Zero Trust Network Intrusion Detection System (NIDS) using Auto Encoder for Attention-based CNN-BiLSTM

Abeer Z. Alalmaie
abeer.z.alalmaie@student.uts.edu.au
School of Electrical and Data Engineering, University of Technology Sydney
Sydney, Australia

Priyadarsi Nanda
priyadarsi.nanda@uts.edu.au
School of Electrical and Data Engineering, University of Technology Sydney
Sydney, Australia

Xiangjian He
sean.he@nottingham.edu.cn
Computer Science, University of Nottingham
Ningbo, China

ABSTRACT

The large number of connected networks that underpin today's IT ecosystem make them more vulnerable to cyber threats because of their connectivity, user diversity, amount of connected devices, and services and applications that are available worldwide. As a response to these cyberthreats, zero trust security has been recommended. However, it's crucial to remember that this kind of security monitoring can be done by outside experts. When cloud-based third parties access network traces, there are threats to data security, thus the present trend in security monitoring needs to change to a "Never Trust, Always Verify" approach.

Network Intrusion Detection System (NIDS) can be used to detect anomalous behavior. Convolution Neural Network (CNN) and Bi-directional Long Short Term Memory (BiLSTM) based classifiers and Auto-Encoder (AE) feature extractors have shown promising results in NIDS. AE feature extractor provides possibility of compressing the most important information and training the model unsupervised. CNNs are capable to capture local spatial relationships, while BiLSTMs are good at exploiting temporal interactions. In addition, Attention modules are good at capturing content-based global interactions, and can be applied on CNNs to attend to the most important contextual information. In this work, we utilized the advantages of all AE, CNN and BiLSTM structures using a multi-head Self Attention mechanism to focus and integrate CNN features for feeding into BiLSTM classifier.

We proposed to use the bottleneck features of a pre-trained AE for an Attention-based CNN-BiLSTM classifier. Our experiments using 6 and 10 category NID system on UNSW-NB15 dataset showed that our proposed method outperforms state-of-the-art methods and achieved accuracy of 89.79% and 88.13% respectively. Also, we proposed a balanced data sampler for training 10 categories of NIDS which improved the accuracy up to 91.72%. We demonstrated the importance of Attention mechanism through our proposed method.

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

KEYWORDS

Zero Trust, Network Intrusion Detection, Network Security, CNN-BiLSTM, Attention

1 INTRODUCTION

As network security monitoring becomes increasingly complex,it becomes necessary to outsource these jobs to external analysts. Additionally, there is an increasing demand for an accurate, categorised Network Intrusion Detection (NID) system. Many firms are increasingly turning to third party analysts to complete these activities as a result of the increased amount of security monitoring tasks. Due to potential security issues, they are typically hesitant to divulge network traces. For instance, if a sensitive piece of information appears in the traces, it could be used to launch an cyber attack on the systems or networks [3].

Modernizing data centres, moving to the cloud, and reforming SOC are all ongoing transformations for businesses. Due to their characteristics, businesses with implicit trusted networks that continuously validate all digital transactions are exposed to unwanted access and data breaches. Since the cloud is the new network edge, it cannot be designed using the outdated implicit trust model because it introduces security flaws. In the digital world, you should trust nothing and always verify everything. This approach is inapplicable in a world where network edges have fragmented. The number of breaches has recently been rising along with the number of security-related incidents. This problem may result in unauthorised access to private data.

The number of security incidents has skyrocketed, and with it the demand for efficient security monitoring, which is rising steadily. It's crucial that the existing strategy is supplemented by an all-encompassing solution that can track and identify anamolous activity in real time rather than remaining restricted to a Trust but Verify strategy. The main goal of this study is to offer data owners a customizable, cost-effective solution. By drawing a line between the trusted and untrusted parts of networks, this strategy hopes to accomplish its objective. The data centre architecture makes this possible.

Anonymization techniques must be applied over the network traces of data owners in order to accomplish this. However, not only is this approach necessary to increase the networks' security, but it is also a difficult undertaking. This paper aims to give a comprehensive review of the many techniques applied to increase network security.

The idea of zero trust security forbids total trust in a network's supporting structures. It is impossible to trust any entity, even those

that are a part of the network. To change the mentality from "Never Trust, Always Verify" to "Always Trust" is the idea behind zero trust. Ensure that every privilege is explicitly authorised. Do not treat any user, programme, or data flow as untrusted. This strategy makes sure that a system only accesses the most vital resources and that only malicious or out-of-the-ordinary activity is picked up. Zero Trust holds that a cyberattack is unavoidable and limits the access to only what is required to stop unauthorised access.

A security model with zero trust characteristics includes a set of guidelines intended to create a safe environment for all users and devices. In order to make sure the system is safe, it eliminates the implicit trust that formerly existed in every component of a network and necessitates constant monitoring and evaluation. This approach is based on the acknowledgement that threats exist both outside and inside the traditional network boundaries [3].

Organizations can defend themselves against threats that are always evolving with the use of a secure perimeter. This problem can be solved in a number of ways, including by using perturbation, poisoning, and encryption. Some of these solutions require for the establishment of a utility that ensures the veracity of the data that is outsourced, while others call for the protection of privacy.

One of the most widely used methods to prevent unauthorised access and data tampering is the implementation of CryptoPAn, a technique that entails maintaining network trace anonymity while substituting real IP addresses with a prefix derived from a pseudonym. This method preserves the hierarchical relationships between all IP addresses.

Zero Trust Security has put in place a number of strategies that are intended to impose restrictions on access and keep an eye on all devices and activities in order to prevent unwanted access and data modification. An anonymized trace is one of these and is made up of any two IP addresses with the same prefix. It has been shown that putting into practise a zero trust paradigm can prevent attacks while preserving a trade-off for original data preservation.

There are numerous ways available to deal with problems that necessitate rigorous data sanitation. By prefix-naming them and then concealing them in a number of false views, the multi-view strategy enables the adversaries to hide their genuine IP addresses. By using this technique, the attackers are unable to discern the difference between legitimate IP addresses and fake ones.

Even if the attackers use semantic or prior knowledge attacks, they can still hide the real IPs from the fake ones by deanonymizing the entire output of the network. This method can be performed by a powerful adversary who can perform injection or fingerprinting.

A zero trust security model can be used to pass the three concepts of access control, privilege, and log traffic. It can also be used to automate the process of monitoring and controlling the network traces. This method can be utilized for the development of Network Intrusion Detection Systems (NIDS). A deep neural network (DNN) is used as a network ID to improve the efficiency of the processes related to automatic authorization. This was the first time that this type of network has been used in Zero-Trust Security [3].

Network Intrusion Detection task is about discovering any attack and detecting the type of attack. Network Intrusion Detection has become an important research field because it can help protect the networks and devices against attacks [16]. Three of the most important applications of NIDS are protecting and securing web based applications (including clients, servers, Internet of Things), securing smart grids, and outsourcing network traces to the analyzers securely [3, 21, 27, 37, 42]. Researches in this field are generally divided into binary and categorical (multi-class) Intrusion Detection Systems. Although, categorical intrusion detection provides more useful information, it requires more processing than binary methods and cannot be used in low-resourced hardware [12, 16].

The remainder of this paper is organized as follows. Section 2 provides a brief review about the related works in NID. The proposed method for NID including both feature extraction and classification is described in Section 3. The used dataset is described and the experimental results and performance analysis are reported in Section 4. We conclude the paper in Section 5.

2 RELATED WORKS

In this section, we explore related works in NID and related fields using machine learning.

Rule based and statistical anomaly detection algorithms are used in preliminary research for Network Intrusion Detection. These methods can be considered as binary Intrusion Detection models since these are based on anomaly detection [8, 31]. With the application of machine learning (ML) in NID, ML based detection methods have become a hotspot in the related researches. For example, a fuzzy IDS for anomaly detection has been proposed which can be considered as a rule-base method [36].

Some classical ML methods have been proposed for NID including K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Decision Tree (DT), Naive Bayes (NB), and Multilayer Perceptron (MLP) [14, 16]. in addition, ensemble classifiers have been proposed, which utilized the benefits of multiple classifier at the same time [23, 39]. A two-stage ensemble classifier including hierarchical rotation forest and bagging classifiers, along with a hybrid evolutionary algorithm for feature selection have been proposed for NID which outperformed the former methods [39]. A four-way ensemble classifier including SVM, LR, NB, and DT has also been proposed which utilized a combination of feature selection methods [23].

Although, most researches have focused on new and optimum classifiers, there exist some researches on feature optimization for NID [10]. On the other hand, some researches have tried to reduce the data imbalance by sampling techniques to improve the learning efficiency [28].

Since classical machine learning methods require handcrafted feature extraction and accurate parameter tuning, deep learning techniques have become a trend in most artificial intelligence problems including image and speech processing and NID [16, 33]

A two-stage deep neural network has been proposed for NIDS including a deep Sparse Auto-Encoder (AE) as the feature extractor and a shallow Neural Network (NN) classifier. Obviously, the AE module is a Self-Taught Learning model since data labels are not used during training [15]. In [9], another Sparse Auto Encoder has been proposed for feature extraction, however, Support Vector Regression is used as the classifier instead of the shallow NN. The AE bottleneck features have been shown to be effective in enhancing the NID systems and giving the ability to feed any type of attributes to the NID model [1, 3]. In addition, Bottleneck features have been shown to be robust against noise [34].

A Recurrent Neural Network (RNN) has been proposed for NID to consider the changes of the input in real-time applications [6]. Also, deeper RNN models have been used for NID which outperformed previous works [5, 41]. Since Long Short Term Memory (LSTM) cells hold the long term dependencies and prevent the vanishing gradient problem, some others extended the RNN models to LSTM and bi-directional LSTM (BiLSTM) for NID [13, 22, 26, 35].

A Convolutional Neural Network (CNN) classifier using a two-stage feature extraction including a PCA and a feature engineering method to select the most relevant have been proposed for NID [2]. In [43], the CNN models have been used in combination of other classifier methods including RNN, LSTM, and GRU to show the capability of CNN in comparison with other architectures. Also, there are some other researches about the effectiveness of CNN and RNN based combination for NID [7].

Variants of Attention mechanism have been popular in many fields of AI to model temporal and spatial relationships [33]. Also, the Attention model is applied on LSTM to handle the temporal relation in NID using a combination of statistical features or reduction and Mutual information based feature extraction methods [25, 40].

In this paper, we proposed a novel Network Intrusion Detection algorithm, constructed from a hybrid deep hierarchical feature extractor and an Attention based BiLSTM classifier. The feature extractor is constructed from an Auto-Encoder and Convolutional Neural Network sequentially. The Auto-Encoder module is a Self-Taught Learning model. Our experiments on UNSW-NB15 [30] dataset show that our proposed method outperforms other baseline methods, both for binary and categorical classification.

In this paper, we proposed a novel Network Intrusion Detection algorithm, constructed from a hybrid deep hierarchical feature extractor and an Attention based BiLSTM classifier. The feature extractor is constructed from an Auto-Encoder and Convolutional Neural Network sequentially. The Auto-Encoder module is a Self-Taught Learning model. Our experiments on UNSW-NB15 [30] dataset show that our proposed method outperforms other baseline methods, both for binary and categorical classification.

3 PROPOSED INTRUSION DETECTION METHOD

Our proposed scheme is presented in Figure 1, which consists of four main modules including Auto-Encoder feature extractor, Convolution blocks, Attention mechanism and LSTM layers. The proposed approach is inspired from Auto-Encoder Convolutional Neural Network for binary Network Intrusion Detection. The effectiveness of AE-CNN for binary Intrusion Detection has been shown previously [3]. Thus, we use the extracted features as the input to our proposed neural network for categorical intrusion detection. We also utilize Attention module to focus on more important features, as well as LSTM layers to handle temporal dynamics [26, 44]. In the following, we first review Auto-Encoder feature extraction method, thereafter, the new three parts are described in the detail.

3.1 Auto-Encoder Feature Extraction

The compressed bottleneck features of Network traces are extracted using a pre-trained deep AE. The bottleneck layer in the AE maps the original input into a compressed representation where the input features are much more correlated. So, the bottleneck features are expected to work better with CNNs in comparison with raw data, because CNNs work better with the data having spatial relationship (e.g. images). An AE consists of two components: the encoder which compress input features, and the decoder which is discarded after pre-training. Consequently, a deep auto-encoder can be used to extract a combined and compressed feature from network trace attributes [3].

In the AE, the bottleneck features z are extracted using the encoder function ψ from the original data X. The decoder function ϕ maps the bottleneck z to the output $\hat{X}$. The decoder is expected to reconstruct the input as shown in equation 1 [3].

$$\begin{aligned} \psi = X \Rightarrow z, \phi = z \Rightarrow \hat{X} \\ \psi, \phi = argmin||\hat{X} - (\phi(\psi(X)))||^{2'} \end{aligned} \tag{1}$$

In other words, mean squared error(mse) loss function of the AE is as equation2

$$L(X, X') = \|X - \hat{X}\|^2 = \|X - \sigma(W_0(\sigma(WX + b)) + b_0)\|^2 \tag{2}$$

where $X - \hat{X}$ is usually averaged over a mini-batch input training set. W, W_0 are weight matrices and b, b_0 are bias vectors for encoder and decoder, respectively. Bias is not used for encoder part to aggregate input feature only [3].

The structure of the AE is shown in Figure 2, where dotted lines are discarded after training the AE. The bottleneck features of the trained AE, which are more spatially related than raw features, are used as input to CNN-LSTM. We use the pre-trained AE from [3], as the input features for both methods are the same.

Since AE with a bottleneck layer accepts any numerical value and compresses the information available in the input numerical values, pre-processing and feature selection is not needed.

3.2 Convolutional Neural Network

We propose to use a CNN to consider spatially related features extracted using the AE. A CNN classifier has been applied on a bottleneck features extracted from a trained AE for Network Intrusion Detection because CNNs work well with data that has a spatial relationship [3]. The CNNs are also known to be good feature extractors because of local convolution filters, repetitive filters among whole input data, and pooling layers which make it robust [32]. In this work, we also propose to use a tuned 1D CNN to handle spatial dependencies within traces of data. The proposed CNN fuse the compressed information extracted by AE according to the classification task. Our proposed CNN structure is shown in Figure 3, and $LeakyReLU$ with 0.2 negative slope is considered as activation function for hidden layers. In convolution layers, the first number is the number of filters and the number in parentheses is the convolution filter size, e.g. first layer has 128 filters, where 11 is the convolution filter size. A pooling with size of 2 is only applied on the first convolution layer.

In the output of the CNN layer we have $256 * 5$ features, which its knowledge needs to be aggregated together, since it has a high dimension of feature vector to feed into non-convolutional layers.

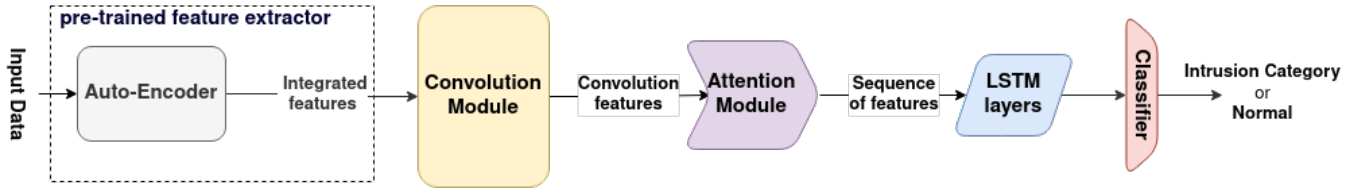

Figure 1: Overview of the proposed method for NID

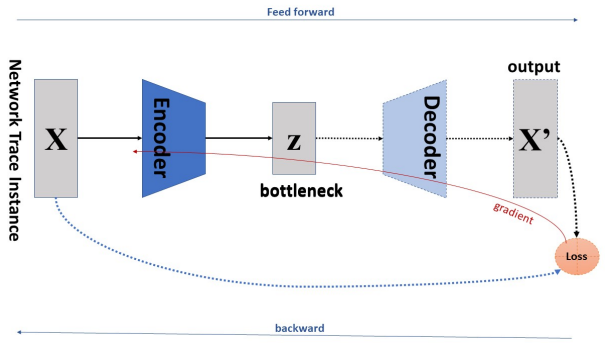

Figure 2: Network traces AE feature extractor [3]

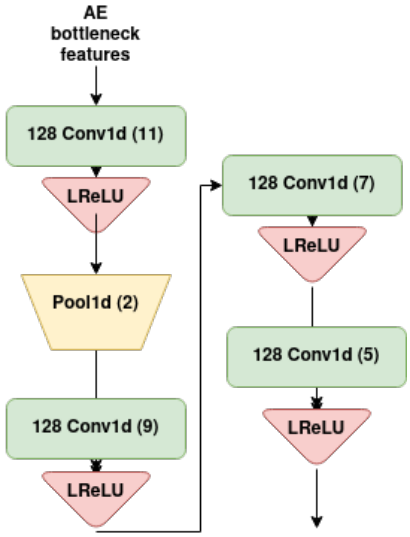

Figure 3: Proposed CNN to handle spatial dependencies of network traces

3.3 Attention Module

We apply a multi-head Self Attention module to aggregate the information available in extracted features and handle the relation between the CNN features and LSTM components (subsequent layers). The attention module dimension is the number of channels from the last CNN layer (256), and uses 8 heads. The output is mapped into 64 dimension to limit the features. The attention module learns to focus on intrusion related features and seems to be effective since the AE and CNN feature maps may have features not related to the intrusion type.

The alternative structure for multi-head self Attention on top of CNN, would be a linear Flatten layer, which maps the CNN multi-dimensional features into one large dimension. The total number of features (neurons) in this layer, is the same as the total number of CNN features in all dimensions. We also report the results of the proposed method with a linear flatten layer instead of Attention mechanism. Finally, we propose to use BiLSTM layers after the Attention to handle the temporal dynamics between the sequences of network traces.

3.4 BiLSTM Classifier

Since LSTMs can hold or forget information for a long time, we propose to use LSTMs to handle the temporal dynamics [13]. Also, BiLSTM is able to take forward and backward sequences into consideration which can be important in handling temporal dynamics [11]. We use 2 BiLSTM layers with 128-dimensional representations. A dropout with probability of 0.2 is applied between 2 layers of BiLSTM. Finally, a linear layer with the number of neurons same as target categories is applied. In the next sections, we will review the training and test conditions along with the evaluation results.

4 EXPERIMENTS AND RESULTS

4.1 Training setup

All experiments are implemented in PyTorch and conducted on Colab platform with a batch size of 32. The AE is trained to minimize mean squared error (mse) criterion as loss function, which is also known as reconstruction error. Both encoder and decoder parameters are considered and trained independently. The dimension of the bottleneck features is considered as 64, which is compact enough to compress input features. We used the Adam optimizer with learning rate of 1e-4 and weight decay of 1e-5 to minimize the reconstruction loss.

The model is trained until no more improvement is possible according to validation results. All data attributes are normalized to numerical values between 0 and 1. Thus, non-numerical attributes are converted into numerical values using one-hot encoding. The training dataset is shuffled to prevent over fitting.

The bottleneck features extracted from the trained AE are fed into the CNN for further training and processing while the AE is frozen. Since the compact features of input attributes are available in the bottleneck layer with 64 neurons, the CNN input spatial dimension is 64 and the sequence number equals the batch size.

Table 1: UNSW-NB15 dataset categories with corresponding number of items in train and test sets [16]

Category	Train	Test
Normal	56000	37005
Backdoor	1746	583
Analysis	2000	677
Fuzzers	18185	6062
Shellcode	1133	378
Reconnaissance	10492	3496
Exploits	33393	11132
DoS	12264	4089
Worms	130	44
Generic	40000	18871
Total	175343	82337

The CNN modules are trained with learning rate of 1e-3, while the Attention and BiLSTM modules are trained with learning rate of 1e-4 for a maximum 50 epochs. We used cross-entropy loss as the classifier loss function and Adam optimizer.

4.2 Data

We evaluate the proposed method on the UNSW-NB15 dataset [30], which is comprised of a hybrid of real modern normal activities and synthetic contemporary attack behaviors. It contains ten classes, namely: Normal, Fuzzers, Analysis, Back- doors, DoS, Exploits, Generic, Reconnaissance, Shell Code, and Worms. We use the train and test subsets of the UNSW-NB15 dataset with 175343 and 82337 records respectively [29, 30].

In UNSW-NB15 dataset each record has a 42-dimensional feature, which 3 features of them are non-numerical values and need pre-processing to be fed into Neural Networks since the input of NN should be a digital matrix. These 3 features are protocol, service, and state with 133, 13, and 11 symbol attributes, respectively [29, 30].

One-hot encoding is used to map non-numerical attributes of the data set to numerical feature vectors. In total, the pre-processed input feature size would be 196. Then, the features are normalized between 0 and 1 which are used to train the AE unsupervised. The 64-dimensional bottleneck features extracted from the trained AE are used for next experiments.

Since 9 classes of attacks are unbalanced as shown in Table 1, most studies reduce this number by merging some categories together or removing some of them. Binary classification means all 9 categories are merged into 1 class as Intrusion, consequently the classes would be Intrusion/Non-Intrusion in this scenario. However, some other works try to merge the categories that are not far from each other. Some other works, try to reduce the amount of imbalance by removing the categories with fewer number of existing items including Backdoor, Analysis, Shellcode, Worms and sometimes Fuzzers which cause imbalance in the data [3, 7, 14, 16, 24, 29, 30].

In our experiments, we compare the results of 10 categories classification with the corresponding articles. Also, we report the result of removing imbalanced data attributes to have a fair comparison with other state of the art methods. On the other hand, to show the advantage of the proposed structure according to the previous structures, we report the binary classification results using the same data structure as [3], which train and test data are used in reverse and cause to have a few training samples.

Table 2: Accuracy results of Ten categories NID on test data

Method	Accuracy
CNN (structure of [3])	78.23%
BiLSTM (structure of the proposed method)	77.46%
CNN-BiLSTM (structure of the proposed method)	78.76%
Data pre-process with scaling + SVM [17]	75.77%
Decision Tree C5 [24]	90.74%
integrated rule based [24]	84.83%
CNN-Attention + BiLSTM (proposed method)	87.76%
CNN-BiLSTM [16]	77.16%
CNN-Attention (structure of the proposed method)	88.13%
Feature Selection + ANN [20]	77.51%

In addition, we propose a nearly balanced sampling procedure to enhance the detection of the categories with fewer samples in the CNN module. Due to the sequential nature required to train the LSTM, we can not use any sampling strategy to train it.

The UNSW-NB15 dataset is highly imbalanced, the Normal category has 56000 samples for training while the Worms category only has 130 samples. We reduce the impact of this imbalance by sampling based on a smoothing probability function as equation 3.

$$P(cl_i) = \frac{\#cl_i - (1 - \frac{\min \#cl}{\#cl_i} + \epsilon)median}{\sum_{j=1}^{10} \#cl_j - (1 - \frac{\min \#cl}{\#cl_j} + \epsilon)median} \tag{3}$$

In this equation, cl_i means i'th category (class), so $P(cl_i)$ is the probability of choosing a sample from i'th category, calculated using number of samples in each category ($\#cl_i$) and median of the number of samples per category. We use a small ϵ (0.1) to prevent zero addition for the category with minimum number of samples. According to Table 1, the minimum number of samples is 130 associated with category Worms, and the median is 11378.

The proposed sampling strategy keeps the ordering of the number of the categories but make the sampling more balanced by reducing the distance between number of items per each category. In the following, we report the results of NID methods for ten, six, and two categories.

4.3 Ten Categories CNN-BiLSTM Data Classification

For hyper parameter optimization, we explored the optimal number of layers and neurons for each part of CNN-BiLSTM with Attention module. The optimal hyper parameters were described in Section 2. To evaluate the effect of each module, the results of BiLSTM, CNN-BiLSTM (with linear layer in between), and Attention based CNN-BiLSTM are compared to previous state of the art methods in Table 2 for ten categories data classification.

Since the results of the baseline CNN model using AE features are not available for categorical classification, we implemented it and reported the results for comparison.

Table 3: Accuracy results of Six categories NID on test data

Method	Accuracy
CNN (structure of [3])	82.01%
BiLSTM (structure of the proposed method)	83.11%
CNN-BiLSTM (structure of the proposed method)	86.28%
CNN-Attention (structure of the proposed method)	87.54%
CNN-Attention + BiLSTM (proposed method)	89.79%
MLP + IGRF-RFE [45]	84.24%
CNN-GRU + RFP [7]	86.25%
Rule Based [24]	84.84%

Table 4: Accuracy results of Ten categories NID on test data for Balanced and Imbalanced sampler

Method	Accuracy
Proposed method (Standard Sampler)	87.76%
Proposed method (Balanced Sampler)	91.72%

Obviously, Attention based CNN-BiLSTM using AE bottleneck features outperformed other related works using deep learning approaches for 10 categories NID. The confusion matrix of the proposed method is shown in Figure 4.

As can be seen from Figure 4, most errors of Analysis, Backdoor and Exploits attacks are misclassified as DoS. In addition, Fuzzers and Exploits are misclassified interchangeably. None of Analysis and Backdoor records are predicted correctly. Only 3 records of Worms class are predicted correctly. Consequently, removing imbalanced data attributes including Backdoor, Analysis, Shellcode and Worms should obviously improve the accuracy. Thus, removing Fuzzers may lead to better accuracy as is done in [24]. In the following, we report the results of our method with the remaining six categories to compare with related works.

4.4 Six Categories CNN-BiLSTM Data Classification

Since recent works removed imbalanced data for their experimental results and most of them reported with six categories, we also experiment the proposed method with removing four imbalanced categories. The results are reported in Table 3.

As can be seen, the proposed method also outperforms state-of-the-art methods for six categories classification. The confusion matrix is available in Figure5.

4.5 Pre-train CNN using balanced data sampler

We propose to use a balanced data sampler to pre-train the CNN for later usage in CNN-BiLSTM with Attention module instead of reducing the number of categories. Our goal is to improve the discrimination of the model to learn discriminate the data better even when the number of training samples are imbalanced. Consequently, we use the trained CNN, which we believe is a better discriminator, in CNN-BiLSTM with Attention module to enhance the detection of network intrusion. The accuracy results of CNN-BiLSTM with Attention module with and without balanced sampler are compared in Table 4.

Table 5: Results of NID on test data in terms of Precision, Recall and F1-Score

Method	Precision	Recall	F1-Score
Feature Selection + ANN [20]	79.50%	77.53%	77.28%
Decision Tree C5 [24]	-	75.8%	75.54%
integrated rule based [24]	-	65.21%	68.13%
Proposed method (Balanced)	60.24%	78.5%	62.62%

Table 6: Accuracy results of NID (binary classification) on test data

Method	Accuracy
CNN [3]	92.23%
BiLSTM (with structure of the proposed method)	90.84%
CNN-BiLSTM (with structure of the proposed method)	78.93%
CNN-Attention + BiLSTM (proposed method)	93.01%
Feature selection + Deep Neural Network [19]	89%

As can be seen, pre-training CNN using balanced sampler outperforms standard training and also other works in terms of accuracy. In order to show the performance of the proposed method, we also compare other metrics including recall, precision and f1-score for the proposed method with related works in Table 5.

The confusion matrix of the proposed method with balanced sampler for ten categories is shown in Figure 6. Obviously, the number of misclassifications for each category is low in this method, specially for Normal category.

4.6 Binary CNN-BiLSTM Data Classification

The hyper parameters for binary classification model have been kept same as the multi-class model.

To evaluate the effect of each module, the results of BiLSTM, CNN-BiLSTM (with linear layer in between), and Attention based CNN-BiLSTM are compared to the most related work using CNN with AE bottleneck features for NID in Table 6 [3].

For a fair comparison, our data should be similar. We used train and test data interchangeably to have a fair comparison with CNN and AE method as it is used by [3].

As can be seen from three first rows of Table 6, using BiLSTM decreases the accuracy of the model especially in combination of CNN. It can be due to the high dimension of CNN output, which is fed into the BiLSTM layers. However, using an Attention module on CNN to aggregate the CNN features for feeding into BiLSTM layers outperformed CNN and BiLSTM models.

Since other binary classification methods using original train and test dataset for NID have reached almost 100% accuracy, more experiments and improvement are not needed [4, 18, 38].

According to the results, CNN and BiLSTM both perform well for NID using AE bottleneck features. However, an Attention module is needed to handle the relation between the components of these two structures and compose them together.

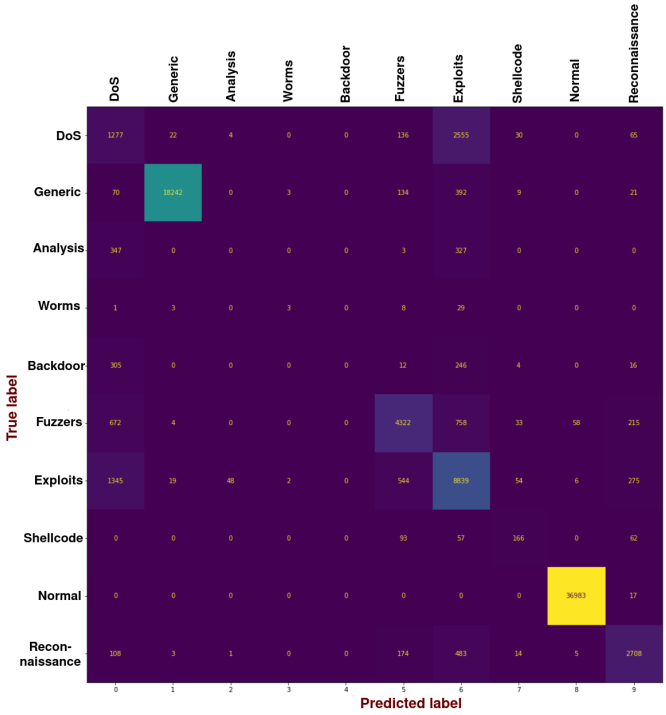

Figure 4: Confusion matrix of ten categories for CNN-BiLSTM with AE feature extractor

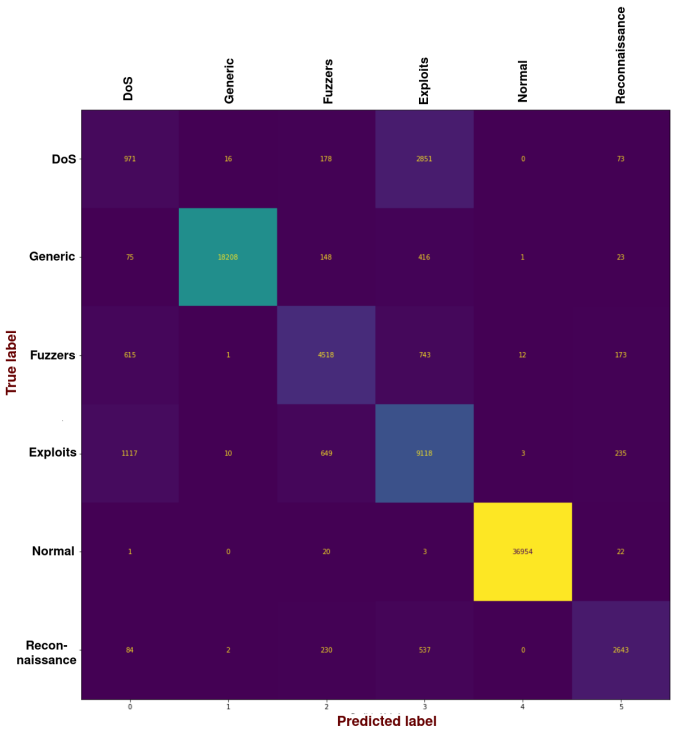

Figure 5: Confusion matrix of six categories for CNN-BiLSTM with AE feature extractor

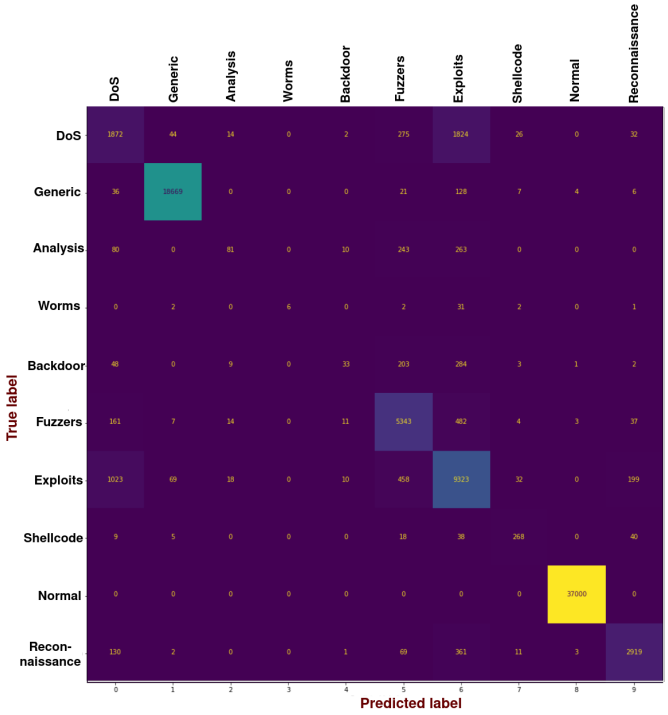

Figure 6: Confusion matrix of pre-trained CNN using balanced sampler + Attention + BiLSTM with AE feature extractor

5 CONCLUSION

In this paper, we have proposed an Attention Convolutional Neural Network with bi-directional Long Short Term Memory (CNN-BiLSTM) using Auto-Encoder Bottleneck features for Network Intrusion Detection system. We utilized the compressed bottleneck features of the Auto-Encoder. We also used a CNN to consider the spatial relation between extracted features. A multi-head Self Attention module is applied on CNN to aggregate the features and attend to the most important parts of the CNN feature maps for BiLSTM in the next layer. Finally, two BiLSTM layers are used for classification. To reduce the problem of data imbalance, we also propose to use a balanced sampler for pre-training the CNN. Our experimental results showed that our proposed approach outperforms state-of-the-art methods for 6 and 10 categories with classification accuracy of 89.79% and 91.72% on test set of UNSW-NB15 dataset. For future works, we propose to use transfer learning methods to reduce the number of parameters in such a complicated structures which can be run on edge devices.

REFERENCES

[1] Bahareh Abolhasanzadeh. 2015. Nonlinear dimensionality reduction for intrusion detection using auto-encoder bottleneck features. In *2015 7th Conference on Information and Knowledge Technology (IKT)*. 1–5. https://doi.org/10.1109/IKT.2015.7288799

[2] Isra Al-Turaiki and Najwa Altwaijry. 2021. A Convolutional Neural Network for Improved Anomaly-Based Network Intrusion Detection. *Big Data* 9, 3 (2021), 233–252. https://doi.org/10.1089/big.2020.0263

[3] Abeer Alalmaie, Priyadarsi Nanda, and Xiangjian He. unpublished. Zero Trust-NIDS: Extended Multi-View Approach for Network Trace Anonymization and Auto-Encoder Convolutional Neural Network for Network Intrusion Detection. *TrustCom* (unpublished).

[4] Mohammed M. Alani. 2022. Implementation-Oriented Feature Selection in UNSW-NB15 Intrusion Detection Dataset. In *Intelligent Systems Design and Applications*, Ajith Abraham, Niketa Gandhi, Thomas Hanne, Tzung-Pei Hong, Tatiane Nogueira Rios, and Weiping Ding (Eds.). Springer International Publishing, Cham, 548–558.

[5] Muder Almiani, Alia AbuGhazleh, Amer Al-Rahayfeh, Saleh Atiewi, and Abdul Razaque. 2020. Deep recurrent neural network for IoT intrusion detection system. *Simulation Modelling Practice and Theory* 101 (2020), 102031. https://doi.org/10.1016/j.simpat.2019.102031 Modeling and Simulation of Fog Computing.

[6] Longy O. Anyanwu, Jared Keengwe, and Gladys A. Arome. 2010. Scalable Intrusion Detection with Recurrent Neural Networks. In *2010 Seventh International Conference on Information Technology: New Generations*. 919–923. https://doi.org/10.1109/ITNG.2010.45

[7] Bo Cao, Chenghai Li, Yafei Song, Yueyi Qin, and Chen Chen. 2022. Network Intrusion Detection Model Based on CNN and GRU. *Applied Sciences* 12, 9 (2022). https://doi.org/10.3390/app12094184

[8] D. E. DENNING. 1987. An Intrusion-Detection Model. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING* 13, 2 (1987), 222–232.

[9] Preethi Devan and Neelu Khare. 2021. Sparse auto encoder driven support vector regression based deep learning model for predicting network intrusions. *Peer-to-Peer Networking and Applications* 14 (2021), 2419–2429.

[10] T. H. Divyasree and K. K. Sherly. 2018. A Network Intrusion Detection System Based On Ensemble CVM Using Efficient Feature Selection Approach. *Procedia Computer Science* 143 (2018), 442–449. https://doi.org/10.1016/j.procs.2018.10.416 8th International Conference on Advances in Computing Communications (ICACC-2018).

[11] Volkmar Frinken and Seiichi Uchida. 2015. Deep BLSTM neural networks for unconstrained continuous handwritten text recognition. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*. 911–915. https://doi.org/10.1109/ICDAR.2015.7333894

[12] Tarfa Hamed, Rozita Dara, and Stefan C. Kremer. 2017. Chapter 6 - Intrusion Detection in Contemporary Environments. In *Computer and Information Security Handbook (Third Edition)* (third edition ed.), J. R. Vacca (Ed.). Morgan Kaufmann, Boston, 109–130. https://doi.org/10.1016/B978-0-12-803843-7.00006-5

[13] Sepp Hochreiter and Jürgen Schmidhuber. 1996. LSTM Can Solve Hard Long Time Lag Problems. In *Proceedings of the 9th International Conference on Neural Information Processing Systems* (Denver, Colorado) *(NIPS'96)*. MIT Press, Cambridge, MA, USA, 473–479.

[14] Ameera S. Jaradat, Malek M. Barhoush, and Rawan S. Bani Easa. 2022. Network intrusion detection system: machine learning approach. *Indonesian Journal of Electrical Engineering and Computer Science* 25, 2 (2022), 1151–1158.

[15] Ahmad Javaid, Quamar Niyaz, Weiqing Sun, and Mansoor Alam. 2016. A deep learning approach for network intrusion detection system. *Eai Endorsed Transactions on Security and Safety* 3, 9 (2016), e2.

[16] Kaiyuan Jiang, Wenya Wang, Aili Wang, and Haibin Wu. 2020. Network Intrusion Detection Combined Hybrid Sampling With Deep Hierarchical Network. *IEEE Access* 8 (2020), 32464–32476. https://doi.org/10.1109/ACCESS.2020.2973730

[17] Dishan Jing and Hai-Bao Chen. 2019. SVM Based Network Intrusion Detection for the UNSW-NB15 Dataset. *2019 IEEE 13th International Conference on ASIC (ASICON)* (2019), 1–4.

[18] Mohammad Humayun Kabir, Md Shahriar Rajib, Abu Saleh Md Towfiqur Rahman, Md. Mahbubur Rahman, and Samrat Kumar Dey. 2022. Network Intrusion Detection Using UNSW-NB15 Dataset: Stacking Machine Learning Based Approach. In *2022 International Conference on Advancement in Electrical and Electronic Engineering (ICAEEE)*. 1–6. https://doi.org/10.1109/ICAEEE54957.2022.9836404

[19] V. Kanimozhi and Prem Jacob. 2019. UNSW-NB15 dataset feature selection and network intrusion detection using deep learning. *International Journal of Recent Technology and Engineering* 7 (01 2019), 443–446.

[20] Sydney M. Kasongo and Yanxia Sun. 2020. Performance Analysis of Intrusion Detection Systems Using a Feature Selection Method on the UNSW-NB15 Dataset. *Journal of Big Data* 7, 105 (2020). https://doi.org/10.1186/s40537-020-00379-6

[21] R. Kaur. 2011. Advances in Intrusion Detection System for WLAN. *Advances in Internet of Things* 1, 3 (2011), 51–54. https://doi.org/10.4236/ait.2011.13007

[22] Jihyun Kim, Jaehyun Kim, Huong Le Thi Thu, and Howon Kim. 2016. Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection. In *2016 International Conference on Platform Technology and Service (PlatCon)*. 1–5. https://doi.org/10.1109/PlatCon.2016.7456805

[23] S. Krishnaveni, S. Sivamohan, S. Sridhar, and S. Prabhakaran. 2022. Network intrusion detection based on ensemble classification and feature selection method for cloud computing. *Concurrency Computat Pract Exper.* 34, 11 (2022).

[24] Vikash Kumar, Ditipriya Sinha, Ayan Kumar Das, Subhash Chandra Pandey, and Radha Tamal Goswami. 2020. An integrated rule based intrusion detection system: analysis on UNSW-NB15 data set and the real time online dataset. *Cluster Comput* 23 (2020), 1397–1418. https://doi.org/10.1007/s10586-019-03008-x

[25] FatimaEzzahra Laghrissi, Samira Douzi, Khadija Douzi, and Badr Hssina. 2021. IDS-attention: an efficient algorithm for intrusion detection systems using attention mechanism. *Journal of Big Data* 8, 1 (2021), 1–21.

[26] Inwoong Lee, Doyoung Kim, and Sanghoon Lee. 2021. 3-D Human Behavior Understanding Using Generalized TS-LSTM Networks. *IEEE Transactions on Multimedia* (2021), 415–428. https://doi.org/10.1109/TMM.2020.2978637

[27] Qi Liu, Veit Hagenmeyer, and Hubert B. Keller. 2021. A Review of Rule Learning-Based Intrusion Detection Systems and Their Prospects in Smart Grids. *IEEE Access* 9 (2021), 57542–57564. https://doi.org/10.1109/ACCESS.2021.3071263

[28] Ahmed Mahfouz, Abdullah Abuhussein, Deepak Venugopal, and Sajjan Shiva. 2020. Ensemble Classifiers for Network Intrusion Detection Using a Novel Network Attack Dataset. *Future Internet* 12, 11 (2020). https://doi.org/10.3390/fi12110180

[29] Soulaiman Moualla, Khaldoun Khorzom, and Assef Jafar. 2021. Improving the Performance of Machine Learning-Based Network Intrusion Detection Systems on the UNSW-NB15 Dataset. *Computational Intelligence and Neuroscience* 2021 (2021). https://doi.org/10.1155/2021/5557577

[30] Nour Moustafa and Jill Slay. 2015. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In *2015 Military Communications and Information Systems Conference (MilCIS)*. 1–6. https://doi.org/10.1109/MilCIS.2015.7348942

[31] B. Mukherjee, L.T. Heberlein, and K.N. Levitt. 1994. Network intrusion detection. *IEEE Network* 8, 3 (1994), 26–41. https://doi.org/10.1109/65.283931

[32] N. Naderi and B. Nasersharif. 2017. Multiresolution convolutional neural network for robust speech recognition. In *2017 Iranian Conference on Electrical Engineering (ICEE)*. 1459–1464. https://doi.org/10.1109/IranianCEE.2017.7985272

[33] N. Naderi, B. Nasersharif, and A. Nikoofard. 2022. Persian speech synthesis using enhanced tacotron based on multi-resolution convolution layers and a convex optimization method. *Multimed Tools Appl* 81 (2022), 3629–3645. https://doi.org/10.1007/s11042-021-11719-w

[34] B. Nasersharif and N. Naderi. 2021. An Information-Theoretic Discussion of Convolutional Bottleneck Features for Robust Speech Recognition. *Iranian Journal of Electrical and Electronic Engineering* 17, 2 (2021). https://doi.org/10.22068/IJEEE.17.2.1563 arXiv:http://ijeee.iust.ac.ir/article-1-1563-en.pdf

[35] Bipraneel Roy and Hon Cheung. 2018. A Deep Learning Approach for Intrusion Detection in Internet of Things using Bi-Directional Long Short-Term Memory Recurrent Neural Network. In *2018 28th International Telecommunication Networks and Applications Conference (ITNAC)*. 1–6. https://doi.org/10.1109/ATNAC.2018.8615294

[36] S. Sangeetha, S. Haripriya, S. G. Mohana Priya, V. Vaidehi, and N. Srinivasan. 2010. Fuzzy Rule-Base Based Intrusion Detection System on Application Layer. In *Recent Trends in Network Security and Applications*, Natarajan Meghanathan, Selma Boumerdassi, Nabendu Chaki, and Dhinaharan Nagamalai (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 27–36.

[37] D. Sathya and S. Sangeetha. 2019. Http Rule Base Intrusion Detection and Prevention System. *International Journal of Innovative Technology and Exploring Engineering (IJITEE)* 8, 9(2) (2019), 438–441.

[38] P. G. V. Suresh Kumar and Shaheda Akthar. 2022. Execution Improvement of Intrusion Detection System Through Dimensionality Reduction for UNSW-NB15 Information. In *Mobile Computing and Sustainable Informatics*, Subarna Shakya, Robert Bestak, Ram Palanisamy, and Khaled A. Kamel (Eds.). Springer Singapore, Singapore, 385–396.

[39] Bayu Adhi Tama, Marco Comuzzi, and Kyung-Hyune Rhee. 2019. TSE-IDS: A Two-Stage Classifier Ensemble for Intelligent Anomaly-Based Intrusion Detection System. *IEEE Access* 7 (2019), 94497–94507. https://doi.org/10.1109/ACCESS.2019.2928048

[40] Mengxuan Tan, Alfonso Iacovazzi, Ngai-Man Man Cheung, and Yuval Elovici. 2019. A Neural Attention Model for Real-Time Network Intrusion Detection. In *2019 IEEE 44th Conference on Local Computer Networks (LCN)*. 291–299. https://doi.org/10.1109/LCN44214.2019.8990890

[41] Tuan A Tang, Lotfi Mhamdi, Des McLernon, Syed Ali Raza Zaidi, and Mounir Ghogho. 2018. Deep Recurrent Neural Network for Intrusion Detection in SDN-based Networks. In *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*.

[42] Abhishek Verma and Virender Ranga. 2019. ELNIDS: Ensemble Learning based Network Intrusion Detection System for RPL based Internet of Things. In *2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU)*. 1–6. https://doi.org/10.1109/IoT-SIU.2019.8777504

[43] Ravi Vinayakumar, Kp Soman, and Prabaharan Poornachandran. 2017. Applying convolutional neural network for network intrusion detection. 1222–1228. https://doi.org/10.1109/ICACCI.2017.8126009

[44] Hanxiang Wang, Yanfen Li, L. Minh Dang, and Hyeonjoon Moon. 2022. An efficient attention module for instance segmentation network in pest monitoring. *Computers and Electronics in Agriculture* 195 (2022), 106853. https://doi.org/10.1016/j.compag.2022.106853

[45] Yuhua Yin, Julian Jang-Jaccard, Wen Xu, Amardeep Singh, Jinting Zhu, Fariza Sabrina, and Jin Kwak. 2022. IGRF-RFE: A Hybrid Feature Selection Method for MLP-based Network Intrusion Detection on UNSW-NB15 Dataset. *arXiv:2203.16365v1* (2022).