International Journal of Engineering Research and Technology. ISSN 0974-3154, Volume 12, Number 12 (2019), pp. 3027-3031

Deep Learning-based Network Attack Detection Using Convolutional and Recurrent Neural Networks

Bayan Alsughayyir[1,3] and Ali Mustafa Qamar[1,2,4]

[1]*Department of Computer Science, College of Computer, Qassim University, Al-Mulaida, 51431, Saudi Arabia.*

[2] *BIND Research Group, College of Computer, Qassim University, Al-Mulaida, 51431, Saudi Arabia.*

[3]*ORCID: 0000-0001-6128-3196,* [4]*ORCID: 0000-0003-2440-7668*

Abstract

With the advancement of networks, attacks on networks have also increased significantly. The importance of maintaining sensitive information being stored and sent through the Internet has led to the urgent need to safeguard the networks. It is imperative to detect any abnormal behavior. Several methods have been employed either to prevent or at least to detect various attacks. In this research, deep learning-based approaches such as auto-encoders, Convolutional Neural Networks (CNN), and Recurrent Neural Networks (RNN) are used in order to create an efficient Intrusion Detection System (IDS). The proposed approaches are aimed at distinguishing the normal behavior on the network from an anomaly-based one. This research focuses on the evaluation and comparison of different deep neural networks. Based on the experimental results, RNN outperforms all other approaches with an accuracy of 0.99 for training and 0.94 for the testing phase.

Keywords: Network Security, Network attack detection, Deep Learning

I. INTRODUCTION

In the Internet environment, network systems have been exposed to violations and some security problems. With the availability of techniques to protect computers as well as networks against intrusions, we still need other methods to help detect increased threats automatically and in advance.

To combat and reduce attacks, we must continue to develop security techniques and defense mechanisms. The Intrusion Detection System (*IDS*) is used to monitor the events occurring in a computer network and to detect any unauthorized usage or abnormal actions. Several techniques have been used to build the *IDS*. There are many challenges to improve efficient *IDS*. Over the years, different techniques have been developed to enhance *IDS* using data mining and machine learning methods. However, all these technologies still suffer from some limitations and problems, which allow an attacker to make a violation of the system. Deep learning methods are a type of machine learning to learn the representation of data for feature extraction without human intervention. *DL* is superior to other techniques because of its ability to achieve a higher classification accuracy and to protect the network from malicious attacks. In recent years, deep learning has gained interest in the research community because these algorithms can automate the feature learning process. In this paper, the aim is to build an IDS to get acceptable results with higher accuracy for intrusion detection. We propose and construct a system to classify each record into one of the possible categories, normal or a particular kind of intrusion. Various popular *DL* techniques have been applied and evaluated in this paper for *IDS.* As such, an earlier version of this research employing Auto-encoders is published as a conference paper [1]. In the current paper, both CNN, as well as RNN, are used to detect the network attacks.

The paper is structured as follows: the related work is discussed in Section II. Section III demonstrates the proposed approach. The experiments and the results are presented in Section IV. The conclusion is provided in Section V.

II. RELATED WORK

To maintain network security, *IDS* tries to detect any unauthorized usages of the network. There are two types of attack detection, either on a network or on each of the hosts. Network-Based intrusion detection is a system used for analyzing network traffic while Host-Based intrusion detection is used for monitoring the host behavior [2]. Many of the researches employ data mining techniques to detect intrusion. Each technique has its pros and cons, and there is no ideal technique. *K-Means* has been used for intrusion detection in [2], [3] and [4] to effectively build clusters of related subclasses.

Support Vector Machines (SVMs) have been applied for *IDS* in numerous articles. [5] and [6] proposed a method of detecting anomaly based on the payload. The system uses one-class *SVM* to detect anomalies. In [6], classifier ensembles are used to achieve a very low false-positive rate with high detection accuracy. *IDS* was constructed from several one-class *SVM* classifiers. The authors in [7] applied *SVM* to study automatic feature selection in anomaly detection. Another *SVM* approach for detecting the intrusion is proposed in [8]. The authors apply the hypothesis test theory to the *SVM* classifier to build an intrusion detection model. The results illustrate that their approach can learn and generalize in a better manner.

In [9], the authors measure the performance of six different classification models for network intrusion detection: SVM, Neural Networks, k-Nearest Neighbor (kNN), Ripper Rule, Naïve Bayes, and Decision Tree. The results show that the kNN got the least computational complexity and the highest classification accuracy. A system for intrusion detection is

proposed in [10], which is based on a data mining technique. The system is a combination of binary classifiers with feature selection and multi-boosting which forms an ensemble method. The cost and accuracy of this method overcame the best entry of the KDD Cup'99. The same data is also used by [11]. The intrusions were detected using a hybrid classifier. This model uses the false alarm rate, accuracy, along with the detection rate. The experimental results show that the hybrid model generates better results with reasonable prediction time. Another study by [12] proposes an intelligent network IDS using the Averaged One Dependence Estimators algorithm to detect any malicious activity on a network. This method can efficiently give low False Alarm Rate (FAR) and high Detection Rate (DR) as compared to Naïve Bayes. In another work [13], data mining techniques were used to build a model to discover a new attacking signature based on the recognized signature. The results showed that the proposed model in comparison with the baseline apriori algorithm performs better.

Using more than one data mining algorithm helps obtain better predictive performance as compared to using just one algorithm. In [14], the authors recommended a model based on boosted decision trees to improve the performance of the *IDS*. The system is validated on the KDD Cup'99 dataset and compared with algorithms like *Naïve Bayes* and *k Nearest Neighbor*. The results show that the ensemble technique performs better than other algorithms to solve the problem of IDS. The authors in [15] introduce a prototype for the IDS by using a database-centric approach. This system uses the apriori algorithm to find intrusions by generating the rules. In [16], the authors make an improvement in the FP-Growth algorithm, which is based on associative analysis for network IDS. The improved algorithm requires less time than the FP-Growth algorithm and has better detection efficiency.

The authors in [17] are able to detect intrusion in a wireless sensor network by applying data mining techniques. They use misuse-based as well as anomaly-based detection techniques in the proposed system. The researchers in [18] compare the performance of different algorithms such as J48, Decision Tree, and OneR to verify whether a system is under Denial of Service (DoS) attack. The experiments are conducted using a subset of the KDD dataset. The results show that both the rule-based as well as the decision tree classifiers perform well and achieve more than 99% accuracy.

Deep Learning algorithms use a neural network with multiple hidden layers between the input and output layer for intrusion detection to construct a self-adaptive system in a dynamic network environment. The architecture can be classified into two kinds: *discriminative* and *generative*. In generative architecture, pattern classification is performed using unsupervised learning. On the other hand, the discriminative architecture use labeled data to train the model [19]. Stacked Restricted Boltzmann Machines (RBM) and Stacked Auto-encoder are used in [20] to find the network attacks accurately. Their techniques classify attacks into five classes with high accuracy. The results show that the Stacked Auto-encoder performs better as compared to the one using *RBM*. Alrawashdeh and Purdy [21] developed a deep network architecture so as to detect various anomalies. Their method uses Logistic Regression softmax for fine-tuning the deep network and got good results on 10% of the test data belonging to the KDD dataset.

III. PROPOSED *IDS*

DL relies on learning representations to capture the inherent structure in the data [22]. *DL* approach is the composition of various interconnected intermediate layers, which learn the input to perform tasks like classification or prediction. There are various models in the deep learning paradigm such as Auto-encoders [23], *RBM*s [24], *Convolutional Neural Networks* (*CNN*s) [25], *Recurrent Neural Networks (RNNs)* [26] and *Long Short-Term Memory* (*LSTM*) [27]. The *DL* configurations that are proposed for this study in order to evaluate the *NSL-KDD* data are based on auto-encoder, *RNN* and *CNN* networks.

III.I Auto-encoders

Auto-encoder is an unsupervised deep learning algorithm based on neural networks [23] and is composed of an encoder and a decoder. It is composed of just three layers: input, one hidden and an output layer. The input layer in auto-encoder is trained by learning the best parameters needed for output. The encoder is used to develop a new feature set that has a lower dimensionality as compared to the input for the hidden layers. The decoder is used to reconstruct its input back from the learned representation [19].

Our training and testing set of encoders consist of two fully connected layers. The input layer has 122 dimensions. The model reduces the number of dimensions to 61. The last layer employs the *Softmax layer* to produce five output classes.

III.II Convolutional Neural Networks (CNNs)

Convolutional Neural Network (CNN) is also a deep learning technique, mostly used in image classification. A convolutional neural network usually contains three kinds of layers: convolution, pooling, and fully connected layers. The convolution layer is used to extract the high-level features from the input. The pooling layer is responsible to decrease the parameters of the input in order to simplify the output. The last layer in the network is a fully connected one, which provides the classification output [25].

The proposed model of CNN has two convolution layers with 64 filters and uses *Relu* as the activation function, followed by a pooling layer with size 2 so as to reduce the complexity of the output. Another two convolution layers follow to learn higher-level features with 128 filters and also use *Relu* as the activation function. Then, the learned features are flattened, which is followed by a fully connected layer passing through a dropout layer. Finally, the five outputs for the model are produced in the output layer using a fully connected layer. Fig. 1 presents this configuration.

III.III Recurrent Neural Networks (RNNs)

Recurrent Neural Networks (RNNs) are widely used for learning from sequential training data. It trains the model by using the back-propagation methodology. Recurrent networks are different than the *Multi-Layer Perceptron (MLP)* since they not

only consider the current input but also take into account what has happened previously [26].

The proposed model of RNN is shown in Fig. 2. It consists of three layers of RNN with dropout followed by the output layer using a fully connected layer.

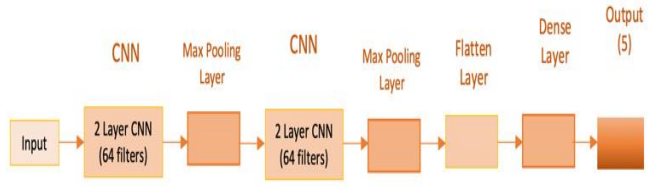

Fig. 1. The architecture of CNN

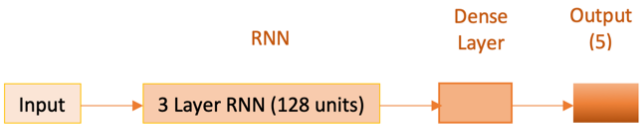

Fig. 2. The architecture of RNN

IV. EXPERIMENTS AND RESULTS

This section provides details about the experimental setup along with a comprehensive evaluation of the proposed approaches.

IV.I Experimental Setup

The KDD dataset is constructed from the data gathered by the IDS evaluation program of DARPA'98. *NSL-KDD* is a new version of the KDD dataset that was proposed as a solution to the problems encountered in the KDD dataset. The novel dataset overcomes and removes the redundant data of the KDD dataset. It contains 125,973 and 22,544 instances for training and testing respectively [28]. The number of features is 41. The data is either labeled as normal or a particular attack type. There are 39 different attack types that are grouped into four main categories of attacks: *DoS*, *Probing*, *Remote to Local* (*R2L*), and *User to Root* (*U2R*) [29].

IV.II Preprocessing and Model Construction

The following preprocessing tasks are applied to the data:

- All string attributes are converted to numerical ones.
- *Min-Max Scaler* is used to make normalization of all features.
- The classes associated with an attack are decreased from 39 to four along with a normal class.
- *Synthetic Minority Oversampling TEchnique (SMOTE)* was applied to the data.

The NSL-KDD dataset suffers from a disparity in the number of attack classes which makes the dataset as imbalanced and affects classification. To solve this problem, we applied *SMOTE* on the training data. Oversampling is applied by increasing the number of instances belonging to the minority class. The number of instances for training is increased from 125,973 instances to 193,264 instances. The training dataset before and after applying *SMOTE* is shown in Fig. 3 and Fig. 4 respectively. One can observe that before applying SMOTE, class 2 and 3 contained very few instances. However, SMOTE helped to increase the instances belonging to class 2.

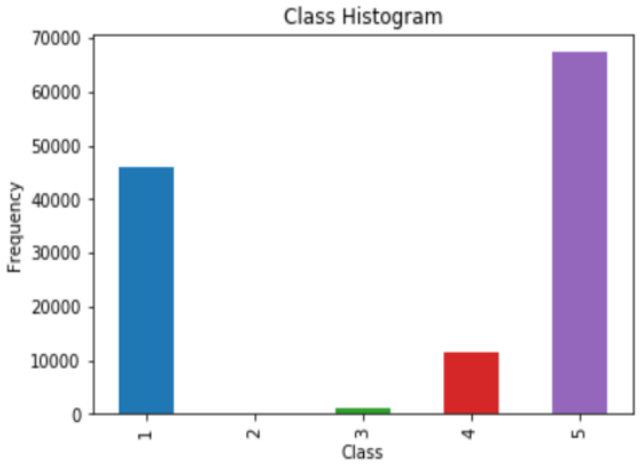

Fig. 3. Training dataset without SMOTE

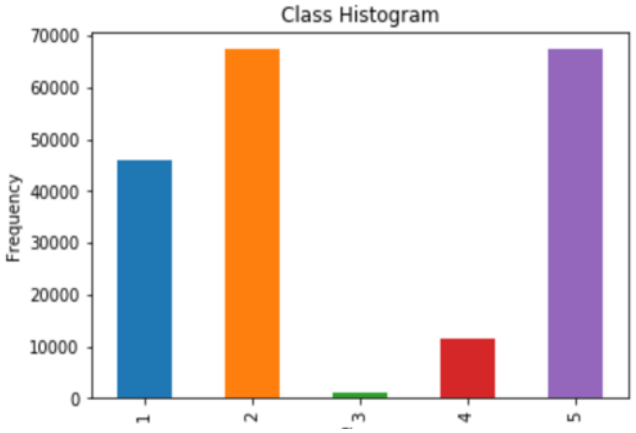

Fig. 4. Training dataset with SMOTE

After preprocessing, a deep neural network is constructed. The experiments are performed to find the optimal deep learning model for IDS having good performance as well as high accuracy. In all of the three models, the activation function for the hidden layers is *Relu* and the activation function of the output layer is *Softmax*. The experiments are performed using *Keras* [30] and *TensorFlow* [31].

IV.III Evaluation

The three proposed approaches are compared in terms of *accuracy*, *precision*, *recall*, *support*, *F1-measure*, and *Average Under Curve (AUC)*. Metric scores of CNN, Auto-encoder, and

RNN are presented in Table 1-3. The best results are given in bold.

Table 1. Results for CNN Model

Attack	*Precision*	*Recall*	F1-Measure	*Support*
U2R	0.71	0.90	0.83	36
DoS	**1.00**	0.99	**1.00**	6,056
R2L	**1.00**	**1.00**	**1.00**	931
Probe	**1.00**	**1.00**	**1.00**	2,260
Normal	**1.00**	**1.00**	**1.00**	**13,260**
Average	*0.942*	*0.98*	*0.97*	*4508.6*

Table 2. Results for Auto-encoder Model

Attack	*Precision*	*Recall*	F1-Measure	*Support*	*AUC*
U2R	0.66	0.13	0.22	202	0.56
DoS	0.93	0.76	**0.84**	7,456	**0.86**
R2L	**0.97**	0.02	0.05	2,754	0.51
Probe	0.78	0.59	0.67	2,421	0.78
Normal	0.65	**0.97**	0.78	**9,710**	0.78
Average	*0.798*	*0.49*	*0.51*	*4508.6*	*0.70*

Table 3. Results for RNN Model

Attack	*Precision*	*Recall*	F1-Measure	*Support*	*AUC*
U2R	0.50	0.89	0.64	202	0.93
DoS	**1.00**	**1.00**	**1.00**	7,456	**0.99**
R2L	0.99	0.61	0.75	2,754	0.77
Probe	0.71	0.92	0.80	2,421	0.93
Normal	0.98	1.00	0.99	**9,710**	**0.99**
Average	*0.836*	*0.88*	*0.84*	*4508.6*	*0.92*

Table 4. Comparison of accuracy for different DL methods

Model	*Accuracy*
CNN	0.74
Auto-encoder	0.91
RNN	0.94

The models were compared using the metric scores obtained on the test data. Table 1-3 show that CNN has very low precision (0.66) as well as recall (0.13) while detecting the U2R attacks. Consequently, the F-measure, which is the harmonic mean of precision and recall is also quite low (0.22). On the other hand, the F-measure for the auto-encoder is 0.64, whereas, for RNN, the value of F-measure is 0.83. The primary reason is that there are very few U2R attacks in the training dataset as compared to the testing dataset. Moreover, there are some new U2R attacks in the testing dataset that never appeared in the training dataset. As a result, these types of attacks are not fully trained by different classifiers which results in a misclassification. Furthermore, the recall for R2L while using CNN is just 0.02. This also results in a very low value for the F-measure (0.05). The auto-encoder is able to detect 61% of the R2L attacks and have an F-measure of 0.75. Lastly, RNN is able to detect all of the R2L attacks and has an F-measure of 1.00.

Overall, our results show that the proposed RNN achieved the best performance results on the NSL-KDD datasets. Table 1-3 also include the average results for all the classes. This shows that the RNN is able to learn the data distribution and the inherent structure in the data. Table 4 shows that RNN has higher accuracy as compared to CNN and Auto-encoder. The proposed RNN performed the best, reaching accuracies of 0.94 on the NSL-KDD while Auto-encoder achieves 0.91 and CNN is able to achieve 0.74.

IV.IV Comparative Analysis

In order to assess the effectiveness of the proposed approaches, we compare them with different machine learning algorithms. The comparison was performed with SVM, J48, Random forest, Logistic regression, and Auto-encoder as depicted in Table 5. These algorithms are considered because of their better performance among the state-of-the-art. One can observe that the RNN outperforms all the other methods for training as well as the testing phases. The performance of Auto-encoder on the training data is also very good. However, the testing accuracy of Auto-encoder is slightly lower than that of RNN.

J48 got the lowest performance among all the classifiers. Similarly, other than RNN, Random forest was able to achieve the best performance. Lastly, the RNN performs better as compared to all of the classical algorithms getting an accuracy of 0.94.

Table 5. Training and Testing Accuracy for Different Machine Learning Algorithms

Algorithm	Training accuracy	Testing accuracy
Logistic Regression	0.822	0.726
J48	0.801	0.714
SVM	0.944	0.735
Random forest	0.968	0.758
Auto-encoder	**0.999**	0.913
RNN	**0.999**	**0.94**

V. CONCLUSION

In this paper, several deep learning-based frameworks for intrusion detection have been presented. The *DL* configurations are constructed from Auto-encoders, RNN and CNN networks based on the *NSL-KDD* dataset. The classification problem is modeled as a five-class problem. The results show that the RNN-based deep learning approach achieves very high accuracy for both the training as well as the test phases. RNN performs better than other approaches with an accuracy of 0.99 for training and 0.94 for the test phase. The proposed models could be further improved by using real-time network traffic.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the Qassim University, represented by the Deanship of Scientific Research, on the material support for this research under the number 5292-coc-2018-1-14-S during the academic year 1439 AH / 2018 AD.

REFERENCES

[1] B. Alsughayyir, A. M. Qamar and R. Khan, "Developing a Network Attack Detection System Using Deep Learning," *Proc. IEEE International Conference on Computer and Information Sciences (ICCIS)*, Sakaka, Saudi Arabia, April 2019.

[2] H. D. Widiputra, "Clustering Based Intrusion Detection for Network Profiling Using K-Means, ECM and K-Nearest Neighbor," *Proc. Konferensi Nasional Sistem dan Informatika, Bali, Indonesia*, pp. 247–251, 2009.

[3] S. Zanero and G. Serazzi, "Unsupervised Learning Algorithms for Intrusion Detection," *Proc. IEEE Network Operations and Management Symposium (NOMS)*, Salvador, Brazil, April 2008.

[4] M. Jianliang, S. Haikun, and B. Ling, "The application on intrusion detection based on K-means cluster algorithm," *Proc. Int. Forum Inf. Technol. Appl. (IFITA)*, China, May 2009.

[5] N. Görnitz, M. Kloft, K. Rieck, and U. Brefeld, "Active learning for network intrusion detection," *Proc. 2nd ACM Work. Secur. Artif. Intell. - AISec '09*, p. 47-54, Nov. 2009.

[6] R. Perdisci, D. Ariu, P. Fogla, G. Giacinto, and W. Lee, "McPAD: A multiple classifier system for accurate payload-based anomaly detection," *Comput. Networks*, vol. 53, no. 6, pp. 864–881, 2009.

[7] M. Kloft, U. Brefeld, P. Düessel, C. Gehl, and P. Laskov, "Automatic feature selection for anomaly detection," *Proc. 1st ACM Workshop on AISec*, pp. 71–76, Oct. 2008.

[8] J. Yuan, H. Li, S. Ding, and L. Cao, "Intrusion detection model based on improved support vector machine," *Proc. 3rd Int. Symp. Intell. Inf. Technol. Secur. Informatics (IITSI)*, pp. 465–469, Jinggangshan, China, April 2010.

[9] C. So-In, N. Mongkonchai, P. Aimtongkham, K. Wijitsopon, and K. Rujirakul, "An evaluation of data mining classification models for network intrusion detection," *Proc. Fourth Int. Conf. Digit. Inf. Commun. Technol. its Appl. (DICTAP)*, pp. 90–94, 2014.

[10] C. Dartigue, H. I. Jang, and W. Zeng, "A new data-mining based approach for network intrusion detection," *Proc. 7th Annu. Commun. Networks Serv. Res. Conf. (CNSR)*, pp. 372–377, Moncton, Canada, May 2009.

[11] V. Barot and D. Toshniwal, "A new data mining based hybrid network Intrusion Detection model," *Proc. IEEE Int. Conf. Data Sci. Eng. (ICDSE)*, pp. 52–57, Cochin, Kerala, India, July 2012.

[12] A. Sultana and M. A. Jabbar, "Intelligent network intrusion detection system using data mining techniques," *Proc. IEEE 2nd Int. Conf. Appl. Theor. Comput. Commun. Technol. (iCATccT)*, pp. 329–333, Bangalore, India, July 2016.

[13] H. Zhengbing, L. Zhitang, and W. Junqi, "A novel Network Intrusion Detection System (NIDS) based on signatures search of data mining," *Proc. - 1st Int. Work. Knowl. Discov. Data Mining, WKDD*, pp. 10–16, Adelaide, SA, Australia, Jan. 2008.

[14] M. Gudadhe, P. Prasad, and K. Wankhade, "A new data mining based network intrusion detection model," *Proc. Int. Conf. Comput. Commun. Technol.*, Allahabad, India, Sept. 2010.

[15] R. G. M. Helali, "Data Mining Based Network Intrusion Detection System: A Survey," *Proc. Int. Conf. Telecommunications and Networking (TeNe)*, pp. 501–505, 2010.

[16] D. Fu, S. Zhou, and P. Guo, "The Design and Implementation of a Distributed Network Intrusion Detection System Based on Data Mining," *Proc. WRI World Congress on Software Engineering*, vol. 3, pp. 446–450, 2009.

[17] L. Coppolino, S. DAntonio, A. Garofalo, and L. Romano, "Applying Data Mining Techniques to Intrusion Detection in Wireless Sensor Networks," *Proc. Eighth Int. Conf. P2P, Parallel, Grid, Cloud Internet Comput.*, pp. 247–254, Compiegne, France, 2013.

[18] H. Waguih, "A Data Mining Approach for the Detection of Denial of Service Attack," IAES *Int. J. Artif. Intell., pp. 99-106*, vol. 2, no. 2, 2013.

[19] I. Goodfellow, Y. Bengio, and A. Courville, "Deep Learning," *MIT Press*, vol. 521, no. 7553, p. 785, 2016.

[20] N. T. Van, T. N. Thinh, and L. T. Sach, "An anomaly-based network intrusion detection system using deep learning," *Proc. Int. Conf. Syst. Sci. Eng. (ICSSE)*, pp. 210–214, Ho Chi Minh City, Vietnam, July 2017.

[21] K. Alrawashdeh and C. Purdy, "Toward an online anomaly intrusion detection system based on deep learning," *Proc. 15th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, pp. 195–200, Anaheim, CA, USA, Dec. 2017.

[22] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 26–38, 2017.

[23] O. Kuchaiev and B. Ginsburg, "Training deep Autoencoders for collaborative filtering," arXiv, Oct. 2017.

[24] G. Hinton, "A practical guide to training restricted Boltzmann machines," *Dept. Comput. Sci., University of Toronto*, Aug. 2010.

[25] W. Rawat and Z. Wang, "Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review," *Neural Computation*, vol. 29, no. 9, pp. 2352–2449, 2017.

[26] N. Ketkar, "Recurrent Neural Networks," in Deep Learning with Python, Berkeley, CA: Apress, 2017, pp. 79–96.

[27] J. Cheng, L. Dong, and M. Lapata, "Long short-term memory-networks for machine reading," *Proc. Conf. Empir. Methods Nat. Lang. Process. (EMNLP)*, pp. 551–561, Austin, Texas, 2016.

[28] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," *Proc. IEEE symposium on Computational intelligence in security and defense applications (CISDA)*, 2009.

[29] V. Kumar, H. Chauhan, and D. Panwar, "K-Means Clustering Approach to Analyze NSL-KDD Intrusion Detection Dataset," *Int. J. Soft Comput. Eng. (IJSCE)*, vol. 3, no. 4, pp. 1–4, 2013.

[30] F. C. Keras., "Keras" Accessed: Dec 1, 2019. [Online]. Available: https://github.com/fchollet/keras," 2017.

[31] M. Abadi et al., "TensorFlow: Large-scale machine learning on heterogeneous distributed systems.", 2015.