A Long Short-Term Memory Enabled Framework for DDoS Detection

1st Xiaoyu Liang
Department of Computer Science
University of Pittsburgh
Pittsburgh, USA
xil160@pitt.edu

2nd Taieb Znati
Department of Computer Science
University of Pittsburgh
Pittsburgh, USA
znati@pitt.edu

***Abstract*—The proliferation of attack-for-hire services, coupled with the advent of Internet of Things (IoT)-enabled botnets, is driving the increase of the frequency and intensity of Distributed Denial of Services (DDoS) attacks, at an alarming rate. Inspired by the success of machine learning in a variety of fields and domain applications, numerous intelligent schemes have been proposed to effectively defend against and mitigate the impact of these attacks. Traditional machine learning methods, however, are limited by the use of an expensive and error-prone feature engineering process. Feature engineering is fundamental to the application of machine learning, and is both difficult and expensive. Furthermore, the ability of these schemes to successfully detect previously unknown attacks is limited. To address these limitations, a novel DDoS detection scheme, based on Long Short-Term Memory (LSTM), is proposed. The basic tenet of the LSTM scheme is its ability to distinguish between attack and legitimate flows by only examining a relatively small number of a network flow packets. The performance evaluation results show that the LSTM-based scheme successfully learns the complex flow-level feature representations embedded in raw input traffic. Furthermore, the results show that the scheme performs better than other approaches that use sophisticated flow-level statistical features. Lastly, the results show the ability of the proposed scheme to accurately capture the dynamic behaviors of unknown network traffic exceeds that of traditional machine learning methods.**

***Index Terms*—DDoS Detection, Deep Learning, LSTM**

I. Introduction

The main objective of a DDoS attack is to make computing and network resources unavailable to their intended users. With the proliferation of attack-for-hire service and the advent of IoT-enabled botnets, DDoS attacks have been growing dramatically in volume, frequency, sophistication and impact [1]. On October 2016, Mirai, the infamous IoT botnet, caused a disruption of multiple major websites via a series of massive DDoS attacks [2]. Later that year, Mirai botnet source code was released to the public, since then multiple DDoS attacks have been lunched with increased severity [3]. Despite significant advances in the state-of-the-art of system and network security, defending against DDoS attacks remains one of the most challenging problems [4].

Traditional schemes use a number of approaches to identify anomalous behaviors and detect intrusion events, including utilizing stochastic analysis to monitor network traffic flows and exploiting the entropy of network traffic [5], [6]. The challenge these methods face stems from the need to reliably define a normal profile and accurately differentiate between normal and anomalous behaviors [7]. Additionally, the performance of these detection schemes can be impacted severely by the location where the detection scheme is deployed [8].

Recently, the trend to mitigate the impact of DDoS attacks is to incorporate "intelligence" into the defense architecture, leveraging machine learning techniques to classify and detect malicious traffic. These techniques are capable to intelligently learn the underlying data attributes without the need to explicitly describe normal and malicious activities, thereby overcoming the limitations of traditional schemes. However, classic machine learning techniques struggle to capture the evolving nature of DDoS attacks. We believe that a significant part of the problem stems from the failure of hand-crafted feature engineering to represent traffic behaviors.

A machine learning based traffic classification workflow requires a two-stepped process for feature engineering. In the first step, an appropriate set of features is extracted to characterize the signature of the collected data. In the second step, a feature selection algorithm is applied to eliminated irrelevant features [9]. This process is not only labour intensive, but also prone to errors. Firstly, generating a feature set that captures previously unseen attacks behavior is challenging, and often involves deep understanding of the network traffic behaviors and characteristics. Secondly, most feature selection algorithms are based upon the strong assumption that features are independent from each other [10]–[12], erroneously ignoring the intrinsic temporal and spatial correlations between the features. Lastly, training and feature selection, which are treated as two separate phases of the classification workflow, cannot be jointly optimized, thereby hindering the overall performance of the detection scheme. Deep learning holds promise for addressing these key limitations.

Deep learning methods stack multiple layers of non-linear transformation hierarchically, so that these methods can automatically extract complex representations hidden inside the raw input [13]. For the task of classification, with ascending the layers, representations that are important for class discrimination are amplified, and the irrelevant representations are suppressed [14]. This process inherently embeds feature selection. As a result, both the transformations of the raw

978-1-7281-0962-6/19/$31.00 ©2019 IEEE

data into the distinctive representations and the classification of these data into legitimate and malicious traffic are optimized jointly within the training process.

In this paper, we propose a novel deep learning DDoS detection scheme, which only uses packet header information as input and does not require feature engineering. At the core of the scheme is Long Short-Term Memory (LSTM), a Recurrent Neural Network (RNN) architecture used to learn the network traffic behavior, and distinguish attack network flows from legitimate flows, by examining a relatively small number of packets from each flow. The main contributions of this paper are summarized as follows:

- The design of a LSTM-based scheme for DDoS detection. The proposed scheme, which obviates the need for feature engineering, successfully learns the complex flow-level feature representations embedded in raw input traffic and examines only a short sequence of packets to detect DDoS attacks.
- The design of an evaluation framework to assess the performance of the proposed scheme and conduct a comparative analysis of its performance relative to other schemes. The experimental results show that:
  - Using raw network traffic trace, the proposed scheme has the ability to automatically learn the traffic flows temporal and spatial relationships to successfully detect DDoS attacks.
  - The scheme achieves higher performance than classic machine learning techniques, which are trained using sophisticated flow-level statistical features.
  - The use of LSTM architecture, at the core of the proposed scheme, significantly improve its ability to capture the dynamic behaviors of attack traffic, in comparison with classic machine learning methods.
  - Increasing the number of packets used to learn the traffic dynamic behaviors does not necessarily improve the scheme's ability to detect DDoS traffic. A relatively small number of packets is sufficient to learn dynamic behaviors.

The rest of the paper is organized as follows. Section II reviews the related works. In section III, we describe the proposed deep learning LSTM-based DDoS detection scheme. In Section IV, we describe the experimental framework, the data preprocessing procedure and introduce the evaluation metrics. In section V, we present three performance evaluation and sensitivity analysis experiments and discuss the outcome of these experiments. In Section VI, we provide the conclusion of this work and explore future directions of research.

II. Related Works

Deep learning, a broad family of machine learning techniques, has been successfully applied to solve challenging problems in a number of fields, including computer vision, social network filtering and video games, natural language processing and machine translation, healthcare and bioinformatics, medical image analysis and drug discovery. Its application to DDoS detection, however, has been limited. In this section, we discuss various recent notable works in this field. Several works utilize autoencoder, an unsurpervised learning technique, to discover non-linear representations from the input data, and then they apply a classification method to distinguish malicious traffic from legitimate traffic [15]–[18].

In [15], the authors combine an auto-encoder with a soft-max regression layer for network intrusion detection. The proposed scheme exhibits promising performance in both binary and multi-class classification task. The authors then extend their work by stacking two auto-encoders to detect DDoS attacks [16]. In [17], the authors stack two autoencoders to learn the complex relationships among features. They claim that soft-max layer is weaker than classic classifiers, and combine the stacked auto-encoder with a Random Forest classifier for intrusion detection. In [18], the authors utilize autoencoder not only for feature learning but also to reduce the number of random variables under consideration. Instead of connecting a classifier with the output layer of the autoencoder, a hidden layer, which represents the compressed features, is used as input for the classifier. A Support Vector Machine (SVM) is used as the classifier. The authors claim that SVM outperforms all other classic classifiers. Although these proposed schemes successfully overcome the difficulty of feature selection, they do not address the challenges of feature extraction.

In addition to autoencoders, RNN is a popular choice for intrusion detection [19]–[23]. RNN, an extension of a conventional neural network, is widely used for modeling sequential data to solve time-series problems. Most schemes apply RNN models to well formatted datasets, such as KDD99 and NSL-KDD, which use hand-crafted flow-level feature engineering. In these datasets, each record corresponds to a specific network flow, and is represented by a set of attributes, such as duration, number of packets, number of bytes, *etc*. Most proposed schemes treat the set of attributes for each flow as a sequential data, which is then used as input for the RNN models [19]–[22]. To improve the classification accuracy and other evaluation metrics, these works apply different types of RNN models. A simple RNN is adopted in [21]. In [19], [20], the authors use LSTM RNN, and focus on selecting a minimal subset of features. In [22], the authors used Gated Recurrent Unit (GRU) RNN. Although the performance of these proposed schemes surpasses traditional machine learning methods, they fail to address the challenge of feature extraction. More importantly, assuming temporal order among features in RNN models is questionable. The advantage of RNNs in modeling sequential data is that they have a memory cell which can remember data received earlier and capture the temporal dependency among data. Although features are not independent of each other, preserving order among features is not necessary.

In [23], the authors propose a RNN-based model, referred to as *DeepDefense*. Unlike previously discussed methods, the model does not use flow-level statistical features. Instead, the authors extracted 20 features from packet headers and applied sliding windows to separate continues network traffic into

sequences of network packets. For a given sequence of packets, the model classifies the last packet either as a legitimate or attack traffic. Their proposed deep learning model achieves lower error rate than traditional machine learning techniques. However, reliance on packet-by-packet inspection may not be practical in a real-world setting.

III. Proposed Detection Scheme

Feature engineering is the process of using domain knowledge of the network traffic data to create features. Such a process is critical to network traffic behavior analysis and intrusion detection. A network flow is usually characterized as a sequence of packets that share the same $\langle$ *source IP*, *source Port*, *destination IP*, *destination Port*, *Protocol* $\rangle$. Given the aggregated packets, a variety of features can be derived. However, the manual process of handcrafted flow-level feature engineering is not only costly and time-consuming, but may also entail the loss of the temporal information inherited in the original sequence of packets. In this section, we propose a deep learning DDoS detection scheme, which obviates the process of handcrafted feature engineering, and learns the network traffic behavior directly from a relatively small sequence of packets. The detection scheme takes a small number of packets of a network flow as input and outputs whether the flow is or is not malicious.

The ability of RNN to learn non-linear representations from sequential data makes it a natural fit to determine if a sequence of packets is or is not malicious. RNN extends the traditional feed-forward neural network by introducing a directional loop, so that the sequential dependence between the current packet and the historical information carried by previously observed packets is preserved. We use LSTM in our detection model, which is one of the most popular and efficient variants of RNN [24]. Figure 1 depicts the four-layered architecture of the proposed model, namely two LSTM layers, a dropout layer, and a fully connected layer. The LSTM layers learn both temporal and spacial representations from the input sequential data. Each unit in the LSTM layer contains three gates, namely *input*, *forget* and *output*, which work together to learn the transformation of input values, the relevant information from previously observed data, and the non-linear representation of the current state during training. The dropout layer masks a random fraction of the input units at each update step, while training the network. It adds noises to the LSTM layer, to avoid over-fitting and improve the robustness of the trained model. A fully connected layer is used for classification.

Formally, a network flow, which consists of N network packets, can be described as a sequence: $F = \{p^{(1)}, p^{(2)}, ..., p^{(i)}, ..., p^{(N)}\}$, where $p^{(i)} (1 \leq i \leq N)$ represents the ith packet of F. Each packet $p^{(i)} \in \mathbb{R}^m$ is an m-dimensional vector contains stored information $p^{(i)} = \{p_1^{(i)}, p_2^{(i)}, ..., p_m^{(i)}\}$. In our proposed detection scheme, the stored information is from packets headers, including: *source Port*, *destination Port*, *packet length*, *Time To Live*, *FIN*, *SYN*, *RST*, *PSH*, *ACK*, *URG*, *ECE*, and *CWR*. To adequately explore the temporal information, we also include three temporal features: *time past since last packet*, *time past since the first packet*, and *average time interval between consecutive packets*.

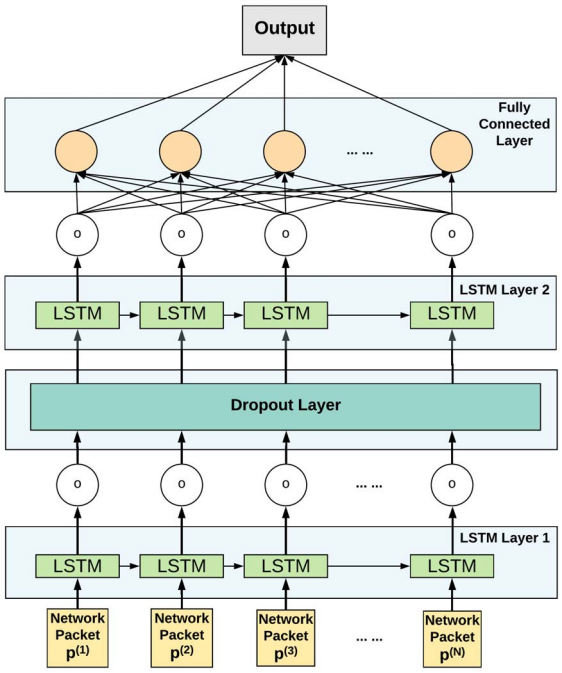

Fig. 1: The Architecture of Proposed DDoS Detection Model

For each network flow, F, a subsequence of n packets, $S = \{p^{(j)}, ..., p^{(j+n-1)}\}, S \subset F$, is inspected. Using S, the model classifies the traffic flow as either legitimate or malicious. The value of n is pre-defined. If a flow does not have enough packets, S will be padded with fake packets. A fake packet is an m-dimensional vector with values of zeros. If a flow has more than n packets, only the first n packets are examined. The remaining packets are discarded. To reduce the detection delay, a time window threshold is applied. The network flow is examined when either n packets are observed or the time window threshold is reached. The sensitivity of the scheme to n is discussed in section IV.

IV. Experimental Framework

In this section, we describe the datasets and evaluation metrics used in the experimental framework. We also describe the data preprocessing procedure.

A. Dataset

The experimental evaluation framework uses a widely accepted benchmark datasets, CICIDS 2017 [25]. It was published by the Canadian Institute of Cybersecurity in 2017, and contains realistic background network traffic and a variety of attack traffic. The datasets cover five days of network traffic, two of which have DoS and DDoS attacks. We use these two days' traffic, denoted as Wednesday and Friday, as the evaluation benchmark. Table I presents the detailed attack types for each traffic collection.

Attack traffic in Wednesday was generated by four different tools. Three of them generated low-bandwidth application layer attack. These attacks require little bandwidth, and very stealthy. They aim to keeping the HTTP connections as long as possible using similar but different strategies. Wednesday's traffic also contains a volumetric attack, which was generated by a tool named "Hulk". Hulk can flood the victim with huge HTTP GET requests from a single device. The packet header values of these requests are generated randomly to confuse the victim, and make it hard to be detected. Friday's attack was generated by Low Orbit Ion Cannon (LOIC). LOIC floods targeted server using junk TCP, UDP and HTTP GET requests through numerous attacking devices. Although classified as volumetric attack, the traffic behavior is more similar to low bandwidth application attack than traffic generated by Hulk, from the perspective of a single flow.

B. Data Preprocessing

CICIDS 2017 datasets provide well formatted data files. In these files, each network flow is characterized by more than 80 statistical features, and associated with a label indicating whether it is a malicious flow. In addition to the well formatted network flow files, the raw trace files (in pcap format) are also provided. Since the raw data is not labeled, we need to reverse engineer the process to find the corresponding set of packets for each network flow. We carefully check the timestamp, the number of forward and backward packets, the time duration, to make sure the mapping is correct and precise. However, the timestamp granularity is not fine enough to find all the mappings. We discard the network flows and packets that we did not find exact matches that satisfy our criteria.

C. Evaluation Metrics

A successful DDoS detection requires correctly identifying attacks, while minimizing the number of false alarms. In this study, we adopt the evaluation metrics widely used to assess the performance of DDoS detection schemes. These metrics are: Precision, Recall, and F1-Measure. TP, TN, FP and FN stand for True Positive, True Negative, False Positive and False Negative, respectively. We label attack traffic as positive.

$$\begin{aligned} \textit{Precision}(P) &= \frac{TP}{TP+FP} \\ \textit{Recall}(R) &= \frac{TP}{TP+FN} \\ \textit{F1-Score}(F1) &= 2 \times \frac{P \times R}{P+R} \end{aligned}$$

TABLE I: Attacks in the Experiment Dataset

Traffic Collection	Attack Generated Tools	Brief Description of Attacks
CICIDS 2017 Wednesday	HTTP Unbearable Load King (Hulk)	Volumetric Attack. Generate volumes of HTTP GET requests with randomly generated header values.
	slowloris	Low-Bandwidth Application Layer Attack. Open multiple HTTP connections. Continuously send partial HTTP requests
	slowHTTP	Low-Bandwidth Application Layer Attack. Send HTTP requests in pieces slowly, one at a time to a Web server.
	Golden Eye	Low-Bandwidth Application Layer Attack. Open multiple HTTP connections, and use "keep alive" packets.
CICIDS 2017 Friday	Low Orbit Ion Cannon (LOIC)	Volumetric Attack. Open multiple HTTP connections and continuously send HTTP request messages

V. Experiments and Results

In this section, we describe three experiments and discuss the results. Experiment 1 follows the standard evaluation process that can be found in most machine learning papers. In Experiment 2, we assess the capability of our proposed scheme to capture the dynamic attack traffic behaviors. In Experiment 3, we evaluate the impact of the value of n on our proposed scheme's performance.

A. Experiment 1 – A Standard Evaluation Experiment

The goal of experiment 1 is to carry out a comparative analysis of the performance of the proposed scheme, which only use the raw packet header information, and the traditional machine learning methods, which rely on manually selected sophisticated features. In this experiment, we split the benchmark datasets into three parts: training, validation and testing, which contains 70%, 10% and 20% of the original data, respectively. We applied cross validation to optimize the hyperparameters for each model. The training and testing process is applied for Wednesday and Friday's traffic collection, separately. To decided the value of n for the proposed LSTM scheme, we analyze the number of packets associated with each flow in the training datasets. We choose 10 as the value of n, which is the round up value of the median. Hence, the LSTM model examines a sequence of 10 packets from each flow and then classifies the flow as either legitimate or malicious.

The proposed LSTM detection model is compared with the state-of-the-art traditional machine learning models, including Decision Tree (DT), Artificial Neural Networks (ANN) and Support Vector Machine (SVM). Results are represented in Table II. It shows that both traditional machine learning methods and our proposed LSTM scheme are capable to achieve nearly perfect performance on the testing case. It is worth noting that without using flow-level statistical features, the proposed LSTM scheme can achieve not only competitive, but slightly better performance in terms of all evaluation metrics.

On one hand, the result confirms that the capability of machine learning techniques to recognize patterns is evident. Taking a closer look at the results, it finds that except for SVM, all other methods only mis-classified less than 20 flows for Friday's dataset. It is impressive, especially considering that tens of thousands of flows are contained in the test data. On the other hand, the results does not benefit our community any further to solve the practical problem by splitting the

same dataset into training and testing. In this experiment, the training and testing set are generated by a same (set of) tools. We can assume all testing samples are sufficiently represented in the training set, so the learning models can capture the inherited patterns in the testing set successfully. However, in the real-world detection scenario, the unseen attack is usually under-represented, if not un-represented at all, in the available training data. The challenge in DDoS detection is to capture the dynamic traffic behavior so that the similar attacking strategies can be identified in the future.

B. Experiment 2 – Testing on Unknown Dataset

In this experiment, we evaluate the proposed scheme's capability to capture the dynamic attack traffic behaviors. Specifically, we train the models on Wednesday and Friday's datasets separately, and test them on Friday and Wednesday's datasets, respectively. For convenience, we denote training on Wednesday's datasets and testing on Friday's datasets as "WeTrFrTest", and the other task as "WeTrFrTest". As discussed in section IV, attacks in these two days' datasets are generated by different tools but with similar attacking strategies. From the perspective of a single flow, the embedded attacking behavior should be similar. A successful detection scheme is expected to capture the commonalities.

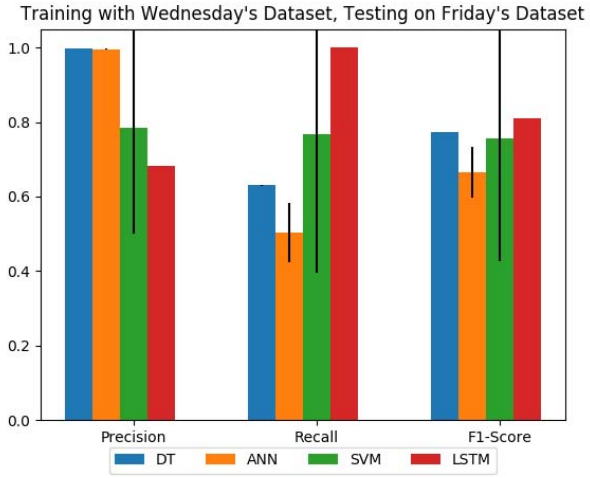

Fig. 2: Training with Wednesday's Dataset, and testing on Friday's Dataset

TABLE II: Experiment 1 Results

Models	Wednesday			Friday		
	P	R	F1	P	R	F1
DT	**0.9986**	0.9985	0.9985	**0.9998**	0.9991	0.9995
ANN	0.9971	0.9992	0.9982	0.9996	0.9998	0.9997
SVM	0.9505	0.5135	0.5925	0.8818	0.4543	0.5997
LSTM	0.9995	**0.9997**	**0.9991**	**0.9998**	**1**	**0.9999**

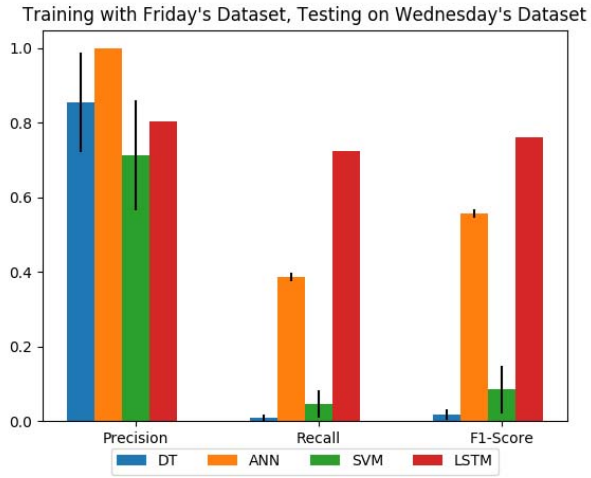

Fig. 3: Training with Friday's Dataset, and testing on Wednesday's Dataset

Figures 2 and 3 represent the results for "WeTrFrTest" and "FrTrWeTest" respectively. The high variance presented by the performance of SVM shows that it is not a stable and trustworthy detection scheme in this evaluated scenario. Thus, we leave SVM out in the following discussions.

Training with Wednesday's dataset, the traditional machine learning methods intend to label more Friday's traffic as legitimate rather than malicious, shown as high precision scores and low recall scores. Differently, our proposed LSTM model intends to label more traffic as malicious rather than legitimate, causing a high recall score but a lower precision score. Balancing between the precision and recall, LSTM model slightly outperforms DT and ANN, shown by the highest F1 Score.

Training with Friday's dataset, the traditional machine learning methods fail to detect the attack traffic in Wednesday's dataset. The extremely low recall score presented by DT reveals that it classifies most traffic flows into the category of legitimate. ANN performs better than DT, but the similar conclusion can be drawn from the result. The proposed LSTM performs equally good in all measurements, and it shows the significantly improvements in both recall and F1 score.

C. Experiment 3 – The Impact of Different Values of n

To classify network traffic, our proposed detection scheme needs to examine n packets of a network flow. In this experiment, we study the impact of different values of n on the performance. We train the proposed models with different values of $n \in \{3, 5, 10, 20, 30, 40, 50\}$.

Examining only the first 3 and 5 packets, the model lose the capability to distinguish attack and legitimate traffic. It simply label all traffic as attack traffic. Analyzing the training data, we observe that flows, which have large number of packets, are usually legitimate traffic flows. The model may have learned this specific feature, and simply make the decision accordingly.

Table III and IV present the results for "WeTrFrTest" and "FrTrWeTest", respectively. From the table, it is observed that allowing the model to examine more packets for each flow, with increasing n values, does not necessarily improve the performance. In Table IV, examining 50 packets significantly degrade the scheme's performance. Additionally, examining larger number of packets increases the standard deviation of all measured metrics, which indicates the consistency of the performance is also degraded by a larger value of n.

It seems to benefit the detection scheme by examining higher number of packets from each flow, since more data is provided for the model to learn traffic behaviors. However, if network flows are mostly short, such as in our benchmark, then the short flows will be padded with zeros. These padding values may confuse the system and cause the performance degradation.

TABLE III: Evaluating the Impact of Different Values of n

n	Ave* P	Ave R	Ave F1	STD* P	STD R	STD F1
10	0.6823	0.9999	0.8111	1.0584	1.74e-05	7.86e-6
20	0.7125	0.9999	0.8321	0.0005	1.46e-5	0.0003
30	0.7205	0.9981	0.8368	0.0143	0.0041	0.0081
40	0.6271	0.8752	0.7283	0.2241	0.3294	0.2729
50	0.7321	0.9949	0.8432	0.0259	0.0064	0.0146

*Ave are average values; *STD is the standard deviation

TABLE IV: Evaluating the Impact of Different Values of n

n	Ave P	Ave R	Ave F1	STD P	STD R	STD F1
10	0.8026	0.7235	0.7610	9.04e-6	2.87e-5	1.42e-5
20	0.8889	0.6010	0.7171	4.49e-6	1.01e-5	6.24e-6
30	0.8870	0.5957	0.7132	0.0007	0.0138	0.0103
40	0.8889	0.6010	0.7171	1.20e-5	7.87e-6	7.17e-6
50	0.8077	0.2821	0.2920	0.0127	0.3636	0.3756

VI. Conclusions and Future Work

A LSTM-based approach for DDoS detection scheme is proposed. The scheme avoids manual feature engineering, thereby addressing a significant shortcoming of classic machine learning methods. The ability of the scheme to automatically learn complex representations to successfully classify legitimate and attack traffic flows is empirically confirmed. Using unknown traffic, the results show that the proposed scheme outperforms traditional machine learning methods. Future work will focus on the design and deployment of the proposed scheme proof-of-concept in a distributed environment of Software Defined Network, OpnenFlow-enabled controllers. The potential of the scheme as a building block for a decentralized adaptive DDoS detection framework will be explored.

References

[1] A. Lohachab and B. Karambir, "Critical Analysis of DDoS An Emerging Security Threat over IoT Networks," *Journal of Communications and Information Networks*, vol. 3, no. 3, pp. 57–78, 2018.

[2] M. Antonakakis, T. April, M. Bailey, M. Bernhard *et al.*, "Understanding the Mirai Botnet," in *26th {USENIX} Security Symposium ({USENIX} Security 17)*, 2017, pp. 1093–1110.

[3] krebsonsecurity, "New Mirai Worm Knocks 900K Germans Offline," Webpage. [Online]. Available: krebsonsecurity.com/2016/11/new-mirai-worm-knocks-900k-germans-offline/

[4] T. Ibragimov *et al.*, "DDoS attacks in Q2 2018," Webpage. [Online]. Available: https://securelist.com/ddos-report-in-q2-2018/86537/

[5] L. Feinstein, D. Schnackenberg, R. Balupari, and D. Kindred, "Statistical Approaches to DDoS Attack Detection and Response," in *Proceedings DARPA information survivability conference and exposition*, vol. 1. IEEE, 2003, pp. 303–314.

[6] S. Oshima, T. Nakashima, and T. Sueyoshi, "Early DoS/DDoS Detection Method using Short-term Statistics," in *2010 International Conference on Complex, Intelligent and Software Intensive Systems*. IEEE, 2010, pp. 168–173.

[7] R. Sommer and V. Paxson, "Outside the Closed World: On using Machine Learning for Network Intrusion Detection," in *2010 IEEE symposium on security and privacy*. IEEE, 2010, pp. 305–316.

[8] X. Liang and T. Znati, "An Empirical Study of Intelligent Approaches to DDoS Detection in Large Scale Networks," in *2019 International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 2019, pp. 821–827.

[9] A. L. Buczak and E. Guven, "A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.

[10] H. Liu and L. Yu, "Toward Integrating Feature Selection Algorithms for Classification and Clustering," *IEEE Transactions on Knowledge & Data Engineering*, no. 4, pp. 491–502, 2005.

[11] J. Tang, S. Alelyani, and H. Liu, "Feature Selection for Classification: A Review," *Data classification: algorithms and applications*, p. 37, 2014.

[12] J. Li and H. Liu, "Challenges of Feature Selection for Big Data Analytics," *IEEE Intelligent Systems*, vol. 32, no. 2, pp. 9–15, 2017.

[13] Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[14] M. D. Zeiler and R. Fergus, "Visualizing and Understanding Convolutional Networks," in *European conference on computer vision*. Springer, 2014, pp. 818–833.

[15] 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*. ICST, 2016, pp. 21–26.

[16] Q. Niyaz, W. Sun, and A. Y. Javaid, "A Deep Learning based DDoS Detection System in Software-Defined Networking (SDN)," *arXiv preprint arXiv:1611.07400*, 2016.

[17] 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.

[18] 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. 52 843–52 856, 2018.

[19] R. C. Staudemeyer, "Applying Long Short-Term Memory Recurrent Neural Networks to Intrusion Detection," *South African Computer Journal*, vol. 56, no. 1, pp. 136–154, 2015.

[20] 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, 2016, pp. 1–5.

[21] 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. 21 954–21 961, 2017.

[22] T. A. Tang, L. Mhamdi *et al.*, "Deep Recurrent Neural Network for Intrusion Detection in SDN-based Networks," in *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*. IEEE, 2018, pp. 202–206.

[23] X. Yuan, C. Li, and X. Li, "DeepDefense: Identifying DDoS Attack via Deep Learning," in *2017 IEEE International Conference on Smart Computing (SMARTCOMP)*. IEEE, 2017, pp. 1–8.

[24] H. Salehinejad, S. Sankar, J. Barfett, E. Colak, and S. Valaee, "Recent Advances in Recurrent Neural Networks," *arXiv preprint arXiv:1801.01078*, 2017.

[25] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," in *ICISSP*, 2018, pp. 108–116.