Attention-Based Deep Learning Modelling for Intrusion Detection

Ban AlOmar[1], Zouheir Trabelsi[2] and Firas Saidi[3]
[1,2]College of Information Technology, United Arab Emirates University, Al-Ain, UAE
[3]Department of Computer Science, SMPCS, University of Reading, UK

700039223@uaeu.ac.ae
trabelsi@uaeu.ac.ae
f.saidi@reading.ac.uk

Abstract: Cyber-attacks are becoming increasingly sophisticated, posing more significant challenges to traditional intrusion detection methods. The inability to prevent intrusions could compromise the credibility of security services, thereby putting data confidentiality, integrity, and availability at risk. In response to this problem, research has been conducted to apply deep learning (DL) models to intrusion detection, leveraging the new era of AI and the proven efficiency of DL in many fields. This study proposes a new intrusion detection system (IDS) based on DL, utilizing attention-based long short-term memory (AT-LSTM) and attention-based bidirectional LSTM (AT-BiLSTM) models. The time-series nature of network traffic data, which changes continuously over time, makes LSTM and BiLSTM particularly effective in handling intrusion detection. These models can capture long-term dependencies in the sequence of events, learn the patterns of normal network behaviour, and detect deviations from this behaviour that may indicate an intrusion. Also, the attention mechanism in the proposed models lets them make predictions based on the most important parts of the network traffic data. This is important for finding intrusions because network traffic data can have many different features, not all of which are important for finding an attack. The attention mechanism lets the models learn which features are most important for making accurate predictions, which improves their performance and efficiency. The UNSW-NB15 benchmark dataset is used in the study to measure and compare the effectiveness and reliability of the proposed system. This dataset contains normal and attack traffic data with a significant class imbalance. To address this issue, the study employs the Synthetic Minority Over-sampling Technique (SMOTE) to balance the dataset, thus reducing the risk of overfitting to the majority class and improving the model's performance in detecting attacks. The performance evaluation results demonstrate that the proposed models achieved a detection rate of over 93%, indicating high precision in detecting intrusions. By harnessing the power of deep learning, these models can learn and adapt to new threats over time, thus ensuring data confidentiality, integrity, and availability in today's interconnected world.

Keywords: Intrusion Detection, Attention Architecture, BiLSTM, LSTM, Network Attacks

1. Introduction

With the increasing frequency of threats and attacks on computer networks, ensuring the effectiveness of intrusion detection systems (IDS) has become difficult. The present-day systems are heterogeneous, dynamic, and complex, making it challenging to address multiple attacks. Based on the behaviour of intrusion detection, network intrusion detection can be divided into two categories: anomaly-based IDS and signature-based IDS (SIDS) (Zhang *et al.*, 2022). The anomaly-based approach detects new attacks by creating a model of normal activities within a system and identifying potential attacks from behaviours that deviate from the established normal behaviour pattern. In contrast, SIDS uses matching techniques to detect known attacks. Whenever a behaviour signature matches one from a previous intrusion existing in the signature database, the behaviour is classified as an intrusion (Bakhsh *et al.*, 2019).

The IDS's effectiveness is evaluated based on its ability to detect actual intrusions accurately. In practice, the IDS's efficacy is determined by its ability to minimize the false-positive rate (FPR) instead of maximizing the true-positive rate (TPR), as described in (Bakhsh *et al.*, 2019).

However, current IDS face several significant limitations. For example, anomaly detection IDS sometimes classifies rare normal behaviours as abnormal. At the same time, the diversity in the implementation mechanisms of the operating systems makes building a unified pattern library very difficult, which decreases the efficiency of the SIDS (Sarhan *et al.*, 2021).To overcome these limitations, various ML and DL techniques have been studied to design IDS. These techniques act as classifiers to determine whether events are attacks or benign occurrences and to specify the kind of attacks. When it comes to managing massive volumes of data and developing models that can be applied to different network environments and attack vectors, DL outperforms standard IDS (Mighan and Kahani, 2021).

The goal of this paper is to show how AT-LSTM and AT-BiLSTM can be used in intrusion detection. The key contributions of this paper are:

- An effective and compact deep learning model is presented that makes use of attention-based LSTM and attention-based BiLSTM architectures. Network traffic data used for intrusion detection is typically sequential, with each network packet being captured as a series of features across time. LSTM and BiLSTM networks, which are intended to handle variable-length input sequences, may efficiently simulate this sequential data.
- Synthetic Minority Oversampling Technique (SMOTE) is used as an oversampling strategy to address the problem of unequal class representation in the dataset.
- Evaluation of the proposed models' performance using the F1-score, accuracy, precision, recall, and the common UNSW2015-NB15 network security dataset.

The structure of this paper is as follows: Section 2 provides an overview of prior research related to intrusion detection using ML and DL techniques. Section 3 outlines the model's design. Model evaluation and discussion are presented in Section 4. The paper concludes with the key findings of our work in Section 5.

2. Related Works

There has been a lot of research on how ML and DL might be employed to increase intrusion detection. This section highlights the research that has applied ML and DL to intrusion detection. A stacked autoencoder network was used by the authors in (Mighan and Kahani, 2021) to extract features before the data was classified using support vector machines, random forests, decision trees, and Naive Bayes. The UNB ISCX 2012 dataset was used to validate the model, and the maximum degree of accuracy was 90.2%. The authors of (Liu, Gu and Wang, 2021) developed a binary classification model using K-means and Random Forest (RF). They used Convolutional Neural Networks (CNN) and LSTM to categorise attacks into different groups. The effectiveness of their model was evaluated on two benchmark datasets, with accuracy rates of 85.24% and 99.91%. GRU and improved LSTM were used by A. Aldallal to build an IDS (Aldallal, 2022). Using information from the CICIDS 2018 dataset, the model was evaluated. The model outperformed the benchmarks by 12.045%. The author (Xu *et al.*, 2018) used GRU and Multi-Layer Perceptron (MLP) to build an IDS model and tested its effectiveness on the KDD-99 and NSL-KDD datasets. The model achieved a prediction rate of 99.42% and 99.31% on KDD-99 and NSL-KDD, respectively. Using LSTM, the author of (Yan *et al.*, 2020) built an IDS using a stacked autoencoder. The model is tested on KDD Cup-99 dataset, demonstrating that it outperforms several conventional ML models. The author (Benaddi *et al.*, 2022) suggested a Deep Reinforcement Learning -based IDS . The NSL-KDD dataset was used to evaluate the proposed DRL-performance IDS. In (Gaifulina and Kotenko, 2021), the authors evaluated deep neural network models for anomaly detection in Internet of Things (IoT) network traffic. They evaluated the quality metrics of anomaly detection as well as the amount of time spent training the models using the UNSW-NB 15 dataset. The author of (Yang *et al.*, 2022) used an intrusion detection model with a multilayer attention mechanism for a power information network. Compared to the no-attention layer, the attention layer increased the detection rate by 1.99%. In (Wei *et al.*, 2023), the efficiency of an attention mechanism and an autoencoder for intrusion detection was investigated. The suggested model outperformed conventional machine learning techniques in tests utilising a dataset of actual CAN bus messages from in-vehicles. An attention-based LSTM neural network was also used in a study reported by (Liu *et al.*, 2020) to forecast air pollution. In (Lan *et al.*, 2020), a threshold-optimized CNN-BiLSTM-Attention technique was suggested to address the issues of class imbalance and low detection rate in minority classes. The suggested method combines the CNN-BiLSTM-Attention model with a threshold adjustment technique based on the ROC curve. The technique was tested using an industrial data set proposed by Mississippi State University in 2014, and the findings showed an accuracy of 96.7%. The author of (Laghrissi *et al.*, 2021) suggested a detection model that combines four feature reduction algorithms—Chi-Square, UMAP, principal components analysis (PCA), and mutual information with LSTM and attention mechanisms. The method was tested on the NSL-KDD dataset and acheived an accuracy of 99.09% and 98.49% for binary and multiclass classification, respectively. In (Cao *et al.*, 2021), the authors suggested a Dense Attention-LSTM (DAL) IDS. The suggested model uses a CuDNN-based LSTM network to learn time-related information about traffic, global max-pooling to compress data and enhance generalisation capabilities, attention mechanism to capture key features, dense dilated convolutions to extract underlying features of network traffic. The suggested model achieved a binary classification accuracy of 92.65% and recognised different attacks with an accuracy of 81.28%.

3. Methodology

This section describes the steps needed to build the proposed intrusion detection system. These steps include pre-processing the data, choosing a model, training the model, evaluating the model, and analysing the results.

3.1 Dataset

The UNSW-NB 15 dataset (Moustafa and Slay, 2016; Sarhan *et al.*, 2021) includes a combination of realistic normal activities and synthetic attack behaviours generated from network traffic. UNSW-NB15 is one of the most enormous and widely used benchmark datasets for evaluating intrusion detection systems (Sarhan *et al.*, 2021). The dataset has over two million network connection records and a wide range of attack types, such as backdoors, denial of service (DoS), exploits, fuzzers, generic, normal, reconnaissance, shellcode, and worms.

3.2 Data Pre-processing

For ML to work well, the data must be properly prepared. This means taking out numerical and statistical features, reducing sparsity, and making the data fit for machine learning algorithms. In the case of the UNSW-NB15 dataset, missing values were taken out, categorical features were turned into numerical values using one-hot encoding, and features were normalised using Min-Max scaling. The UNSW-NB15 dataset had a problem with a class imbalance. Some attack types were much less common than the "normal" class. Specifically, the "worms," "backdoor," "shellcode," and "analysis" attack types were the minority classes with the smallest number of samples. SMOTE was used to handle the class imbalance in the dataset. This ensures that the model gets a more balanced exposure to all classes during training and can improve the overall performance of the model (Zhang *et al.*, 2018; Waqar *et al.*, 2021).

3.3 Attention Mechanism

The attention layer is part of a neural network architecture that can be used to pay more attention to certain parts of the input sequence. The basic idea is to compute a weight for each element of the input sequence based on how relevant it is to the problem. The equations for the attention layer can be written as follows (Zhang *et al.*, 2019; Liu *et al.*, 2022).

Let $h_1, h_2, \ldots, h_T$ be the input sequence of hidden states, where $h_t \in R_d$ for t=1, 2, …, T and d is the dimensionality of the hidden state. First, the attention scores α_t between the current hidden state of the network and each element of the input sequence are calculated using the following equation:

$$\boldsymbol{\alpha_t} = \frac{\boldsymbol{exp(e_t)}}{\sum_{j=1}^{T} \boldsymbol{exp(e_j)}} \quad \textbf{(1)}$$

Where, e_t is the relevance of the t^{th} element of the input sequence, defined by some function of the current hidden state of the network and the t^{th} hidden state of the input sequence:

$$\boldsymbol{e_t} = \boldsymbol{f}\left(\boldsymbol{h_t}, \widetilde{\boldsymbol{h}}\right) \quad \textbf{(2)}$$

Where $\hat{h}$ is a summary vector of the hidden states, computed using a weighted sum:

$$\widetilde{\boldsymbol{h}} = \sum_{i=1}^{T} \boldsymbol{\beta_i h_i} \quad \textbf{(3)}$$

And β_i is a weight assigned to each hidden state h_i, computed as follows:

$$\boldsymbol{\beta_i} = \frac{\boldsymbol{exp}\left(\boldsymbol{g}(\boldsymbol{h_i})\right)}{\sum_{j=1}^{T} \boldsymbol{exp}\left(\boldsymbol{g}(\boldsymbol{h_j})\right)} \quad \textbf{(4)}$$

where g is a function that maps each hidden state to a scalar, a single-layer feed-forward neural network. The output of the attention layer is the weighted sum of the input sequence, using the attention weights as the weights:

$$\boldsymbol{c} = \sum_{t=1}^{T} \boldsymbol{\alpha_t h_t} \quad \textbf{(5)}$$

The vector c is then concatenated with the current hidden state of the network and passed through a linear layer and non-linear activation function to compute the final output of the network.

The self-attention mechanism is a type of attention mechanism that enables a model to focus on different parts of the input sequence and weigh their importance when making predictions. The input sequence is transformed into three vectors: the query vector, the key vector, and the value vector. The query vector is used to calculate the similarity between the input sequence and a particular element, the key vector is used to store the representation of the input sequence, and the value vector is used to store the output. The self-attention weight for position i is computed as:

$$\alpha_i = \frac{exp(g(h_i))}{\sum_{j=1}^{T} exp\left(g(h_j)\right)} \quad (6)$$

Where h_i is the hidden state of the input sequence at position i, and g is a function that computes the score between the query and the key. The context vector for position i is then computed as:

$$c_i = \sum_{j=1}^{T} \alpha_j h_j \quad (7)$$

This is the same as the equation for global attention, but with the self-attention weights instead of the alignment weights. The context vector *c* is then used as input to the feedforward network (Cao *et al.*, 2021).

3.4 AT-LSTM

Long-short-term memory provides gates in their basic unit. These gates can capture long-term and short-term memory along the time steps and avoid the gradient exploding and vanishing in traditional RNNs. The gates are named forget gate, input gate, and output gate. The LSTM has two memory cells: the candidate memory cell $\widehat{C_t}$, and the memory cell C_t. The memory cell C_t was proposed as a long-term memory to aggregate the relevant information throughout the time steps. The detailed structure of LSTM is presented in Figure 1. The equations for the forward pass of the LSTM cell given a time step *t* are as follows (Freire *et al.*, 2022):

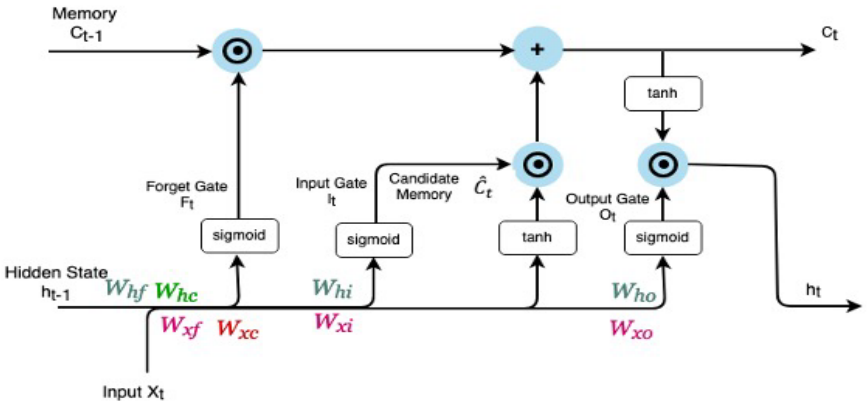

Figure 1: The structure of one LSTM neuron.

$$I_t = \sigma\left(X_t W_{xi} + h_{t-1} W_{hi} + b_i\right) \text{ (8)}$$

$$F_t = \sigma\left(X_t W_{xf} + h_{t-1} W_{hf} + b_f\right) \text{ (9)}$$

$$O_t = \sigma\left(X_t W_{xo} + h_{t-1} W_{ho} + b_o\right) \text{ (10)}$$

$$\widehat{C_t} = tanh\left(X_t W_{xc} + h_{t-1} W_{hc} + b_c\right) \text{ (11)}$$

$$C_t = F_t \odot C_{t-1} + I_t \odot \widehat{C_t} \text{ (12)}$$

$$h_t = O_t \odot tan\, h(C_t) \text{ (13)}$$

$$\hat{y}_t = f(W_h h_t + b_h) \text{ (14)}$$

Where I_t, F_t, O_t, C_t, h_t represent the vector value of the LSTM node at time t in the input gate, the forget gate, the output gate, the cell state, and the output of the LSTM block respectively. W_{xi}, W_{xf}, W_{xo} are the weights between the input layer and the gates at a timestep t. W_{hf}, W_{hi}, W_{ho} are the weights between the hidden recurrent layer and the gates. W_{ci}, W_{cf}, W_{co} are the weights between the cell state and the gates. b_i, b_f, b_o are the biases of the input gate, the forget gate, and the output gate. The output, weights, bias of the activation function f are represented by $\hat{y}, W_h, b_h$,respectively. The set of activation functions consists of the sigmoid function (σ), the element-wise multiplication ($\odot$) and the hyperbolic activation function tanh.

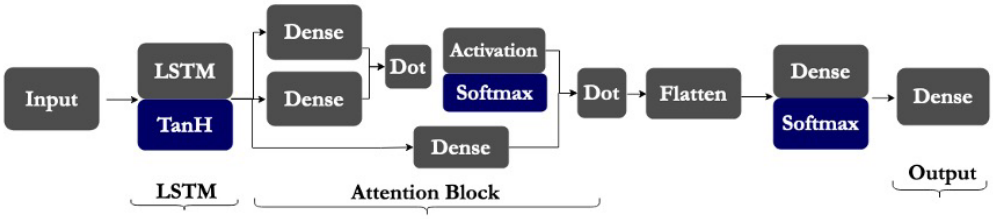

Figure 2: AT-LSTM Model.

Figure 2 shows the structure of the AT-LSTM model. The input to the model is a pre-processed sequence of network traffic records in the form of an LSTM layer. The LSTM layer is followed by a self-attention mechanism, where the query, key, and value are extracted using dense layers.. The attention weights are computed using the dot product of the query and key, followed by a softmax activation function. The attention vector is then computed as the weighted sum of the values using the attention weights. The output of the attention layer is then flattened and passed through a dense layer with a softmax activation function to make the final prediction.

3.5 AT-BiLSTM

Bidirectional LSTM (BiLSTM) uses two separate hidden LSTMs in each layer, one processing the input sequence forward and the other processing the input sequence backwards. The outputs from both LSTMs are concatenated and used as the final hidden representation of the input sequence. To implement BiLSTM, the following changes are implemented to the LSTM equations:

$$\boldsymbol{I'_t = \sigma(X_t W'_{xi} + h'_{t+1} W'_{hi} + b'_i)} \textbf{ (15)}$$

$$\boldsymbol{F'_t = \sigma\left(X_t W'_{xf} + h'_{t+1} W'_{hf} + b'_f\right)} \textbf{ (16)}$$

$$\boldsymbol{O'_t = \sigma(X_t W'_{xo} + h'_{t+1} W'_{ho} + b'_o)} \textbf{ (17)}$$

$$\boldsymbol{\widehat{C_t}' = tanh(X_t W'_{xc} + h_{t+1} W'_{hc} + b'_c)} \textbf{ (18)}$$

$$\boldsymbol{h'_t = tanh(C'_t) \odot O'_t} \textbf{ (19)}$$

$$\boldsymbol{\hat{y}_t = \sigma(h_t W_h + b_h) + \sigma(h'_t W'_h + b'_h)} \textbf{ (20)}$$

For the input gate, forget gate, and output gate equations, new weight matrices, and bias vectors are added for the backward LSTM, denoted by $W'_{xi}, W'_{xf}, W'_{xo}, W'_{hi}, W'_{hf}, W'_{ho}, b'_i, b'_f, b'_o$. For the hidden state update equation, new weight matrices and bias vectors are added for the backward LSTM, denoted by W'_{xc} and W'_{hc}, and b'_c. Then the forward and backward hidden states, h_t and h'_t, are concatenated at each time step to obtain a combined hidden state, denoted by $h_t^{comb} = [h_t, h'_t]$. The output layer equation for each time step would depend on the combined hidden state h_t^{comb}. The gradient updates for the backward LSTM would need to be accumulated separately from the forward LSTM, and the backward LSTM would need to be run in reverse order.

4. Experimental Results

4.1 Setup Environment

The experiment was carried out on Google Jupyter notebooks using the GPU for model training. The model was trained using varied batch sizes and epochs. To prevent overfitting and save computational resources, the model was also set up with an early stopping of a patience of 8 and a minimum delta of 0.001. Table 1 shows the details of the environment specification.

Table 1: Environment Specification

Unit	Description
Processor	Intel(R) Xeon(R) CPU @ 2.30GHz
GPU	Tesla P100-PCIE-16GB

Unit	Description
System RAM	13 GB
OS	Linux 5.10.147+

4.2 Results and Discussions

Accuracy, precision, recall, and F1 score are used for assessing the efficacy of the model. Figure 3 and Figure 4 show confusion matrices, which are used to measure how well classification models work. The results of the models are visualised by comparing the actual classes of attacks to the predicted attack classes made by the models. The diagonals of the matrix show where the actual class and the predicted class match up. The off-diagonal values represent the cases where the model has made incorrect predictions.

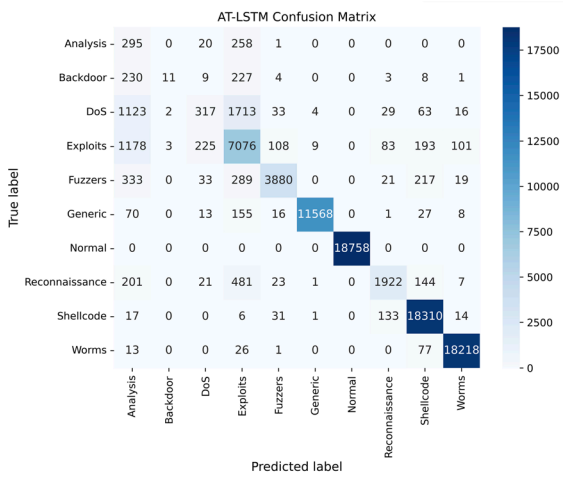

Figure 3: AT-LSTM confusion matrix

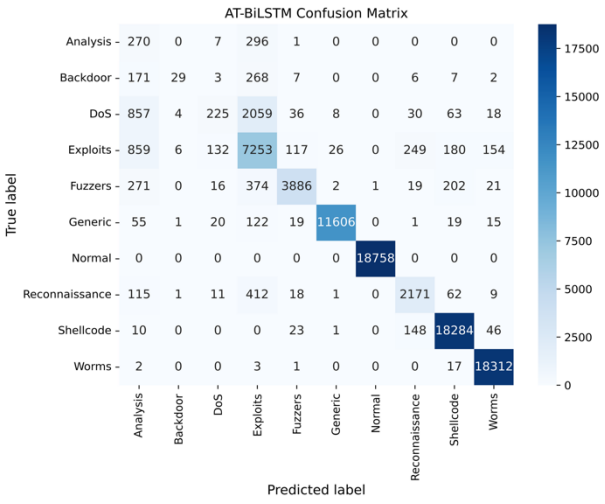

Figure 4: AT-BiLSTM confusion matrix

Table 2 displays the performance metrics of two intrusion detection models, AT-LSTM and AT-BiLSTM, on the dataset. The models were evaluated based on their accuracy, precision, recall, and F1-score. Both models have a high accuracy, with AT-LSTM having an accuracy of 89.90% and AT-BiLSTM having an accuracy of 92.20%. The precision of both models is also high, with AT-LSTM having a precision of 93.06% and AT-BiLSTM having a precision of 93.83%. This suggests that both models can correctly identify most positive instances. The recall of the AT-LSTM model is 87.07%, while the recall of the AT-BiLSTM model is 91.19%. The F1-score values for the AT-LSTM and AT-BiLSTM models are 89.96% and 92.50%, respectively. A higher F1-score value indicates a better balance between precision and recall, which implies better overall performance of the model. In this case, the AT-BiLSTM model has a slightly higher F1-score value, indicating that it may be a better model than the AT-LSTM in terms of overall performance. BiLSTM model is advantageous due to its ability to process the input sequence

in both forward and backward directions. This allows the model to capture more contextual information and dependencies within the sequence, which is particularly useful for detecting network intrusions. Attacks may involve complex and subtle patterns of activity that occur over time, that may not be evident from a single direction.

Table 2: Performance Metrics of AT-LSTM and AT-BiLSTM

Model	Performance Metrics			
	Accuracy	Precision	Recall	F1-Score
AT-LSTM	89.90%	93.06%	87.07%	89.96%
AT-BiLSTM	92.20%	93.83%	91.19 %	92.50 %

The AT-LSTM and AT-BiLSTM deep learning models were evaluated on various attack categories to assess their effectiveness in detecting network intrusions. The results in Table 3 demonstrate that both models have high detection rates for most of the attack categories. Notably, the models achieved a 100% detection rate for generic and normal network flows, which are the most common types of network traffic. This indicates that the models are highly effective in identifying normal network activity. For attacks that involve large-scale network scans and automated propagation, such as worms and fuzzers, the models achieved high detection rates. The AT-LSTM model had a detection rate of 89% for fuzzers, while the AT-BiLSTM model achieved a higher rate of 92%. Similarly, the AT-LSTM model detected 97% of worm attacks, and the AT-BiLSTM model achieved a detection rate of 98%. These high detection rates demonstrate the models' effectiveness in detecting attacks propagating quickly and widely across a network. The models also performed reasonably well in detecting reconnaissance, backdoor, exploits, and shellcode attacks. For example, the AT-LSTM model achieved a detection rate of 75% for reconnaissance, while the AT-BiLSTM model achieved a higher rate of 85%. The AT-LSTM model detected 79% of backdoor attacks, while the AT-BiLSTM model achieved a higher rate of 100%. In addition, the AT-LSTM model achieved a detection rate of 78% for exploits, while the AT-BiLSTM model detected 73%. Finally, the AT-LSTM model detected 93% of shellcode attacks, while the AT-BiLSTM model achieved a slightly higher rate of 96%. These attack categories can be particularly challenging to detect due to their sophisticated techniques that can evade traditional security measures. However, the high detection rates of the models for these attacks demonstrate their effectiveness in identifying such threats. For instance, the backdoor attack is a type of malware that allows attackers to access a system remotely and control it without the user's knowledge. The high detection rate of 100% achieved by the AT-BiLSTM model for this attack type indicates its ability to detect such subtle and complex activity patterns.

Table 3: Detection Rate of Attacks Categories

Attack Type	Detection Rate	
	AT-LSTM	AT-BiLSTM
Analysis	7%	8%
Backdoor	79%	63%
DoS	43%	51%
Exploits	78%	73%
Fuzzers	89%	92%
Generic	100%	100%
Normal	100%	100%
Reconnaissance	75%	85%
Shellcode	93%	96%
Worms	97%	98%

The accuracy vs. epochs plot is a keyway to measure how well the models are doing while they are being trained. It is used to figure out if the model is too good or too bad and to find the best number of epochs. Figures 4 and 5 show that the two models' training accuracy steadily improves with each epoch. This means that the models are learning from the data and getting better over time. The validation accuracy, which assesses the model's ability to perform well on new, unseen data, exhibits some fluctuations, particularly in the AT-BiLSTM model,

despite an overall upward trend. This may be due to overfitting, where the complex AT-BiLSTM model is too closely fitted to the training data, resulting in variability in its performance on new data. As a result, the training accuracy may continue to rise with each epoch, while the testing accuracy may fluctuate.

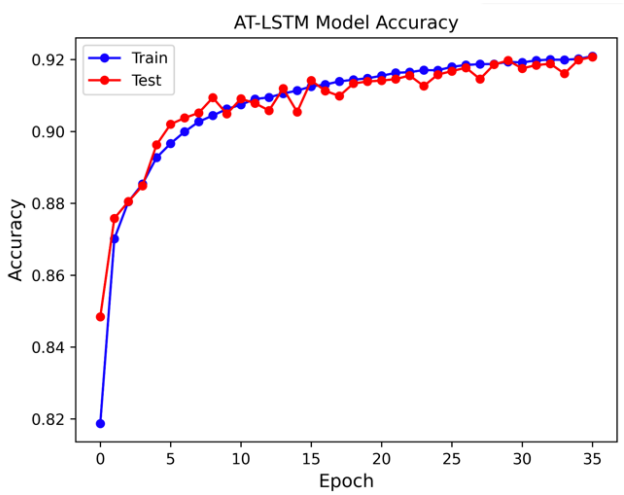

Figure 4: AT-LSTM Accuracy vs. Epochs.

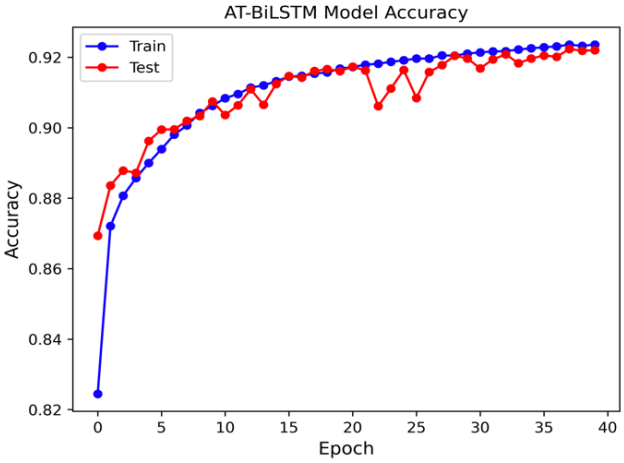

Figure 5: AT-BiLSTM Accuracy vs. Epochs.

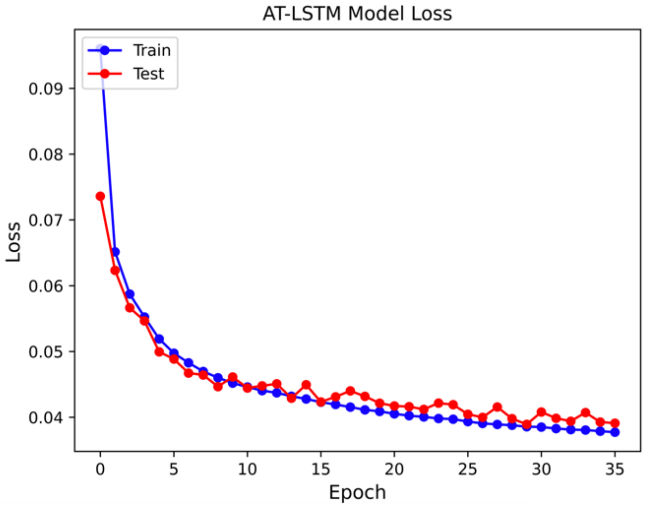

Figure 6: AT-LSTM Loss vs Epochs.

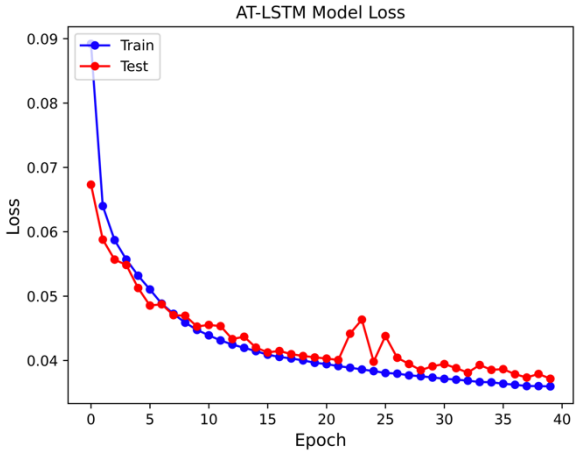

Figure 7: AT BiLSTM Loss vs Epochs.

The loss vs. epochs plot shows how the training and validation loss values change as the model trains over a certain number of epochs. The loss is a measure of how well the models can predict the target output. In Figures 6 and 7, we can see that as the number of epochs goes up, both the training loss and the validation loss tend to go down. Figure 8 shows that the AT-LSTM model with smaller batch sizes tends to converge faster and achieve higher accuracy than those with larger batch sizes. This is because smaller batch sizes allow for more frequent parameter updates, leading to faster convergence and better generalisation. On the other hand, larger batch sizes may converge more slowly and suffer from overfitting due to seeing a biased representation of the overall dataset. Thus, the figure shows a clear trend of increasing accuracy with smaller batch sizes over the course of training.

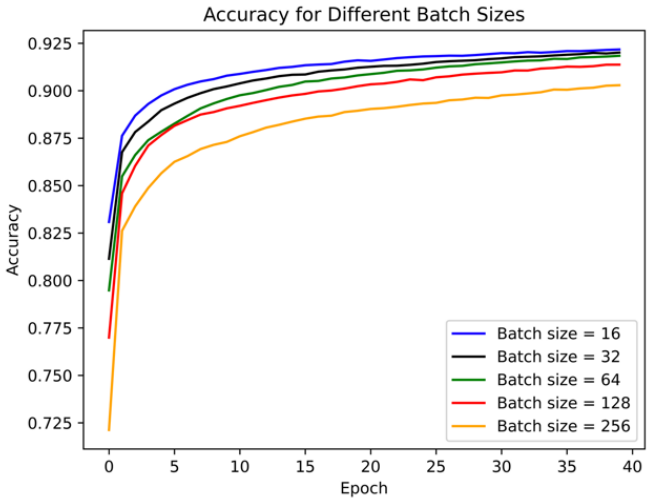

Figure 8: Accuracy vs Epochs for different batch sizes.

5. Conclusion

The study demonstrated the effectiveness of attention-based LSTM and BiLSTM models in detecting network intrusions with precision rates exceeding 93%. The models could capture complex and subtle patterns of activity associated with different types of attacks. However, to ensure their practical applicability, the models' limitations must be addressed in future research. This includes developing models that are more sensitive to the characteristics of DoS attacks, such as high traffic volume and unusual traffic patterns, and incorporating data from multiple sources to provide a more comprehensive view of network activity. Additionally, incorporating information about the source of the traffic, such as IP addresses, could help to identify better and block malicious traffic sources. Finally, developing methods to identify and mitigate adversarial attacks, such as adding noise to the input data or using robust training techniques, could further enhance the security and effectiveness of the intrusion detection system.

Acknowledgement

This work was supported by the United Arab Emirates (UAE) University Program for Advanced Research (UPAR), under Grant number 31T122.

References

Aldallal, A. (2022) 'Toward Efficient Intrusion Detection System Using Hybrid Deep Learning Approach', *Symmetry*, 14(9), p. 1916.

Bakhsh, S.T. *et al.* (2019) 'An adaptive intrusion detection and prevention system for Internet of Things', *International Journal of Distributed Sensor Networks*, 15(11), p. 155014771988810.

Benaddi, H. *et al.* (2022) 'Robust Enhancement of Intrusion Detection Systems Using Deep Reinforcement Learning and Stochastic Game', *IEEE Transactions on Vehicular Technology*, 71(10), pp. 11089–11102.

Cao, K. *et al.* (2021) 'Network Intrusion Detection based on Dense Dilated Convolutions and Attention Mechanism', in *2021 International Wireless Communications and Mobile Computing (IWCMC)*. *2021 International Wireless Communications and Mobile Computing (IWCMC)*, Harbin City, China: IEEE, pp. 463–468.

Fan, S. *et al.* (2019) 'ALEAP: Attention-based LSTM with Event Embedding for Attack Projection', in *2019 IEEE 38th International Performance Computing and Communications Conference (IPCCC)*. *2019 IEEE 38th International Performance Computing and Communications Conference (IPCCC)*, London, United Kingdom: IEEE, pp. 1–8.

Freire, P.J. *et al.* (2022) 'Computational Complexity Evaluation of Neural Network Applications in Signal Processing'. arXiv.

Gaifulina, D. and Kotenko, I. (2021) 'Selection of Deep Neural Network Models for IoT Anomaly Detection Experiments', in *2021 29th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)*., Valladolid, Spain: IEEE, pp. 260–265.

Han, X. *et al.* (2020) 'STIDM: A Spatial and Temporal Aware Intrusion Detection Model', in *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, Guangzhou, China: IEEE, pp. 370–377.

Laghrissi, F. *et al.* (2021) 'IDS-attention: an efficient algorithm for intrusion detection systems using attention mechanism', *Journal of Big Data*, 8(1), p. 149.

Lan, M. *et al.* (2020) 'A Novel Industrial Intrusion Detection Method based on Threshold-optimized CNN-BiLSTM-Attention using ROC Curve', in *2020 39th Chinese Control Conference (CCC)*. *2020 39th Chinese Control Conference (CCC)*, Shenyang, China: IEEE, pp. 7384–7389.

Liu, C., Gu, Z. and Wang, J. (2021) 'A Hybrid Intrusion Detection System Based on Scalable K-Means+ Random Forest and Deep Learning', *IEEE Access*, 9, pp. 75729–75740.

Liu, D. *et al.* (2020) 'Air pollution forecasting based on attention-based LSTM neural network and ensemble learning', *Expert Systems*, 37(3).

Liu, Q. *et al.* (2022) 'Improving wireless indoor non-intrusive load disaggregation using attention-based deep learning networks', *Physical Communication*, 51, p. 101584.

Mighan, S.N. and Kahani, M. (2021) 'A novel scalable intrusion detection system based on deep learning', *International Journal of Information Security*, 20(3), pp. 387–403.

Mirza, A.H. and Cosan, S. (2018) 'Computer network intrusion detection using sequential LSTM Neural Networks autoencoders', in *2018 26th Signal Processing and Communications Applications Conference (SIU)*. *2018 26th Signal Processing and Communications Applications Conference (SIU)*, Izmir: IEEE, pp. 1–4.

Moustafa, N. and Slay, J. (2016) 'The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set', *Information Security Journal: A Global Perspective*, 25(1–3), pp. 18–31.

Sarhan, M. *et al.* (2021) 'NetFlow Datasets for Machine Learning-Based Network Intrusion Detection Systems', in Z. Deze et al. (eds) *Big Data Technologies and Applications*. Cham: Springer International Publishing (Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering), pp. 117–135.

Saurabh, K. *et al.* (2022) 'LBDMIDS: LSTM Based Deep Learning Model for Intrusion Detection Systems for IoT Networks'. arXiv.

Waqar, M. *et al.* (2021) 'An Efficient SMOTE-Based Deep Learning Model for Heart Attack Prediction', *Scientific Programming*. Edited by M.-C. Chen, 2021, pp. 1–12.

Wei, P. *et al.* (2023) 'A novel intrusion detection model for the CAN bus packet of in-vehicle network based on attention mechanism and autoencoder', *Digital Communications and Networks*, 9(1), pp. 14–21.

Wu, Z. *et al.* (2022) 'RTIDS: A Robust Transformer-Based Approach for Intrusion Detection System', *IEEE Access*, 10, pp. 64375–64387.

Xu, C. *et al.* (2018) 'An Intrusion Detection System Using a Deep Neural Network With Gated Recurrent Units', *IEEE Access*, 6, pp. 48697–48707.

Yan, Y. *et al.* (2020) 'A Network Intrusion Detection Method Based on Stacked Autoencoder and LSTM', in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*. *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, Dublin, Ireland: IEEE, pp. 1–6.

Yang, H. *et al.* (2022) 'Intrusion Detection Model For Power Information Network Based On Multi-layer Attention Mechanism', in *2022 IEEE 10th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*. *2022 IEEE 10th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*, Chongqing, China: IEEE, pp. 825–828.

Yu, Y. *et al.* (2022) 'LSTM-Based Intrusion Detection System for VANETs: A Time Series Classification Approach to False Message Detection', *IEEE Transactions on Intelligent Transportation Systems*, 23(12), pp. 23906–23918.

Zhang, C. *et al.* (2022) 'Comparative research on network intrusion detection methods based on machine learning', *Computers & Security*, 121, p. 102861.

Zhang, Y. *et al.* (2018) 'Deep Learning Intrusion Detection Model Based on Optimized Imbalanced Network Data', in *2018 IEEE 18th International Conference on Communication Technology (ICCT)*. *2018 IEEE 18th International Conference on Communication Technology (ICCT)*, Chongqing: IEEE, pp. 1128–1132.

Zhang, Y. *et al.* (2019) 'A Text Sentiment Classification Modeling Method Based on Coordinated CNN-LSTM-Attention Model', *Chinese Journal of Electronics*, 28(1), pp. 120–126.