Trans-IDS: A Transformer-Based Intrusion Detection System

El Mahdi Mercha[1,2], El Mostapha Chakir[2] and Mohammed Erradi[1]
[1]*ENSIAS, Mohammed V University, Rabat, Morocco*
[2]*HENCEFORTH, Rabat, Morocco*

Keywords: Cyber Security, Intrusion Detection System, Deep Learning, Transformer.

Abstract: The increasing number of online systems and services has led to a rise in cyber security threats and attacks, making Intrusion Detection Systems (IDS) more crucial than ever. Intrusion Detection Systems (IDS) are designed to detect unauthorized access to computer systems and networks by monitoring network traffic and system activities. Owing to the valuable values provided by IDS, several machine learning-based approaches have been developed. However, most of these approaches rely on feature selection methods to overcome the problem of high-dimensional feature space. These methods may lead to the exclusion of important features or the inclusion of irrelevant ones, which can negatively impact the accuracy of the system. In this work, we propose Trans-IDS (transformer-based intrusion detection system), a transformer-based system for intrusion detection, which does not rely on feature selection methods. Trans-IDS learns efficient contextualized representations for both categorical and numerical features to achieve high prediction performance. Extensive experiments have been conducted on two publicly available datasets, namely UNSW-NB15 and NSL-KDD, and the achieved results show the efficiency of the proposed approach.

1 INTRODUCTION

Intrusion detection systems (IDS) are security mechanisms designed to detect unauthorized access to computer systems and networks (Aydın et al., 2022). They monitor network traffic and system activities to identify potentially malicious behavior.

Although IDS has made significant progress in recent years, many current solutions continue to rely on signature-based detection methods instead of using anomaly-based methods. Signature-based methods monitor network traffic for patterns and sequences that match a known attack signature (Zhang et al., 2022). While these methods can effectively detect known attacks, they may not be able to identify new or previously unknown attacks. However, anomaly-based methods learn the normal behavior of a system or network and then identify any deviations from its behavior that may indicate an intrusion.

Over the past few years researchers have directed their efforts towards developing IDS using a variety of machine learning algorithms such as support vector machine (Roopa Devi and Suganthe, 2020) with different feature selecting methods. Recently, deep learning has been widely applied to IDS and has achieved interesting results. Its success in the domain of intrusion detection can be primarily attributed to its ability to automatically learns complex patterns and behaviors from raw data, such as network traffic or system logs, without relying on explicit feature engineering. Several deep learning architectures have been used for IDS such as Recurrent Neural Networks (RNN) (Donkol et al., 2023), and Autoencoders (Basati and Faghih, 2022).

Previous traditional approaches often depend on manually defined features or a subset of features extracted from network traffic data. However, such feature selection techniques may lead to the exclusion of important features or the inclusion of irrelevant ones, increasing the amount of noisy data and affecting the classification accuracy negatively (Ayesha et al., 2020; Taha et al., 2022).

In this work, we suggest a system, namely Trans-IDS (transformer-based intrusion detection system) for intrusion detection. Trans-IDS aims to overcome the limitation of relying on feature selection methods in traditional IDS approaches. Inspired by the success of FT-Transformer (Gorishniy et al., 2021), we use the transformer to automatically learn contextualized representations for all features without relying on feature selection techniques. The proposed system identifies complex patterns and relationships in

Mercha, E., Chakir, E. and Erradi, M.
Trans-IDS: A Transformer-Based Intrusion Detection System.
DOI: 10.5220/0012085800003555
In *Proceedings of the 20th International Conference on Security and Cryptography (SECRYPT 2023)*, pages 402-409
ISBN: 978-989-758-666-8; ISSN: 2184-7711

the data to enhance the performance of IDS. Therefore, extensive experiments have been conducted on two publicly available datasets, namely UNSW-NB15 and NSL-KDD, and the achieved results show the efficiency of the proposed approach. The main contributions are summarized as follows:

- This work proposes a transformer-based method for intrusion detection which can automatically learn contextualized representations for all the features without relying on feature selection techniques.
- Extensive experiments are conducted on two publicly available datasets to investigate the performance of the proposed approach against several baseline models.

2 BACKGROUND ON IDS

IDS is considered as a passive monitoring device used to detect unauthorized access, malicious activity, and other security threats to computer networks, systems, and applications. This section provides an overview of IDS and the threat model considered in this paper.

2.1 IDS Overview

IDSs are divided into two main types: network-based IDS (NIDS) and host-based IDS (HIDS). On the one hand, NIDS monitors network traffic at various points/nodes on the network and can detect suspicious activities, such as port scanning and packet sniffing. On the other hand, HIDS are installed on individual systems and monitor activities on the host, including file changes, system calls, login attempts, and other activities. In this work, we are focusing on NIDS.

A network architecture is composed of multiple nodes, which are the individual devices or computers that are connected to the network. Each node can communicate with other nodes on the network through traffic, which enables data transfer and sharing. Each traffic flow has a source IP address, source port, destination IP address, destination port, and network protocol. The architecture of a network can vary depending on its size and purpose, but it typically includes routers, firewalls, switches, servers, wireless access points, and other networking equipment. The communication process of all the nodes on a network can be formulated by using a tuple of (IP source, port source, IP destination, port destination, and network protocol). The transmission process of the traffic flows between two nodes can be formulated as:

$$H_{(SIP,SP)} \xrightarrow{(NP)} H_{(DIP,DP)} \quad (1)$$

where H represents the host/node, the NP refers to the network protocol. We can represent a network traffic flow as a set of tuples, where each tuple represents a unique communication flow between two devices. For example, we might have a set of traffic flows $F = \{F1, F2, F3, \ldots, Fn\}$, where each Fi represents a distinct communication flow. Specifically, Fi is a tuple $(SIP_i, SP_i, DIP_i, DP_i, NP_i)$ representing a unique communication flow between two devices. SIP_i, SP_i, DIP_i, DP_i, NP_i denote respectively the source IP address, source port number, destination IP address, destination port number, network protocol of the i-th flow.

2.2 Threat Model

We consider a threat model applicable to networks where adversaries lack knowledge of the network topology. Adversaries may launch malicious traffic flows to attack nodes, both from within and outside the network. The malicious traffic flows may take various forms, including Backdoors, Fuzzers, DDoS attacks, and Men in the Middle (MITM) attacks.

Successful attacks may compromise and take control over vulnerable nodes, rendering them untrustworthy, and converting them into malicious ones that launch a large volume of traffic to attack other normal nodes. The goal of the adversaries is to infect as many nodes as possible, eventually leading to the network's paralysis.

IDS sensors, such as port mirroring, packet capture, Network test access points, and netflow, capture each traffic flow and extract statistical features from both the network and transport layer, then transmit them to the IDS for analysis and detection. These features including packet lengths, duration, start time, and source bytes, can indicate potentially malicious behavior. Figure 1 shows a network architecture where numerous IDS monitoring tools (sensors) are positioned at multiple positions in the network to capture and forward traffic to the IDS.

3 RELATED WORKS

Several studies have been conducted to evaluate and compare the performance of various machine learning algorithms on different datasets. (Zhang et al., 2022) applied several machine learning algorithms with feature selection for NIDS. (Das et al., 2021)

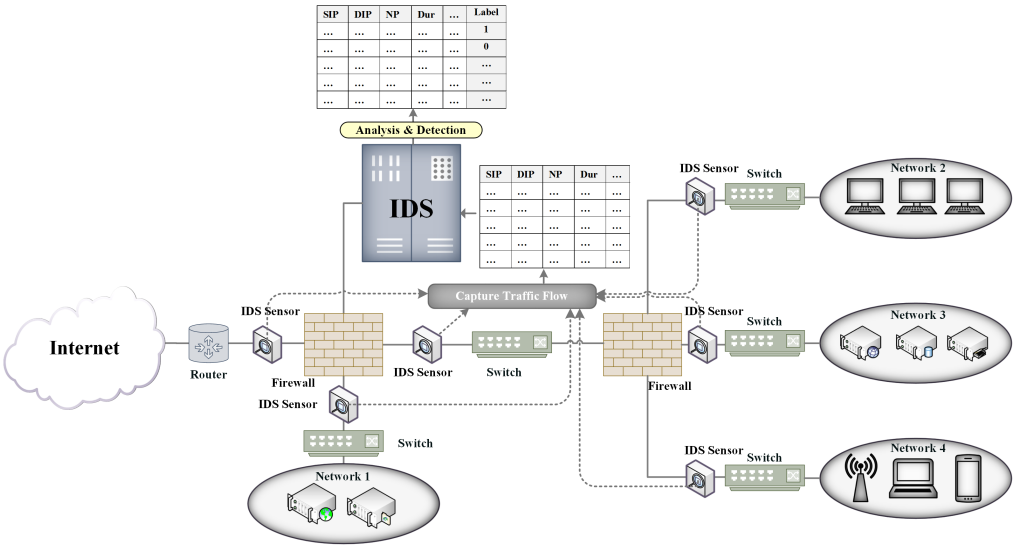

Figure 1: Flowchart of IDS monitoring tools capturing and transmitting traffic to the IDS for analysis and classification of the traffic flow to malicious or benign (0,1).

implemented a supervised ensemble machine learning framework, which incorporates multiple machine learning classifiers with ensemble feature selection technique for NIDS.

Deep learning techniques such as RNN, Graph Neural Networks (GNN), Graph Convolutional Networks (GCN), and Transformer methods have been widely used to IDS with promising results.

(Kasongo, 2023) proposed an IDS framework using different types of RNNs techniques coupled with the XGBoost-based feature selection method. (Caville et al., 2022) presented a GNN approach to IDS that leverages edge features and a graph topological structure in a self-supervised manner. (Zheng and Li, 2019) combined the traffic trace graph with statistical features in the training process, to enhance the classification accuracy of the GCN model. (Sun et al., 2020) built a graph to represent the structure of traffic data, which contains more similar information about the traffic, and combined a two-layer GCN and autoencoder for feature representation learning. (Wang and Li, 2021) proposed a hybrid neural network model by combining transformers and CNN to detect distributed denial-of-service attacks on software-defined networks. (Wu et al., 2022) proposed a robust transformer-based intrusion detection that uses the positional embedding technique to associate sequential information between features.

While the use of machine learning and deep learning techniques in IDS has shown promising results, they rely on manually defined features or a subset of features extracted from network traffic data. However, such feature selection techniques can result in the loss of important information that can be useful for more accurate predictions.

4 THE PROPOSED METHOD

We consider a network architecture made up of multiple nodes, where each node has a unique IP address for communication with other nodes on the network. Each node communicates with other nodes through traffic flow, which is characterized by multiple features. These features can be divided into numerical and categorical features. Numerical features may include variables such as the size of the data packets, and the duration. However, categorical features may include variables such as the protocol type, and the source and destination IP addresses. In this section, we provide more details about the global architecture of Trans-IDS shown in figure 2.

4.1 Feature Embedding

We consider a dataset $D = (X, Y)$ where $X \equiv \{X^{cat}, X^{num}\}$ is the features, X^{cat} and X^{num} denote respectively the categorical and numerical features, and Y the list of the labels (Normal or attack). For each sample i in the dataset, let $X_i^{cat} = \{x_1^{cat}, x_2^{cat}, ..., x_n^{cat}\}$ the list of categorical features with each x_j^{cat} being a categorical feature, for $j \in \{1, ..., n\}$. Also, let $X_i^{num} = \{x_1^{num}, x_2^{num}, ..., x_m^{num}\}$ the list of numerical features with each x_j^{num} being a numerical feature, for $j \in \{1, ..., m\}$. The Trans-IDS transforms each numerical feature x_i^{num} into parametric embedding $e_{\phi_i}(x_i^{num})$

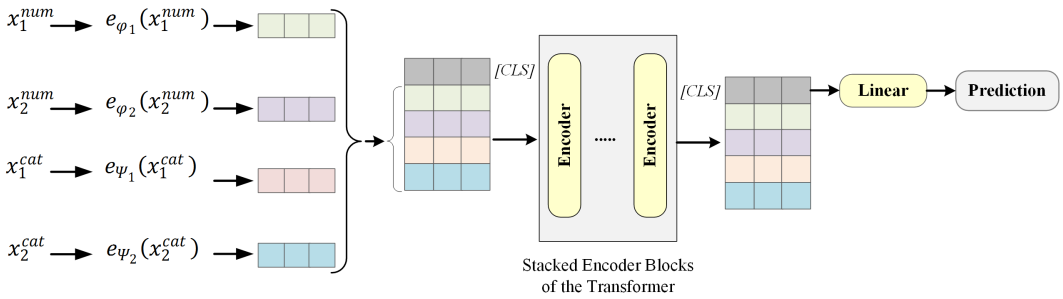

Figure 2: The global Trans-IDS architecture.

of dimension d as follows:

$$e_{\phi_i}(x_i^{num}) = x_i^{num} \times W_i^{num} + b_i^{num} \qquad \in \mathbb{R}^d \qquad (2)$$

where W_i^{num} is a hyper-parameter vector of dimension d, and b_i^{num} is the bias of the i-th numerical feature. However, each categorical feature x_i^{cat} is transformed into parametric embedding $e_{\psi_i}(x_i^{cat})$ of dimension d as follows:

$$e_{\psi_i}(x_i^{cat}) = e_i^T W_i^{cat} + b_i^{cat} \qquad \in \mathbb{R}^d \qquad (3)$$

where e_i^T is the transpose of the one-hot vector for the corresponding categorical feature, W_i^{cat} is a lookup table for categorical features, b_i^{cat} is the bias of the i-th categorical feature. The numerical embeddings are concatenated along with categorical embeddings to form a matrix H^0 of dimension $(n+m,d)$.

$$H^{(0)} = concat[e_{\phi_1}(x_1^{num}),...,e_{\phi_m}(x_m^{num}), e_{\psi_1}(x_1^{cat}),...,e_{\psi_n}(x_n^{cat})] \qquad \in \mathbb{R}^{n+m\times d} \qquad (4)$$

Then, the embedding of the [CLS] (Devlin et al., 2018) token is appended to $H^{(0)}$, and the resulting matrix is inputted to stacked encoder blocks of the transformer to learn contextualized representations.

4.2 Transformer Encoder

We propose a slight variant of the transformer encoder block introduced in (Vaswani et al., 2017). The core component of the transformer encoder block is the multi-head self-attention layer, which allows the model to capture dependencies between different positions in the input. The self-attention layer comprises three parametric matrices Key, Query, and Value. It computes a weighted sum of the value vectors, where the weights are determined by the similarity between the query and key vectors. Formally, let $K \in \mathbb{R}^{s\times k}$, $Q \in \mathbb{R}^{s\times k}$ and $V \in \mathbb{R}^{s\times v}$ be the matrices comprising key, query and value vectors of all the numerical and categorical embeddings. For each query vector Q_i, a score is computed by taking the dot product between Q_i and each key vector K_j. This can be expressed as:

$$score(Q_i,K_j) = Q_i * K_j \qquad (5)$$

The scores are then scaled by dividing them by the square root of the dimension of the key vectors k:

$$score(Q_i,K_j) = Q_i * K_j/\sqrt{k} \qquad (6)$$

The scores are passed through a softmax function to obtain attention weights that sum to 1:

$$weight(Q_i,K_j) = softmax(score(Q_i,K_j)) \qquad (7)$$

Then, the attention weights are used to compute a weighted sum of the value vectors V_j, where the weights are the attention weights. This can be expressed as:

$$output(Q_i) = sum(weight(Q_i,K_j) * V_j) \qquad (8)$$

This process is repeated for each head in the multi-head attention mechanism, and the results are concatenated and passed through a linear layer to obtain the final output of the attention module.

The output of the multi-head attention is then passed through two feed-forward layers, where the first layer expands the embedding and the second layer projects it back to its original size. After the feed-forward layers, element-wise addition and layer normalization (Add & Norm) are performed to normalize the output of the layer and facilitate the training.

Figure 3 shows the architecture of the transformer encoder.

Finally, the learned representation of the [CLS] token is used to predict the label $\hat{y}$ through a linear function as follows:

$$\hat{y} = \sigma(W.x + b) \qquad (9)$$

where x denotes the learned vector of the [CLS] token, σ is the sigmoid activation function, and W and b denote the weight and bias of the linear layer respectively.

5 EXPERIMENTS

To evaluate the performance of the proposed Trans-IDS, we conduct several experiments. In this section,

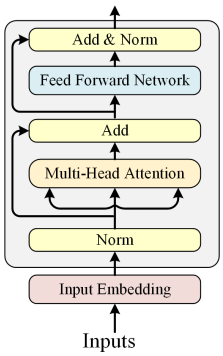

Figure 3: The architecture of the transformer encoder.

we provide more details about the experiment settings and we discuss the achieved results.

5.1 Datasets

5.1.1 UNSW-NB15 Dataset

The UNSW-NB15 is a dataset that is widely used to evaluate IDS. It contains 42 attributes, consisting of 3 categorical inputs and 39 numerical inputs. The numerical inputs have binary, integer, and float data types. Table 1 provides a detailed description of the distribution of attacks across the UNSW-NB15 subsets.

5.1.2 NSL-KDD Dataset

The NSL-KDD is a public network traffic dataset that was created for the purpose of evaluating IDS. The dataset was developed as an improvement over the earlier KDD Cup 99 dataset, which had some limitations and flaws. The dataset contains a total of 41 features. Table 2 provides a detailed description of the distribution of attacks across the NSL-KDD subsets.

Table 1: UNSW-NB15 distribution subsets: Training (Train), Validation (Val), and Test.

Attack Type	All	Train	Val	Test
Normal	56000	41911	14089	37000
Generic	40000	30081	9919	18871
Exploits	33393	25034	8359	11132
Fuzzers	18184	13608	4576	6062
DoS	12264	9237	3027	4089
Reconnaissance	10491	7875	2616	3496
Analysis	2000	1477	523	677
Backdoor	1746	1330	416	583
Shellcode	1133	854	279	378
Worms	130	99	31	44
Total	**175341**	**131506**	**43835**	**82332**

Table 2: NSL-KDD distribution data subsets: Training (Train), Validation (Val), and Test.

Attack Type	All	Train	Val	Test
Normal	67343	50494	16849	9711
DoS	45927	34478	11449	7458
Probe	11656	8717	2939	2754
R2L	995	747	246	2421
U2R	52	42	10	200
Total	**125973**	**94480**	**31493**	**22544**

5.2 Baseline Models

We compare the Trans-IDS against several baselines and state-of-the-art methods. The baselines include:

- **MCA-LSTM:** a network intrusion detection model based on the multivariate correlations analysis – long short-term memory network (Dong et al., 2020)
- **RNN:** a deep learning approach for intrusion detection using recurrent neural networks (Yin et al., 2017).
- **FFDNN:** a feed-forward deep neural network wireless IDS using a wrapper-based feature extraction unit (Kasongo and Sun, 2020).
- **Simple RNN:** Recurrent neural network implemented in conjunction with a feature selection method that is inspired by the Extreme Gradient Boosting (XGBoost) algorithm (Kasongo, 2023).
- **LSTM:** Long short-term memory used in combination with a feature selection method that is inspired from XGBoost (Kasongo, 2023).
- **GRU:** Gated Recurrent Unit developed in combination with an XGBoost-inspired feature selection method (Kasongo, 2023).
- **CNN-LSTM:** a combination of convolutional neural networks and long short-term memory for IDS (Hsu et al., 2019).

5.3 Experiment Settings

In our experiments, we divided the training set of UNSW-NB15 into two subsets, 75% of the training data and 25% for validation. Furthermore, we reduce the problem to binary classification. So, we project all the attacks into a single class, and we keep the class normal. In addition, we distinguish the categorical and numerical features (see table 3). The same process is applied to the NSL-KDD dataset. Table 4 shows the sets of categorical and numerical features for this dataset.

We performed hyper-parameter tuning for the Trans-IDS to define the optimal values for various

parameters that significantly impact the performance of the model. The best settings were 16 for embedding dimension, 4 for the number of transformer blocks, 12 and 8 for the number of attention heads for UNSW-NB15 and NSL-KDD, respectively, 0.2 for the dropout rate in the transformer and multilayer perceptron, 0.001 for the learning rate, and 10 for the number of epochs. For the evaluation metric, we use accuracy. Finally, the entire architecture is trained end-to-end using backpropagation and Adam with decoupled weight decay (Loshchilov and Hutter, 2017).

Table 3: UNSW-NB15 features description.

Numerical Features	'dur', 'sbytes', 'dbytes', 'sttl', 'dttl', 'sload', 'dload', 'spkts', 'dpkts', 'stcpb','dtcpb', 'smean', 'dmean', 'response_body_len', 'sjit', 'djit', 'sinpkt', 'dinpkt', 'tcprtt', 'synack', 'swin', 'ackdat', 'ct_ftp_cmd', 'rate', 'ct_srv_src', 'ct_srv_dst', 'dwin', 'ct_dst_ltm', 'ct_src_ltm', 'ct_src_dport_ltm', 'ct_dst_sport_ltm', 'ct_dst_src_ltm', 'trans_depth'
Categorical Features	'proto', 'state', 'sloss', 'dloss', 'service', 'is_sm_ips_ports', 'ct_state_ttl', 'ct_flw_http_mthd', 'is_ftp_login'

Table 4: NSL-KDD features description.

Numerical Features	'duration', 'src_bytes', 'dst_bytes', 'num_failed_logins','wrong_fragment', 'urgent', 'hot', 'root_shell', 'num_compromised', 'su_attempted', 'num_root', 'num_file_creations', 'num_shells', 'num_access_files', 'num_outbound_cmds', 'count', 'dst_host_same_srv_rate', 'srv_count', 'serror_rate', 'srv_serror_rate', 'rerror_rate', 'srv_rerror_rate', 'same_srv_rate', 'diff_srv_rate', 'srv_diff_host_rate', 'dst_host_count', 'dst_host_srv_count', 'dst_host_serror_rate', 'dst_host_srv_serror_rate', 'dst_host_rerror_rate', 'dst_host_srv_rerror_rate', 'dst_host_same_src_port_rate', 'dst_host_srv_diff_host_rate', 'dst_host_diff_srv_rate'
Categorical Features	'protocol_type', 'service', 'flag', 'logged_in', 'is_host_login', 'is_guest_login', 'land'

5.4 Results & Discussion

A comprehensive experiment is conducted on two publicly available datasets. The achieved results reveal that the proposed Trans-IDS outperformed all the baseline models in the UNSW-NB15 dataset and achieve competitive results on the NSL-KDD dataset without relying on any feature selection method.

In Table 5, we provide a comparison between the proposed Trans-IDS and the baseline models presented in (section 5.2). Based on a more detailed performance analysis, it was found that Simple RNN, LSTM, and GRU achieved competitive results compared to the Trans-IDS model. This suggests that combining recurrent models with feature selection methods, especially XGBoost can be an effective approach for building intrusion detection systems. However, such feature selection techniques may lead to the exclusion of important features or the inclusion of irrelevant ones, which can negatively impact the accuracy of the system (Ayesha et al., 2020; Taha et al., 2022). This can be observed in the results achieved in the UNSW-NB15 dataset. This finding is also supported by the results obtained by MCA-LSTM and FFDNN models, which used different feature selection methods. Moreover, it was found that using just an RNN model without any feature selection method resulted in poor performance.

From the achieved results in the NSL-KDD, we can see that Trans-IDS achieves modest results compared with the other baseline models. The best results were achieved by Simple RNN, LSTM, and GRU models using XGBoost for feature selection.The performance of Trans-IDS is mainly affected by the modest size of the dataset used to train the model. Generally, the performance of transformer models such as Trans-IDS is known to improve with larger amounts of training data. However, the NSL-KDD dataset is relatively modest in size, which may limit the ability of the model to learn complex patterns and generalize to new examples.

Overall, while Trans-IDS may not have achieved the best performance on the NSL-KDD dataset, it is important to consider the limitation of the amount of data used to train the model when interpreting these results. Further analysis may be needed to identify specific areas for improvement and to validate the performance of the model on other datasets.

In the Trans-IDS, importance weights are used in the self-attention mechanism to determine the importance of each feature in the input when computing the representation of each feature. These weights deter-

Table 5: Comparison with other methods (Binary classification).

Method	Dataset	FS	Accuracy
MCA-LSTM	UNSW-NB15	IG	88.11%
RNN	UNSW-NB15	–	83.28%
FFDNN	UNSW-NB15	ExtraTrees	87.10%
Simple RNN	UNSW-NB15	XGBoost	87.07%
LSTM	UNSW-NB15	XGBoost	85.08%
GRU	UNSW-NB15	XGBoost	88.42%
Trans-IDS	**UNSW-NB15**	**–**	**89.14%**
MCA-LSTM	NSL-KDD	IG	80.52%
CNN-LSTM	NSL-KDD	–	74.77%
Simple RNN	NSL-KDD	XGBoost	83.70%
LSTM	**NSL-KDD**	**XGBoost**	**88.13%**
GRU	NSL-KDD	XGBoost	84.66%
Trans-IDS	NSL-KDD	–	81.86%

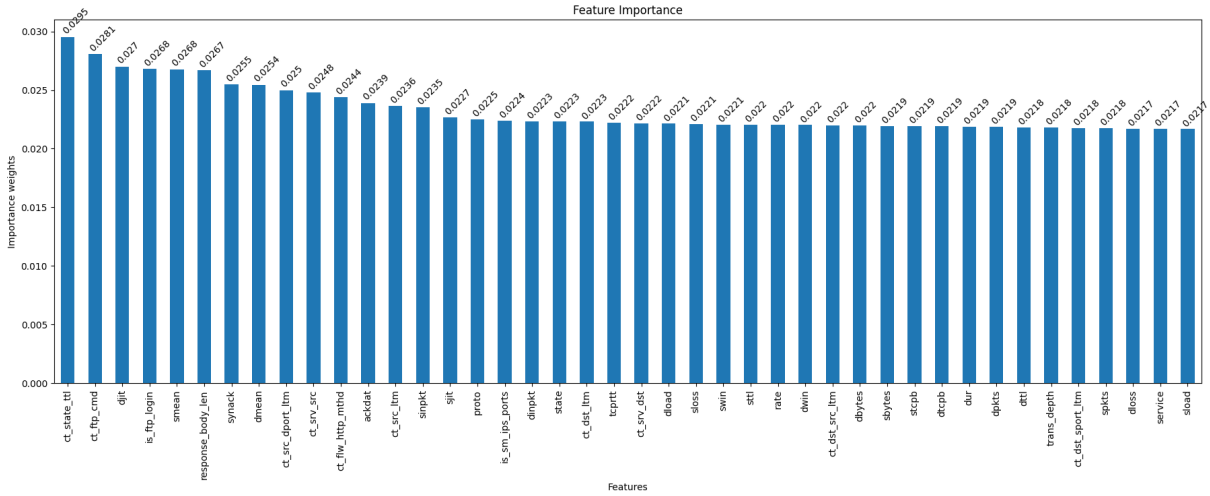

Figure 4: The importance weights for all features generated by Trans-IDS on the UNSW-NB15 dataset.

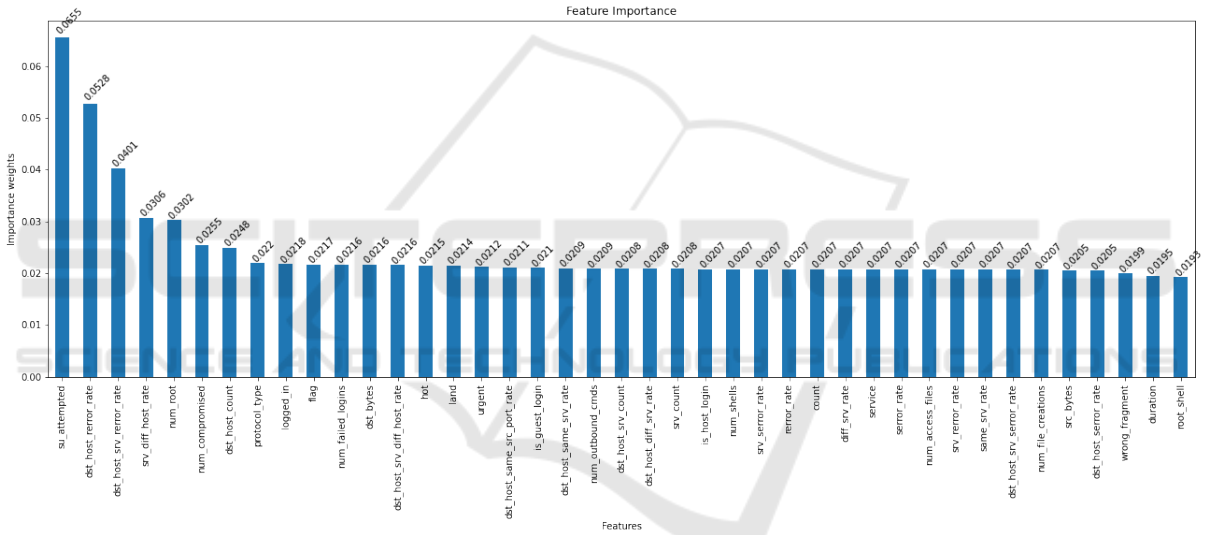

Figure 5: The importance weights for all features generated by Trans-IDS on the NSL-KDD dataset.

mine how much attention to pay to each feature when computing the representation of the current feature. Features with high weights will have a greater impact on the representation of the current feature, while features with low weights will have less impact. Figure 4 and 5 show the importance weights generated by Trans-IDS in UNSW-NB15 and NSL-KDD datasets respectively. The importance weights generated reveal that considering all features with slightly different weights is very efficient to learn predictive representations instead of focusing on some specific features. Hence, developing an IDS without relying on any feature selection technique can enhance the performance and robustness of the approach. This is because feature selection techniques can sometimes remove important features that contribute to the predictive power of the system, leading to a loss in performance and robustness.

6 CONCLUSION

In this paper, we proposed Trans-IDS (transformer-based intrusion detection system), a system that presents an approach for intrusion detection by using a transformer-based method. This method learns contextualized representations for both categorical and numerical features without relying on feature selection techniques. By avoiding the need for feature selection, Trans-IDS is able to identify complex patterns that result in a more accurate IDS system. The importance weights generated by Trans-IDS further

confirmed the significance of considering all features with slightly different weights, as opposed to focusing on specific features.

The experimental results on two publicly available datasets, namely UNSW-NB15 and NSL-KDD, demonstrate the effectiveness of the Trans-IDS system. Overall, the suggested approach for intrusion detection has the potential to overcome some of the limitations of traditional methods while avoiding the need for feature selection techniques.

In future work, we plan to investigate the scalability of Trans-IDS and its performance on other datasets.

REFERENCES

Aydın, H., Orman, Z., and Aydın, M. A. (2022). A long short-term memory (lstm)-based distributed denial of service (ddos) detection and defense system design in public cloud network environment. *Computers & Security*, 118:102725.

Ayesha, S., Hanif, M. K., and Talib, R. (2020). Overview and comparative study of dimensionality reduction techniques for high dimensional data. *Information Fusion*, 59:44–58.

Basati, A. and Faghih, M. M. (2022). Pdae: Efficient network intrusion detection in iot using parallel deep auto-encoders. *Information Sciences*, 598:57–74.

Caville, E., Lo, W. W., Layeghy, S., and Portmann, M. (2022). Anomal-e: A self-supervised network intrusion detection system based on graph neural networks. *Knowledge-Based Systems*, 258:110030.

Das, S., Saha, S., Priyoti, A. T., Roy, E. K., Sheldon, F. T., Haque, A., and Shiva, S. (2021). Network intrusion detection and comparative analysis using ensemble machine learning and feature selection. *IEEE Transactions on Network and Service Management*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dong, R.-H., Li, X.-Y., Zhang, Q.-Y., and Yuan, H. (2020). Network intrusion detection model based on multivariate correlation analysis–long short-time memory network. *IET Information Security*, 14(2):166–174.

Donkol, A. A. E.-B., Hafez, A. G., Hussein, A. I., and Mabrook, M. M. (2023). Optimization of intrusion detection using likely point pso and enhanced lstm-rnn hybrid technique in communication networks. *IEEE Access*, 11:9469–9482.

Gorishniy, Y., Rubachev, I., Khrulkov, V., and Babenko, A. (2021). Revisiting deep learning models for tabular data. *Advances in Neural Information Processing Systems*, 34:18932–18943.

Hsu, C.-M., Hsieh, H.-Y., Prakosa, S. W., Azhari, M. Z., and Leu, J.-S. (2019). Using long-short-term memory based convolutional neural networks for network intrusion detection. In *Wireless Internet: 11th EAI International Conference, WiCON 2018, Taipei, Taiwan, October 15-16, 2018, Proceedings 11*, pages 86–94. Springer.

Kasongo, S. M. (2023). A deep learning technique for intrusion detection system using a recurrent neural networks based framework. *Computer Communications*, 199:113–125.

Kasongo, S. M. and Sun, Y. (2020). A deep learning method with wrapper based feature extraction for wireless intrusion detection system. *Computers & Security*, 92:101752.

Loshchilov, I. and Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Roopa Devi, E. and Suganthe, R. (2020). Enhanced transductive support vector machine classification with grey wolf optimizer cuckoo search optimization for intrusion detection system. *Concurrency and Computation: Practice and Experience*, 32(4):e4999.

Sun, B., Yang, W., Yan, M., Wu, D., Zhu, Y., and Bai, Z. (2020). An encrypted traffic classification method combining graph convolutional network and autoencoder. In *2020 IEEE 39th International Performance Computing and Communications Conference (IPCCC)*, pages 1–8. IEEE.

Taha, A., Cosgrave, B., and Mckeever, S. (2022). Using feature selection with machine learning for generation of insurance insights. *Applied Sciences*, 12(6):3209.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, H. and Li, W. (2021). Ddostc: A transformer-based network attack detection hybrid mechanism in sdn. *Sensors*, 21(15):5047.

Wu, Z., Zhang, H., Wang, P., and Sun, Z. (2022). Rtids: A robust transformer-based approach for intrusion detection system. *IEEE Access*, 10:64375–64387.

Yin, C., Zhu, Y., Fei, J., and He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. *Ieee Access*, 5:21954–21961.

Zhang, C., Jia, D., Wang, L., Wang, W., Liu, F., and Yang, A. (2022). Comparative research on network intrusion detection methods based on machine learning. *Computers & Security*, page 102861.

Zheng, J. and Li, D. (2019). Gcn-tc: combining trace graph with statistical features for network traffic classification. In *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE.