight integration of the CP and DP, network traffic monitoring is more difficult, resulting in low QoS-aware flow management as well as poor resource allocation. Separately managing network devices is difficult, time-consuming and costly.

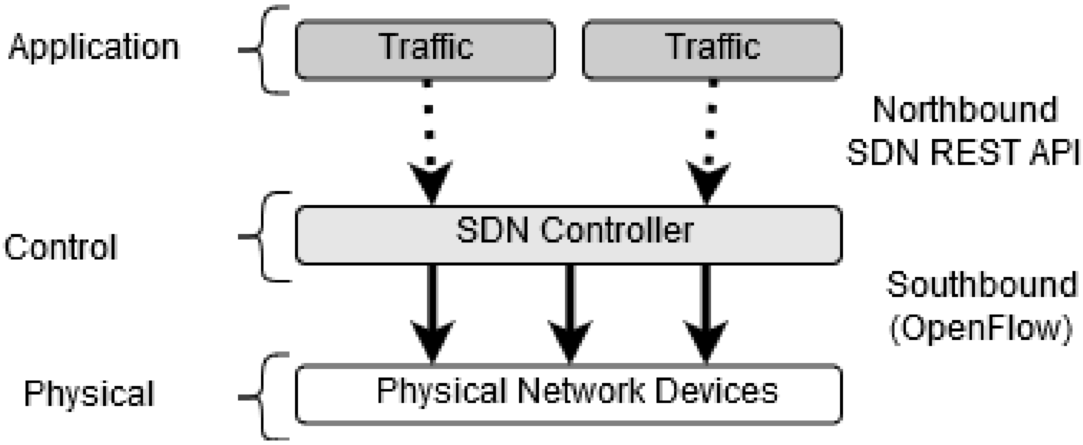

Figure 1: SDN Architecture

While on the other hand SDN is a new network architecture (as shown in figure 1) which assures to make network management easier, improved network resource usage, and increased network growth and innovation. SDN abstracts the network layers by dividing the CP (networking logic) from the DPelements and transferring them to an external entity (controller). In comparison to typical network designs, SDN separates network devices from management and allows both planes to grow independently, providing design flexibility and programmability. A centralized SDN controller makes it easier to deploy, upgrade, and manage traffic forwarding devices in SDN architecture [2]. Network monitoring and configuration become considerably easier since they can be done from a centralized location in the network lowering operating costs dramatically.

SDN along with its separation of planes feature comes up with many additional advantages like design flexibility, easy updating and up gradation, and various others spectacular advantages of SDN [4].SDN gives you freedom of location. Users can use network resources without concern for where resources are physically placed, how much they cost, or how they are arranged etc. SDNs are highly scalable and are able to change size, quantity as per the demand of the situation. Any time new workloads, sites, devices and resources can be added to the network. Virtualization as discussed above easily provides scaling. SDNs are easily programmable and can change behavior on mere one click. Moreover they can control and administer number of devices with just one command. SDNs are highly automated which means minimal manual involvement and minimal troubleshooting which lowers both the operational expense as well as lower the downtime. SDN can automatically Provide, Re-provide and Segment the resources such as adding new workloads, sites, devices, etc. SDNs enhances the network performance by optimizing network device utilization. SDN performs bandwidth management, handle traffic efficiently, perform load balancing and can handle a failure fast. SDN can place Intrusion Detection Systems (IDS), firewalls, Load balancers, and other resources on demand appropriately on the traffic path.

SDN is its infancy and faces various challenges [2, 5, 6, and 7] during deployment and after deployment too. One of the biggest challenge SDN face is security that has been the focus of this paper. To yet, there has been little discussion in the research industry society on the security issues surrounding SDN. If SDN is to be accepted in a larger deployment, a stronger attention on security is essential. The Open Networking Foundation (ONF) has formed a security working group to address security issue [8]. A lot of challenges are raised here, all points requiremore research and development to find security solutions. Possible security flaws can be found throughout the SDN framework. Authentication and authorization systems that allow numerous organizations to use network resources at the same time providing sufficient protection of these resources have been raised at the controller-application. All applications do not need the equal level of network access, thus a security model is be required to be established.A Denial of Service (DoS) attack is one type of malicious attack that can happen to SDN. Techniques for detecting and protecting against threats will continue to improve.Techniques for detecting and protecting against threats will continue to improve. Individual businesses must, however, set their security policies properly and exhaustively so as to fully utilize the available resources. The existing SDN anomaly detection methods [9, 10 and 11], on the other hand, are known for their low accuracy and overall performance. They do not, however, facilitate gradual learning. As a result, improving and optimizing existing methods and establishing great precision real-time classification models for abnormal flows is critical. Also the flow detection mechanism must be able to support giga-bit level networks in real-time. The aimof study is to develop such efficient anomaly detection algorithm for SDN flows.The work demonstrates this by utilizing Deep Neural Networks combined with Transductive learningfor identifying anomalous flows. An attempt is made to further improve the learning capabilities of deep neural networks and simultaneously reducing the amount of time and resource required. This is achieved by processing the input flows using transductive confidence machines and then enabling deep neural networks tomake better decisions during training. To test the efficiency and performance of the proposedsystem, many methods are tested for anomaly detection using NSL-KDD data set [12].

The rest of this manuscript is sectioned as follows: the 2nd section presents review of the literature, 3rd section focuses over the proposed TL-DNN system for SDN and elaborates different transduction measures used and the proposed skewness measure alongside the proposed dense-net architecture, 4th section focuses on results achieved and the performance evaluation, section five presents conclusions as well as related future works.

2. Literature Review

In the statistics literatureand data mining, anomalies are known as abnormalities, outliers,or deviants [13]. Traditionally the various methods to detect anomalies are Supervised, Unsupervised and Reinforcement Learning (RL) [14][15] [16]. Traditional algorithms, on the other hand, perform poorly in finding outliers in medical images and sequence datasets because they fall short to capture complex structures in the data. Moreover it becomes practically hard for existing methods to expand to such huge size data to discover outliers when the volume of data grows to terabytes. Deep Learning (DL) networks [17]also reduces the requirement for domain experts to discover features and minimize complexity because it incrementally learns high-level characteristics from data. As a result, advocates for solving the problem from beginning to end using raw input data in domains like speechand text [18, 19, and 20].

Anomaly based intrusion detection is implemented utilizing numerous machine learning methodology, a unique dataset, and a module that communicates with the centralized controller. ML-based IDS trains to distinguish attack from usual traffic by creating a pattern through multiple features in the traffic dataset. The four types of AIDS approaches [21] include supervised learning [22], unsupervised learning, reinforcement learning, and deep learning. In comparison to older techniques, Machine learning based intrusion detection is more precise. To extract information from intrusion databases different ML techniques can be used to perform IDS example Neural Networks [23], Naive-Bayes [24], Decision Tree [25], Logistic Regression [26], Support Vector Machine (SVM) [27] etc.

Singular Value Decomposition (SVD) is a technique for reforming a matrix into other small matrices by various linear approximations which reveal the matrix's original meaning-structure. Its purpose is to find the optimal feature set that accurately predict the intrusion detection [28].ICA isalso utilized to reveal hidden elements which lie beneath a set of random features. In [29] authors proposed a redundancy-based resilience solution. Theyproposed a specialized network sub layer that can handle context by collecting consensual information from the control network's driver nodes on a regular basis and discerning perspective differences using data mining approaches like k-nearest neighbor [30]and k-means [31]. Some authors have used various ML approaches to filter abnormal traffic and discover intrusions in SDN [32].

The SVM technique works by creating an n-1 dimensional dividing hyper plane to distinguish between two classes in an n-dimensional space. Many features in IDS datasets are redundant or have little influence in classifying data items. As a result, feature selection must be taken into account during SVM training [24].

Multiple classifiers are trained at the same time to recognize distinct attacks, and their results are then combined to boost the detection rate. The abilityof ensemble is typically superior to that of one sole alone classifier, as it can improve weak classifiers to yield improve results [34]. Boosting, Bagging, and Stacking are just a few of the ensemble approaches that have been presented. Boosting is a term that implies a group of algorithms that can turn weak classifier into strong classifier [35].

For building IDS, Deep Reinforcement learning makes uses ofDL and RL principles [36]. Reinforcement learning includes interaction with an agent in an environment trying to accomplish some kind of objective.

DL is a sub part of ML in which a computer creates numerous layers as an output from a hierarchy of data based on prior experience. Both supervised and unsupervised DL are possible [37]. Data can be categorized in supervised deep learning, whereas data patterns may be studied in unsupervised deep learning. DL is very muchassociated with AI, in which machines learn via experience and ultimatelysubstitute human intellect. DL analyses enormous data by use of algorithms created by intelligence of human on neural architecture. The neural networks are now a days referred to as DL because can they have many deep layers which makes them learn.Different ML methods have been useful in Distributed DoS detection of SDN [38] [39].

Transduction is the process of reasoning from training data to classify test data attributes. In other words transduction means an attempt to find out the classifications of the points in the test data set given the categorizationof the points in the training data set. Individual data instances are given reliability metrics by transduction, which simply implies that the data instances are independent and created by the same stochastic mechanism. TCM [40,41] were constructed using transduction to first find the unknown class of a point and then to identify anomalies in a data set. TCM gives a statistical confidence to its classifications and anomaly classification.

Researchers have developed TCMs that can predict the unknown class of a data instance and assign confidence to the prediction, as well as identify outliers in a data collection, using transduction. At present the majority of machine learning algorithms lack measurements that may indicate how "excellent" the estimations are. In [42], proves that in SDN environment the combination algorithm of K-Nearest Neighbors (KNN) with TCM has great precision. But, KNN's time consumption is high and it alsoinvolveshuge amount of computing resources. KNN puts a huge stresson the SDN detection device since it must compute and sort the distance between each training entity and every detected node, which might result in a significant detection delay. For this same reason the controller can only be trained using very small no of instances

3. TL-DNN Design

The proposed method (TL-DNN) uses rules such as strangeness, independence, probabilities and newly proposed skewness to infer predictions. These measures are used to acquire the confidence level of a detection point throughstochastic algorithm, and it utilizes the probability to check if the detection point belongs to the attack class. The larger the value of p is, the more likely it is that the detection point corresponds to the attack class. Also predictive performance TCM is combined with deep learning algorithm, however the confidence measures it outputs giveswell-nighvaluable information for individual predictions the deep learning algorithm classifies the instances with these confidence measures more effectively. To test the efficiency and performance of the proposed system, different techniques tested for anomaly detection, the work uses NSL-KDD data set [12]. Proposed method (TL-DNN) algorithm is explained below:

Let's take a training set T of a vector point n elements i.e. T = $(t_1, C_1), (t_1, C_1) \dots . (t_n, C_n)$

where T = (F_1, F_2 .... F_n) is collection of attributes, for instance j is the instance and having class C_j

from a subset of $\{1,2,...,\mathbb{C}\}$, where $\mathbb{C}$ is number of classes, for the SDN attack identification $\mathbb{C}$ signifiesas an attack class. $\mathbb{C}$ can be an attack id from the available classes of attacks in the dataset for example, load module, multi-hop, guess_passwd out of total 19 attack classes. Also, we have a test set of examples alike training dataset, but in testingreal classifications ids remain hidden. The major aim work is allot each test example with one value from attack class $\mathbb{C}$ .To expand insight in the predictions, we want to give some confidence measure for each classification; next sections will explore how we can assign the confidence measure to the predictions.

Parameters: r - Feature vectors, ϑ - count of transduction vectors (π, Ψ, $\mu_1(\Omega_i)$, $\mu_2(\varphi_i)$, σ)

Input: **W**(DatasetNSL-KDD with N data points)

Output: Classification of every instance i in dataset

Initialize W' with size of W data set having updated dataset

Step 1. ***while*** *i=1* to k *perform*

Step 2. *Normalize the dataset and save to* W'

Step 3. ***close while***

Step 4. ***while*** *i=1* to k *perform*

Step 5. *Compute l2-Norm*$(f_i, \dots f_r)$ *for every class with respect to normal class*

Step 6. *Save distances*

Step 7. ***end while***

Step 8. ***while*** *i=1* to k *perform*

Step 9. For all data points input dataset. Compute strangeness π_{iy}

Step 10. Compute Independence Measure (Ψ)

Step 11. Compute class wise probability of abnormality $\mu_1(\Omega_i)$ *and* $\mu_2(\varphi_i)$

Step 12. Select a class c having shortest dist $(\mu_1(\Omega_i), \mu_2(\varphi_i))$ *are nearest*

Step 13. Save weight for the class with having shortest distance

Step 14. Compute Skewness (σ) *for each class c*

Step 15. ***end while***

Step 16. Assign existing dataset W with weightsinto W'

Step 17. Train dense-net classifier on updated dataset W' *and true classes* $\mathbb{Y}$

Step 18. Get $\hat{\mathbb{Y}} = densenet(W')$

Step 19. Evaluate Confusion Matrix, Accuracy, precision, recall, F1-Score using true class $\mathbb{Y}$ *and estimated class* $\hat{\mathbb{Y}}$

The algorithm initializes by inputting the **W** dataset (a NSL-KDD Dataset) with N data points.

So as to speed up the training process, the next step is to normalize the dataset within a same range and update the same dataset with normalized values. Next step is to compute the distances among the points of each class with respect to the normal class and store their distances. Next step is to compute strangeness (to find the degree of uncertainty), independence measure (to find error margin between support point & its feature vectors) and probability of abnormality (how mucha vector point of a class belongs to that class) for all the data points in the feature vector. As well as for all the pints in feature vector find class with the minimum distance from transduction vectors $(\mu_1(\Omega_i)$, $\mu_2(\varphi_i))$ and attach a confidence value with it, also compute its skewness. Attach the entire dataset X with the confidence value $\widehat{\boldsymbol{W}}$. The final step is to train the updated dataset with confidence values on densenet classifier, the densenet classifier is train against the input class $\mathbb{Y}$. Next the trained dense-net is used to predict the estimated class $\hat{\mathbb{Y}}$ for the input X. At last check the performance of classifier by evaluating the output using confusion matrix, Accuracy, precision, recall, F1-Score.

3.1 Dataset Normalization

A normalization technique [43] is necessary to ready the raw data for training. Furthermore, it is essential to speed up the training process for neural networks. There are several forms of data normalization. It sometimes used for scaling input data in the required range in order to decrease biases, the biases may exist for a single or multiple features. It's especially useful in modeling applications when the inputs are on vastly different sizes. If,x_i is the presentinput, $x_{min}$and $x_{max}$are the min and themax values respectively. So the min-max normalization is:

$$\mathrm{Norm}(X_i) = \frac{x_i - x_{min}}{x_{max} - x_{min}}.$$

3.2 Distance Function

For determining the distance between two pointsthe most commonly used distance function is "Euclidean Distance." Let A and B be the two points shown by A = (F_1, F_2, F_N) and B = (C_1, C_2, C_n) feature vectors respectively, where k is the feature space's dimensionality.The normalized Euclidean metric to compute distance between A and B, is given by

$$dist\ (A, B) = \sqrt{\frac{\sum_{i=1}^{k}(F_i - \mathbb{C}_i)^2}{k}}$$

3.3 Strangeness measure (1.1

Strangeness measure (π) also known as nonconformity score,the probability valueis a confidence interval test for randomness testing, which is non-computable in its general form. The concept it conveys is the uncertainty of the point being measured in relation to all other categorized cases in the class. The more the value of π, the greater the degree of uncertainty. It's also known as the ratio of the total of the distances between a point and its classification inside the class and the total of the distances between the point and its classificationout of the class. The strangenessiy of a point i with relation to a class y can be shown as:

$$\pi_{iy} = \frac{\sum_{i=1}^{k} D_{ij}^{y}}{\sum_{i=1}^{k} D_{ij}^{-y}}$$

3.4 Independence Measure (1.1

The error margin between TL-DNN classification boundaries, as well as the distance between the decision points, is the Ψin the TL-DNN classifier.Independence measure(Ψ) is shown in equation below

$$\varphi_{\mathbb{Y}i} = \sum_{i=1}^{\Psi} Distance_{pq}^{\mathbb{Y}}$$

In the equation,Ψ represents feature vectors, φ_{yi} indicate individuality of the instance for a class𝕐; $Distance_{ij}^{\mathbb{Y}}$ is the distance between two features f_1 and f_2. Also $\mathbb{Y}i$ denotes the point's margin of error for a certain class 𝕐. The lower the independence value, the more probable the feature vector point is independent to the class 𝕐 i.e it is very likely corresponds to this specific class. Contrast to strangeness, φ is employed to determine whether near an instance i is to the class 𝕐.

3.5 Probability of Abnormality (1.)

The Confidence measure, Probability of abnormality ($\boldsymbol{\mu}$) can beemployedto determine the probability of anomaly of a feature vector in relation to other instances in the dataset. It determines how much the features in the instance fitted in to that class.As the value$\boldsymbol{\mu}$for the class 𝕐grows, the probability that the instance belonging to this particular class increases likewise, $\boldsymbol{\mu}$valuefor a TL-DNN canbe shown as:

$$\mu_1(\Omega_i) = \frac{\{j = (1 \dots n): \Omega_j \geq \Omega_i\}}{n+1}$$

In the above equation, the SM of instances *i* and *j* for the class $\mathbb{Y}$ is Ω_i and Ω_j respectively $\{j = (1 \dots n): \Omega_j \geq \Omega_i\}$ indicates j instance as the number of feature vector points whose given strangeness is greater thanvector j in class $\mathbb{Y}$. Likewise μ_2 confidence measure μ can be determined from independence measure φ as:

$$\mu_2(\varphi_i) = \frac{\{j = (1 \dots n): \varphi_j \geq \varphi_i\}}{n+1}$$

Theconfidence measure μ_1 and μ_2 are taken to classify abnormal instances i and$j$. If μ_1 and μ_2 both the confidences are normal, the feature point is also considered as normal.

3.6 Skewness

A distribution is called to be symmetrical distribution when it is equally distributed on both the sides of the mean value. The symmetrical distribution shape can form a u-shape or bell shape. The values ofmedian, mean, and mode are all equal as visible in the figure below. A skewed distribution on the other hand is used to measure the level of asymmetry. Skewness is a measure of asymmetry that occurs when the information contained in the packets deviates from the norm during the attack.

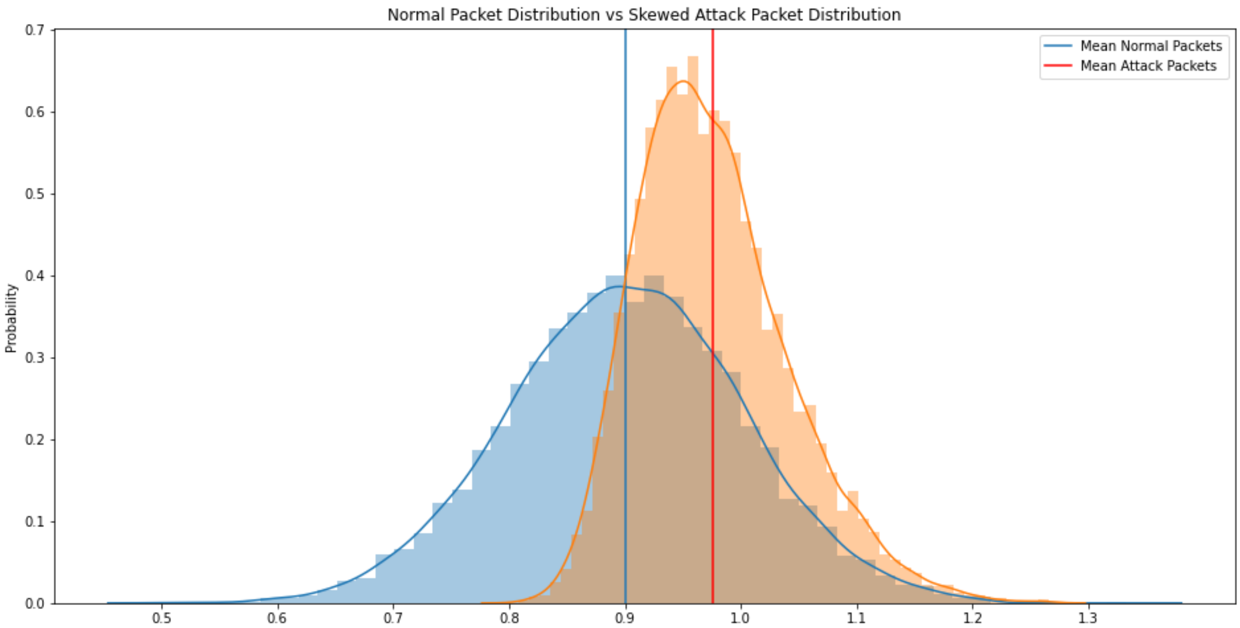

Figure 2: Negatively Skewed distribution of the Attack Packets with Normal distribution

The skewed distribution happens to be tilted more on either of the side -left or right, this is due to the fact that probability of data being more or less than the mean is higher which makes the distribution asymmetrical (refer figure 2). It could also indicate that the information contained the packets is not equally distributed.The skewness distribution of packets can be both positively skewed and negatively skewed. Negatively skewed in a distribution are the data points that are more concentrated towards the right hand side of the distribution as visible in the figure above this makes the all three mode,median and mean bend towards the right hence these values are always negative in this distribution mode is greater than median and meter is greater than mean.

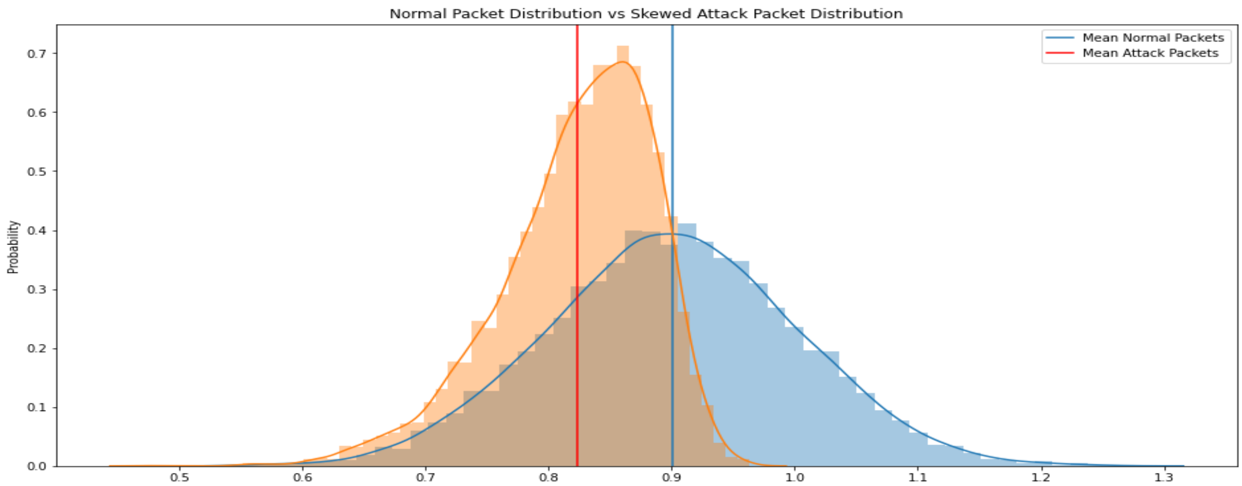

Figure 3: Negatively Skewed distribution of the Attack Packets with Normal distribution

Also a distribution is said to be positively skewed when the values are more concentrated towards the right side and the left tail is spread out as visible in the graph as shown in figure 3. Hence the results are skewedtowards the left therefore the mean median and mode are always positive in this distribution. Mean is larger than median and median is larger than mode. Additionally, the skewness measure enables us to determine the magnitude and direction of the frequency distribution's departure from symmetry. Positive or negative skewness can be seen graphically depending on whether the left tail or the right tail is longer, but we are unable to determine the magnitude of the deviation in a negatively skewed symmetry, where values are always negative, which is the symmetrical distribution, and in a positively skewed symmetry, where values are always positive.

$$Skewness\ (\sigma) = \frac{\sum_{i=1}^{n}(x_i - \overline{x})^3}{(n-1)\ \times std^3}$$

Where, $\overline{x}$ is the mean of normal packets, n is no of samples and std is the standard deviation

3.7 Network Architecture

The TL-DNN is built using deep neural network (refer figure 4) which is a dense-net [44] the densenet architecture is depicted below. In this architecture packets are sequentially fed into DNN layers for the training as show in the figure, the DNN constitutes of multiple deep layers of neural networks. Each Part of the DNN is further explained in the next sections.

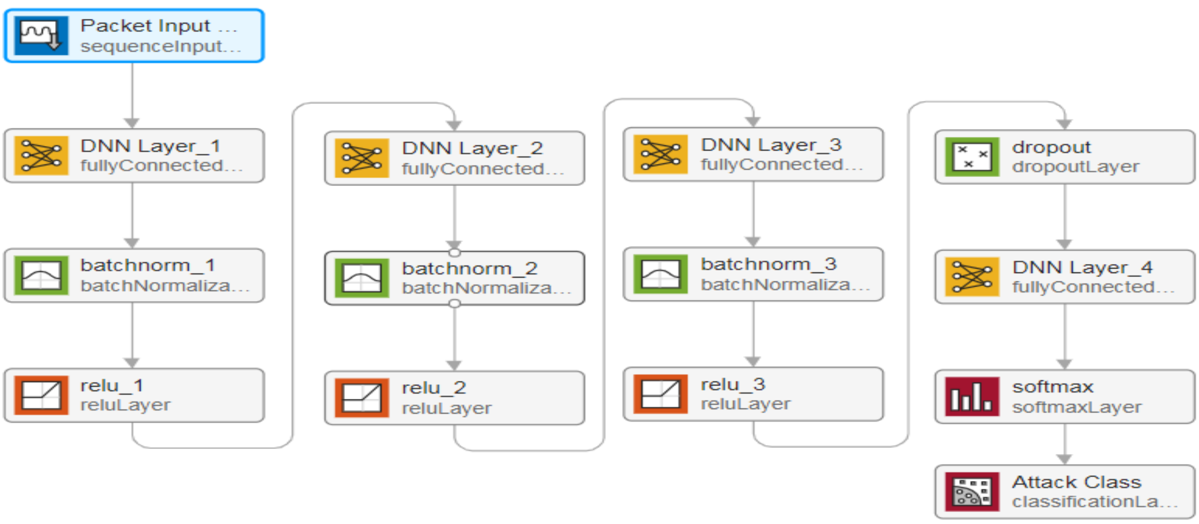

Figure 4: Block by Block Layout of the Proposed Anomaly Detection Framework

3.8 Dense net Blocks

Dense net is a feed-forward deep network that connects all layersdirectly [45, 46, and 47]. Every layer usesinputs as feature-maps from previous levels, and it also uses its own feature-maps as inputs into next layers. Each layer in Dense Net gets additional input from every previous level (refer figure 5) and sends its own feature-maps to all successive layers. Each layer obtains "collective knowledge" from the above levels, just like concatenation. Because each layer gets feature maps from all preceding layers this makes network more compact and thinner, consequently there are lesser channels.For each composition layer, function like Pre-Activation Batch Norm (BN) and Rectified Linear Unit and 3×3 Convolution are performedwith output feature maps of k channels [48].

The network is made up of L layers, all layers put into operation a non-linear transformation $H_G(.)$, where G indexes the layer [47] and $[\mathbf{j}_0, \mathbf{j}_1, \dots, \mathbf{j}_{\ell-1}]$ is the concatenation of the feature-maps created in $0, \dots \dots, G-1$ layers.

$$\mathbf{j}_G = H_G([\mathbf{j}_0, \mathbf{j}_1, \dots, \mathbf{j}_{\ell-1}])$$

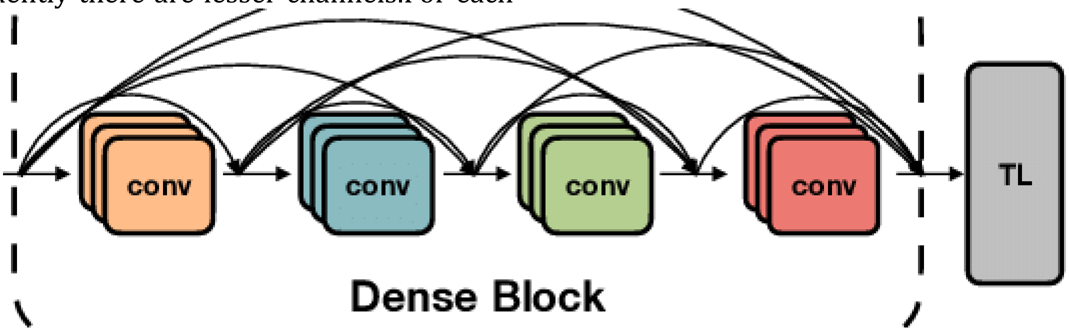

Figure 5: Proposed Dense Net block

4. Results and Analysis

The University of California Irvine ML repository has the Network Security Laboratory (NSL) Knowledge Discovery in Databases (KDD) Cup 99 benchmark dataset. The NSL-KDD Cup 99 dataset is the latest version of the KDD Cup 99 dataset [49]. The NSL-KDD Cup 99 dataset addresses a few of the KDD Cup 99 dataset's shortcomings. In the 3rd International Knowledge Discovery and Data Mining Tools Competition, the KDD 1999 cup benchmark intrusion detection dataset was used.

4.1 Performance Measures

Accuracyof a classifier predicting the class label for both cases -training as well as testing, is the performance measure of a classifier. To check the performance of learning classifiers these four concepts must be clear: 1. TP - True Positivesmeaning learning algorithm correctly classifying positive instances, 2.TN- True Negatives meaning learning algorithm correctly classifyingnegative instances , 3. FP - False positiveslearning algorithm incorrectly categorizednegative instances as positive, and 4. FN -False negatives, learning algorithm incorrectly categorized positive instances as negative. These Matrices are part of a general term called confusion matrix. Also, the accuracy of a model calculated using confusion matrix is the shown by the matrix given below.

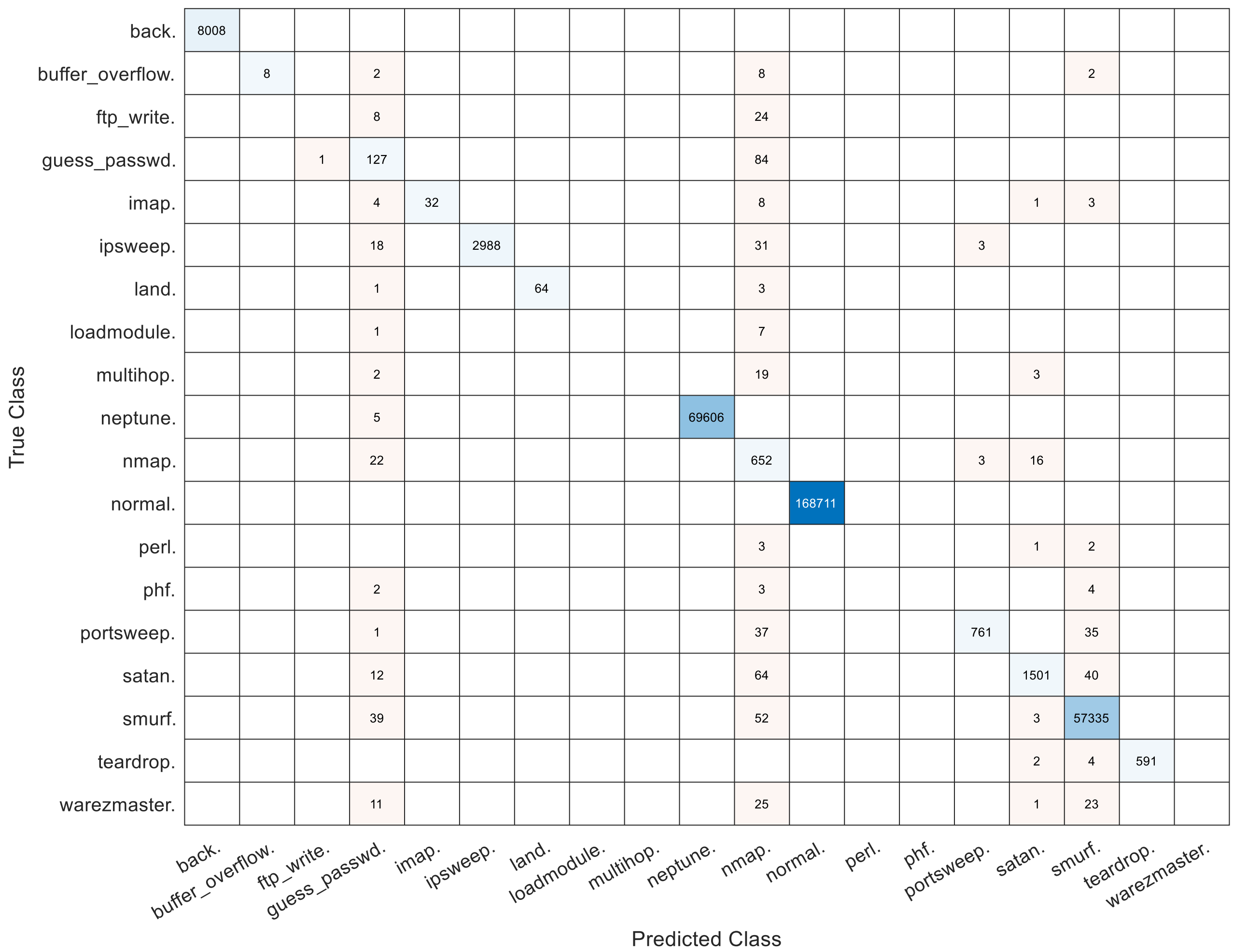

Figure 6: Confusion Matrix of Proposed TL-DNN Method

In the Figure 6, we have evaluated confusion matrix for 39 attack classes and control group. Normal class is about the packets having attacks and normal class is about the packets not showing any sign of attack. This figure 6 shows that the value for True Positive is, indicating the no of packets classified by the model as normal. The value of false positive is, indicating the percentage of the packets not having attacks classified by the model as attackers. The values of false negative are those packets which are attacks however don't get classified as attacks class. Similarly, the values of false positivesare those packets which are normal attacks however get classified as attacks class. Using values of this confusion matrix other metrics are derived such as accuracy, precision, recall etc. which are explained as:

4.2 Accuracy of the Network

Accuracy ofa learning algorithm is the percentage of test instances that are correctly categorizedon a given test data.

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

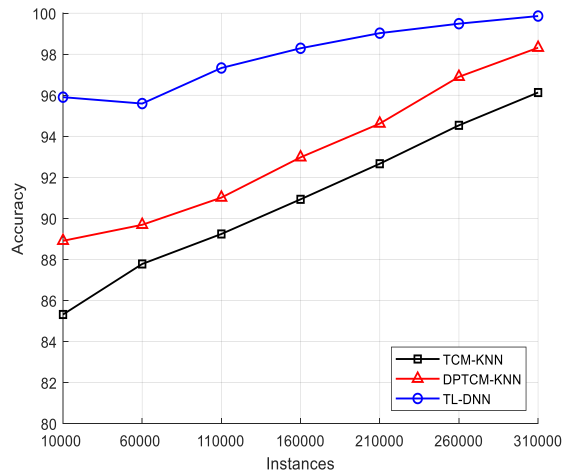

Figure 7: Comparison TL-DNN, TCM-KNN and DPTCM-KNN in terms of Accuracy

For 200K instances In terms of accuracy, TL-DNN outperformed all other methods TCM-KNN and

DPTCM-KNN significantly. Accuracy measures for anomalous flow detection obtained using the TL-DNN, TCM-KNN and DPTCM-KNN for 20K Instances are 95.91, 85.32 and 88.91 respectively. Also in terms of accuracy the TL-DNN shows improvement in accuracy against the baseline methods which is 11.04% better than TCM-KNN, 7.30% better than DPTCM-KNN as depicted in figure 7. This trend continues for all instances from 200K to 500K instances. Even at 50K instances the TL-DNN outperformed all other methods TCM-KNN and DPTCM-KNN significantly.

At 50K the Accuracy obtained using the TL-DNN, TCM-KNN and DPTCM-KNN are 99.86, 96.14 and 98.32 respectively.Which is 11.04% better than TCM-KNN, 7.30% better than DPTCM-KNN. It is evident from the table 2 on average TL-DNN achieved 97.93% whereas the baseline methods TCM-KNN and DPTCM-KNN achieved only 93.20% and 90.94% accuracy respectively, even though the proposed work had 100 times more instances(refer table 1).

Instances	TCM-KNN	DPTCM-KNN	TL-DNN
200000	85.318	88.909	95.91
250000	87.78	89.689	95.598
300000	89.243	91.015	97.333
350000	90.932	92.978	98.294
400000	92.667	94.622	99.03
450000	94.538	96.902	99.491
500000	96.136	98.318	99.864
Average	**90.945**	**93.205**	**97.931**

Table 1: Performance of the TCM-KNN, DPTCM-KNN and TL-DNN

4.3 Sensitivity

The sensitivity is the true positive rate of a learning algorithm (the ratio of positive cases that are correctly recognized by the model), Sensitivity is also called as recall which is a measure of completeness (the percentage of positive casesthat are categorized as such). The more the sensitivity the better the algorithm.

$$\text{sensitivity} = \frac{TP}{P}$$

$$recall = \frac{TP}{TP + FN} = \frac{TP}{P}$$

When evaluating for the recall at 200K instances the TL-DNN outperformed all other methods DPTCM-KNN and TCM-KNN. As shown in the table 3at 200K the Recall obtained using the TL-DNN, DPTCM-KNN and TCM-KNN are 94.04, 87.38 and 84.19 respectively.Recall of the TL-DNN was 6.66 and 9.86 higher than DPTCM-KNN and TCM-KNN as shown in figure 8.

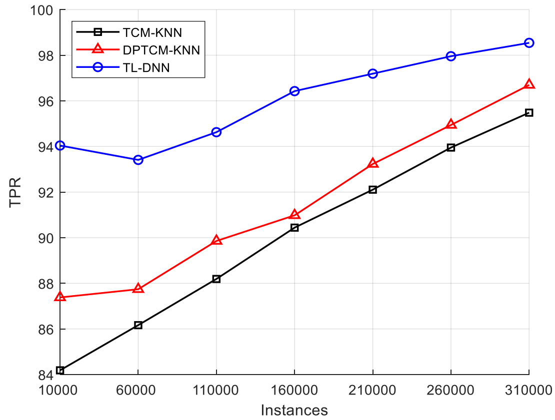

Figure 8: Comparison TL-DNN, TCM-KNN and DPTCM-KNN in terms of TPR (Recall)

Overall the TL-DNN showed the improvement of 7.08% and 10.48% against baseline methods for 200K instances. If we compare the proposed method at higher instances i.e. 500K the Recall obtained using the TL-DNN, DPTCM-KNN and TCM-KNN are 98.54, 96.69 and 95.48 respectively(refer table 2).

Instances	TCM-KNN	DPTCM-KNN	TL-DNN
200000	84.185	87.38	94.04
250000	86.165	87.74	93.41
300000	88.19	89.855	94.625
350000	90.44	90.98	96.425
400000	92.105	93.23	97.19
450000	93.95	94.94	97.955
500000	95.48	96.695	98.54
Total	**90.074**	**91.546**	**96.026**

Table 2: Performance of the TCM-KNN, DPTCM-KNN and TL-DNN with respect to no. of instances

Which is still 1.87% better than DPTCM-KNN, 3.11% better than TCM-KNN in percentage terms, even though in baseline methods instance count is very low.On Average TL-DNN achieved 96.03%, DPTCM-KNN achieved 91.55% and TCM-KNN achieved 90.07% recall Which is 4.67% better than DPTCM-KNN, 6.20% better than TCM-KNN on average as shown in figure 9.

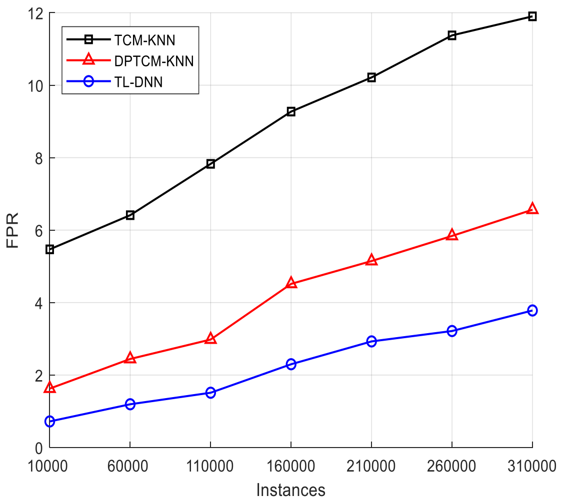

Figure 9: Comparison of TL-DNN, TCM-KNN and DPTCM-KNN in terms of FPR

4.4 False Positive Rate (FPR)

The FPR in DPTCM-KNN algorithm is quite competitive compared to the other algorithms. The False Positives are assigned to those instances where the Algorithmclassifies an incomingpacket as an attack however the packer is actually a normalone. The false positive are false alarms which are fired by the algorithm it must be lower as much as possible. False Positive Rate is calculated using following formula

$$\frac{FP}{FP + TN}$$

Figure 9 shows the comparison of TL-DNN, TCM-KNN and DPTCM-KNN in terms of False Positive rate. At lower instances of 200K the False Positive Rate measures for Anomalous Flow Detection obtained using the TL-DNN, TCM-KNN and DPTCM-KNN for 200K Instances are0.72, 5.47 and 1.63 respectively, which outperformed the baseline methods by 4.75 and 0.91 in absolute terms. Even though the difference looks small which is 86.86% better than TCM-KNN, 55.77% better than DPTCM-KNN in percentage terms (refer table 3).

Instances	TCM-KNN	DPTCM-KNN	TL-DNN
200000	5.4689	1.6253	0.71882
250000	6.4116	2.4429	1.1929
300000	7.8284	2.9795	1.5109
350000	9.2711	4.5159	2.2975
400000	10.214	5.1462	2.9277
450000	11.375	5.839	3.2144
500000	11.9	6.5627	3.7814
Average	**8.924**	**4.159**	**2.235**

Table 3: FPR measures of TL-DNN, TCM-KNN and DPTCM-KNN

At higher 500K, FPR measures for anomalous flow detection obtained using the TL-DNN, TCM-KNN and DPTCM-KNN for 500K Instances are 3.78, 11.90 and 6.56 respectively which is still 68.22% better than TCM-KNN, 42.38% better than DPTCM-KNN in percentage terms. On average TL-DNN achieved 2.23%, TCM-KNN achieved 8.92% and DPTCM-KNN achieved 4.16% FPR which is 74.96% better than TCM-KNN and 46.26% better than DPTCM-KNN.

4.5 Effect of Skewness Transduction

To study the effects of introducing the skewness measure in the transduction process the NSL-KDD dataset is processed two times once with no skewness measure and with skewness measure as shown in table 4 below. The dense-net is then trained over both the dataset for 200,000 to 500,000 instances and effects on TPR are evaluated.

Instances	No Skew	With Skew
200K	88.04	94.0
250K	88.41	93.4
300K	91.63	94.6
350K	90.43	96.4
400K	93.19	97.2
450K	91.96	98.0
500K	93.54	98.5
Average	**91.0**	**96.0**

Table 4: Effect on TPR with skewness no measure and measure.

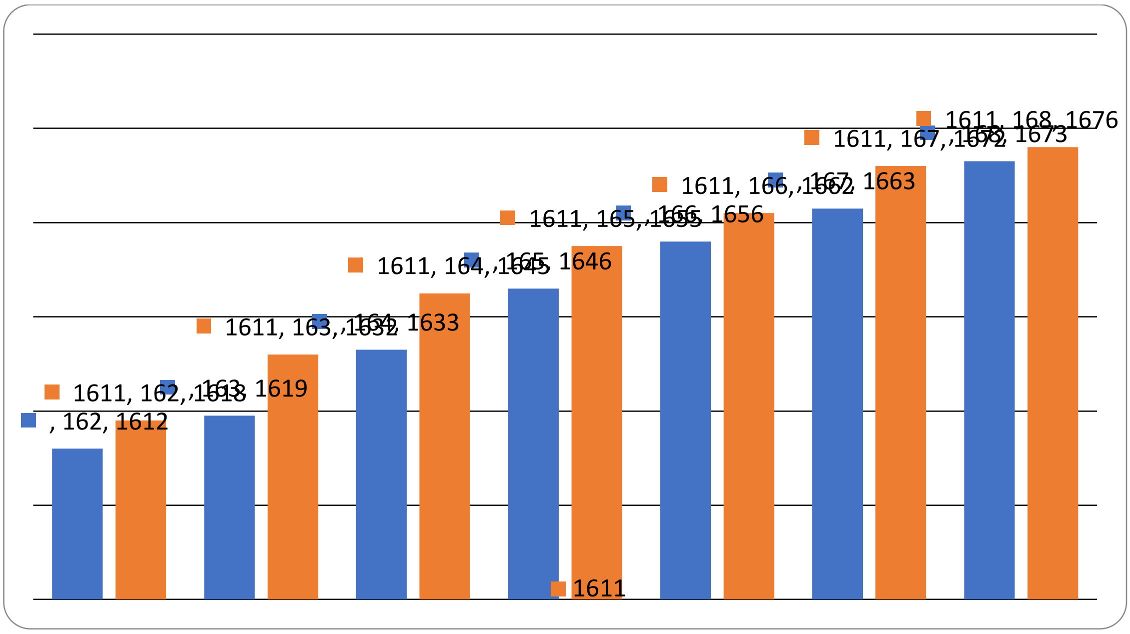

Figure 10: Effect on TPR with skewness no measure and with the skewness measure.

Overall the TPR improves significantly from 200lk-500K we see a positive improvements in the TPR rate as in figure 10. For 200K instances without skew the TPR was 88.04% which shot up to 94% which showed a 5.96% improvement. This improvement was positive for all instances up to 500K i.e. 4.96%, on average the Dense Net showed a TPR of 91% without the skewness measure and 96% with the skewness.

5. Conclusion and Future Scope

One of the biggest challenge SDN face is Security that has been the focus of this paper. The proposed approach presents an Efficient DDoS Detection framework namely TL-DNN, TL-DNN works by combining TCM with Deep Learning (DL) especially designed for SDN architectures. This TL-DNN solves the problems of SDN DDoS attacks regarding

(i) The high-speed network is the limitation of IDS.
(ii) Applicability of ML algorithms, especially DL algorithms over the SDN architectures.
(iii) It also provides a good predictor of multitude attacks with good detection performance.

The experimental outcomes indicate that the framework TL-DNN is successful for high-speed network such as SDN. In comparison to previous techniques, the proposed TL-DNN algorithm gives considerably reduced False Positives and greater Accuracy based on experimental findings on the NSL-KDD dataset. The work also shows that the performance of suggested method is better in terms of FPR, TPR, MSE, and Accuracy through various simulations.

This is the early phase of framework TL-DNN. The scaling of the framework is not performed because of the limitation of the experimental environment. Further, the proposed system in future can be advanced by

- Ability to predict the behavior of even more anomalies like flash worms and crowds attacks in to the framework
- Framework scaling with additional destination ports and for larger attacks.

6. References

[1] Feamster, N., Rexford, J., &Zegura, E. (2014). The road to SDN: an intellectual history of programmable networks. ACM SIGCOMM Computer Communication Review, 44(2), 87-98.

[2] Alsaeedi, M., Mohamad, M. M., & Al-Roubaiey, A. A. (2019). Toward adaptive and scalable OpenFlow-SDN flow control: A survey. IEEE Access, 7, 107346-107379.

[3] Software Defined Network: Definition and Purpose. Study.com.,from https://study.com/academy/lesson/software-defined-network-definition purpose.html
[4] Jain, R. (2013). Introduction to Software Defined Networking (SDN). Washingt. Univ. Saint Louis, 11(7), 1-44.
[5] Shamugam, V., Murray, I., Leong, J. A., & Sidhu, A. S. (2016, March). Software Defined Networking challenges and future direction: A case study of implementing SDN features on OpenStack private cloud. In IOP Conference Series: Materials Science and Engineering (Vol. 121, No. 1, p. 012003). IOP Publishing
[6] Bannour, F., Souihi, S., &Mellouk, A. (2017). Distributed SDN control: Survey, taxonomy, and challenges. IEEE Communications Surveys & Tutorials, 20(1), 333-354.
[7] Sezer, S., Scott-Hayward, S., Chouhan, P. K., Fraser, B., Lake, D., Finnegan, J., ...& Rao, N. (2013). Are we ready for SDN? Implementation challenges for software-defined networks. IEEE Communications magazine, 51(7), 36-43.
[8] Schehlmann, L., Abt, S., &Baier, H. (2014, November). Blessing or curse? Revisiting security aspects of Software-Defined Networking. In 10th International Conference on Network and Service Management (CNSM) and Workshop (pp. 382-387). IEEE.
[9] Khairi, M. H., Ariffin, S. H., Latiff, N. A., Abdullah, A. S., & Hassan, M. K. (2018). A review of anomaly detection techniques and distributed denial of service (DDoS) on software defined network (SDN). Engineering, Technology & Applied Science Research, 8(2), 2724-2730.
[10] Sultana, N., Chilamkurti, N., Peng, W., &Alhadad, R. (2019). Survey on SDN based network intrusion detection system using machine learning approaches. Peer-to-Peer Networking and Applications, 12(2), 493-501.
[11] Erhan, L., Ndubuaku, M., Di Mauro, M., Song, W., Chen, M., Fortino, G., ...&Liotta, A. (2021). Smart anomaly detection in sensor systems: A multi-perspective review. Information Fusion, 67, 64-79.
[12] Bala, R., &Nagpal, R. (2019). A review on kdd cup99 and nslnsl-kdd dataset. International Journal of Advanced Research in Computer Science, 10(2).
[13] Chalapathy, R., & Chawla, S. (2019). Deep learning for anomaly detection: A survey. arXiv preprint arXiv:1901.03407.
[14] Yang, L., Song, Y., Gao, S., Xiao, B., & Hu, A. (2020, December). Griffin: an ensemble of autoencoders for anomaly traffic detection in SDN. In GLOBECOM 2020-2020 IEEE Global Communications Conference (pp. 1-6). IEEE.
[15] Chung, H., Park, J., Keum, J., Ki, H., & Kang, S. (2020). Unsupervised anomaly detection using style distillation. IEEE Access, 8, 221494-221502.
[16] Sampaio, L. S., Faustini, P. H., Silva, A. S., Granville, L. Z., & Schaeffer-Filho, A. (2018, June). Using NFV and reinforcement learning for anomalies detection and mitigation in SDN. In 2018 IEEE Symposium on Computers and Communications (ISCC) (pp. 00432-00437). IEEE.
[17] Chalapathy, R., & Chawla, S. (2019). Deep learning for anomaly detection: A survey. arXiv preprint arXiv:1901.03407.
[18] Pang, G., Shen, C., Cao, L., &Hengel, A. V. D. (2021). Deep learning for anomaly detection: A review. ACM Computing Surveys (CSUR), 54(2), 1-38.
[19] Garg, S., Kaur, K., Kumar, N., & Rodrigues, J. J. (2019). Hybrid deep-learning-based anomaly detection scheme for suspicious flow detection in SDN: A social multimedia perspective. IEEE Transactions on Multimedia, 21(3), 566-578.
[20] Artificial Intelligence vs. Machine Learning vs. Deep Learning. (2021). Retrieved 13 May 2022, from https://towardsdatascience.com/artificial-intelligence-vs-machine-learning-vs-deep-learning-2210ba8cc4ac
[21] Khraisat, A., &Alazab, A. (2021). A critical review of intrusion detection systems in the internet of things: techniques, deployment strategy, validation strategy, attacks, public datasets and challenges. Cybersecurity, 4(1), 1-27.
[22] Chao L, Wen S, Fong C (2015) CANN: An intrusion detection system based on combining cluster centers and nearest neighbors. Knowledge Based Syst 78:13–21
[23] Ahmad, Z., Shahid Khan, A., Nisar, K., Haider, I., Hassan, R., Haque, M. R., ...& Rodrigues, J. J.

(2021). Anomaly detection using deep neural network for IoT architecture. Applied Sciences, 11(15), 7050.

[24] Dey, S. K., & Rahman, M. M. (2019). Effects of machine learning approach in flow-based anomaly detection on software-defined networking. Symmetry, 12(1), 7.

[25] Chen, Y., Pei, J., & Li, D. (2019, May). DETPro: a high-efficiency and low-latency system against DDoS attacks in SDN based on decision tree. In ICC 2019-2019 IEEE International Conference on Communications (ICC) (pp. 1-6). IEEE.

[26] Lee, S., Kim, J., Shin, S., Porras, P., &Yegneswaran, V. (2017, June). Athena: A framework for scalable anomaly detection in software-defined networks. In 2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN) (pp. 249-260). IEEE.

[27] Kokila, R. T., Selvi, S. T., &Govindarajan, K. (2014, December). DDoS detection and analysis in SDN-based environment using support vector machine classifier. In 2014 sixth international conference on advanced computing (ICoAC) (pp. 205-210). IEEE.

[28] Reddy, D. K., Behera, H. S., Nayak, J., Vijayakumar, P., Naik, B., & Singh, P. K. (2021). Deep neural network based anomaly detection in Internet of Things network traffic tracking for the applications of future smart cities. Transactions on Emerging Telecommunications Technologies, 32(7), e4121.

[29] Alcaraz C (2018) Cloud-assisted dynamic resilience for cyber-physical control systems. IEEE WirelCommun 25(1):76–82

[30] Peng, H., Sun, Z., Zhao, X., Tan, S., & Sun, Z. (2018). A detection method for anomaly flow in software defined network. IEEE Access, 6, 27809-27817.

[31] Cui, J., Zhang, J., He, J., Zhong, H., & Lu, Y. (2020, December). DDoS detection and defense mechanism for SDN controllers with K-Means. In 2020 IEEE/ACM 13th International Conference on Utility and Cloud Computing (UCC) (pp. 394-401). IEEE.

[32] Xiao, P., Li, Z., Qi, H., Qu, W., & Yu, H. (2016, August). An efficient DDoS detection with bloom filter in SDN. In 2016 IEEE Trustcom/BigDataSE/ISPA (pp. 1-6). IEEE.

[33] [24] Khraisat, A., &Alazab, A. (2021). A critical review of intrusion detection systems in the internet of things: techniques, deployment strategy, validation strategy, attacks, public datasets and challenges. Cybersecurity, 4(1), 1-27.

[34] Li, Y., & Wu, H. (2012). A clustering method based on K-means algorithm. Physics Procedia, 25, 1104-1109.Xue, H. J., Dai, X., Zhang, J., Huang, S., & Chen, J. (2017, August).

[35] Deepa, V., Sudar, K. M., &Deepalakshmi, P. (2019, March). Design of ensemble learning methods for DDoS detection in SDN environment. In 2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN) (pp. 1-6). IEEE.

[36] Popoola, S. I., Adebisi, B., Ande, R., Hammoudeh, M., Anoh, K., &Atayero, A. A. (2021). smote-drnn: A deep learning algorithm for botnet detection in the internet-of-things networks. Sensors, 21(9), 2985.

[37] Chen, Y. R., Rezapour, A., Tzeng, W. G., & Tsai, S. C. (2020). Rl-routing: An sdn routing algorithm based on deep reinforcement learning. IEEE Transactions on Network Science and Engineering, 7(4), 3185-3199.

[38] Niyaz, Q., Sun, W., &Javaid, A. Y. (2016). A deep learning based DDoS detection system in software-defined networking (SDN). arXiv preprint arXiv:1611.07400.

[39] Tang, T. A., Mhamdi, L., McLernon, D., Zaidi, S. A. R., &Ghogho, M. (2016, October). Deep learning approach for network intrusion detection in software defined networking. In 2016 international conference on wireless networks and mobile communications (WINCOM) (pp. 258-263). IEEE.

[40] Rogers, J. (2011). Transductive confidence machines. Wiley Interdisciplinary Reviews: Computational Statistics, 3(3), 216-220.

[41] Proedrou, K., Nouretdinov, I., Vovk, V., &Gammerman, A. (2002, August). Transductive confidence machines for pattern recognition. In European Conference on Machine Learning (pp. 381-390). Springer, Berlin, Heidelberg.

[42] Peng, H., Sun, Z., Zhao, X., Tan, S., & Sun, Z. (2018). A detection method for anomaly flow in software defined network. IEEE Access, 6, 27809-27817.

[43] Wang, W., Zhang, X., Gombault, S., &Knapskog, S. J. (2009, December). Attribute normalization in network intrusion

detection. In 2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks (pp. 448-453). IEEE.

[44] Iandola, F., Moskewicz, M., Karayev, S., Girshick, R., Darrell, T., &Keutzer, K. (2014). Densenet: Implementing efficient convnet descriptor pyramids. arXiv preprint arXiv:1404.1869.

[45] Zhu, Y., &Newsam, S. (2017, September). Densenet for dense flow. In 2017 IEEE international conference on image processing (ICIP) (pp. 790-794). IEEE.

[46] Brodzicki, A., Jaworek-Korjakowska, J., Kleczek, P., Garland, M., &Bogyo, M. (2020). Pre-trained deep convolutional neural network for clostridioidesdifficile

[47] Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4700-4708).

[48] Tsang, S. H. (2018). Review: Dense Net—Dense Convolutional Network (Image Classification). enlínea].[consulta: 7 abril 2019]. Disponible en: https://towardsdatascience. com/review-densenetimage-classification-b6631a8ef803.

[49] Tavallaee, M., Bagheri, E., Lu, W., &Ghorbani, A. A. (2009, July). A detailed analysis of the KDD CUP 99 data set. In 2009 IEEE symposium on computational intelligence for security and defense applications (pp. 1-6). Ieee.