*electronics*

MDPI

*Article*

# Network Traffic Classification Model Based on Spatio-Temporal Feature Extraction

**Cheng Wang [1,2], Wei Zhang [1,2], Hao Hao [1,2,*] and Huiling Shi [1,2,*]**

[1] Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Shandong Computer Science Center (National Supercomputer Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences), Jinan 250353, China; 10431220530@stu.qlu.edu.cn (C.W.); wzhang@sdas.org (W.Z.)

[2] Shandong Provincial Key Laboratory of Computer Networks, Shandong Fundamental Research Center for Computer Science, Jinan 250014, China

* Correspondence: haoh@sdas.org (H.H.); shihl@sdas.org (H.S.)

**Abstract:** The demand for encrypted communication is increasing with the continuous development of secure and trustworthy networks. In edge computing scenarios, the requirement for data processing security is becoming increasingly high. Therefore, the accurate identification of encrypted traffic has become a prerequisite to ensure edge intelligent device security. Currently, encrypted network traffic classification relies on single-feature extraction methods. These methods have simple feature extraction, making distinguishing encrypted network data flows and designing compelling manual features challenging. This leads to low accuracy in multi-classification tasks involving encrypted network traffic. This paper proposes a hybrid deep learning model for multi-classification tasks to address this issue based on the synergy of dilated convolution and gating unit mechanisms. The model comprises a Gated Dilated Convolution (GDC) module and a CA-LSTM module. The GDC module completes the spatial feature extraction of encrypted network traffic through dilated convolution and gating unit mechanisms. In contrast, the CA-LSTM module focuses on extracting temporal network traffic features. By employing a collaborative approach to extract spatio-temporal features, the model ensures feature extraction diversity, guarantees robustness, and effectively enhances the feature extraction rate. We evaluate our multi-classification model using the ISCX VPN-nonVPN public dataset. Experimental results show that the proposed method achieves an accuracy rate of over 95% and a recall rate of over 90%, significantly outperforming existing methods.

**Keywords:** edge computing; encrypted traffic classification; deep learning; intelligent network



**Citation:** Wang, C.; Zhang, W.; Hao, H.; Shi, H. Network Traffic Classification Model Based on Spatio-Temporal Feature Extraction. *Electronics* **2024**, *13*, 1236. https://doi.org/10.3390/electronics13071236

Academic Editor: Dimitra I. Kaklamani

Received: 13 February 2024
Revised: 16 March 2024
Accepted: 26 March 2024
Published: 27 March 2024



## 1. Introduction

With the continuous development of secure and reliable networks, Internet technology has permeated every aspect of human life. The diversity and quantity of network traffic data have significantly increased. In network security, there is a rising trend towards encrypting network traffic. Against this backdrop, network traffic classification [1] has become a key research focus in edge intelligent network management. Network traffic classification is the process of categorizing traffic according to different requirements, and is essential in enhancing network security, optimizing network resource management, and improving network service quality [2,3]. Therefore, the effectiveness and accuracy of network traffic classification are crucial for maintaining network service quality and detecting the early signs of potential abnormal network activities, making it a highly significant topic. The complexity of network traffic feature distribution poses challenges for traditional methods which require manual feature extraction, often necessitating prolonged traffic collection and consuming substantial amounts of cache and storage resources. In contrast, deep learning-based methods for encrypted network traffic classification no longer rely

on manual feature design, effectively identifying encrypted and anonymous traffic and providing a more granular traffic identification mechanism.

As an emerging computing paradigm, edge computing has become a research hotspot due to its computing advantages close to data sources at the network's edge, data privacy and security protection, and the efficient response to and processing of data locally [4]. For example, intelligent terminals provide users with a more stable service experience with proximity, real-time security, and quick response processing [5]. How to combine network traffic classification with the real-time processing capabilities of edge computing to promote the development of edge intelligence. As a critical component of edge computing architecture, edge intelligent gateways facilitate data processing, decision support, and thoughtful service deployment at the network's edge [6], improving user-perceivable network service quality.

In network security, deep learning technologies [7] have shown initial advancements in classifying network traffic, yet existing methodologies have the potential for enhancement. For example, considering an entire network flow as a sequence of either traffic bytes or network packets allows the application of Long Short-Term Memory (LSTM) networks to extract temporal characteristics [8]. Conversely, interpreting a network flow as a traffic image composed of bytes enables Convolutional Neural Networks (CNNs) to discern spatial features [9]. Thus, CNNs and LSTMs, as the predominant techniques in deep learning, are apt for varied scenarios, with their feature learning capabilities being optimally leveraged according to the specific objectives of the application [10]. Nonetheless, these approaches are generally confined to extracting solely temporal or spatial features. A more sophisticated approach involving a hybrid structure integrating spatial and temporal feature extraction can produce markedly superior performance outcomes.

Different types of encrypted traffic exhibit distinct temporal and spatial characteristics. Temporal features are often reflected in attributes such as packet arrival order, inter-arrival time intervals of flows, and flow duration. There are significant differences in the temporal characteristics of encrypted traffic among various service types, including packet arrival order, inter-arrival time intervals of flows, and flow duration. Therefore, utilizing neural networks to extract the temporal characteristics of encrypted traffic enables the accurate and efficient classification of encrypted traffic. As for spatial features, different services employ distinct communication patterns and encryption algorithms, such as packet sizes and flow sizes. Hence, leveraging neural networks to extract the spatial characteristics of encrypted traffic facilitates precise and efficient classification of encrypted traffic.

Researchers are exploring using multiple neural networks to comprehensively analyze heterogeneous feature information in network traffic, a task traditionally performed by a single neural network for feature extraction in deep learning tasks. Reference [11] introduces a hybrid deep learning model that leverages an autoencoder (an unsupervised learning algorithm) and bidirectional extended short-term memory networks (BLSTM, a type of recurrent neural network suitable for sequence data) to reduce the feature dimensionality of large-scale Internet of Things (IoT) network traffic data, thereby enhancing network traffic classification. However, this approach must account for the increasing complexity of large-scale Internet of Things (IoT) network traffic data. Solely focusing on temporal features may hinder achieving accurate and efficient classification. Reference [12] combines recurrent neural networks (RNNs, specialized neural networks for handling sequence data) and convolutional neural networks (CNNs, neural networks used for tasks like image recognition) to produce outstanding detection results, surpassing other algorithms without the need for any feature engineering. However, this method needs to be further improved when more complex and long-term dependent sequence data need to be processed in the natural network environment, especially when the sequence data are very long or contain complex context information. Reference [13] proposes a TSCRNN model aimed at learning the spatiotemporal features of network traffic data, providing a finer mechanism for traffic representation to support advancements in core industrial IoT technologies. The model's potential requirement to model traffic data over long periods and consider the complex

relationships of spatiotemporal features necessitate more computational resources and time for training and inference, limiting its scalability and practicality in real-world applications.

To address the above-mentioned challenges, we propose a hybrid deep learning model for encrypted network traffic classification, capitalizing on the spatio-temporal characteristics of encrypted network traffic. This model leverages dilated convolution [14] for extracting spatial features from network flows, and Long Short-Term Memory networks for mining the temporal characteristics of network flows. Additionally, it employs a channel attention mechanism [15] to filter out irrelevant and unnecessary noise that is not pertinent to the task. The primary contributions of this work are as follows:

- We introduce a novel method for spatiotemporal feature extraction that integrates dilated convolution with a gating mechanism. This method initially employs dilated convolution [14] and the gating mechanism for preliminary feature extraction from network flows, followed by a secondary extraction and learning of features using Long Short-Term Memory networks. Incorporating a channel attention mechanism serves to avoid irrelevant noise, thereby effectively integrating spatio-temporal features. This approach provides a new perspective for efficiently processing and analyzing complex network traffic data.
- Our model adopts an innovative, collaborative processing mechanism compared to traditional network traffic classification methods. Conventional approaches typically focus on a single aspect of temporal or spatial features, neglecting other network flow dimensions. Our process, by extracting features from multiple sizes, ensures a more comprehensive data analysis, thereby enhancing the accuracy and efficiency of classification.
- To ascertain the efficacy of our proposed model, we conducted experiments on the ISCX VPN-nonVPN [16] public dataset, comparing our approach against several existing methodologies. The experimental results demonstrate that the method proposed in this paper achieves an accuracy rate of over 95% and a recall rate of over 90%, significantly superior to other existing methods. However, in the experiments, the imbalanced distribution of categories in the dataset resulted in differences between the overall experimental results of accuracy, precision, recall, and F1 score. The results highlight the superiority of our model in network traffic classification, demonstrating higher accuracy and performance compared to other methods, thereby affirming its effectiveness and distinction.

## 2. Related Works

### *2.1. Network Traffic Classification Methods*

Since the development of the Internet, a large amount of research has been conducted on network traffic classification methods in academia and industry [17], and considerable progress has been made. According to the different technologies used, general network traffic classification methods can be divided into four categories: traffic classification methods based on port numbers, traffic classification methods based on payload, statistics-based, and behavior-based.

The method of traffic classification based on port numbers involves identifying the type of service or application by recognizing the port numbers in the data packets [18]. Its advantage lies in different network services and applications typically use specific port numbers. This method does not require extensive feature extraction or complex computations; it simply involves detecting unique port number information to accomplish classification. However, this method of traffic identification based on port numbers is relatively straightforward. With the increasing number of applications and the adoption of techniques such as port obfuscation, the accuracy of this method has been affected, compared to the early days of the Internet when applications and network services used fixed port numbers. This method is suitable for early Internet usage and certain specialized areas of network security.

Traffic classification based on payload involves delving into the content of packets by examining the entire packet content [19], including headers and payloads, to identify specific strings or string patterns predefined within the packets. Unlike port-based methods, payload-based methods analyze particular data generated by network services or applications that remain unaltered. Therefore, this method can provide higher accuracy and detection robustness. However, it can only identify known non-encrypted traffic and may struggle when different network services or application payloads are similar. This method is suitable for simple, known network traffic identification and classification.

Statistical traffic classification relies on the statistical characteristics of network traffic to perform classification tasks [20]. Compared to payload-based methods, this approach quickly extracts packet length, timestamps, transmission directions, and other information from packet headers based on statistical features of network traffic without the need for deep packet payload analysis. Therefore, it has lower computational complexity. However, relying solely on feature data renders this method less suitable for processing today's vast amount of data. This method is ideal for known, simple network traffic identification and classification.

Traffic classification based on host behavior analyzes host behavior patterns at the transport layer to perform traffic classification [21]. Its advantage lies in identifying tasks by the communication protocols, port numbers used by the host, and the unique behavior patterns of different applications. Consequently, it can reveal deeper characteristics of host communication behavior, enabling more precise traffic classification. However, this approach is challenging to adapt to the heterogeneous network environment brought about by the explosive growth of mobile terminals and IoT devices. It is suitable for early and specific network traffic identification and classification task scenarios.

### *2.2. Network Traffic Classification Method Based on Deep Learning*

Deep learning is a machine learning technique based on the concept of representation learning, typically implemented through deep neural networks [22]. Its core lies in the ability to learn multi-level features progressively from raw data, thereby obtaining high-level feature representations, which can subsequently be utilized for complex tasks such as classification. A key advantage of deep learning is its end-to-end application approach, eliminating the need for manual feature design and extraction. Neural networks can autonomously learn and output high-level features directly from raw data. This advantage has led to the widespread application of deep learning in domains where feature design is particularly challenging, achieving remarkable results.

With advanced deep learning technology, various models have demonstrated significant performance improvements in network traffic classification. Specifically, the "Deep Packet" model proposed by Lotfollahi et al. [23] effectively handles network traffic classification and identifies user applications, making it the first traffic classification system to utilize deep learning algorithms. However, this method has limitations as it fails to explore the deep-level network traffic features. Wang et al. [24] introduced an end-to-end encrypted traffic classification method based on 1D-CNN, achieving an accuracy of 86.6% on the ISCX-VPN dataset. Nevertheless, this approach relies solely on capturing byte features for automatic traffic feature extraction, overlooking the importance of temporal features in network traffic. The Flowpic model proposed by Shapira et al. [25] converts primary traffic data into images and employs CNN for image classification. However, Flowpic requires prolonged traffic capture, making it less suitable for scenarios with limited computational resources and time-sensitive requirements. Tong et al. [26] employed LSTM for encrypted traffic classification, achieving a 91% accuracy rate in classification tasks based on the ISCX-VPN dataset. However, this method does not consider utilizing spatial features and solely focuses on temporal features.

### *2.3. Network Traffic Classification in Edge Intelligence*

With the continuous increase in the number of Internet of Things (IoT) and mobile devices, the capability for immediate data processing has become a crucial need. Edge computing enhances the efficiency and speed of data processing by shifting tasks from the central nodes of a network to edge nodes closer to the data sources [27]. This transition significantly reduces data transmission latency, as user data no longer need to travel long distances to remote servers, enabling real-time local data analysis.

Edge intelligent gateways are vital in optimizing data flow and service transmission. They manage data streams from the cloud center to devices and handle information flow from devices back to the cloud center. These gateways reduce reliance on network central nodes through localized data processing, making data analysis and services more efficient. Additionally, edge intelligent gateways can directly control terminal nodes connected to them, such as smart home devices, industrial sensors, or vehicular systems [28]. However, due to edge nodes' distributed and dynamic nature, network conditions can vary depending on location, time, device type, or user behavior. Therefore, edge gateways must possess high adaptability and intelligent decision-making capabilities to adjust real-time network configurations to accommodate these fluctuations [29].

As network security threats rise, the classification and analysis of encrypted network traffic in edge intelligence become particularly crucial. Encryption protocols, while securing data and user privacy, also challenge network monitoring and encrypted traffic detection. Traditional content-based detection methods are ineffective on encrypted traffic, so traffic classification in edge computing environments must employ more advanced techniques. These methods need not only to respond more quickly to security threats but also to identify potential attacks in the early stages of data processing, further reducing the protection needs of network central nodes.

The uniqueness of edge computing compared to other networks lies in several aspects. In edge computing scenarios, edge gateways typically require frequent communication requests and lengthy model-building times for encrypted traffic detection. This poses challenges for resource-constrained edge intelligent gateways. Therefore, edge intelligence technology addresses this issue by coordinating between terminal devices and edge servers, integrating the advantages of local computation and high computation, thus reducing the energy consumption of deep learning model establishment and inference in edge computing environments. This, in turn, resolves the problem of inefficient classification and accurate identification of encrypted traffic on the edge.

The encrypted traffic detection model based on edge intelligence is illustrated in Figure 1. Edge intelligence network devices receive heterogeneous traffic from the network, and then high-performance aggregation nodes (LOT gateways) are used to capture essential information from the encrypted network traffic data to update and aggregate the model. Each IoT gateway independently constructs its local machine-learning model. The global model can rapidly adapt to new data and environmental changes through this real-time, dynamic parameter-sharing mechanism.

In summary, manually extracting features for encrypted network traffic classification methods poses numerous challenges in specific scenarios, including complex traffic feature distributions, lengthy traffic collection times, and the significant consumption of cache and storage resources. Additionally, using single-feature extraction methods based solely on temporal or spatial features makes it difficult to fully extract network traffic characteristics and adapt to the current complex network environment. In contrast, encrypted network traffic classification methods based on deep learning and employing hybrid feature extraction no longer rely on manual feature design. They can effectively identify encrypted and anonymous traffic, providing a more granular traffic identification mechanism. The application of edge computing further enhances its advantages by reducing reliance on centralized clouds, decreasing latency, and improving responsiveness and data processing efficiency. Additionally, local processing better protects privacy since sensitive data does not need to leave the local network.

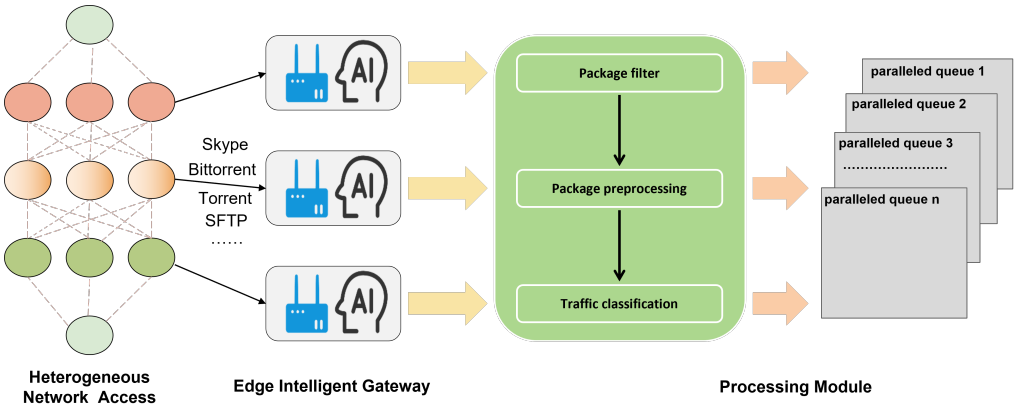



**Figure 1.** Traffic Classification Schemes in Edge Smart Gateways.

## 3. Model, Design, and Implementation

*Model Building*

To address the complex spatio-temporal characteristics of encrypted network traffic, we have designed a deep learning model architecture, as depicted in Figure 2, which consists of four key components. Algorithm 1 outlines the entire structure. (1) Preprocessing Stage: This stage involves transforming the raw network traffic data into a format that the model can process at any time. (2) Spatial Feature Extraction Stage: Preliminary screening and extraction of spatio-temporal feature information are performed through dilated convolution and gating mechanisms. This structural design lets the model focus on more important features, accurately capturing spatio-temporal characteristics. (3) Temporal Feature Extraction Stage: Further feature extraction and learning enhancement is achieved through CA-LSTM, effectively integrating temporal attributes. (4) Classification Stage: the model utilizes fully connected layers and a softmax layer to achieve the final classification results.

**Algorithm 1** The flow of the algorithm framework.

**Input**: Raw traffic data $D_t$, total number of epochs $t$, total number of batch $m$.
**Output**: Predicted category $\hat{Y}$ of traffic flows.

**procedure** DATA_PREPROCESSING($D_t$)
    Traffic data cleansing and conversion to IDX format files.
    The extraction of spatio-temporal features is accomplished through the GDC module.
    $V \leftarrow$ Obtain the vector containing the features.
    **return** $V$.
**end procedure**
**procedure** CA-LSTM MODEL($V$)
    **for** epoch $\leftarrow i$ **to** $t$ **do**
        **for** batch $\leftarrow j$ **to** $m$ **do**
            Transfer the vector $V$ obtained from the GDC module to the CA-LSTM.
            At each time step $t$, receive the current input vector $V_t$.
            Update the hidden state and cell state.
            Calculate and minimize loss function.
            Update all the parameters.
        **end for**
    **end for**
    **return** $\hat{Y}$.
**end procedure**

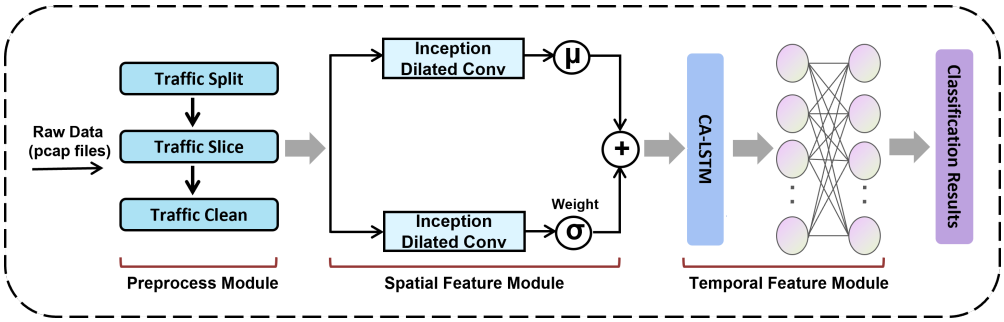



**Figure 2.** Module System Architecture.

Spatial Feature Extraction: Spatial feature extraction in traditional Convolutional Neural Networks (CNN) [30] typically relies on stacking convolutional layers. For deep neural networks, an intuitive method to enhance model performance is to increase the network's depth (i.e., the number of layers) and width (i.e., the number of neurons per layer). However, this approach rapidly increases the number of model parameters, which can cause overfitting and significantly raise computational complexity. Additionally, CNNs need a small receptive field during feature extraction, which limits the range of features the model can capture in each operation. Although increasing the convolutional layers can expand the receptive field, it also adds to the network's depth, potentially leading to the problem of vanishing gradients during backpropagation.

The Inception module [31] offers an innovative solution to address these challenges, as depicted in Figure 3. Unlike traditional CNN architectures, the Inception module incorporates multiple parallel paths, each capable of performing different operations (e.g., convolution operations of varying sizes). By utilizing filters of multiple sizes in parallel within the same layer, the Inception module is designed to provide multiple receptive field sizes at the same layer, facilitating feature extraction at different levels. An Inception module employs $1 \times 1$, $3 \times 3$, and $5 \times 5$ convolutional operations, allowing each module to capture information at different scales and offer a richer feature representation. We illustrate the above advantages through $1 \times 1$, $3 \times 3$, and $5 \times 5$ convolution operations. For a given input image $X$, the convolution operation can be expressed as Equation (1).

$$\begin{aligned} F_{1\times1}(X) &= X * W_{1\times1} + b_{1\times1} \\ F_{3\times3}(X) &= X * W_{3\times3} + b_{3\times3} \\ F_{5\times5}(X) &= X * W_{5\times5} + b_{5\times5} \end{aligned} \tag{1}$$

Among them, $*$ represents the convolution operation, $W_{1\times1}$,$W_{3\times3}$, and $W_{5\times5}$ represent the weights of the $1 \times 1$, $3 \times 3$, and $5 \times 5$ convolution kernels, respectively. $b_{1\times1}$, $b_{3\times3}$, and $b_{5\times5}$ are the bias terms corresponding to the $1 \times 1$, $3 \times 3$, and $5 \times 5$ convolution operations, respectively.

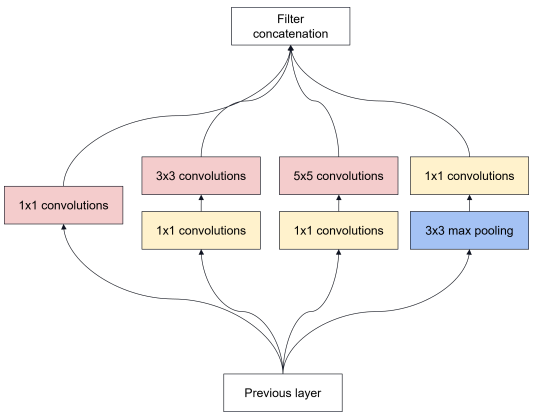



**Figure 3.** Inception Module.

The Inception architecture extracts features from different levels, allowing the network to capture a broader range of features. Additionally, using $1 \times 1$ convolutional kernels reduces the number of parameters in the network, thereby enhancing computational efficiency. This approach can reduce computational costs without sacrificing model performance, making deploying the network in resource-constrained environments easier. In summary, employing the Inception architecture enhances the model's symbolic power, better capturing complex structures and patterns in network data.

Based on the abovementioned content, we propose the Gated Inception Dilated Convolution (GDC) module for spatial feature extraction. This module replaces the conventional convolutional layers (Conv) with dilated convolutions within the Inception architecture. It achieves receptive fields of varying sizes by setting different dilation rates, thereby capturing multi-scale information and effectively supplementing global information, as shown in Figure 4. We will control the spatial range the convolution kernel covers by adjusting the expansion rate (Dilation Rate). Suppose there is a two-dimensional convolution operation whose input is $X$, the convolution kernel is $W$, and the expansion rate is $r$. Then, the expansion convolution completes the operation through Equation (2).

$$F(X) = (X *_r W)(i,j) = \sum_m \sum_n X(i + r \cdot m, j + r \cdot n) \cdot W(m,n) \tag{2}$$

Among them, $*_r$ represents the dilation convolution operation, $(i,j)$ represents the position of the output feature map, $(m,n)$ represents the position within the convolution kernel, and $r$ is the expansion rate, which determines the interval between the convolution kernel values. Increasing the dilation rate allows the convolution kernel to cover a larger input area, thus increasing the receptive field. For example, using a convolution kernel of size $3 \times 3$, a dilated convolution with a dilation rate of 2 will cover an input area of size $5 \times 5$, but the number of parameters of the convolution kernel remains unchanged. Inception's parallel structure helps prevent issues related to gradient vanishing or explosion. Recognizing that not all information in network flows is relevant for feature extraction, we integrate a gating mechanism within the GDC module to perform weighted information filtering. The GDC module employs a dual-branch structure. One branch processes input through an activation function to serve as part of the input for the subsequent stage, while the other branch inputs the processed data into a gating unit to generate a weight between 0 and 1. The filtered information from both branches is merged to serve as the input for the next phase.

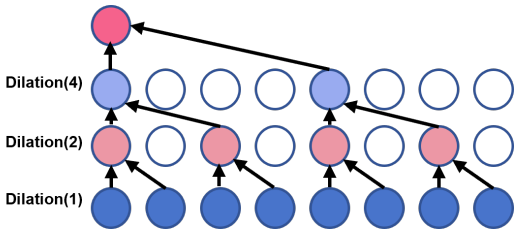



**Figure 4.** Dilated convolution diagram.

Gate units utilize gating mechanisms to selectively remember and forget information, enabling better capture of long-term dependencies within sequences. Their design is more flexible, accommodating sequences of varying lengths and structures while reducing the number of model parameters. This is highly advantageous for completing tasks in resource-constrained computational environments.

Initially, the Inception Dilated Conv architecture reads network traffic image data of size $28 \times 28 \times 1$ from IDX files, with the images undergoing normalization, where pixel values are scaled from the original range of [0, 255] to [0, 1]. As shown in Equation (3).

$$P_{\text{norm}}(i,j) = \frac{P_{\text{orig}}(i,j) - \min(P_{\text{orig}})}{\max(P_{\text{orig}}) - \min(P_{\text{orig}})} \tag{3}$$

For a general grayscale image, we set $\min(P_{\text{orig}})$ to 0 and $\max(P_{\text{orig}})$ to 255. $P_{\text{orig}}(i,j)$ is the pixel value located in row $i$ and column $j$, and $P_{\text{norm}}(i,j)$ is the normalized pixel value. We can simplify the above process as shown in Equation (4).

$$P_{\text{norm}}(i,j) = \frac{P_{\text{orig}}(i,j)}{255} \tag{4}$$

Subsequently, the model input is processed in parallel across multiple convolutional branches, each utilizing convolutional kernels of different sizes to capture feature information at various levels. After processing through two Inception Dilated Conv modules and a gating mechanism, the resulting features are transformed by a flattening layer, which unfolds the two-dimensional feature maps into a one-dimensional vector. Finally, the model outputs a one-dimensional vector V with 1240 elements, which contains the information after feature extraction and fusion.

Temporal Feature Extraction: Network traffic data exhibit distinct structured characteristics, with hierarchically organized elements such as bytes, packets, and flows. Within this structure, there is a significant difference between the correlations among bytes within and across packets. Adequate packets often represent a complete data exchange between the sender and receiver, and their temporal correlations should be extracted and analyzed separately. However, not all features are relevant and valuable for a specific task; some irrelevant features may introduce unnecessary noise, interfering with the model's judgment.

Long Short-Term Memory Networks (LSTMs), a particular Recurrent Neural Network (RNN), effectively address the vanishing and exploding gradients that traditional RNNs face when processing long sequence data. With their unique gating mechanisms, LSTMs can capture long-term dependencies in sequence data, making them particularly suitable for processing network traffic data with significant temporal characteristics. The Attention Mechanism offers an effective solution for unnecessary noise. It allows the model to dynamically allocate different attention weights to different parts of the input data, thereby highlighting more critical and relevant information and enhancing the model's accuracy without significantly increasing computational and storage burdens.

Building on the discussion above, we introduce Channel Attention, also known as Squeeze-and-Excitation networks. Its primary function is to adaptively learn the importance of each feature channel and accordingly weigh the features, enabling the model to focus on more crucial information. By integrating the Long Short-Term Memory (LSTM) network, which extracts temporal features, with the attention mechanism that avoids irrelevant noise, we propose the CA-LSTM module. LSTM specially designed three gate mechanisms: input, forget, and output. These gates work together to determine how the cell state is updated and how the unit output is calculated. Specifically, the input gate regulates the reception of new information, the forgetting gate determines which information in the cell state should be discarded, and the output gate controls the flow of information from the cell state to the unit output. The introduction of this gate control mechanism not only gives the LSTM network the ability to maintain stability when processing long sequence data but also makes it excellent at capturing long-term dependencies in sequence data, and is especially suitable for network traffic with time series characteristics data analysis.

We transfer the vector $V$ obtained through the GDC module to the CA-LSTM. At each time step $t$, we receive the current input vector $V_t$ and update the hidden and unit states.

Forget gate: The first step is to decide what information we want to discard from the cell state. The calculation of the forget gate is completed by Equation (5):

$$f_t = \sigma(W_f \cdot [h_{t-1}, V_t] + b_f) \tag{5}$$

Among them, $\sigma$ represents the *sigmoid* function, $W_f$ represents the weight matrix of the forgetting gate, $b_f$ is the bias term, $h_{t-1}$ is the hidden state of the previous time step, and $x_t$ is the input of the current time step.

Input gate: The second step determines what information we want to store in the cell state, and it consists of two parts: a $\sigma$ layer that determines which values we will update and a tanh layer that creates a new candidate value vector, specifically through Equations (6) and (7):

$$i_t = \sigma(W_i \cdot [h_{t-1}, V_t] + b_i) \tag{6}$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, V_t] + b_C) \tag{7}$$

Then, the cell state needs to be updated, and the previous state, $C_{t-1}$ is updated to $C_t$. Specifically, the new state is to pass the old state through the forgetting gate to complete the forgetting of part of the information and then add the new information stored in the input gate, specifically through Equations (8) :

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{8}$$

Output gate: Finally, the output value is determined through the output gate. The *sigma* layer is used to determine which parts of the cell state will be output. Then, the cell state is passed through *tanh* (obtaining a value between $-1$ and 1) and multiplied by the output of the *sigma* gate, and then the output is completed, specifically through Equations (9) and (10):

$$o_t = \sigma(W_o \cdot [h_{t-1}, V_t] + b_o) \tag{9}$$

$$h_t = o_t * \tanh(C_t) \tag{10}$$

The channel attention mechanism [32] is divided into two core steps: Squeeze and Excitation. The main goal of the compression step is to aggregate the global information of the input features to reduce the computational complexity while retaining critical information. The subsequent excitation step performs feature learning in the channel dimension. The learned weights emphasize essential channels' features and suppress unimportant ones. This can effectively assign different weights to the features of different channels, allowing the model to focus on more informative features, as illustrated in Figure 5.

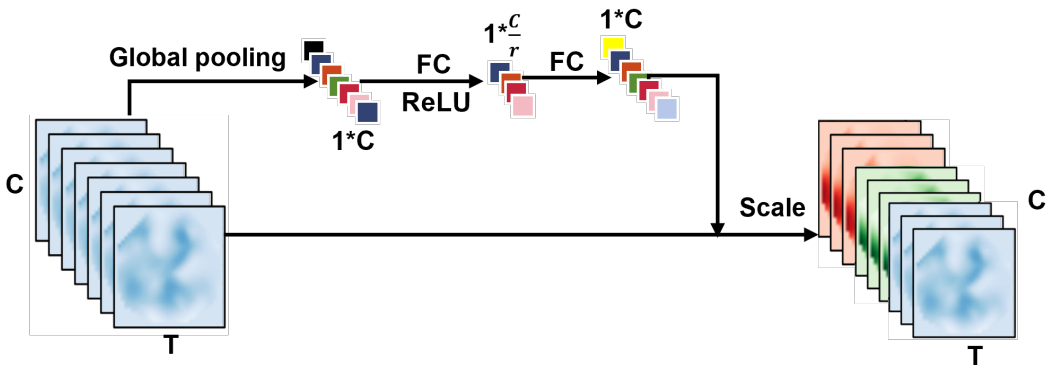



**Figure 5.** Diagram of the SENet module.

Let $F \in \mathbb{R}^{C \times H \times W}$ be the output feature map of lstm, where $C$ is the number of channels, and $H$ and $W$ are the height and width of the feature map, respectively. First, calculate the statistical characteristics of each channel through global average pooling, specifically through Equations (11):

$$F_c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} F(c, i, j) \tag{11}$$

where $F(c, i, j)$ represents the feature value of the $c$ channel at position $(i, j)$, and $F_c$ is the average pooling result of the $c$ channel. The weight of each channel is obtained through Equation (12).

$$W_c = \sigma(g(F_c)) = \sigma(W_U \cdot \delta(W_D \cdot F_c)) \tag{12}$$

We multiply the learned channel weight $W_c$ with the original feature map $F$ to obtain the weighted feature map $\tilde{F}$, as shown in Equation (13).

$$\tilde{F}(c, i, j) = W_c \cdot F(c, i, j) \tag{13}$$

Finally, we use the softmax layer to complete traffic classification.

The distribution of network traffic features is complex, especially in edge computing scenarios where storage and computational resources are limited. To better adapt to this environment, we employ the Inception structure for spatial feature extraction, which reduces the number of parameters in the network and improves computational efficiency. This approach lowers computational costs without sacrificing model performance, making deploying the network in resource-constrained environments easier. To filter out adequate information from network traffic, we introduce a gated units mechanism that better controls the flow of information, thereby reducing the unnecessary consumption of computational and storage resources. We utilize the LSTM structure and a channel attention mechanism for temporal feature extraction. This mechanism allows the attention mechanism to select important information from a large amount of data, assigning different weights to different parts of the input, mainly focusing on the temporal features of encrypted network traffic. In summary, employing the above structures can address the shortcomings of current methods and better adapt to various resource constraints in edge computing environments.

## 4. Evaluation

In this study, we conducted several multi-classification experiments on the ISCX VPN-nonVPN dataset to prevent the effects caused by chance. The results of the experiments, which are the averages of several runs, are used to evaluate the performance of the proposed model, and the experimental results are analyzed in depth.

### *4.1. Dataset*

Many studies on encrypted traffic classification depend on traffic data collected by security firms or private traffic data, which can impact the general trustworthiness of the research outcomes. At present, there is a relative scarcity of publicly available encrypted traffic datasets [33], and these datasets usually offer manually designed feature data rather than raw traffic data, with a limited range of categories covered. Considering our research task requires engaging with current mainstream applications. It demands a diverse and representative dataset from the real world, so we turned our attention to the ISCX VPN 2016 dataset [16] produced by the Canadian Institute for Cybersecurity. To generate a dataset representative of real-world traffic, they used real accounts to capture regular and VPN sessions through applications like Skype and Facebook, resulting in 14 traffic categories: VOIP, VPN-VOIP, P2P, VPN-P2P, etc. The dataset contains 14 types of encrypted traffic, with seven being regular encrypted traffic and the other seven being protocol-encapsulated traffic. This dataset includes the temporal flow feature data used in their research and provides raw traffic data (in Pcap format).

In the ISCX VPN-nonVPN dataset, the flow feature data are categorized into 14 classification labels. We also observed that the original traffic data did not come with labels, and there were ambiguities in the classification of some file types. For instance, certain file types could be categorized as 'Browser' or 'Streaming'. In light of this, we decided not to add additional annotations for files with ambiguous classifications. Following this revision, the variety of raw traffic data in the dataset was reduced to 12 types, including six types of regular encrypted traffic and six types of VPN protocol-encapsulated traffic. Table 1 describes what the dataset contains. The dataset shows that there is class imbalance

and potential influence from minority class data. Therefore, even if the model achieves a high level of accuracy, relying solely on accuracy can lead to misleading conclusions. In this scenario, precision, recall, and F1 score become particularly important for result analysis and discussion as they provide a more comprehensive reflection of the model's predictive ability for minority classes. In this context, "Traffic Type" is used to denote the service type of network traffic, "Applications" refers to the application type of network traffic, and "Percentage" indicates the proportion of the dataset.

**Table 1.** Dataset Categories and Contents.

| ISCX VPN-NonVPN Dataset | | |
|---|---|---|
| **Traffic Type** | **Applications** | **Percentage** |
| Chat | ICQ, AIM, Skype, Facebook, Hangouts | 3.94% |
| Email | SMPT, POP3, IMAP | 2.66% |
| FTP | Skype, FTPS, SFTP | 25.24% |
| Streaming | Viemo, Youtube, Netfilx, Spotify | 0.68% |
| VoIP | Facebook, Skype, Hangouts, Voipbuster | 60.59% |
| P2P | Torrent | 0.15% |
| VpnChat | ICQ, AIM, Skype, Facebook, Hangouts | 1.47% |
| VpnEmail | SMPT, POP3, IMAP | 0.11% |
| VpnFTP | Skype, FTPS, SFTP | 0.37% |
| VpnStreaming | Viemo, Youtube, Netfilx, Spotify | 0.24% |
| VpnVoIP | Facebook, Skype, Hangouts, Voipbuster | 4.37% |
| VpnP2P | Bittorrent | 0.17% |
| **TOTAL** | | **100**% |

### *4.2. Data Preprocessing*

The dataset requires preprocessing of the captured raw network traffic data in PCAP format to create a standardized dataset suitable for model input. The preprocessing process involves the following steps: traffic segmentation, traffic cleaning, packet truncation, and padding, vectorization, and normalization.

**Traffic Segmentation:** This step involves dividing the original network traffic data into multiple independent traffic flows. It segments the original traffic into individual flows by analyzing five-tuple information (source IP address, destination IP address, protocol type, source port number, and destination port number), saving each flow as a separate PCAP file.

**Traffic Cleaning:** Filter out empty and duplicate files from the segmented PCAP files to eliminate irrelevant noise. To prevent model overfitting, replace the MAC addresses in the data link layer and the IP addresses in the network layer with newly generated random addresses, as this information, dependent on the specific data collection environment, could interfere with the classification results. Subsequently, randomly allocate the data to the training and test sets in an 8:2 ratio.

**Packet Truncation and Padding:** Deep learning models usually require fixed-sized inputs. Therefore, this step processes each file to a fixed length of 784 bytes. It truncates files exceeding this length and pads files shorter than this length with zeros to reach the required fixed size.

**Conversion to IDX:** The images are transformed into IDX format files, which are commonly used in the field of deep learning.

### *4.3. Experimental Setup*

In order to ensure the reproducibility and validity of the experimental results, a standardized experimental setup was established; the hardware platform utilized for the experiments was equipped with a 12th generation Intel® Core™ i5-12500 processor with a base frequency of 3.00 GHz, an NVIDIA GeForce GTX 1060-6GB GPU, 16 GB of RAM, and was operating on Windows 11 OS. The model training used the Python program-

ming language and the PyTorch deep learning framework. The detailed experimental environment is shown in Table 2.

**Table 2.** Detailed Experimental Environment.

| Category | Parameters |
| --- | --- |
| GPU | Nvidia GPU (GeForce GTX 1060-6 GB) |
| Deep Learning Platform | Pytorch 1.10.2 |
| CUDA Version | 10.2 |
| CuDNN Version | 7.6.5 |

The main parameters of the model are as follows: the batch size is set to 500, and the number of epochs is fixed at 40. The Adam optimizer is selected with a learning rate (lr) of 0.00025.

### *4.4. Evaluation Metrics*

Commonly used indicators in encrypted traffic classification include accuracy rate, precision rate, recall rate, and F1 score. The calculation is shown in Equations (14)–(17). To train and test the model, 80% of the preprocessed traffic data is randomly selected as the training set and 20% as the test set. In the experimental evaluation, the model's performance is evaluated using the above key performance metrics.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{14}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{15}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{16}$$

$$F1\text{-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{17}$$

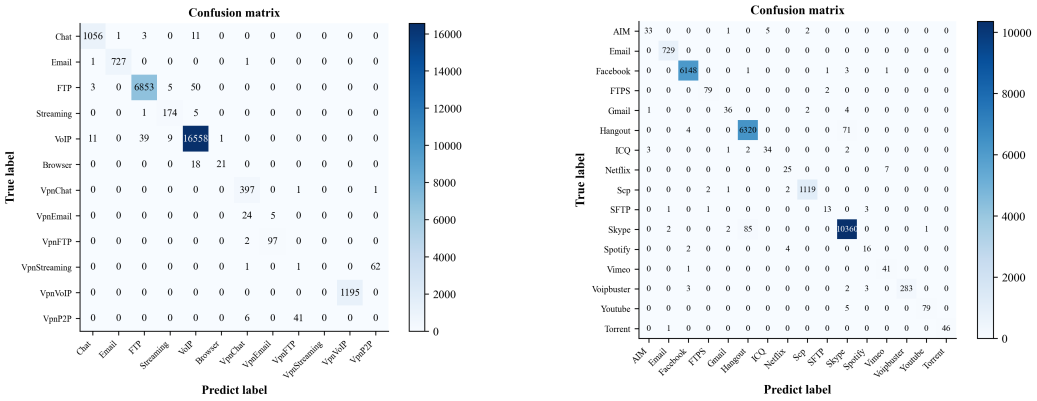
True Positive (TP) represents the number of target traffic correctly identified, False Positive (FP) represents the number of target traffic incorrectly identified, True Negative (TN) represents the number of other traffic correctly identified, and False Negative (FN) represents the number of target traffic that was missed.

### *4.5. Analysis of Results*

We conducted multiple classification experiments using the ISCX VPN-nonVPN dataset according to service programs (ISCX VPN-Service) and applications (ISCX VPN-App). The resulting confusion matrix of the classification is shown in Figure 6. As indicated by Figure 6a,b, most of the values are concentrated along the diagonal of the confusion matrix, which suggests that the model's overall classification performance is high.

The two datasets' overall accuracy, recall, precision, and F1 scores are shown in Figure 7, which provides a comprehensive view of the model's integrated performance. The illustration shows that the model exceeds 90% accuracy, recall, precision, and F1 scores on both datasets, with a false positive rate of less than 0.01%. This indicates that the model has good classification performance, achieving high accuracy while effectively addressing the issue of a high false alarm rate. At the same time, we observe some discrepancies between the overall experimental results of accuracy and the experimental results of accuracy, recall, precision, and F1 scores. By analyzing the confusion matrix, we attribute these discrepancies to the influence of the minority classes in the dataset, especially in datasets with an unbalanced class distribution, where relying solely on accuracy can be misleading. This is because even if the model only predicts the main class, it can still obtain

a very high accuracy rate. At this point, precision, recall, and F1 scores become particularly important because they better reflect the model's predictive ability for minority classes.




(**a**) ISCX-VPN-Service (**b**) ISCX-VPN-App

**Figure 6.** Multi-class confusion matrix results.

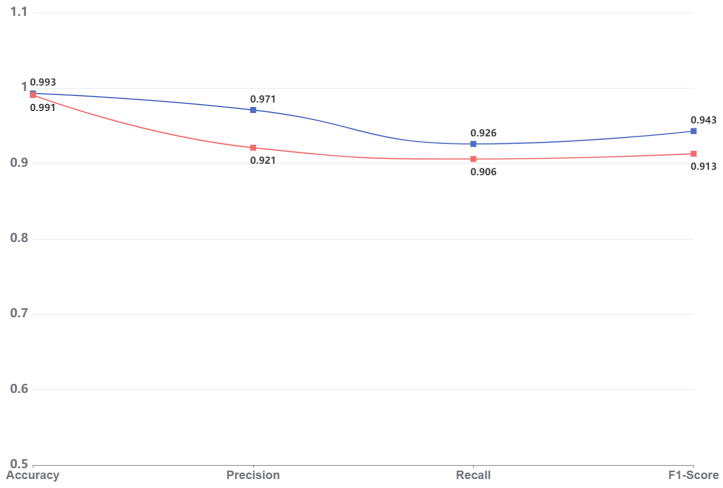


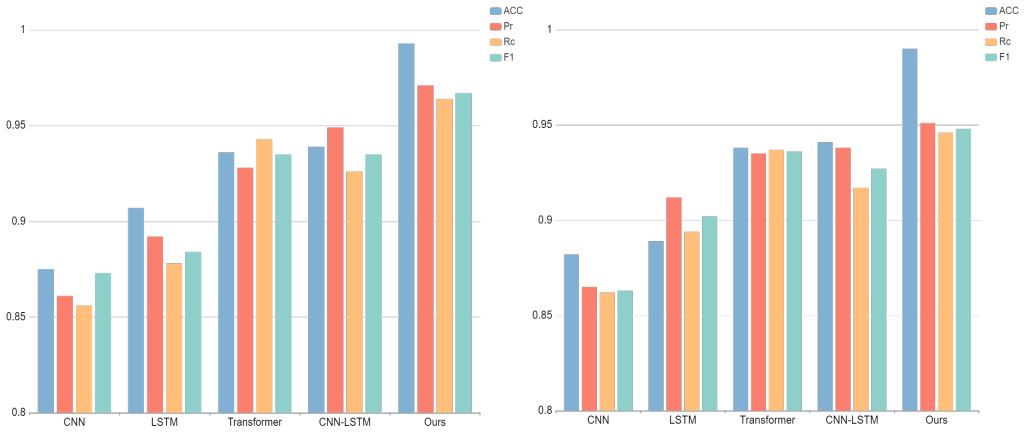

**Figure 7.** Experimental Performance.

Through the analysis above, the proposed model accomplishes identifying and classifying encrypted network traffic by extracting spatiotemporal features, making it adaptable to the current diverse encrypted and anonymous traffic. In contrast to conventional methods, our model achieves end-to-end automatic feature extraction, eliminating the reliance on traditional manual design. This enhances efficiency and enables adaptation to more complex network environments. Compared to other deep learning approaches, our model realizes encrypted network traffic classification through spatiotemporal feature extraction, eliminating the dependence on single time or space feature extraction. This enables more thorough feature extraction, thus improving the model's robustness.

### *4.6. Comparison with Existing Research*

To validate the effectiveness of the proposed model in this paper, we utilized the same ISCX VPN-nonVPN dataset and compared it with recent deep-learning models. As shown in Figure 8, our model outperforms other models by more than 2% in terms of accuracy, precision, recall, and F1 scores. It is noteworthy that in the multi-class experiments for the two categories, the classification performance of the hybrid model surpasses that of the individual models. This indicates that deep learning models considering temporal and spatial

feature extraction have a more decisive advantage in extracting heterogeneous encrypted network traffic features. This aligns with the research findings we mentioned earlier.




(**a**) ISCX-VPN-Service (**b**) ISCX-VPN-App

**Figure 8.** Compared to other DL Models.

## 5. Discussion

This paper proposes a model for encrypted network traffic classification. Our model also has some limitations. First, we should have considered the computational overhead and resource consumption caused by adding additional models. Secondly, the processing speed issue when receiving the impact of real-time network data needs to be considered. In the future, we will optimize the parameters in the model to reduce additional computing overhead and resource issues and try to add more network traffic classifications to test performance.

## 6. Conclusions

This paper presents a novel model for classifying encrypted network traffic to enhance traffic classification performance. We introduce an extraction module, Gated Dilated Convolution (GDC), which collaborates with dilated convolution and gating mechanisms to address the feature loss issue caused by previous single-feature extraction methods. Subsequently, we employ the CA-LSTM module to achieve secondary enhancement of features, thereby improving the detection efficiency and accuracy. We conduct two multi-classification experiments based on the ISCX-VPN dataset to validate the model's performance. Experimental results demonstrate that the proposed method achieves an accuracy rate of over 95% and a recall rate of over 90%, significantly outperforming existing methods, thus significantly enhancing the model's performance.

As network environments become increasingly complex in future work, we plan to improve the model's performance further, enhance its performance on imbalanced category distribution datasets, and apply it to network traffic classification tasks in more scenarios to improve the model's identification accuracy and robustness.

**Author Contributions:** Conceptualization—methodology, C.W.; validation—investigation, W.Z.; formal analysis—data curation, H.H.; writing—original draft preparation, H.S. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work is Supported by the Taishan Scholars Program under Grant TSQN202312230, the National Natural Science Foundation of Shandong Province under Grant ZR2022QF040, ZR2022LZH015, and ZR2023LZH011, the QLU Talent Research Project under grant 2023RCKY138.

**Data Availability Statement:** Data are contained within the article.

**Conflicts of Interest:** The authors declare that there are no conflicts of interest regarding the publication of this paper.

## References


1. Papadogiannaki, E.; Ioannidis, S. A survey on encrypted network traffic analysis applications, techniques, and countermeasures. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 123. [CrossRef]
2. Shen, M.; Ye, K.; Liu, X.; Zhu, L.; Kang, J.; Yu, S.; Li, Q.; Xu, K. Machine learning-powered encrypted network traffic analysis: A comprehensive survey. *IEEE Commun. Surv. Tutor.* **2022**, *25*, 791–824. [CrossRef]
3. Huang, Y.F.; Lin, C.B.; Chung, C.M.; Chen, C.M. Research on qos classification of network encrypted traffic behavior based on machine learning. *Electronics* **2021**, *10*, 1376. [CrossRef]
4. Cao, K.; Liu, Y.; Meng, G.; Sun, Q. An overview on edge computing research. *IEEE Access* **2020**, *8*, 85714–85728. [CrossRef]
5. Xu, D.; Li, T.; Li, Y.; Su, X.; Tarkoma, S.; Jiang, T.; Crowcroft, J.; Hui, P. Edge intelligence: Architectures, challenges, and applications. *arXiv* **2020**, arXiv:2003.12172.
6. Deng, S.; Zhao, H.; Fang, W.; Yin, J.; Dustdar, S.; Zomaya, A.Y. Edge intelligence: The confluence of edge computing and artificial intelligence. *IEEE Internet Things J.* **2020**, *7*, 7457–7469. [CrossRef]
7. Dong, S.; Wang, P.; Abbas, K. A survey on deep learning and its applications. *Comput. Sci. Rev.* **2021**, *40*, 100379. [CrossRef]
8. Lindemann, B.; Maschler, B.; Sahlab, N.; Weyrich, M. A survey on anomaly detection for technical systems using LSTM networks. *Comput. Ind.* **2021**, *131*, 103498. [CrossRef]
9. Salman, O.; Elhajj, I.H.; Kayssi, A.; Chehab, A. Data representation for CNN based internet traffic classification: A comparative study. *Multimed. Tools Appl.* **2021**, *80*, 16951–16977. [CrossRef]
10. Li, Z.; Liu, F.; Yang, W.; Peng, S.; Zhou, J. A survey of convolutional neural networks: Analysis, applications, and prospects. *IEEE Trans. Neural Networks Learn. Syst.* **2021**, *33*, 6999–7019. [CrossRef]
11. Popoola, S.I.; Adebisi, B.; Hammoudeh, M.; Gui, G.; Gacanin, H. Hybrid deep learning for botnet attack detection in the internet-of-things networks. *IEEE Internet Things J.* **2020**, *8*, 4944–4956. [CrossRef]
12. Lopez-Martin, M.; Carro, B.; Sanchez-Esguevillas, A.; Lloret, J. Network traffic classifier with convolutional and recurrent neural networks for Internet of Things. *IEEE Access* **2017**, *5*, 18042–18050. [CrossRef]
13. Lin, K.; Xu, X.; Gao, H. TSCRNN: A novel classification scheme of encrypted traffic based on flow spatiotemporal features for efficient management of IIoT. *Comput. Netw.* **2021**, *190*, 107974. [CrossRef]
14. Khan, A.; Sohail, A.; Zahoora, U.; Qureshi, A.S. A survey of the recent architectures of deep convolutional neural networks. *Artif. Intell. Rev.* **2020**, *53*, 5455–5516. [CrossRef]
15. Niu, Z.; Zhong, G.; Yu, H. A review on the attention mechanism of deep learning. *Neurocomputing* **2021**, *452*, 48–62. [CrossRef]
16. 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.
17. Zhao, J.; Jing, X.; Yan, Z.; Pedrycz, W. Network traffic classification for data fusion: A survey. *Inf. Fusion* **2021**, *72*, 22–47. [CrossRef]
18. Cheng, G.; Wang, S. Traffic classification based on port connection pattern. In Proceedings of the 2011 International Conference on Computer Science and Service System (CSSS), Nanjing, China, 27-29 June 2011; IEEE: Piscataway, NJ, USA, 2011; pp. 914–917.
19. Özdel, S.; Ateş, Ç.; Ateş, P.D.; Koca, M.; Anarım, E. Payload-based network traffic analysis for application classification and intrusion detection. In Proceedings of the 2022 30th European Signal Processing Conference (EUSIPCO), Belgrade, Serbia, 29 August–2 September 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 638–642.
20. Jenefa, A.; BalaSingh Moses, M. A multi-phased statistical learning based classification for network traffic. *J. Intell. Fuzzy Syst.* **2021**, *40*, 5139–5157. [CrossRef]
21. Azab, A.; Khasawneh, M.; Alrabaee, S.; Choo, K.K.R.; Sarsour, M. Network traffic classification: Techniques, datasets, and challenges. *Digit. Commun. Netw.* **2022**, *in press* [CrossRef]
22. Sharifani, K.; Amini, M. Machine learning and deep learning: A review of methods and applications. *World Inf. Technol. Eng. J.* **2023**, *10*, 3897–3904.
23. 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]
24. 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.
25. Shapira, T.; Shavitt, Y. Flowpic: Encrypted internet traffic classification is as easy as image recognition. In Proceedings of the IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Paris, France, 29 April–2 May 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 680–687.
26. Tong, X.; Tan, X.; Chen, L.; Yang, J.; Zheng, Q. BFSN: A novel method of encrypted traffic classification based on bidirectional flow sequence network. In Proceedings of the 2020 3rd International Conference on Hot Information-Centric Networking (HotICN), Hefei, China, 12–14 December 2020; IEEE: Piscataway, NJ, USA, 2020, pp. 160–165.
27. Hua, H.; Li, Y.; Wang, T.; Dong, N.; Li, W.; Cao, J. Edge computing with artificial intelligence: A machine learning perspective. *ACM Comput. Surv.* **2023**, *55*, 184. [CrossRef]
28. Barbuto, V.; Savaglio, C.; Chen, M.; Fortino, G. Disclosing edge intelligence: A systematic meta-survey. *Big Data Cogn. Comput.* **2023**, *7*, 44. [CrossRef]

29. Pujol, V.C.; Donta, P.K.; Morichetta, A.; Murturi, I.; Dustdar, S. Edge intelligence—research opportunities for distributed computing continuum systems. *IEEE Internet Comput.* **2023**, *27*, 53–74. [CrossRef]
30. Alzubaidi, L.; Zhang, J.; Humaidi, A.J.; Al-Dujaili, A.; Duan, Y.; Al-Shamma, O.; Santamaría, J.; Fadhel, M.A.; Al-Amidie, M.; Farhan, L. Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. *J. Big Data* **2021**, *8*, 53. [CrossRef] [PubMed]
31. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
32. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7132–7141.
33. Ring, M.; Wunderlich, S.; Scheuring, D.; Landes, D.; Hotho, A. A survey of network-based intrusion detection data sets. *Comput. Secur.* **2019**, *86*, 147–167. [CrossRef]