Sequence to Sequence Pattern Learning Algorithm for Real-Time Anomaly Detection in Network Traffic

Conference Paper · May 2018

DOI: 10.1109/CCECE.2018.8447597

3 authors, including:

Gobinath Loganathan
The University of Western Ontario
5 PUBLICATIONS **22** CITATIONS

SEE PROFILE

Xianbin Wang
The University of Western Ontario
553 PUBLICATIONS **11,116** CITATIONS

SEE PROFILE

Sequence to Sequence Pattern Learning Algorithm for Real-time Anomaly Detection in Network Traffic

Gobinath Loganathan*, Jagath Samarabandu† and Xianbin Wang‡
Department of Electrical and Computer Engineering
The University of Western Ontario
London, Ontario, N6A 5B9, Canada
*lgobinat@uwo.ca, †jagath@uwo.ca, ‡xianbin.wang@uwo.ca

***Abstract*—Network intrusions can be modeled as anomalies in network traffic in which the expected order of packets and their attributes deviate from regular traffic. Algorithms that predict the next sequence of events based on previous sequences are a promising avenue for detecting such anomalies. In this paper, we present a novel multi-attribute model for predicting a network packet sequence based on previous packets using a sequence-to-sequence (Seq2Seq) encoder-decoder model. This model is trained on an attack-free dataset to learn the normal sequence of packets in TCP connections and then it is used to detect anomalous packets in TCP traffic. We show that in DARPA 1999 dataset, the proposed multi-attribute Seq2Seq model detects anomalous raw TCP packets which are part of intrusions with 97% accuracy. Also, it can detect selected intrusions in real-time with 100% accuracy and outperforms existing algorithms based on recurrent neural network models such as LSTM.**

I. Introduction

Intrusion detection is a typical example of anomaly detection problem as most network intrusions occur as anomalous patterns in network traffic [1]. Several classification techniques have been proposed using available preprocessed datasets to identify intrusions. However, due to limitations of these datasets they failed to capture sequential relationships between raw packets [1], [2].

Anomaly detection in sequential data has been investigated separately using Long Short-Term Memory (LSTM) Recurrent Neural Networks (RNN) [3], [4]. Only recently have LSTM RNN being used to train on network packets in sequential order and to detect anomalies in upcoming packets based on prediction error [5].

Predicting upcoming packets based on previous packets is a Sequence-to-Sequence (Seq2Seq) problem. Seq2Seq encoder-decoder models are already being used in Neural Machine Translation (NMT) and object recognition in videos to get state of the art results [6]–[9]. Even though Seq2Seq prediction model has been used for anomaly detection in sensor data, it was trained using sequences with single attribute elements [10], which is similar to predicting words in NMT.

None of these applications use Seq2Seq model with sequences of multi-attribute elements like packet prediction. In packet prediction, each attribute of a packet can have different contribution weight on predicting upcoming packets. However, none of these attributes can disappear in output elements. In NMT and object recognition in videos, attention layers and convolutional encoders have been proposed to focus on relatively important parts of input sequence [11]–[13]. However, they do not address the problem of multi-attribute elements. Therefore a common attention layer that treats all input elements equally is not suitable for packet prediction.

In this research, we detect anomalies in raw TCP packets using a Seq2Seq model specially designed for multi-attribute sequences. To train the model, we use packets from regular network traffic split into connections. In testing, actual packets highly deviating from predicted packets are classified as anomalies. Training the model on normal traffic instead of intrusion traffic gives access to large training data and lets the model detect even new unknown attacks those are deviating from a regular pattern. Network attacks like Port Sweep and Neptune DOS use a mass amount of TCP packets but do not follow a regular TCP connection pattern. Therefore, we defined packets with certain prediction error as anomalies. We were able to detect packets from Port Sweep and Neptune DOS attacks with 97.02% detection ratio. The actual attacks containing such anomalous packets were detected in real-time with 100% detection ratio.

II. Background

Encoder-Decoder based Seq2Seq model with additional enhancements is mainly used in NMT [6], [7], [11]–[14]. It is also used in object recognition in video [8], [9] and anomaly detection in series of events [10]. However, LSTM RNNs have been widely used for sequence prediction and anomaly detection in sequences [3], [4].

Malhotra et.al trained stacked LSTM RNN and Seq2Seq model on non-anomalous sensor data [3], [10]. Prediction error of their model was fitted into multivariate Gaussian distribution. Observations with a likelihood of observing an error greater than a threshold are marked as the anomaly. [3]. Seq2Seq model gave better results for unpredictable datasets, whereas stacked LSTM RNN gave better results for predictable datasets [10]. In all these anomaly detectors, anomalies were defined based on the Gaussian distribution of prediction errors.

The sequence in NMT is a sentence formulated by words in a given order and the sequence in video analysis is a grid of pixels aligned sequentially. In a video frame, some pixels may be more important than others. Such important pixels

are extracted using convolutional encoders [8], [9]. Similarly in NMT, more relevant words to the translation are focused using attention layer [11]–[13].

In both NMT and video analysis, elements of a sequence are from same class: word and pixel respectively. Datasets used by Malhotra et.al also contains sequences of single attribute sensor readings. However, in packet prediction, both input and output are sequences of packets with multiple attributes. Those attributes may or may not have interdependencies. In such a multi-attribute Seq2Seq problem, the model must focus on all attributes and learn their interdependencies and how they change in upcoming elements.

Our work is closely related to that of Bontemps et al. where a simple LSTM RNN was used to detect Neptune DOS attacks using collective anomalies in a network [5]. In their research, the model was trained using an attack-free dataset to predict a packet using last three packets and the average prediction error was used to define anomalies. Bontemps et al. defined all the packets arrived within a fixed time window as a sequence. However, within a time window, there can be packets from multiple connections which are independent of each other. Further, the number of packets and types of packets arrived within a time window is depending on external factors like peak business hours. Therefore, their definition of sequence does not have a constant pattern and the model may not give the claimed accuracy with real datasets.

III. Methodology

A. Sequence to Sequence Model

Seq2Seq model has two RNNs named encoder and decoder. The goal of the encoder is converting an input sequence $X = (x_1, \ldots, x_n)$, into a vector c and the goal of the decoder is converting c into an output sequence $Y = (y_1, \ldots, y_n)$.

At each time step t, hidden state of an RNN h_t is computed by (1) where f is a non-linear activation function and x_t is the input at time t.

$$h_t = f(x_t, h_{t-1}) \tag{1}$$

In our model the non-linear activation function f is an LSTM and c is the final hidden state h_n of that LSTM. Decoder is another LSTM which generates $Y = (y_t, ..., y_i)$ using $Y = (y_1, ..., y_{t-1})$ as the input and c as the initial internal state. Hidden state of the decoder is computed by:

$$h_t = f(h_{t-1}, Y_{t-1}, c) \tag{2}$$

Figure 1 depicts the proposed multi-attribute Seq2Seq model. Every attribute of input sequence element is received by a dedicated Artificial Neural Network (ANN) branch. Output of those branches are merged into one and fed to the first encoder. This way, the model has the ability to learn the importance of each attribute independently. The output of the last decoder is shared across different ANN branches to predict individual output attributes of each element in the output sequence.

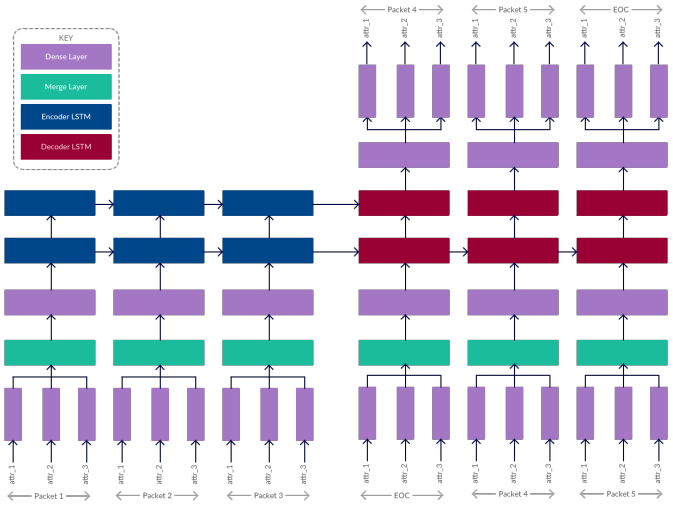

Fig. 1. Multi-attribute sequence to sequence model. We use four layers of encoders and decoders but only two are displayed here.

B. Training

The DARPA 1999 [15] dataset was used to train and test the model. The TCP packets from attack-free outside sniffing data were split into connections based on their sessions using PcapSplitter[1]. Connections with less than 4 packets were not used for training and connections with more than 60 packets were pruned to 60 because a connection must have at least 4 packets to train the model and 96.96% TCP connections in the training dataset have less than 60 packets. Connections having packets between 3 and 60, were padded by empty packets to maintain desired batch input format. The first three packets of a connection were used as input sequence to predict the rest.

The decoder was trained to predict $\{y_4, y_5, \ldots, y_n, EOC\}$ using $\{EOC, y_4, y_5, \ldots, y_n\}$ as input and hidden state of the encoder as the initial state. Here, EOC is a vector representation of End-of-Connection and n is the number of packets in that connection. We use Teacher Forcing [16] to train the model because it reduces error propagation in testing. Even if we predict more than one upcoming packets, we need to wait for actual packets to compare with.. Therefore, predicting $i+1^{\text{th}}$ packet after the arrival of i^{th} packet is enough and gives better results.

C. Evaluation

We tested our model using two datasets: (1) attack-free tcpdumps split into connections as we did for training; (2) tcpdumps containing both attack and normal traffic within a day; and (3) tcpdumps containing both attack and normal traffic over a week.

1) Test 1: The first dataset was used to check the accuracy of our model on predicting packets of normal TCP connections and end of connections with a goal of validating the model. For this test, the first three packets of valid TCP connections were fed to the encoder and the rest were fed to the decoder one by

[1]PcapPlusPlus available at https://github.com/seladb/PcapPlusPlus

Fig. 2. Weighted packet distance algorithm

Input: $actual\_packet, predicted\_packet, weights$
Output: $distance$
Initialize:
$distance \leftarrow 0$
for all $name \in actual\_packet.attributes$ **do**
  if $(actual\_packet[name] \neq actual\_packet[name])$ **then**
    $distance \leftarrow distance + weights[name]$
  end if
end for
$distance \leftarrow distance / \sum weights$
return $distance$

one to predict next packet. Two packets were considered equal only if all their attributes match to each other. The accuracy of a predicted connection is calculated by (3).

$$accuracy = \frac{No\ of\ correct\ predictions}{No\ of\ packets\ in\ connection} * 100 \quad (3)$$

2) Test 2: TCP packets collected on Thursday of the second week outside sniffing data were split into streams having requests and responses between same source and destination. The first three packets from each stream were fed to the encoder and the rest were fed to the decoder one by one to predict the next packet until the decoder generates an EOC. Suppose an EOC is generated after i^{th} packet in a stream, the first i packets will be considered as a connection and compared with predicted packets. The remaining packets in the stream will be used to predict next connection. If an EOC is not generated within τ packets, the decoder will emit predicted τ number of packets as a connection and a new prediction cycle will start from $\tau + 1^{th}$ packet. Even though most connections have less than 60 packets, we defined τ as 100 to be on safe side.

In preliminary Test 2, the accuracy defined using exact match of packets resulted in more false positives. Therefore, in Test 2 we compared the predicted packets with actual packets using a distance algorithm which compares each individual categorical attributes and produces a weighted distance. The weight of each attribute is determined based on preliminary observations. The computed distance of each connection was used to define the accuracy of prediction. Suppose a predicted connection has n packets, prediction accuracy of that connection is given by (4).

$$accuracy = \frac{\sum_{i=1}^{n} 1 - distance_i}{n} * 100 \quad (4)$$

D. Test 3

The goal of Test 3 is validating the application of our model in real-time anomaly detection. We preprocessed the packets collected in the second week of DARPA 1999 dataset as we did in Test 2 and fed to the model. The system will raise an alarm in real-time, if the average weighted prediction error of 60 packets is less than 12.5% which is the mean accuracy of anomalous packets in Test 2. The percentage of true positive alarms and number of false positive alarms were compared with the results obtained by Bontemps et al. using LSTM on the same dataset.

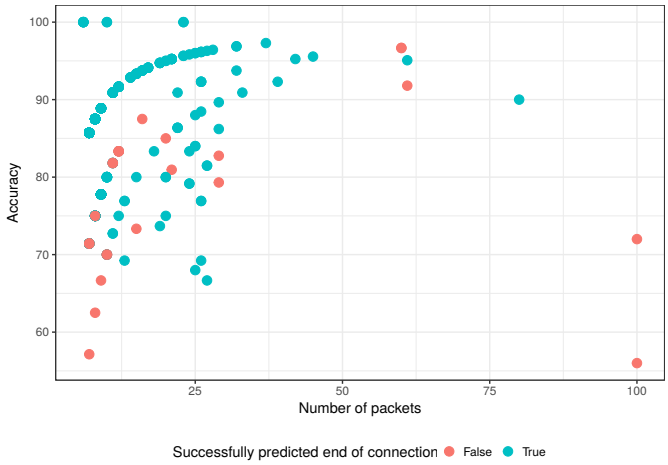

Fig. 3. Test 1 results

TABLE I
CLUSTER MEANS OF ACCURACY AND PREDICTED PACKETS IN TEST 2

Cluster	Accuracy	No of packets
1	95.19	97.08
2	61.80	9.52
3	76.52	7.68
4	89.16	13.06
5	76.66	96.49
6	12.25	94.47

IV. RESULTS

In Test 1, the model was able to predict connections with 84.97% accuracy and end of connections with 89.57% accuracy as shown in Figure 3. At the end of Test 2, we prepared a dataset with the number of packets in each predicted connection and the accuracy of connection. This dataset was further analyzed to define thresholds for anomaly detection.

A. Hypothesis

If the decoder is unable to find a connection in a stream of events, it will iterate τ times. Therefore, if the number of packets in a predicted connection is equal to τ (set to 100 in this experiment), either those packets are anomalies or the actual connection has greater than or equal to τ number of packets. However, if those packets are from an actual connection, the model may be able to predict them with a higher accuracy even though it cannot reach the end of the connection.

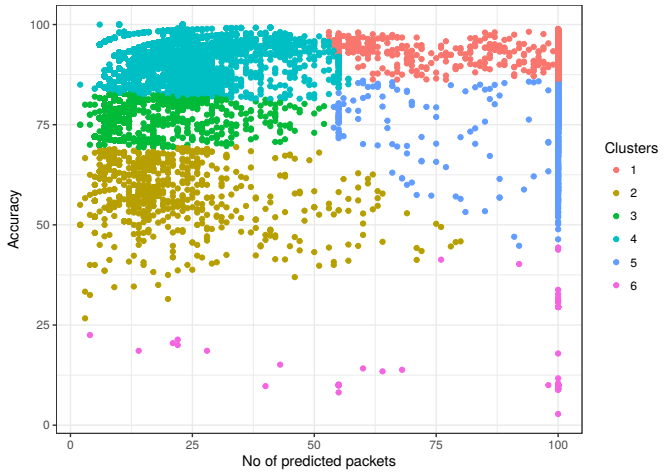

Fig. 4. Clusters of predicted connections.

B. Analysis

Results from Test 2 were clustered using K-means clustering algorithm into six clusters as shown in Figure 4. The number of clusters was determined by cross-validation. As shown in Table I, Cluster 6 has the lowest accuracy (12.25%) and a high number of predicted packets (94.47), which supports our hypothesis.

Even though the model classifies all attacks as anomalies without further distinctions, we manually analyzed the true positive packets from Cluster 6 and figured out that anomalous packets are from either Port-Sweep or Neptune DOS attacks. The proposed model is able to identify such anomalous packets with 97.02% Detection Ratio and 0.07% False Alarm Ratio.

In Test 3, the model was able to raise alarms on all Port-Sweep and Neptune DOS attacks with 100% true positive rate. Only a single False Alarm was raised in five days of network traffic. Bontemps et al. claimed 100% true positive alarms and 63 false alarms using LSTM on the same dataset [5]. Further, their model was able to detect only the Neptune DOS attack. Therefore, we claim that our Seq2Seq model outperforms LSTM in detecting anomalies in TCP traffic.

V. Conclusion

In this research, we have used Seq2Seq model for network anomaly detection in TCP requests. We have analyzed the results using offline unsupervised machine learning to learn threshold values and shown that the model can be used in online anomaly detection using those thresholds. If τ is large enough, it can capture more actual packet terminations but it will fail to identify anomalies at their early stage. If τ is too small, the number of false positives will increase because there is a higher chance of splitting connections into chunks. Therefore, τ must be selected carefully, considering the average number of packets received in a connection depending on the network.

Even though the testing dataset contains several other types of attacks, our model identifies only Port-Sweep and Neptune DOS because those are the only two found in testing data that are using TCP protocol and having more anomalous packets. However, the proposed model can be trained on different protocols and used to detect other attacks which are using those specific protocols.

References

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," vol. 41, no. 3, pp. 15:1–15:58, 2009. [Online]. Available: http://doi.acm.org/10.1145/1541880.1541882

[2] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Towards generating real-life datasets for network intrusion detection," vol. 17, no. 6, pp. 683–701, 2015.

[3] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal, "Long short term memory networks for anomaly detection in time series," in *ESANN, 23rd European Symposium on Artificial Neural Networks, Compu- tational Intelligence and Machine Learning*, 2015.

[4] S. Chauhan and L. Vig, "Anomaly detection in ecg time signals via deep long short-term memory networks," in *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, 2015, pp. 1–7.

[5] L. Bontemps, V. L. Cao, J. McDermott, and N. Le-Khac, "Collective anomaly detection based on long short term memory recurrent neural network," 2017. [Online]. Available: http://arxiv.org/abs/1703.09752

[6] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. NIPS*, 2014. [Online]. Available: http://arxiv.org/abs/1409.3215

[7] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder–decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2014, pp. 1724–1734. [Online]. Available: http://www.aclweb.org/anthology/D14-1179

[8] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville, "Describing videos by exploiting temporal structure," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 4507–4515.

[9] N. Srivastava, E. Mansimov, and R. Salakhutdinov, "Unsupervised learning of video representations using lstms," in *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15. JMLR.org, 2015, pp. 843–852. [Online]. Available: http://dl.acm.org/citation.cfm?id=3045118.3045209

[10] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "Lstm-based encoder-decoder for multi-sensor anomaly detection," vol. abs/1607.00148, 2016. [Online]. Available: http://arxiv.org/abs/1607.00148

[11] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," vol. abs/1409.0473, 2014. [Online]. Available: http://arxiv.org/abs/1409.0473

[12] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2015, pp. 1412–1421. [Online]. Available: http://aclweb.org/anthology/D15-1166

[13] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2016, pp. 1480–1489. [Online]. Available: http://www.aclweb.org/anthology/N16-1174

[14] J. Gehring, M. Auli, D. Grangier, and Y. N. Dauphin, "A Convolutional Encoder Model for Neural Machine Translation," 2016.

[15] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, "The 1999 darpa off-line intrusion detection evaluation," *Comput. Netw.*, vol. 34, no. 4, pp. 579–595, 2000. [Online]. Available: http://dx.doi.org/10.1016/S1389-1286(00)00139-0

[16] J. W. Ronald and Z. David, "A learning algorithm for continually running fully recurrent neural networks," *Neural Computation*, vol. 1, no. 2, pp. 270–280, 1989.