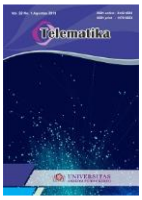

Available online at :
http://ejournal.amikompurwokerto.ac.id/index.php/telematika/

Telematika

Accreditated SINTA "2" Kemenristek/BRIN, No. 85/M/KPT/2020

UNIVERSITAS
AMIKOM PURWOKERTO

Stacked LSTM-GRU Model for Traffic Anomalies Detection

Omar Muhammad Altoumi Alsyaibani[1], Ema Utami[2], Suwanto Raharjo, and Anggit Dwi Hartanto[3]

[1,2,4]Department Master of Informatics, Universitas AMIKOM Yogyakarta
[3]Department Informatics Institut Sains & Teknologi AKPRIND Yogyakarta
E-mail: omar.muhammad.aa@gmail.com[1], ema.u@amikom.ac.id[2], wa2n@akrpind.ac.id[3]
anggit@amikom.ac.id[4]

ARTICLE INFO

History of the article:
Received December 17, 2021
Revised June 1, 2022
Accepted July 18, 2022

Keywords:
LSTM, GRU, IDS, ModelCheckPoint, Bidirectional Layer

Correspondece:
E-mail:
omar.muhammad.aa@gmail.com

ABSTRACT

This study aims to improve the accuracy of the intrusion detection system model. It focused on LSTM and GRU methods proposed by several previous studies. The bidirectional layer was also tested to see if it improves model performance. Dataset used in the study was CIC IDS 2017. The dataset was divided into 3 parts, for training, validation, and testing purposes. Validation data was used to evaluate model performance in every training iteration. It helped to make the model would not overfit the training data. Furthermore, Dropout layer and L2 regularization were also added to the model architecture. The training model was done in a binary classification approach with a learning rate of 0.0001. We found that the stacked method reached accuracy 98.1087% in 100 iteration training. This result is slightly higher than LSTM, GRU, Bidirectional LSTM, and Bidirectional GRU. The method which contains LSTM layer performed its best accuracy using Tanh activation. Differently, GRU and Bidirectional GRU performed the best performance with Lrelu and Prelu activation function, respectively. All models could reach the plateau in the first 20 iterations, while in the next 80 iterations the model performance still could be fluctuately improved. Even though the model already reached the plateau in 20 iteration training, it is still possible for the model to slowly improve by using a small learning rate and by implementing Dropout layer and L2 regularization. Fluctuation of model performance implies that the highest model performance was not always reached in the last training iteration. ModelCheckPoint could help to overcome the issue. In addition, the Bidirectional layer increased the complexity of the model which certainly increased training duration. The bidirectional layer improved the performance of the GRU method, but it did not improve the performance LSTM method.

INTRODUCTION

Nowadays, Internet has become the backbone of activities in various aspects of life. The COVID-19 pandemic is forcing us to do activities online, thus the need for an Internet connection is increasing. With the increase in people's activities on the Internet, it certainly makes network security aspects important to pay attention to. Based on data released by (CyberEdge Group, 2021) that 86.2% of organizations have experienced a successful attack at least once. This number increased significantly when compared to previous years. For example, in 2020 it only reached 80.7% and in 2019 it reached 78.0%.

Intrusion Detection System (IDS) is a system which capable of detecting attacks or intrusions on the network (Bace & Mell, 2001). Some companies use IDS as additional system to support firewall in their network (Mazini et al., 2019). Traditional IDS uses a signature database to be able to distinguish between normal traffic and attack (Bhuyan et al., 2014). If a type of traffic is not in the signature database, the IDS will not be able to recognize the traffic (Liu & Lang, 2019). Therefore, researchers are using artificial intelligence such as using machine learning or deep learning in developing IDS.

Several studies have used deep learning methods in developing IDS models. For instance, the study conducted by the authors (Al-Emadi et al., 2020) compared the performance of the Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU) and Convolutional Neural Network (CNN) methods on NSL-KDD dataset. Each method was trained up to 1000 epochs. Their experimental results showed that CNN, LSTM and GRU reached 97.01%, 81.60% and 50.25% of accuracy, respectively. Authors (Al-Emadi et al., 2020) mentioned that increasing the number of hidden layers will increase the training duration and usage of machine resources, but will not increase the performance of the model. At the end, the authors revealed that the CNN method requires a longer training duration than the other two models.

Similarly, the authors (Andalib & Vakili, 2020) examined the performance of GRU and CNN methods. They also compared the methods to Random Forest (RF) method. To solve the vanishing gradient problem in GRU, the authors implemented the Leaky Relu and Exponential Linear Unit (ELU) activation functions. The models were trained for 100 epochs. The authors (Andalib & Vakili, 2020) stated that Adam's optimization function produces better performance compared to SGD, RMSProp, Adagrad and Adadelta. The accuracy produced by the model is very promising which reaches 99.76% on the GRU method.

Authors (Ramasamy & Eric, 2022) tried to improve CNN performance by stacking it with bagging method. This method was also combined with Improved Inertia Weight Based Dragon Fly Optimizer (IIWDFO) as feature selection method which resulted 7 selected features. The method was tested against NSL-KDD dataset. The accuracy of proposed method reached arracuray 98.71%.

RNN family methods were investigated in the study (Le et al., 2019). The study used NSL-KDD and ISCX as datasets. Result of their study showed that LSTM reached 92% and 97.5% where GRU reached 91.8% and 97.08% on NSL-KDD and ISCX datasets respectively. RNN method only reached 89.6% and 94.75% on the same datasets. LSTM method produced the highest result compared to other methods and RNN method yields the lowest result. The proposed method had higher result than previous result proposed by other authors.

The result of studies (Al-Emadi et al., 2020), (Andalib & Vakili, 2020), (Ramasamy & Eric, 2022), and (Le et al., 2019) were very promising and need to be continued on the latest IDS dataset, especially bigger size dataset as stated by authors (Le et al., 2019) in their paper's conclusion. Thus, this study intended to examine deep learning methods CIC IDS 2017 dataset. Furthermore, several studies stacked deep learning methods with other methods. For instance, the authors (Ramasamy & Eric, 2022) successfully increased CNN performance by stacking it with bagging method. Authors (Bhuvaneswari Amma et al., 2021) stacked GRU method with Autoencoder. Feature extraction from training data was carried out by Autoencoder, which was then processed by GRU that became the hidden layer. The proposed method was measured on the NSL-KDD dataset. As a comparison, the authors (Bhuvaneswari Amma et al., 2021) also used RNN, LSTM and GRU methods. The result of the research shows that the proposed method could improve accuracy when compared to the other three methods. Furthermore, the GRU method has a shorter training duration than LSTM and RNN.

Differently, authors (Lohiya & Thakkar, 2021) combined AntiRectifier layer with Deep Neural Network (DNN) to improve DNN accuracy. In addition, the use of the Dropout layer could also increase the accuracy of DNN when compared to DNN without a Dropout layer. However, the study (Lohiya &

Thakkar, 2021) did not use of L2 Regularization which also has the same function as the Dropout layer, which is to prevent the model from overfitting.

In the study (Khan, 2021), RNN method was stacked with CNN method. Performance of Stochastic Gradient Descent (SGD), Adam, RMSProp and Adamax were compared in the study to find the best optimization function. They also tested several learning rates to get the highest result. The research resulted in a pretty promising performance, reaching F1 Score of 97.6%. Similar study conducted by authors (Kim et al., 2020) stacked LSTM with CNN method. After the data was processed by the LSTM layer, the data was then entered into the 8 Dense layers which also performed as hidden layer. To avoid overfitting, the authors added a Dropout layer after each Dense layer which followed by a Leaky Relu layer. In the output layer, the Dense layer was implemented over with the Sigmoid activation function. The authors (Kim et al., 2020) chose Adam as optimization function. The accuracy produced by the model was quite high, reaching 98.07%, 91.52% and 93.00% in the KF-ISAC, CSIC-2010 and CIC IDS 2017 datasets, respectively.

Furthermore, LSTM method was combined with Dense by authors (Nayyar et al., 2020). They implemented 5 LSTM layers and 7 Dense layers. Tanh was applied as an activation function in the LSTM layer and Relu as an activation function in the Dense Layer. The study (Nayyar et al., 2020) was conducted using a binary classification approach so that the Sigmoid activation function was implemented in the output layer. The accuracy of the model was quite promising, reaching 96.7%. The next researchers are suggested to combine LSTM layer with other methods.

Lastly, authors (Xu et al., 2018) combined the LSTM and GRU methods with Bidirectional layer and MLP. The performance produced by each combination of methods is quite promising with the highest accuracy produced by the Bidirectional GRU method combined with MLP, both on the KDD99 dataset and on the NSL-KDD dataset. The model configuration for the batch size used the default value and the learning rate value of 0.01, which is 10 times greater than the default value. The large value of this learning rate will certainly speed up the training process with a fairly high risk of overfit.

All stacked methods proposed by authors (Ramasamy & Eric, 2022), (Bhuvaneswari Amma et al., 2021), (Lohiya & Thakkar, 2021), (Khan, 2021), (Kim et al., 2020), (Nayyar et al., 2020), and (Xu et al., 2018) produced surprisingly high accuracy on various datasets. Thus, this study proposed stacked method as well. Even though LSTM layer was proposed as hidden layer in studies (Khan, 2021) and (Kim et al., 2020), but studies (Nayyar et al., 2020) and (Xu et al., 2018) also found that LSTM layer can be very excellent input layer for other methods. On the other hand, GRU method was also performed pretty well as hidden layer in study (Bhuvaneswari Amma et al., 2021). In this study, stacked LSTM-GRU method was proposed to be compared with non-stacked methods. The experiment was conducted on CIC IDS 2017 dataset. As additional, methods which combined with Bidirectional layer were measured and compared as well.

The stacked LSTM-GRU method has been used by several authors in different research areas. Authors (Ni & Cao, 2020) used this method to perform sentiment analysis on the IMDB dataset. Authors (Wang et al., 2019) used this method to make predictions on the Ball Screw. Authors (Muhammad et al., 2020) used this method to predict water prices. And lastly, authors (Zhu et al., 2021) used this method to identify conditions in the heating process in steel companies. Result of their studies shows that the stacked LSTM-GRU method performed high accuracy in various tasks. Thus, we believe that this method has great potential to achieve high performance in this study as well. GRU-LSTM method was also tested for comparison purposes.

METHODS

1. Dataset

In this experiment we used the CIC-IDS-2017 dataset published by the authors (Sharafaldin et al., 2018). This dataset was chosen because it has a complete attack type when compared to other widely used public datasets such as KDD Cup 99 and NSL-KDD (Thapa et al., 2020). In addition, this dataset is also in accordance with the IDS dataset framework published by (Gharib et al., 2017). On the other hand, CIC IDS 2017 dataset is bigger than previous datasets. Authors (Liu & Lang, 2019) stated that deep learning tends to perform better than machine learning when processing large datasets. Both CIC-IDS-2017 and CIC-IDS-2018 were published by the same author, but CIC-IDS-2018 is more specifically used for cloud computing environments (Leevy & Khoshgoftaar, 2020).

The CIC-IDS-2017 dataset has 15 attack types and was published in two file formats, CSV and PCAP. In this study, CSV file format dataset was used. Traffic data collection was conducted by the authors (Sharafaldin et al., 2018) using the CICFlowMeter software for five days. Normal traffic was labelled Benign and other traffics were labelled according to the attack types. Due to imbalance number of benign and attack data in this dataset, undersampling technique was implemented to handle the issue.

The dataset consists of 8 CSV files. First of all, all CSV files were merged into a Data Frame and then continued with the cleaning stage of the dataset by removing data rows which contain null values. The data were analyzed and cleaned from data that were too divergent in value. In this study, we performed binary classification experiments on the data as conducted by authors (Lohiya & Thakkar, 2021), (Khan, 2021) and (Al-Emadi et al., 2020). All benign labels were changed to 1 and all attack type labels were changed to 0. The KDD-99 and NSL-KDD datasets were divided by the dataset publisher into train data and validation data (Gamage & Samarabandu, 2020). The CIC-IDS-2017 dataset is divided into several days of data collection so that the data separation is carried out by the researcher or dataset user. Our models were tested using data that is different from the data used for training and validation to avoid score bias (Ripley, 2008) with a percentage division of 60% for training, 20% for validation and 20% for testing. Data validation is used to evaluate the model output each time the model completes one iteration of training. The output of the test on this data validation will be used by the model to update the value of each neuron.

2. Proposed Method

RNN is widely used for modeling time-series data. However, conventional RNN often experiences gradient vanishing/exploding problems during model training. This can affect the performance of the RNN in modeling long-term dependencies in time series data. To solve this problem, the authors (Hochreiter & Urgen Schmidhuber, 1997) proposed LSTM method. The LSTM tries to use multiple gates to control information to store or discard, so as to capture the long-term dependencies of the sequence.

The main components of the LSTM are the sequence input layer and the LSTM layer. The input layer imports data sequences into the network while the LSTM layer can remember the network status (Yan, 2021). Conventional LSTM only pays attention to sequential information in one direction which is forward direction. An LSTM that can understand future and past contexts is called a Bidirectional LSTM which contains a forward layer and a backward layer to process data in forward and backward directions (Chen et al., 2020). In addition to LSTM, there is also another modified form of RNN which is GRU. GRU has a simpler structure and fewer parameters when compared to LSTM (Bansal et al., 2016). Memory cell in GRU is not separate as in LSTM (Zhang & Kabuka, 2018). GRU has only reset and update gate. This allows the GRU to perform the same computational process as the LSTM, but in a faster time.

The Stacked LSTM-GRU method extracts learning features from the LSTM and GRU models. By combining them, the model will have the ability of both methods in a single model architecture. All features of the dataset will enter the input gate on the LSTM layer. LSTM layer is capable of fully read all dataset features (Zhu et al., 2021). The output of the LSTM cell will be the input for the GRU layer. GRU layer not only will process the data faster, but also process the data as good as LSTM layer. Although the LSTM layer is stacked with the GRU layer, the training duration will not increase significantly because the GRU layer uses fewer parameters than the LSTM so that the computational process becomes lighter and faster. The output of the GRU layer will be processed by the output layer. The proposed model is illustrated in Figure 1.

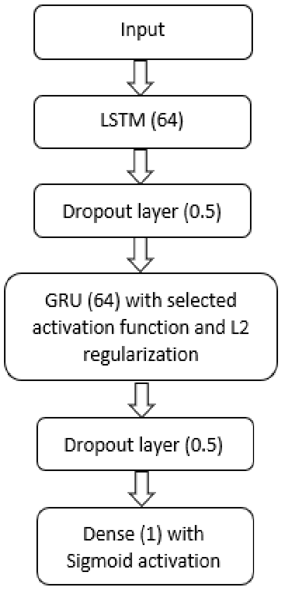

Figure 1. Performance of each model during training process

3. Model Configuration

Activation function was implemented in each method tested. Authors (Goodfellow et al., 2016) stated that activation function is a crucial component in training deep neural networks. There were 5 activation functions tested, namely Relu, Lrelu, Prelu, Softmax and Tanh. Activation function which produced the highest performance in every method was proposed to be compared in this paper.

The output layer used Dense layer with Sigmoid activation function which has an output of 0 and 1. This output corresponds to the changed label on the dataset. We implemented a Dropout layer with a parameter value of 0.5 after each main layer of the method in hope that the training model would not overfit the dataset according to the recommendations of the authors (Lohiya & Thakkar, 2021) and the authors (Kim et al., 2020). In our architecture, we also implemented L2 regularization as recommended by the authors (El-Amir & Hamdy, 2020). This implementation was done by using recurrent_regularizer variable as parameter in the neural network with value 0.01 as parameter.

http://dx.doi.org/10.35671/telematika.v15i2.1855

In experiment conducted by authors (Khan, 2021) and (Kim et al., 2020), Adam activation function was used. Adam algorithm was introduced by authors (Kingma & Ba, 2015). The authors stated that Adam is easy to be implemented and the implementation itself requires little memory requirement because the computational technique is efficient. The authors also mentioned that this algorithm is suitable for handling big data. Furthermore, the authors (Andalib & Vakili, 2020) stated that Adam optimization function has better performance than other optimization functions, so in our experiment the Adam optimization function was used. Thus, Adam was chosen to optimize model in this study.

In training the deep learning model, the learning rate determines how much the weight value changes on each neuron in the model (Moolayil, 2019). The default learning rate on the Keras framework is 0.001. Considering that a model with small learning rate will have a chance to reach higher performance in a large number of iterations of training, in this study learning rate 0.0001 was configured in training scenario.

RESULTS AND DISCUSSION

In this study, Keras, Tensor Flow and Scikit Learn framework were used to create the model and run the training. All models were trained in 100 iterations as was done in the study (Andalib & Vakili, 2020). The model for each scenario was stored using functions from the Keras framework. In addition, the model with the highest validation accuracy will be saved by ModelCheckPoint function because it is possible that validation accuracy of model in the 100th iteration to be lower than validation accuracy of model in the several previous epochs. The models with the highest validation accuracy were tested using test data and then the results were presented in this study.

Authors (Al-Emadi et al., 2020), (Andalib & Vakili, 2020), (Lohiya & Thakkar, 2021), (Khan, 2021), (Kim et al., 2020), and (Nayyar et al., 2020) trained their model using binary classification approach. Thus, in this study all of our models were trained with the same approach to simulate anomaly detection. This study used the default value for batch size parameter as done by authors (Xu et al., 2018). Every method was tested on several activation functions to get the best configuration for each method. Table 1 shows the best activation function for each method. It can be inferred from Table 1 that every method tested which contains the LSTM layer, except GRU-LSTM method, performed its best performance with Tanh activation function which is the default activation function for LSTM and GRU layer in Keras framework. Differently, GRU and GRU-LSTM performed the best performance with Lrelu activation function. Only Bidirectional GRU performed the best performance with Prelu activation function.

Table 1. Final activation function used in each method

Method	*Activation Function*
LSTM	Tanh
GRU	Lrelu
Bidirectional LSTM	Tanh
Bidirectional GRU	Prelu
LSTM-GRU	Tanh
GRU-LSTM	Lrelu

The training results of each model with the final configuration are shown in Figure 2. The Y-Axis represents validation accuracy. X-Axis represents the number of iterations of the training model. Accuracy is displayed for every 10 iterations by taking the highest accuracy in those 10 iterations. It can be concluded from Figure 2 that the increase in validation accuracy occurred in the first 20 iterations of training. In the next iterations, the accuracy of the model starts to fluctuate. The highest validation accuracy of the LSTM, GRU, Bidirectional LSTM and GRU-LSTM models were obtained in the last 10 iterations, which in iteration were 95, 98, 93 and 91, respectively. In contrast, the validation accuracy of the Bidirectional GRU and LSTM-GRU was achieved at 75 and 88 iterations, respectively.

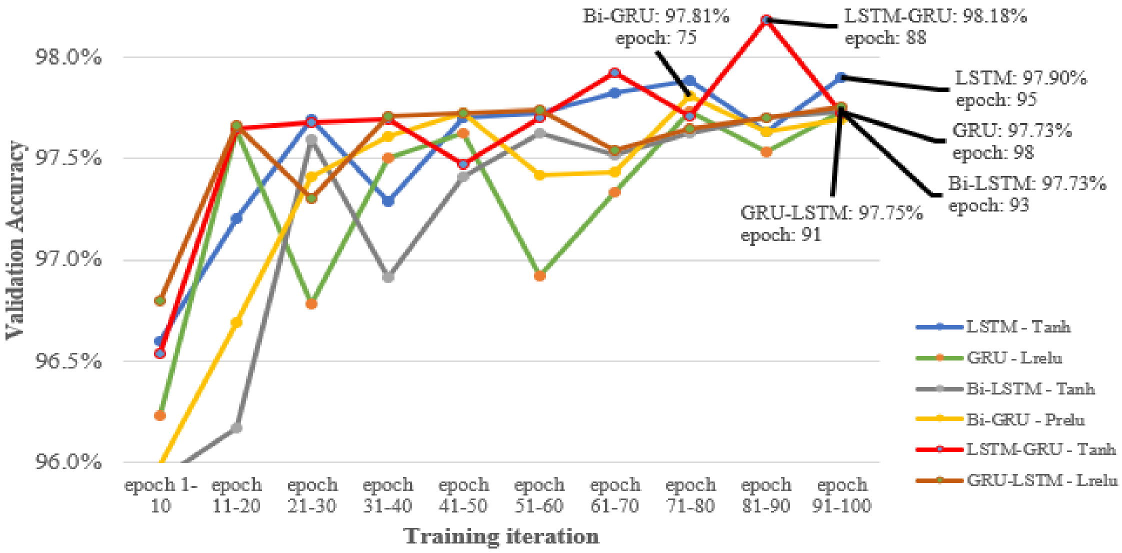

Figure 2. Performance of each model during training process

If the validation accuracy taken was from the last iteration model, then the LSTM will be the model with the highest validation accuracy. ModelCheckPoint made it possible to save the model before the training process is complete as mentioned in study (Alsyaibani et al., 2021). In this study, the compared accuracy is the highest accuracy achieved by each model during the training process. Thus, the LSTM-GRU model which achieved a validation accuracy of 98.18% in 88 training iterations became the model with the highest accuracy.

Furthermore, the maximum and the average validation accuracy for each model are presented in Table 2. It can be seen from the table that even though validation accuracy of LSTM-GRU method started quite low which was under LSTM and GRU-LSTM methods, yet the average validation accuracy of LSTM-GRU method is the highest compared to other methods. This means that in spite of its fluctuation, the validation accuracy of LSTM-GRU did not drop too low. The same behaviour was also produced by LSTM and GRU-LSTM methods. The gap of their average validation accuracy is only quite small compared to the validation accuracy of LSTM-GRU method.

Table 2. Final activation function used in each method

Method	*Maximum Validation Accuracy*	*Average Validation Accuracy*
LSTM	97.90%	97.54%
GRU	97.73%	97.30%
Bidirectional LSTM	97.73%	97.22%
Bidirectional GRU	97.81%	97.34%
LSTM-GRU	98.18%	97.63%
GRU-LSTM	97.75%	97.56%

The model with the highest validation accuracy in each method was tested using testing data. The test results are shown in Figure 3. Y-Axis represents the test accuracy of each model. The difference between validation accuracy and testing accuracy for each method is quite small because model performance during the training was measured and evaluated using validation data, not training data.

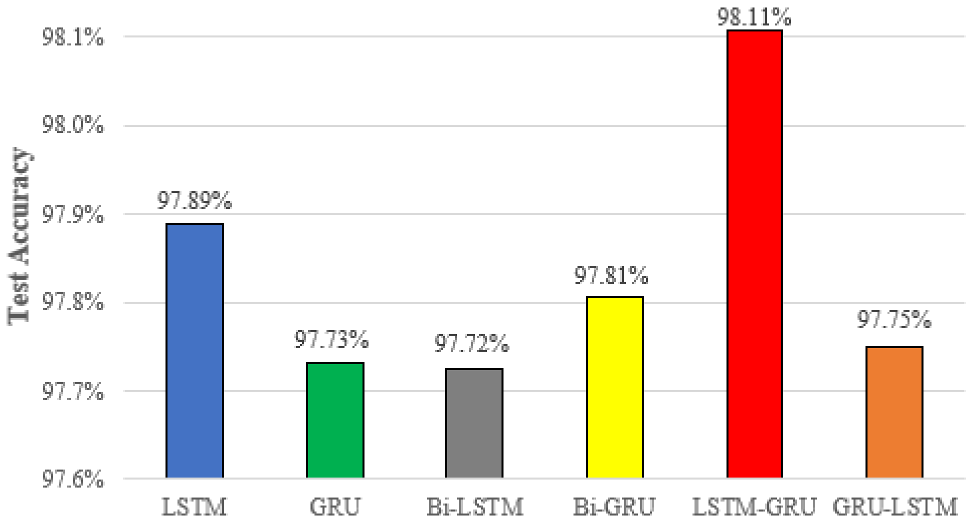

Figure 3. Test accuracy of each model

Based on Figure 3, it can be concluded that stacking the LSTM method with the GRU method could increase the accuracy of the model when compared to non-stacked LSTM or GRU methods. Unfortunately, stacking the GRU method with the LSTM method did not produce the same performance. The former results are in line with the results of studies (Ramasamy & Eric, 2022), (Bhuvaneswari Amma et al., 2021), (Lohiya & Thakkar, 2021), (Khan, 2021), (Kim et al., 2020), (Nayyar et al., 2020), and (Xu et al., 2018) where stacking method could improve model performance. Specifically, the yielded result confirmed study (Bhuvaneswari Amma et al., 2021) where GRU was used as hidden layer and study (Kim et al., 2020) where LSTM was used as the first of stacked layer. However, the difference in accuracy is not significant. The gap between accuracy of LSTM-GRU method which achieved the highest performance and accuracy of Bidirectional LSTM method which achieved the lowest performance is only 0.39%.

On the other hand, the Bidirectional LSTM and Bidirectional GRU methods required a higher training duration, due to nature of Bidirectional layer which will try to learn a sequence both in forward and backward direction (Schuster & Paliwal, 1997). Authors (Graves & Schmidhuber, 2005) stated the same fact in their paper. Experiment result of authors (Imrana et al., 2021) also revealed that Bidirectional LSTM has longer training duration than LSTM due to its computational complexity. It seemed that LSTM layer has better accuracy on the dataset when it was implemented without Bidirectional layer. On the contrary, GRU layer seemed to have higher accuracy when it was implemented with Bidirectional layer. Fact about GRU obtained in this experiment matches the result of study (Xu et al., 2018) which proposed Bidirectional GRU for building neural network architecture.

Table 3 presents the comparison of result between this study with previous studies which used the same dataset. The proposed method produced slightly higher accuracy compared to previous studies which used the same dataset and were trained in binary classification.

Table 3. Accuracy comparison with previous studies

Study	*Methods*	*Accuracy (%)*
This study	LSTM-GRU	98.11
Study (Lohiya & Thakkar, 2021)	DNN + AntiRectifier	97.49
Study (Khan, 2021)	HCRNN	97.75
Study (Kim et al., 2020)	CNN-LSTM	93.00
Study (Nayyar et al., 2020)	LSTM+DNN	96.703

CONCLUSIONS AND RECOMMENDATIONS

In this study, stacked LSTM-GRU method was proposed to create IDS model. It was found that stacking LSTM with GRU method could slightly improve model performance compared to non-stacked methods. Methods which contain LSTM layer such as LSTM, Bidirectional LSTM and stacked LSTM-GRU performed the best performance on default activation function. Differently, GRU and Bidirectional GRU performed the best performance with Lrelu and Prelu activation function, respectively.

All methods tested reached plateau in just 20 iteration training because they were optimized using Adam which can adapt its parameter to speed up model improvement. In the next 80 iterations, model performance still can fluctuately increased because this study used small learning rate value and implemented Dropout layer and L2 regularization to avoid the model from overfit. The proposed method did not achieve its best performance in the last training iteration, but in the middle of the training process. To overcome the issue, ModelCheckPoint helped to save the model which had the highest validation accuracy during training. By validating trained model in each training iteration using validation data, the accuracy of every model on test data was just slightly lower that validation accuracy.

Based on the result of this experiment, combining Bidirectional layer with LSTM layer did not increase model accuracy. On the contrary, combining Bidirectional layer with GRU layer resulted the opposite. Lastly, the accuracy of proposed method is slightly higher than the previous studies.

It is possible for further researchers to perform feature ranking on datasets with various methods in the hope of shortening the training and validation duration of model. Several newest activation functions which have promising result such in study (K. & K., 2020) have not been tested in this experiment. Most of studies, including this study, were conducted in binary classification approach, so it is necessary to increase the intensity of research in multiclass classification approach on this dataset. Lastly, the creation of a dataset with the latest types of attacks will certainly help the development of deep learning-based IDS models.

REFERENCES

Al-Emadi, S., Al-Mohannadi, A., & Al-Senaid, F. (2020). Using Deep Learning Techniques for Network Intrusion Detection. *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies, ICIoT 2020*. https://doi.org/10.1109/ICIoT48696.2020.9089524

Alsyaibani, O. M. A., Utami, E., & Hartanto, A. D. (2021). Intrusion Detection System Model Based on Gated Recurrent Unit to Detect Anomaly Traffic. *2021 4th International Conference on Information and Communications Technology (ICOIACT)*, 5–10. https://doi.org/10.1109/ICOIACT53268.2021.9564003

Andalib, A., & Vakili, V. T. (2020). An autonomous intrusion detection system using an ensemble of advanced learners. *2020 28th Iranian Conference on Electrical Engineering, ICEE 2020*. https://doi.org/10.1109/ICEE50131.2020.9260808

Bace, R., & Mell, P. (2001). NIST special publication on intrusion detection systems. In *Nist Special Publication*.

Bansal, T., Belanger, D., & McCallum, A. (2016). Ask the GRU: Multi-task learning for deep text recommendations. *RecSys 2016 - Proceedings of the 10th ACM Conference on Recommender Systems*. https://doi.org/10.1145/2959100.2959180

Bhuvaneswari Amma, N. G., Selvakumar, S., & Leela Velusamy, R. (2021). Sagru: A stacked autoencoder-based gated recurrent unit approach to intrusion detection. *Advances in Intelligent Systems and Computing*, *1177*. https://doi.org/10.1007/978-981-15-5679-1_5

Bhuyan, M. H., Bhattacharyya, D. K., & Kalita, J. K. (2014). Network anomaly detection: Methods, systems and tools. *IEEE Communications Surveys and Tutorials*, *16*(1). https://doi.org/10.1109/SURV.2013.052213.00046

Chen, Z., Jiang, C., Masood, M. K., Soh, Y. C., Wu, M., & Li, X. (2020). Deep learning for building occupancy estimation using environmental sensors. In *Studies in Computational Intelligence*. https://doi.org/10.1007/978-3-030-31760-7_11

CyberEdge Group. (2021). *2021 Cyberthreat Defense Report*. https://cyber-edge.com/cdr/

El-Amir, H., & Hamdy, M. (2020). *Deep Learning Pipeline: Building a Deep Learning Model with TensorFlow*. Apress.

Gamage, S., & Samarabandu, J. (2020). Deep learning methods in network intrusion detection: A survey and an objective comparison. *Journal of Network and Computer Applications*, *169*. https://doi.org/10.1016/j.jnca.2020.102767

Gharib, A., Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2017). An Evaluation Framework for Intrusion Detection Dataset. *ICISS 2016 - 2016 International Conference on Information Science and Security*. https://doi.org/10.1109/ICISSEC.2016.7885840

Goodfellow, I. J., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press. www.deeplearningbook.org

Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM networks. *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, *4*, 2047–2052 vol. 4. https://doi.org/10.1109/IJCNN.2005.1556215

Hochreiter, S., & Urgen Schmidhuber, J. J. (1997). Long Short-Term Memroy. In *Neural Computation*.

Imrana, Y., Xiang, Y., Ali, L., & Abdul-Rauf, Z. (2021). A bidirectional LSTM deep learning approach for intrusion detection. *Expert Systems with Applications*, *185*, 115524. https://doi.org/https://doi.org/10.1016/j.eswa.2021.115524

K., V., & K., S. (2020). Towards activation function search for long short-term model network: A differential evolution based approach. *Journal of King Saud University - Computer and Information Sciences*. https://doi.org/https://doi.org/10.1016/j.jksuci.2020.04.015

Khan, M. A. (2021). HCRNNIDS: Hybrid convolutional recurrent neural network-based network intrusion detection system. *Processes*, *9*(5). https://doi.org/10.3390/pr9050834

Kim, A., Park, M., & Lee, D. H. (2020). AI-IDS: Application of Deep Learning to Real-Time Web Intrusion Detection. *IEEE Access*, *8*, 70245–70261. https://doi.org/10.1109/ACCESS.2020.2986882

Kingma, D. P., & Ba, J. L. (2015). Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*.

Le, T.-T.-H., Kim, Y., & Kim, H. (2019). Network Intrusion Detection Based on Novel Feature Selection Model and Various Recurrent Neural Networks. *Applied Sciences*, *9*(7). https://doi.org/10.3390/app9071392

Leevy, J. L., & Khoshgoftaar, T. M. (2020). A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 Big Data. *Journal of Big Data*, *7*(1). https://doi.org/10.1186/s40537-020-00382-x

Liu, H., & Lang, B. (2019). Machine learning and deep learning methods for intrusion detection systems: A survey. In *Applied Sciences (Switzerland)*. https://doi.org/10.3390/app9204396

Lohiya, R., & Thakkar, A. (2021). Intrusion Detection Using Deep Neural Network with AntiRectifier Layer. In S. M. Thampi, J. Lloret Mauri, X. Fernando, R. Boppana, S. Geetha, & A. Sikora (Eds.), *Applied Soft Computing and Communication Networks* (pp. 89–105). Springer Singapore.

Mazini, M., Shirazi, B., & Mahdavi, I. (2019). Anomaly network-based intrusion detection system using a reliable hybrid artificial bee colony and AdaBoost algorithms. *Journal of King Saud University - Computer and Information Sciences*, *31*(4), 541–553. https://doi.org/https://doi.org/10.1016/j.jksuci.2018.03.011

Moolayil, J. (2019). *Learn Keras for Deep Neural Networks: A Fast-Track Approach to Modern Deep Learning with Python*. Apress. https://doi.org/https://doi.org/10.1007/978-1-4842-4240-7

Muhammad, A. U., Yahaya, A. S., Kamal, S. M., Adam, J. M., Muhammad, W. I., & Elsafi, A. (2020). A Hybrid Deep Stacked LSTM and GRU for Water Price Prediction. *2020 2nd International Conference on Computer and Information Sciences (ICCIS)*, 1–6. https://doi.org/10.1109/ICCIS49240.2020.9257651

Nayyar, S., Arora, S., & Singh, M. (2020). Recurrent Neural Network Based Intrusion Detection System. *2020 International Conference on Communication and Signal Processing (ICCSP)*, 136–140. https://doi.org/10.1109/ICCSP48568.2020.9182099

Ni, R., & Cao, H. (2020). Sentiment Analysis based on GloVe and LSTM-GRU. *2020 39th Chinese Control Conference (CCC)*, 7492–7497. https://doi.org/10.23919/CCC50068.2020.9188578

Ramasamy, M., & Eric, P. V. (2022). An improved deep bagging convolutional neural network classifier for efficient intrusion detection system. *Bulletin of Electrical Engineering and Informatics*, *11*(1), 405–413.

Ripley, B. D. (2008). *Pattern recognition and neural networks* (1st ed.). Cambridge University Press.

Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*. https://doi.org/10.1109/78.650093

Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSP 2018 - Proceedings of the 4th International Conference on Information Systems Security and Privacy*. https://doi.org/10.5220/0006639801080116

Thapa, N., Liu, Z., Kc, D. B., Gokaraju, B., & Roy, K. (2020). Comparison of machine learning and deep learning models for network intrusion detection systems. *Future Internet*. https://doi.org/10.3390/fi12100167

Wang, K., Huang, Y., Gong, L., Cai, C., & Zhang, Y. (2019). State-Wise LSTM-GRU Method for Ball Screw Prediction. *2019 IEEE Aerospace Conference*, 1–8. https://doi.org/10.1109/AERO.2019.8741555

Xu, C., Shen, J., Du, X., & Zhang, F. (2018). An Intrusion Detection System Using a Deep Neural Network with Gated Recurrent Units. *IEEE Access*, *6*. https://doi.org/10.1109/ACCESS.2018.2867564

Yan, W. (2021). *Computational methods for deep learning*. Springer.

Zhang, D., & Kabuka, M. R. (2018). Combining Weather Condition Data to Predict Traffic Flow: A GRU Based Deep Learning Approach. *Proceedings - 2017 IEEE 15th International Conference on Dependable, Autonomic and Secure Computing, 2017 IEEE 15th International Conference on Pervasive Intelligence and Computing, 2017 IEEE 3rd International Conference on Big Data Intelligence and Compu*. https://doi.org/10.1109/DASC-PICom-DataCom-CyberSciTec.2017.194

Zhu, Q., Cao, W., & Song, W. (2021). Multi-condition recognition method based on LSTM_GRU for heating processes. *2021 40th Chinese Control Conference (CCC)*, 6544–6549. https://doi.org/10.23919/CCC52363.2021.9550672