BioGecko Vol 12 Issue 03 2023

ISSN NO: 2230-5807

INTRUSION DETECTION SYSTEM TO CLASSIFY THE THREATS USING GREEDY BASED RECURRENT NEURAL NETWORKS

Hemavathi K
Research Scholar, Department of Computer Science and Applications,
St. Peter's Institute of Higher Education and Research,
Dr. R. Latha
Professor and Head, Department of Computer Science and Applications,
St. Peter 's Institution of higher education and research, Chennai.

Abstract
This paper describes a new intrusion detection system that can accurately spot suspicious traffic by using a variety of feature sets, making feature subsets that include these different feature sets, and then figuring out if these feature subsets contain the best feature. The proposed method utilizes a genetic crossover to determine which of the different features comprise the most desirable combination. The results show that the proposed method can obtain an accuracy of 98% in 5-class classification by utilizing LSTM-RNN in combination with crossover operation. In addition, when the research was working with a five-class classification, the accuracy of our training was 99%. According to the findings, the technique proposed can accurately determine the feature subset that contains the best features while utilizing only a modest amount of memory.
Keywords: Intrusion Detection System, Threats, Greedy, Crossover, Recurrent Neural Networks

1. Introduction

Cyber security is becoming an increasingly essential area of research of the growing cultural significance of networks in today society. Anti-virus software, network firewalls, and penetration surveillance systems are the fundamental components upon which all additional cyber security measures are constructed. It is possible to prevent malicious actors from gaining access to networks by utilizing these various strategies. One illustration of this category of surveillance system is a device known as an intrusion detection system (IDSs). To ensure that a network is safe from potential threats, an IDS will monitor both the software and hardware of the network [1].

The first generation of intrusion monitoring systems still has a high rate of false alarms. These systems send out a great number of alerts for low-level situations that do not represent a threat, which not only adds to the amount of work that security analysts must do but also increases the likelihood that they will ignore attacks that could cause significant damage [2].

A significant amount of effort has been made toward the development of improved IDS that are capable of more accurate identification and generate fewer false positives. Existing intrusion detection systems have the additional disadvantage of being unable to recognize risks that they have never been exposed to before, which is a significant limitation. The dynamic character of the environments in which networks operate, new forms of attack and variants of existing ones are continually being developed and implemented [3]. This highlights the importance of developing intrusion detection systems that can recognize new types of attack that have never been seen before.

Researchers have begun focusing their attention on developing IDS by making use of techniques associated with machine learning to find a solution that will fix the issues that were discussed earlier. Machine learning is a branch of artificial intelligence that enables massive datasets to be automatically mined for information in the search for new insights [4]. Intrusion detection systems that are based on machine learning can achieve detection levels that are satisfactory when sufficient training data is made available.

Additionally, the machine learning models have sufficient generalizability to distinguish attack variants as well as novel attacks when sufficient training data is made available [5]. In addition, to

design and implement intrusion detection systems that are founded on machine learning, one needs very little knowledge in the field that they are intended to protect [6].

In recent years, the subfield of machine learning known as deep learning has shown that it is capable of some remarkable feats. When it comes to the administration of enormous amounts of data, the performance of strategies based on deep learning is noticeably superior to that of their more conventional machine learning analogues. In addition, the methodologies of deep learning are efficient and effective because they can automatically acquire feature representations from raw data and then generate results [7]-[11].

In this paper, we introduce a deep learning-based intrusion detection system that can reliably identify malicious traffic using a wide range of features, generate feature subsets that incorporate these features, and assess the quality of these features.

2. Related works

The inability to directly handle flow data, feature engineering is required before conventional machine learning models can be applied. Feature vectors and deep models are the configuration that is used by default for method types that are built on feature engineering. Every single feature vector measure can be read off on its own and understood on its own. These methods of detection come with several advantages, the most important of which are that they are straightforward to put into action, produce superior results, and can keep up with the requirements that are enforced in real time [12]-[16].

During the classification process, Goeschel et al. [17] proposed utilizing a combination of SVM, a decision tree, and a Naive Bayes algorithm. The next step they took was to sort the samples into normal and abnormal classifications by applying a trained SVM algorithm to the data and seeing what results it produced. They were able to eliminate some of the possibilities for the different kinds of assaults that could have taken place on the out-of-the-ordinary samples by employing a model called a decision tree. However, a decision tree algorithm can only recognize dangers that it has already encountered. Therefore, a Naive Bayes classifier was also used to distinguish attacks that had not been seen before. This hybrid approach, which incorporates three different classifier types, obtained an accuracy of 99.62 on the KDD99 dataset while having a false alarm rate of 1.57%.

The rate of discovery is yet another intriguing subject that should be investigated [18] and proposed a detection method that was based on KNN as well as an accelerated calculation using parallel computing techniques that were carried out on a graphics processing unit. This was accomplished on a graphics processing unit (GPU). They changed the neighbor selection method that the KNN algorithm makes use of in its computations. In contrast to the traditional KNN, which chooses the K samples that are geographically closest to it as neighbors, the enhanced algorithm chooses a predetermined proportion of the samples that are geographically closest to it to act in that capacity. The method that was proposed is effective for use with sparse data and considers the fact that the distribution of the data is not consistent in any way. Experiments were performed using the KDD99 dataset, and it was revealed that the overall success rate was 99.3%.

IDS also makes use of autonomous learning techniques, and clustering algorithms provide a common data-splitting strategy. Both methods are utilized in the system. In addition, IDS makes use of a wide diversity of methods for splitting the data. When it comes to working with large databases, the conventional K-means algorithm struggles to deliver satisfactory results. A modified version of the K-means detection technique that made use of mini batches was proposed by Peng et al. [19] to improve the overall performance of the detection process. They started off by preprocessing the data that was given to them by KDD99. Standardizing each feature measure was accomplished with the help of the max-min technique, and the nominal features were transformed into forms that could be represented by numbers. After that, a method known as principal component analysis (PCA) was utilized to cut the number of dimensions down an even further. After that, the samples were organized into groups by making use of the K-means procedure, albeit with two adjustments. To prevent themselves from becoming mired in a situation where they were only able to achieve a suboptimal solution, they modified their initialization strategy and made the necessary modifications. They used the technique

of preparing the food in mini-batch quantities to reduce the amount of time necessary for the preparation. In comparison to the K-means algorithm used as a baseline, the recommended method produced results that are superior in terms of both accuracy and efficiency.

3. Proposed Method

Figure 1 illustrates the proposed architecture for the intrusion detection system. This plan is broken down into four distinct steps, also known as portions. The raw data must first go through a preprocessing step before moving on to the next part of the procedure. In the second part of this series, the primary emphasis is on the process of creating feature groups in SFS.

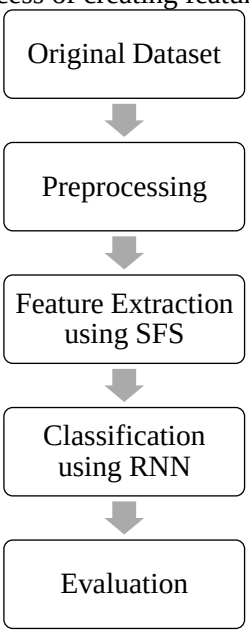

Figure 1. Proposed IDS framework

The accuracy and error ratings of each subset that was produced by a machine learning model are evaluated and taken into consideration. This step follows the previous step. The artificial intelligence method known as RNN is the one that is being employed in this scenario.

RNN is to make predictions about the levels of accuracy and loss ratings for each combined feature, and this is the assignment that they are currently working on. Following that, the feature subset that yields the greatest accuracy number is selected, which ushers us into the subsequent stage of the process. Third, when constructing IDS classifiers, the component of the feature data that was found to be the most reliable is the one that is used.

In the last stage of this procedure, the research will evaluate a variety of RNN models for IDS based on the values of two parameters. An investigation into the numerous IDS methods must initially be carried out as the first step in the process. The recommended model memory profilers are an example of the second type of measurement that can be performed.

3.1. Preprocessing dataset:

Perform the processing of the information in advance. The IDS procedure is now in the first portion of its progression. Selecting the XML files in the ISCX collection that contain the raw data, also known as unprocessed or unstructured information After that, the following steps are carried out in the following order: re-formatting XML data into CSV format; de-duplicating. Utilizing CSV files, you can generate two separate data sets to use for testing and training purposes. Imputation is used in circumstances where the NSL-KDD dataset includes values that are NULL to address the problem. This is done so that the problem can be solved. Utilizing this method makes it possible to fill in the blanks with several selection.

3.2. Feature Extraction

The efficiency of models based on machine learning is enhanced when there are fewer co-dependencies and when features contain a greater amount of information that is of value. Having many features in the feature field leads to unbiased classification but also a high rate of false positives in the findings. Within the confines of this study, the research tests the hypothesis that reducing the total number of input features could contribute to improvements in overall performance.

Gain ratio algorithm for the reduction of features are explored and discussed. This was one of the many discoveries that came about because of the research. This was just one of the many discoveries that came to light because of following this trail of investigation. The research has decreased the number of features (see Figure 2) by making use of the approaches that were discussed earlier, and the research make use of comparisons to evaluate the performance of RNN-ABC.

3.3.1. Subset Feature Selection (SFS) Model

SFS is narrow down the extensive feature accumulation of the original dataset to the components that are of the utmost importance to the undertaking.The goal is to discover a subset, denoted by $x=x_i|i=1\cdots n$, for any feature set in which $m<n$ maximizes some criterion function (CF), most particularly the probability of accurate classification. This is something that can be done for any feature collection that has the form $x_m=x_{i1},x_{i2},\cdots,x_{im}$.

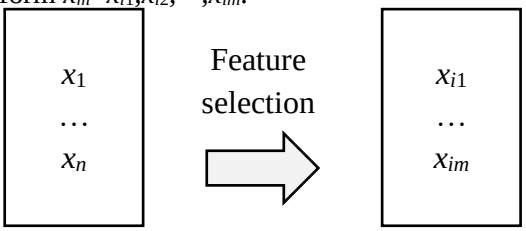

Figure 2. Feature Selection

The primary objective of the feature selection algorithms that have been discussed thus far is to condense the feature space $D=x_1,x_2,\cdots,x_n$ into a set of features D_n. This has been stated as the case throughout the discussion. This is done to improve or optimize the computational performance of the classification and to stay away from the curse of dimensionality. The second objective is to choose a subset of features from the feature space D that is small enough to be meaningful but not so small that it substantially degrades the performance of the classifier. This is referred to as the small enough to be meaningful threshold.

This corpus of work suggests that a hybridized SFS algorithm be used when selecting model features. The SFS is the method of performing a greedy search that is the most efficient and straightforward, and it works backward, beginning with the most recent findings. The sparse feature set (SFS) technique starts with an empty feature subset, which it then slowly starts to fill up with features drawn from the full input feature space.

When a new feature is added to a subset, the complete subset is evaluated, as is customary for the features that are already a part of the subset. This continues until the new feature is found to be compatible with the rest of the features in the subset. This process will continue until feature contained in the subset has been analyzed. The critical factor (CF) is the component that is accountable for establishing which feature, if added to the feature subgroup, will result in the greatest improvement in performance.

3.3.2. Data Encoding

Experiments to detect intrusion typically involve the accumulation of categorical data, which may or may not also contain numerical values, depending on the nature of the experiment. In general, mathematical models utilized in machine learning function at their highest levels when presented with quantitative data. Before being fed into the RNN, any textual values must, first and foremost, be transformed into their corresponding quantitative values. Most contributions consist of numerical data rather than written explanations. Because of this requirement, a technique known as one-hot encoding was implemented. This technique includes associating a one-of-a-kind binary variable with each value that is not numerical. If this were to be done, the system would be capable of producing superior

results while simultaneously reducing the probability that any unanticipated breakdowns would take place.

At the conclusion of the process, the data were processed using a method known as normalization to further reduce the amount of time that RNN-LSTM required to train and to generate results that were objective. It is standard practice to transform the data you are working with into the range [0, 1] whenever you are dealing with multiple types of data. The research were successful in accomplishing this by making use of a method known as min-max normalization, which can be formulated as follows:

$si=di-(\min(d)/\max(d)-\min(d))$,

where

$s(i)$ - normalized output and

$d=(d1,...,dn)$ - normalizedinput values.

The research begins the first iteration of the SFS algorithm with an empty set, and the research uses features x^+ in an iterative fashion until the model achieves the highest possible accuracy score. This continues until the final iteration of the algorithm. It is possible for the model to produce the greatest degree of precision when it is applied to the validation dataset, also known as the feature subset. The number of features has consequently been cut down because of this, most likely of the issue of dimensionality being resolved, which has resulted in the problem being eased.

Algorithm 1: Pseudocode of Feature Extraction

Input:$outputclass(Y_d)$

Output: accuracy and error score, Y_k

estimate the accuracyscores;

estimate the errorscores;

Find the complete feature d

Initialize the empty set ($Y_0=\emptyset$)

Pick the optimal feature using following expression:

$x^+=argmax[prediction(Y_k{}^+x^+)]$

$prediction= predictionscore(x^+, Y_k)$

$error = errorscore(x+, Y_d)$

prediction score.append(Y_k)

errorscores.append (Y_d)

Update the values of $Y_k{}^{+1}=Y_k{}^+x^+;k=k+1$

Estimate the optimal feature using step 5

Termination criterion is reached $k = d$

returnthe score

End

Input: Each dataset has an attachment type that is denoted by the variable output_class. This attachment type is class data. The input to the SFS is the aggregation of all of the features, which is indicated by $Y=y1,y2,\ldots,yd$. Each of the two features is taken into consideration as an input by the SFS algorithm. During the process of comparing the projected class of the model to the output class of the actual world, the model is employed. Following this step, STS will be able to ascertain the error and precision values that are particular to each feature generation that is included within the subgroup.

Output: The findings consist of a collection of features, along with a rating of how successfully they performed and an indication of how badly they failed. a group of features that are denoted by Y_k, $Y_k=y_j$, where $k=(0,1,2,\ldots,d)$. When the algorithm is provided with an a priori determination of the size of $k<d$, it will generate a subset of the feature space.

Initialization: The procedure is initialized with the empty set $Y_0=\emptyset$, the value 0 is assigned to the subgroup size variable, k. This is the value that will be used unless overridden.

Searching procedure: The method that will be used for the investigation will improve the operational capabilities of the feature subdivision $Y_k$by incorporating a recently discovered feature called x^+. The

optimal criterion function is optimized to the x^+feature. When combined with Y_k, accuracy provides the best potential criterion function for success in the classification performed by the model. The collection of behaviors that can be discovered in $Y_k$is the most desirable one that could possibly exist. The pattern will continue until the conclusion that was previously decided upon is achieved.

Termination: Once the number of features in your preferred combination (k) has been attained, the procedure will come to an end. Additional feature subsets of size k, denoted by Y_k, are added until the feature subset of size k includes the number of desired features d that was determined a priori.

3.3. Classifier

Relational neural networks, also known as RNNs, are a subtype of artificial neural networks. RNNs are distinguished from other types of artificial neural networks by the fact that the interactions between their nodes are designed to resemble the connections that are made between neurons in the human brain. Impulses can be passed along from one neuron or node in a neural network to another in a fashion that is analogous to the way synapses in a living brain transmit information between neurons.

The artificial neuron processes and evaluates the information that is being received, and then it communicates the findings of its analysis to the other neurons that are a part of the network. In most neural networks, individual neurons and connections are each given weights, which are then capable of being modified to fine-tune the way the network learns.

The signal degree of amplification can have its level adjusted as it travels from the input levels to the output layers by changing the weight. This is possible because the signal passes through a weighted system. In an artificial neural network (ANN), there are typically hidden layers located in the middle of the network, between the input layer and the output layer. RNNs are composed of input units, output units, and hidden units; the hidden units are the ones that carry out all the computations and produce the outputs.

The results of a comparison in which the error of one hidden layer is evaluated in relation to the error of the hidden layer that came before it is used by the RNN model to make modifications to the weights that are used between the hidden layers. These adjustments are based on the findings of the comparison. A fundamental RNN architecture is depicted in Figure 1; it has two hidden layers and comprises of those layers.

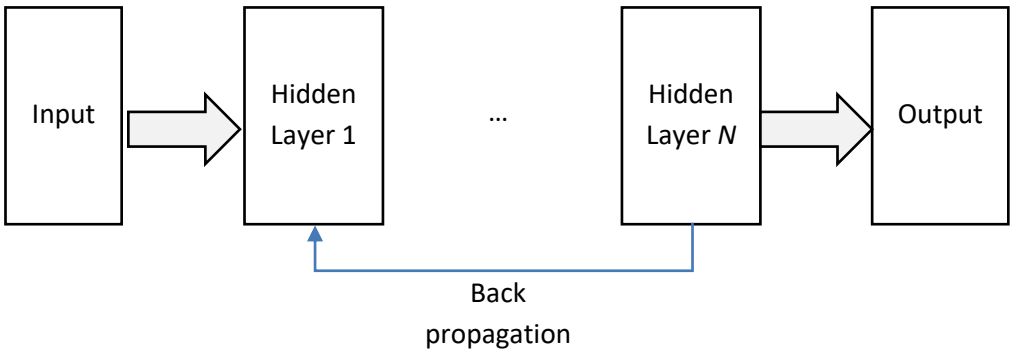

Figure 3: A simple RNN.

In feed-forward neural network (FFNNs), there are no cycles or loops; instead, the flow of data is unidirectional, traveling from input to concealed to output nodes and then back again. This pattern repeats itself. In conventional FFNNs, the treatment of hidden levels is managed as if it were a discretionary option. In this instance, X, H, and Y each stand for a distinct sequence of vectors. More specifically, these sequences are referred to as the supposed input vector sequence, the concealed vector sequence, and the output vector sequence, respectively. Take, for instance, the example of the input vector sequence, which is represented by X = ($x1$, $x2$, …, xT)When provided the input that t = 1 to T, the following is an example of how a conventional RNN would generate the sequences of hidden vectors H = ($h1$, $h2$, …, hT) and and output vectors Y = ($y1$, $y2$, …, yT).

$ht=\sigma$ ($Wxhxt+Whhht-1+bh$)
$yt=Whyht+by$
where
σ – nonlinearactivation function,
W - weight matrix, and
b - bias term.
The output of the hidden layer at time step t is denoted by the symbol $h_t$in equation (1), whereas the output of the hidden layer that came before it is denoted by the symbol h_{t-1}in the same equation.

Long Short-Term Memory

The use of LSTM makes it completely possible to learn with no errors at all. Long-term short-term memory, also referred to as LSTM, is a form of memory that can be taught to associate more than a thousand distinct time intervals. Memory fragments are used in place of each node that was originally contained within an LSTM network secret layer. At a bare minimum, one memory cell is included in each storage device. This is the absolute minimum that is required. A simple LSTM network is represented here by a single cell, which can be seen in the illustration found in Figure 2.
The regulating valves provide a stimulus for the activity of the memory cells. If it is determined that the data being stored there is no longer essential, forget gates make it possible for the linear unit data storage space to be cleared out and the data to be deleted entirely. The focus of the conversation right now is on these logical gates, which are also sometimes referred to as fundamental sigmoid threshold units. These activation functions have numbers that range from 0 to 1 and it is calculated as follows:
$ycj(t)=youtj(t)h(scj(t))$
where
$yout_j$- output gate activation,
sc_j- internal state of the output gate, and
h- output of the hidden layer.

4. Results and Discussions

The LSTM-RNN is implemented in TensorFlow and Keras were utilized; however, scikit-learn was utilized in the construction of the SVM and RF models. During the investigation into the effectiveness of the LSTM-RNN model, binary and multi-class/5-category classifications were both employed.
The results of a binary classification are the detection of normal and unusual behavior, while the results of a multi-class classification are the detection of normal behavior in addition to DoS, Probe, R2L, and L2R attacks. A binary classification has two possible outcomes: either normal behavior or abnormal behavior.
A total of 122 features, as well as the most comprehensive collection of 99 features that could be used, were utilized to successfully complete these classifications. The research was able to figure out, with the help of a genetic algorithm, which of the 99 different features comprised the most desirable combination.

Table 1: Parametric settings and its description

Hyper-Parameter	**Multi-Class Classification**
Learning Rate	0.01
Dropout	0.3
Activation function	Softmax
Optimizer	SGD
Epoch	500
Batch size	50
Loss function	Crossentropy

The researchers found that implementing the deep learning approach led to accuracy levels of 88.39% for binary classification and 79.10% for 5-class classification. The research concluded that the performance of the LSTM-RNN algorithm is after conducting research on the KDDTrain+ dataset.

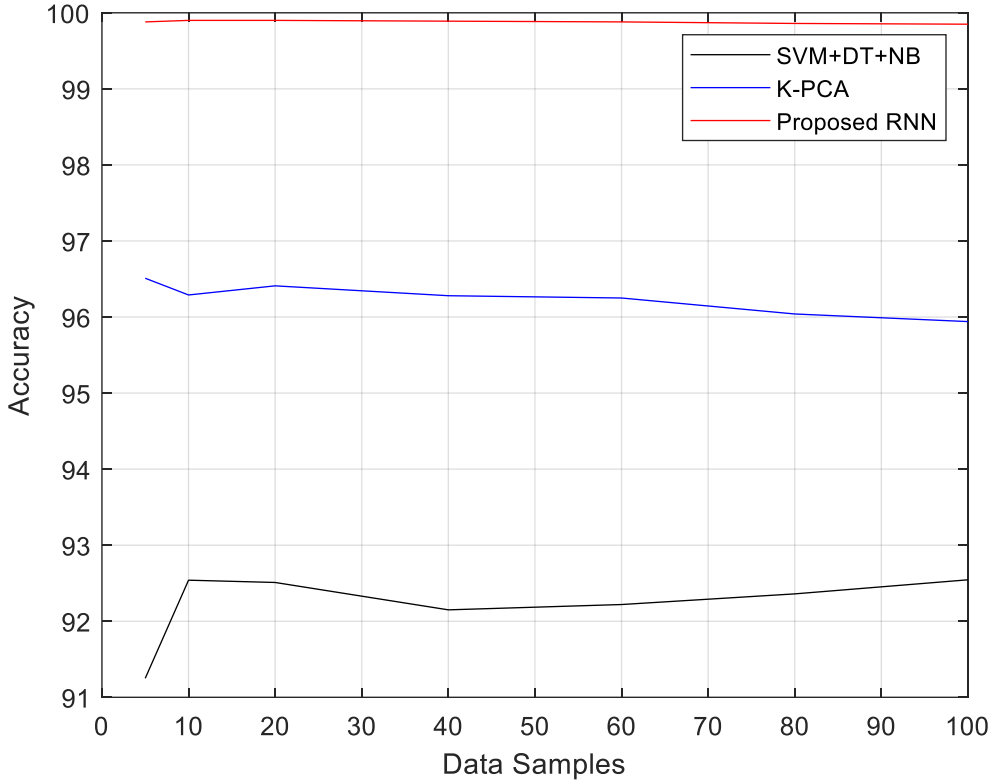

Figure 4: Accuracy

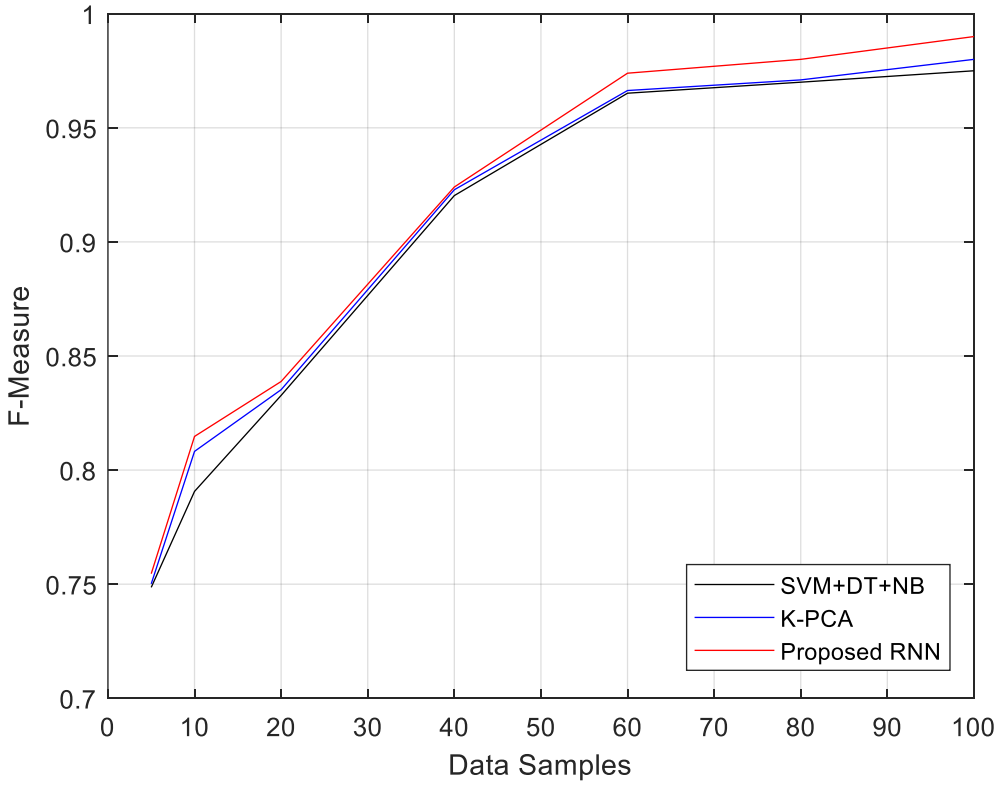

Figure 5: F-Measure

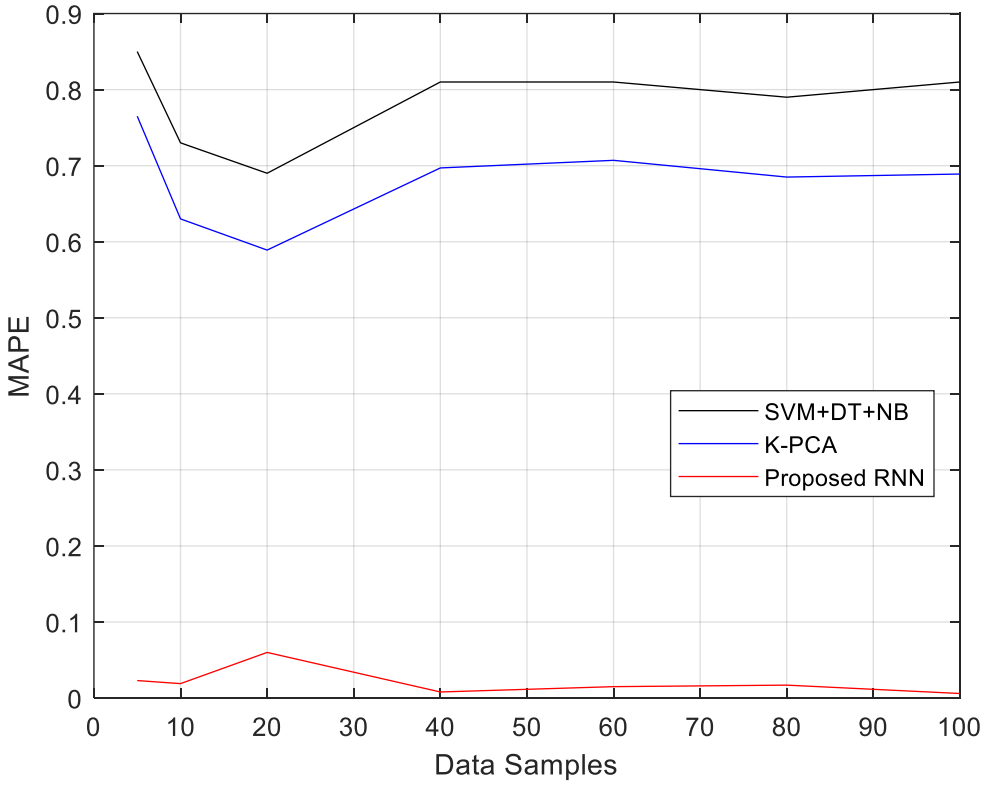

Figure 6: MAPE

The precision of 93.82% in 5-class classification by utilizing LSTM-RNN with the datasets from the KDD Cup' 99 after 1000 training repetitions at a learning rate of 0.1. This was achieved by using the datasets from the KDD Cup' 99. It was possible, as demonstrated by the results of our investigation, to achieve an accuracy of 99.99% when training for binary classification by utilizing LSTM-RNN in combination with SFS (Figure 4-6).

Additionally, a testing precision of 99.91% was attained using this very same combination. In addition, when the research was working with a five-class classification exercise, the accuracy of our training was 99%, while the accuracy of our evaluation was 93.88%.

In the plan that has been proposed, the purpose of the SFS is to produce and determine which feature subset from the entire feature collection is the best possible one. This is done by comparing all the potential subsets to each other. Even though the dataset will always have the same 15 original features, the results of applying the method that was proposed to data will change daily because the data that is most pertinent to the analysis will be different.

The result that was achieved by utilizing the method that was recommended is one that is sensitive to the values of the features. The model that has been proposed is capable of accurately recognizing suspicious traffic with a variety of feature sets, generating feature subsets that include this variety of features, and then evaluating whether these subsets contain the best feature.

In addition, the SFS that has been recommended has the eventual objective of model-driven feature selection as one of its goals. The paradigm that is being proposed is one that can deal with high-dimensional data, which is the primary contributor to the problem of dimensionality that is present. According to the findings, the technique proposed can accurately determine the feature subset that contains the best features while utilizing only a modest amount of memory.

5. Conclusions

DL-IDSs can identify suspicious traffic by utilizing multiple feature sets, generating feature subsets that include these multiple feature sets, and finally determining if these feature subsets contain the finest feature. In this article, we introduce a novel IDS that can reliably identify malicious data transfers. The suggested system can detect suspicious traffic by generating multiple features and then determining which feature set contains the most relevant information. The model employed in this investigation is a crossbreeding variation of the SFS algorithm to identify potentially useful characteristics for the classifier. Using LSTM-RNN, 99% accuracy in binary classification training is feasible. The results had a 99% success rate in a five-class classification exercise used in the study. The results show that the suggested method can efficiently identify the best feature subset with minimal memory consumption.

References

[1] Thakkar, A., & Lohiya, R. (2023). Fusion of statistical importance for feature selection in Deep Neural Network-based Intrusion Detection System. *Information Fusion*, *90*, 353-363.

[2] Kasongo, S. M. (2023). A deep learning technique for intrusion detection system using a Recurrent Neural Networks based framework. *Computer Communications*, *199*, 113-125.

[3] Alavizadeh, H., Alavizadeh, H., & Jang-Jaccard, J. (2022). Deep Q-learning based reinforcement learning approach for network intrusion detection. *Computers*, *11*(3), 41.

[4] Zivkovic, M., Bacanin, N., Arandjelovic, J., Strumberger, I., & Venkatachalam, K. (2022). Firefly algorithm and deep neural network approach for intrusion detection. In *Applications of Artificial Intelligence and Machine Learning: Select Proceedings of ICAAAIML 2021* (pp. 1-12). Singapore: Springer Nature Singapore.

[5] Soni, M., Singhal, M., & Katarya, R. (2022, June). Optimizing Deep Neural Network using Enhanced Artificial Bee Colony Algorithm for an Efficient Intrusion Detection System. In *2022 2nd International Conference on Intelligent Technologies (CONIT)* (pp. 1-7). IEEE.

[6] Mohamed, S., & Ejbali, R. (2023). Deep SARSA-based reinforcement learning approach for anomaly network intrusion detection system. *International Journal of Information Security*, *22*(1), 235-247.

[7] Belarbi, O., Khan, A., Carnelli, P., & Spyridopoulos, T. (2022, September). An intrusion detection system based on deep belief networks. In Science of Cyber Security: 4th International Conference, SciSec 2022, Matsue, Japan, August 10–12, 2022, Revised Selected Papers (pp. 377-392). Cham: Springer International Publishing.

[8] Halbouni, A. H., Gunawan, T. S., Halbouni, M., Assaig, F. A. A., Effendi, M. R., & Ismail, N. (2022, July). CNN-IDS: Convolutional Neural Network for Network Intrusion Detection System. In *2022 8th International Conference on Wireless and Telematics (ICWT)* (pp. 1-4). IEEE.

[9] Gupta, S. K., Tripathi, M., & Grover, J. (2022). Hybrid optimization and deep learning based intrusion detection system. *Computers and Electrical Engineering*, *100*, 107876.

[10] Ugendhar, A., Illuri, B., Vulapula, S. R., Radha, M., Alenezi, F., Althubiti, S. A., & Polat, K. (2022). A Novel Intelligent-Based Intrusion Detection System Approach Using Deep Multilayer Classification. *Mathematical Problems in Engineering*, *2022*.

[11] Durairaj, D., Venkatasamy, T. K., Mehbodniya, A., Umar, S., & Alam, T. (2022). Intrusion detection and mitigation of attacks in microgrid using enhanced deep belief network. *Energy Sources, Part A: Recovery, Utilization, and Environmental Effects*, 1-23.

[12] Ren, K., Wang, M., Zeng, Y., & Zhang, Y. (2022, October). An Unmanned Network Intrusion Detection Model Based on Deep Reinforcement Learning. In *2022 IEEE International Conference on Unmanned Systems (ICUS)* (pp. 1070-1076). IEEE.

[13] Chawla, A., Jacob, P., Farrell, P., Aumayr, E., & Fallon, S. (2022, April). Towards Interpretable Anomaly Detection: Unsupervised Deep Neural Network Approach using Feedback Loop. In *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium* (pp. 1-9). IEEE.

[14] Amanoul, S. V., & Abdulazeez, A. M. (2022, May). Intrusion detection system based on machine learning algorithms: A review. In *2022 IEEE 18th International Colloquium on Signal Processing & Applications (CSPA)* (pp. 79-84). IEEE.

[15] Heidari, A., & Jabraeil Jamali, M. A. (2022). Internet of Things intrusion detection systems: A comprehensive review and future directions. *Cluster Computing*, 1-28.

[16] Gobinathan, B., Mukunthan, M. A., Surendran, S., Somasundaram, K., Moeed, S. A., Niranjan, P., ... & Sundramurthy, V. P. (2021). A novel method to solve real time security issues in software industry using advanced cryptographic techniques. Scientific Programming, 2021, 1-9.

[17] Goeschel, K. (2016, March). Reducing false positives in intrusion detection systems using data-mining techniques utilizing support vector machines, decision trees, and naive Bayes for off-line analysis. In *SoutheastCon 2016* (pp. 1-6). IEEE.

[18] Kuttranont, P., Boonprakob, K., Phaudphut, C., Permpol, S., Aimtongkhamand, P., KoKaew, U., ... & So-In, C. (2017). Parallel KNN and neighborhood classification implementations on GPU for network intrusion detection. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, *9*(2-2), 29-33.

[19] Peng, K., Leung, V. C., & Huang, Q. (2018). Clustering approach based on mini batch kmeans for intrusion detection system over big data. *IEEE Access*, *6*, 11897-11906.