Flow-based Network Intrusion Detection Based on BERT Masked Language Model

Anonymous Author(s)

ABSTRACT

A Network Intrusion Detection System (NIDS) is an important tool that identifies potential threats to a network. Recently, different flow-based NIDS designs utilizing Machine Learning (ML) algorithms have been proposed as potential solutions to detect intrusions efficiently. However, conventional ML-based classifiers have not seen widespread adoption in the real-world due to their poor domain adaptation capability. In this research, our goal is to explore the possibility of using sequences of flows to improve the domain adaptation capability of NIDS. Our proposal employs Natural Language Processing (NLP) techniques and Bidirectional Encoder Representations from Transformers (BERT) framework. The proposed method achieved positive results when tested on data from different domains.

1 INTRODUCTION

It is common in practical application of NIDS for there to be a change in the data distribution between its training data and the data it encounters when deployed. Conventional ML algorithms often adapt poorly to such change, which limit their usefulness in real-world scenarios [1]. To address this, Energy-based Flow Classifier (EFC) [2] was proposed as a solution. Despite having good adaptability, EFC produces high false positives rate for domains where the distribution of features of malicious flows overlap with that of benign flows. We theorize that the reason for the limitations of conventional ML algorithms and EFC is the use of singular flows as input data, as the classifier can only model the distribution of features within a flow. This limitation can be overcome with the use of sequences of flows, allowing the classifier to further models the distribution of a flow in relation to other flows. To utilize the context information from a sequence of flow, we use the BERT framework, which is able to process inputs in relation to all the other inputs in a sequence.

2 MATERIALS

2.1 Network Flow Data

Flow-based NIDS classifies traffic by analyzing network traffic flows. A flow is a sequence of packets carrying information between two hosts where packets share the same 5-tuple: Src IP, Dst IP, Src Pt, Dst Pt, and Proto.

Our research employs CIDDS-001[3] and CIDDS-002[4] data sets that contains flow samples from a small business environment emulated using OpenStack. CIDDS-001 also contains real traffic flow samples captured from an external server directly deployed on the internet.

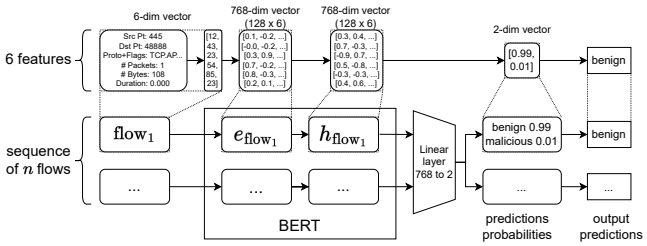

Figure 1: Proposed system architecture

2.2 BERT

BERT[5] is a transformer-based machine learning technique for NLP developed by Google. The BERT framework is comprised of two steps: pre-training and fine-tuning. In pre-training, the BERT model is trained on unlabeled data. For fine-tuning, the model is first initialized using the pre-trained parameters, and then trained using labeled data from the downstream tasks. BERT is pre-trained with two unsupervised tasks, which are Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). In MLM, some of the words in a sentence is replaced with a different token. The objective is to predict the original value of the masked words based on other unmasked words in the sentence. In NSP, BERT takes sentence pairs as input. The objective is to predict whether the second sentence in the pair is the next sentence in the document. For fine-tuning, task-specific inputs and outputs are added to a pre-trained BERT model.

3 METHOD

We first organize network traffic flows into structures similar to natural languages, treating a flow as a word and a sequence of flows as a sentence. We then pre-train BERT using MLM task. For fine-tuning, a linear layer with softmax output is used. It is important to preserve the distribution of flows within a sequence; therefore,

Table 1: Average composition of data sets

CIDDS-001 OpenStack				CIDDS-001 External Server		CIDDS-002	
1 set		10 sets		10 sets		10 sets	
CIDDS-001 large		CIDDS-001 internal		CIDDS-001 external		CIDDS-002	
label	#	label	#	label	#	label	#
normal	25178585	*normal*	10000	*unknown*	10000	*normal*	10000
dos	2775655	*dos*	9000	*suspicious*	10000	*scan*	10000
portScan	194642	*portScan*	935				
pingScan	5266	*pingScan*	45				
bruteForce	4992	*bruteForce*	20				
Total	28159140	Total	20000	Total	20000	Total	20000

during training, the data set is not shuffled. A training sample is generated by selecting a segment of flows from the data set at random.

The overall architecture of the system is illustrated in Figure 1. Each flow contains six features, these features are encoded as numbers (flow_i). BERT decodes each number into a 128-dimension vector, concatenates them to form a 768-dimension vector (e_{flow_i}), and processes it to produce a different 768-dimension vector (h_{flow_i}). The output of BERT is then passed through a Multilayer Perceptron classifier (a linear layer with softmax output), which reduces the dimension from 768 to 2. This 2-dimension vector represents the probability of each class (benign, malicious).

We used data from three different domains: CIDDS-001 OpenStack, CIDDS-001 External Server, and CIDDS-002 to evaluate the domain adaptation capability of the proposed method. Average composition of the data sets used in the experiment are shown in Table 1. Training was performed on one set of *CIDDS-001 large*. While testing was performed on *CIDDS-001 internal*, *CIDDS-001 external*, and *CIDDS-002*, each containing ten sets randomly selected from the full data sets. Flows labeled *normal* are considered benign, while those labeled otherwise are considered malicious. For *CIDDS-001 external*, flows labeled *unknown* and *suspicious* are considered benign and malicious respectively.

We assess the performance of our method in comparison to EFC and ML classifiers including Decision Tree (DT), K-Nearest Neighbors (KNN), Multilayer Perceptron (MLP), Naive Bayes (NB), and Support Vector Machine (SVM). Each classifier's performance is measured using Accuracy and F1 score [1].

4 RESULTS AND DISCUSSION

Table 2 shows the average performance and standard error for each classifier. All classifiers achieved higher Accuracy and F1 Score on *CIDDS-001 internal* test sets (same domain as training data) compared to the other test sets (different domains from training data). Both the proposed method and EFC maintained performance across the two different domains, with the proposed method outperforming EFC.

Table 2: Average classification performance and standard error

	Train CIDDS-001 large					
	Test CIDDS-001 internal		Test CIDDS-001 external		Test CIDDS-002	
Classifier	Accuracy	F1 score	Accuracy	F1 score	Accuracy	F1 score
Proposal	0.994(0.002)	0.994(0.002)	**0.895(0.027)**	**0.904(0.022)**	**0.916(0.045)**	**0.877(0.072)**
EFC	0.941(0.005)	0.941(0.004)	0.747(0.039)	0.796(0.025)	0.846(0.046)	0.800(0.070)
DT	**0.996(0.001)**	**0.996(0.001)**	0.870(0.028)	0.874(0.024)	0.818(0.061)	0.707(0.106)
KNN	0.989(0.002)	0.989(0.003)	0.839(0.008)	0.811(0.011)	0.818(0.061)	0.707(0.106)
MLP	0.992(0.001)	0.992(0.001)	0.573(0.009)	0.285(0.023)	0.832(0.059)	0.729(0.108)
NB	0.903(0.002)	0.892(0.002)	0.500(0.000)	0.000(0.000)	0.499(0.000)	0.001(0.000)
SVM	0.570(0.015)	0.245(0.047)	0.738(0.016)	0.718(0.021)	0.513(0.013)	0.090(0.037)

We also experimented with training the classifiers on smaller but balanced data sets (containing 80000 flows with the same proportion of labels as in *CIDDS-001 internal*). However, performance was worse for all classifiers when compared to those trained on *CIDDS-001 large*. Notably, the performance of the proposed method was significantly affected for all domains. By creating balanced data sets, the distribution of flows within a sequence was also altered. This suggests that the distribution of flows within a sequence is learned by the model of the proposed method.

5 CONCLUSION

In this study, we suggest the use of singular flows input to be a possible explanation for the poor domain adaptation capability of conventional ML-based classifiers. Then we proposed the used of sequences of flows to address this limitation. We utilized BERT model for the representation of flow sequences and an MLP classifier to discriminate between benign and malicious flows. Early experimental results showed that the proposed method is capable of achieving good and consistent results across different domains.

In future work, we aim to conduct a more comprehensive investigation on the use of flow sequences and BERT framework for network traffic classification. Finally, we also plan to use only benign flows for training to further improve the domain adaptation capability of our method.

REFERENCES

[1] Z. Ahmad et al. 2021. Network Intrusion Detection System: A Systematic Study of Machine Learning and Deep Learning Approaches. *Trans. Emerg. Telecommun. Technol.* 32, 1.

[2] C. F. T. Pontes et al. 2021. A New Method for Flow-Based Network Intrusion Detection Using the Inverse Potts Model. *IEEE Trans. Netw. Serv. Manag.* 18, 2.

[3] M. Ring et al. 2017. Flow—based Benchmark Data Sets for Intrusion Detection. In *Proc. of ECCWS 2017.*

[4] M. Ring et al. 2017, to appear. Creation of Flow—based Data Sets for Intrusion Detection. *Information Warfare* 16, 4.

[5] J. Devlin et al. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proc. of NACCL 2019.*