ABSTRACT Due to the rapid growth in network traffic and increasing security threats, Intrusion Detection Systems (IDS) have become increasingly critical in the field of cyber security for providing secure communications against cyber adversaries. However, there exist many challenges for designing a robust, efficient and accurate IDS, especially when dealing with high-dimensional anomaly data with unforeseen and unpredictable attacks. In this paper, we propose a Robust Transformer-based Intrusion Detection System (RTIDS) reconstructing feature representations to make a trade-off between dimensionality reduction and feature retention in imbalanced datasets. The proposed method utilizes positional embedding technique to associate sequential information between features, then a variant stacked encoder-decoder neural network is used to learn low-dimensional feature representations from high-dimensional raw data. Furthermore, we apply self-attention mechanism to facilitate network traffic type classifications. Extensive experiments reveal the effectiveness of the proposed RTIDS on two publicly available real traffic intrusion detection datasets named CICIDS2017 and CIC-DDoS2019 with F1-Score of 99.17% and 98.48% respectively. A comparative study with classical machine learning algorithm support vector machine (SVM) and deep learning algorithms that include recurrent neural network (RNN), fuzzy neural network (FNN), and long short-term memory network (LSTM) is conducted to demonstrate the validity of the proposed method.

INDEX TERMS Intrusion detection, feature representation, self-attention mechanism, transformer.
In recent years, self-attention mechanism based models such as transformer and its variants have obtained huge success in text classification, dialogue recognition, machine translation, and other natural language processing (NLP) tasks. The essence of transformer is to pre-train on a vast text corpus first then applied the trained model on a smaller task-dedicated dataset to fine tune [18]. Transformer has the advantage of computation efficiency and scalability, it has also been used in the field of image classification and computer vision [19].

Enlightened by transformer’s ability to handle ordered sequences of data, Huang et al. [20] used a variant of transformer to detect system log anomalies and proved its robustness on unstable log data. Bikmukhamedov et al. [21] applied a transformer model to classify traffic data and obtained good result.

In this paper, the main goal of our research is to propose a robust transformer-based intrusion detection system (RTIDS) that is able to process a large volume of complicated raw network data efficiently and provide effective detection performance. RTIDS consists of three modules and features an innovative hierarchy self-attention design inspired by the Transformer model [22]. Specifically, we apply input and positional embedding to convert input network traffic into fixed-dimension vectors as input representations. Then we use stacked encoders and decoders for feature extraction and learning the contextual relations between inputs. Since the input features have different impacts on the classification result, we utilize self-attention mechanism to learn different weights of those feature representations. Additionally, the performance of RTIDS was compared to that of four other mainstream machine learning and deep learning algorithms using two popular intrusion detection evaluation datasets (CICIDS2017 and CIC-DDoS2019), and RTIDS has shown significant performance improvement in comparison with other popular intrusion detection methods.

The main contributions of this article can be summarized as follows:

- We present an innovative Robust Transformer-based Intrusion Detection System (RTIDS), which efficiently extracts and transforms high-dimensional raw data into low-dimensional representations.
- The RTIDS can effectively balance the dimensionality reduction and feature retention in highly imbalanced and high-dimensional datasets.
- We design a self-attention mechanism that can capture contextual information between network traffic features for detecting intrusions.
- The RTIDS has a much higher detection performance than other popular intrusion detection models.

The rest of this article is organized as follows: section 1 is the introduction of our study. Section 2 summarizes related works. In section 3, we illustrate the background information and the proposed model in detail. Section 4 presents the experimental setup used in our study. In section 5, we evaluate the experimental results and make a comparative analysis with another four intrusion detection methods. Finally, section 6 concludes this article.

II. RELATED WORK
To detect and prevent networks from cyber attacks, researchers have invested much effort in proposing various Network Intrusion Detection Systems (NIDS) [23], [24]. In general, NIDS has two main categories: signature-based IDS and anomaly-based IDS.

Signature-based IDS detects threats by comparing the network activities with the known indicator of compromise (IOC), which could be a hash value of a file, a malicious IP address, etc. [25]. Chowdhury et al. [26] suggested combining two machine learning algorithms together to classify signature-based intrusions. Their work applied a simulated annealing technique to generate three random feature sets and an SVM algorithm to identify anomalous behaviors. Veeraiah et al. [27] built a trust-aware signature-based IDS to identify potential intrusions in the MANET nodes using trust tables. In their study, they utilized fuzzy Naïve Bayes and fuzzy clustering algorithms. Both He [28] and Sutskever [29] designed signature-based routing protocols for detecting Sybil attacks that exist in the Internet of Things. However, due to the nature of detecting known threats, the outdated and limited sources of IOCs, nowadays, a considerable number of sophisticated cyber attacks can bypass the signature-based IDS easily.

In contrast, anomaly-based intrusion detection systems compare all network activities with a pre-trained and normalized baseline that presents how the system normally behaves, which makes the system be able to detect unknown malicious network activities. Alamiedy et al. [30] proposed an enhanced anomaly-based IDS model based on multi-objective grey wolf optimization (GWO) algorithm. The GWO algorithm was employed as a feature selection mechanism to identify the most relevant features from the NSL-KDD dataset, which contributed to high classification accuracy. Furthermore, a support vector machine was used to estimate the capability of selected features in predicting attacks accurately. Satam et al. [31] presented a Wireless Intrusion Detection System (WIDS) to detect attacks on Wi-Fi networks. In this approach, they used n-grams to model the normal behavior of the Wi-Fi protocol and used Random Forest, AdaBoost, and other machine learning methods to classify Wi-Fi traffic flows. Gothawal et al. [32] formulated a game-theoretic model-based anomaly Intrusion Detection System (IDS) to detect the RPL attacks and verify their malicious activities. The proposed approach consists of two interrelated formulations, such as a stochastic game for attack detection and an evolutionary game for attack confirmation.

To further empower the IDS to automatically detect network intrusions, Saeed et al. [33] applied artificial neural network (ANN), probabilistic neural network, and chi-square algorithms to detect distributed denial of service (DDoS) attacks. Mahmood et al. [34] combined a log tracking model and a spatio-temporal ML model to build...
an anomaly detection framework. However, due to the high-dimensional presentations of the raw data nowadays, these machine learning-based models cannot efficiently process the data in a timely manner and have a poor detection performance [35], [36].

In recent years, deep learning techniques have been widely used in the field of intrusion detection and has achieved remarkable results. Moustafa et al. [37] proposed an adversarial statistical learning mechanism that applied Outlier Dirichlet Mixture-based ADS (ODM-ADS) method to detect abnormal behaviors for the KDD-99 dataset. Jiang et al. [38] developed a novel dense random neural network that included a hidden Markov model (HMM) algorithm to detect network attacks. Iranmanesh, Saeid, et al. [39] proposed a heuristic distributed scheme (HIDE) to validate the mobility pattern of vehicles and identify malicious vehicles by penalizing or rewarding vehicles based on the contacts’ conformation. Maseer et al. [40] applied 10 popular supervised and unsupervised machine learning algorithms for identifying effective and efficient anomaly-based IDS (AIDS) networks and computers. Their models were tested by using a recent and highly unbalanced multiclass CICIDS2017 dataset that involves real-world network attacks. Moreover, Abdulhammed et al. [41] implemented a web attack detection system based on distributed edge devices, which employed multiple concurrent learning models to improve stability and performance. Furthermore, the researcher also found that recurrent neural network (RNN) based IDS can effectively detect network intrusions and identify attack types on the NSL-KDD dataset [42], [43].

Although those deep learning-based models can improve detection performance dramatically [44], some challenges cannot be overlooked. RNN model is too slow to take full advantage of modern fast computing devices and tends to forget long-term information. In order to solve this problem, Chung et al. [45] proposed Gated Recurrent Unit (GRU) method to help examine long time series. While CNN model easily loses much valuable information due to the pooling layer and requires a large dataset. Ding et al. [46] presented Asymmetric Convolution Block (ACB), an architecture-neural structure as a CNN building block, which uses 1D asymmetric convolutions to strengthen the square convolution kernels. Their model can help retain important feature information and also reduce sample dependency. Wang et al. [47] designed a hybrid neural network structure called DDoS_TC that combined transformers and a convolutional neural network (CNN) to detect distributed denial-of-service (DDoS) attacks on software-defined network (SDN) and tested on the dedicated DDoS testbed dataset CICDDoS2019.

III. BACKGROUND AND PROPOSED FRAMEWORK

In this section, we explore the background knowledge of the transformer model and the detailed structure and processing mechanisms of the proposed robust Transformer-based intrusion detection system (RTIDS). At first, We introduce the related background information about the transformer model, then we present the general framework for the proposed method. Lastly, we illustrate the concrete design of the detection model.

A. TRANSFORMER MODEL

The Transformer model is similar to most competitive neural sequence conduction models, and it also uses an encoder-decoder structure. The encoder maps the input symbolic representation sequence \( (x^1, x^2, \ldots, x^n) \) into a continuous representation sequence \( z = (z^1, z^2, \ldots, z^m) \). Every step of the model is autoregressive, which means the output generated by each encoder or decoder is the input of the next encoder or decoder except the input of the first encoder at the bottom of the encoder stack [48], [49]. Transformer model aims at transforming the input feature sequence to the corresponding vector representations. A major difference between transformer and RNN is a self-attention mechanism that utilizes attention matrices instead of the recurrent connection [50].

B. RTIDS FRAMEWORK

We propose an innovative hierarchical transformer structure-based IDS for efficiently processing complex network traffic data without losing critical details. Fig. 1 shows the overall framework of RTIDS.

RTIDS consists of three components: data preparation module, RTIDS model construction module, and real-time intrusion detection module. As part of the data preparation module, raw network traffic data are processed through four steps: data cleaning, data normalization, feature selection, and dataset splitting. Afterward, we train a variant Transformer model that is fine-tuned to detect abnormal network activities in the RTIDS model construction module. Three units comprise the real-time intrusion detection module. The network connection unit is responsible for receiving and forwarding network packets, and the intrusion detection unit takes advantage of the well-tuned RTIDS model to detect suspicious network activities. Depending on the type of attacks detected, the mitigation unit employs predetermined strategies in order to minimize the system’s risk.

C. MODEL DESIGN

The detection model is the brain of RTIDS, and it is composed of three components: input embedding, encoder and decoder stacks, and the softmax layer. Typically, the model presents all raw inputs with equal-length vectors by using input embedding. Encoders and decoders then process these vectors via mechanisms of masked multi-head self-attention in order to integrate information of the inputs and help enhance the model. Lastly, the softmax layer is used to calculate the malicious probability of the network activities. Algorithm 1 illustrates the pseudocode description of the proposed framework.
FIGURE 1. The framework of RTIDS. RTIDS consists of three modules: the data preparation module, the model construction module and the real-time intrusion detection module.

Algorithm 1 RTIDS Algorithm

**Input:** Training set $S = (x_i, y_i), i = 1, 2, \ldots, N$, $x_i$ is the network traffic sample, $y_i$ is the corresponding label

**Output:** Classification probabilities of the predicted class

1. for $i \leftarrow 0$ until numOfEpochs do
2. for Sample s:Batch do
   - get its vectorized representation $s_r$
   - put $s_r$ into encoder and decoder stacks for feature extraction and selection
   - use transformerModel.MultiHeadAttention function to calculate the attention scores of features
   - use transformerModel.SoftMax function to get classification probabilities
3. end for
   - use stochastic gradient descent (SGD) algorithm to minimize loss function
4. end for

1) INPUT EMBEDDING
In our model, we first apply embedded techniques to process the raw data inputs. Due to network traffic contains some sequential feature information such as source and destination port number and IP address quadruple. Our model includes additional positional encoding information at the bottom of the encoder stack to take advantage of the input features’ location information. Since the positional embedding dimensions are identical to input embedding dimensions, the final input of the self-attention layer is the summation of embedding and positional encoding. Moreover, the input embedding uses the sine function and cosine function with different frequencies to implement positional encoding. In particular, the odd positions are encoded with a cosine function, and the even positions are encoded with a sine function, shown in Eq. (1) and Eq. (2).

$$PE(pos, 2i) = \sin(pos/1000^{2i/d_{model}})$$ (1)

$$PE(pos, 2i + 1) = \cos(pos/1000^{2i/d_{model}})$$ (2)

2) ENCODER AND DECODER STACK

a: ENCODER STACK
There are six encoders in the encoder stack, and each encoder is composed of a multi-head self-attention network and a point-wise feed-forward network (FFN). The dimension of FFN layers is a hyperparameter that can be adjusted during training. For the optimal performance, we assign 1024 neurons in FFN layers and set 32 as the padding size of embedding. Additionally, residual connection and layer regularization are used to calculate the results of each sub-layer, which are then passed to the next encoder in the stack (see Fig. 2).

b: DECODER STACK
We add an additional multi-headed masked self-attention sub-layer into each decoder in contrast to the original transformer
model. In order to improve the robustness of the proposed intrusion detection model, we mask a portion of features randomly and then predict them using other unmasked features. Additionally, to maintain the hierarchical structure of our model, we deploy six decoders in the decoder stack. After reconstructing encoder and decoder stacks, we apply the softmax layer as the final output layer for classification.

As depicted in Fig. 2, there is a residual connection between input’s self-attention sublayer and point-wise feed-forward network, followed by layer normalization. The purpose of these techniques is to improve the model’s performance by addressing potential problems, such as the vanishing gradient and the covariate shift. In addition to the attention sublayer, each encoder and decoder also contains a fully connected sublayer called point-wise feed-forward network. It contains two linear transformations activated by the ReLU function $\sigma$, see Eq. (3) for illustration. Furthermore, identical weights are used for each row of the attention matrices, which can be considered as a convolution on every row of the attention-transformed matrix. It can be viewed that this step enriches the embeddings with additional information.

$$FFN(x) = \sigma(max(0, xW_1 + b_1)W_2 + b_2)$$  \hspace{1cm} (3)

3) MASKED SELF-ATTENTION MECHANISM

The masked scaled dot-product attention mechanism for head $h$ in decoder is shown in Fig. 3. It takes input as a set of queries($Q$), keys($K$) and values($V$). Masked scaled dot-product attention can be computed with Eq. (4), where $d_k$ is the scaling factor, and $M$ is the masking matrix:

$$Attention(Q, K, V) = \text{softmax}(M + \frac{QK^T}{\sqrt{d_k}})V$$ \hspace{1cm} (4)

In Eq. (4), the division by $\sqrt{d_k}$ stabilizes the gradients during training. After that, $M \in \mathbb{R}^{k \times k}$ is used to prevent attending subsequent positions, which ensures that the prediction for
FIGURE 4. The multi-head attention mechanism, specifically, the raw input \( x_i \) is first embedded using embedding functions, then, \( x_i \) is multiplied with weight matrices \( W_Q^i, W_K^i, W_V^i \) respectively, and the queries, keys and values are linearly projected \( i \) times with different, learned linear projections to \( d_q, d_k \) and \( d_v \) dimensions respectively. The attention function is then performed in parallel on each of these projected versions of queries, keys and values, yielding \( i \) output values. These values are concatenated and projected again, resulting in the final values.

Position \( i \) depends only on the known outputs at positions less than \( i \). This mechanism can help tackle the overfitting problem. Moreover, the masking operation is performed in the softmax function by adding \(-\infty\). After that, each row of the matrix is then normalized into a probability distribution using the softmax activation function. Finally, a new input representation is constructed via the dot product of the normalized matrix and \( V \). The performance of the self-attention layer can be further improved by the multi-head mechanism, illustrated in Fig. 4. With multi-head attention, each attention head independently maintains its own \( Q/K/V \) weight matrix. The calculation process is similar to the single head attention, shown in Eqs (5) and (6). Note that \( \text{head}_i \) denotes the \( i \)th attention head of the self-attention sublayer. Moreover, the dimensions of the parameter matrix are \( W_Q^i \in \mathbb{R}^{d_{\text{model}} \times d_k} \), \( W_K^i \in \mathbb{R}^{d_{\text{model}} \times d_k} \), \( W_V^i \in \mathbb{R}^{d_{\text{model}} \times d_v} \), \( W_O^i \in \mathbb{R}^{d_{\text{model}} \times h d_v} \), respectively. The results obtained by each head are concatenated together to construct the final result. This mechanism enables the model to focus on different positions and provides multiple representation sub-spaces for the attention layer.

\[
\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W_O \\
\text{head}_i = \text{Attention}(Q W_Q^i, K W_K^i, V W_V^i) 
\]

For the purpose of stable training and faster convergence, layer normalization is applied on each sample \( x \in \mathbb{R}^d \) as Eq. (7):

\[
\text{LN}(x) = \frac{x - \mu}{\delta} \cdot \alpha + \beta
\]

where \( \mu \in \mathbb{R}, \delta \in \mathbb{R} \) are the mean and standard deviation of the input feature respectively, \( \cdot \) is the element-wise dot operator, \( \alpha \in \mathbb{R}^d, \beta \in \mathbb{R}^d \) are trainable affine transform parameters.

We apply Stochastic Gradient Descent (SGD) algorithm to train the model and we choose the cross-entropy as the loss function.

IV. EXPERIMENT SETUP

In this section, we first introduce the hardware and software configurations that we utilize to conduct our experiments then illustrate the dataset used in this work as well as the data preprocessing methods. At last, we explain the evaluation metrics used to show the RTIDS’s performance.

A. EXPERIMENT CONFIGURATION

Our experiments are executed on a computer with 64-bit Intel Core i7 2.70 GHz processor, Nvidia GeForce RTX 2080 graphics card, 32 GB RAM, running Windows 10 Operating System. The proposed model is implemented by Pytorch 1.9.0. The version of Python we choose to execute the program is 3.7.0 and the main software packages we use include NumPy, Keras, Pandas, Sklearn and Matplotlib. We train our model with 25 training epochs, a
0.5 dropout rate, a learning rate of $5e - 4$, and a batch size of 128.

B. DESCRIPTIONS OF CICIDS2017 DATASET

The intrusion detection evaluation dataset CICIDS2017 has been widely used by researchers to analyze and develop new models and algorithms since it was first introduced by the Canadian Institute for Cybersecurity (CIC) [51]. Compared with the NSL-KDD dataset, the CICIDS2017 dataset is up-to-date and offers a broader protocol and attack pool. The traffic records with 79 distinct features are divided into 15 traffic types: which are Benign, FTP-Patator, SSH-Patator, DoS Slowhttptest, DoS Hulk, DoS GoldenEye, DoS slowloris, Heartbleed, Web Attack-Brute Force, Web Attack-XSS, Web Attack-SQL Injection, Infiltration, Botnet, PortScan and DDoS [52]. The dataset spans over eight different files containing five days normal and attacks traffic data of Canadian Institute of Cybersecurity (CIC). A brief description of those data files is shown in Table 1. We summarise the CICIDS2017 dataset in terms of attack types in Table 2. It can be seen that the dataset is highly imbalanced, resulting in a significant challenge in anomaly detection. For example, the total number of “Benign” traffic type in the dataset is 2,273,097 which occupies a very high proportion (80.30%), while the total number of traffic labelled “Heartbleed” is only 11 (<0.01%).

C. DESCRIPTIONS OF CIC-DDoS 2019 DATASET

The CIC-DDoS 2019 dataset is the latest designed dataset which was shared by the Canadian Institute for Cybersecurity (CIC), it was prepared in a proper test context and includes the result of real network traffic analyses [53]. The CIC-DDoS2019 dataset includes 30,480,823 records, including 30,423,960 DDoS attacks records and 56,863 benign records. Furthermore, the DDoS attacks are divided into 11 subtypes. Each record is described by 86 features. The statistic information and attack types of the dataset is summarized in Table 3.

The dataset is implemented with two networks, namely Attack-Network and Victim-Network. The Victim-Network is a highly security infrastructure with firewall, router, switches, and several common operating systems along with an agent that provides the benign behaviors on each PC. The Attack-Network is a completely separated third party infrastructure that executes different types of DDoS attacks.

D. DATA PREPROCESSING

Since both CICIDS2017 and CIC-DDoS2019 datasets have high class imbalanced rate (CIR), we employ Synthetic Minority Oversampling Technique (SMOTE) to increase the quantity of minority class samples by generating samples that does not exist in the original dataset. With this arrangement, it can avoid overfitting problem when constructing classification model.

Unlike other machine learning models, our method aim to retain as many features as possible for the purpose of accuracy improvement, the self-attention mechanism in our model can select features automatically.

We transform all the symbolic features contained in the datasets into numerical values. After converting, the dataset is normalized into the range of [0-1] by using the Min-Max normalization technique which is given by Eq. (8).

$$\bar{x} = \frac{x - \min_{x_{\text{train}}}}{\max_{x_{\text{train}}} - \min_{x_{\text{train}}}}$$

Note that $\max_{x_{\text{train}}}$ and $\min_{x_{\text{train}}}$ refer to the maximum value and minimum value of a certain feature X in the training set respectively. We then split the entire dataset into the training set (70%), the validation set (15%) and the testing set (15%).

E. EVALUATION METRICS

In addition to accuracy, we also use precision, recall, and F1-score, which are widely used for anomaly detection tasks in order to evaluate model performance. These metrics can be expressed by Eqs. (9), (10), (11) and (12) respectively:

Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{9} \quad \text{Precision} = \frac{TP}{TP + FP} \tag{10} \quad \text{Recall} = \frac{TP}{TP + FN} \tag{11} \quad F1 - \text{Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{12}

True positive (TP) refers to abnormal network traffics that are correctly detected; false positive (FP) corresponds to normal traffics that are incorrectly classified as abnormal ones; true negative (TN) represents normal traffics correctly classified as normal, and false negative (FN) is abnormal traffics incorrectly classified as normal ones. The details of TP, FP, TN, and FN can be seen in the confusion matrix in Table 4. In the experiments, we use the training set to learn the general pattern of the traffic sequence and then predict the abnormal traffic in the test set to achieve the purpose of anomaly detection.

V. EXPERIMENT EVALUATION AND ANALYSIS

Our evaluation baselines for network traffic classification employ a classical machine learning algorithm, support vector machines (SVM), and deep learning algorithms that include recurrent neural network (RNN), fuzzy neural network (FNN), and long short-term memory (LSTM) [54], [55].

A. EVALUATION RESULTS OF CICIDS2017 AND CIC-DDoS2019 DATASETS

Table 5 shows the performance of the RTIDS model on CICIDS2017 dataset by using accuracy, precision, recall and F1-score. We find that the proposed method achieves the highest detection accuracy (99.98%) for traffic type “Benign,” while having a lowest performance on detecting “SQL Injection” (50.36%). Due to the “SQL Injection"
TABLE 1. Description of CICIDS2017 dataset files.

| Name of Files                      | Class Found                          |
|------------------------------------|--------------------------------------|
| Monday-Hours_pcap_ISCX.csv         | Benign                               |
| Tuesday-Hours_pcap_ISCX.csv        | Benign, PTP-Patator, SSH-Patator     |
| Wednesday_pcap_ISCX.csv            | Benign, DoS GoldenEye, DoS Hulk, DoS Lowhttp, DoS slowloris, Heartbleed |
| Thursday-WebAttacks_pcap_ISCX.csv  | Benign, Brute Force, SQL Injection, XSS |
| Thursday-Infiltration_pcap_ISCX.csv| Benign, Infiltration                 |
| Friday_pcap_ISCX.csv               | Benign, Bot                          |
| Friday-PortScan_pcap_ISCX.csv      | Benign, PortScan                     |
| Friday-DDoS_pcap_ISCX.csv          | Benign, DDoS                         |

TABLE 2. Summary of the CICIDS-2017 dataset.

| Class Type          | Flow Count | Ratio  |
|---------------------|------------|--------|
| Benign              | 2,273,097  | 80.30% |
| DoS Hulk            | 231,073    | 8.16%  |
| PortScan            | 158,930    | 5.61%  |
| DDoS                | 128,027    | 4.52%  |
| DoS GoldenEye       | 10,293     | 0.36%  |
| PTP-Patator         | 7,938      | 0.28%  |
| SSH-Patator         | 5,897      | 0.21%  |
| DoS slowloris       | 5,796      | 0.20%  |
| DoS Lowhttp         | 5,499      | 0.19%  |
| Bot                 | 1,966      | 0.07%  |
| Web Attack-Brute Force | 1,507   | 0.05%  |
| Web Attack-XSS      | 652        | 0.02%  |
| Infiltration        | 36         | 0.01%  |
| Web Attack-SQL Injection | 21     | < 0.01%|
| Heartbleed          | 11         | < 0.01%|
| Total               | 2,830,743  | 100.00%|

TABLE 3. Statistic information of CIC-DDoS2019 dataset.

| Class Type          | Flow Count | Ratio  |
|---------------------|------------|--------|
| Benign              | 56,563     | 0.02%  |
| DDoS_DNS            | 5,071,011  | 16.64% |
| DDoS_LDAP           | 2,179,930  | 7.15%  |
| DDoS_MSSQL          | 4,522,492  | 14.84% |
| DDoS_neBIOS         | 4,093,279  | 13.43% |
| DDoS_NTP            | 1,202,642  | 3.95%  |
| DDoS_SNMQ           | 5,159,879  | 16.93% |
| DDoS_SSDP           | 2,610,613  | 8.56%  |
| DDoS_TTP            | 2,082,580  | 6.83%  |
| DDoS_UDP            | 3,134,645  | 10.28% |
| DDoS_UDP-Lag        | 366,461    | 1.20%  |
| DDoS_webDDoS        | 439        | < 0.01%|
| Total               | 30,480,823 | 100.00%|

TABLE 4. Confusion matrix of evaluation metrics.

| Type                  | Predicted Positive | Predicted Negative |
|-----------------------|--------------------|--------------------|
| Attack (Positive)     | Predicted Attack as attack (TP) | Predicted Normal as Normal (TN) |
| Normal (Negative)     | Predicted Normal as attack (FP) | Predicted normal as Normal (FN) |

sample is exceptionally scarce in the entire dataset, leading to undesirable performance for our model. The behavior pattern of traffic type “Bot” is analogous to normal network traffic, which increases the difficulty for our model to correctly identify “Bot” attacks, resulting in poor performance.

The performance of our proposed RTIDS model on CIC-DDoS2019 dataset is shown in Table 6. It can be seen that RTIDS model achieves highest detection rate on traffic type “DDoS_NTP” with accuracy of 99.65%, and the detection rate for class “DDoS_webDDoS” can also reach 89.77%. Thus, it proves that the proposed method in this research effectively improves the issue of low detection rate due to data scariness.

B. COMPARISON ANALYSIS WITH BASELINE MODELS ON CICIDS2017 DATASET

In this section, we compare the results of our model with those of the classic machine learning model SVM and other deep learning models. The overall classification results of all models is summarized in Table 7. In terms of classification accuracy, the proposed method is this paper is 0.89%, 1.33%, 2.14% and 1.0% higher than SVM-IDS, RNN-IDS, LSTM-IDS and FNN-IDS respectively. From the
The initial evaluation focuses on the detection performance of traffic with the label “Benign.” Fig. 5 shows that our model outperforms all other detection algorithms with an accuracy of 99.65%. The intrusion detection accuracy of the proposed model on traffic labeled “Bot” is 97.70%, which is a significant improvement over other models (see Fig. 6). Since the malicious traffic labeled with “FTP-Patator” and “SSH-Patator” are similar in their attack behavior and characteristics, we combine and label them “Patator Attacks,”
resulting in 13,835 records with this new label. The detection performance of this type of traffic is shown in Fig.7. Compared with other models, our designed model with an accuracy of 97.56% is better at detecting “Patator Attacks.” The traffic labeled “Web Attack” includes three sub-types: “Web Attack-Brute Force,” “Web Attack-XSS” and “Web Attack-SQL Injection.” Fig.8 shows the detection performance of different intrusion detection models for this type of traffic. While the accuracy of our proposed model is higher than other classification models, its performance metrics such as precision and recall are slightly below those of these baseline models. The reason is that it is not easy to choose a proper threshold value that provides high detection accuracy rate or false alarm rate for this traffic type in our model. We merge the traffic data labeled “DoS GoldenEye,” “DoS Hulk,” “DoS Slowhttptest,” “DoS slowloris” and “Heartbleed” into a new dataset labeled “DoS,” resulting in a total of 252,627 records. Fig.9 shows the overall detection performance of this type of data. As shown, the proposed model substantially improves the accuracy, precision, recall, and F1-score of identifying “DoS” traffic data. Due to the differences in traffic characteristics and behavior patterns between DDoS attacks and DoS attacks, for example, DoS attacks usually occur between single machines, whereas DDoS attacks typically involve a large number of computers and multiple networks. Thus, we exclusively evaluate traffic data labelled “DDoS” in the dataset. Fig.10 depicts the detection results, and we find that our model still outperforms others with an accuracy of 99.90%. Finally, we evaluate the classification performance of different models using traffic data labelled as “PortScan” and “Infiltration.” Fig.11 and Fig.12 show the performance results. It can be observed that our proposed model performs better than other models on both detection tasks.

C. COMPARISON ANALYSIS WITH BASELINE MODELS ON CIC-DDoS2019 DATASET

In order to further validate the effectiveness of the proposed method in this paper, we also perform experiments on CIC-DDoS2019 dataset. The total experiment results of all classifiers are summarized in Table 8. As the table shows, the classification accuracy of our proposed method has improved by 4.56%, 1.67%, 0.81% and 3.03% compared with SVM-IDS, RNN-IDS, LSTM-IDS and FNN-IDS. However, the training time of the proposed model is still not the lowest, but since the model needs to be trained only once and can be used for off-line intrusion detection in the network, the time performance receivable.

Fig.13 gives the multi-class classification accuracy of various intrusion detection models on CIC-DDoS2019 dataset. Both classical machine learning and deep learning classifiers have achieved less ideal performance on class type “Web-DDoS” and “SSDP” in comparison with other class types. During experiments, we find that the features have similar characteristics between the “Web-DDoS” and “SSDP.” This means that the classifiers require additional features to classify the traffic record “Web-DDoS” and “SSDP” correctly. In spite of this fact, the detection accuracy of “Web-DDoS” and “SSDP” of the proposed method reaches 89.77% and 91.41% respectively. Since our model incorporates as many
features as possible during training, thus, the experiments testify the advantage of the proposed framework.

VI. CONCLUSION AND FUTURE WORK
In this article, we propose a robust Transformer-based intrusion detection system, called RTIDS, for detecting abnormal activities and traffic violations in networks. RTIDS provides an all-in-one intrusion detection solution composed of three modules: a data preparation module, a RTIDS model-construction module, and a real-time intrusion detection module. The framework employs transformer model for feature extraction and selection. In addition, to prevent our neural network detection model from overfitting, we design a variant transformer with one additional masked multi-head self-attention sublayer in the decoder stack. We also employ the Synthetic Minority Over-sampling Technique (SMOTE) for oversampling minority class samples to combat the class imbalance issue. Furthermore, we evaluate the performance of RTIDS on the CICIDS2017 and CIC-DDoS2019 datasets. The accuracy of the RTIDS algorithm we proposed is 98.45%, and the precision is 98.32%, the Recall is 98.73%, the F1-score is 98.02% for CICIDS2017 dataset and the accuracy is 98.58%, the precision is 98.82%, the Recall is 98.66%, the F1-score is 98.45% for CIC-DDoS2019 dataset respectively. Experimental results demonstrate that RTIDS performs better than the mainstream classical and deep learning intrusion detection algorithms used in other IDSs. In terms of training time, the time performance of our proposed method is acceptable as an off-line intrusion detection tool in the network.

Our future work will focus on how to increase the speed of the transformer algorithm for the quick-response intrusion detection system in order to significantly reduce the damage caused by anomalous events. Furthermore, in the next step of our work, we will consider employ meta-learning method to tackle the few shot classification problem.

REFERENCES
[1] S. Caton and R. Landman, “Internet safety, online radicalisation and young people with learning disabilities,” Brit. J. Learn. Disabilities, vol. 50, no. 1, pp. 88–97, Mar. 2022.
[2] J. Lewis, “Economic impact of cybercrime, no slowing down,” Center Strategic Int. Stud., McAfee, San Jose, CA, USA, 2018, vol. 13, p. 2019.
[3] P. Devan and N. Khare, “An efficient XGBoost–DNN-based classification model for network intrusion detection system,” Neural Comput. Appl., vol. 32, no. 16, pp. 12499–12514, 2020.
[4] H. Wang, Z. Cao, and B. Hong, “A network intrusion detection system based on convolutional neural network,” J. Intell. Fuzzy Syst., vol. 38, no. 6, pp. 7623–7637, Jun. 2020.
[5] S. Al and M. Dener, “STL-HDL: A new hybrid network intrusion detection system for imbalanced dataset on big data environment,” Comput. Secur., vol. 110, Nov. 2021, Art. no. 102435.
[6] Z. Wu, J. Wang, L. Hu, Z. Zhang, and H. Wu, “A network intrusion detection method based on semantic re-encoding and deep learning,” J. Netw. Comput. Appl., vol. 164, Aug. 2020, Art. no. 102688.
[51] Z. Tian, C. Luo, J. Qiu, X. Du, and M. Guizani, “A distributed deep learning system for web attack detection on edge devices,” IEEE Trans. Ind. Informat., vol. 16, no. 3, pp. 1963–1971, Mar. 2020.

[52] R. Panigrahi and S. Borah, “A detailed analysis of CICIDS2017 dataset for designing intrusion detection systems,” Int. J. Eng. Technol., vol. 7, no. 3, pp. 49–482, Dec. 2018.

[53] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, “Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy,” in Proc. Int. Carnahan Conf. Secur. Technol. (IC CST), Oct. 2019, pp. 1–8.

[54] E. Anthi, L. Williams, M. Slowinska, G. Theodorakopoulos, and P. Burnap, “A supervised intrusion detection system for smart home IoT devices,” IEEE Internet Things J., vol. 6, no. 5, pp. 9042–9053, Oct. 2019.

[55] Y. Lin, J. Wang, Y. Tu, L. Chen, and Z. Dou, “Time-related network intrusion detection model: A deep learning method,” in Proc. IEEE Global Commun. Conf. (GLOBECOM), Dec. 2019, pp. 1–6.

ZI HAN WU received the M.S. degree in applied psychology from the University of Electronic Science and Technology of China, in 2016. He is currently pursuing the M.S. degree in cyber space security with the School of Cyber Security and Computer, Hebei University. His current research interests include network security, intrusion detection, and machine learning.

HONG ZHANG received the Ph.D. degree in computer science from the University of Central Florida, in 2018. He is currently an Assistant Professor with the School of Cyber Security and Computer, Hebei University. His research interests include the design and analysis of parallel systems for big-data computing, which includes two aspects, such as design and analysis. For design, he is currently working on optimizing performance, scalability, resilience, load balancing of data-intensive computing, and distributed machine learning. For the aspect of analysis, he focuses on using program analysis to detect programming errors and performance defects in large-scale parallel computing systems.

PENG HAI WANG is currently pursuing the master’s degree with the School of Cyber Security and Computer, Hebei University, Baoding, China. His current research interests include cloud computing, smart communication, machine learning, and the Internet of Things.

ZHIBO SUN received the Ph.D. degree from Arizona State University. He is currently a Cyber Security Researcher. His research interests include human-centric security and threat intelligence analytics.

**VOLUME 10, 2022**