2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

EN-BERT: A Transformer-based Model for Encrypted Attack Traffic Detection with Pre-training and Fine-tuning Phases

Xiaoying Huang[1], Yanping Xu[1], Yanbo Fang[1], Yuxin Shen[1], Yongxing Xu[1]
[1] Hangzhou Dianzi University, Hangzhou 310018, China
xiaoying963@outlook.com

Abstract. Encrypted attacks represented by ransomware and APT are becoming increasingly complex, posing a huge threat to cyberspace. As a result, encrypted attack traffic detection is imperative. Traditional encrypted attack traffic detection methods face challenges which include feature extraction limitations, dataset imbalance, and poor generalization capabilities. To address these issues, this paper proposes EN-Bert, a Transformer-based model with both pre-training and fine-tuning phases. In the pre-training stage, the encrypted traffic dataset is first processed using Token serialization for traffic shunting, segmentation, and feature extraction. Then the model is pre-trained on two tasks: Masked Flow Model (MFM) and Same-origin Flow Prediction (SFP) to uncover the contextual relationships between traffic flows. In the fine-tuning stage, this paper addresses the imbalance issues through data enhancement. From the model's perspective, weighted cross-entropy loss and K-L divergence are designed in this phase to optimize the model's performance and enhance its generalization ability. In the experimental section, with the CIC-IDS-2017 dataset and a self-collected encrypted DOS attack traffic dataset, comparative and ablation experiments demonstrate that EN-Bert model proficiently addresses challenges related to dataset imbalance and poor model generalization, proving to be an effective and reliable approach for encrypted attack traffic detection.

Keywords: encrypted attack traffic detection, EN-Bert, pre-training phase, data enhancement, weighted cross-entropy, K-L divergence.

1 Introduction

With the rapid development of Internet technology, network communication security has become a major concern for numerous users. The phenomenon of covert attacks conducted through various encrypted channels is becoming increasingly severe, with encrypted attacks frequently targeting individual users, enterprises, and even industrial control systems (ICS). According to the Microsoft Digital Defense Report 2024 [1], an average of 600 million cyberattacks are launched against Windows customers daily, with DDoS attacks peaking at 4,500 occurrences per day in June 2024. As encrypted attack traffic poses growing challenges to traditional security defenses, research on its detection has become an urgent necessity.

Before the widespread use of encrypted communication, Deep Packet Inspection (DPI) [2] was once the primary approach for analyzing packet content. However, with the extensive deployment of encryption protocols such as SSL and TLS, DPI has suffered a significant decline in detection capability. Meanwhile, encrypted attack traffic exhibits new characteristics, including difficult data parsing, complex traffic patterns, and highly covert attack behaviors, making traditional rule-based detection methods ineffective. In recent years, the application of deep learning and self-supervised learning in encrypted traffic detection has gained increasing attention. For example, the introduction of Transformer architectures has enhanced contextual understanding in traffic classification [3], while Bert-based traffic detection models have demonstrated the effectiveness of Natural Language Processing (NLP) techniques in encrypted traffic analysis [4].

Against this backdrop, we propose a model named EN-Bert tailored for encrypted attack traffic detection. By integrating a self-supervised pre-training phase with a supervised fine-tuning phase, it enhances the model's capability to detect unknown attacks. Instead of extracting traditional metadata, the model utilizes Token serialization to process encrypted traffic and employs a Bigram tokenization approach to enhance the understanding of encrypted payloads. Additionally, techniques such as data enhancement, weighted cross-entropy, and Kullback-Leibler (KL) divergence loss are introduced to address the dataset imbalance problem and optimize loss computation, thereby mitigating accuracy degradation and enhancing the model's generalization ability.

The main contributions of this paper are summarized as follows: (1) We propose an integrated framework combining pre-training and fine-tuning, where the model learns generic traffic representations in the pre-training phase and achieves accurate encrypted attack traffic detection for downstream task in the fine-tuning phase. (2) Encrypted traffic token serialization is used to process features, replacing the method of extracting statistical features from network metadata. It enhances model's understanding of encrypted payloads. (3) We introduce an innovative data enhancement module that effectively tackles the challenge of dataset imbalance through flow concatenation. (4) In the fine-tuning phase, we design weighted cross-entropy and K-L divergence loss functions to mitigate the performance difference caused by overfitting. It helps the model maintain accurate prediction distribution and decisively addresses issues related to poor generalization ability. (5) EN-Bert model demonstrates strong overall performance, achieving accuracy improvements of 30.2%, 0.5%, and 1.7% compared to other intelligent detection models (RNN, MLP, and KNN) on our self-collected encrypted DOS attack traffic dataset.

2 EN-Bert Overall Structure

The encrypted attack traffic detection framework based on EN-Bert model consists of two stages. The first stage is the pre-training phase, which includes traffic Token serialization and model pre-training. This phase aims to learn general traffic representations from large-scale encrypted traffic while capturing contextual relationships between

different data flows and identifying fundamental attack patterns. The second stage is the fine-tuning phase, which involves data enhancement and model fine-tuning. By refining the pre-trained model, this phase transfers traffic representation, association, and recognition capabilities to downstream detection task, enabling accurate detection of encrypted traffic. The overall structure is shown in Fig. 1.

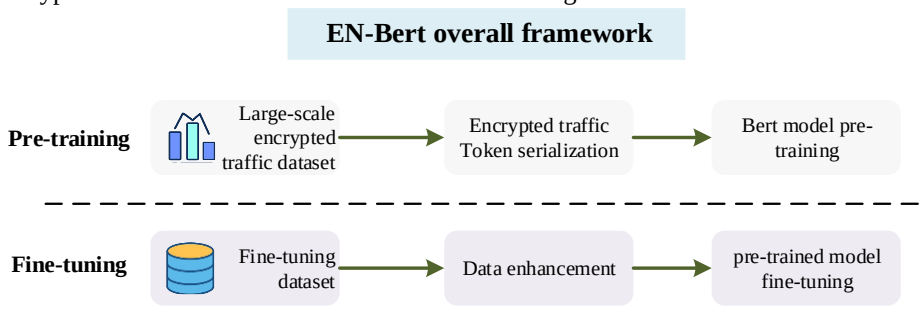

Fig. 1. EN-Bert overall framework

2.1 Pre-training phase

The first component of EN-Bert overall structure is the pre-training phase, which strengthens the model's ability to understand encrypted traffic and differentiate traffic patterns linked to various attack types. In this phase, large-scale encrypted traffic data are transformed through tokenization, converted into vectors and subsequently fed into the Bert model. The model is then pre-trained on two tasks: Masked Flow Model (MFM) and Same-Origin Flow Prediction (SFP). By learning from extensive encrypted attack traffic, the model captures contextual relationships between different data flows, laying a foundation for precise detection in the fine-tuning phase.

Encrypted traffic Token serialization. Since the Bert model cannot directly process network data in PCAP format, a tokenization approach is used to segment encrypted traffic. It converts encrypted traffic into vectors and divides them into sequences. The framework is illustrated in Fig. 2.

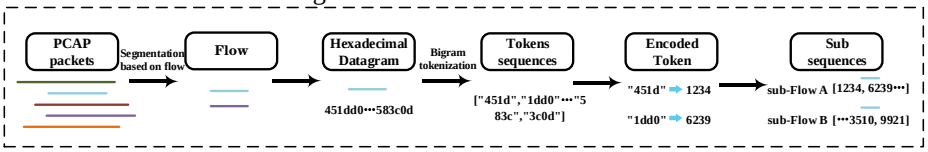

Fig. 2. Encrypted traffic Token serialization

Firstly, load the flow and return the list of packets it contains. Each packet is then converted to hexadecimal format. Every four hexadecimal digits are grouped together to form a Token. Then Special Tokens are inserted into the sequence. The [CLS] tokens are placed at the beginning of each flow. If the concatenated Tokens are shorter than the required minimum length, the [PAD] tokens will be added at the end. As for the [SEP] tokens, they are used to separate two subsequences.

As the Bert model processes only numeric data and cannot directly interpret text or byte streams, so converting each character Token into a numeric representation is necessary. Each Token is encoded, forming a sequence of numeric Tokens with values

ranging from 0 to 65535, based on the vocabulary size (|V| = 65536). For example, "451d" is encoded as 1234. Finally based on the [SEP] token, every sequence gonna be split into two subsequences, sub-Flow A and sub-Flow B, in preparation for the Same-Origin Flow Prediction task in the pre-training phase. The module's final output is numeric sequences of Tokens, such as [1234, 6239, 9012, 1001].

Model pre-training. Different types of encrypted traffic exhibit distinct characteristics. Exploring the contextual relationships between Tokens within each flow, as well as the interactions between different flows, helps identify distinct attack types. Inspired by natural language features, the Bert model adopts Masked Language Model (MLM) and Next Sentence Prediction (NSP) as pretraining tasks [5]. In the context of this paper, considering the flow features of Tokens sequences, the pretraining tasks are adjusted to Masked Flow Model (MFM) and Same-origin Flow Prediction (SFP), which can enhance the model's understanding of various Tokens sequences. For example, in the case of a DDoS encrypted attack, the traffic typically presents a sharp increase within a short period, displaying a "continuous" fixed pattern. At this time the Masked Flow Model can help identify logical relationships between flows, while the Same-origin Flow Prediction model strengthens the understanding of the attack's internal logic. The overall process is illustrated in Fig. 3.

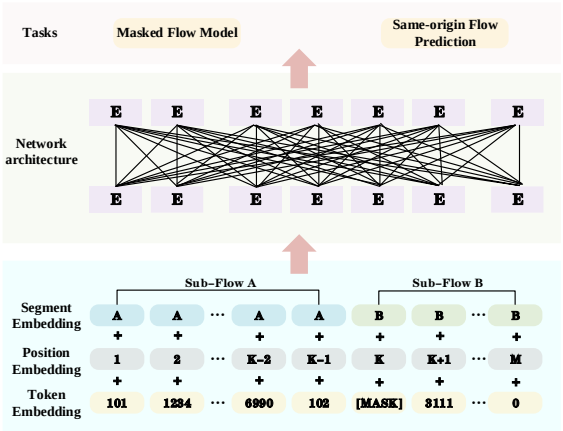

Fig. 3. The pre-training phase of the model

Token vectorization. Before model pre-training, all Tokens sequences need to be converted into fixed-dimensional vectors. This process involves three levels of embedding: Token Embedding, Position Embedding, and Segment Embedding.

Token Embedding utilizes both word embedding and character embedding to convert each Token into a 768-dimensional vector representation. This forms the foundation of EN-Bert model pretraining, enabling the model to better understand each fundamental unit. Position Embedding provides the model with the position information of each Token in the flow, resolving the limitation of the transformer architecture, which does not naturally account for the order of sequences. By assigning a position-related vector to each Token, Position Embedding resolves this problem. Segment Embedding is primarily designed for the Same-origin Flow Prediction task. Each Token is assigned a different embedding vector depending on the subsequence it belongs to.

However, based on the binary classification of subsequences, the vector values can only exist in two possible forms. Each embedding level generates a 768-dimensional fixed-length vector for the Token. The final model input is the sum of these vectors.

These three Token embedding levels collectively process fundamental Token data, Token position information and sub-sequence classification, providing strong support for subsequent model training.

Masked Flow Model pre-training. Before the Masked Flow Model task, Token sequences have been processed by selecting 15% of all Tokens as masking targets. Among them, 80% are replaced with [MASK], 10% are substituted with random Tokens and the remaining 10% keep unchanged. During the pre-training process, the model predicts the masked positions and determines which word in the vocabulary has been replaced by [MASK]. This enhances model's understanding of encrypted features within each flow. The loss function for this task is defined as follows:

$$L_{\mathrm{MFM}} = -\sum_{i=1}^{n} \log\left(\mathrm{P}\left(MASK_i = token_i \mid X;\theta\right)\right) \tag{1}$$

where θ represents the parameters of EN-Bert model; X denotes the input Token sequence after three levels of embedding; $token_i$ is the true value at the i_{th} masked position; $MASK_i$ is the predicted value at the i_{th} masked position.

Same-origin Flow Prediction pre-training. In the Same-origin Flow Prediction task, the model treats each flow as a combination of two subsequences, sub-Flow A and sub-Flow B. To help the model better learn relationships between subsequences, the dataset is prepared before training, consisting of two categories with an equal 50% distribution. Category 1 represents cases where sub-Flow A and sub-Flow B originate from the same flow, labeled as SameOrigin. Category 2 represents cases where sub-Flow A and sub-Flow B come from different flows, labeled as NotSameOrigin. During pre-training process, the model is trained and evaluated on this dataset to determine whether the subsequences belong to the same flow. This enhances the model's understanding of internal data relationships within flows. The loss function for this task is given below:

$$L_{\mathrm{SFP}} = -\sum_{j=1}^{k} \log\left(\mathrm{P}\left(SAME_j = origin_j \mid Y;\theta\right)\right) \tag{2}$$

where θ represents the parameters of EN-Bert model; Y denotes the dataset sampled before training; $origin_j$ refers to the true label of the j_{th} test sample; $SAME_j$ represents the predicted label of the j_{th} test sample.

Since the Masked Flow Model task involves a larger number of predictions, it generates a higher loss. On the other hand, the Same-Origin Flow Prediction task is a binary classification problem with fewer predictions, resulting in a lower loss. So different weights must be assigned to the losses of these two tasks. In this paper, the loss of the MFM task is scaled down by a factor of 10 before being summed with the loss of the SFP task to compute the final loss. This adjustment prevents poor training results caused by loss imbalance. The overall loss function is defined as follows:

$$Loss = L_{MFM} + L_{SFP} \tag{3}$$

where L_{MFM} represents the loss of the Masked Flow Model pre-training task; L_{SFP} represents the loss of the Same-origin Flow Prediction pre-training task.

Once the loss function is calculated, the model undergoes backpropagation. During this process, the model computes the gradients for each layer's parameters based on the loss. After the gradients are calculated, the optimizer updates the parameters according to the predefined learning rate. Steps outlined above make up a single training iteration. After each round of training, the model evaluates its performance by calculating accuracy on the validation dataset. This helps assess model's effectiveness.

2.2 Fine-tuning phase

The second phase is the fine-tuning stage, aimed at achieving precise detection of encrypted attack traffic. Built upon the pre-training phase, it refines both the input dataset and the training process through data enhancement and loss optimization, enhancing the model's performance on the target task.

Data enhancement. Within the proposed framework, the processed Token sequences extracted from collected PCAP packets serve as the model's input. Analysis of these sequences reveals a significant class imbalance in the dataset, resulting from differences in attack characteristics and traffic data collection. This imbalance affects the model's accuracy in recognizing both minority-class samples and overall samples while also limiting its generalization ability beyond the training set [6].

Considering the unique characteristics of traffic data, we propose a data enhancement method based on data concatenation, specifically designed for processing input Token sequences. For minority-class data, half of an adjacent data flow is concatenated to form a new data flow, which is then merged with the original dataset to create a balanced dataset, as illustrated in Fig. 4. This approach maintains the semantic relationships between adjacent Tokens and reduces the bias introduced by traditional slicing. It also helps prevent potential issues in later tokenization.

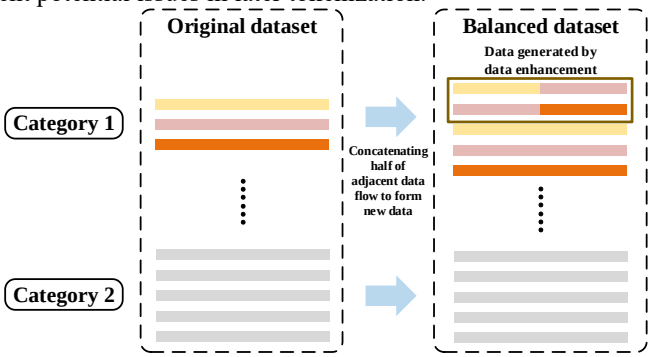

Fig. 4. Data enhancement module

After this section, the class distribution is essentially balanced and the dataset is reorganized into two lists: labels and data, enabling easier processing in the next stages. The dataset is then split into three subsets following an 8:1:1 ratio: fine-tuning training

set, test set and validation set. The fine-tuning training set is used to train the model by learning the mapping between input data (text_a) and output labels (label). The test set assesses the final model performance including its generalization ability, while the validation set is utilized for hyperparameter tuning and performance evaluation.

Model fine-tuning. The model structure in the fine-tuning phase is essentially the same as in the pre-training phase, with the main difference being the change in the task objectives between two stages. The pre-training phase focuses on the Masked Flow Model and Same-origin Flow Prediction tasks, whereas the fine-tuning phase targets precise detection of encrypted attack traffic with model results applied in practical scenarios. In this module, the model must not only achieve the downstream task but also maximize detection accuracy and enhance the model's generalization ability. However, issues such as dataset imbalance and poor generalization often arise in practice. We overcome these issues by proposing a method that combines weighted cross-entropy with K-L divergence to replace the negative log-likelihood loss, guiding the model's fine-tuning process to overcome dataset imbalance and improve generalization.

Weighted cross-entropy. It's an improvement over the traditional cross-entropy loss function, adjusting the importance of different classes by assigning weights to each class. When using the weighted cross-entropy loss function, a weight can be assigned to each class, typically represented as a 1D tensor. During the calculation of the loss function, the loss for each sample is adjusted according to the weight of its respective class, thus achieving weighted handling of different classes. Aurelio et al. [7] applied a similar approach when addressing the concern of imbalanced datasets in neural networks. The specific formula for calculating the weighted cross-entropy loss in this study is as follows:

$$L_{\mathrm{WCE}} = -\frac{1}{N}\sum_{i=1}^{N} w_i \cdot \log\left(p_i\right) \tag{4}$$

where $\log\left(p_i\right)$ represents the initial loss of each sample; w_i is the weight assigned to the category of each sample; N denotes the total number of samples.

We use weighted cross-entropy to replace the negative log-likelihood loss and combines soft labels for loss calculation to alleviate the majority-class overfitting problem caused by dataset imbalance. The weighted method allows the model to prioritize features from minority-class data during training, reducing overfitting on majority classes. It also helps avoid large accuracy discrepancies between classes and improves overall model performance. Huang et al. [8] also applied the concept of weighted cross-entropy in practical applications, specially in diabetic retinopathy detection.

K-L divergence. K-L divergence is commonly used to measure the distance between two probability distribution functions. In the EN-Bert model training, K-L divergence is applied to quantify the relationship between the model's soft target distribution (the probability distribution of target classes) and the model's predicted probabilities distribution. By detecting the change in K-L divergence, the model's predicted distribution Q(X) is guided to approach the tangible distribution P(X), thereby improving the accuracy of malicious attack detection. Kim et al. [9] explored the differences between K-L divergence and MSE loss in knowledge distillation and sequential distillation, showing that K-L divergence contributes to improving model performance in both areas. Specifically in sequential distillation, using K-L divergence with a smaller temperature

parameter significantly helps reduce the impact of label noise, enhancing the model's generalization ability. Therefore, we incorporate K-L divergence into the loss calculation during the fine-tuning, guiding the model to align its predictions with actual distribution, thus improving accuracy.

This module combines the weighted cross-entropy and the K-L divergence as the model loss. By calculating gradients through backpropagation, the model's parameters are updated in each iteration, gradually reducing the value of the loss function. The total loss calculation formula is presented below:

$$Loss = L_{WCE} + KL \tag{5}$$

where L_{WCE} represents the model loss processed by weighted cross-entropy module; KL denotes the K-L divergence.

The model saved after fine-tuning can be used for detecting encrypted attack traffic in real-world scenarios. Additionally, it is tested by an unlabeled test dataset. Analyzing the prediction results from the unlabeled data using the fine-tuned model allows the model to perform effectively in actual use cases.

3 Experiments

To evaluate the performance of EN-Bert in encrypted attack traffic detection, two experimental datasets are selected. One is the CIC-IDS-2017 public dataset, and the other is a self-collected encrypted DOS attack traffic dataset. For the first part, the performance of the EN-Bert model is compared with other intelligent detection models to demonstrate its overall capability. Then ablation experiments are conducted on the three innovative modules in the EN-Bert model architecture: data enhancement, weighted cross-entropy and K-L divergence to assess the modular performance. For the last section, the role of each module is summarized and compared.

3.1 Dataset

The first experimental dataset consists of four types of encrypted attack traffic generated through the use of SlowHTTPTest, which performs slow HTTP Denial-Of-Service (DoS) attacks. These attacks exploit slow data transmission or request handling to consume excessive server resources, leading to service denial or performance degradation. This dataset includes four attack types: Slow Read, Slow Body, Slow Header and Slowloris. The specific distribution is presented in Table 1.

Table 1. Encrypted DOS Attack Traffic Dataset

Encrypted DOS Attack Traffic Dataset	#Flow	#Packet
Slow Read	2233	44642
Slow Body	1216	24318
Slow Header	1140	22795
Slowloris	1544	30862
All	6133	122617

The second experimental dataset was released by Canadian Institute for Cybersecurity (CIC). It was collected over five days, from July 3 to July 7, 2017, and includes both normal and various attack traffic. The dataset consists of 16 categories: one for normal traffic and 15 for attack traffic. The data is stored in PCAP format, with the specific distribution presented in Table 2.

Table 2. CIC-IDS-2017 Dataset

CIC-IDS-2017 Dataset	#Flow	#Packet
Dos_Slowhttptest	83	4150
DDos_LOIT	764	38200
Heartbleed_Port_444	521	26050
Infiltration-Cool_disk-MAC	520	26000
Web_Attack-Sql-Injection	1298	64900
FTP_Patator	2921	146050
Benigh	6805	340250
Dos_slowloris	492	24600
SSH_Patator	974	48700
Web_Attack-Brute_Force	1577	78850
Botnet_ARES	3206	160300
Dos_GoldenEye	2838	141900
Web_Attack-XSS	582	29100
Dos_Hulk	1114	55700
Infiltration-Dropbox_download_Win_Vista	288	14400
Infiltration-Dropbox_download_Meta_exploit_Win_Vista	462	23100
All	24445	1222250

3.2 Experimental Evaluation Methods

Ji, Lee and others [10] summarized commonly used performance evaluation metrics for AI-based encrypted traffic anomaly detection techniques in their review. Among these metrics, Recall and F1-score are the most frequently used. To assess the model performance more accurately, we choose three evaluation metrics—Precision, Recall and F1-score—to evaluate each category. Additionally, Accuracy is used to assess the overall model performance.

The formula for calculating Precision is as follows:

$$Precision = \frac{TP}{TP + FP} \tag{6}$$

The formula for calculating Recall is as follows:

$$Recall = \frac{TP}{TP + FN} \tag{7}$$

The formula for calculating F1-score is as follows:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{8}$$

The formula for calculating Accuracy is as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

where *TP* (True Positive) represents the number of correctly predicted positive samples; *TN* (True Negative) represents the number of correctly predicted negative samples; *FP* (False Positive) represents the number of negative samples incorrectly predicted as positive; *FN* (False Negative) represents the number of positive samples incorrectly predicted as negative.

3.3 Experimental Results

Comparison of EN-Bert with other intelligent detection models. This module evaluates the overall performance of EN-Bert model by comparing its accuracy with other models under the same target task. At present, artificial intelligence widely employs various machine learning and deep learning models. Among machine learning algorithms, Random Forest [11] and XGBoost [12] are the most commonly used. In deep learning algorithms, CNN [13] (such as 1-D CNN and 2-D CNN) and other algorithms are frequently utilized. In this section, three representative artificial intelligence models—RNN, MLP, and KNN—are selected for experimentation. According to the experimental results, Fig. 5 shows that EN-Bert achieves over 99% accuracy for each category in the encrypted DoS attack traffic dataset, with an accuracy difference of no more than 0.5% between categories. The overall accuracy reaches 99.35%, slightly outperforming MLP and KNN while showing a significant improvement over RNN. These results demonstrate the model's strong detection capability and generalization in encrypted attack traffic detection.

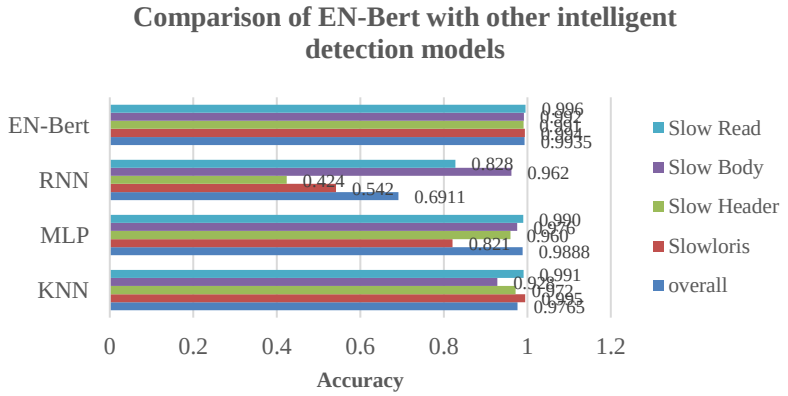

Fig. 5. Comparison of EN-Bert with other intelligent detection models

Comparison of data enhancement effect. To evaluate the effect of data enhancement module on model performance, this section examines EN-Bert with data enhancement applied and compares it to EN-Bert-D, which excludes this module. The accuracy

differences between these two models across both datasets are presented in Table 3 and Table 4.

For the encrypted DOS attack traffic dataset, a comparison between columns 2 and 5 in Table 3 shows that the precision of the Slow Read, Slow Body and Slow Header categories exceeds that of EN-Bert-D by an average of 2%. This result indicates that the data enhancement module enriches the distribution of training samples, enabling the model to better learn distinct category features and thereby improving accuracy in identifying positive samples. The recall improvement is reflected in columns 3 and 6 of Table 3, where EN-Bert outperforms EN-Bert-D across all categories.

Table 3. Comparison of data enhancement effect (Encrypted DOS Attack Traffic Dataset)

Encrypted DOS Attack Traffic Dataset	EN-Bert-D			EN-Bert		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Slow Read	0.991	0.996	0.993	**1.000**	**1.000**	**1.000**
Slow Body	0.968	0.984	0.976	**1.000**	**0.990**	**0.995**
Slow Header	0.982	0.965	0.973	**1.000**	**0.995**	**0.997**
Slowloris	**0.993**	0.987	0.990	0.985	**1.000**	**0.993**
Overall	Accuracy：0.9853			**Accuracy：0.9962**		

For the CIC-IDS-2017 dataset, Table 4 demonstrates that compared to EN-Bert-D, EN-Bert improves precision for 75% of categories, with an average increase of approximately 22%. Recall improves significantly in 87.5% of the categories, confirming enhanced accuracy and recognition capability for positive samples. In terms of F1-score, the Dos_Slowhttptest and Heartbleed_Port_444 categories show improvements of over 65%, further validating the substantial performance advances achieved through data enhancement. As indicated in the last row of Table 4, the overall accuracy of EN-Bert reaches 72.42%, reflecting an 11% increase compared to EN-Bert-D. This result strongly supports the module's effectiveness in improving both generalization and classification performance.

Table 4. Comparison of data enhancement effect (CIC-IDS-2017 Dataset)

CIC-IDS-2017 Dataset	EN-Bert-D			EN-Bert		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Dos_Slowhttptest	0.000	0.000	0.000	**0.741**	**0.628**	**0.680**
DDos_LOIT	0.529	0.351	0.422	**0.904**	**0.868**	**0.886**
Heartbleed_Port_444	0.562	0.173	0.265	**0.996**	**1.000**	**0.998**
Infiltration-Cool_disk-MAC	**0.905**	0.365	0.521	0.673	**0.544**	**0.602**
Web_Attack-Sql-Injection	0.573	0.515	0.543	**0.801**	**0.740**	**0.769**
FTP_Patator	0.447	0.548	0.492	**0.817**	**0.624**	**0.707**
Benigh	0.667	0.449	**0.537**	0.462	**0.624**	0.531
Dos_slowloris	0.653	**0.828**	**0.730**	**0.763**	0.668	0.712
SSH_Patator	0.531	0.351	0.422	**0.649**	**0.688**	**0.668**
Web_Attack-Brute_Force	0.643	0.573	0.606	**0.673**	**0.756**	**0.712**
Botnet_ARES	0.734	**0.706**	**0.720**	**0.740**	0.684	0.711
Dos_GoldenEye	0.477	0.472	0.474	**0.684**	**0.812**	**0.742**
Web_Attack-XSS	**0.805**	0.569	**0.667**	0.529	**0.648**	0.583
Dos_Hulk	0.759	0.536	0.628	**0.902**	**0.956**	**0.928**
Infiltration-Dropbox_download_Win_Vista	**0.778**	0.483	0.596	0.614	**0.636**	**0.625**
InfiltrationDropbox_download_Meta_eploit_Win_Vista	0.741	0.435	0.548	**0.840**	**0.712**	**0.771**
Overall	Accuracy：0.6136			Accuracy：**0.7242**		

Comparison of weighted cross-entropy effect. This section compares the performance of EN-Bert with weighted cross-entropy loss to that of EN-Bert-W, which excludes this module. The accuracy differences between these two models across two datasets are shown in Table 5 and Table 6.

In the encrypted DOS attack traffic dataset, as seen in columns 2 and 5 of Table 5, the precision of low Body and Slow Header categories shows a significant increase, while the precision for the Slowloris category remains unchanged. This indicates that the weighted cross-entropy module effectively adjusts the class weights, improving the model's ability to learn from minority-class data and leading to more accurate detection of positive classes. However, the effect of the weighted cross-entropy module on majority-class categories is less pronounced. Since the module is mainly designed to optimize for class imbalance, the improvements in recall and F1-score are limited for categories with relatively balanced distributions. By observing columns 3 and 6 of Table 5, it is evident that EN-Bert outperforms EN-Bert-W in recall values for the Slow Body and Slow Header categories. The overall F1-score distribution is consistent with recall improvements, as shown in the final columns of both models in the table.

Table 5. Comparison of weighted cross-entropy effect (Encrypted DOS Attack Traffic Dataset)

Encrypted DOS Attack Traffic Dataset	EN-Bert-W			EN-Bert		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Slow Read	**0.991**	0.996	**0.993**	0.978	0.996	0.987
Slow Body	0.968	0.984	0.976	**1.000**	**0.992**	**0.996**
Slow Header	0.982	0.965	0.973	**1.000**	**0.991**	**0.996**
Slowloris	0.993	**0.987**	**0.990**	0.993	0.981	0.987
Overall	Accuracy：0.9853			**Accuracy：0.9902**		

Table 6. Comparison of weighted cross-entropy effect (CIC-IDS-2017 Dataset)

CIC-IDS-2017 Dataset	EN-Bert-W			EN-Bert		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Dos_Slowhttptest	0.000	0.000	0.000	**0.111**	**0.111**	**0.111**
DDos_LOIT	0.529	0.351	0.422	**0.851**	**0.818**	**0.834**
Heartbleed_Port_444	**0.562**	0.173	0.265	0.389	**0.404**	**0.396**
Infiltration-Cool_disk-MAC	**0.905**	0.365	0.521	0.667	**0.615**	**0.640**
Web_Attack-Sql-Injection	0.573	0.515	0.543	**0.639**	**0.531**	**0.580**
FTP_Patator	0.447	0.548	0.492	**0.728**	**0.623**	**0.672**
Benigh	0.667	0.449	0.537	**0.699**	**0.757**	**0.727**
Dos_slowloris	**0.653**	**0.828**	**0.730**	0.385	0.204	0.267
SSH_Patator	**0.531**	0.351	0.422	0.522	**0.612**	**0.563**
Web_Attack-Brute_Force	**0.643**	0.573	0.606	0.605	**0.643**	**0.623**
Botnet_ARES	**0.734**	**0.706**	**0.720**	0.707	0.656	0.681
Dos_GoldenEye	0.477	0.472	0.474	**0.686**	**0.806**	**0.741**
Web_Attack-XSS	**0.805**	0.569	**0.667**	0.417	0.431	0.424
Dos_Hulk	0.759	0.536	0.628	**0.907**	**0.883**	**0.895**
Infiltration-Dropbox_download_Win_Vista	**0.778**	0.483	**0.596**	0.682	**0.517**	0.588
InfiltrationDropbox_download_Meta_eploit_Win_Vista	**0.741**	0.435	0.548	0.714	**0.543**	**0.617**
Overall	Accuracy：0.6136			Accuracy：**0.6776**		

The model performance comparison on the CIC-IDS-2017 dataset is shown in Table 6. 37.5% of the categories in EN-Bert show higher accuracy than the corresponding categories in the EN-Bert-W model, with an average improvement of around 19%. This indicates that EN-Bert has optimized classification accuracy for certain categories.

Regarding recall, except for the categories Dos_slowloris, Botnet_ARES, and Web_Attack-XSS, recall rates for the remaining categories have increased. In terms of overall accuracy, EN-Bert achieves a 6.4% improvement. Compared to the previous dataset, the increase in overall accuracy is more significant, possibly due to the more severe class imbalance in the CIC-IDS-2017 dataset. The weighted cross-entropy module has had a more pronounced effect on improving the model's performance on this dataset.

Comparison of K-L divergence loss effect. To evaluate the improvement brought by the inclusion of K-L divergence, this section compares the performance of EN-Bert with K-L divergence to that of EN-Bert-K, which excludes this module. The accuracy differences between EN-Bert and EN-Bert-K on two datasets are shown in Table 7 and Table 8.

In the encrypted DOS attack traffic dataset, as shown in columns 2 and 5 of Table 7, the accuracy of all categories detected by EN-Bert model is higher than that of EN-Bert-K, with each category achieving an accuracy rate of over 99%. This demonstrates the model's high accuracy in identifying positive classes. As shown in columns 3-4 and 5-6 of Table 7, EN-Bert outperforms EN-Bert-K in both recall and F1-score, indicating improvements in the model's ability to identify positive classes as well as its overall precision and completeness. In terms of overall accuracy, the last row of the table shows that EN-Bert achieves an accuracy rate of 99.67%, which is 1.14% higher than EN-Bert-K.

Table 7. Comparison of K-L divergence loss effect (Encrypted DOS Attack Traffic Dataset)

Encrypted DOS Attack Traffic Dataset	EN-Bert-K			EN-Bert		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Slow Read	0.991	0.996	0.993	**1.000**	0.996	**0.998**
Slow Body	0.968	0.984	0.976	**0.992**	**1.000**	**0.996**
Slow Header	0.982	0.965	0.973	**1.000**	**0.991**	**0.996**
Slowloris	0.993	0.987	0.990	**0.994**	1.000	**0.997**
Overall	Accuracy：0.9853			**Accuracy：0.9967**		

Table 8. Comparison of K-L divergence loss effect (CIC-IDS-2017 Dataset)

CIC-IDS-2017 Dataset	EN-Bert-K			EN-Bert		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Dos_Slowhttptest	0.000	0.000	0.000	**0.250**	**0.111**	**0.154**
DDos_LOIT	0.529	0.351	0.422	**0.903**	**0.844**	**0.872**
Heartbleed_Port_444	**0.562**	0.173	0.265	0.415	**0.327**	**0.366**
Infiltration-Cool_disk-MAC	**0.905**	0.365	0.521	0.861	**0.596**	**0.705**
Web_Attack-Sql-Injection	0.573	0.515	0.543	**0.773**	**0.577**	**0.661**
FTP_Patator	0.447	0.548	0.492	**0.724**	**0.630**	**0.674**
Benigh	**0.667**	0.449	0.537	0.629	**0.797**	**0.703**
Dos_slowloris	**0.653**	**0.828**	**0.730**	0.571	0.163	0.254
SSH_Patator	0.531	0.351	0.422	**0.772**	**0.622**	**0.689**
Web_Attack-Brute_Force	**0.643**	0.573	0.606	0.624	**0.688**	**0.655**
Botnet_ARES	0.734	**0.706**	**0.720**	**0.755**	0.653	0.700
Dos_GoldenEye	0.477	0.472	0.474	**0.689**	**0.782**	**0.733**
Web_Attack-XSS	**0.805**	**0.569**	**0.667**	0.451	0.397	0.422
Dos_Hulk	0.759	0.536	0.628	**0.970**	**0.883**	**0.925**
Infiltration-Dropbox_download_Win_Vista	**0.778**	0.483	**0.596**	0.586	**0.586**	0.586
InfiltrationDropbox_download_Meta_eploit_Win_Vista	0.741	0.435	0.548	**0.750**	**0.522**	**0.615**
Overall	Accuracy：0.6136			Accuracy：**0.6894**		

In the CIC-IDS-2017 dataset, as shown in Table 8, 43.75% of the categories have higher precision than EN-Bert-K, while 37.5% of the categories show similar precision to EN-Bert-K. With respect to recall, the DDos_LOIT category shows a significant improvement, while the Dos_slowloris category experiences a decline. The F1-score follows a similar trend as the recall rates. As for overall accuracy, EN-Bert model achieves 68.94%, a 7.4% improvement over EN-Bert-K. The experimental results effectively demonstrate that K-L divergence enhances the model's generalization ability, allowing further optimization of detection performance on complex datasets.

Comparison of the performance analysis of each module. The effectiveness of different modules in improving model performance varies. In the encrypted DOS attack traffic dataset, the K-L divergence module shows the most significant improvement, making detection more stable across all four categories and resulting in notable performance enhancement. In contrast, in the CIC-IDS-2017 dataset, data enhancement contributes the most to performance improvement due to the more severe imbalance of dataset. By increasing the number of minority-class samples, the data enhancement module significantly relieves the low accuracy issue caused by imbalance, refining the model's performance. In summary, each module plays a distinct role in enhancing the EN-Bert model. They contribute to the overall success of EN-Bert in encrypted attack traffic detection.

4 Conclusion

This paper reviews and analyzes existing academic literature on related topics, summarizing the key challenges and difficulties in encrypted attack traffic detection. We propose a new model architecture and processing flow, offering novel methods to address tough issues. In particular, data enhancement addresses the problem of imbalanced dataset, boosting the model's performance. The use of weighted cross-entropy and K-L divergence plays a significant role in optimizing the loss function during the fine-tuning phase, helping the model overcome performance issues and improve generalization. Through extensive experimental validation, EN-Bert model has shown promising results in encrypted attack traffic detection. Despite the model's achieving progress in detection performance, there are still areas for further optimization in response to the evolving trends in the current era:

(1) Recently, Deepseek has gained tremendous popularity worldwide due to its superior performance and domestically designed architecture. The computational and storage requirements of large AI models have also sparked significant research and discussion. Although EN-Bert pre-trained model obtained in this paper is not as large as models like Deepseek or ChatGPT, the model deployment challenges arising from the huge number of parameters also pose difficulties for developers. If the model can be lightweighted, both the overall computation speed and storage requirements can be optimized.

(2) For encrypted attack traffic, traditional feature extraction methods are not applicable due to the high likelihood of being unable to decrypt the traffic. In addition to the encrypted token serialization method used in this paper, Poh G. S. and others [14] proposed techniques that support regular expression matching and complex rule

detection. Combining deep learning and reinforcement learning can also provide detection capabilities equivalent to those of plaintext traffic analysis.

Acknowledgments. This study was funded by the Key Program of National Natural Science Foundation of China (grant number 62232005), the Key Research and Development Project of Zhejiang Province (grant number 2022C01200) and the Special Fund of the Basic Scientific Research Business expenses of the universities of Zhejiang Province (grant number GK249909299001-).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Microsoft: Microsoft Digital Defense Report 2024, https://www.microsoft.com/en-us/security/security-insider/intelligence-reports/microsoft-digital-defense-report-2024, last accessed 2025/02/23
2. Antonello, R., Fernandes, S., Kamienski, C., et al.: Deep packet inspection tools and techniques in commodity platforms: Challenges and trends. J. Netw. Comput. Appl. 35(6), 1863–1878 (2012)
3. Han, K., Wang, Y.H., Chen, H.T., et al.: A survey on vision transformer. IEEE Trans. Pattern Anal. Mach. Intell. 45(1), 87–110 (2022)
4. Yang, H., He, Q., Liu, Z.Y., et al.: Malicious encryption traffic detection based on NLP. Secur. Commun. Netw. 2021(1), 9960822 (2021)
5. Liu, Y.H., Ott, M., Goyal, N., et al.: RoBERTa: A robustly optimized BERT pretraining approach. arXiv abs/1907.11692 (2019)
6. He, H.B., Garcia, E.A.: Learning from imbalanced data. IEEE Trans. Knowl. Data Eng. 21(9), 1263–1284 (2009)
7. Aurelio, Y.S., de Almeida, G.M., de Castro, C.L., et al.: Learning from imbalanced data sets with weighted cross-entropy function. Neural Process. Lett. 50, 1937–1949 (2019)
8. Huang, J.T., Wu, X.H., Qi, H.S., et al.: Diabetic retinopathy detection with weighted cross-entropy loss. In: Proc. 42nd Chinese Control Conf. (CCC), pp. 8814–8819. IEEE, China (2023)
9. Kim, T.Y., Oh, J.H., Kim, N.Y., et al.: Comparing Kullback-Leibler divergence and mean squared error loss in knowledge distillation. arXiv abs/2105.08919 (2021)
10. Ji, I.H., Lee, J.H., Kang, M.J., et al.: Artificial intelligence-based anomaly detection technology over encrypted traffic: a systematic literature review. Sensors 24(3), 898 (2024)
11. Speiser, J.L., Miller, M.E., Tooze, J., et al.: A comparison of random forest variable selection methods for classification prediction modeling. Expert Syst. Appl. 134, 93–101 (2019)
12. Chen, T.Q., Guestrin, C.: XGBoost: A scalable tree boosting system. In: Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., pp. 785–794 (2016)
13. Bhatt, D., Patel, C., Talsania, H., et al.: CNN variants for computer vision: history, architecture, application, challenges and future scope. Electronics 10(20), 2470 (2021)
14. Poh, G.S., Divakaran, D.M., Lim, H.W., et al.: A survey of privacy-preserving techniques for encrypted traffic inspection over network middleboxes. arXiv abs/2101.04338 (2021)