Improving Intrusion Detection System using Artificial Neural Network

Marwan Ali Albahar[1], Muhammad Binsawad[2], Jameel Almalki[3], Sherif El-etriby[4], Sami Karali[5]
Umm Al Qura University[1,3,5], King Abdulaziz University[2], Menoufia University[4]

***Abstract*—Currently, network communication is more susceptible to different forms of attacks due to its expanded usage, accessibility, and complexity in most areas, consequently imposing greater security risks. One method to halt attacks is to identify different forms of irregularities in the data transmitted and processed during communication. Detection of anomalies is a vital process to secure a system. To this end, machine learning plays a key role in identifying abnormalities and intrusion in communication over a network. The term regularization is one of the major aspects of training machine learning models, in which, it plays a primary role in several successful Artificial neural network models, by inducing regularization in the model training. Then, this technique is integrated with an Artificial Neural Network (ANN) for classifying and detecting irregularities in network communication efficiency. The purpose of regularization is to discourage learning a more flexible or complex model. Thus, the machine learning model generalizes enough to perform accurately on unseen data. For training and testing purposes, NSL-KDD, CIDDS-001 (External and Internal Server Data), and UNSW-NB15 datasets were utilized. Through extensive experiments, the proposed regularizer reaches higher True Positive Rate (TPR) and precision compared L1 and L2 norm regularization algorithms. Thus, it is concluded that the proposed regularizer demonstrates a strong intrusion detection ability.**

***Keywords*—New regularizer; anomaly detection; NSL-KDD dataset; CIDDS-001 dataset; UNSW-NB15**

I. Introduction

Now-a-days, network communication's threats and attacks are growing as it is widely utilized in every field. To prevent such attacks, it is a crucial and necessary task to classify network communication as normal and suspicious. Such task is generally known as anomaly detection, dealing with unlikely events in network communication. The standard approach to detect an anomaly is computing the accurate mathematical model of normal data. Every new receiving instance is compared with the model of normality and, accordingly, an anomaly score is computed. The score will describe the deviations of the new instance compared to the average data instance and, if the deviation is relatively high, then the instance will be considered as suspicious and classified as anomalous and hence processed adequately [1] [2] [3].

In machine learning, generally, we are looking for the best-fitting model among other models in a large solution space. Similarly, in the context of ANN, solution space is defined as the space of all approximated or precise functions that a network can represent.

Network depth and activation functions are used to determine the size of this solution space. One hidden layer with an activation function makes the space of functions very huge, so this space grows exponentially when the depth of the network is increased; hence, finding a most-fit solution becomes a difficult task.

Multiple optimizer functions tend to minimize the loss function, of which Stochastic Gradient Descent (SGD) being very common. Using SGD as an optimizer, one can seek a solution by moving in the opposite direction of the gradient of loss function. Due to complexity and richness of the solution space, this method of learning might overfit the learning model and affects the generalization error or performance significantly on unseen data while giving good results on training data [4]. To solve this issue, the concept of regularization is introduced in machine learning to avoid the complexity of the learning model. There are different regularization algorithms used to avoid overfitting of the machine learning model [4]. For example, in iterative learning, the most common regularization algorithm is early stopping, and, in the neural network, the commonly used regularization algorithm is a dropout. Generally, in statistics and machine learning, the regularization term is used in combination with the loss or error function. This method is beneficial as it incorporates the model complexity into the function to be minimized. Such methods are used in many algorithms such as Support Vector Machines (SVMs) [5] as optimization problems.

However, the existing regularization algorithms come with drawbacks due to the nature of the regularizers. In a challenging setting, where the number of features is greater than the number of samples and correlated, the existing regularization algorithms either do not promote sparsity or poorly perform because of the absence of relevant information.

The purpose of this paper is to implement a new regularization algorithm to search for the optimal solution in a large solution space by taking into consideration the relationship between weight matrix entries. Hence, the limit of space is increased and can be controlled by squeezing and expanding this space based on the penalty term λ. Consequently, it provides the ability to find the least complex learning model. To differentiate between normal and various malicious connections, we plan to examine the algorithm from a multiclass classification perspective.

In this paper, we introduced new regularization design considerations and a general outline of an intrusion detection technique based on using the standard deviation to decay the weight matrices in order to get the regularization term. Compared with well-known regularization techniques, we embedded the proposed regularizer with ANN model for classification tasks and employed NSL-KDD, CIDDS-001 (External and Internal Server Data), and UNSW-NB15 datasets with

separate testing and training sets to evaluate the efficiency in detecting anomalies.

The main contributions of this paper are summarized as follows:

1) We present the design and implementation of an ANN intrusion detection system based on a new regularizer.
2) We study the performance of the model with different regularization parameters impacting accuracy.

The outline of the paper is as follows: We provide the related works in Section II. Then, we give background and formalization in Section III. Next, we present the used datasets in Section IV. In the same section, we also study anomaly detection and the new regularization technique. In Section V, results and discussion are presented. The limitation of proposed regularization technique provided in Section VI. Finally, we conclude our study in Section VII.

II. Related Works

The first IDS or anomaly detection system was introduced by Dr. Dorothy in SRI international, and it is still an actively and heavily researched topic due to its broad applications in network communication ([6],[7]). Supervised learning techniques are popular methods for solving such problems. These techniques give more satisfactory results when statistical and regression techniques are incorporated [21].

A novel intrusion system and a multilevel hybrid classifier were proposed in [8]. The proposed system is combined the unsupervised Bayesian clustering with the supervised tree classifiers to detect the intrusions. Based on the Modular Multiple Classifier System (MCS), the authors in [9] proposed an unlabeled network anomaly IDS, where every module was created to model network services or a specific group of similar protocols. Moreover, they conducted experimental studies on the KDD Cup 1999 dataset, which revealed that the proposed anomaly IDS was able to accomplish high attack detection along with the low rate of false alarm. In [10], authors developed an intrusion detection system based on the AdaBoost algorithm. Within this algorithm, the decision rules were provided for the continuous and categorical features and the decision stumps were used as weak classifiers. The combination of the weak classifiers for the continuous and categorical features with the strong classifier allowed handling the relation between these features without the need for any forced conversations. According to the authors' experimental analysis, they reported that the algorithm had low error rates and computational complexity. Data mining techniques were utilized in [11]; the authors devised a novel framework for intrusion detection accordingly. For building classifiers, the authors proposed a classification algorithm which uses fuzzy association rules. However, the outcomes regarding the unseen attacks were not promising. In [12], the authors used a supervised learning classifier system for intrusion detection. To learn signatures for network intrusion detection, they presented a biologically inspired computational approach which can learn adaptively and dynamically. For the futuristic establishment of the intrusion detection system, authors in [13] presented a reference for the comparison of the efficiency of different machine learning techniques, including SVM and the tree classification. Moreover, the authors proposed a method to compute the mean value through sampling different ratios within the normal data for every measurement, resulting in obtaining a better rate of accuracy when observing the data in the real world. A novel machine-learning algorithm was proposed in [14], namely, Boosted Subspace Probabilistic Neural Network (BSPNN), which combined a semiparametric and an adaptive boosting approach to attain better trade-off between the generality and the accuracy. Hence, the method depicted prominent improvements with respect to detection accuracy, comparatively low computational complexity, and negligible false alarms. A new approach for intrusion detection was proposed in [15]. This approach is based on ANN and fuzzy clustering (FC-ANN). To evaluate the proposal, the authors conducted an experiment using the KDD Cup 1999 dataset. Experimental results demonstrated that FC-ANN enhanced the detection stability and the detection precision. For the prediction of the anomaly detection, a random-effects logistic regression model was proposed in [16].

Imbalanced class distribution is an inevitable problem in real network traffic due to the large size of traffic and low frequency of certain types of anomalies. Authors in [17] used sampling approaches to combat imbalanced class distributions for network intrusion detection. It performed flow-based classification on a network flow dataset: CIDDS-001. The system was able to detect attacks with up to 99.99% accuracy.

In [18], the statistical and complexity analysis of CIDDS-001 dataset is considered. The authors utilized the k-nearest neighbor classifier on CIDDS-001 to build an IDS. Their system achieved an overall accuracy of 99.6% with 2nn and a minimum accuracy of 99.3% with 5nn. Using the same dataset, the authors in [19] conducted an analytical study to assess the performance of KNN and k-means clustering algorithms when classifying traffic. Both algorithms achieved over 99% accuracy. In [20], authors proposed an effective anomaly-based intrusion detection system using a gradient boosted machine (GBM). Three different datasets, NSL-KDD, UNSW-NB15, and GPRS dataset, were utilized with either tenfold cross-validation or hold-out method. In [21], the authors proposed an improved IDS based on hybrid feature selection and two-level classifier ensembles. Two intrusion datasets (NSL-KDD and UNSW-NB15) have been employed to evaluate the performance. Based on the statistics and significance tests, on the NSL-KDD dataset, the proposed classifier shows 85.8% accuracy, 86.8% sensitivity, and 88.0% detection rate. By taking advantage of the multiple classification abilities of neural networks and the fuzzy logic, authors in [22] developed a novel model for the intrusion detection system. A new learning algorithm was proposed in [23] for adaptive intrusion detection using naïve Bayesian and boosting classifiers. Additionally, they conducted an experiment using the KDD Cup 1999 dataset. The experiment proved that the proposed algorithm offered higher detection rates with a remarkable reduction in the number of false positives for multiple types of network intrusion. A GA combined with the KNN for feature weighting and selection was proposed in [24]. The proposed model was applied on the KDD Cup 1999 dataset for identifying DDoS/DoS attacks. The result showed that the accuracy for unknown attacks was found to be 78%, whereas the accuracy for known attacks was calculated to be 97.24%. Based on the Pittsburgh, iterative rule learning (IRL), and Michigan approaches, the authors in [25] proposed three

different types of genetic fuzzy systems for intrusion detection. A novel feature representation approach was proposed in [26]. This approach is called the cluster center and nearest-neighbor approach (CANN), in which the distance between data and its nearest neighbor and data sample and its cluster center were measured and summed. The authors conducted the experiments using the KDD Cup 1999 dataset, showing that the CANN classifier performed similarly or slightly better than SVM and k-NN.

Two dimensionality reduction techniques, namely, PCA and fuzzy PCZ, were used and compared in [27], where the authors classified the test samples of connections into attack or normal category by applying KNN algorithm. In addition, they conducted experiments using KDD Cup 1999 dataset. The results showed that fuzzy PCA performed better than the PCA in detecting the DoS and U2R attacks. In [28], the authors proposed a deep learning approach using recurrent neural networks (RNN-IDS). The experimental results demonstrated that the RNN-IDS was ideal for modeling a classification model with relatively high accuracy, and its performance was also superior compared to conventional machine learning classification techniques in multiclass and binary classification. The authors in [29] built an anomaly detection system using backpropagation algorithm optimized by Conjugate Gradient (CG) algorithm. Then, they analyzed the use of CG optimization (Polak-Ribiere, Fletcher Reeves, Powell Beale). Based on their experiment results, the average accuracy was 93.2% for two classes "intrusion" and "normal". Applications of LSTM to RNN for modeling the IDS modeling were proposed in [30]. The ideology of the experiment was dependent on the hyperparameter values, the rate of learning, and changes in the performance; the size of the hidden layer had a significant impact on the performance. According to their experiments, the average rate of detection was computed to be 98.8%. The authors in [31] proposed a learning model, namely, PSO-FLN for fast learning network (FLN), based on particle swarm optimization (PSO). A deep learning model was proposed in [32]. The model is based on the DBN and stacked nonsymmetric deep autoencoder (NDAE). They used KDD Cup 1999 and NSL-KDD datasets to evaluate their model, which accurately detected the Probe attacks and the DoS. Nevertheless, R2L attacks were barely identified, while no detection of the U2R attacks was recorded. The precision value was found to be 99.99%, with an overall accuracy of 97.85%.

III. Background and Formalization

The most critical issue in machine learning is developing a generalized training model that will perform accurately on training data and at least will provide almost the same results on unseen data.

There are many algorithms used whose primary goal is to decrease classification error on unseen data at the cost of increased training error. In other words, we may say that reducing the model's generalization error without any effect on training error is known as regularization.

Many techniques have been used to improve the generalization performance of the learning model [33]. Some add constraints to the machine learning model, such as putting constraint on model parameters values, and others add further statistical terms in the objective function that are known as a soft constraint on model parameters [34].

Developing a more effective regularization algorithm is a crucial task in the field of machine learning; hence, it is the main focus of research in this field. In a statistical model of learning algorithms, such constraints and penalties are used to encode prior knowledge. On occasion, these penalties and constraints are designed to promote generalization by expressing generic preferences for a simple classification model. However, it is necessary to incorporate such penalties and constraints to make an undetermined problem determined.

As explained above, there are multiple strategies to incorporate regularization in machine learning algorithms [35], [36], [37]. Among these methods, L1- and L2-norm are the most common regularization methods. L2 regularization is also referred to as *Tikhonov regularization*, and, in statistics, as ridge regression. It is combined with the cost function as a complexity term.

L2 regularization is the squared Euclidean of all feature weights of the hidden layer, and, in the case of multiple hidden layers, it is the sum of all such squared norms including the output layer of the neural network [38].

Another regularization parameter,λ, is multiplied with regularization in order to put a penalty on and control the strength of the magnitude of weights. Due to this regularization, the model results in much smaller weights for each layer. Similarly, L1 regularization produces many zeros in the weight matrix and makes it sparse, hence, controlling the complexity of the model. Both L1 and L2 regularizations have a well-defined probabilistic interpretation which is similar to adding a Gaussian prior over the distribution of weight matrix W in case of L2 and Laplacian in case of L1 [39]. However, several tried and tested regularization methods exist for both neural networks and other machine learning algorithms (Random forests, SVM, etc.) [5], [37], [40],[41]. For succinctness, we will focus purely on methods used on ANNs. For simplicity, we can split the methods into categories, with one being sparsity-based regularization and the other not.

The sparsity-based methods to be considered are L1 and L2 norms.

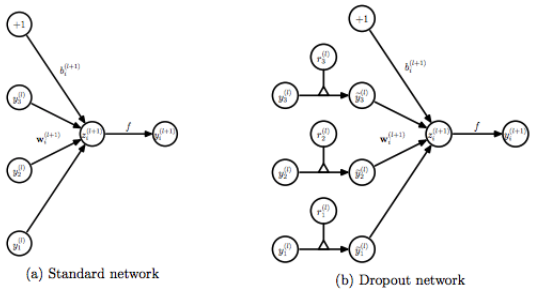
Both methods take a sum over the absolute value and square, respectively.

There is a great amount of previous work comparing L1 and L2 along with other regularization methods in a variety of problem domains [42].

On the other hand, we can also apply methods such as early stopping. This would reduce the number of parameters the network learns; thus, this is considered a form of regularization. The goal of early stopping along with other forms of regularization is to reduce generalization error or increase generalization accuracy while allowing training error to increase.

The most seminal regularization method and one of the more significant breakthroughs in machine learning is dropout [43].

Dropout is an intuitively brilliant discovery that *drops out* or deactivates and removes a portion of neurons randomly according to an arbitrary value. Pushing a neural network to acquire more stable and strong characteristics together with various random subsets of other neurons. Depending on the problem's context, it can be used in combination with sparsity regularizers to good effect (see Fig. 1).



(a) Standard network (b) Dropout network

Fig. 1. Comparison between a standard neural network and a network implementing dropout

IV. Anomaly Detection and New Regularization Technique

A. Anomaly Detection

Anomaly detection can be framed in many ways. Outlier detection, for instance, can often fall under this umbrella. Here, let us define an anomaly as something that significantly differs from the rest of the data or otherwise grossly misfits the distribution of data. We trained and tested in a supervised context (classification) using a feedforward network to test differences across other regularization techniques and our new regularization in our problem domain.

B. DATA

Due to the nature of the problem, the following 3 datasets were chosen to carry out analysis based on the proposed regularization, L1 and L2 norm regularizations. The competition task was to build a network intrusion detector to analyze the performance of the proposed regularizer with existing regularizers and a predictive model capable of distinguishing between normal connections and attack connections.

1) NSL-KDD Dataset.
2) UNSW-NB15 Dataset
3) CIDDS-001 Dataset

1) NSL-KDD dataset: NSL-KDD dataset is a replacement of KDD-CUP dataset and it solves some problems in the KDD CUP 1999 dataset. In NSL-KDD dataset there are 4 attack categories that represent anomalous data and 1 normal category which shows that the corresponding instances are normal. The dataset is quite imbalance and due to this nature, training a classifier is a challenging task. Various types of attacks categories are shown in Table I.

TABLE I. Attack classes based on different attack types

Attack Class	Training Set	Testing Set
DOS	back, land, neptune, pod, smurf, teardrop	back, land, neptune, pod, smurf, teardrop, mailbomb, processtable, udpstorm, apache2, worm
Probe	ipsweep, nmap, portsweep, satan	ipsweep, nmap, portsweep, satan, mscan, saint
U2R	buffer-overflow, loadmodule, perl, rootkit	buffer-overflow, loadmodule, perl, rootkit, sqlattack, xterm, ps
R2L	fpt-write, guess-passwd, imap, multihop, phf, spy, warezclient, warezmaster	fpt-write, guess-passwd, imap, multihop, phf, spy, warezmaster, xlock, xsnoop, snmpguess, snmpgetattack, httptunnel, sendmail, named

2) UNSW-NB15 dataset: UNSW-NB15 dataset is created by IXIA PerfectStorm tool in the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) and its purpose is generating a hybrid of real modern normal activities and synthetic contemporary attack behaviors. It contains nine different types of attacks Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. The whole dataset contains 2,540,044 records and it is available to download as one file or split into several different CSV files. There is also one list of event files which contains information about the number of events categorized by attack category and attack subcategory for all 2.5M records. From that dataset, the training and the test dataset are produced, wherein the training dataset contains175,341 records and 82,332 records in the test dataset [44].

3) CIDDS-001 Dataset: Another dataset we used for our experiment is the CIDDS-001 dataset [45]. This is a labeled flow-based dataset used for intrusion detection system. The following attributes from the dataset are used for training the model: *Src IP, Src Port, Dest IP, Dest Port, Proto, Duration, Bytes, Packets, Flags*.

There are two types of server through which this data is collected (o*pen stack and external server)*. Data from both servers contain the aforementioned attributes, the only difference is in the attack categories.

Data from o*pen stack server* contains the following three categories:

n*ormal, victim and attacker*. While data from the *external server* contains the following 5 categories: n*ormal, victim, attacker, unknown, suspicious*.

C. New Regularization Technique

In the machine learning field, the commonly applied regularization techniques are L1-norm and L2-norm. During optimization, these regularizers consider the complexity of weights to induce the networks towards a more general mapping. L1-norm imposes the sum of the absolute values as a penalty, while L2-norm imposes the sum of the squared values as a penalty. The purpose of this article is to introduce a new regularization that employs the standard deviation of the weight matrix and then multiplies it by λ to make the regularization term. Consequently, the regularizer computes the weights standard deviation of the weights to the loss function.

After studying the L1 and L2 regularizers, we found one

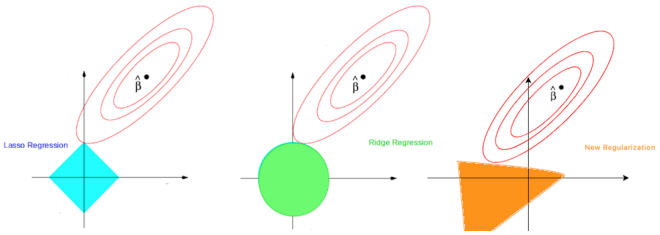


Fig. 2. Contours of L1, L2, and new regularizers

significant drawback, that is regulating the weights' individual values without taking into consideration the relationship between weight matrix entries. To resolve this downside, the new regularization technique utilizes the standard deviation to get the regularization term. This is to construct an adaptive form of weight decay. Thus, the regularizer does not allow the learning model to adapt widespread values from weight space.

The contour of the new regularizer was displayed highlighting the efficacy and potency of the new regularizer. 2 represents the feasible region of L1, L2, and new regularization techniques. The contours of each regularizer represent different loss values. The behavior of the L2-norm is circular and incorporates L1, while the new regularization acts like a parabola and takes values beyond the L2-norm limit. This helps in a sense, it increases the limit of values (space) to be adopted, and based on the penalty term λ this space can be expanded. The formalization as follows (See equations 1), with ω denoting the standard deviation of weight matrix w_i.

$$\lambda \sum_{i=1}^{k} \omega \tag{1}$$

During the training process, λ denotes the regularization parameter that sets a penalty to restrict weights from selecting high values. In other words, the loss function in our case will become (see equations 2):

$$min_w\{f(X, y : w) + \lambda\sigma(w)\} \tag{2}$$

Therefore, if the weight values of all layers are large, the weight values of the selected λ will be large. Thus, the weight values cannot be equal, as they will have more freedom to search in a large space. Consequently, our regularization technique is more effective compared to the L1 and L2 regularization techniques.

The model was trained using the ***Nesterov ADAM*** optimizer, with $tanh$ activation functions. The model was trained over 100 epochs with a batch size of 32. The labeled data were classified with a feedforward network.

D. Artificial Neural Network (ANN) Based IDS with New Regularization Method

In this section, we present the diagrammatic representation of data preprocessing and training ANN model which employed our new regularizer as shown in Fig. 3. There are generally four steps involved in this process (Fig. 2) as follows: There are generally four steps involved in this process (Fig. 3) as explained below.

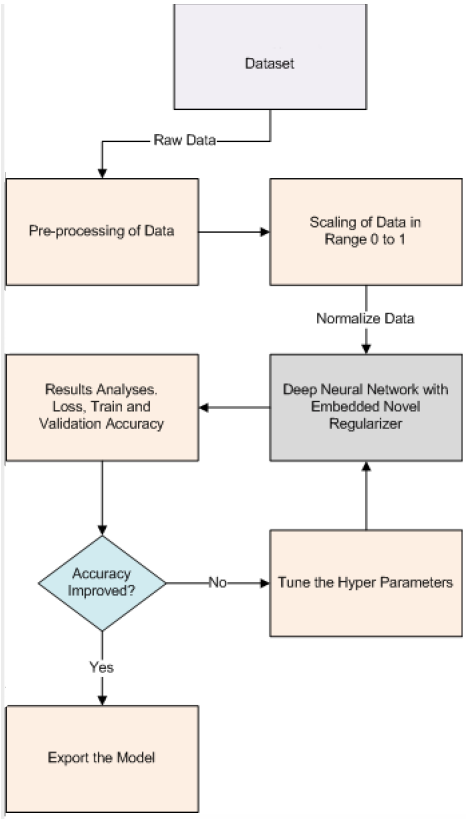


Fig. 3. New regularizer based IDS workflow

1) Data Preprocessing: Artificial Neural Network uses only numerical data for training and testing. So, the initial step is to transform nominal and textual data into numerical data. To do this, the following steps were performed:

- All the nominal and textual attributes were converted by using one-hot encoding (nominal to binary conversion in Weka). Conversion of attributes to one-hot encoding leads to increasing of attributes in attributes. Therefore, the number of units in ANN is adjusted according to attributes.
- Each category of attack types was converted by one-hot encoding.

2) Data Scaling: After data preprocessing, each dataset contains attributes of numerical values and one-hot encoded values. The numerical values were normalized according to the formulation given in equation 3.

$$\bar{X}_i = \frac{X_i - min(X_i)}{max(X_i) - min(X_i)} \tag{3}$$

For $i = 1, ..., n$ where n represents the number of records, and x represents a specific column in the dataset. Next, duplicate records were removed from the dataset to restrict classifiers from giving biased results.

3) Training the ANN model: After data preprocessing and data scaling phases, our next task is to implement ANN model. Python was picked to be the implementation language and the Keras framework was employed for ANN. The ANN model

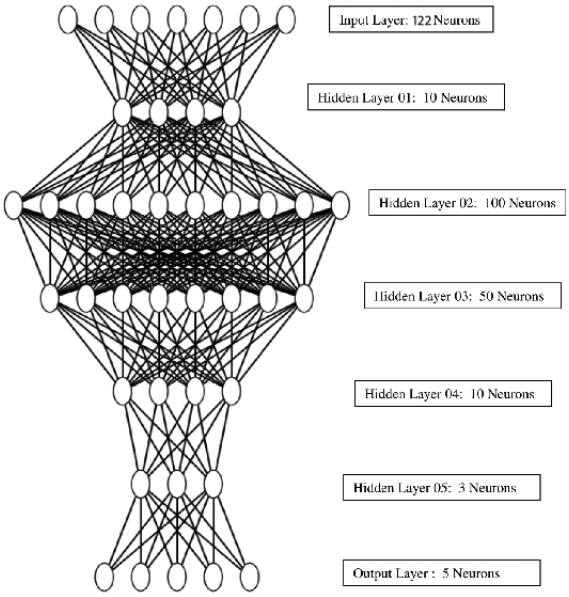


Fig. 4. ANN architecture with embedded new regularizer

is incorporated with a new regularizer to test our method. Employing the mathematical description in equation 1, the proposed regularizer is applied as a function. In fact, the kernel_regularizer was assigned with this new function rather than built-in regularizers in the ANN model. The ANN predictive model includes two hidden five- and three-unit layers, respectively. The last layer is composed of two units according to class values. $tanh$ is the activation function employed in each layer, except for the last layer, where the $softmax$ activation function is utilized. In the first two layers, the weight matrix was initialized with Gaussian random distribution. Due to the large number of neurons in each layer, we show only the reduced version of the model as depicted in Fig. 4. The first layer consists of 122 neurons, the first hidden layer 10 neurons, and the second layer 100 neurons.

Likewise, the third hidden layer has 50 neurons. In the fourth hidden layer, the input size is reduced to 10. Hence, we added 10 neurons to it. Further, these neurons are connected to 3 neurons in the fifth hidden layer which is further connected to 5 output neurons each for one of the five specific categories. In each layer, a Kernel matrix is initialized with uniform distribution and $tanh$ activation function except the last layer which has $softmax$ activation function. Further, for binary classification, the last layer has 2 output neurons. Finally, the model is compiled with an $adam$ optimizer with a default learning rate and other parameters.

4) HyperParameters Adjustment: After each 100-epoch run of the ANN model, precision and loss values were evaluated and the hyperparameters were adjusted accordingly. Activation functions and kernel initializer distributions were determined after several iterations and examining the depth of the ANN model. According to our optimal desired outputs, regularization parameter λ was also adjusted. The number of layers and hyperparameters remained the same for each regularizer. λ parameter was constantly updated and fixed to the value resulting in the highest and best accuracy for the corresponding regularizer.

V. Results and Discussion

To produce results based on the proposed method, we implemented our model for multiclass classification *(normal and four different attack categories (5-class))*. In addition, we applied the 10-fold cross-validation on each dataset. All simulations were carried out on a server having 32 GB RAM, GeForce GTX 1080 GPU of 8 GB GDDR5X memory, and 2560 NVIDIA CUDA cores. We compared results for multiclass problems in each case and demonstrated our results. For each dataset, the corresponding attack categories were considered as classes and the ANN with new, L1, and L2 regularizations is trained by using 10-fold cross-validation. For every attack class in each dataset, the performance measures described in equations 5–9 were computed and presented. In the following sections, results for each dataset based on our new regularization are compared with other regularization algorithms. In each type of classification, the proposed regularization demonstrates a good performance and is superior to L1 and L2 regularizations. Furthermore, other hyperparameters and results on each classification category are discussed in detail.

A. Evaluation Protocols

For multiclass classification, the loss function used is categorical cross entropy as given in equation 4.

$$CrossEntropy = -\sum_{i}^{C} t_i \log f(s_i) \tag{4}$$

where C is the number of classes, t_i is the ith class and $f(s_i)$is the ith output after activation function f.
To evaluate our model, training and validation accuracy are reported for the data partitions as explained in Results and Discussion. Accuracy is calculated based on the following mathematical representation. Apart from accuracy, other performance measures, that is, TPR also known as Recall, False Positive Rate (FPR), Precision (Pre), and F1 measures, are calculated based on equations 5, 6, 7, 8, and 9, respectively.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5}$$

$$TPR = \frac{TP}{TP + FN} \tag{6}$$

$$FPR = \frac{FP}{FP + TN} \tag{7}$$

$$Pre = \frac{TP}{TP + FP} \tag{8}$$

$$F1 = 2 * \frac{TPR \times Pre}{TPR + Pre} \tag{9}$$

where TP, TN, FP and FN denote true positives, true negatives, false positives, and false negatives, respectively.

B. Performance Measurement of Different Regularization Techniques

1) NSL-KDD dataset: ANN model with embedded new regularization is trained on 25192 samples (**The 20% of NSL-KDD training set (KDDTrain+)** by using 10-fold cross-validation. The trained model is then tested on a separate test dataset containing 22544 samples (**KDDTest+**). After that the results were recorded in Tables II, III, and IV. NSL-KDD dataset is an imbalanced dataset; therefore, the individual performance measures for each class are significantly affected. For example, R2L attack type has a total of 224 samples and the performance is lower for this category type. In such a situation, the classifier is biased towards more frequent samples, for example, a normal category having 9711 samples.

New regularization: NSL-KDD dataset has four different attack categories. For each attack category, different performance measures were computed (see Table II). Experimental results for TPR are also demonstrated in Fig. 5.

TABLE II. PERFORMANCE MEASURES FOR NSL-KDD DATASET BY USING NEW REGULARIZATION

Labels	TPR	FPR	Pr.	F1	Acc
Normal	0.980	0.0073	0.990	0.985	
DoS	0.978	0.0054	0.989	0.983	
R2L	0.924	0.0087	0.520	0.665	98.5%
U2R	0.969	0.0033	0.977	0.973	
Probe	0.956	0.0067	0.941	0.948	

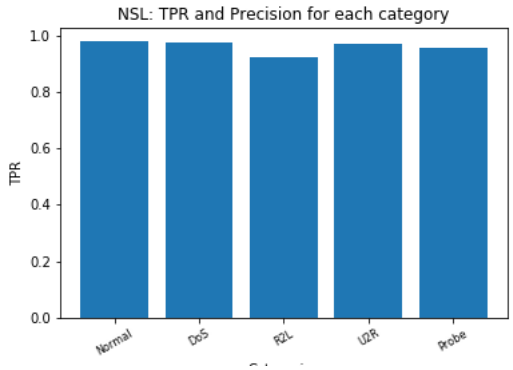


Fig. 5. TPR observerd using the new regularizer for NSL-KDD dataset

L1-Norm regularization: For the sake of comparison, we also used L1-norm regularization for the 5 classes of NSL-KDD dataset. Finally, we computed the performance measures results (see Table III).

TABLE III. PERFORMANCE MEASURES FOR NSL-KDD DATSET USING L1-NORM REGULARIZATION

Labels	TPR	FPR	Pr.	F1	Acc
Normal	0.975	0.011	0.986	0.981	
DoS	0.96	0.013	0.974	0.967	
R2L	0.938	0.01	0.502	0.654	95.4%
U2R	0.948	0.01	0.937	0.942	
Probe	0.943	0.006	0.95	0.947	

L2-norm regularization: Further, we trained the classifier by using L2-norm regularization. The result of NSL-KDD datasets is shown in Table IV.

TABLE IV. PERFORMANCE MEASURES FOR NSL-KDD DATASET USING L2-NORM REGULARIZATION

Labels	TPR	FPR	Pr.	F1	Acc
Normal	0.978	0.011	0.986	0.982	
DoS	0.967	0.007	0.985	0.976	
R2L	0.929	0.011	0.472	0.626	97.2%
U2R	0.956	0.004	0.975	0.966	
Probe	0.948	0.009	0.927	0.938	

Based on our analysis of the above results, we observed that, in terms of average TPR and FPR, our proposed technique outperformed the L1 and L2 regularizations. The average TPR for the proposed regularizer is 96.2%, while the L1 and L2 regularizations' average TPR was 95.27% and 95.56%, respectively. Similarly, the average FPR is lower using the proposed regularizer, being 0.63%. For L1 and L2 regularizations, the average FPR is 0.97% and 0.8%, respectively. Regarding the training time of each regularizer, our proposed regularization took 177.3 seconds, whereas L1 and L2 regularizers took almost 176.5 seconds. Obviously, the training time for all models is almost the same. While in testing, the parameters are kept static (as we do not change them during testing); thus, the role of tuning the regularization has vanished during testing. Consequently, the testing time was less than training. For the NSL-KDD dataset, the testing time for all models was 46.02 seconds. Hence, we undoubtedly can state that our proposed regularizer performed better than other regularizers on the NSL-KDD dataset.

2) UNSW-NB 15 dataset: In addition, we provided the comparison of different performance measures for the UNSW-NB15 dataset using new, L1-norm, and L2-norm regularizers. We tested these different regularizations using 10 classes of the UNSW- NB15 dataset on 175,341 samples given in an explicit training set (NSW_NB15_Train). Similarly, the models are tested on 82,332 samples (UNSW_NB15_Test), and then we computed the results (see Tables V, VI, and VII).

New regularizaton : We embedded the proposed regularization with ANN model and then tested it using 10 different categories of the UNSW-NB15 dataset as shown in Table V. TPR results can also be viewed from Fig. 6.

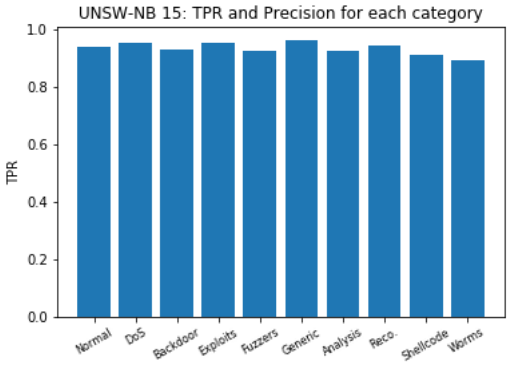


Fig. 6. TPR observed using proposed regularizer on UNSW-NB 15 dataset

L1-norm regularization: Table VI represents our simulation results using L1-norm regularization *(using the same ANN model having an equal number of layers and units)*.

TABLE V. NEW REGULARIZATION RESULTS ON UNSW-NB 15 DATASET

Labels	TPR	FPR	Pr.	F1	Acc
Normal	0.941	0.009	0.98	0.96	
DoS	0.952	0.01	0.878	0.914	
Backdoor	0.933	0.006	0.641	0.76	
Exploits	0.956	0.007	0.972	0.964	
Fuzzers	0.924	0.006	0.95	0.937	94.58%
Generic	0.964	0.034	0.966	0.965	
Analysis	0.927	0.006	0.635	0.753	
Reconnaissance	0.944	0.005	0.92	0.932	
Shellcode	0.911	0.002	0.72	0.804	
Worms	0.892	0.001	0.504	0.644	

TABLE VI. L1-NORM RESULTS ON UNSW-NB 15 DATASET

Labels	TPR	FPR	Pr.	F1	Acc
Normal	0.937	0.015	0.969	0.953	
DoS	0.942	0.013	0.854	0.896	
Backdoor	0.919	0.007	0.586	0.716	
Exploits	0.932	0.009	0.964	0.948	92.4%
Fuzzers	0.908	0.009	0.928	0.918	
Generic	0.949	0.039	0.961	0.955	
Analysis	0.925	0.008	0.584	0.716	
Reconnaissance	0.925	0.005	0.926	0.925	
Shellcode	0.902	0.003	0.7	0.789	
Worms	0.862	0.001	0.407	0.553	

L2-norm regularization: We embedded L2-norm regularization with ANN model having an equal number of layers and units *(as that used for the proposed regularization)*. The results are shown in Table VII. Based on the analysis of our results,

TABLE VII. L2-NORM REGULARIZATION RESULTS ON UNSW-NB 15 DATASET

Labels	TPR	FPR	Pr.	F1	Acc
Normal	0.94	0.011	0.977	0.958	
DoS	0.948	0.011	0.873	0.909	
Backdoor	0.925	0.006	0.62	0.743	
Exploits	0.943	0.008	0.969	0.956	94.3%
Fuzzers	0.918	0.007	0.938	0.928	
Generic	0.957	0.036	0.964	0.961	
Analysis	0.923	0.008	0.599	0.726	
Reconnaissance	0.93	0.006	0.915	0.922	
Shellcode	0.91	0.003	0.689	0.784	
Worms	0.877	0.001	0.427	0.574	

the average TPR computed for this dataset using the proposed regularization is 93.43%. However, for L1- and L2-norm regularizations, the average TPR is 92% and 92.7%. Here, again our model outperformed the existing regularizations in terms of TPR. Similarly, our proposed regularization surpassed the existing regularizations in terms of FPR. The average FPR achieved using the proposed regularizer is 0.86%, whereas the average TPR for L1 is 1.06% and for L2 is 0.96%. Regarding the training time, the proposed regularization took 425.9 seconds, while L1 and L2 took 424.7 seconds *(which is an acceptable difference)*. Noteworthy, the testing time was much less than training *(the testing time was 126.2 seconds)*.

3) CIDDS-001 dataset: Here, we carried out several experiments on the CIDDS-001 dataset using different regularizations. CIDDS-001 dataset has two parts:

1) External Server Dataset
The model is trained over all the dataset provided using 10-fold cross-validation, except for a set of 339030 samples which were kept separate for testing purposes. Apart from normal samples, there are 4 different attack categories in this dataset which are $victim$, $attacker$, $unknown$, and $suspicious$.

2) Open Stack dataset
The same approach is applied to this dataset. The ANN model is trained over all dataset using the 10-fold cross-validation, except for a set of 2789002 samples. There are only 2 attack categories which are $victim$ and $attacker$.

We carried out our experiments on both datasets and the following performance measures were observed for each regularization.

New regularization: The experimental results on the two types of dataset using the new regularization are given in Tables VIII and IX. TPR results for each category are also shown in Fig. 7 and 8, respectively.

TABLE VIII. SIMULATION RESULTS ON CIDDS-001 OPENSTACK DATASET USING THE PROPOSED REGULARIZATION

Labels	TPR	FPR	Pr.	F1	Acc
Normal	0.988	0.019	0.996	0.992	
Victim	0.979	0.007	0.928	0.953	97.87%
attacker	0.969	0.005	0.945	0.957	

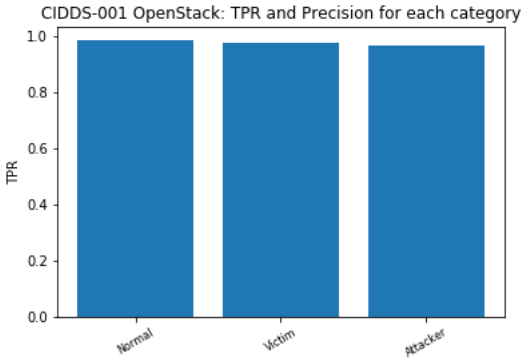


Fig. 7. TPR observed for OpenStack server dataset using the new regularization

L1-norm regularization: Results obtained using L1-norm regularization for each of the two types of dataset are shown in Tables X and XI.

L2-norm regularization: Tables XII and XIII demonstrated the results using L2-norm regularization for each of the two types of the dataset.

TABLE IX. SIMULATION RESULTS ON CIDDS-001 EXTERNAL SERVER DATASET USING THE PROPOSED REGULARIZATION

Labels	TPR	FPR	Pr.	F1	Acc
Normal	0.979	0.006	0.969	0.974	
Victim	0.94	0.002	0.901	0.92	
attacker	0.957	0.002	0.941	0.949	96.8%
unknown	0.978	0.006	0.968	0.973	
suspicious	0.986	0.011	0.993	0.99	

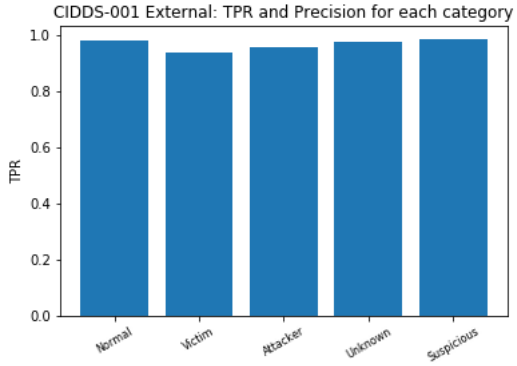


Fig. 8. TPR observed for External server dataset using the new regularization

TABLE X. SIMULATION RESULTS FOR CIDDS-001 OPEN STACK DATASET USING L1-NORM REGULARIZATION

Labels	TPR	FPR	Pr.	F1	Acc
Normal	0.969	0.025	0.995	0.982	
Victim	0.965	0.016	0.848	0.903	96.6%
attacker	0.95	0.016	0.852	0.898	

TABLE XI. SIMULATION RESULTS ON CIDDS-001 EXTERNAL SERVER DATASET USING L1-NORM REGULARIZATION

Labels	TPR	FPR	Pr.	F1	Acc
Normal	0.965	0.008	0.962	0.964	
Victim	0.928	0.009	0.674	0.781	
attacker	0.937	0.007	0.8	0.863	95.3%
unknown	0.964	0.01	0.949	0.956	
suspicious	0.968	0.014	0.992	0.98	

TABLE XII. SIMULATION RESULTS FOR CIDDS-001 OPEN STACK DATASET USING L2-NORM REGULARIZATION

Labels	TPR	FPR	Pr.	F1	Acc
Normal	0.981	0.019	0.996	0.988	
Victim	0.979	0.009	0.91	0.943	96.0%
attacker	0.968	0.01	0.903	0.935	

TABLE XIII. SIMULATION RESULTS FOR CIDDS-001 EXTERNAL SERVER DATASET USING L2-NORM REGULARIZATION

Labels	TPR	FPR	Pr.	F1	Acc
Normal	0.972	0.011	0.947	0.959	
Victim	0.935	0.003	0.868	0.9	
attacker	0.947	0.005	0.856	0.899	96.4%
unknown	0.969	0.009	0.953	0.961	
suspicious	0.974	0.013	0.992	0.983	

- External Server Dataset
  Average TPR of 96.8%, 95.24%, and 95.93% was observed for the external server dataset. Similarly, the average FPR computed is 0.55%, 0.95%, and 0.82% for the proposed, L1, and L2 regularizations, respectively. The training time was 1028, 1019, and 1022 seconds for the proposed, L1, and L2 regularizations. In this case, the testing time was the same for all regularizations which was 76.6 seconds. Based on the analysis of the above results, it can be concluded that the proposed regularization outperformed the other regularizations.
- OpenStack Server Dataset
  For this dataset, the average TPR computed was 97.86%, while the FPR was 1.03%. As for the L1 and L2 regularizers, the average TPR is 96.13% and 97.6%. Similarly, the average FPR for L1 and L2 is 1.9% and 1.26%, respectively. As far as the training time is concerned, the training time took 1728.4, 1720.0, and 1723.2 seconds for the proposed, L1, and L2 regularizations, respectively. In terms of testing, the time taken was 97.7, 100.2, and 92.8 seconds, respectively. Hence, from the results above, we can conclude that the performance of the proposed regularization is slightly higher than other regularizers.

VI. LIMITATION OF NEW REGULARIZATION

Several researchers employed multiple regularization algorithms, the most common ones being lasso regularizations and ridge regression. However, some disadvantages are inherent in the regularization framework. For example, In a challenging setting, where the number of instances is very low and the dimensionality is very high, it is impractical to utilize these regularizations. Likewise, our regularization algorithm had multiple limitations, as follows:

- It cannot be employed for selecting or reducing features.
- It is challenging to choose a suitable value of λ, due to the fact that it is a continuous value. In addition, the process of picking a suitable value from multiple attempts will be computationally costly and time consuming.

VII. CONCLUSION

The field of ANN regularizers is one that is still ripe for new research and innovation. From attempts in adaptive weight decay to new techniques altogether, many innovations in improving generalization through reducing model complexity are possible. In this paper, we proposed a new regularization technique for anomaly detection based on the standard deviation of the weight matrix. Based on the analysis of our experimental results, it is evident that our proposed regularization algorithm makes the ANN capable of identifying good patterns in data and classifying them efficiently. Moreover, the proposed regularizer has outperformed the existing regularization algorithms when incorporated with ANN. As a result, the overall average accuracy achieved on NSL-KDD, UNSW-NB15, and CIDDS-001 datasets using 10-folds cross-validation is 98.53%, 94.58%, and 97.87%, respectively.

FUNDING

This work was supported by the Deanship of Scientific Research (DSR), King Abdulaziz University, Jeddah . The

authors, therefore, gratefully acknowledge DSR technical and financial support.

CONFLICT OF INTEREST

The authors declare that they have no conflict of interest.

ETHICAL APPROVAL

This article does not contain any studies with human participants performed by any of the authors.

DATA AVAILABILITY

Datasets used to support the findings of this study are included within the article.

REFERENCES

[1] M. Markos and S. Singh, "Novelty Detection: A Review-Part 1: Statistical Approaches," *J. Signal Processing*, vol. 83, 2003. 2481 2497,2003.

[2] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys*, vol. (CSUR), pp. 41(3)–15, 2009.

[3] M. Markou and S. Singh, "Novelty detection: a review – part 2: neural network based approaches, Signal Process," *Signal Process*, vol. 12, pp. 2499–2521, 2003.

[4] P. Murugan and S. Durairaj, "Regularization and optimization strategies in deep convolutional neural network," 2017.

[5] A. Ben-Hur and J. Weston, "A user's guide to support vector machines," vol. 609, pp. 223–239, 2010.

[6] K. Das, "Detecting Patterns of Anomalies," *Carnegie Mellon University*, 2009.

[7] H. T. M, "The Science of Anomaly Detection, Numenta," 2015.

[8] C. Xiang, P. C. Yong, and L. S. Meng, "Design of multiple-level hybrid classifier for intrusion detection system using Bayesian clustering and decision trees," pp. 918–924, 2008.

[9] G. Giacinto, R. Perdisci, M. D. Rio, and F. Roli, "Intrusion detection in computer networks by a modular ensemble of oneclass classifiers," 2008.

[10] W. Hu, W. Hu, and S. Maybank, "AdaBoost-based algorithm for network intrusion detection," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 38, no. 2, pp. 577–583, 2008. Cited By :137.

[11] A. Tajbakhsh, M. Rahmati, and A. Mirzaei, "Intrusion detection using fuzzy association rules," 2009.

[12] K. Shafi and H. A. Abbass, "An adaptive genetic-based signature learning system for intrusion detection," pp. 12036–12043, 2009.

[13] S.-Y. Wu and E. Yen, "Data mining-based intrusion detectors," pp. 5605–5612, 2009. doi.org/10.1016/j.eswa.2008.06.138 ID: 271506.

[14] T. P. Tran, L. Cao, D. Tran, and C. D. Nguyen, "Novel intrusion detection using probabilistic neural network and adaptive boosting," *arXiv preprint*, vol. 911, no. 0485 6, pp. 83–91, 2009.

[15] G. Wang, J. Hao, J. Ma, and L. Huang, "A new approach to intrusion detection using Artificial Neural Networks and fuzzy clustering," pp. 6225–6232, 2010.

[16] M. S. Mok, S. Y. Sohn, and Y. H. Ju, "Random effects logistic regression model for anomaly detection," pp. 7162–7166, 2010.

[17] R. Abdulhammed, M. Faezipour, A. Abuzneid, and A. AbuMallouh, "Deep and machine learning approaches for anomaly-based intrusion detection of imbalanced network traffic," *IEEE Sensors Letters*, vol. 3, p. 1–4, Jan 2019.

[18] A. Verma and V. Ranga, "On evaluation of network intrusion detection systems: Statistical analysis of cidds-001 dataset using machine learning techniques," *Pertanika Journal of Science & Technology*, vol. 26, pp. 1307–1332, march 2018.

[19] A. Verma and V. Ranga, "Statistical analysis of cidds-001 dataset for network intrusion detection systems using distance-based machine learning," *Procedia Computer Science*, vol. 125, p. 709–716, 2018.

[20] B. A. Tama and K.-H. Rhee, "An in-depth experimental study of anomaly detection using gradient boosted machine," *Neural Computing and Applications*, vol. 31, pp. 955–965, Apr 2019.

[21] B. A. Tama, M. Comuzzi, and K.-H. Rhee, "Tse-ids: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system," *IEEE Access*, vol. 7, p. 94497–94507, 2019.

[22] M. M. T.Jawhar and M. Mehrotra, "Design network intrusion detection system using hybrid fuzzy-neural network," *International Journal of Computer Science and Security*, vol. 4, no. 3, pp. 285–294, 2010.

[23] C. M. Rahman, D. M. Farid, and M. Z. Rahman, "Adaptive intrusion detection based on boosting and naive bayesian classifier," *International Journal of Computer Applications*, vol. 24, no. 3, pp. 11–19, 2011.

[24] M.-Y. Su, "Real-time anomaly detection systems for Denial-of-Service attacks by weighted k-nearest neighbor classifiers," pp. 3492–3498, 2011.

[25] M. S. Abadeh, H. Mohamadi, and J. Habibi, "Design and analysis of genetic fuzzy systems for intrusion detection in computer networks," pp. 7067–7075, 2011.

[26] W.-C. Lin, S.-W. Ke, and C.-F. Tsai, "CANN: An intrusion detection system based on combining cluster centers and nearest neighbors," 2015.

[27] A. Hadri, K. Chougdali, and R. Touahni, "Intrusion detection system using PCA and Fuzzy PCA techniques," in *Advanced Communication Systems and Information Security (ACOSIS), International Conference on. IEEE*, pp. 1–7, 2016.

[28] C. Yin, Y. Zhu, J. Fei, and X. He, "A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks," *IEEE Access*, vol. 5, no. 2017, pp. 21954–21961, 2017.

[29] U. N. Wisesty and Adiwijaya, "Comparative study of conjugate gradient to optimize learning process of neural network for Intrusion Detection System (IDS).," in *In Science in Information Technology (ICSITech), 2017 3rd International Conference on. IEEE*, pp. 459–464, 2017.

[30] J. Kim, J. Kim, H. L. T. Thu, and H. Kim, "Long short term memory recurrent neural network classifier for intrusion detection,," in *2016 International Conference on Platform Technology and Service (PlatCon) IEEE*, pp. 1–5, 2 2016.

[31] M. H. Ali, B. A. D. A. Mohammed, M. A. B. Ismail, and M. F. Zolkipli, "A new intrusion detection system based on Fast Learning Network and Particle swarm optimization," *IEEE Access*, vol. 6, no. 2018, pp. 20255–20261, 2018.

[32] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018.

[33] M. Tanaka, V. Sladek, and J. Sladek, "Regularization techniques applied to boundary element methods," *Applied Mechanics Reviews*, vol. 47, pp. 457–499, 1994.

[34] B. S. Kautz and Y. Jiang, "A general stochastic approach to solving problems with hard and soft constraints," *The Satisfiability Problem: Theory and Applications*, pp. 573–586, 1997.

[35] P. Simard, B. Victorri, Y. LeCun, and J. Denker, "Tangent prop-a formalism for specifying selected invariances in an adaptive network,," *Advances in neural information processing systems*, pp. 895–903, 1992.

[36] S. J. Hanson and L. Y. Pratt, "Comparing biases for minimal network construction withback-propagation," *Advances in neural information processing systems*, pp. 177–185, 1989.

[37] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout:a simple way to prevent neural networks from overfitting," *Journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[38] R. Moore and J. DeNero, "L1 and L2 Regularization for Multiclass Hinge Loss Models," in *Symposium on Machine Learning in Speech and Natural Language Processing (MLSLP), WA*, pp. 1–5, 2011.

[39] A. Y. Ng, "Feature selection, L1 vs. L2 regularization, and rotational invariance," in *ICML*, 2004.

[40] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[41] H. Deng and G. Runger, "Feature selection via regularized trees," in *Proc. 12th IEEE International Joint Conference on Neural*, pp. 1–8, 2012.

[42] H. Peng, L. Mou, G. Li, Y. Chen, Y. Lu, and Z. Jin, "A Comparative Study on Regularization Strategies for Embedding-based Neural Networks," 2015.

[43] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580," 2012.

[44] N. Moustafa and J. Slay, ""The significant features of the UNSW-NB15 and the KDD99 sets for Network Intrusion Detection Systems", the 4th International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS," 2015.

[45] M. Ring, S. Wunderlich, D. Grüdl, D. Landes, and A. Hotho, "Flow-based benchmark data sets for intrusion detection," in *Proceedings of the 16th European Conference on Cyber Warfare and Security (ECCWS)*, pp. 361–369, ACPI, 2017.