Bidirectional LSTM Autoencoder for Sequence Based Anomaly Detection in Cyber Security

Conference Paper *in* International Journal of Simulation: Systems · October 2019

DOI: 10.5013/IJSSST.a.20.05.07

Ashima Chawla
Ericsson Athlone
5 PUBLICATIONS 18 CITATIONS
SEE PROFILE

Paul Jacob
Athlone Institute of Technology
29 PUBLICATIONS 110 CITATIONS
SEE PROFILE

Brian Lee
Athlone Institute of Technology
75 PUBLICATIONS 703 CITATIONS
SEE PROFILE

Sheila Fallon
Athlone Institute of Technology
15 PUBLICATIONS 34 CITATIONS
SEE PROFILE

Bidirectional LSTM Autoencoder for Sequence based Anomaly Detection in Cyber Security

Ashima Chawla, Paul Jacob, Brian Lee, Sheila Fallon

Department of Electronics & Informatics, Athlone Institute of Technology, AIT, Athlone, Ireland.

E-mail: a.chawla@research.ait.ie; pjacob@ait.ie; blee@ait.ie; sheilafallon@ait.ie

Abstract - **Cyber-security is concerned with protecting information, a vital asset in today's world. The volume of data that is generated can be usefully analyzed when cyber-security systems are effectively implemented with the aid of software support. Our approach is to determine normal behavior of a system based on sequences of system call traces made by the kernel processes in the system. This paper describes a robust and computationally efficient anomaly based host based intrusion detection system using an Encoder-Decoder mechanism. Using CuDNNLSTM networks, it is possible to obtain a set of comparable results with reduced training times. The Bidirectional Encoder and a unidirectional Decoder is trained on normal call sequences in the ADFA-LD dataset. Intrusion Detection is evaluated based on determining the probability of a sequence being reconstructed by the model representing normal data. The sequences with a low probability value are classified as an anomaly.**

Keywords – Autoencoders, CuDNNLSTM, Embeddings, Host Based Intrusion Detection, system call

I. INTRODUCTION

An Intrusion Detection System (IDS) is a software system built for detecting vulnerability and protecting an organization's digital assets. Intrusion Detection Systems can protect an individual computer (Host Based IDSs) or networks (Network Based IDSs). Two further types of IDS are signature-based, and anomaly based. The signature based approach [4], also known as misuse detection operates in much the same way as a virus scanner, by searching for identities or signatures of known intrusion events while the anomaly based approach establishes a baseline of normal patterns and flags anomalous behavior. Anomaly based intrusion detection is difficult and can benefit greatly from the use of Artificial Neural Network (ANN) techniques which can learn normal patterns and detect outliers.

In this context, Australian Defense Force Academy Linux Dataset (ADFA-LD), recently released system call dataset consists of 833 normal training sequences, 746 attack, 4372 validation sequences [1] and has been used for evaluating a system call based HIDS. The system call traces consists of call sequences of integers. The ADFA-LD system call patterns represents true performance against contemporary modern attacks in the system.

An autoencoder, an unsupervised learning technique is a neural network that is trained to produce an output which is the same as its input. Since the concept of autoencoders does not have dedicated targets, hence it is trained in an unsupervised manner to learn data representations. An autoencoder involves dimensionality reduction of high dimensional data, like Principal Component Analysis (PCA) [2]. The difference lies in the fact that autoencoders have shown to be more efficient in case of nonlinear encoding/decoding compared to PCA. The figure below shows an autoencoder which consists of encoder, latent space representation compressed state after dimensionality reduction (z) and a decoder reconstructing the output values ($X^{'}$) similar to the inputs (X).

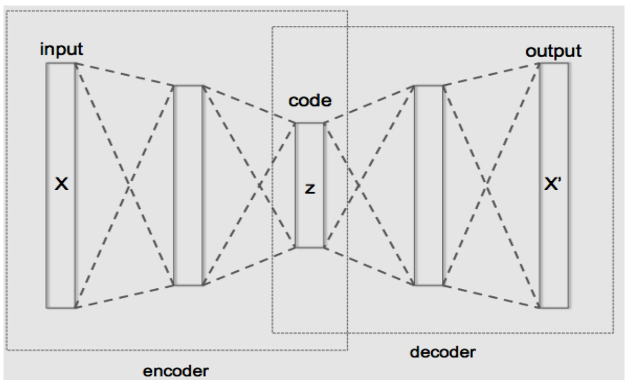

Figure 1. Auto-encoders

Recurrent Neural Networks (RNN) work effectively in the case of small range dependencies, but often prove to be inefficient in long range term dependencies. With longer sequences, the Back Propagation through Time mechanism (BPTT) can result in vanishing gradient descent problems [28]. In 1990s, Hochreiter and Schmidhuber [5] proposed a novel recurrent architecture which could retain the important information over long sequences, known as **Long Short Term Memory** (LSTM) neural network. These are now usually used in handling sequential data for many natural language processing tasks, such as speech recognition [9] and various other text summarization models [10], [11], [12]. In section II, we review the related work conducted by various researchers in the same field. An overview of methodology is presented in section III. Section IV outlines the model architecture defining the layers of the model. The Anomaly Detection algorithm is explained in section V. Experimental results and conclusion is covered in section VI and VII.

II. RELATED WORK

Authors of [3] proposed a robust IDS using LSTM [5] to capture the semantic meaning of system calls in ADFA-LD dataset. The authors represented an end to end framework to detect anomalous patterns in system calls. Additionally, they proposed leaky ReLU [26] based ensemble method to combine the multiple classifiers. The ensemble method achieved 90 % TDR, with 16% FAR using ensemble proposed methodology, with an AUC value of 0.928. Motivated by [3], the authors [4] used a combination of CNN and GRUs to substantially reduce training and testing time for the ADFA-LD dataset over LSTM based models. The CNN+GRU model achieved 90% Detection Rate with 30% false alarm rate, resulting in an AUC value of 0.81.

In [8], the proposed methodology implemented a combination of Multi-Channel CNN and BiLSTM for detecting the intrusion in ADFA-LD dataset. The authors used the concept of BiLSTM with embedding to extract the relationship of system call traces. The proposed algorithm performed better on average low false alarm rates and high detection rates when compared to One-class SVM(Support Vector Machine), RNN, CNN without word embedding, BiLSTM. The authors achieved 0.97 F1 score, which improved the state-of-the-art classification performance of designing a good IDS.

In [15], the authors proposed attention mechanism in combination with GRU based Encoder-Decoder mechanism for evaluating the performance of the anomaly detection algorithms. The model performed well when the long system call traces of ADFA-LD where the longer sequences were broken down into further smaller sequences. The length of sequence length varies from 10, 20 15, 18, 20, 22, 25 and 30. The HIDS achieved AUC of 0.94 with 90% True Detection Rate and ~15% False Alarm Rate for predicted ROC. When compared to other models such as SVM, CNN and Random Forest, RNNs performed the best classifier.

In 2014, [9] undertook a sequence to sequence learning with Deep Neural Networks (DNN) model using LSTM units. The multi-layered LSTM model performed markedly on English to French translation task. The authors outlined a comprehensive review of the state-of-the-art in sequence to sequence learning, which typically involves carrying information from previous time step to later ones. This work outlined the methodology implemented for mapping the input sequence to a fixed sized dimensionality vector. Further the target sequences were decoded using the vector information. The authors concluded that LSTM based approach worked significantly well on many other sequence problems obtained by reversing the words in a sentence.

Inspired by [9], [13] proposed another LSTM based Auto-encoder which obtained state-of-the-art results in anomaly detection algorithms. An Auto-encoder consist of an encoder and a decoder where the model predicts the output same as the given input. The encoder learns a compressed vector representation of input data, and decoder use this information to produce the target sequence. The Encoder-Decoder were successfully able to learn to reconstruct the normal behavior, and thereafter uses the reconstruction error to detect anomalies.

Work from [13] inspired the authors [14], who investigated a stateful LSTM based architecture to detect the root cause analysis of anomalous behavior in multivariate time series data. They introduced the concept of one handle for filtering the faults which were of not any interest. The authors used only dataset with normal behavior to train their multilayer LSTM model. The proposed mechanism involved two parts: one is to detect and then another is to forecast the values using conditioned observed values. The evaluation was conducted using Mean Squared Error, where the value above tunable threshold is considered as fault type, else a normal observation. The experimental result from this model showed that the model achieved high precision and recall values as 0.976 and 0.788 respectively over classic fault detection methods. The authors mentioned that using this architecture, it is possible to handle a lot of false positive alerts in monitoring systems. This is another approach of Encoder-Decoder mechanism that we have used in our algorithm.

Our proposed model consists of a Bidirectional Encoder to read the information from both past and future directions with a unidirectional Decoder. Our two significant contributions from the model architecture are as follows. Firstly, it was observed that the CuDNNLSTM language model implementation has substantially reduced the training time when compared to an LSTM model. The time taken for CuDNNLSTM is approximately 10x faster than LSTM due to faster convergence in training. The stateful LSTM units in the model are processed with a fully connected softmax layer that outputs a probability distribution over system call integers.

Secondly, the experimental results show that the Area under Curve (AUC) as compared to our previous model [4] has significantly improved and approached the state-of-the-art performance. Results from [15] shows that with attention mechanism, it is possible to improve the AUC.

III. METHODOLOGY

A. Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a form of network with backward connections, where output from a layer in the network is fed back into either that layer or a previous layer in the network [27]. RNNs maintain state, in that values calculated at a previous timestep are used in the current timestep. This state is used as a form of short term memory in the network and RNNs are typically used as a model for time series and sequential data where values at previous time steps can affect the current calculation [4].

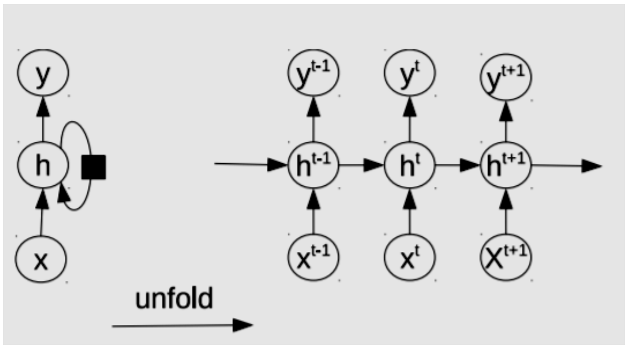

Figure 2. RNN Model Architecture

As shown in figure 2, RNNs can be unfolded to become a regular neural network. In this diagram a single node represents a complete layer in the RNN. Backpropagation applied to this unfolded network is known as Backpropagation Through Time and can be used to train the RNN. While RNN can be trained to capture short term dependencies between time steps, it has proved difficult to train RNNs to capture long term dependencies due to the so called "vanishing gradient" problem [5]. To overcome this, special types of RNNS have been designed, in particular Long Short Term Memory networks (LSTM) and Gated Recurrent Units (GRU).

B. Long Short Term Memory (LSTM)

LSTM consists of "context" or "cell" state which acts like a conveyor belt which runs over the sequence of input and stores additional state values over time [5]. LSTM replaces traditional RNN with input gates, output gates and forget gates. Each gate provide access to the cells in such a way that values can be stored in the cell for either short or long periods of time and removed when no longer needed.

C. Bidirectional RNN

With respect to natural language processing tasks, such as speech recognition [9], bidirectional recurrent neural networks [6] have produced the state-of-the-art performance. Bidirectional recurrent neural networks seek information from both earlier and later (chronologically and anti-chronologically) in the sequence. The output of the forward and backward layer is combined at each time step by one of the following merge modes means:

- Concat: The outputs are concatenated together, hence providing double the number of outputs to the next layer. In our proposed model, we have implemented "concat" to merge the states.
- Mul: The outputs are multiplied.
- Sum: The outputs are added.
- Ave: Average of the two output is taken.

The neural network is principally trained using an algorithm such as Backpropagation Through Time (BPTT) which solves the vanishing/exploding gradient problem. The final prediction of the neural network at time (t) is calculated using the following equation:

$$y^{\wedge<t>} = g(W_y[a^{\rightarrow<t>}, a^{\leftarrow<t>})] + b_y) \qquad (1)$$

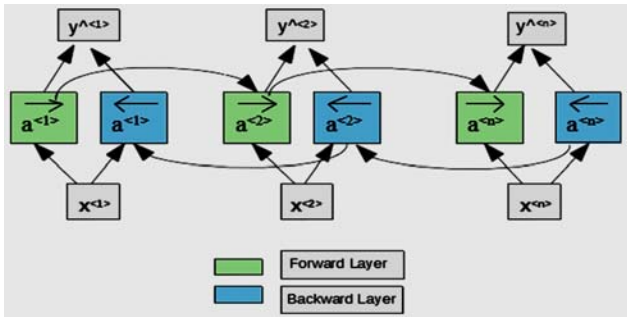

Figure 3. Acyclic Bidirectional Recurrent Neural Network Graph

Where, g denotes the non-linear activation function, W_y denotes weight matrix, $a^{\rightarrow<t>} a^{\leftarrow<t>}$ denotes forward and backward recurrent components respectively. b_y denotes bias and $y^{\wedge<t>}$ denotes output at timestep "t". Here figure 3 shows that predicted output at $y^{\wedge<1>}$, $y^{\wedge<2>}$ .. $y^{\wedge<n>}$ is calculated for each of the recurrent unit inputs defined, for forward and backward connected to each other.

Note that BRNNs require input of the entire sequence of data before making any predictions. Also, when implementing an autoencoder using BRNNs there is no need to reverse the source/target sequence, as BRNNs takes information from both the forward and backward direction.

D. Embedding Layer

Embedding is another dense layer in the neural network, where the output vectors obtained are of low dimensionality [20]. Generally, one hot encoding of input vectors is sparse with very high dimensional output. Like any other dense layer in a neural network, the word embedding layer is also trainable, where each input integer is used as the index to access a table that contains all possible vectors. Hence, it results in a lower dimensional output vector. Here, figure 3 shows the sparse one-hot encoded vectors on the left and the dense embedded representation on the right.

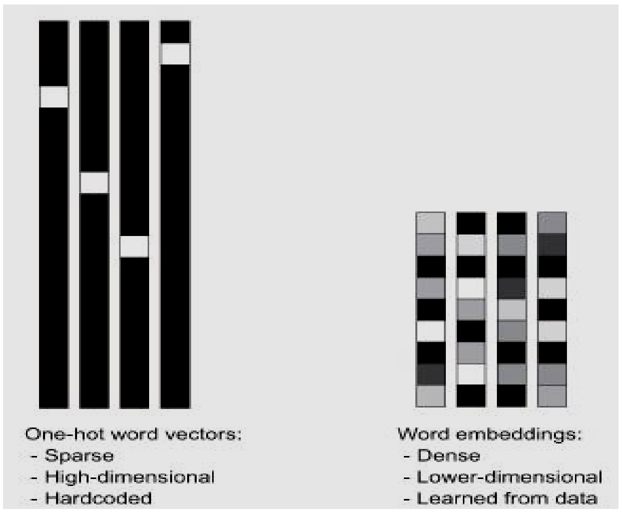

Figure 4. Embedding representation [20].

A. Sequential Language Modeling

A sequential neural network model is trained, given a set of input training sequences, defined as ($x1, x2, \dots . x_T$) where x_i is an integer. The proposed RNN Encoder-Decoder model is jointly trained to produce the probability distribution over a sequence; given the initial hidden state set to the representation *c* of ($x1, x2, \dots . x_T$). The encoder part of the model encodes a variable length sequence into a fixed vector representation, which consists of a hidden state used to initialize the states of the decoder. RN

The decoder maps the fixed vector presentation *c* of the input sequence to generate the target sequence$(y_1, \dots, y'_T)$ predicting the next integer in the sequence. The conditional probability distribution of the next integer is represented using a softmax activation function over the system call traces in the vocabulary.

$$P(y|x) = \Pi_{t=1}^{T'} P(y_t|c, y_1, \dots\dots, y_{t-1}) \tag{2}$$

In practice the negative log of the value defined in equation (2) is used, resulting in high values for unlikely sequences and low values for likely sequences. Anomaly detection for sequences is carried out by imposing a threshold for this negative log likelihood (L) and predicting an anomaly for sequences with an L value above this threshold [4].

B. CuDNNLSTM

CuDNNLSTM are variants of LSTM which are supported only on GPU systems with the Tensorflow-gpu backend. CuDNNLSTMs have proved to significantly faster as compared to LSTM [21] [22]. By default, CuDNNLSTM supports tanh activation function.

IV. MODEL ARCHITECTURE

The model architecture in figure 5 shows the keras embedding layer [4] which performs the word embedding and transforms one hot encoding of integers in the call sequence. The range varies from 1 to 340 is compressed into a dense vector of size 64. The Keras Batch Normalization layers were incorporated to reduce internal covariate shift, by normalizing layer inputs, resulting in faster training. The bidirectional encoder and unidirectional decoder part of the model is jointly trained. The decoder RNN contains the same number of layers and units as the encoder RNN. The encoder part of the model reads the variable length input and compresses the input data into latent space representation, known as context vector. The encoder units are stateful with keras parameter $return\ state = True$ and the output from the encoders is discarded. The decoder is then initialized by using the states of the encoder (hidden state + cell state) to generate the target sequence probabilities. The decoder layer returns both state and sequences [23].

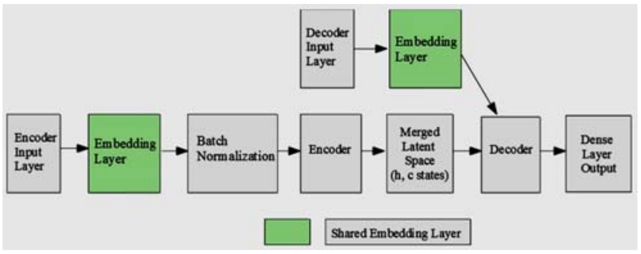

Figure 5. Auto-encoder Model Architecture

We used the Adam optimizer with a batch size of 64. The probability of a sequence [4] occurring is calculated using the conditional probability distributions. The output layer is a keras Dense layer, essentially a regular densely connected neural network layer. It is used with a softmax activation function in order to predict a probability [4] distribution for the next integer in the call sequence.

V. ANOMALY DETECTION

Given a set of normal sequences, the anomaly detection process evaluates the unseen system call trace as anomalous or normal. As the implementation is cost efficient, it consumes much less computational resources in comparison to the traditional deep neural network models. In this section, we discuss the anomaly detection process based on the approach in section III (*E*).

For the purposes of evaluation, Detection Rate (DR) and False Alarm Rates (FAR) were defined as:

$$DR = TP/(TP + FN) \tag{3}$$

$$FAR = FP/(FP + TN) \quad (4)$$

Where TP is True Positive, FN is False Negative, TN is True Negative, and FP is False Positive.

A. Training

• The Encoder-Decoder model is trained to reconstruct the input sequence.

• The model is trained in minibatches with the set of normal system call traces (no attack sequences) ADFA-LD dataset. The loss function is the categorical cross entropy function.

B. Calculation of Sequence Probability

• The input sequence is fed through the model. The output is a sequence of probability distributions using hidden states, which represent the probability distribution for the next integer in the sequence.

• A sequence probability value is calculated using equation 2, essentially multiplying the probabilities of the next integer in the sequence occurring, across the entire length of the sequence.

C. Calculation of ROC curve data and AUC for validation data

- The negative log of the sequence probability is calculated for every sequence in the validation data.
- For a range of threshold values
- For each sequence in the validation data
- If the negative log value for the sequence is greater than the threshold, the sequence is classified as attack (Positive), otherwise it is classified as normal (Negative). (This corresponds to the sequence probability for abnormal classification being below a threshold, i.e. unlikely)
- The sequence is identified as either TP, FP, TN, FN
- The DR and FAR for the particular threshold value is calculated
- The ROC curve is plotted for the range of threshold values
- The AUC value is calculated for the ROC curve.

VI. EXPERIMENTAL RESULTS

The ADFA intrusion detection dataset consists of 833 normal training sequences, 4372 normal validation and 746 attack system call traces. In order to train the model well and generalize (avoid overfitting), we split the ADFA-LD dataset into two partitions, training and testing. We train a Bidirectional CuDNNLSTM based encoder decoder with two third of normal system call traces (normal validation) 3470 traces. The rest one third of the normal system call traces (1735) and attack sequences (746), total 2481 system call traces were tested after the model is trained.

The specification of the computational machine includes Intel core i7-8700@3.20GHz processor, 16GB of RAM and NVIDIA GeForce GTX1070 GPU running 64-bit Windows 10 operating system and the NVIDIA CUDA Deep Neural Network library (CuDNN). The Keras python library [7] was used running on top of a source build of TensorFlow 1.7.1 with CUDA support.

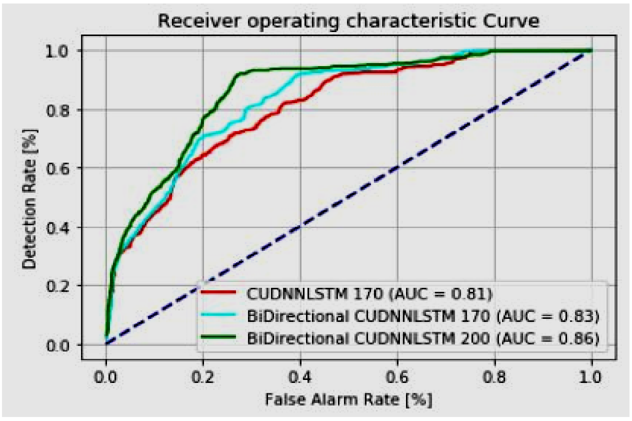

Figure 6: ROC curve comparing different models of ADFA Dataset

Figure 6 shows the ROC curves for the above outlined model. The model with 200 units gave the best value (0.86) for the Area Under the ROC curve (AUC). CuDNNLSTM with 170 was behind resulting in an AUC value of 0.81. The model produces approximately 90% True Detection Rate with a False Alarm Rate of 25%.

VII. CONCLUSION

In this experiment, we proposed an RNN based Encoder-Decoder model, where the entire sequence is read by the neural network. We evaluated a bidirectional CuDNNLSTM which can read the information from both the past and the future. It was observed that the CuDNNLSTM language model implementation has substantially reduced the training time when compared to an LSTM model. The time taken for CuDNNLSTM is approximately 10x faster than LSTM due to faster convergence in training.

Secondly, the area under the curve as compared to our previous model [4] has significantly improved and reached the state-of-the art performance. There was no specific requirement to reverse the source/target sequences, as mentioned in most of the encoder decoder papers for effective Encoder-Decoder learning [8]. We have maintained nearly state-of-the-art performance for neural network models by designing computationally efficient models, with a substantial smaller training times when compared to the other traditional machine and deep learning models. The empirical evaluation demonstrates that this experiment has improved the overall anomaly detection performance, in terms of AUC score.

For future work, a relatively new technique in artificial neural networks, known as an 'attention mechanism', supports interpretation of the neural network by determining the sub-sequence(s) that are causing the sequence to be classified as anomalous. The attention mechanism concept will be an important future research to apply to HIDS. The key findings and the methods from this work could be beneficial to many Computer Security Incident Response Teams (CSIRTs), where the anomalies could be detected in lesser amount of time.

Secondly, we intend to implement this anomaly detection technique for evaluating anomalies using combined clustering/Encoding-Decoding techniques. The use of the reduced representation for a cluster algorithm in autoencoders [25] will be a great benefit to anomaly detection applications.

Finally, an ensemble method will most likely give the best results and we plan to build and evaluate such a model.

ACKNOWLEDGMENT

This project reported in this paper has received funding from the European Union Horizon 2020 research and innovation programme under grant agreement No. 700071 for the PROTECTIVE project.

REFERENCES

[1] G. H. J. Creech, Generation of a New IDS Test Dataset,"Time to Retire the KDD Collection", in IEEE Wireless Communications and Networking Conference (WCNC), 2013.

[2] M. Brems, "A One-Stop Shop for Principal Component Analysis," Medium towards Data Science, 17 April 2017. [Online]. Available: https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c. [Accessed 2 March 2019].

[3] G. Kim, H. Yi, J. Lee, Y. Paek, and S. Yoon. (2016). ''LSTM based system-call language modeling and robust ensemble method for designing host-based intrusion detection systems.'' [Online]. Available: https://arxiv.org/abs/1611.01726.

[4] A. Chawla, B.Lee, S.Fallon and P. Jacob, Host based Intrusion Detection System with Combined CNN/RNN Model, in ECML-PKDD, Dublin, 2018.

[5] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory", Neural Computation, vol. 9, no. 8, pp. 17351780, 1997.

[6] M. Schuster and K. Paliwal,"Bidirectional Recurrent Neural Networks", IEEE TRANSACTIONS ON SIGNAL PROCESSING, VOL. 45, NO. 11, NOVEMBER 1997.

[7] Keras Home Page, https://keras.io/, Last accessed on July 19, 2019.

[8] N. N. Diep, N. T. T. Thuy and P. H. Duy, "COMBINATION OF MULTI-CHANNEL CNN AND BiLSTM FOR HOST-BASED INTRUSION DETECTION", Southeast Asian J. of Sciences, vol. 6, no. 2, pp. 47-159, 2018

[9] I. Sutskever, O. Vinyals and Q. V. Le, Sequence to Sequence Learning with Neural Networks, NIPS'14 Proceedings of the 27th International Conference on Neural Information Processing Systems, 2014.

[10] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho and Y. Bengio," Attention-Based Models for Speech Recognition", Neural Information Processing Systems, 2015.

[11] R.Nallapati, B.Zhou, C.N.d.Santos, C.Gulcehre and B.Xiang,"Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond", Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL), pages 280–290, Berlin, Germany, August 7-12, 2016.

[12] Y. Shen, P.Huang, J.Gao and W.Chen,"ReasoNet: Learning to Stop Reading in Machine Comprehension. Microsoft Research", Neural and Evolutionary Computing (cs.NE), 17 Sep 2016.

[13] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal and G. Shroff, LSTM based Encoder-Decoder for Multi-sensor Anomaly Detection, in ICML, New York, 2016.

[14] P. Filonov, A. Lavrentyev and A. Vorontsov, "Multivariate Industrial Time Series with Cyber-Attack Simulation: Fault Detection Using an LSTM-based Predictive Data Model", in arXiv:1612.06676, 2016.

[15] S. Lv, J.Wang, Y.Yang and J.Liu," Intrusion Prediction with System-call Sequenceto Sequence Model," arXiv:1808.01717, 2018.

[16] G. Kim, H. Yi, J. Lee, Y. Paek and S. Yoon, LSTM-Based System-Call Language Modeling and Robust Ensemble Method for Designing Host-Based Intrusion Detection Systems, eprint arXiv:1611.01726, 2016.

[17] Bengio Y, Vincent P, Janvin C.,"A neural probabilistic language model", [J]. Journal of Machine Learning Research, 3(6):1137-1155, 2006.

[18] A. Graves, Navdeep. J and A. Mohamed," Hybrid speech recognition with Deep Bidirectional LSTM," Automatic Speech Recognition and Understanding (ASRU), IEEE Workshop, December 2013.

[19] A. Graves, A. Mohamed, and G. Hinton, Speech recognition with deep recurrent neural networks, in Proc ICASSP 2013, Vancouver, Canada, May 2013.

[20] F. Chollet, Deep Learning with Python, Manning Publications, 2018.

[21] M. Zhang, S. Rajbhandari, W. Wang and Y. He, "Deep CPU: Serving RNN based Deep Learning Models 10x Faster," in 2018 USENIX Annual Technical Conference(USENIXATC'18), Boston, MA, USA, 2018.

[22] J. Appleyard, T. Kociský and P. Blunsom," Optimizing Performance of Recurrent Neural Networks on GPUs," in arXiv:1604.01946v1, 2016.

[23] J. Brownlee, Machine Learning Mastery, 13 October 2017. [Online]. Available: https://machinelearningmastery.com/how-does-attention-work-in-encoderdecoder-recurrent-neural-networks/. Last accessed on July 19, 2019

[24] Z. Ghahhramani, "An Introduction to Hidden Markov Models and Bayesian Networks", International Journal of Pattern Recognition and Artificial Intelligence, 2001.

[25] C.Song, F.Liu, Y.Huang, L.Wang and T.Tan, "Auto-encoder Based Data Clustering", Ruiz-Shulcloper J., Sanniti di Baja G. (eds) Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications. CIARP 2013. Lecture Notes in Computer Science, vol 8258. Springer, Berlin, Heidelberg.

[26] A. L. Maas and A. Y. N. A. Y. Hannun, "Rectifier Nonlinearities Improve Neural Network Acoustic Models", in Proceedings of the 30th International Conferenceon Machine Learning, Atlanta, Georgia, USA, 2013.

[27] Graves, Alex: Supervised Sequence Labeling with Recurrent Neural Networks. Studies in Computational Intelligence. Springer,(2012)

[28] Y.Bengio , P.Simard , P. Frasconi," Learning long-term dependencies with gradient descent is difficult[J]". IEEE Trans Neural Netw, 2002, 5(2):157-166