symmetry

MDPI

Article

The Attention-Based Autoencoder for Network Traffic Classification with Interpretable Feature Representation

Jun Cui [1], Longkun Bai [2], Xiaofeng Zhang [2], Zhigui Lin [2] and Qi Liu [3,*]

[1] School of Life Sciences, Tiangong University, Tianjin 300380, China
[2] School of Electronics and Information Engineering, Tiangong University, Tianjin 300380, China
[3] School of Computer and Information Engineering, Tianjin Chengjian University, Tianjin 300380, China
* Correspondence: liuqicj@126.com

Abstract: Network traffic classification is crucial for identifying network applications and defending against network threats. Traditional traffic classification approaches struggle to extract structural features and suffer from poor interpretability of feature representations. The high symmetry between network traffic classification and its interpretable feature representation is vital for network traffic analysis. To address these issues, this paper proposes a traffic classification and feature representation model named the attention mechanism autoencoder (AMAE). The AMAE model extracts the global spatial structural features of network traffic through attention mechanisms and employs an autoencoder to extract local structural features and perform dimensionality reduction. This process maps different network traffic features into one-dimensional coordinate systems in the form of spectra, termed FlowSpectrum. The spectra of different network traffic represent different intervals in the coordinate system. This paper tests the interpretability and classification performance of network traffic features of the AMAE model using the ISCX-VPN2016 dataset. Experimental results demonstrate that by analyzing the overall distribution of attention weights and local weight values of network traffic, the model effectively explains the differences in the spectral representation intervals of different types of network traffic. Furthermore, our approach achieves the highest classification accuracy of up to 100% for non-VPN-encrypted traffic and 99.69% for VPN-encrypted traffic, surpassing existing traffic classification schemes.

Keywords: traffic classification; feature representation; attention mechanism; autoencoder; interpretability

Citation: Cui, J.; Bai, L.; Zhang, X.; Lin, Z.; Liu, Q. The Attention-Based Autoencoder for Network Traffic Classification with Interpretable Feature Representation. *Symmetry* **2024**, *16*, 589. https://doi.org/10.3390/sym16050589

Academic Editor: Tomohiro Inagaki

Received: 5 April 2024
Revised: 25 April 2024
Accepted: 6 May 2024
Published: 10 May 2024

1. Introduction

With the widespread popularity of the internet and the continuous development of network technologies, network traffic has experienced explosive growth. This growth poses challenges to various aspects, such as network security, bandwidth management, and network performance optimization. To address these challenges, network traffic classification has become an important research area. Network traffic classification involves building algorithmic models to extract features from network traffic and associate them with specific classes based on different requirements, such as quality of service (QoS), routing improvement, and billing systems [1]. By classifying network traffic, we can better understand the operation of networks, detect abnormal traffic in real time, and enhance network security and performance.

In the early stages of network traffic classification, researchers primarily classified network flows by inspecting port numbers in data packets [2]. However, with the randomization of network port numbers [3], this approach gradually became less effective. Subsequently, payload-based [4] classification techniques began to be widely applied in traffic classification, which involves categorizing network flows by analyzing payload information in data packets. However, the proliferation of encrypted traffic has imposed

significant limitations on this method [5]. To address encrypted traffic and enhance classification accuracy, researchers have turned to statistical-based methods [6], extracting a series of statistical features (such as packet count, average transmission rate, etc.) from data packets and utilizing machine learning algorithms (such as KNN [7], decision trees [8], and support vector machines (SVM) [9]) to classify these features. However, the selection of statistical features often relies on researchers' experience, resulting not only in time consumption but also in potential failure to comprehensively reflect all characteristics of network flows, leading to low classification accuracy.

The remarkable achievements of deep learning (DL) in fields such as computer vision [10,11] and natural language processing [12,13] have prompted network researchers to apply this technology to network traffic classification [14]. Compared to traditional machine learning-based methods, deep learning can automatically and effectively learn and extract complex features from raw data, reducing computational complexity and avoiding manual feature engineering. Currently, a variety of DL models have been applied in network traffic classification, including the convolutional neural network (CNN) [15], recurrent neural network (RNN) [16], autoencoder [17], and attention mechanism (AM) [18], which are widely used to handle high-dimensional and encrypted traffic payload classification tasks. However, technological advancements inevitably bring new challenges. DL models lack interpretability in their output results, and researchers often focus solely on the output without considering changes in network traffic characteristics after passing through the model, thus lacking representation of network traffic features.

Recently, the concept of FlowSpectrum theory [19] has been proposed, providing new insights into network traffic analysis. FlowSpectrum is a new approach for characterizing network traffic features, with its core aim being to construct a representation space for network behavior to depict or describe the behavior of network space flows, thereby assisting researchers and users in the process of traffic analysis [20,21]. FlowSpectrum theory primarily focuses on extracting, decomposing, and dimensionally mapping features from complex and sparse network space traffic, mapping low-dimensional features of network traffic onto a coordinate system, thereby characterizing and analyzing network flow behavior features. By describing network traffic features in low dimensions, researchers can intuitively analyze the behavior characteristics of network flows and compare differences between different types of network traffic. Meanwhile, researchers can utilize differences in FlowSpectrum to classify network traffic.

This paper proposes an interpretable FlowSpectrum model, the AMAE model, for generating and classifying encrypted FlowSpectrum. Overall, the attention-based autoencoder not only extracts network flow structural features more comprehensively but also effectively explains the representation results of the FlowSpectrum model through attention values. Specifically, the AMAE model first utilizes an attention module to perform global feature learning on the grayscale images of packet bytes, generating a weight matrix composed of pixel weights. These weights provide effective explanations for the differences between different types of network flow spectra. Furthermore, by using attention weight values to weigh the original input data, a new feature matrix is obtained. This matrix is then fed into an autoencoder for dimensionality reduction, during which the convolutional layers of the autoencoder further capture local structural features. Finally, based on the FlowSpectrum, we classify network flows using a Bayesian optimal classifier [22].

For this purpose, the key technical challenges addressed in this chapter are as follows:

- Extracting global structural features: To preserve the original spatial structure features of network flows and address the difficulty in extracting global spatial structures, we utilize the spatial attention mechanism. By dynamically learning features at each position in the grayscale image, relevant attention weights are obtained, reflecting the importance of spatial information of network flow bytes at specified positions. This approach captures the global structural features of the entire grayscale image. Subsequently, these attention weights are applied to the original network flow matrix to enhance relevant useful features and diminish irrelevant ones.

- Resolving the issue of generating different FlowSpectrum intervals by the model: We introduce a self-attention mechanism and autoencoder model structure. A single-channel attention module is added to the model to simultaneously generate channel attention weights and spatial attention weights, thereby producing blended attention weights for application in the self-attention mechanism.
- The primary contributions of this chapter are as follows: A proposal of a "global + local" structural feature capture scheme to comprehensively extract features from encrypted flows. Initially, we capture global spatial structural features of original bytes through the design of spatial domain attention and single-channel attention mechanisms. Data weighed by global structural features undergo dimensionality reduction via an autoencoder module. Simultaneously, CNN convolutional layers are integrated within the autoencoder to capture local structural features.
- Proposing the AMAE model for generating interpretable representations of encrypted flow features: By leveraging the self-attention mechanism, we blend spatial attention weights and single-channel attention weights onto the original byte feature matrix, followed by dimensional analysis. Consequently, the desired encrypted FlowSpectrum and its attention weights are derived. Through the FlowSpectrum, distinct ranges of encrypted flows within the coordinate system can be observed clearly. Analysis of attention weight values elucidates specific reasons for the varying intervals of flow spectrum lines generated by the model.
- Utilizing encrypted FlowSpectrum for encrypted flow classification, enhancing the accuracy of encrypted flow classification: This section employs the ISCX-VPN2016 dataset. Initially, the AMAE model is trained using the training set to obtain the FlowSpectrum, followed by the classification of the test set. Experimental validation demonstrates the excellent performance of the AMAE model in encrypted traffic classification. Furthermore, it achieves symmetry between the differences in FlowSpectrum intervals and attention weight values.

The remaining sections of this paper are organized as follows. In Section 2, we present several existing common technical approaches to flow classification and discuss the research status related to FlowSpectrum. In Section 3, we introduce the data types of this paper, the framework of the FlowSpectrum model, and the mechanism of flow classification using FlowSpectrum. In Section 4, we present multiple experiments and discuss and compare the experimental results. In Section 5, we summarize the work of this article and elaborate on future research content.

2. Related Work

As mentioned in the previous section, traffic classification has become an important research area in the field of cyberspace security. With the exponential growth of network traffic and the increasing proportion of encrypted traffic [23,24], traditional traffic classification schemes based on ports and payloads are no longer sufficient to meet current demands. The development of machine learning and deep learning technologies has greatly improved the accuracy and efficiency of network traffic classification. Therefore, in recent years, researchers have begun to focus on studying traffic classification techniques related to machine learning and deep learning. Meanwhile, FlowSpectrum theory, as a new approach to network flow analysis, aims to address the weak feature representation and interpretability issues of machine learning and deep learning. This section will provide a detailed overview of the current research status of machine learning-based, deep learning-based, and flow spectrum-related traffic classification.

2.1. Machine Learning-Based Methods

Machine learning-based methods for network traffic classification are highly diverse and have garnered significant attention in the initial applications of flow classification [25]. These algorithms primarily rely on learning statistical features of network flows and may also utilize dimensionality reduction techniques (such as PCA or LDA) to classify mul-

tidimensional features of network flows after dimensionality reduction. Dusi et al. [26] proposed a method combining the Gaussian mixture model (GMM) with support vector machines (SVMs) for SSH-encrypted traffic classification, which effectively categorizes SSH-encrypted sessions. Lashkari et al. [27] introduced a scheme utilizing time statistical information to analyze and detect Tor network flows. The approach employs C4.5 and k-nearest neighbors (KNNs) as classifiers, achieving a classification accuracy of 70% according to experimental results. Additionally, Gil et al. [28] achieved an 88% classification accuracy by utilizing decision trees to classify statistical information extracted from the ISCX-VPN2016 dataset, considering only time features such as duration, forward and backward arrivals, and traffic arrivals. Zong et al. [29] proposed a PCA-based 3D network data visualization method to facilitate the understanding of geometric relationships between various categories of network traffic. Imran et al. [30] employed the linear discriminant analysis (LDA) algorithm and genetic algorithm (GA) for feature transformation and optimal subset selection, respectively. Santos et al. [31] introduced a method utilizing time series analysis and computational intelligence techniques (including kurtosis, DFA, and SOM-based clustering algorithm) to characterize computer network traffic.

While machine learning is effective in addressing network traffic classification problems, it comes with challenges such as the need for manual feature selection, high computational complexity, and significant resource consumption. Moreover, machine learning algorithms have narrow applicability, requiring changes to algorithms or feature selection methods for different traffic scenarios, making rapid migration difficult.

2.2. Deep Learning-Based Methods

Unlike ML-based methods, DL-based methods typically take raw traffic samples as input, effectively improving the efficiency of network traffic classification. Wang et al. [32] proposed an end-to-end encrypted traffic classification method using one-dimensional convolutional neural networks (1D-CNNs). This is the first application of an end-to-end deep learning model in the field of encrypted traffic classification. The team processed raw network flows at both the flow level and session level. Experimental results demonstrate that the end-to-end 1D-CNN model effectively improves the accuracy of encrypted traffic classification. Zeng et al. [33] proposed a technique for classifying network traffic using multiple deep learning models. They developed three deep learning models, including CNN, LSTM, and stacked autoencoders (SAEs), to extract features from different perspectives. Wang et al. [34] proposed a HAST model to enhance feature extraction capabilities. This model uses a "CNN + LSTM" structure, with low-level CNN layers aimed at extracting "spatial" features and high-level LSTM layers aimed at increasing the CNN receptive field. The HAST model obtains "global" and "temporal" features from structural and temporal perspectives. Dai et al. [35] proposed an encrypted traffic classification model (GLADS), which combines an attention mechanism with the CNN network. The team first encoded network flows using a temporal sliding window, then extracted local structural features using a CNN backbone network and simultaneously learned the relationship between input data and global information through an attention matrix. As shown in Figure 1, the team believes that the framework for extracting local and global features can improve classification performance. Experimental results demonstrate that GLADS effectively improves the classification results of the network. Lotfollahi et al. [36] proposed a deep learning-based deep packet parsing classification model. The team used 1D-CNN and SAE networks to extract features and classify traffic in one system. This approach can identify encrypted traffic and distinguish between VPN and non-VPN traffic. Experiments were conducted using the ISCX-VPN2016 dataset, with the results showing an average classification accuracy of 94%. Höchst et al. [37] proposed a method framework that utilized a deep neural network with autoencoders and a private dataset containing seven different traffic types. The average accuracy of this method was 80%. Ferreira et al. [38] obtained meaningful low-dimensional data representations from an attack detection perspective using a semi-supervised autoencoder. Niyaz et al. [39] proposed an efficient and flexible

NIDS based on sparse autoencoders and softmax regression. Xie et al. [40] proposed a self-attention model for traffic classification—SAM. The research team described the attention of the model during the classification process through self-attention mechanisms, effectively explaining the relationship between input and output results by analyzing the attention weights assigned to network flow features used for classification. Meanwhile, experimental classification accuracy on the ISCX dataset can reach 90.3%.

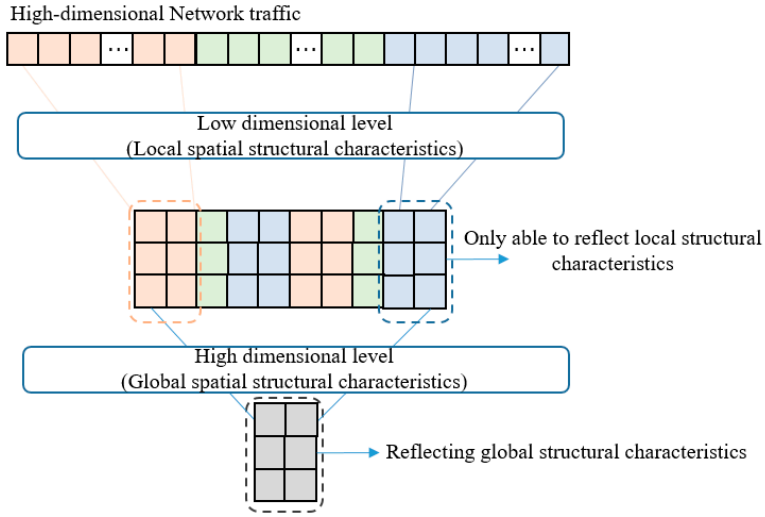


Figure 1. Global and local structural features in high and low dimensions.

We found that deep learning schemes are very effective for network flow classification. However, it has several issues: Firstly, there is currently little attention paid to the representation of network flow features by deep learning models, as well as the interpretability of DL models, including the explanation of the relationship between input and output results and the explanation of the model training process. The second issue is that deep learning models still lack the ability to extract network flow features. Although some have attempted to use CNN + LSTM or AM + CNN schemes to extract structural features of flows, it is difficult to fully analyze network flow features without a good dimensionality reduction scheme when the amount of data is huge.

2.3. FlowSpectrum

Methods based on deep learning represent an end-to-end strategy, where the learning process turns neural networks into black boxes. In other words, neural networks are still primarily viewed as black-box function approximators that map given inputs to classification outputs. Apart from the final output, it is challenging to understand the predictive logic hidden inside the neural network. A key insight of deep learning is interpretability—users should be able to understand the learned outputs. The lack of interpretability raises questions about the reliability of deep learning and may hinder the application of neural networks in production environments. For example, what kind of features does deep learning learn? Where does its discriminative power come from? This black-box approach may lead to skepticism about its reliability [29].

As described in the first section, FlowSpectrum is a new solution for traffic analysis, providing a specific representation method for discernible features in the network space. The core of the FlowSpectrum theory is to simplify the sparse and high-dimensional network traffic features in the network space into a simple domain expression, simplifying the process of network flow analysis, improving the performance of network traffic classification, threat detection, and interception of abnormal traffic, thereby maintaining high QoS and service availability for service providers. In [19], the Yang research team first proposed the FlowSpectrum theory and designed a neural network structure based on a semi-supervised autoencoder for network flow data decomposition and dimensionality

reduction. Specifically, the research team decomposed and reduced the features of the NSL-KDD dataset using a semi-supervised autoencoder model, then mapped them into a one-dimensional standard coordinate system to form feature spectra. Different types of traffic features are reflected in different interval positions of the coordinate system. The team tested the feature representation and intrusion detection capabilities of the FlowSpectrum using the NSL-KDD intrusion detection dataset, preliminarily establishing the correspondence between network behavior and spectral domain, as well as the intrusion detection capabilities. Guo et al. supplemented the FlowSpectrum theory in [20] and proposed a basic method for mapping network flow data from the original flow space to the FlowSpectrum space, including the supplementation of FlowSpectrum mathematical theory and the basic principles of FlowSpectrum model construction. Meanwhile, based on the FlowSpectrum theory, the research team conducted experimental verification using the UNSW-NB15 dataset. For the first time, they mapped the features of nine types of attack traffic in the UNSW-NB15 dataset to a two-dimensional coordinate system and classified the nine types of traffic, achieving a classification accuracy of 67.72%.

In the current classic network flow analysis schemes (including DPI-based, machine learning-based, and deep learning-based methods), there are problems such as high computational complexity, high resource consumption, insufficient extraction of network flow structure features, and lack of ability to represent flow features. The existing flow spectrum models face problems such as poor network flow feature extraction ability, weak generalization ability, low classification accuracy, and the generated flow spectra not having interpretable representations.

To address the shortcomings of different approaches in network flow classification and to improve and expand existing FlowSpectrum technology, this paper proposes an AMAE model for generating flow spectra of encrypted flows and utilizes flow spectra for classifying encrypted flows. The design scheme of this paper has three main advantages: First, to preserve the original network flow and spatial structure features, we cannot use common text processing transform models. To address the difficulty of extracting global spatial structure, we use spatial domain attention methods to learn weights for each pixel, where these weights represent the importance of spatial position information, and then attach this spatial attention matrix to the original network flow matrix to amplify useful correlated features and weaken useless features. Second, to obtain the overall output result weights of the model, we adjust the self-attention mechanism and the structure of the autoencoder model. The combination of our self-attention mechanism with spatial attention and single-channel attention ensures the coherence of the entire model's input and output results. Third, we enhance the interpretability of the FlowSpectrum theory in the process of network flow analysis by analyzing attention scores to explain the spectrum intervals.

3. Methodology

As mentioned earlier, the purpose of the AMAE model is to map the original network flow to a FlowSpectrum and classify encrypted flows. This section first discusses the issue of selecting the input data type for the model in Section 3.1. Then in Section 3.2, we provide the mechanism for generating and classifying flow spectra. In Section 3.3, the overall architecture of the AMAE model and the design details of its different modules are introduced.

3.1. Input Type

Currently, in the research of network flow analysis, the most common representations include packet-level, flow-level, and network session-level-based ones. In our previous work [41], we discussed research on the level of network flow analysis, as this study chooses to focus on the network session-level. As illustrated in Figure 2, the multidimensional form of network data packets and the sequence of data packets are shown. Data packet lengths vary, and when data packet lengths are too long, it is difficult for the model to capture the global characteristics of data packets in the session flow through local features during

the entire training process. This is also why we choose spatial attention to capture the global spatial structural features. Real data packets have features between structures, and we convert the original bytes into grayscale images for subsequent operations. The data packets include the Ethernet layer (such as destination IP, source IP, etc.), the network layer (such as time and protocol, total length, etc.), the transport layer (source and destination port, etc.), and the application layer (such as payload), with close connections between different layers, because capturing structural features is crucial.

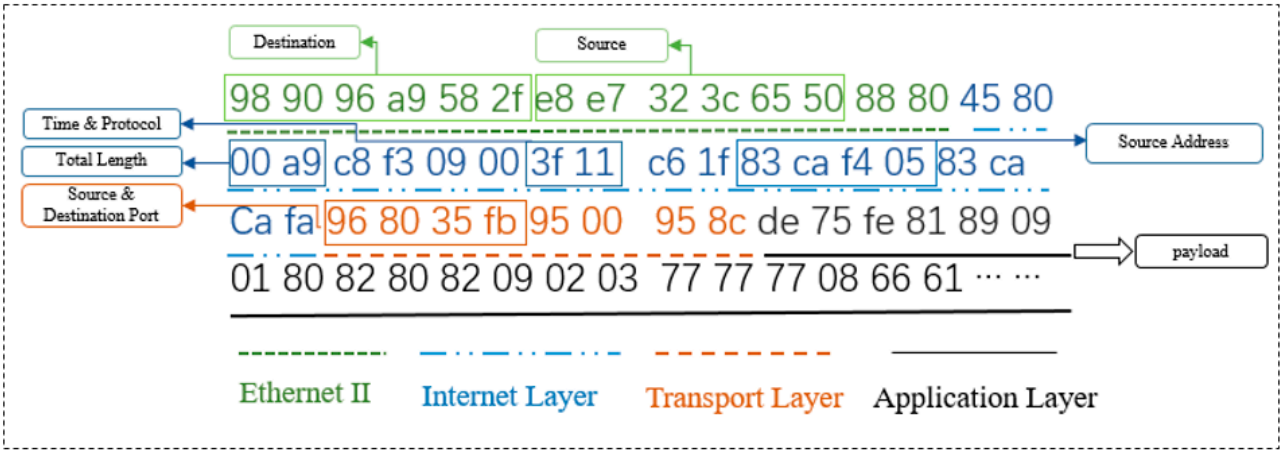


Figure 2. Original network packet structure information.

We adopted the data preprocessing process suggested in [42], extracting the first 784 bytes of each data packet and converting them into images of size 28 × 28 for the final representation. The specific extraction process is divided into three steps:

(1) Packet segmentation.

Packet segmentation involves splitting the original pcap file into standardized discrete session files based on the session level. Firstly, parse and extract the packet information within the pcap file. Then, based on the five-tuple information of the packets (source address, destination address, source port, destination port, and protocol type), identify packets that are interrelated and determine if they belong to the same session. Group packets belonging to the same session and divide them according to sessions, forming different session files. During this process, sessions will merge or split based on traffic type to ensure each session file contains complete and meaningful data. Finally, save the segmented session data as separate session files for subsequent data processing.

(2) Session filtering.

This step consists of two processes: data cleansing and data padding. Firstly, check if there are incomplete or duplicate data packets in the session files. Incomplete data may result from packet loss or corruption during network transmission, while duplicate packets may result from retransmission in network transmission or errors in packet capture devices. Therefore, it is necessary to identify and remove these incomplete or duplicate data packets from each session file to ensure the integrity and accuracy of the packet information. After data cleansing, unify the size of session data to fit the input of the neural network. In this paper, the input size required for the traffic classification model is 28 × 28 = 784. Therefore, when padding sessions, only the first 784 bytes are retained, and if the length is insufficient, fill with 0x00 to 784 bytes.

(3) IDX file generation.

Next, sequentially convert each byte to a pixel size between 0 and 255 to achieve the conversion of traffic data to an image. Finally, save the file in IDX format.

3.2. FlowSpectrum Theory

Representation of FlowSpectrum. Network traffic characteristics in cyberspace are sparse and discrete. We denote the cyberspace as $\mathcal{X}$ and the real number space as $\mathbb{R}$. We define network flow instances within $\mathcal{X}$ as A, thus having $\mathcal{X} = \{A_1, A_2, A_3, \ldots\ldots, A_h\}$. It follows that $A_w \subseteq \mathcal{X} \subseteq \mathbb{R}^m$, where $w \in [1, h]$. Each instance flow consists of multiple data points denoted as $A = (x_1, x_2, \cdots, x_m)$, where x_i represents the value of the instance flow point, and i $\in [1, \mathrm{m}]$. Each data point corresponds to specific features represented by a vector $\overrightarrow{F} = \{f_1, f_2, \cdots, f_m\}$. The output of network flow instances within $\mathcal{X}$ to the output set within $\mathbb{R}$ is denoted as $R = \{r_0, r_1, \ldots, r_K\}$, where r_n represents an output type. We define $\mathcal{D} = \left\{ \left(A_1, \overrightarrow{L}_1 \right), \left(A_2, \overrightarrow{L}_2 \right), \ldots, \left(A_\mathrm{m}, \overrightarrow{L}_\mathrm{m} \right) \right\}$, where $\mathcal{D}$ represents the set of input instances A corresponding to output instance labels $\overrightarrow{L}_\mathrm{w}$, with $\overrightarrow{L}_\mathrm{w} \in R$. We use $P_r(A)$ to denote the probability of A being of output type r, given by the following:

$$P_r(A) = P\{Y = r \parallel X = A\}, \tag{1}$$

where Y and X are random variables from R and $\mathcal{X}$, respectively. Ultimately, we use $FS(\mathcal{X}_r)$ to represent the spectrum of mapping network space flows to real number space ($\mathcal{X} \to \mathbb{R}$), denoted as follows:

$$FS(\mathcal{X}_r) = \{d(A_1) : P_r(A_1), d(A_2) : P_r(A_2), \cdots, d(A_n) : P_r(A_n)\}, \tag{2}$$

where $d(A_1)$ is the value in the real number space $\mathbb{R}$. We call $FS(\mathcal{X}_r)$ the FlowSpectrum.

FlowSpectrum is applied to flow classification. Generating FlowSpectrum and using it for flow classification belongs to the process of dimensionality reduction and the representation of network flows. As previously described, FlowSpectrum generation occurs during the training phase, while flow classification occurs during the classification phase. In the classification phase, we employ the Bayesian optimal classifier. First, we establish a test flow set $\mathcal{X}' = \{A_1', A_2', A_3', \ldots\ldots, A_h'\}$, where the probability of instance A_1' is $P(A_1' \parallel \mathcal{X}')$. The spectrum of this set is set as follows:

$$FS(\mathcal{X}') = \begin{Bmatrix} d(A_1') : P(A_1' \parallel \mathcal{X}') \\ d(A_2') : P(A_2' \parallel \mathcal{X}'), \\ \cdots, \\ d(A_n') : P(A_n' \parallel \mathcal{X}') \end{Bmatrix}, \tag{3}$$

And then, the similarity probability of the FlowSpectrum is found by clicking the formula as follows:

$$FS(\mathcal{X}') \cdot FS(\mathcal{X}_r) = P\{Y = r \parallel X \in \mathcal{X}'\}, \tag{4}$$

where $P\{Y = r \parallel X \in \mathcal{X}'\}$ represents the probability of $\mathcal{X}'$ belonging to $\mathcal{X}_r$. According to the theory of the Bayesian optimal classifier, we have the following calculation:

$$\hat{y} = \mathrm{argmin}_{r \in R}\, R(r \mid \boldsymbol{x}) = \mathrm{argmax}_{r \in R}\, P\{Y = r \mid X = \boldsymbol{x}\}, \tag{5}$$

where $R(r \parallel \mathcal{X}')$ is the risk function in the Bayesian classifier, with its value calculated as follows:

$$\hat{y} = \mathrm{argmax}_{r \in R}\, P\{Y = r \parallel X \in \mathcal{X}'\} = \mathrm{argmax}_{r \in R}\, \mathfrak{F}(\mathcal{X}') \cdot \mathfrak{F}(\mathcal{X}_r), \tag{6}$$

When there are unidentified spectral line values within $\mathcal{X}'$, we need to calculate the minimum distance between $d(A)$ and $d(A')$ to determine the unknown spectral line type. Yang's research team [19] proposed a method of exponential decay, where according to the specific representation of $d(A)$ and $d(A')$, we have the following:

$$A' = \mathrm{argmin}_{A' \in \mathcal{X}_t} \parallel d(A) - d(A') \parallel, \tag{7}$$

$$P_r(A) = P_r(A')e^{-\|d(A)-d(A')\|} \tag{8}$$

where A' represents the test instance flow.

3.3. Frame Design

As shown in Figure 3, this represents the overall framework of our research. In the Flow preprocessing stage, packets are converted into a 28 × 28 byte grayscale image dataset. We divided the dataset into training and testing sets at a ratio of 4:1. In the Flow classification stage, we first split the model's input training set into a ratio of 3:1 for training and validation of the model after each training iteration. Then, we train our AMAE model (comprising Self-Attention, Spatial-Attention, and AutoEncoder components) using the training set to generate the standard flowspectrum line charts for different encrypted flows. Following training, we obtain a stable and effective flowspectrum Model. Subsequently, we evaluate the Model using the testing set to obtain the Unsorted Spectrum of the testing data. Finally, based on a flowspectrum classification algorithm, we classify the Unsorted Spectrum to obtain our classification results.

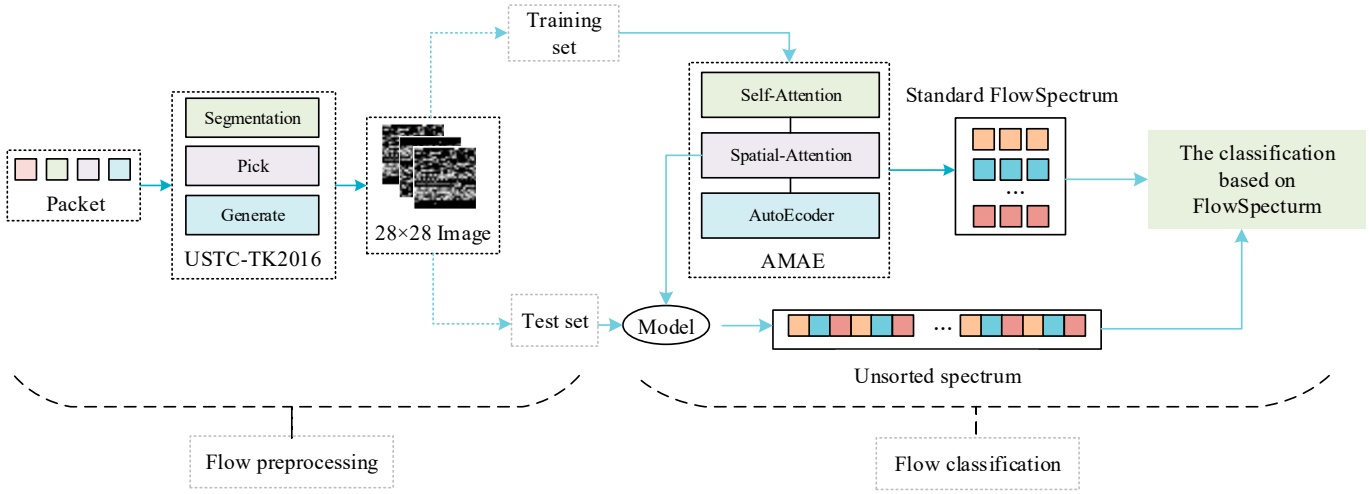


Figure 3. AMAE model structure.

The attention mechanism (AM) was initially introduced for machine translation and is now widely used in artificial intelligence. Our visual processing systems tend to selectively focus on certain parts of images while ignoring other irrelevant information, thus facilitating perception, which is the intuitive explanation behind the AM mechanism. By assigning attention weights to different parts of the input, AM allows the model to dynamically focus on certain parts of the input that are helpful in effectively performing the task at hand. In formal terms, AM can be described as mapping a query and a set of key-value pairs to an output:

$$\text{weights} = \text{AM}(\text{Query}, \text{Key}, \text{Value}) \tag{9}$$

However, for some tasks (such as the network encryption traffic classification required in this paper), the model is given only one input instead of three. For such cases, self-attention, also known as internal attention, has been proposed. In self-attention mechanisms, we set Query = Key = Value = Input and send the three values as different inputs to the single-channel attention, global attention module, and the final weighted module, as shown in Figure 4. Further details will be provided in Section 3.3.

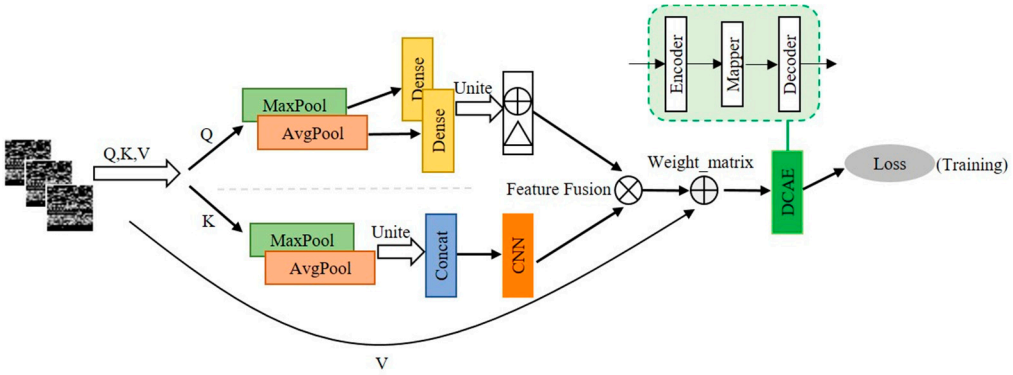


Figure 4. AMAE Structure Block Diagram. Among them, $\oplus$ represents the superposition of the results of two Dense layers; $\triangle$ represent the output of shaping the superposition result; $\otimes$ represents the fusion of two single channel features and global features.

The primary purpose of designing both channel and spatial dimensions is to focus on important features of traffic bytes while suppressing unnecessary regional responses. Through combined analysis in both channel and spatial dimensions, we propose a self-attention module to enhance network performance and suppress irrelevant noise information. We process packet bytes into grayscale images, and the attention module is capable of generating attention grayscale map information in both channel and spatial dimensions. Subsequently, the weights of these two pieces of information are multiplied with the original input grayscale image for adaptive feature correction, producing the final weighted feature matrix. Additionally, this weight matrix explains the byte attention level of features in both dimensions, providing some interpretability to the separation of our encrypted flowspectrum intervals.

Single-channel attention module. Each channel in the input grayscale image is treated as a feature detector, so focusing on channel features is about the "content" of useful information in the image. To more effectively calculate channel attention features, the features of the image need to be further compressed in the spatial dimension. Park et al. [43] adopted a combination of average pooling and max pooling. They experimentally demonstrated that this method can effectively extract features from images. This method is adopted in this paper. Input network flow as $\vec{x}$ in the channel attention module, after pooling output, the channel attention feature matrix $M_c\left(\vec{x}\right)$ is generated through the hidden layer's perceptron MLP, with the following formula for this module, where L represents the learning operator, and W_1 and W_0 represent the weights of transformation:

$$\begin{aligned} M_c\left(\vec{x}\right) &= L\left(MLP\left(\text{AvgPool}\left(\vec{x}\right) + \text{MaxPool}\left(\vec{x}\right)\right)\right) \\ &= L\left(W_1\left(W_0\left(\vec{x}^{\,c}_{\text{avg}} + \vec{x}^{\,c}_{\text{max}}\right)\right)\right) \end{aligned} \tag{10}$$

Spatial attention module. The spatial attention module is an important component that applies self-attention mechanism in the field of computer vision. In convolutional neural networks, each convolution operation only focuses on the size range of the neighborhood convolution kernel, even if the receptive field of the convolutional layer is large, it can only cover a local area. This ignores the influence of pixels that are far apart on the current region, or in other words, ignores their relationship. The goal of the spatial attention module is to capture these wide-ranging relationship features, which are global features. For grayscale images, these features are the relationship weights of pixels at all positions in the image to a certain position. Specifically, given the input network flow $\vec{x}$, first perform global average pooling and global max pooling to produce two two-dimensional outputs $\vec{x}^{\,s}_{\text{avg}}$ and $\vec{x}^{\,s}_{\text{max}}$ respectively (corresponding to the height and width of the input network flow matrix),

representing the spatial distribution of different statistical information. Then, these two feature matrices are stacked together to form a new feature matrix $\overrightarrow{C}$, represented as:

$$\overrightarrow{C} = \mathrm{Con}\left[\overrightarrow{x}^{s}_{\mathrm{avg}}, \overrightarrow{x}^{s}_{\mathrm{max}}\right] \tag{11}$$

The feature matrix $\overrightarrow{C}$ is then fed into a convolutional layer containing convolution operations. This convolutional layer uses a 1×1 convolutional kernel to capture structural information in the space. Finally, the convolutional layer outputs a feature matrix representing the attention weights of each spatial position in the original input grayscale image. These data values are transformed through a Sigmoid function, mapping them to the interval (0, 1). This allows these values to be used as weights, applied to the original input grayscale image through element-wise multiplication, giving different spatial positions different importance. This approach highlights important spatial regions while suppressing less important ones, improving the model's feature extraction and representation effectiveness for network flow data. The attention weight values are denoted as $M_S\left(\overrightarrow{x}\right)$, and their calculation formula is:

$$M_S\left(\overrightarrow{x}\right) = S^j = f\left(b^j + \sum_i w^{ij} * \left(\mathrm{Con}\left[\overrightarrow{x}^{s}_{\mathrm{avg}}, \overrightarrow{x}^{s}_{\mathrm{max}}\right]\right)\right) \tag{12}$$

S^j and x_{i} represent the j-th output map and the i-th input map, respectively. w^{ij} denotes the weights of the 1×1 convolutional filter, * denotes convolution, b^j is the bias parameter of the j-th map, and f represents the activation function.

Self-attention mechanism. As mentioned earlier, in the self-attention mechanism, the QKV triplet values are first determined. Here, we take the channel attention feature matrix $M_c\left(\overrightarrow{x}\right)$ and the spatial attention feature matrix $M_S\left(\overrightarrow{x}\right)$ as the values for Q and K. Then, softmax operation is performed on these two feature matrices (($M_c\left(\overrightarrow{x}\right)$ and $M_S\left(\overrightarrow{x}\right)$) to obtain weight values $Weight\_matrix(W_m)$ ranging from 0 to 1. The specific calculation is as follows, where d represents the dimensionality of K:

$$W_m = \mathrm{softmax}\left(\frac{QK^T}{\sqrt{d}}\right) = \mathrm{softmax}\left(\frac{M_c\left(\overrightarrow{x}\right)\left(M_S\left(\overrightarrow{x}\right)\right)^T}{\sqrt{d}}\right) \tag{13}$$

Finally, we overlay the *Weight_matrix* on V, resulting in our weighted byte feature matrix $M_w\left(\overrightarrow{x}\right)$, which is given by the following:

$$M_w\left(\overrightarrow{x}\right) = W_m * V = W_m * \overrightarrow{x} \tag{14}$$

Here, we analyze the values of the W_m matrix, which represents the pixel weights of the 28×28 grayscale image, reflecting the importance of 784 bytes through the W_m matrix. In Section 4.3, we will analyze the byte importance under different encrypted network flows.

Autoencoder module. The autoencoder [44] is a deep learning dimensionality reduction model under the paradigm of unsupervised learning. Its task is to copy the input to the output as closely as possible to learn a data representation that can be used to reconstruct the provided input. We further optimize the model by using two layers of convolutional layers and one layer of max-pooling layer in the encoder part, while using two layers of convolutional layers and one layer of upsampling layer in the decoder part. We use the

weighted feature matrix $M_w\left(\overrightarrow{x}\right)$ as input data to the autoencoder. Overall, we input the weighted feature matrix $M_w\left(\overrightarrow{x}\right)$ into the encoder E, resulting in the encoded data E:

$$\mathrm{E}_{ed} = \mathrm{E}\left[M_w\left(\overrightarrow{x}\right)\right] \tag{15}$$

After encoding, the data are mapped through a mapper to obtain the spectrum mapping, and the spectrum output is obtained through the softmax function, consistent with our previous work [41]. Finally, we use the decoder to decode the encoded data, and the entire encoding–decoding process is computed using the mean squared error function.

4. Experiments Result and Discussion

To demonstrate the performance of the AMAE FlowSpectrum model, we evaluate our approach through comprehensive experiments. In Section 4.1, we introduce the preparation work before the experiment and provide classification evaluation indicators. In Section 4.2, we present the model-related parameter settings. In Section 4.3, we present the FlowSpectrum generated by the AMAE model through experiments and explain in detail the differences between spectral lines through a comprehensive analysis of attention weight values. Finally, in Section 4.4, we present the performance of FlowSpectrum generated by AMAE for encrypted traffic classification and compare it with some existing experiments to demonstrate the advantages of the proposed model.

4.1. Experiment Setup

(1) Dataset.

This paper selects the ISCX-VPN2016 dataset [45] for analysis. The ISCX-VPN2016 dataset is a classic encrypted flow dataset widely used in research fields such as network security and privacy protection. The dataset is provided and maintained by the Information Security Center of Excellence (ISCX) at the University of Calgary, Canada. It contains original pcap files of non-VPN-encrypted flows and VPN-encrypted flows, allowing us to process the data according to our own requirements. In using this dataset, we removed packets without specific labels, such as browsing and VPN browsing data, and retained six types of encrypted traffic for each encryption type: Chat, Email, File, P2P, Streaming, and VoIP.

(2) Analysis and comparison of scheme design.

To demonstrate the capability of our FlowSpectrum feature representation and the effectiveness of encrypted flow classification, we designed four comparative models: 1D-CNN [32], CNN+RNN [18], Semi-AE [19], and AE models. The AE model represents the part of this paper without the attention module. By comparing with the AE model, we can directly observe the effect of the attention mechanism introduced by the AMAE model. Additionally, the Semi-AE model and AE model can also be compared with our model in terms of FlowSpectrum representation capability. Regarding the interpretable representation of FlowSpectrum, we first obtained 784 attention weights for each class of encrypted flows. Then, we analyzed the different distributions of byte positions that different network flows focus on overall. Furthermore, we provided comparative diagrams of local attention weights to further validate the differences in bytes focused on by the model during FlowSpectrum generation, both numerically and in terms of position.

(3) Experimental tools.

All of these methods work with 8CPU × Intel (R) Xeon (R) Platinum 8375C CPU@2.90GHz Run on the server with RTX 3090 Ti GPU. Python version 3.9.0. The deep learning framework used in this article is Keras.

(4) Evaluation indicators.

In order to evaluate the performance of different methods, we use four indicators: accuracy (ACC), accuracy (precision), recall (Recall), and F1 score (F1) in the experimental evaluation. Each value is calculated as follows:

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN} + \mathrm{TN}}; \tag{16}$$

$$\mathrm{precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}; \tag{17}$$

$$\mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}; \tag{18}$$

$$\mathrm{F1} = \frac{2\mathrm{Recall} * \mathrm{precision}}{\mathrm{precision} + \mathrm{Recall}} \tag{19}$$

Among them, we have the following: ① TP, predicted as positive case and actually positive case; ② FP, predicted as positive case but actually negative case; ③ FN, predicted as negative case but actually positive case; ④ TN, predicted as negative case and actually negative case.

4.2. Parameter

As shown in Table 1, we provide the main network layer parameter settings for the AMAE model. In the AMAE model, the training Epoch is set to 250, Batch_size is 64, and the learning rate is 0.0005. Similar to our previous work [41], we choose the PReLU activation function instead of the ReLU activation function used in the global self-attention model. In the model designed in this study, PreLU activation functions are employed in all neural layers. Overall, selecting PreLU as our activation function offers four advantages:

Table 1. AMAE Network Layer Parameter Settings.

Module	Layer	Setting
Self-attention		--
Single channel attention	Avgpool	--
	Maxpool	--
Spatial attention	Avgpool	--
	Maxpool	--
	Conv2D	kernel_size = 1,#kernels = 1
	Prelu	--
Autoencoder	Conv2D	kernel_size = 3,#kernels = 64
	Conv2D	kernel_size = 3,#kernels = 32
	Prelu	--
	Maxpool	pool_size = 2

(1) Alleviating the dying neuron problem: Training the model with the PreLU function ensures that even if the input data contains negative values, the corresponding neurons will still be activated, thereby avoiding neuron death. If ReLU is used, negative inputs are directly set to zero, which may cause neurons to remain inactive during training, thus affecting the model's learning ability.

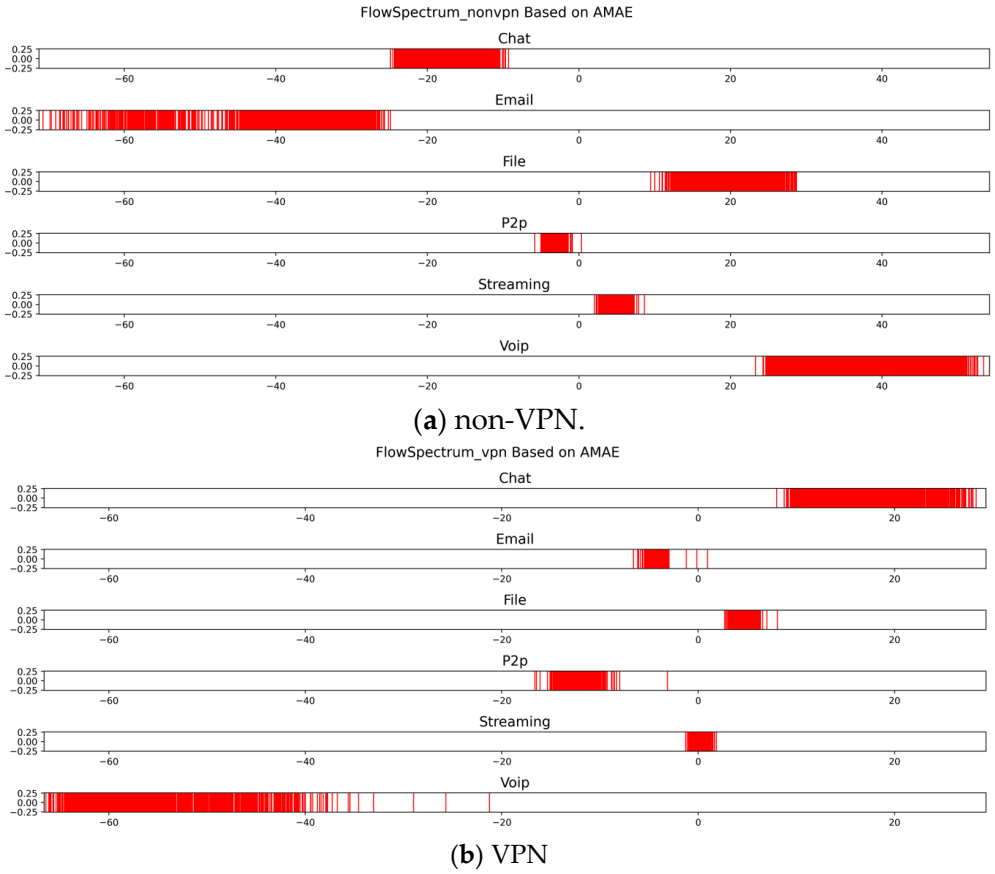
(2) Increasing model expressiveness: PreLU introduces a learnable parameter that can dynamically adjust the shape of the activation function based on the data's characteristics. This enhances the model's flexibility and expressiveness when dealing with complex data distributions.

(3) Mitigating the vanishing gradient problem: With ReLU, the gradients for negative inputs are zero, which may lead to the vanishing gradient problem, especially in deep networks. However, with PreLU, the gradients for negative inputs also propagate, helping alleviate the vanishing gradient problem.

(4) Improved robustness: PreLU exhibits robustness to outliers. ReLU is highly sensitive to negative outliers, as setting them to zero may lead to information loss. PreLU allows the negative part to retain some information, making the model more robust to outliers.

4.3. Analysis of Interpretability Characterization of Features

Our objective is to analyze the characteristics of different types of network traffic on the FlowSpectrum and to interpret the differences in spectral ranges among various network traffic. Initially, we employ the AMAE model to generate FlowSpectrum for VPN-encrypted traffic and non-VPN-encrypted traffic. As illustrated in Figure 5, the spectral characteristics of non-VPN-encrypted traffic and VPN-encrypted traffic generated based on the AMAE model in this study are, respectively, presented. From Figure 5, it can be observed that different types of traffic correspond to different intervals on the FlowSpectrum. For instance, in non-VPN-encrypted traffic, the spectral range of chat flows mostly lies within the interval $(-15, -30)$, while $(-20, -60)$ is the interval where email traffic is predominantly distributed. In VPN-encrypted traffic, the spectral range of chat traffic mostly falls within the interval $(10, 30)$, while $(0, -10)$ is the distribution interval for email traffic. Figure 5 effectively demonstrates VPN-encrypted flows and non-VPN-encrypted flows in a one-dimensional coordinate system. Moreover, by identifying the interval ranges, the distinguishing features of different types of encrypted traffic can be clearly discerned.



(**a**) non-VPN.

(**b**) VPN

Figure 5. FlowSpectrum based on the AMAE model.

Through FlowSpectrum Figure 5, VPN-encrypted traffic and non-VPN-encrypted traffic can be intuitively and effectively presented in a one-dimensional coordinate system. To better illustrate the distribution of spectrum lines generated by the AMAE model (how spectrum lines are distributed near a certain position) and further demonstrate the

significant differences in FlowSpectrum feature representation of different encrypted flows, we start from the perspective of statistical probability and calculate the probability density function (PDF) of FlowSpectrum to reflect the probability density distribution of spectrum lines in the spectrum domain for different encrypted flows. We use the kernel density estimation (KDE) method based on Gaussian kernels to calculate the distribution function curve of spectrum lines. KDE is a non-parametric method used to estimate probability density functions. As shown in Equation (2), the collection of spectrum lines for each type of FlowSpectrum $FS(\mathcal{X}_r)$, where each class of FlowSpectrum contains n spectrum lines. We consider each $d(A_n)$ value in $FS(\mathcal{X}_r)$ as an observation of the probability density function $\hat{f}(x)$ for KDE estimation, which can be estimated using the following formula:

$$\hat{f}(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(d(A) - d(A_i)) \tag{20}$$

where $K_h(\cdot)$ is the kernel function, and h is the bandwidth parameter used to control the width of the kernel function. In this paper, we use the Gaussian kernel function with a bandwidth parameter h of 0.1. The formula for the Gaussian kernel function is as follows:

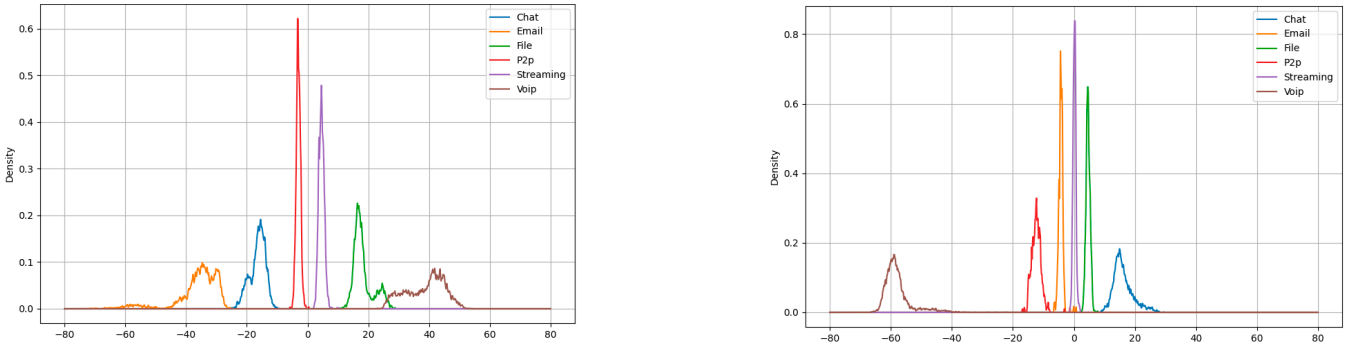
$$K_h(u) = \frac{1}{\sqrt{2\pi}h} e^{-\frac{u^2}{2h^2}}, \tag{21}$$

among them, we have the following:

$$u = d(A) - d(A_i), \tag{22}$$

where u is the distance from the spectral line $d(A_i)$ to the point to be estimated $d(A)$.

As shown in Figure 6, the probability density curve graphs of FlowSpectrum line distribution for different network flows based on the AMAE model are depicted. Observing the probability density distribution curves in the graph, it can be noted that the probability density peaks, distribution shapes, and data dispersion levels vary for different types of FlowSpectrum. Whether it is non-VPN-encrypted FlowSpectrum or VPN-encrypted FlowSpectrum, the AMAE FlowSpectrum model effectively characterizes their features. There is a clear distinction in the separability of their features, including the feature values in the one-dimensional coordinate system and the probability distribution of different FlowSpectrum values. This also demonstrates the feasibility of using FlowSpectrum for classifying network flows.

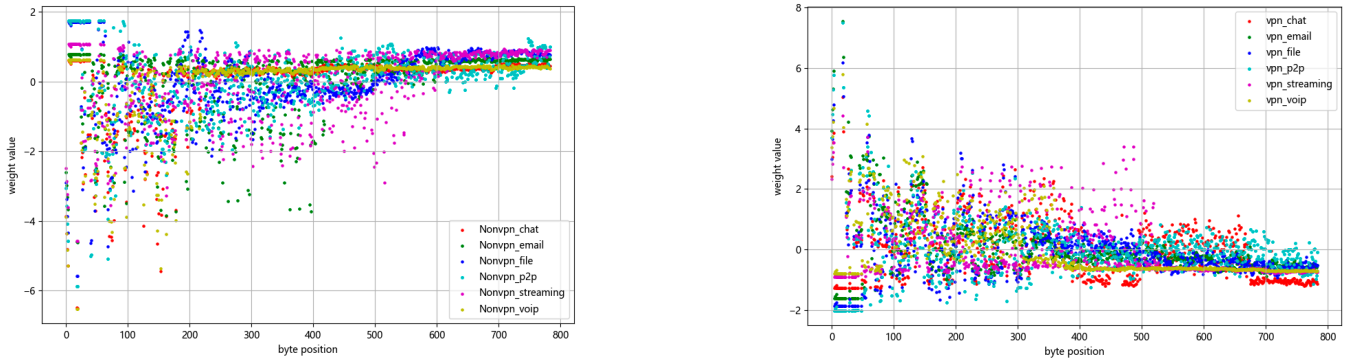


(**a**) Non-VPN (**b**) VPN

Figure 6. Probability density of flow spectral line distribution for different network flows based on the AMAE model.

This article explains the reasons why different FlowSpectrum are located in different intervals from the perspective of input and output results, based on the interpretation type of input. During the training process of the AMAE FlowSpectrum model, the attention mechanism dynamically adjusts the attention weights to reflect the important bytes that

the model is concerned about. We extracted the weight values of each pixel from each 28×28 grayscale image and merged and standardized the weight values of all grayscale images in each traffic class to 784 attention weights. As shown in Figure 7, we visualized these 784 attention weight values. From Figure 7, it can be observed that the interval with the highest dispersion of weight values is between (100, 200). This indicates that the byte differences that different encrypted stream spectra focus on during the formation process are mainly within the position range of (100, 200). In the (600, 784) interval, the dispersion of scatterers is the lowest, the distribution of attention values is uniform, and the model's attention to this position is consistent. We believe that during the dataset processing, some incomplete data packets were filled with 0x00 bytes, resulting in consistent attention from the model to the end of this grayscale image.



(**a**) Non-VPN (**b**) VPN

Figure 7. Attention weight scatter plot.

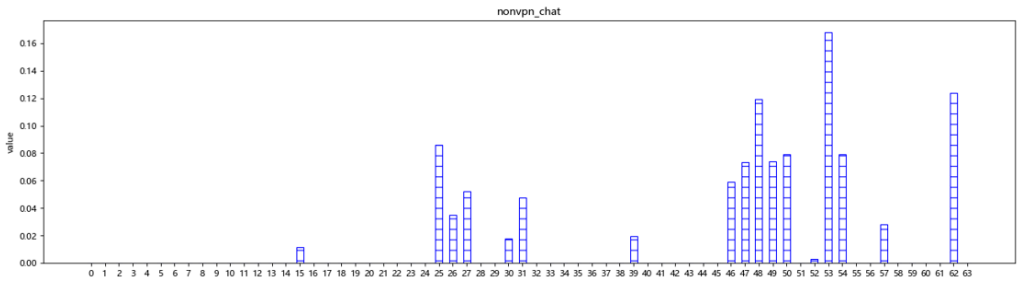
From Figure 7, it can be observed that the positional interval (100, 200) exhibits the maximum dispersion, indicating that the model's attention divergence towards different traffic types is highest within this interval. Therefore, we extracted the weights of the first 64 positions from this interval. As shown in Figures 8 and 9, these extracted weights were normalized and plotted in bar charts. The bar charts highlight the disparity in attention levels for different byte positions across various traffic types. As evident from Figures 8 and 9, the attention weight values of different traffic types at the same position differ. For example, in Figure 8, the attention positions for chat and email traffic are quite similar, with minimal byte attention within the interval (100, 125). P2p traffic exhibits the maximum byte attention within this interval, indicating its significant impact compared to other traffic types. In Figure 9, it can be observed that the attention weight values for VPN-encrypted traffic are much higher compared to non-VPN-encrypted traffic, indicating a significant disparity between the two, thereby explaining the factors contributing to the substantial differences in the spectra of non-VPN-encrypted and VPN-encrypted traffic.

To further explain the differences in FlowSpectrum intervals in relation to the learned weights of the model, we conducted an overall analysis of the weight values. As shown in Figure 10, it presents boxplots of the weight values w_m for different types of traffic. The boxplot displays the minimum and maximum values of the weight w_m, the median (represented by the yellow line inside the box), and the first and third quartiles (top and bottom of the box, respectively). Firstly, it can be observed from the plot that the black circles above the upper limit are outliers, which we refer to as irrelevant or lightweight weights. These weight values have a minor impact on the FlowSpectrum intervals. A higher number of lightweight weight values indicates fewer byte features that the model focuses on during the generation process of such traffic FlowSpectrum. In non-VPN-encrypted traffic, chat traffic has the highest number of lightweight weight values, while VoIP traffic has the lowest, indicating that chat traffic focuses on the least number of bytes, whereas VoIP traffic focuses on the most. In VPN-encrypted traffic, file traffic focuses on the least number of bytes, while email traffic focuses on the most. In non-VPN-encrypted traffic,

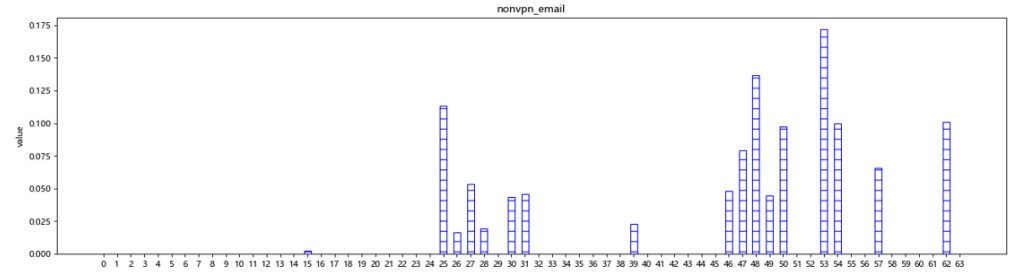
the median values from largest to smallest are email, chat, file, VoIP, streaming, and P2P, while in VPN-encrypted traffic, the median values from largest to smallest are chat, VoIP, file, email, P2P, and streaming. The boxplot also reveals differences in other statistics such as minimum and maximum values, as well as the first and third quartiles. Therefore, this further explains that the differences in FlowSpectrum intervals are due to the different traffic byte loads of different traffic features, and the model generates different intervals of FlowSpectrum by focusing on these bytes.

In summary, from the comprehensive analysis of the FlowSpectrum line distribution of encrypted traffic and the attention weights, it can be concluded that the FlowSpectrum features generated using the AMAE model exhibit excellent interpretable representation performance.

As shown in Figures 11 and 12, we present the FlowSpectrum diagrams of two contrasting models: AE and Semi-AE. From Figure 11, it can be observed that the FlowSpectrum generated by the AE model for VPN-encrypted traffic exhibits slight overlaps. For example, in Figure 11a, there is considerable overlap between the FlowSpectrum of "file" and "VoIP" within the interval $(-20, -10)$. In Figure 11b, significant overlaps are observed in the FlowSpectrum generated by the AE model for VPN-encrypted traffic, indicating poor representation effectiveness. Between the interval $(0, -10)$, the FlowSpectrum of "Email", "File", "P2P", and "Streaming" classes show substantial overlap. In Figure 12, the flow spectra generated by the Semi-AE model for both VPN-encrypted and non-VPN-encrypted traffic exhibit overlaps. For example, as shown in Figure 12a, significant overlaps are observed among the six classes of FlowSpectrum. In Figure 12b, within the interval $(0, 10)$, overlaps are observed among the "Email", "P2P", and "Streaming" classes. We attribute the significant overlaps in the FlowSpectrum generated by the Semi-AE model to its inability to extract structural features of encrypted network traffic. Overall, compared to the AE and Semi-AE models, our AMAE model not only generates better FlowSpectrum diagrams but also effectively explains the differences in spectrum interval by capturing the structural features of the original byte space domain, thereby producing reliable FlowSpectrum models and spectra.

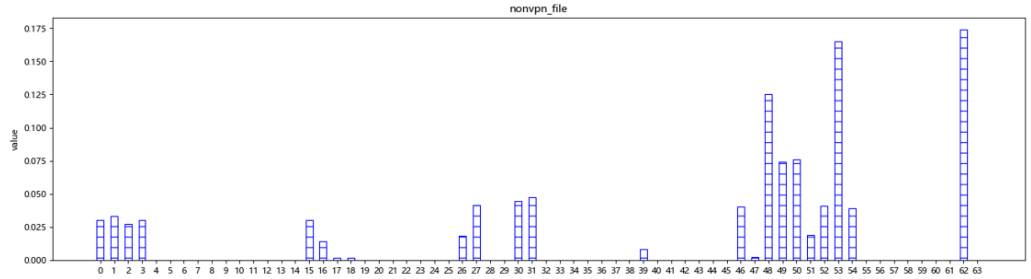


(**a**) Chat

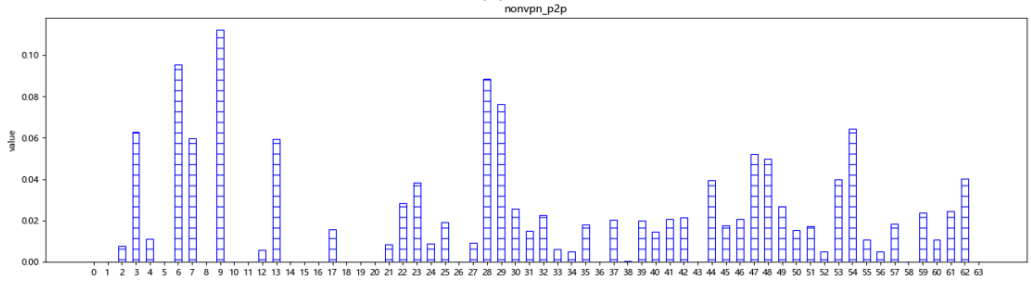


(**b**) Email

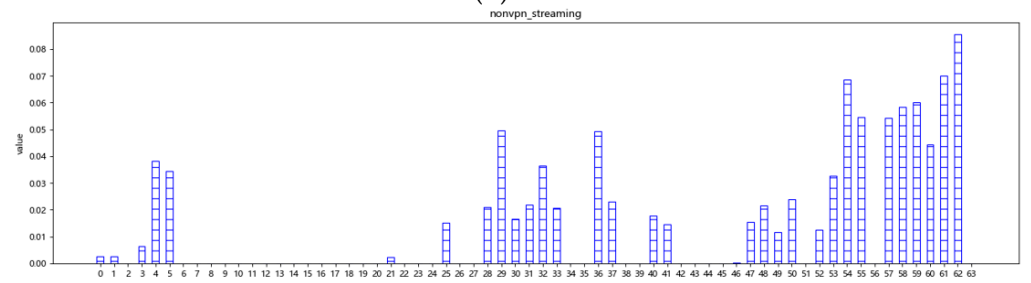
Figure 8. *Cont.*



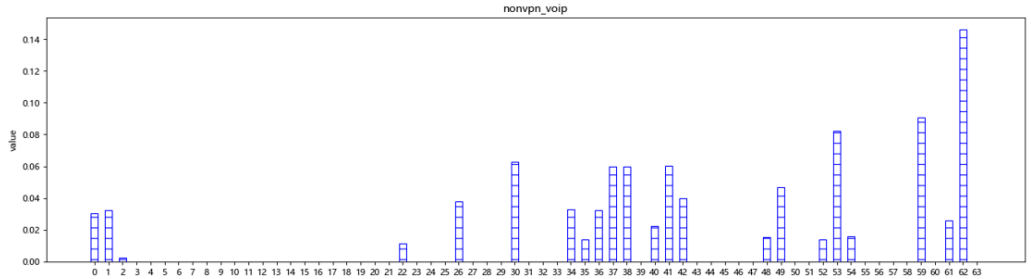
(**c**) File



(**d**) P2P

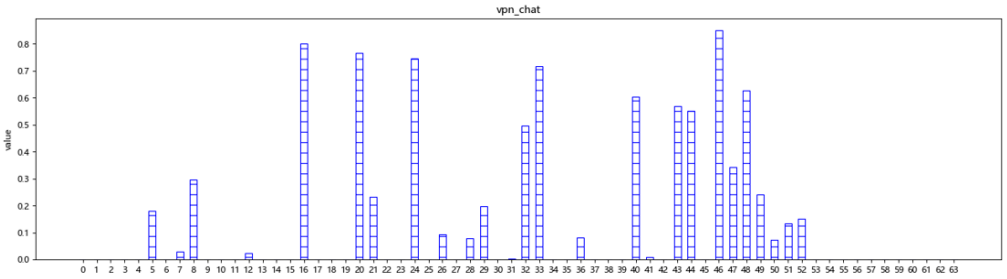


(**e**) Streaming



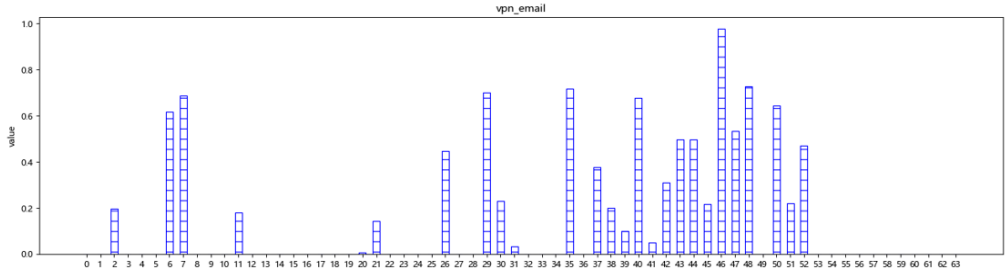
(**f**) VoIP

Figure 8. Non-VPN-encrypted traffic partial byte weight value.

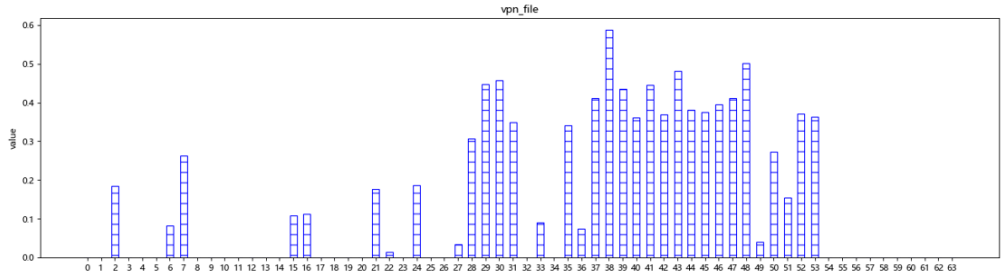


(**a**) Chat

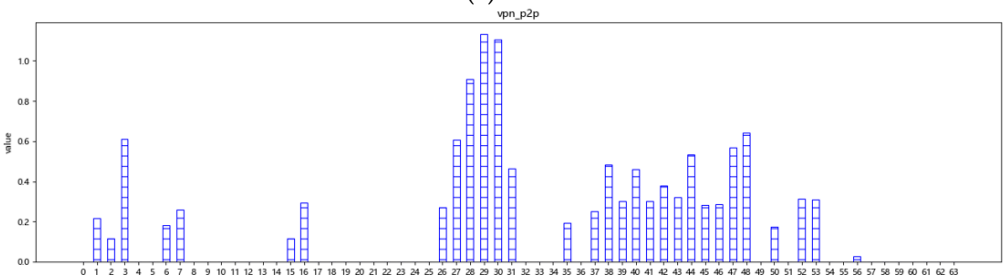
Figure 9. *Cont*.



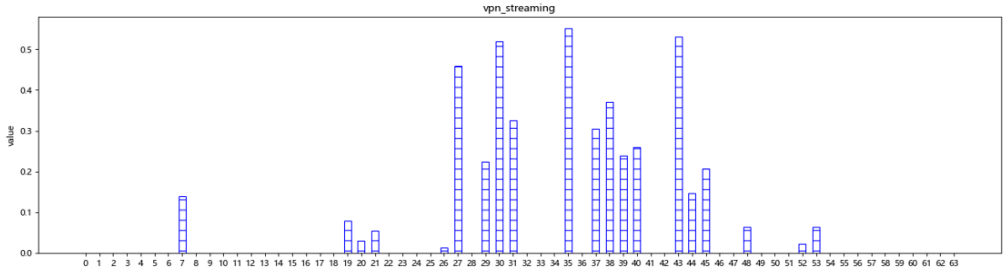
(**b**) Email



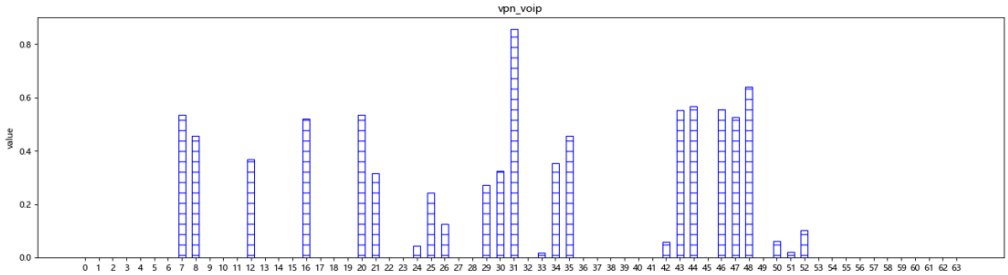
(**c**) File



(**d**) P2P

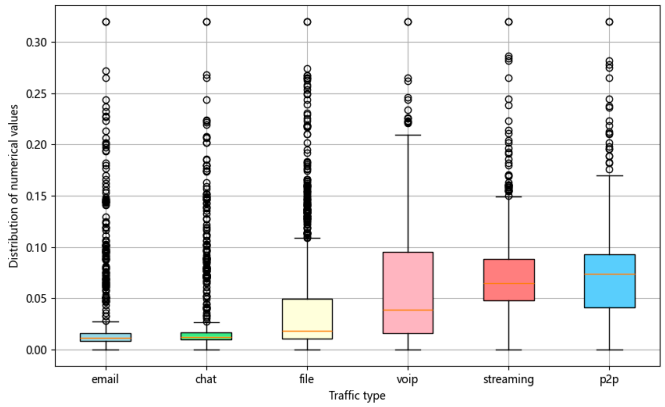


(**e**) Streaming

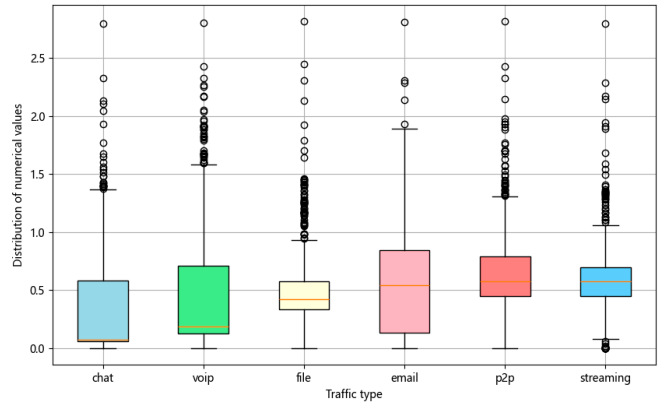


(**f**) VoIP

Figure 9. VPN-encrypted traffic partial byte weight value.

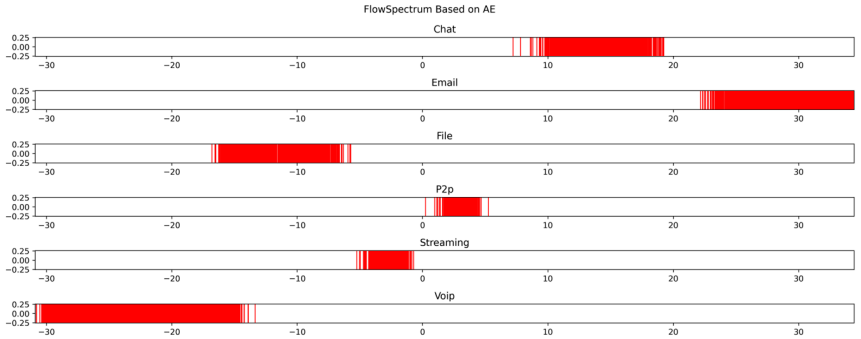


(**a**) non-VPN

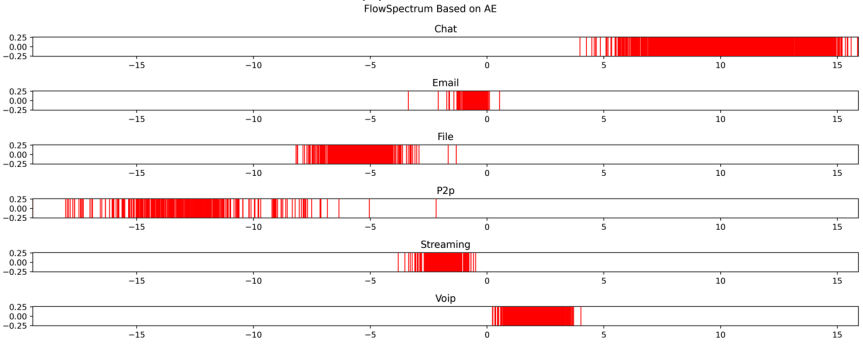


(**b**) VPN

Figure 10. Boxplot shows the average distribution of 784-byte weight values for VPN-encrypted traffic and non-VPN-encrypted traffic. Sort traffic types from left to right in order of median size.

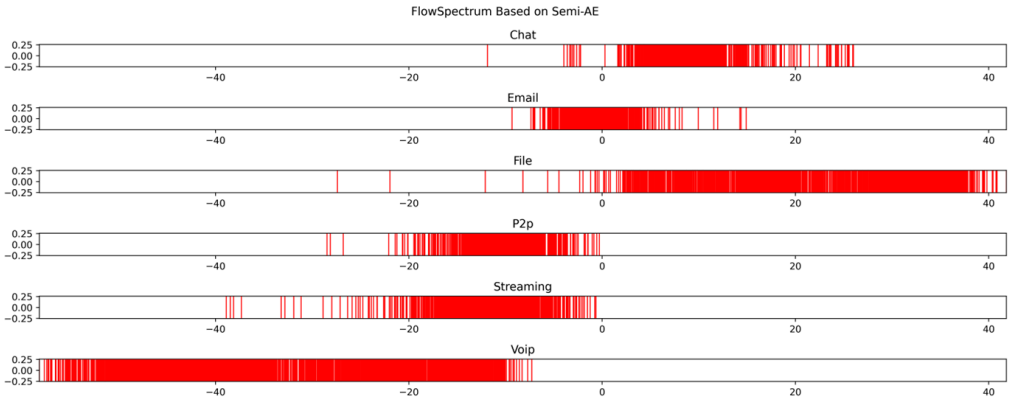


(**a**) Non-VPN

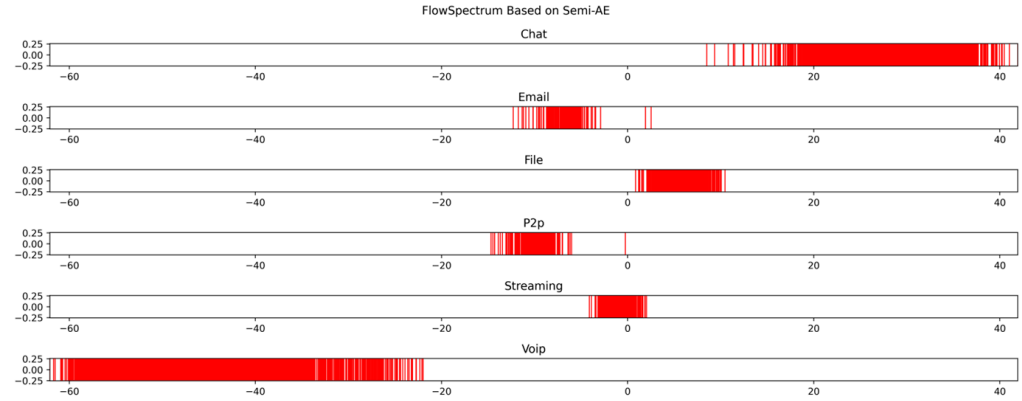


(**b**) VPN

Figure 11. FlowSpectrum based on the AE model.



(**a**) Non-VPN



(**b**) VPN

Figure 12. FlowSpectrum based on the Semi-AE model.

4.4. Comparison and Analysis of Classification Tests

As presented in Section 4.3, FlowSpectrum graphs based on different FlowSpectrum models for encrypted traffic feature representation were demonstrated. In this section, we will utilize FlowSpectrum for encrypted traffic classification and compare them with four other benchmarks (Semi-AE, AE, 1D-CNN, and CNN+RNN) models. Tables 2 and 3 display the average results of 10 classifications for non-VPN-encrypted traffic and VPN-encrypted traffic. In the classification of non-VPN-encrypted traffic, the classification results based on our method reached 99.99%, which is the highest, while the classification result based on 1D-CNN was 76.11%, the lowest. Regarding the classification of VPN-encrypted traffic, our method achieved a classification result of 99.766%, also the highest, while the classification result based on Semi-AE was 91.26%, the lowest. Table 3 demonstrates the effectiveness of our FlowSpectrum model. Overall, our classification scheme outperforms both classical and state-of-the-art FlowSpectrum classification schemes.

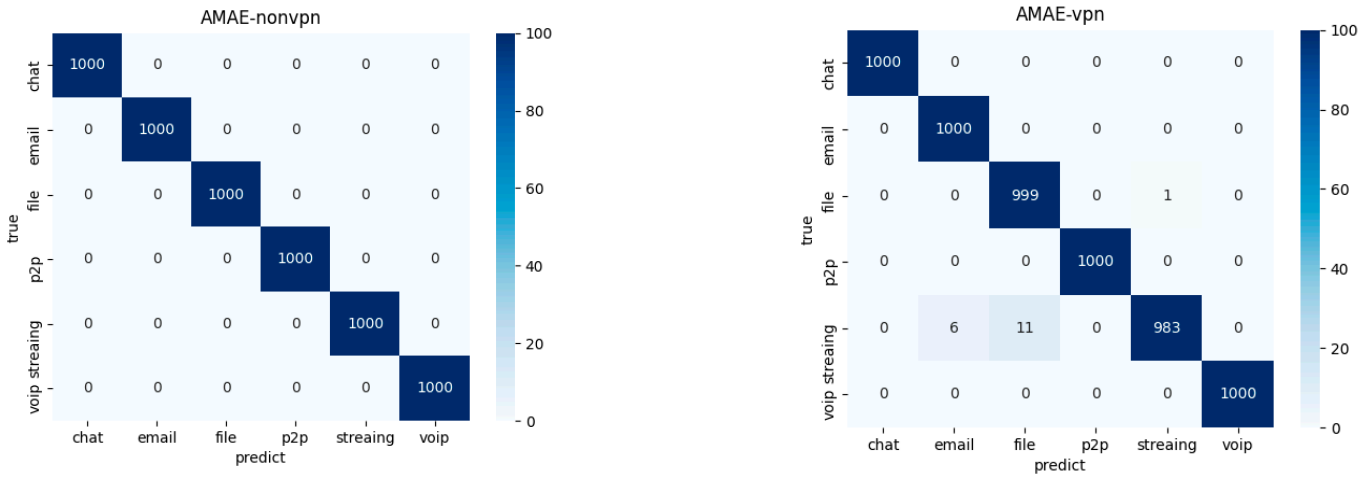
Table 2. Non-VPN-encrypted traffic classification test results.

Model	Accuracy	Precision	Recall	F1-Score
1D-CNN	76.1 ± 0.014%	81.4 ± 0.056%	80.8 ± 0.006%	79.4 ± 0.039%
CNN+RNN	89.6 ± 0.073%	91.8 ± 0.037%	90.9 ± 0.035%	91.2 ± 0.063%
Semi-AE	81.4 ± 0.083%	83.2 ± 0.067%	81.4 ± 0.083%	79.5 ± 0.005%
AE	97.3 ± 0.003%	97.6 ± 0.076%	97.3 ± 0.003%	97.2 ± 0.082%
SAAE	99.9 ± 0.098%	99.9 ± 0.098%	99.9 ± 0.082%	99.9 ± 0.097%

Table 3. VPN-encrypted traffic classification test results.

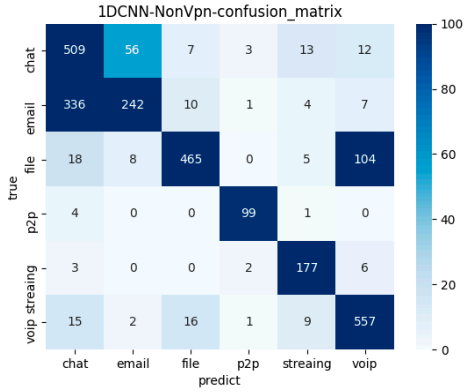
Model	Accuracy	Precision	Recall	F1-Score
1D-CNN	91.4 ± 0.060%	85.8 ± 0.067%	82.1 ± 0.078%	83.3 ± 0.051%
CNN+RNN	98.5 ± 0.063%	96.8 ± 0.052%	96.9 ± 0.085%	96.8 ± 0.034%
Semi-AE	91.2 ± 0.067%	92.1 ± 0.080%	91.2 ± 0.067%	91.1 ± 0.030%
AE	98.5 ± 0.063%	96.8 ± 0.052%	96.9 ± 0.085%	96.8 ± 0.034%
SAAE	99.7 ± 0.066%	99.7 ± 0.066%	99.7 ± 0.066%	99.7 ± 0.066%

The discussion above mainly focuses on overall performance. Now, we analyze the classification performance of the five approaches on different types of flows for two encryption methods. As shown in Figures 13 and 14, we further provide the confusion matrices for the classification of encrypted traffic by the AMAE model and the other four comparison models. In the confusion matrix, we present the specific classification performance evaluation for each class of traffic. The darker the color, the closer the numbers on the main diagonal are to 1000, indicating better classification performance. For example, Figure 13a shows that the AMAE model achieves the best classification accuracy of 100% for each class of traffic for non-VPN-encrypted traffic; from Figure 13b, it can be observed that some streaming traffic was predicted as email traffic and file traffic types, resulting in the value of "983" on the diagonal corresponding to streaming traffic. Figure 14g,h provide the confusion matrices for classification based on the AE FlowSpectrum model. From the figures, it can be seen that different models have slightly different single-class classification accuracies for encrypted traffic. However, compared to the AMAE model, there are slight deficiencies.

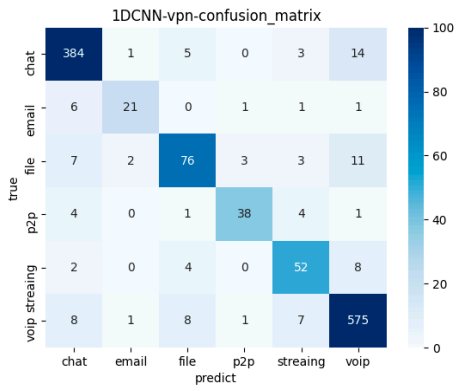


(**a**) Non-VPN (**b**) VPN

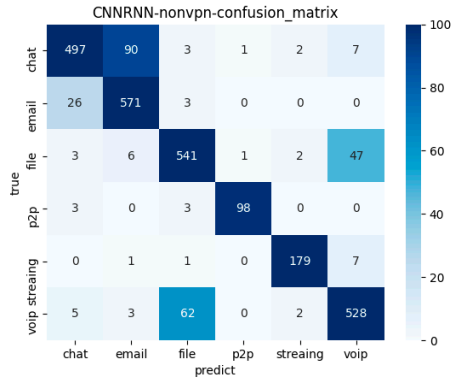
Figure 13. Classification confusion matrix for encrypted traffic of VPN and non-VPN based on AMAE model.



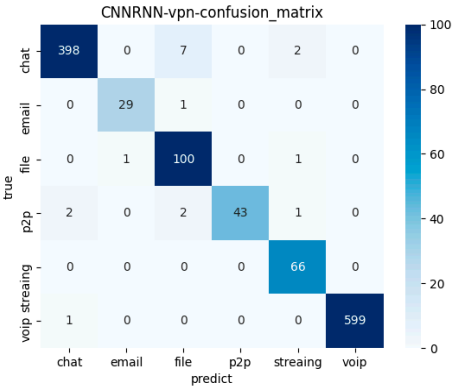
(**a**) Non-VPN



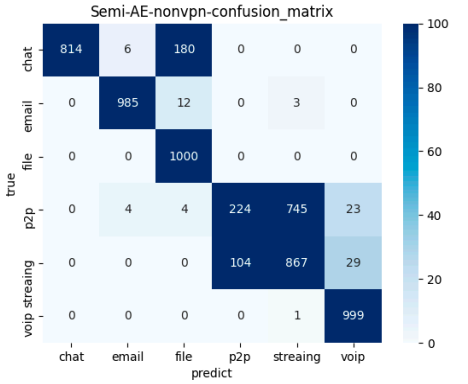
(**b**) VPN



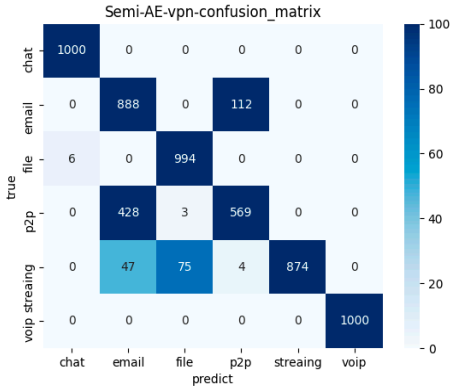
(**c**) Non-VPN



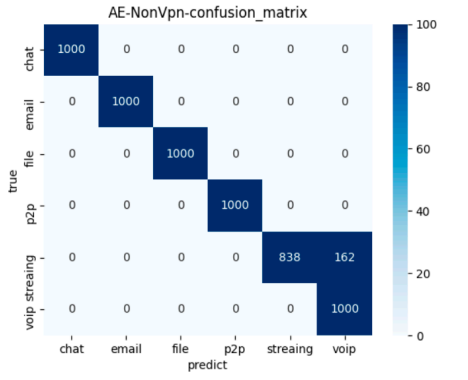
(**d**) VPN



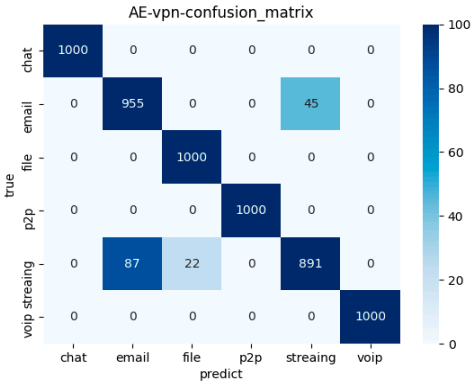
(**e**) Non-VPN



(**f**) VPN



(**g**) Non-VPN



(**h**) VPN

Figure 14. Classification confusion matrix for encrypted traffic of VPN and non-VPN based on four types of comparison models.

5. Conclusions and Future Work

5.1. Conclusions

With the increasingly prominent issues of network security, network resource allocation, and network service quality, researchers in network traffic analysis have begun to use machine learning and deep learning techniques to classify network traffic, in order to improve the accuracy and efficiency of classification. However, existing traffic classification design methods have some limitations, such as the complexity of machine learning methods, the "black box" nature of deep learning methods, and the inability to effectively represent network popularity as features. The proposal of FlowSpectrum theory has brought new ideas to network traffic analysis. By describing the characteristics of network popularity in a low-dimensional manner, traffic behavior characteristics can be intuitively analyzed, thereby improving classification performance.

Based on the theory of FlowSpectrum, this article proposes an AMAE FlowSpectrum model for interpretable representation and traffic classification of encrypted traffic features. The AMAE model combines channel attention and spatial attention through a self-attention mechanism, which not only fully extracts the global spatial structural features of network traffic bytes, but also utilizes the generated attention weight values to explain the differences in generated FlowSpectrum results. Overall, this article first utilizes the AMAE model to generate FlowSpectrum for different encrypted traffic. Then, the probability density distribution of FlowSpectrum lines is analyzed using a kernel density estimation method based on the Gaussian kernel. Furthermore, based on the input interpretation type, this article conducts an overall and local analysis of the weight values generated by different encrypted traffic during the training process. By effectively explaining the differences in the FlowSpectrum interval through the different attention positions and levels of bytes in different encrypted traffic. Finally, this article conducted classification testing on encrypted traffic using the generated FlowSpectrum.

Our solution provides an effective approach to addressing the classification and feature representation of encrypted network traffic, with practical significance. Through our method, the management and analysis of cybersecurity in cyberspace can be enhanced. We believe that this will offer new insights for researchers in the field of cyberspace and provide them with additional avenues for exploration.

5.2. Future Work

This article studies the interpretable feature representation and classification of network encrypted traffic based on FlowSpectrum theory combined with deep learning-related technologies. In the future, our research will mainly involve the following two aspects:

(1) Multidimensional characterization of FlowSpectrum.

In the future, we will continue to study both FlowSpectrum characterization and interpretability. In our model, mapping network traffic to one-dimensional coordinates is the process of obtaining one-dimensional features by transforming the spatial characteristics of the traffic through a series of transformations. In the future, we will explore multidimensional (including two-dimensional and three-dimensional) spectral line features under various mapping methods, including time domain, spatial domain, and frequency domain, to further better characterize network traffic content features, behavioral features, and so on.

(2) The interpretability of FlowSpectrum.

In the future, based on the characterization curves and multi-dimensional representation methods of coordinate systems, we can establish more comprehensive interpretable schemes for FlowSpectrum, such as interpretable schemes based on model networks and interpretable schemes based on processes. The completeness of the interpretable representation of FlowSpectrum can greatly expand the practical application of FlowSpectrum, enabling them to interpret the results of FlowSpectrum models in the field of network security.

Author Contributions: Conceptualization, J.C. and L.B.; methodology, J.C., L.B. and X.Z.; software, X.Z. and L.B.; validation, J.C. and L.B.; formal analysis, L.B.; investigation, L.B. and X.Z.; resource, Z.L. and Q.L; data curation, L.B.; writing—original draft preparation, J.C. and L.B.; writing—review and editing, J.C., Q.L. and L.B.; visualization, J.C. and L.B.; supervision, J.C., Z.L. and Q.L; project administration, J.C. and L.B.; funding acquisition, Q.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: No new Data were created.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Azab, A.; Khasawneh, M.; Alrabaee, S.; Choo, K.-K.R.; Sarsour, M. Network traffic classification: Techniques, datasets, and challenges. *Digit. Commun. Netw.* **2022**. [CrossRef]
2. Karagiannis, T.; Broido, A.; Faloutsos, M.; Claffy, K. Transport layer identification of P2P traffic. In Proceedings of the Fourth ACM SIGCOMM Conference on Internet Measurement, Sicily, Italy, 25–27 October 2004; pp. 121–134.
3. Tahaei, H.; Afifi, F.; Asemi, A.; Zaki, F.; Anuar, N.B. The rise of traffic classification in IoT networks: A survey. *J. Netw. Comput. Appl.* **2020**, *154*, 102538. [CrossRef]
4. Wang, Y.; Xiang, Y.; Yu, S.Z. Automatic application signature construction from unknown traffic. In Proceedings of the IEEE International Conference on Advanced Information Networking and Applications, Perth, Australia, 20–23 April 2010; pp. 1115–1120.
5. Gai, K.; Qiu, M.; Zhao, H. Privacy-preserving data encryption strategy for big data in mobile cloud computing. In Proceedings of the IEEE Transactions on Big Data, Seattle, WA, USA, 10–13 December 2018; pp. 678–688.
6. Dong, Y.N.; Zhao, J.J.; Jin, J. Novel feature selection and classification of internet video traffic based on a hierarchical scheme. *Comput. Netw.* **2017**, *119*, 102–111. [CrossRef]
7. Govindarajan, M.; Chandrasekaran, R.M. Intrusion detection using k-Nearest Neighbor. In Proceedings of the 2009 First International Conference on Advanced Computing, Chennai, India, 13–15 December 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 13–20.
8. Quinlan, J.R. Induction of decision trees. *Mach. Learn.* **1986**, *1*, 81–106. [CrossRef]
9. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
10. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
11. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
12. Peng, H.; Li, J.; He, Y.; Liu, Y.; Bao, M.; Wang, L.; Yang, Q. Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In Proceedings of the 2018 World Wide Web Conference, Lyon, France, 23–27 April 2018; pp. 1063–1072.
13. Yao, L.; Mao, C.; Luo, Y. Graph convolutional networks for text classification. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 7370–7377.
14. Bayat, N.; Jackson, W.; Liu, D. Deep learning for network traffic classification. *arXiv preprint* **2021**, arXiv:2106.12693.
15. Krupski, J.; Graniszewski, W.; Iwanowski, M. Data transformation schemes for cnn-based network traffic analysis: A survey. *Electronics* **2021**, *10*, 2042. [CrossRef]
16. Ren, X.; Gu, H.; Wei, W. Tree-RNN: Tree structural recurrent neural network for network traffic classification. *Expert Syst. Appl.* **2021**, *167*, 114363. [CrossRef]
17. D'Angelo, G.; Palmieri, F. Network traffic classification using deep convolutional recurrent autoencoder neural networks for spatial–temporal features extraction. *J. Netw. Comput. Appl.* **2021**, *173*, 102890. [CrossRef]
18. Yao, H.; Liu, C.; Zhang, P.; Wu, S.; Jiang, C.; Yu, S. Identification of encrypted traffic through attention mechanism based long short term memory. *IEEE Trans. Big Data* **2019**, *8*, 241–252. [CrossRef]
19. Yang, L.; Fu, S.; Zhang, X.; Guo, S.; Wang, Y.; Yang, C. FlowSpectrum: A concrete characterization scheme of network traffic behavior for anomaly detection. *World Wide Web* **2022**, *25*, 2139–2161. [CrossRef]
20. Guo, S.; Lü, R.; He, M.; Zhang, J.; Yu, S. Application of Flow Spectrum Theory in Network Defense. *J. Beijing Univ. Posts Telecommun.* **2022**, *45*, 19–25.
21. Guo, S.; Wang, X.; He, M. Research on Intelligent Monitoring Technology in Cyberspace Adversarial Defense. *Inf. Secur. Commun. Priv.* **2021**, *11*, 79–94.
22. Moore, A.W.; Zuev, D. Internet traffic classification using bayesian analysis techniques. In Proceedings of the 2005 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems, Banff, AB, Canada, 6–10 June 2003; pp. 50–60.
23. Velan, P.; Čermák, M.; Čeleda, P.; Drašar, M. A survey of methods for encrypted traffic classification and analysis. *Int. J. Netw. Manag.* **2015**, *25*, 355–374. [CrossRef]

24. Vlăduţu, A.; Comăneci, D.; Dobre, C. Internet traffic classification based on flows' statistical properties with machine learning. *Int. J. Netw. Manag.* **2017**, *27*, e1929. [CrossRef]
25. Pacheco, F.; Exposito, E.; Gineste, M.; Baudoin, C.; Aguilar, J. Towards the deployment of machine learning solutions in network traffic classification: A systematic survey. *IEEE Commun. Surv. Tutor.* **2018**, *21*, 1988–2014. [CrossRef]
26. Dusi, M.; Este, A.; Gringoli, F.; Salgarelli, L. Using GMM and SVM-based techniques for the classification of SSH-encrypted traffic. In Proceedings of the 2009 IEEE International Conference on Communications, Dresden, Germany, 14–18 June 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 1–6.
27. Lashkari, A.H.; Gil, G.D.; Mamun, M.S.I.; Ghorbani, A.A. Characterization of tor traffic using time based features. In Proceedings of the International Conference on Information Systems Security and Privacy, Porto, Portugal, 19–21 February 2017; SciTePress: Setubal, Portugal, 2017; Volume 2, pp. 253–262.
28. Gil, G.D.; Lashkari, A.H.; Mamun, M.; Ghorbani, A.A. Characterization of encrypted and VPN traffic using time-related features. In Proceedings of the 2nd International Conference on Information Systems Security and Privacy (ICISSP 2016), Rome, Italy, 19–21 February 2016; SciTePress: Setubal, Portugal, 2016; pp. 407–414.
29. Zong, W.; Chow, Y.W.; Susilo, W. A 3d approach for the visualization of network intrusion detection data. In Proceedings of the 2018 International Conference on Cyberworlds (CW), Singapore, 3–5 October 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 308–315.
30. Imran, H.M.; Abdullah, A.B.; Hussain, M.; Palaniappan, S.; Ahmad, I. Intrusions detection based on optimum features subset and efficient dataset selection. *Int. J. Eng. Innov. Technol.* **2012**, *2*, 265–270.
31. Santos, A.C.F.; da Silva, J.D.S.; de Sá Silva, L.; da Costa Sene, M.P. Network traffic characterization based on time series analysis and computational intelligence. *J. Comput. Interdiscip. Sci.* **2011**, *2*, 197–205.
32. Wang, W.; Zhu, M.; Wang, J.; Zeng, X.; Yang, Z. End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In Proceedings of the 2017 IEEE International Conference on Intelligence and Security Informatics (ISI), Beijing, China, 22–24 July 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 43–48.
33. Zeng, Y.; Gu, H.; Wei, W.; Guo, Y. *Deep-Full-Range*: A deep learning based network encrypted traffic classification and intrusion detection framework. *IEEE Access* **2019**, *7*, 45182–45190. [CrossRef]
34. Wang, W.; Sheng, Y.; Wang, J.; Zeng, X.; Ye, X.; Huang, Y.; Zhu, M. HAST-IDS: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection. *IEEE Access* **2017**, *6*, 1792–1806. [CrossRef]
35. Dai, J.; Xu, X.; Xiao, F. Glads: A global-local attention data selection model for multimodal multitask encrypted traffic classification of iot. *Comput. Netw.* **2023**, *225*, 109652. [CrossRef]
36. Lotfollahi, M.; Jafari Siavoshani, M.; Shirali Hossein Zade, R.; Saberian, M. Deep packet: A novel approach for encrypted traffic classification using deep learning. *Soft Comput.* **2020**, *24*, 1999–2012. [CrossRef]
37. Höchst, J.; Baumgärtner, L.; Hollick, M.; Freisleben, B. Unsupervised traffic flow classification using a neural autoencoder. In Proceedings of the 2017 IEEE 42Nd Conference on Local Computer Networks (LCN), Singapore, 9–12 October 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 523–526.
38. Ferreira, D.C.; Vázquez, F.I.; Zseby, T. Extreme dimensionality reduction for network attack visualization with autoencoders. In Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, 14–19 July 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–10.
39. Javaid, A.; Niyaz, Q.; Sun, W. A deep learning approach for network intrusion detection system. In Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (Formerly BIONETICS), Utrecht, The Netherlands, 28–30 June 2016; pp. 21–26.
40. Xie, G.; Li, Q.; Jiang, Y. Self-attentive deep learning method for online traffic classification and its interpretability. *Comput. Netw.* **2021**, *196*, 108267. [CrossRef]
41. Cui, J.; Bai, L.; Li, G.; Zeng, O. Semi-2DCAE: A semi-supervision 2D-CNN AutoEncoder model for feature representation and classification of encrypted traffic. *PeerJ Comput. Sci.* **2023**, *9*, e1635. [CrossRef]
42. Wang, W.; Zhu, M.; Zeng, X.; Ye, X. Malware traffic classification using convolutional neural network for representation learning. In Proceedings of the 2017 International Conference on Information Networking (ICOIN), Da Nang, Vietnam, 11–13 January 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 712–717.
43. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. Cbam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.
44. Hinton, G.E.; Salakhutdinov, R.R. Reducing the dimensionality of data with neural networks. *Science* **2006**, *313*, 504–507. [CrossRef]
45. Draper-Gil, G.; Lashkari, A.H.; Mamun, M.S.I.; Ghorbani, A.A. Characterization of encrypted and vpn traffic using time-related. In Proceedings of the 2nd International Conference on Information Systems Security and Privacy (ICISSP), Rome, Italy, 19–21 February 2016; pp. 407–414.