An Approach to Build an Efficient Intrusion Detection Classifier

[1]Thi-Thu-Huong Le, [1]Jihyun Kim, [1]Jaehyun Kim, *[1]Howon Kim

[1, First Author] *Department of Computer Science and Engineering, Pusan National University, lehuong7885@gmail.com,*
[1] *Department of Computer Science and Engineering, Pusan National University, jihyunkim@pusan.ac.kr*
[1] *Department of Computer Science and Engineering, Pusan National University, jh9012@pusan.ac.kr*
[*1, Corresponding Author] *Department of Computer Science and Engineering, Pusan National University, howonkim@pusan.ac.kr*

Abstract

The Intrusion Detection System (IDS) is an effective method to deal with the types of problem in networks. In particular, IDS is an effective way to obtain high security in detecting intrusion activities. An anomaly detection is one of intrusion detection with a high false alarm, a moderate accuracy and a detection rates when it is unable to detect all types of attacks correctly. To address this problem, we propose new approach through deep learning field. By applying Gated Recurrent Unit (GRU) Recurrent Neural Network (RNN), we build an effective intrusion detection classifier with higher performance on the dataset is extracted from KDD Cup 1999 dataset. From our experiments, we obtain 97.06%, 98.65%, and 10.01% for detection rate, accuracy, and false alarm rate respectively. Our result shows that the proposed approach model outperforms other model using FNNN, GNNN, RBNN, KNN, SVM, etc. for accuracy, detection rate with a significant false alarm rate.

Keywords*: Intrusion Detection System, IDS, Recurrent Neural Network, Gated Recurrent Unit, Recurrent Neural Network*

1. Introduction

Intrusion Detection System (IDS) is a famous system to protect a network system from malicious software attacks. IDS is a software or hardware based systems that detect intrusions on network or host. The main purpose of IDS are the detection, reporting, log generation and correlation of system and network security events.

Intrusion detection attacks can be classified into two groups based on the method of detection, misuse or signature based system and anomaly based system. In *misuse* or *signature based system*, the system analyzes data from audit logs to create a rule or signature for the attack. Therefore, it can detect only known attacks with the fewer false alarms. And *anomaly based system*, it relies on learning about past behavior of users. Analysis of the audit logs determines what behavior is normal for users every time. Any deviations generate alerts. There are some approaches to build IDS classifiers such as neural networks which are applied to IDS field by many researchers recently. Although neural networks are good for detecting attack on IDS, we hereby extend this research deep learning approach as followings. Deep learning is extended from neural network. Deep learning can model complex system behaviors due to training a high level data abstraction. By this work, we apply one of the method in deep learning. This approach is named as Gated Recurrent Unit Recurrent Neural Network (GRU RNN). In this paper, the KDD 99 dataset is used to training and testing data and we show how to improve detecting intrusion performance on IDS.

[*] Corresponding Author

Received: Nov. 02, 2015, Revised: Dec. 12, 2015, Accepted: Jan. 17, 2016

The remaining of this paper is given as followings: In Section 2, the related work used for intrusion detection is discussed. In Section 3, we summary of the background regarding GRU RNN. Section 4, briefly description of the extracting dataset from KDD Cup 1999 dataset for training. In Section 5, our experimental results are presented in details. Finally, we conclude our works in Section 6.

2. Related work

In recent years, Intrusion Detection System (IDS) has important role in IT industry. Many researchers have been reported based on techniques such as classification, statistical, soft computing and hybrid techniques together. Almost the techniques to build IDS based on the attack existing pattern to recognize [1]. They help to reduce the cost of detection. There are many approaches for detecting intrusion classifiers including Artificial Neural Network (ANNs) [11] is proposed by Zhang, decision tree system [10], Support Vector Machine (SVM) [12], genetic algorithm [13] and so on. By using K-mean clustering with Naive Bayes classifier on KDD Cup 99 dataset, randomly selected patterns are used for training and testing by Muda et al. [14].

Moreover neural network is one of the method used widely in the recently. There are many kinds of neural network which are applied in IDS, such as SOM [2], feedforward neural network [3], combination between neural network and SVM [4], Xue et al. [5] with Jordan recurrent neural network.

The paper [17] has proposed five types of classifiers that are applied on IDS. They are FFNN (Feed Forward Neural Network), ENN (Elman Neural Network), GRNN (Generalized Regression Neural Network), PNN (Probabilistic Neural Network), and RBNN (Radial Basis Neural Network). The conclusion of this paper is that the better accuracy than rest of other neural networks.

3. Background

Feedforward Neural Network (FNN) is the first ANN. FNN includes input layer, hidden layer, output layer and the connections between layers, are called weight. FNN accepts a fixed-sized vector as input. Besides it produces a fixed-sized vector as output. However, FNN is really hard to model with time series data. Cause the input and output data should be measured made over a time interval and time interval is continuous. When FNN is a directed cycle, we call it as Recurrent Neural Network (RNN). Unfolding a recurrent computation of cyclic FNN is the basic idea of RNN. Therefore, we can confirm that RNN can be a deep neural network. RNN is used in applications such as handwriting analysis, video analysis, translation, and other interpretation of various human tasks. RNN still uses feedforward and backpropagation [6]. Like FNN, RNN allows a recurrent hidden state whose activation at each time is dependent on that of the previous time (cycle). In other words, it allows node to form cycles, create the potentially for storage of information within the network. The architecture of simple RNN is shown in Figure 1. Input layer, hidden layer, and output layer are denoted by x_t,h_t, and y_t respectively. And W, U, and V are corresponding to weight matrices of input, hidden, and output layer.

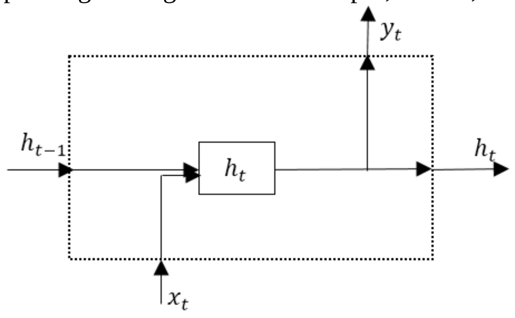


Figure 1. Simple Recurrent Neural Network

We need to compute the values of hidden layer h_t and output layer y_t in formulas as followings:

$$h_t = \sigma(W * x_t + U * h_{t-1}) \qquad (1)$$

$$y_t = f(V * h_t) \qquad (2)$$

However, RNN model is difficult to train [7]. Cho et al. in 2014 [8] proposed the method, named as GRU to overcome this problem. When we use GRU for training, we can address vanishing gradient problem. In hidden layer, we replace hidden node by GRU node. Each GRU node consists of two gates that are described in Figure 2, update gate z_t and reset gate r_t. Update gate decides how much unit updates its activation or content. And reset gate allows to forget previously computed states.

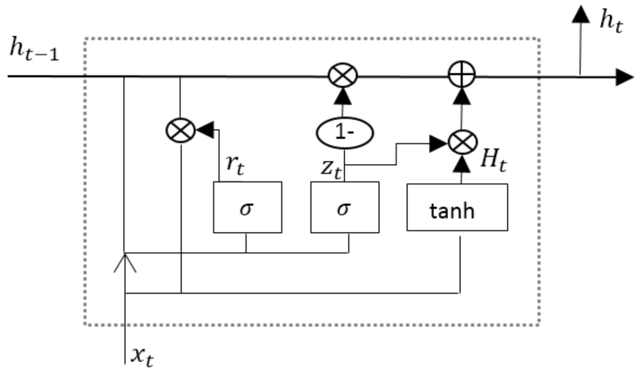


Figure 2. Gated Recurrent Unit

In the GRU-RNN we use model parameters including x_t which is the input at time and weight matrices are denoted by W_z, W_r, W_H. We need to calculate at these gates in following equations:

$$z_t = \sigma(W_z * x_t + W_z * h_{t-1}) \quad (3)$$

$$z_t = \sigma(W_r * x_t + W_r * h_{t-1}) \quad (4)$$

$$H_t = tanh\big(W_h * x_t + W_h * (r_t * h_{t-1})\big) \quad (5)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * H_t \quad (6)$$

4. Dataset

KDD cup 99 dataset is one of the famous dataset which is used in IDS by many researchers. It is prepared by Stolfo et al. and built based on the data captured in DARPA'98 IDS evaluation program. DARPA Intrusion detection evaluation program was prepared the KDD Cup 1999 Intrusion detection contest data [15] by MIT Licoln Laboratory.

Although the dataset was born long time ago, there are many researchers implement on this dataset for their IDS. This dataset includes approximately five million instances as training data and two million instances for testing data. Each of instances in this dataset consists of 41 features that shows a connection and is labeled as either normal or an attack. In training and testing data, there are 38 different types of attacks. These types attack belong to four categories which are named DoS (Denial of Service Attack), R2L (Remote to Local Attack), U2R (User to Root Attack), and Probe [17]. Each attack consists of many small attacks. The training dataset have 22 types of attacks is shown in Table 1. And Table 2 shows 37 types of attacks for testing dataset.

Table 1. List of attacks for traing dataset

Name of category	*Name of attacks*
DoS	back, land, neptune, pod, smurf, teardrop
Proble	ipsweep, nmap, portsweep, satan
R2L	ftpwrite, guespasswd, imap, multihop, phf, spy, warezclient, Warezmaster
U2R	bufferiverflow, loadmodule, perl, rootkit

Table 2. List of attacks for testing dataset

Name of category	*Name of attacks*
DoS	Apache2, back, land, mailbomb, Neptune, pod, processtable, smurf, teardrop, teardrop
Proble	ipsweep, mscan, nmap, portsweep, saint, satan
R2L	ftpwrite, guespasswd, httptunnel, imap, multihop, named, phf, sendmail, snmpgetattack, warezmaster, xlock, xsnoop
U2R	bufferiverflow, loadmodule, perl, ps, rootkit, snmpguess, sqlattack, worm, xterm

DoS is a denying legitimate requests to a system. U2R is an unauthorized access to local super user (root) privileges. R2L is an unauthorized access from a remote machine. Probing (Probe) is a surveillance and other probing. DoS and Probe attacks can easy separate from normal activities cause they come with greater frequency. In contrast, U2R and R2L attacks are difficult to obtain detection rate since they are embedded in the data portions of the packet.

In order to perform our model, we generate a new training and testing dataset from the original dataset. We extract 3, 00 instances from each attack category except U2R attack with 20 signals. Besides, we use 1,000 normal instances as well. For training this model we generated a dataset by extracting instances from KDD Cup 1999 dataset. For testing this model we made 10 tests dataset to measure performance. Till now, we have mentioned how to generate the training and testing data in our previous work [9].

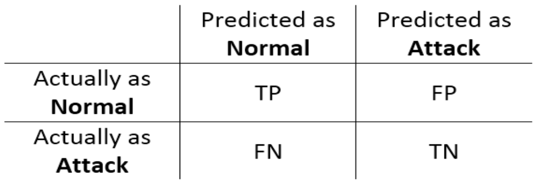
5. Experiment

In this section, we perform two experiments. The first experiment is about searching hyper-parameter values to get the best performances of IDS. The second one is about evaluating the classification performance with the hyper-parameter values are chosen in the first experiment. Our experiments are implemented as the environment in below:

- CPU: Intel ® CoreTM i7-4790 CPU @3.60GHz
- GPU: NIVIA GeForce GTX 750
- RAM: 8GB
- OS: Windows 7

5.1. Evaluation Metric

Generally we use confusion matrix to measurement classification performance. Detection Rate (DR) and False Alarm Rate (FAR) are particularly used to evaluate the performance of IDS. The ratio of intrusion instances detected belongs to DR. FAR shows the ratio of misclassified normal attack. Figure 3 describes about confusion matrix.



	Predicted as **Normal**	Predicted as **Attack**
Actually as **Normal**	TP	FP
Actually as **Attack**	FN	TN

Figure 3. Confusion Matrix

Where,

- TP (True Positive) is the number of being predicted as Normal while they actually were Normal.
- FP (False Positive) is the number of being predicted as Attack while they actually were Normal.

- FN (False Negative) is the number of being predicted as Normal while they actually were Attack.
- TN (True Negative) is the number of being predicted as Attack while they actually were Attack.

We calculate the Precision, Recall, and Accuracy with the following equations in below:

$$recall = \frac{TP}{TP + FN} = DR \quad (7)$$

$$precision = \frac{TP}{TP + FP} \quad (8)$$

$$accuracy = \frac{TP + TN}{(TP + FP) + (FN + TN)} \quad (9)$$

Besides the metric of IDS evaluation as known FAR is computed by the formular:

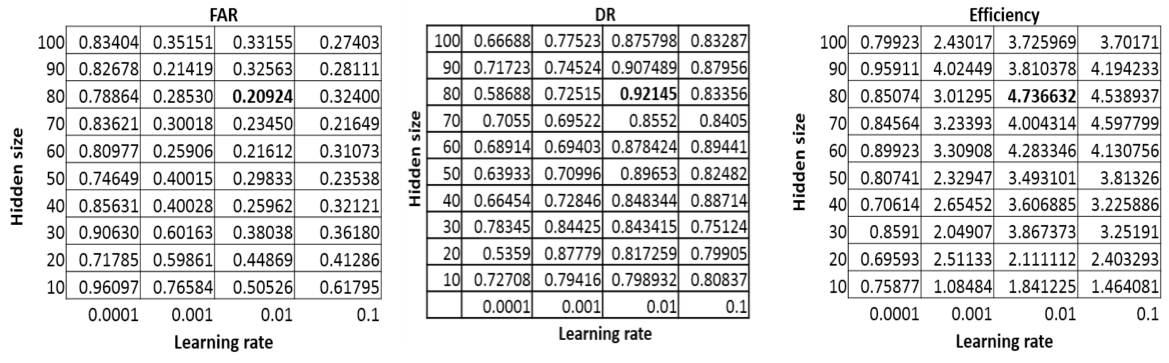
$$FAR = \frac{FP}{FP + TN} \quad (10)$$

By calculating the efficiency, it is proved that the performance grows better when DR increases and FAR decreases. And efficiency is comptuted by the equations in below:

$$Efficiency = \frac{DR}{FAR} \quad (11)$$

5.2. Hyperparameter setup

The classification performance of GRU RNN model depends on hyperparameters. In 2015, Klaus Greff et al. [16] analyzed the impact of hyperparameters for training model. Even there are many hyperparameters, however, learning rate and hidden layer size have great effect on the performance of training model. These hyperparameters are confirmed to be the most effective to get higher performance of classifying by Greff's experiments. Therefore, we perform two experiments to search these values for our model in this works. First of all, we find the value of learning rate. Then, we search the hidden layer size value base on the learning rate is chosen. We setup the values of learning rate and hidden size to experiment with the discrete values, which are [0.0001; 0.001; 0.01; 0.1] and [10; 20; 30; 40; 50; 60; 70; 80; 90; 100] respectively. We need to choose the optimized values which can lead the most effective result to get the high classification performance on IDS. For getting the accuracy values, we have calculated the values of FAR, DR and efficiency on 10 tests. The results in details are shown in Figure 4(a), Figure 4(b), and Figure 4(c).



FAR

Hidden size				
100	0.83404	0.35151	0.33155	0.27403
90	0.82678	0.21419	0.32563	0.28111
80	0.78864	0.28530	**0.20924**	0.32400
70	0.83621	0.30018	0.23450	0.21649
60	0.80977	0.25906	0.21612	0.31073
50	0.74649	0.40015	0.29833	0.23538
40	0.85631	0.40028	0.25962	0.32121
30	0.90630	0.60163	0.38038	0.36180
20	0.71785	0.59861	0.44869	0.41286
10	0.96097	0.76584	0.50526	0.61795
Learning rate	0.0001	0.001	0.01	0.1

(a)

DR

Hidden size				
100	0.66688	0.77523	0.875798	0.83287
90	0.71723	0.74524	0.907489	0.87956
80	0.58688	0.72515	**0.92145**	0.83356
70	0.7055	0.69522	0.8552	0.8405
60	0.68914	0.69403	0.878424	0.89441
50	0.63933	0.70996	0.89653	0.82482
40	0.66454	0.72846	0.848344	0.88714
30	0.78345	0.84425	0.843415	0.75124
20	0.5359	0.87779	0.817259	0.79905
10	0.72708	0.79416	0.798932	0.80837
Learning rate	0.0001	0.001	0.01	0.1

(b)

Efficiency

Hidden size				
100	0.79923	2.43017	3.725969	3.70171
90	0.95911	4.02449	3.810378	4.194233
80	0.85074	3.01295	**4.736632**	4.538937
70	0.84564	3.23393	4.004314	4.597799
60	0.89923	3.30908	4.283346	4.130756
50	0.80741	2.32947	3.493101	3.81326
40	0.70614	2.65452	3.606885	3.225886
30	0.8591	2.04907	3.867373	3.25191
20	0.69593	2.51133	2.111112	2.403293
10	0.75877	1.08484	1.841225	1.464081
Learning rate	0.0001	0.001	0.01	0.1

(c)

Figure 4. FAR, DR, and Efficiency of IDS

5.2.1. Searching learning rate value

From Figure 4(a), Figure 4(b), and Figure 4(c) we choose the learning rate =0.01 because the two reasons. First of all, we have achieved the highest average of DR and Efficiency which are 86.43% and 3.4062 respectively. Furthermore, we have got the smallest average of FAR, 0.3209. Table 3 shows the results of the various learning rate and the bold one is the result what we have chosen for the optimization performance.

Table 3. The average of FAR, DR, Efficiency

Learning rate	*FAR*	*DR*	*Efficiency*
0.1	0.336	0.395	0.794
0.01	**0.3209**	**0.8643**	**3.4062**
0.001	0,4177	0.7589	2.6640
0.0001	0.8283	0.6716	0.8181

Second of all, we choose minimum value for FAR and maximum values for DR and Efficiency per each learning rate values. In Table 4, we find 20.094% is the smallest value of FAR and the biggest values of DR and Efficiency are 92.145% and 4.736632 respectively with learning rate = 0.01.

Table 4. The minimum value of FAR and maximum values of DR, Efficiency as follow learning rate

Learning rate	*FAR*	*DR*	*Efficiency*
0.1	0.21649	0.89441	4.597799
0.01	**0.20924**	**0.92145**	**4.736632**
0.001	0.21419	0.87779	4.02449
0.0001	0.71785	0.78345	0.95911

5.2.2. Searching hidden layer size value

From Section 5.2.1 we choose the value of learning rate is 0.01, which is the most effective to GRU model. Hence, we need to find the suitable value of hidden size corresponding to this learning rate. In Figure 7, we observe that the DR's values rise up when we increase the number of hidden size from 10 to 80. However DR's values decrease from the hidden sizes are larger than 80. Furthermore, the trend of FAR also decreases when we increase in hidden size from 10 to 80. This trends also increases when learning rate is larger 80 as well. Through that indicators it is obvious that we can find the best of hidden layer size is 80, cause the maximum value of DR is approximately 92.145% and minimum value of FAR is 0.20924121 at this hidden layer size.

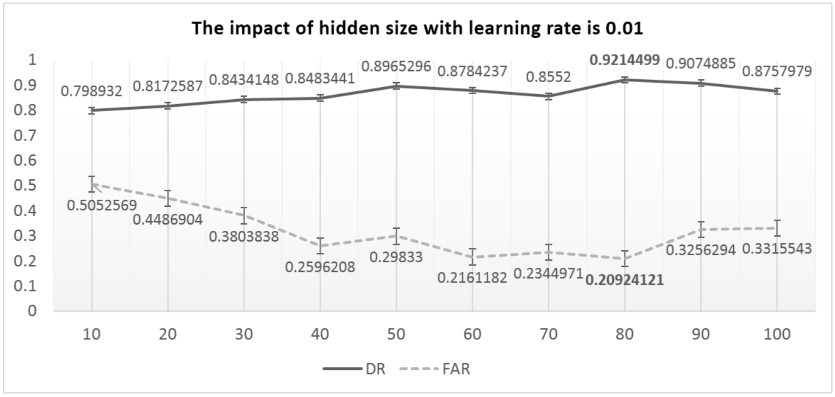


Figure 5. The impact of hidden layer size with learning rate is 0.01

5.3. Experimental results

As being mention in Section 5.2, we set the hyperparameters for our training model with learning rate = 0.01 and hidden layer size = 80. For testing, we generate 10 tests which are selected from *kddup.data.txt*. The classification performance of IDS is shown in Figure 6.

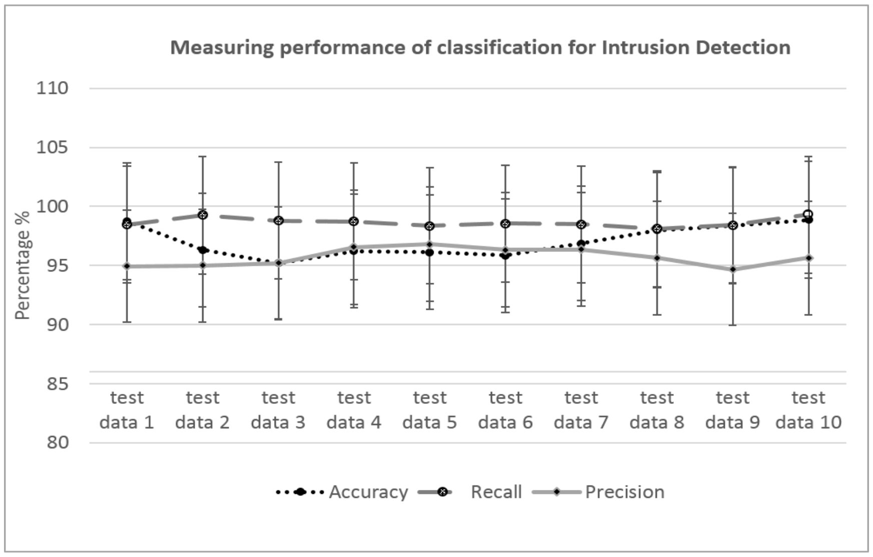


Figure 6. The result of classification performance for IDS

Particularly, we observe the result of our model which is shown in details on Table 5.

Table 5. Classification performance of IDS

Test	***Precison***	***Recall***	***Accuracy***	***FAR***	***Effiency***
Test 1	0.949402	0.984747	0.9876	0.1000329	9.8442312
Test 2	0.950037	0.992615	0.9632	0.1000003	9.926102
Test 3	0.952287	0.987946	0.952	0.1000005	9.8794106
Test 4	0.965629	0.987268	0.9624	0.1000611	9.8666515
Test 5	0.967958	0.983594	0.9614	0.1000277	9.8332162
Test 6	0.963415	0.985529	0.9584	0.1000121	9.8540977
Test 7	0.963766	0.984887	0.9686	0.1000718	9.8418036
Test 8	0.956468	0.981223	0.98	0.1000195	9.810317
Test 9	0.946689	0.984308	0.9838	0.1010018	9.745453
Test 10	0.956416	0.992922	0.9886	0.1001001	9.9192948

Moreover we summarize the result of performance which is applied GRU RNN model at some criteria such as the best, the worst, and the average in Table 6.

Table 6. Some cases of classification performance of IDS

Criteria	***Precison***	***Recall***	***Accuracy***	***FAR***	***Effiency***
Best	0.967958	0.992922	0.9886	0.1000003	9.9291902
Worst	0.946689	0.981223	0.952	0.1010018	9.714909
Average	0.9572067	0.9865039	0.9706	0.1001328	9.8520596

Furthermore, we compare with two model which are applied to the IDS regarding the average percentages of each attack detection. LSTM RNN was the previously presented approach in our previous work [25] and GRU RNN is our newly updated approach in this work. The result is shown in details in Figure 7. There are two for the best detecting attacks, DoS and Normal with the percentage are 98.12% and 96.32% respectively. U2R attack is never detected because only 30 instances of U2R are used for training model. Although the results of Probe and R2L attack of our model are better than LSTM RNN's, however, these results need to be improving in the future.

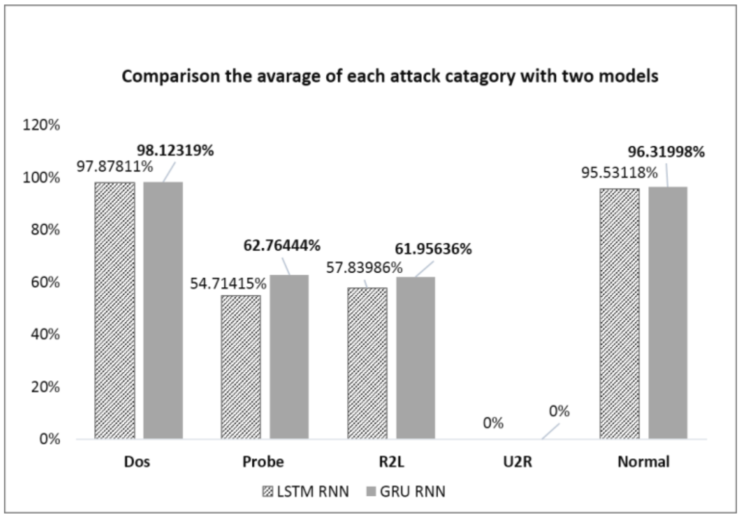


Figure 7. The comparison the average of each attack category with two models

In addition, we compare our result to the previous other IDS classifiers and the result is shown in Table 7. Even FAR is higher than some algorithms, the accuracy, DR and precision of our model are can be regarded as the best.

Table 7. Comparison between classifying algorithms on IDS

Algorithm	***Precison** (%)*	***DR** (%)*	***Accuracy** (%)*	***FAR** (%)*
FNN [18]	92.47	86.89	97.35	2.65
GNNN [18]	87.08	59.12	93.05	12.46
RBNN [18]	69.56	69.83	93.05	6.95
KNN [22]	-	91	-	8
Fuzzy with associate rule [23]	-	91	-	3.34
Jordan ANN [24]	-	62.9	-	37.09
SVM [19]	70	-	95.7	-
KMean-KNN [20]	98	98.68	93.55	47.9
RNN with Hessian-free [9]	-	95.37	-	2.1
LSTM RNN [21]	-	**98.88**	96.93	10.04
GRU RNN	**95.72**	***98.65***	**97.06**	**10.01**

6. Conclusion

In this paper, we built a new IDS classifier by applying Gated Recurrent Neural Network Recurrent Neural Network as effectively. The results of our experiment demonstrates that our approach outperform classification performance than other previous models. The detection rate was 98.65% and False Alarm Rate was 10.01%. The classification accuracy of each attack was so high except for Probe and R2L. However, these results are better than our previous results with LSTM RNN model on the same training and testing data. By comparing our model with other classifier algorithms on IDS, we found that GRU RNN model can be the best intrusion detection classifier.

7. Acknowledgments

This work is supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (No. 10043907, Development of high performance IoT device and Open Platform with Intelligent Software).

8. References

[1] Bai, Yuebin, and Hidetsune Kobayashi, "Intrusion detection systems: technology and development", AINA 2003, 17th International Conference on. IEEE, 2003.
[2] H. Kayacik, A. Zincir-Heywood, and M. Heywood, "A hierarchical SOM-based intrusion detection system", In Proc. Elsevier Engineering Application of Artificial Intelligence, pp.439-451, 2007.
[3] J. Shum and H.A. Malki, "Network intrusion detection system using neural network", In Proc. IEEE Fourth Int. Conference on Natural Computation, pp.242-246, 2008.
[4] Mukkamala, Srinivas, Andrew H. Sung, and Ajith Abraham, "Intrusion detection using ensemble of soft computing paradigms, Intelligent Systems Design and Applications", Springer Berlin Heidelberg, pp.239-248, 2003.
[5] J.-S. Xue, J.-Z. Sun, and X. Zhang, "Recurrent network in network intrusion detection system", In Machine Learning and Cybernetics, Proceedings, 2004.
[6] A. Graves, "Supervised Sequence Labelling with Recurrent Neural Networks", Studies in Computational Intelligence. Springer, 2012.
[7] Bengio, Yoshua, S. Patrice, and F.Paolo, "Learning long-term dependencies with gradient descent is difficult", Neural Networks, IEEE Transactions on 5.2, pp.157-166, 1994.
[8] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches", arXiv preprint arXiv: 1409.1259, 2014.
[9] K. Jihyun, K. Howon, "Applying Recurrent Neural Network to Intrusion Detection with Hessian Free Optimization", WISA, 2015.
[10] X. B. Li, "A salable decision tree system and its application in patter recognition and intrusion detection", Decision Support Systems, 41, pp.112-130, 2005.
[11] Y. Yu, J.R. Wang, J. Zhang, "Researches on intrusion detection system based on RBF and Elman hybrid neural network", Microelectronics & Computer (in Chinese), 8, pp.154-157, 2009.
[12] D. Zhang, F. Ren, K. Zhao, "A SYM-based system for on-line unsupervised intrusion detection", Journal of Jilin University (Science Edition) (in Chinese), 2, pp. 323-328, 2009.
[13] H.L. Guo, Y. Tan, D.L. Zhang, "Application of genetic algorithm in rule extraction of intrusion detection", Journal of Harbin Institute of Technology (in Chinese), l, pp.248-250, 2009.
[14] Muda, Z., Yassin, W., Sulaiman, M. N., & Udzir, N. I. "Intrusion Detection Based On K-Means Clustering and Naïve Bayes Classification", (CITA 11), 2011 7th IEEE International Conference on In Information Technology in Asia. pp. 1-6, 2011.
[15] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
[16] K. Greff, et al., "LSTM: A Search Space Odyssey", arXiv preprint arXiv: 1503. 04069, 2015.
[17] S. Devaraju, S. Ramakrishnan, "Performance Analysis of Intrusion Detection System Using Various Neural Network Classifiers", International Conference on International Conference on Recent Trends in Information Technology, pp. 1033-1038, 2011.
[18] S. Devaraju, S. Ramakrishnan, "Performance comparison for intrusion detection system using neural network with KDD dataset", ICTACT Journal on Soft Computing, 2014.
[19] S-J Horng, M-Y Su and Y-H Chen, "A novel intrusion detection system based on hierarchical clustering and support vector machines", Expert Systems with Applications, 38:306–313, 2011.
[20] C. F. Tsai, and C.Y Lin, "A triangle area-based nearest neighbors approach to intrusion detection", Pattern Recognition, 43(1):222-229, 2010.
[21] K. Jihyun, K. Jaehyun, L.T.T. Huong, K. Howon, "Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection", International Conference on Platform Technology and Service, 2015.
[22] Han, Sang-Jun, and Sung-Bae Cho, "Detecting intrusion with rule-based integration of multiple models", Computers & Security 22.7, pp.613-623, 2003.

[23] Tajbakhsh et al., “Intrusion detection using fuzzy association rules”, Applied Soft Computing 9.2, pp.462-469, 2009.
[24] Beghdad, Rachid, “Training all the KDD data set to classify and detect attacks”, Neural Network World 17.2, p.81, 2007.