Comparative Analysis of Deep Learning Models for Network Intrusion Detection Systems

Brenton Budler
School of Computer Science and Applied Mathematics
The Universiy of the Witwatersrand
Johannesburg, South Africa
1827655@students.wits.ac.za

Ritesh Ajoodha
School of Computer Science and Applied Mathematics
The Universiy of the Witwatersrand
Johannesburg, South Africa
ritesh.adjoodha@wits.ac.za

***Abstract*—Detecting network intrusions is an imperative part of the modern cybersecurity landscape. Over the years, researchers have leveraged the ability of Machine Learning to identify and prevent network attacks. Recently there has been an increased interest in the applicability of Deep Learning in the network intrusion detection domain. However, Network Intrusion Detection Systems developed using Deep Learning approaches are being evaluated using the outdated KDD Cup '99 and NSL-KDD datasets which are not representative of real-world network traffic. Recent comparisons of these approaches on the more modern CSE-CIC-IDS2018 dataset, fail to address the severe class imbalance in the dataset which leads to significantly biased results. By addressing this class imbalance and performing an experimental evaluation of a Deep Neural Network, Convolutional Neural Network and Long Short-Term Memory Network on the balanced dataset, this research provides deeper insights into the performance of these models in classifying modern network traffic data. The Deep Neural Network demonstrated the best classification performance with the highest accuracy (84.312%) and F1-Score (83.799%) as well as the lowest False Alarm Rate (2.615%).**

I. Introduction

The use of the internet and other interconnected networks has grown exponentially in recent years. This rapid development of online technologies has, however, been accompanied by an equally rapid increase in cyberattacks. Globally, Cybersecurity Ventures predicts that cybercrimes will cost companies $10.5 trillion by 2025. The Cost of Data Breach Report 2020 by researchers at IBM suggests that these devastating consequences are a result of cybersecurity breaches taking an average of 280 days to identify and contain. One solution to these constantly evolving attack scenarios is cybersecurity systems built on the foundations of Artificial Intelligence (AI) which can leverage the ability of Machine Learning (ML) to identify anomalies and prevent these attacks. One such system is a Network Intrusion Detection System (NIDS) which monitors incoming and outgoing network traffic in an attempt to identify threats.

NIDSs are divided into two major categories; signature-based and anomaly-based [1]. Signature-based systems, also known as misuse-based systems, operate by comparing network traffic with a stored database of known attack patterns and flagging potential threats based on their similarity to these attacks [2]. The most prevalent drawback of signature-based systems is the inability to identify novel attacks [3]. The storing and maintaining of a database of all known attack patterns is also computationally expensive [4]. On the other hand, anomaly-based systems first determine a benign network traffic baseline and then identify any network activity which deviates significantly from this baseline as a potential threat [5]. The ability of anomaly-based systems to identify unknown attacks has resulted in it becoming the most widely used approach in identifying network intrusions [6].

Recently, researchers have developed anomaly-based systems using ML techniques and have demonstrated the ability of these systems to efficiently solve the network intrusion detection problem [7]. The construction of these traditional ML models unfortunately relies heavily on feature engineering, a time-consuming task requiring a high-level of domain expertise [8]. Conversely, multi-layer Deep Learning (DL) models are capable of automatically extracting complex feature representations from data with little to no human interaction [9]. Researchers have demonstrated that incorporating DL models, such as Deep Neural Networks (DNNs) and Convolutional Neural Networks (CNNs), into the development of NIDS, results in faster and more accurate network intrusion detections, when compared to traditional ML approaches [10], [11]. This research provides deeper insights into the performance of DL models in detecting and classifying network intrusion by presenting an unbiased experimental comparison of three different DL models; A DNN, a CNN and a Long Short-Term Memory Recurrent Neural Network (LSTM).

II. Related Work

The Self-Taught Learning (STL) framework has become a popular approach in NIDS research. This two-stage approach involves first learning an effective feature representation of the data using a large collection of unlabelled data, referred to as the Unsupervised Feature Learning (UFL) stage of the framework. In the second stage, the learnt feature representations are used as the input to a supervised algorithm which performs the classification task. The initial proposal of this framework, utilized a Sparse AutoEncoder (SAE) for UFL followed by a SoftMax Regression (SMR) for classification and was able to significantly improve the

978-1-6654- 9653-7/22/$31.00 © 2022 IEEE

TABLE I
KEY CONTRIBUTIONS IN THE LITERATURE
PERFORMANCE METRICS FOR FIVE-CLASS CLASSIFICATION

Author	Dataset	Model	Results
Javaid, 2016 [1]	NSL-KDD	SAE + SMR	Accuracy: 79.1% Precision: 84.0% Recall: 69.0% F1: 75.76%
Alrawashdeh, 2016 [6]	KDD Cup ‘99	DBN + LR	Accuracy: **97.9%** Recall: 97.5%
Yin, 2017 [7]	NSL-KDD	RNN	Accuracy: 81.29%
Shone, 2018 [9]	NSL-KDD	NDAE + RF	Accuracy: **85.4%** Precision: 100% Recall: 85.42% F1: 87.37%
Al-Qatf, 2018 [13]	NSL-KDD	SAE + SVM	Accuracy: 80.48% Precision: 93.92% Recall: 68.25% F1: 79.08%
Vinayakumar, 2019 [10]	NSL-KDD	DNN	Accuracy: 78.5% Precision: 81.0% Recall: 78.5% F1: 76.5%
Xiao, 2019 [11]	KDD Cup ‘99	AE + CNN	Accuracy: 94% Recall: 93%

classification performance of a SMR on the NSL-KDD dataset [1]. Following the success of the STL framework, numerous researchers have experimented with different combinations of models to perform the feature extraction and classification tasks as outlined in Table I above. The performance metrics in the table are for the five-class classification task as the records in the KDD Cup ’99 and NSL-KDD datasets are classified as either benign or as one of four attack types; User-to-Remote, Denial-of-Service , Probe or Remote-to-Local. The highest accuracy result for each dataset is emphasized in the table.

A Non-symmetric Deep AutoEncoder (NDAE) and Random Forest (RF) were combined to perform the dimensionality reduction and classification respectively [9]. Compared to utilizing a Deep Belief Network (DBN) for feature extraction [6], this approach demonstrated a 5% improvement in accuracy on the NSL-KDD dataset and improved precision, recall, F1-Score and FAR. Using a SAE for feature learning and an SVM to perform the classification task proved that along with the improvements in accuracy, the STL approach also offered dramatic reduction in model training and testing times [13]. These time reductions are imperative as timeliness is crucial when developing systems to detect network intrusions in real-world environments. Further improvements in model efficiency were illustrated by reducing dimensionality using an AutoEncoder (AE) before transforming the data into a two-dimensional matrix and using a CNN to perform further feature extraction and the classification task [11]. Due to Recurrent Neural Networks (RNNs) being developed as a means to model sequential data, utilizing these models in NIDS development allows for the incorporation of the temporal aspect of the network data and this approach has been shown to outperform a number of traditional ML approaches on the NSL-KDD dataset [7]. DNNs have also been utilized to perform both the feature extraction and classification tasks and demonstrated superior performance when compared to traditional ML models [10]. The literature, however, lacks an objective comparison of these different DL approaches. The primary disadvantage of the approaches outlined above is the utilization of the KDD Cup ’99 and NSL-KDD datasets when evaluating model performance. Network architectures have changed dramatically over the past 20 years and these older datasets no longer represent modern attack styles [4]. Researchers continue to validate their use of these older datasets with the fact that it allows them to draw comparisons with other academic literature but if future researchers continue using only these datasets to evaluate their work, their conclusions will only become more disputable [9] . In order to address the lack of an empirical comparison of DL models for network intrusion detection on newer datasets, researchers have recently begun performing comparisons of these models on the CSE-CIC-IDS2018 dataset. Their results are summarized in Table II and a brief overview of each of the comparisons is given below.

TABLE II
DEEP LEARNING MODEL PERFORMANCE IN RECENT COMPARISONS

Author	Model	CSE-CIC-IDS2018 Results
Ferrag, 2020 [14]	DNN	Accuracy: 97.281%
	RNN	Accuracy: 97.310%
	CNN	Accuracy: **97.376%**
	RBM	Accuracy: 97.280%
	DBN	Accuracy: 97.302%
	DBM	Accuracy: 97.371%
	DAE	Accuracy: 97.372%
Gamage, 2020 [15]	DNN	Accuracy: **98.38**%
	AE + DNN	Accuracy: 98.22%
	DBN + DNN	Accuracy: 98.31%
	LSTM	Accuracy: 97.60%

In the first comparison, the authors utilize the taxonomy presented by Deng and Yu [16] to classify DL approaches into two categories; deep discriminative models and generative/unsupervised models. The models they select for their comparison include a RNN, a DNN and a CNN (deep discriminative models) as well as a Deep AE, a Restricted Boltzmann Machine (RBM) , a Deep Boltzmann Machine and a DBN (generative/unsupervised models) [14]. Although their comparison includes a wide variety of models, it contains very little information about data pre-processing and does not provide details about the model architectures and hyperparameters used. A new taxonomy of DL models for network intrusion detection is proposed by the authors of the second comparison. Their taxonomy divides models into 4 categories namely; supervised instance learning, supervised sequence learning, semi-supervised instance learning and other learning paradigms. In their evaluation, a DNN is selected to represent the first category, an LSTM to represent the second and an AE as well as a DBN to represent the third [15]. The authors believe these models are representative of the different approaches for building DL models, but their evaluation is undermined by the fact that they do not include a CNN in their comparison, which the authors of the first comparison found to be the best performing model on the CSE-CIC-IDS2018 dataset.

The major downfall of both these comparisons is that the authors do not address the class imbalance present in the CSE-CIC-IDS2018 dataset. This class imbalance makes the identification of minority classes by DL models difficult and introduces bias in favour of the majority classes resulting in deceptively high performance metrics [17].

III. Methodology

The related work section above identified a number of shortfalls found in the literature. The list below outlines how each of these shortfalls were addressed in this research.

- **The use of outdated datasets:** The NSL-KDD dataset was used purely to verify that the models were implemented correctly and attained results similar to those reported in the literature. Thereafter, the models were evaluated using the modern CSE-CIC-IDS2018 dataset.
- **Not addressing the class imbalance of the CSE-CIC-IDS2018 dataset:** An undersampling technique which ensures an equal class distribution was incorporated into the preprocssing of the CSE-CIC-IDS2018 dataset.
- **Unrepresentative model selection:** In order to ensure that the comparison is representative of all categories across both taxonomies presented in the literature a DNN, a CNN, a LSTM were included in the comparison.
- **Not providing information about data preprocessing, model architectures and hyperparameters:** The data preprocessing procedure used for both the NSL-KDD and CSE-CIC-IDS2018 dataset, the model architectures and hyperparemeters are all delineated below.

A. Data

As evident in the related work section , the two most used datasets in NIDS research are the KDD Cup '99 and the NSL-KDD datasets. The KDD Cup '99 dataset was generated in 1999 from the DARPA98 network traffic which was collected over 9 weeks in raw tdpdump format [1]. The KDD Cup '99 has been used as a benchmark in NIDS research for many years but suffers from the major drawback that about 75% of the records are redundant. The NSL-KDD dataset was derived from the original KDD Cup '99 dataset and effectively solved the inherent redundant record problem [7]. It also classified the records into different difficulty levels based on the number of traditional ML algorithms that were able to correctly classify the records. The NSL-KDD dataset consists of 41 network traffic features which contain both host-based and time-based information. In order to address the lack of a modern network traffic dataset, the CSE-CIC-IDS2018 dataset was developed as a comprehensive and diverse benchmark dataset for network intrusion detection, specifically anomaly-based methods [18]. The dataset is the result of a collaborative project between the Communications Security Establishment (CSE) and the Canadian Institute of Cybersecurity(CIC). The CSE-CIC-IDS2018 dataset is the most recent network intrusion detection dataset that is publicly available, contains a wide range of attack types and consists of enough network traffic to be considered big data [17]. The dataset contains 16,233,002 instances of network traffic captured over the course of 10 days. Six different attack scenarios are represented in the dataset; Denial-of-Service (DoS), Distributed Denial-of-Service (DDoS), Brute-force, Heartbleed, Botnet and inside infiltration. The dataset consists of 80 features extracted from the raw network data by the CICFlowMeter-V3, a network traffic flow analyser [18]. As previously noted, a prevalent issue in the CSE-CIC-IDS2018 dataset is the severe class imbalance, only 17% of the dataset consits of attack records whilst the remaining 83% is benign network traffic data.

B. Data Preprocessing

The NSL-KDD dataset is provided as two separate csv files, one containing the data for model training and one containing testing data. The files contain a total of 43 columns, including the 41 features, an attack type label and a difficulty level. Due to the way the difficulty level was derived it was removed from the dataset. The 'num_outbound_calls' feature was also removed as it has a value of zero for all records and would not contribute to the classification ability of the models. The dataset originally contains 23 different categories for the attack type label which are grouped into 5 different attack classes for the classification task as outlined in Table III below. The features indicating the protocol type used in the connection, the destination service used and the status of the connection are all categorical features with 3, 70 and 11 categories respectively. These categorical features were one-hot encoded, yielding a total of 121 features, and 1 attack type class label. Although the dataset is already divided into training and testing data by the providers, 20% of the training data was used as validation data for hyperparameter tuning. Min-max normalization was used to scale all of the features in both the training data and testing data of the NSL-KDD and CSE-CIC-IDS2018 datasets.

TABLE III
NSL-KDD Class Distribution for Training and Testing Data Sets

Class	**Training Data**		**Testing Data**	
	Records	Percentage	Records	Percentage
Benign	67 343	53.458%	9711	43.076%
Dos Attack	45 927	36.458%	7460	33.091%
Probe Attack	11 656	9.253%	2421	10.739%
U2R Attack	52	0.041%	67	0.297%
R2L Attack	995	0.790%	2885	12.797%

TABLE IV
CLASS DISTRIBUTION FOR CSE-CIC-IDS2018 PARTITIONS

Partition	**Files Included**	**Attack Records**	
		Before Cleaning	After Cleaning
BruteForce	02-14-2018.csv	380 949	156 668
DoS	02-15-2018.csv, 02-16-2018.csv	654 300	506 075
Web	02-22-2018.csv, 02-23-2018.csv	928	928
Infiltration	02-28-2018.csv, 03-01-2018.csv	161 934	135 270
Botnet	03-02-2018.csv	286 191	282 310
DDoS	02-20-2018.csv, 02-21-2018.csv	1 263 933	1 246 366

The CSE-CIC-IDS2018 dataset is provided as 10 files corresponding to the 10 day period over which the network traffic was captured. As previously noted, an undersampling technique was utilized to address the heavy class imbalance present in the CSE-CIC-IDS2018 dataset. This undersampling technique involved identifying the attack class with the least number of records in the dataset and then randomly selecting the same number of records for each of the other attack classes. The least frequent attack type class in the dataset after removing duplicates and missing data was the Web Attack class with 928 records, this meant that 928 attack records were selected from each partition. Benign records were then selected from each partition according to the proportion of total benign records in the original dataset that each partition contained. 60% of the undersampled data was used for training, 20% was used for validation and 20% was used for evaluation. This split was done in a stratified manner, to ensure the distribution of the target class was similar in each subset of data.

C. Models

The initial implementation of each model was done using the model architectures and hyperparameter values reported by the authors who originally proposed the models in prior work. Further hyperparameter tuning was performed experimentally by observing the effect of varying model architectures and hyperparameters on the performance metrics of each model on the validation dataset. Table V contains a detailed breakdown of the hidden layer architectures used for each model, the number of input neurons was determined by the input features of each dataset whilst the number of neurons in the output layer coincided with the number of target classes in the dataset.

D. Evaluation Metrics

The objective of a NIDS is to obtain a high accuracy and F1-Score with a low False Alarm Rate [7]. These three indicators are the most frequently used performance metrics when evaluating DL-based NIDS and are therefore the primary evaluation metrics used to empirically evaluate and compare models in this research.

TABLE V
HIDDEN LAYER ARCHITECTURE FOR MODELS

Model	**Hidden Layer Architecture**
DNN	Fully Connected - 64 Neurons Batch Normalization Dropout - $p = 0.2$ Fully Connected - 32 Neurons Batch Normalization Dropout - $p = 0.2$ Fully Connected - 16 Neurons Batch Normalization Dropout - $p = 0.2$
CNN	Convolution - 8 filters , 2×2 kernel Batch Normalization MaxPool - 2×2 pool size Convolution - 16 filters , 2×2 kernel MaxPool - 2×2 pool size Dropout - $p = 0.2$ Fully Connected - 64 Neurons
LSTM	Fully Connected LSTM- 64 Neurons Batch Normalization Dropout - $p = 0.2$ Fully Connected LSTM- 32 Neurons Batch Normalization Dropout - $p = 0.2$

E. Software and Hardware

In order to ensure a fair comparison of the different DL models, each model was constructed, trained, tested and evaluated on the same computing device using the same software frameworks. Model construction was done using Python with the TensorFlow and Keras frameworks. A 13-inch Apple M1 MacBook Pro with 8GB RAM was used to carry out the experimental evaluation.

IV. RESULTS

A. Verification of Model Implementation

To verify the implementation of the models used in this comparison, the results attained on the benchmark dataset, the NSL-KDD, in prior works were compared to the results achieved in this evaluation. As evident in Table VI, the classification performance of these models were very similar to the results found in the literature (within 3%). These minor disparities can be attributed to differences in preprocessing techniques and the variance associated with training neural networks.

B. Comparing Model Classification Performance

In order to account for any variance in the data splitting and model training processes, each model was trained and evaluated five times, with the reported metrics being the mean of the five iterations and the error bars on the plots in this section representing a single standard deviation, calculated over the five runs. As evident in Figure 1, all three DL models were able to outperform the classification performance of the traditional ML model, the SVM. Although the DNN was able

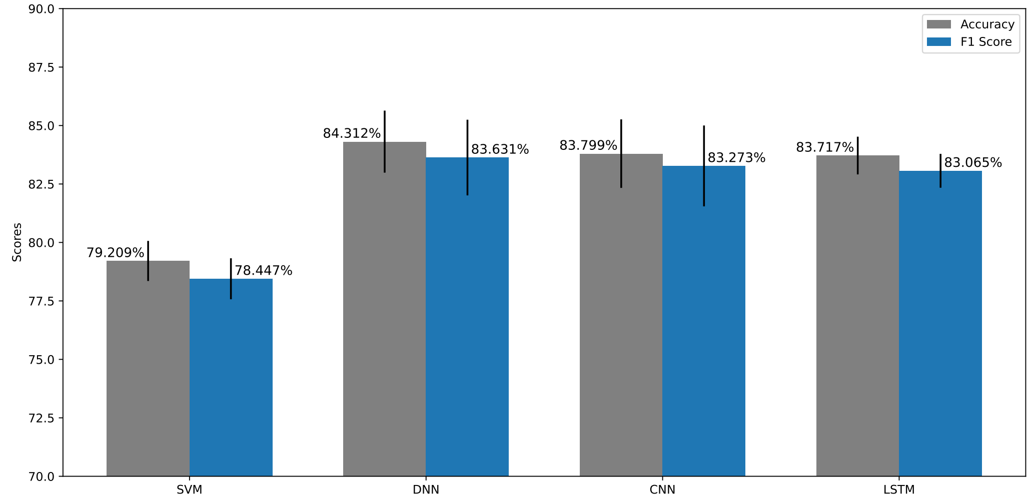

Fig. 1. Model classification performance on undersampled CSE-CIC-IDS2018 dataset, results averaged over 5 iterations with error bars indicating 1 standard deviation.

TABLE VI
NSL-KDD FIVE CLASS CLASSIFICATION ACCURACY COMPARISON WITH PRIOR WORK

Model	Prior Work	This Work
SVM	76.761% [13]	74.778%
DNN	78.5% [10]	76.127%
CNN	NA	74.101%
LSTM	77.26% [15]	76.132%

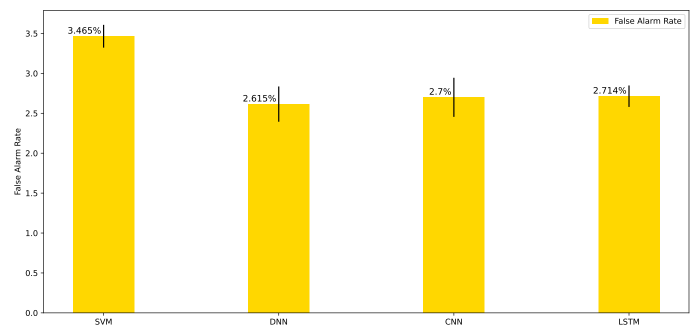

Fig. 2. False Alarm Rate of models on undersampled CSE-CIC-IDS2018 dataset, results averaged over 5 iterations with error bars indicating 1 standard deviation.

to attain the highest average accuracy and F1-Score , the LSTM achieved the lowest variance across multiple runs. By reducing the number of neurons from hidden layer to hidden layer in the DNN, the model is able to learn complex features and extract better representations of the data which in turn can be used to make more accurate classifications. The low variance of the LSTM model is a result of the feedback loop incorporated in the architecture of LSTMs which is capable of memorizing previous information and applying it to the current output. This allows it to learn long term dependencies in the network data and reduce the variability of the run-to-run classification performance. The transformation of the network traffic data into two-dimensional images and the utilization of a CNN to classify these image data does not appear to improve classification performance. Due to the fact that the network traffic data does not inherently contain any spatial information, the positional relation of the features in the two-dimensional form does not appear to provide the model with any useful additional information resulting in similar performance to the other neural networks.

Once again in Figure 2, it can be observed that all three of the DL models were able to reduce the FAR on the undersampled CSE-CIC-IDS2018 dataset, when compared to the shallow learning baseline. The lowest FAR is attained by the DNN whilst the lowest variance is achieved by the LSTM, which coincides with the observations made above and reaffirms the corresponding interpretations.

V. Discussion and Conclusion

The rapid development and deployment of network-based technologies has led to an increase in the number of cyberattacks that occur. Network Intrusion Detection Systems are a vital cybersecurity tool which aim to identify potential threats and prevent these attacks by monitoring network traffic. Recently, the incorporation of DL algorithms into the development of these systems has led to a drastic increase in effectiveness and efficiency. Particularly, the hybrid STL approach of using an unsupervised model to reduce the dimensionality of the data through feature extraction, followed by a supervised model to perform the classification has gained significant traction in the literature. Alternative approaches which incorporate the temporal aspect of network data have also shown promising results. These novel systems have, however, been evaluated using the outdated KDD Cup ’99 and NSL-KDD datasets which are not representative of modern network traffic. The utilization of these legacy datasets allows researchers to draw comparisons with other academic literature but leads to unrealistic results and disputable conclusions. Researchers who have begun evaluating these DL approaches on the more modern CSE-CIC-IDS2018 dataset do not address the severe class imbalance in the data which causes biased performance metrics. By addressing this class imbalance, and performing a comparative analysis of the classification ability and model efficiency of a DNN, a CNN and a LSTM on the balanced CSE-CIC-IDS2018 dataset, this research addresses the lack of a fair evaluation of DL models found in the literature by providing insights into the performance of these models in classifying modern network traffic data. All three DL approaches were able to out-perfrom the shallow learning baseline, a SVM. Although the LSTM was the most reliable model, the DNN boasted superior classification ability. Potential avenues for future research include the comparison of utilizing different dimensionality reduction techniques such as AEs, RBMs and DBNs to perform feature extraction in an STL approach. Reinforcement Learning and Transfer Learning have also demonstrated impressive results in recent NIDS research [20], [21]. Comparing these approaches with the approaches evaluated in this research would be a beneficial contribution to the NIDS research community.

VI. Acknowledgements

This work is based on the research supported in part by the National Research Foundation of South Africa (Grant number: 121835).

References

[1] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, “A deep learning approach for network intrusion detection system”, in Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS), pp. 21–26, 2016.

[2] W. Wang et al., “HAST-IDS: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection”, IEEE Access, vol 6, pp. 1792–1806, 2017.

[3] B. Yan and G. Han, ”Effective Feature Extraction via Stacked Sparse Autoencoder to Improve Intrusion Detection System,” in IEEE Access, vol. 6, pp. 41238-41248, 2018, doi: 10.1109/ACCESS.2018.2858277.

[4] Y. Dong, R. Wang, and J. He, “Real-Time Network Intrusion Detection System Based on Deep Learning”, in 2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS), pp. 1–4, 2019.

[5] N. Marir, H. Wang, G. Feng, B. Li, and M. Jia, “Distributed abnormal behavior detection approach based on deep belief network and ensemble svm using spark”, IEEE Access, vol 6, pp. 59657–59671, 2018.

[6] K. Alrawashdeh and C. Purdy, “Toward an online anomaly intrusion detection system based on deep learning”, in 2016 15th IEEE international conference on machine learning and applications (ICMLA), pp. 195–200, 2016.

[7] C. Yin, Y. Zhu, J. Fei, and X. He, “A deep learning approach for intrusion detection using recurrent neural networks”, IEEE Access, vol 5, pp. 21954–21961, 2017.

[8] M. S. Elsayed, N.-A. Le-Khac, S. Dev, and A. D. Jurcut, “Ddosnet: A deep-learning model for detecting network attacks”, in 2020 IEEE 21st International Symposium on” A World of Wireless, Mobile and Multimedia Networks”(WoWMoM), pp. 391–396, 2020.

[9] 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.

[10] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, “Deep learning approach for intelligent intrusion detection system”, IEEE Access, vol 7, pp 41525–41550, 2019.

[11] Y. Xiao, C. Xing, T. Zhang, and Z. Zhao, “An intrusion detection model based on feature reduction and convolutional neural networks”, IEEE Access, vol 7, pp 42210–42219, 2019.

[12] J. Van Hulse, T. M. Khoshgoftaar, and A. Napolitano, “An empirical comparison of repetitive undersampling techniques”, in 2009 IEEE international conference on information reuse integration, pp 29–34, 2009.

[13] M. Al-Qatf, Y. Lasheng, M. Al-Habib, and K. Al-Sabahi, “Deep learning approach combining sparse autoencoder with SVM for network intrusion detection”, IEEE Access, vol 6, pp 52843–52856, 2018.

[14] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, “Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study”, Journal of Information Security and Applications, vol 50, pp 102419, 2020.

[15] S. Gamage and J. Samarabandu, “Deep learning methods in network intrusion detection: A survey and an objective comparison”, Journal of Network and Computer Applications, vol 169, pp 102767, 2020.

[16] L. Deng and D. Yu, “Deep learning: methods and applications”, Foundations and trends in signal processing, vol 7, no 3–4, pp 197–387, 2014.

[17] J. L. Leevy and T. M. Khoshgoftaar, “A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 Big Data”, Journal of Big Data, vol 7, no 1, pp 1–19, 2020.

[18] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization”, in ICISSp, pp 108–116, 2018.

[19] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting”, The journal of machine learning research, vol 15, no 1, pp 1929–1958, 2014.

[20] Y.-F. Hsu and M. Matsuoka, “A Deep Reinforcement Learning Approach for Anomaly Network Intrusion Detection System”, in 2020 IEEE 9th International Conference on Cloud Networking (CloudNet), pp 1–6, 2020.

[21] J. Li, W. Wu, and D. Xue, “An intrusion detection method based on active transfer learning”, Intelligent Data Analysis, vol 24, no 2, pp 363–383, 2020.