symmetry

MDPI

Article

Bidirectional Statistical Feature Extraction Based on Time Window for Tor Flow Classification

Hongping Yan [1], Liukun He [1,2], Xiangmei Song [1,*], Wang Yao [1], Chang Li [1] and Qiang Zhou [1]

[1] School of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang 212013, China
[2] Nanjing Huafei Data Technology Co., Ltd., Nanjing 210019, China
* Correspondence: jlsxm@ujs.edu.cn

Abstract: The anonymous system Tor uses an asymmetric algorithm to protect the content of communications, allowing criminals to conceal their identities and hide their tracks. This malicious usage brings serious security threats to public security and social stability. Statistical analysis of traffic flows can effectively identify and classify Tor flow. However, few features can be extracted from Tor traffic, which have a weak representational ability, making it challenging to combat cybercrime in real-time effectively. Extracting and utilizing more accurate features is the key point to improving the real-time detection performance of Tor traffic. In this paper, we design an efficient and real-time identification scheme for Tor traffic based on the time window method and bidirectional statistical characteristics. In this paper, we divide the network traffic by sliding the time window and then calculate the relative entropy of the flows in the time window to identify Tor traffic. We adopt a sequential pattern mining method to extract bidirectional statistical features and classify the application types in the Tor traffic. Finally, extensive experiments are carried out on the UNB public dataset (ISCXTor2016) to validate our proposal's effectiveness and real-time property. The experiment results show that the proposed method can detect Tor flow and classify Tor flow types with an accuracy of 93.5% and 91%, respectively, and the speed of processing and classifying a single flow is 0.05 s, which is superior to the state-of-the-art methods.

Keywords: slide window; statistical feature; Tor flow classification; application classification

Citation: Yan, H.; He, L.; Song, X.; Yao, W.; Li, C.; Zhou, Q. Bidirectional Statistical Feature Extraction Based on Time Window for Tor Flow Classification. *Symmetry* **2022**, *14*, 2002. https://doi.org/10.3390/sym14102002

Academic Editors: Haiqin Wu, Liangmin Wang and Keyang Cheng

Received: 3 September 2022
Accepted: 16 September 2022
Published: 24 September 2022

1. Introduction

Tor (The onion router) is a low-latency anonymous communication system developed by the U.S. Navy [1]. Since it is easy to deploy, Tor is widely used by many users all over the world. As of March 2022, global users of Tor exceeded 3 million. While Tor is designed to protect users' privacy, some criminals maliciously use it to publish rumors, sell contraband, and engage in other illegal activities. The Tor network utilizes the RSA asymmetric encryption algorithm to encrypt its traffic in layers and forward the information through multiple hops, which the criminal exploits to hide identity and obscure traces. Tor's abuse threatens to trace the source and obtain evidence of cybercrimes.

In the scene of actual network case investigation, we need to grasp the network activities of the target user in real-time, to detect whether there is Tor traffic in the current user traffic and identify its application. Tor traffic only accounts for a tiny proportion of network flow. Moreover, few features can be extracted from Tor flow with a weak representation ability, which reduces the efficiency of machine learning methods based on conventional statistical features. The recognition method based on deep learning [2–5] needs to obtain and calculate a large amount of network flow data for improving the performance of the detection model, and the algorithm is very complex, which consumes a lot of storage and computing resources. It cannot meet the real-time requirements of identifying Tor anonymous flow and classifying its application. Therefore, extracting and utilizing more concise features is one of the biggest challenges to improving the real-time detection performance of Tor traffic flow.

In this paper, we propose a real-time identification and classification method called bidirectional statistical feature [6,7] extraction based on the time window for Tor traffic identification and classification in real-time. By setting the time window and sliding step size, a six-tuple is used to represent the flow characteristics, which are calculated by the relative entropy of the flow probability distribution [8]. The six-tuple employed can depict the deviation degree of sequence in the current sliding window from the typical behavior profile. For the identified Tor flow, the bidirectional statistical feature extraction method based on the Apriori algorithm [9] is adopted, which is used for feature selection, thus, reducing the resource consumption in the Tor flow identification and classification process. The main contributions of this paper are summarized as follows:

(1) A Bidirectional Statistic Feature (BSF) extraction method based on the Apriori algorithm is designed, which can mine the deep correlation features concerning the interactive information of network flow and reduce the time consumption in feature extraction.

(2) An anonymous network flow discovery method of Tor flow based on a sliding time window is proposed; that is, by calculating the relative entropy of flow attributes in the time window, the storage consumption for flow detection is reduced.

(3) A real-time Tor anonymous flow detection and application classification model is proposed based on a sliding time window and bidirectional statistical characteristics. The proposed method's real-time performance and effectiveness are verified by conducting experiments on the UNB public data set (ISCXTor2016).

The rest of this article is as follows. In Section 2, the related work of Tor flow identification and application classification is introduced. In Section 3, the difference between Tor flow characteristics and normal flow and the sliding time window is defined. In Section 4, the proposed Tor flow hierarchical identification and classification model are described in detail. In Section 5, the real-time performance and effectiveness of the model are evaluated by simulation experiments. Finally, this paper is summarized in Section 6.

2. Related Works

Due to the presence of vast and asymmetric real-time traffic at the central nodes of the network relative to ordinary nodes [10], the real-time monitoring and analysis of flow requires a very high cost for storage and computation. Therefore, the existing research has focused on identifying Tor flow using short stream segments, which help shorten the classification time and achieve the real-time detection effect.

Wang et al. [11] extracted each message load's maximum, minimum, and average entropy characteristics, time, and header characteristics and proposed two flow window strategies for machine learning training. The machine learning classification algorithm was used to identify the confusing flow, and the recognition rate of Obfs3 and Obfs4 was 97.2% and 97%, respectively. Draper et al. [12] hold that the stream duration is a valuable feature of the tracing protocol, and 30, 60, and 120 s were suitable for testing. The experimental results showed that the time window of 120 s was the most appropriate. Wagner et al. [13] suggested that the network flow tended to be stable when 5 min was used as the time window and 1 min was used as the sliding window, considering the network equipment delay.

When flow is segmented, the short time window can be quickly detected, but the small time window is sensitive to the change of flow, which leads to a significant fluctuation of the recognition rate. A long time window can stabilize the flow distribution and recognition rate, leading to a long detection delay. Therefore, the selection of time window length needs to be weighed according to specific identification methods.

Before classifying the applications in Tor traffic, we must first extract Tor flow from the original network traffic. Existing privacy protection technologies have made some efforts on feature disclosure. For example, Nicolazzo et al. [14] proposed a privacy-preserving method that can prevent the leakage of sensitive user information that may occur simply by examining the device's characteristics. Therefore, to protect the privacy of ordinary users from being leaked while keeping track of the network activities of potentially mali-

cious users, we focus on studying the side-channel features of the traffic. At the network layer, flow usually presents some unique statistical characteristics, such as network flow characteristics represented by idle time, average flow length, flow density, etc. Message characteristics are described by average message length, message interval time, etc., and application layer characteristics represented by the ratio of source–destination communication data in specific application scenarios. Almubayed et al. [15] collected Tor flow and standard flow, extracted 40 kinds of flow characteristics, such as the total number of bytes, the total number of messages, and the duration of each flow, and used naive Bayes and random forest algorithms to classify normal HTTP flow and Tor flow. Lashkari et al. [16] suggest that only the time-based features can characterize Tor traffic to some degree. They used the combination of Infogain and Ranker algorithms to filter out 14 time-based features to train the C4.5 model and obtained a precision and recall above 0.9. Wang et al. [17] extracted 79 time-related and non-time-related features, respectively, and then sorted the features by using the Ranker search method and selected the top n ($5 \leq n \leq 70$) statistical features to identify Tor flow, which could achieve an overall accuracy of over 96%. These studies have shown that many features can be used to detect Tor flow effectively. Still, most of them are unsuitable in the real-time monitoring environment because of their significant computation. This paper focuses on how to mine the features or patterns with small computation and strong representation ability, and existing methods have made some progress in pattern mining. For example, Bonifazi et al. [18] considered the similarity between patterns, proposed a content semantic network CS-Net for managing reviews, and measured the similarity of two networks by calculating the similarity of structural features between different networks. Cauteruccio et al. [19] focused on extracting patterns with high frequency and utility, which they used to construct three social networks. Experiments show that these three networks can be used to identify communities of users exchanging Not Safe For Work (NSFW) adult posts and comments, analyze groups of text patterns that frequently appear together in NSFW content, and extract virtual communities of users employing the same text patterns, respectively.

For Tor flow classification, apart from using conventional statistical features, several studies have utilized deep learning technology to extract features automatically. Such techniques are asymmetric because a limited number of resources (e.g., deep learning algorithms) can extract information from encrypted traffic generated by cryptographic systems implemented with a significant number of resources. Lan et al. [20] proposed a DarknetSec technology based on side-channel characteristics (i.e., flow sequence, flow statistical characteristics) and packet payload byte sequence. In their opinion, only relying on manual features will lose the internal relationship between the bytes in the same position, which will reduce the classification accuracy. Therefore, they designed two deep learning models to extract the deep features from the side-channel features and the original byte sequence and fused the two depth features. Finally, the accuracy of anonymous flow classification was as high as 92.22%. The Flowpic method [21,22] converted the original data packet's size sequence and direction sequence into a two-dimensional histogram, which can be input to train a simple convolutional neural network model for flow classification. Although the above technologies omit the process of manual feature extracting, it increases the design, training, and deployment of the deep model. Furthermore, more training data is required. For example, the Flowpic method takes at least 60 s of stream duration to generate a sample. In the real-time monitoring environment, designing a model with simple deployment and a small amount of data is also a problem to be considered.

Most researchers pay attention to improving the performance of Tor anonymous flow identification and classification but ignore the real-time requirement of flow detection in practical application scenarios. Therefore, this paper introduces a sliding time window to design a bidirectional statistical feature extraction method, which can fully mine the deep correlation features of the interactive information of the flow in the time window and identify and classify Tor flow with the most concise features.

3. Tor Flow Characteristics and Sliding Time Window

Selecting appropriate traffic characteristics is the key to traffic classification and identification. This paper mainly studies the data packet characteristics and data flow characteristics of Tor flow. Firstly, by analyzing the size and interval time of data packets, the typical characteristics of flow based on Tor protocol are obtained to identify Tor flow and non-Tor flow. Secondly, through the interaction between different applications and the server, the personality characteristics of various applications based on bidirectional flow are extracted, and different applications in Tor flow are classified. The statistical features based on packets and flows are more macroscopic and higher-order, which perform better in countermeasure analysis technology. In addition, the standard burst feature only considers one-way burst [23], while the flow in bidirectional flow is related due to the characteristics of the application design. Therefore, the statistical feature of the bidirectional flow used in this paper is an essential upper-level feature, which is more instructive for Tor flow classification.

3.1. Subsection Tor Flow Characteristics

In the Tor flow captured by the client, the destination IP of the data packet is the entrance node of Tor, which makes the previous method of using the destination IP as the feature classification invalid [24]. In addition, because of the encryption of the data packet, the effectiveness based on the payload characteristics of the data packet is low [25].

Characteristics based on packets in Tor flow identification include packet length, total number of packages, total number of bytes, average packet length, packet arrival time interval, and average arrival time interval [15,26–29], etc. Theoretically, the more feature dimensions there are, the more accurate the recognition results will be. However, in real-time stream detection, the excessive dimensions of flow features will increase the time for feature extraction and detection. Considering the research purpose, this paper selects Tor flow features with high discrimination and low eigenvalue calculation.

(1) The characteristic of packet length. There are apparent differences in the packet length distribution of Tor flow compared with other flows. This paper analyzes the reasons for the distribution demonstrated in Figure 1 by examining the protocol resolution of Tor.

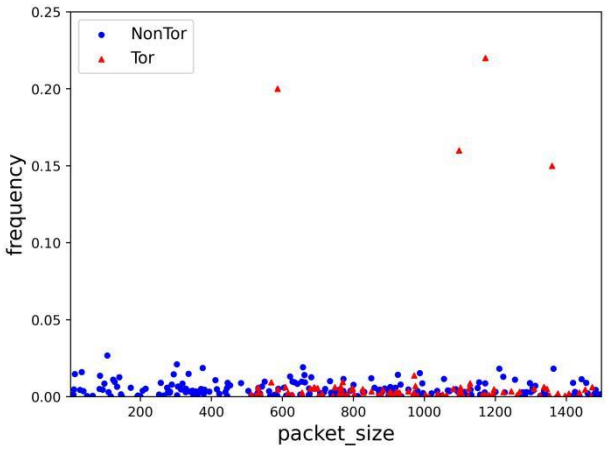


Figure 1. Packet length distribution of Tor and non-Tor flows.

To prevent message length-based attacks, the Tor anonymous communication system fixes the Tor cell length in the application layer. The cell is the smallest data unit in the Tor protocol. There are two types of data packets in the Tor protocol: fixed-length cell and indefinite-length cell. The fixed-length cell is mainly used for link establishment, extended destruction, data transmission, etc. The indefinite cell is primarily used for the Tor handshake protocol. The length of the fixed-length cell is 512 bytes, including 2 bytes in the link ID field, 1 byte in the data instruction field, and 509 bytes in the payload.

To ensure link security, Tor employs TLS encrypted links. TLS uses asymmetric encryption to authenticate Tor entities and uses symmetric session keys as encryption keys for data transfer between entities. Tor cells are generated and then encrypted by TLS, as shown in Table 1. Then, the TLS message goes through TCP encapsulation, the TCP can send a buffer containing one or more TLS messages, and each TLS message includes one or more Tor cell messages. These TLS messages are sent through one or more TCP messages. Assuming that there are k TLS messages in the TCP sending buffer and m Tor cell messages in the k TLS messages, the total length of data in the TCP sending buffer is $512 \times m + (5 + 20 + 12) \times 2 \times k = 512 \times m + 74 \times k$. Thus, the length of TCP packets in Tor is 1360, 1172, 1098, and 586 accounting for the vast majority [26]. Therefore, the distribution of TCP packet length can be used to identify Tor flow. Furthermore, the calculation of this feature is simple and convenient, which can meet the demand for rapid extraction in real-time detection.

Table 1. TLS message structure and Tor's TLS message structure.

1	**2**	**2**	***n***		**20**	**12**
Content types	Version	Length	Data		MAC	Pad
(**a**) TLS message structure						
5	**20**	**12**	**5**	**512 × *m***	**20**	**12**
TLS Record header	MAC	Padding length	TLS Record header	Cells	MAC	Pad
(**b**) Tor's TLS message structure						

(2) The characteristic of packet spacing. The time interval of data packets is an essential characteristic of network flow [19,30]. It is usually assumed that the time interval of data packets obeys independent Poisson distribution. The difference in the network environment or the use of encryption protocol [31] will significantly impact the time interval of data packets [32], and the arrival time interval of data packets is far from the exponential distribution. Since Tor is a low-latency anonymous communication system, it affects the time-related characteristics, such as the number of data packets and the arrival time of data packets, so these time-related flow statistics can be used as an effective way [33–35] to distinguish Tor flow.

The Tor network consists of three parts: Onion proxy (Op), Directory server (Ds), and Onion router (Or) [1]. Op mainly completes proxy work for Tor users, such as routing node selection, circuit establishment, and packet sending and receiving. In the process of establishing a connection on the Tor network, the Op randomly selects three available Ors as the Guard node (Gn), a Relay node (Rn), and Exit node (En) of the Tor network and uses the Diffie–Hellman encryption algorithm for them. When the key negotiation is finished, three-session keys are obtained, and then the message is encrypted in sequence. Finally, the Op sends the encrypted data packet to Gn three times. Gn, Rn, and En sequentially use the shared session key to decrypt the data packet and send it to the next hop, hence, En finally sends it to the target site in plaintext. On the contrary, during the return process, the data packet will pass through the three nodes Ex, Rn, and Gn successively and use the shared session key to encrypt it in turn, and finally, Gn will send the encrypted data packet to Op three times. The Op then decrypts them sequentially using the keys of the three shared sessions and sends the resulting plaintext to the Tor user.

According to the Tor data transmission process, it can be found that its data will be forwarded, encrypted, and decrypted by three nodes. Compared to other streams, access time in the same network environment has a relatively apparent delay compared with the characteristics of other streams. Considering the difference in the response time of different network links, the time interval is considered in this paper as the characteristic.

(3) Representation of data packets and data streams. To better characterize Tor anonymous flow, some definitions are given first.

Definition 1. *In this study, data packets are defined in the form of hexagrams p = sip,dip,sprt, dprt,time,len , where sip, dip, sprt, dprt, time, and len represent the source IP, destination IP, source port, destination port, the timestamp of the arrival of the data packet and the length of the data packet, respectively.*

A series of data packets with the same IP, port number, and transport protocol is called a unidirectional flow [36,37] in the unidirectional communication direction. A bidirectional flow consists of two unidirectional flows with source IP and destination IP in opposite directions.

Bidirectional flow can ensure data integrity and interactive relationship. Therefore, data packets of the same source and destination IP can be combined into a unidirectional data flow. Then, the data packets of two unidirectional flows in the same session can be integrated into a bidirectional data flow, denoted as $flow = \{flow_{in}, flow_{out}\}$. The detailed process of merging unidirectional flows into bidirectional flows is illustrated in Figure 2.

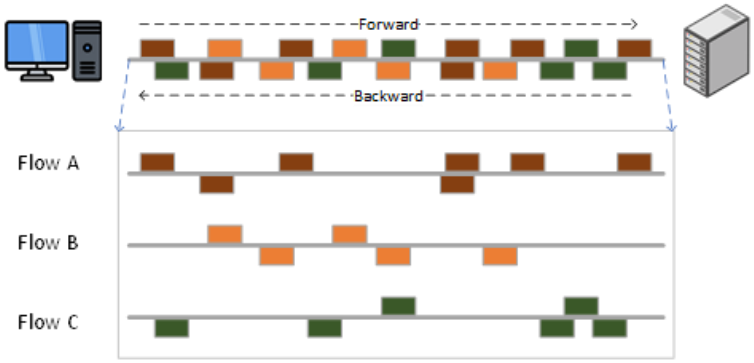


Figure 2. Schematic diagram of bidirectional flow.

Definition 2. *According to Definition 1, the data packet is defined as p, and the network flow is represented as* $F = \{p_1, p_2, p_3, \ldots, p_n\}$*, where n represents the total number of data packets in a bidirectional flow.*

Further, in this paper, the flow vector is used as the primary feature extraction unit, and each flow vector is represented by the two dimensions of packet length and inter-arrival time. The in-stream packet length sequence feature is defined as: $F = \{len_1, len_2, len_3, \ldots, len_Z\}$, where len_Z is the length of the z-th packet in the stream, $1 \leq z \leq Z$. When $len_z > 0$, it represents the outgoing direction, and when $len_z < 0$ it means the incoming packet.

The time interval sequence characteristic of the arrival of two adjacent data packets in the flow is defined as: $F_{\Delta t} = \{\Delta t_1, \Delta t_2, \Delta t_3, \ldots, \Delta t_{z-1}\}$, where Δt_z is the arrival time interval between the z-th and z + 1-th packets (p_z and p_{z+1}) in flow F.

3.2. Bidirectional Statistical Features Based on Application Mode

The interaction statistical features between the applications and servers are essential in the Tor network. Burst is often used as a characteristic for identifying an encrypted flow. The burst feature is the length of the continuous flow sequence that has the same direction. For example, for the sequence (1, −1, −1, −1, 1), where 1 indicates the direction of the outgoing packet, −1 indicates the direction of the incoming packet, and the burst characteristic value is −3 (3 consecutive −1 s). The burst feature can express the burst characteristics of flow, but burst can only express unidirectional burst flow at a certain point in time. Burst cannot express the correlation between the request and response packets in the flow nor show the interaction characteristics between the application and the server.

The flow of each application has a unique pattern, which can be presented as a high-order logical feature so that the flow classification model can be used to classify different

applications [38]. For example, when a browser application [39] is used, a small amount of interaction is required between the user and the server to establish a TCP connection. When a web page is requested, the server will send an HTML file to the user, causing a large number of data packets to be sent to the user, while the browser parses the HTML file. After that, a small amount of data packets will be sent to the outside world again to request other resource data, such as corresponding pictures, videos, etc., which will then trigger the server to send a large number of data packets to the user again. While music playback, video playback, file download, and mail download are servers that send a large number of data packets to the user, the user only needs to send a request to the server once in a short period. In addition, voice calls and chat tools have prominent interactive characteristics, thus, the data packet transmission volume between the two users is similar. In contrast, the user sends more flow when uploading emails and files. Most of the internet traffic is asymmetric. However, P2P traffic is almost symmetrical. For a P2P host, it is both a server and a client, showing the following characteristics: there are both a large number of incoming connections and a large number of outgoing connections, and the upstream and downstream traffic are roughly symmetrical.

Therefore, the interaction pattern in the application traffic is an important traffic layer characteristic, which reflects the application design pattern and logic. Similar patterns will appear during transmission, and the distribution of packet sizes will be different for different applications.

This paper extracts the packet size and direction in different application flow to verify our inference, which will be converted into a shock graph to show the flow patterns of different applications. As shown in Figure 3, there are significant differences in the amplitude of oscillations across different applications. First, applications with high traffic flow have lower oscillation amplitudes than those with a small flow, indicating that most of the data packet sizes of the flow transmitted on the former are the same. Since the interval between application packets with a large flow is smaller, the transmission speed is faster and more stable. However, the interval between two data packets for small-flow applications is larger, and there is a greater probability that the TCP send buffer is not full. At this time, the size of the last transmitted data packet will have discrete values, which will cause this type of application to have a relatively large oscillation in the oscillation graph. Based on the shock graph, the packet size can provide some interaction rules between different types of Tor applications and servers. In the following analysis, the bidirectional packet size is considered as a key feature in classifying different types of Tor applications.

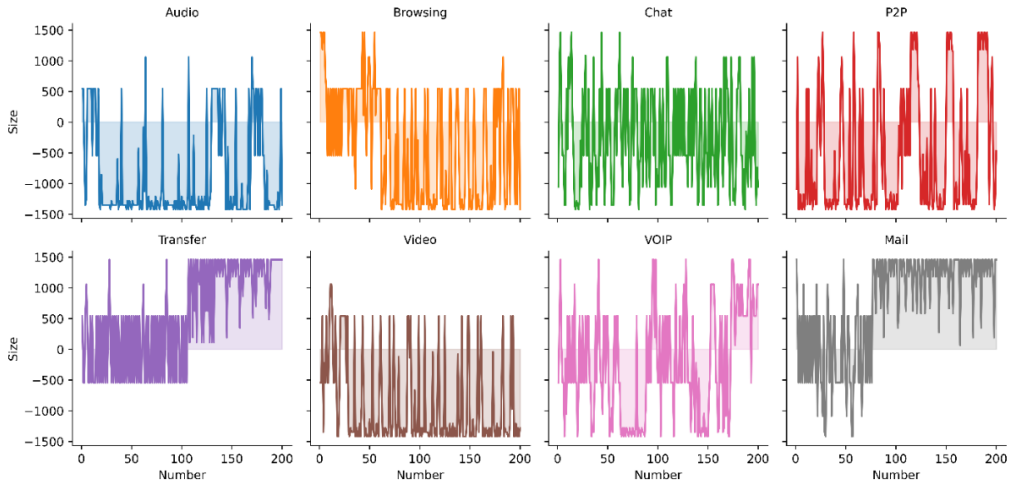


Figure 3. The oscillation diagram of the packet size of different types of applications.

3.3. Flow Time Window

The real-time flow is divided and intercepted according to the method of the sliding window model. The Tor flow is quickly detected in the divided flow. The sliding window only needs to save the flow data in one window, reducing memory and time consumption.

The time window W_t slides forward continuously with a fixed step Δt. If W_t is too large, it can easily cause false negatives and cause a considerable delay in detection. If W_t is too small, the influence of noise will be considerable, resulting in more false positives. At the same time, the size determines the frequency of new windows created by the detection system, and the smaller it is, the more windows there are. When $\Delta t < W_t$, adjacent windows will overlap, and one event will be assigned to multiple windows. When $\Delta t > W_t$, some events may be dropped. Therefore, determining the appropriate window size W_t and step size Δt is the primary task.

Let nt represent the current timestamp, the sliding window model only cares about W packets in the data stream (W is also called the size of the sliding window), and its query range is $\{a_{nt-W+1}, \ldots, a_{nt}\}$. As the data continues to arrive, the data in the window constantly shifts. The process of the time-based sliding window mechanism is as follows:

(1) The current window W_t acts on the data stream sequence received by the controller, the window contains multiple data streams, and the data packet of each data stream corresponds to a six-tuple feature.

(2) The number of each feature is counted in the current window, and the probability distribution of each feature is calculated based on the frequency and the size of the window.

(3) The entropy value of each feature is calculated in the current window according to the probability of features. The deviation degree is calculated between the sequence in the sliding window and the typical sequence.

(4) The sliding window backward is moved by Δt, and steps 1–3 are repeated.

4. The Key Technology of the Tor Flow Hierarchical Identification Model

4.1. Hierarchical Recognition Model

Different classification methods are used for Tor traffic classification tasks with different granularity. This paper adopts the flow hierarchical identification model shown in Figure 4 to perform serially Tor flow detection and classification tasks. Although the IPs of the Tor client and guard node do not change at the network layer, the routings from the user to the guard node and the guard node back to the Tor user are different. Therefore, there is an asymmetry in Tor routing, resulting in different traffic characteristics in the sending and receiving of directions. To reasonably characterize the asymmetric nature of internet routing, the model collects traffic in both directions.

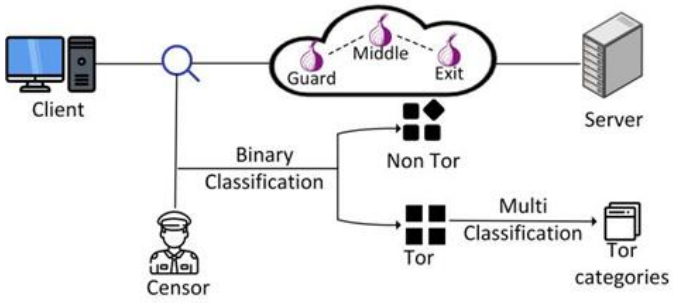


Figure 4. Hierarchical identification model.

The first layer is the Tor flow detection layer, and the specific detection steps are as follows.

(1) Capture flow based on time windows and split the captured network flow into data stream sets $flow = \{flow_1, flow_2, flow_3, \ldots, flow_U\}$.

(2) For each data flow $flow_u$ in the data flow set $flow$, call the Tor flow detection method based on the data packet characteristics to obtain $flow_u \in F_{Tor}$, $u \in [1, U]$.

The second layer builds bidirectional flow statistics features for Tor to identify applications in Tor flow.

(3) Collect various application flow generated by different types of applications when Tor is used:

$$\mathrm{F_{Tor}} = \{\{flow_{11}, flow_{12}, \ldots, flow_{1u}\}, \{flow_{21}, flow_{22}, \ldots, flow_{2u}\}, \ldots, \{flow_{k1}, flow_{k2}, \ldots, flow_{ku}\}\} \quad (1)$$

Construct a bidirectional flow and extract bidirectional statistical features using a frequent set [40,41] mining algorithm.

(4) The extracted bidirectional statistical features are used to build a classifier to output the application type of flow and, thus, $flow_u \in F_{Tor}$, $u \in [1, U]$ is obtained.

4.2. Identification of Tor Flow through Multi-Relative Entropy Joint Detection

In information theory, information entropy is defined as the probability of the occurrence of discrete random events, and information entropy is usually used to represent the measure of information uncertainty. Assuming a discrete random variable X, it has V possible values, namely $X \in \{x_1, x_2, x_3, \ldots, x_V\}$, x_v is the value of the discrete random variable X, and the probability of each value occurring is p_v, respectively. Then, the entropy of a discrete random variable X is defined as:

$$H(x) = -\sum_{v=1}^{V} p_v \log p_v \quad (2)$$

According to Section 3.1, Tor flow packet length and packet interval probability distribution are quite different from the normal flow. This paper adopts the relative entropy [42] method to quantify the difference between Tor and non-Tor flow. Relative entropy quantifies the distribution difference between the P and Q. Q represents the true distribution of the data, and P represents the theoretical or model distribution of the data. Assuming that $p(x)$ and $q(x)$ are two probability density functions of a random variable X, the difference can be expressed by relative entropy as:

$$D[p(x)\|q(x)] = \sum_{v=1}^{V} p(x) \log \frac{p(x)}{q(x)} \quad (3)$$

where, if and only if, $p(x) = q(x)$, $D[p(x)\|q(x)] = 0$.

For the relative entropy analysis of flow data in a time window, the number of features is set as M. The number of data packets corresponding to the M features are $x_1, x_2, x_3, \ldots, x_m$, respectively.

The proportion of the data packets of each feature to the total data packets is $p_1, p_2, p_3, \ldots, p_m$, where $p_m = x_m / \sum x_m (m = 1, 2, 3, \ldots, M)$ and $p_1, p_2, p_3, \ldots, p_m$ constitute an eigenvalue. After collecting network flow data of multiple time periods, eigenvalues of the period are respectively formed, and these eigenvalues are averaged to obtain a standard probability distribution P of Tor measurement statistics. In the detection process, the collected network flow is also divided into M items, and its characteristic value data is calculated to obtain $Q = \{q_1, q_2, q_3, \ldots, q_m\}$.

For K features, we calculate the relative entropy $D_1, D_2, D_3, \ldots, D_k$, respectively. After accumulation, the multi-dimensional relative entropy is calculated. With the pre-defined offset threshold L, if the calculation is $\sum_{k=1}^{k} D_k > L$, it indicates that the flow is Tor flow. The specific process is shown in Figure 5.

4.3. The Classification of Tor Application Flow Based on Bidirectional Flow Characteristics

The upper-layer logic and patterns of the application are expressed as a combination of flow requests and responses at the network layer, which can effectively distinguish flow categories. The orderliness of specific applications is used in network communication and

propose a flow classification method based on sequential pattern mining—the sequence features derived from packets.

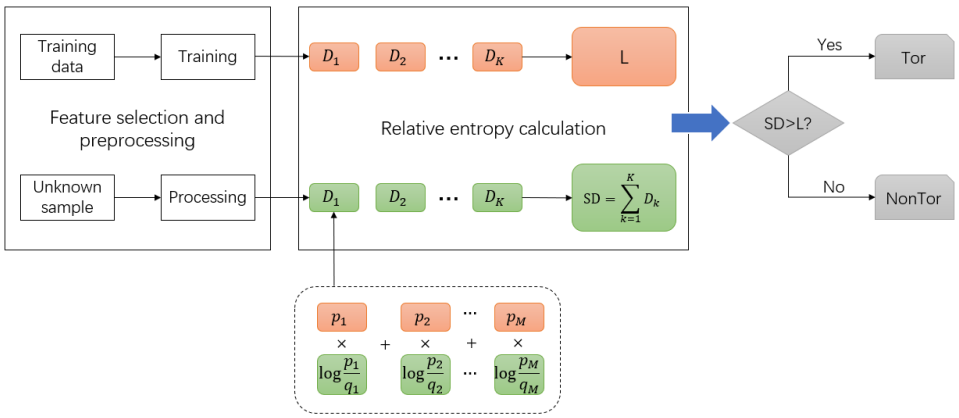


Figure 5. The process of using relative entropy joint detection to identify Tor flow.

According to this, the problem studied in this paper is described as follows. Given a session set of the same kind of bidirectional flow $flow = \{flow_1, flow_2, flow_3, \ldots, flow_R\}$, where the bidirectional statistical feature is $S = \{s_1, s_2, s_3, \ldots, s_O\}$, and s_o is a subset of $flow_r$, $o \in (1, O)$, $r \in (1, R)$, the solution of the problem is regarded as the process of generating the frequent set of a session set.

(1) Sequential pattern mining method

Definition 3. *K-sequences. A flow sequence is an ordered list of packets, for example, the sequence s_q is denoted as $s_q = \{p_1, p_2, p_3, \ldots, p_n\}$, which contains n packets in total, and each p_i is a packet. As the sequence length is the number of itemsets (or packets) it contains, if the length is K, it is recorded as a K-sequence.*

Definition 4. *If the all itemsets in sequence $a = \{a_1, a_2, a_3, \ldots, a_e\}$ is contained in $b = \{b_1, b_2, b_3, \ldots, b_f\}$, then a is said to be a subsequence of b, that is, there exists $e_1 < e_2 < e_3, \ldots, e_n$ such that a_{e_1} is included in b_{e_1}, a_{e_2} is included in b_{e_2}, ... , a_{e_n} is included in b_{e_n}, then a is a subsequence of b, or b contains a. For example, <(3) (4, 5) (8)> is enclosed in <(7) (3, 8) (9) (4, 5, 6) (8)>. Note: <(3) (5)> is not included in <(3, 5)>, because the former means that (3) and (5) appear respectively, and the latter means (3, 5) appear simultaneously.*

Definition 5. *The support of sequence s refers to the number (or percentage) of sequence s in all sequences.*

Definition 6. *The support of itemset i refers to the number of itemseti contained in all sequences. Therefore, itemset i and 1-sequence $< i >$ have the same support.*

In this paper, the support is employed here as criterion for judging frequent itemsets. The function of the Apriori algorithm is to find the maximum *K*-item frequent set. According to the flow pattern measures, such as the packet length, Apriori first extracts the set of packet length LIST1 in the whole traffic. Since application traffic is a combination of various upper-layer behaviors, based on this characteristic, the packet length of each flow record is extracted from LIST1 to form the set LIST2. Then, the support threshold is set to be q, and traverse LIST2 once to calculate the support of the candidate 1-itemset (each packet length), and then prune to remove the itemsets lower than q to obtain frequent 1-itemsets. Then, the frequent 1-itemsets are combined in pairs, output the frequent candidate 2-itemsets, remove the candidate 2-itemsets with support lower than q, output the frequent 2-itemsets, and so on, iterate until the frequent w + 1-itemsets cannot be found. The corresponding

set of frequent w-itemset is the output result of the algorithm. The j-th iterative process includes scanning to calculate the support of candidate frequent j itemsets, pruning to get the true frequent i itemsets, and connecting to generate candidate frequent $j+1$ itemsets.

Based on the above research, we propose a basic algorithm for mining BSF features based on the Apriori algorithm. Details are shown in Algorithm 1.

Algorithm 1: BSF Feature Extraction Algorithm Based on Frequent Set Mining

Input: Streaming data record LIST1, support threshold is q
Output: Two-way statistical feature K-itemset

```
(1) Apriori_permeation (LIST1, q)
(2) {
(3)       //Extract flow measure from LIST1 to LIST2
(4)       LIST2 ← LIST1
(5)       //find out frequent 1-itemsets
(6)       L1 = Find_frequent_1 (LIST2);
(7)       //find frequent k itemsets
(8)       for (i = 2; i < k; i++)
(9)       {
(10)          //Generate candidates and prune
(11)          Ci = apriori_gen (Li−1);
(12)          for each transaction t in Ct:
(13)             c.count ++;
(14)          Lk ={c∈C k | c.count ≥ q};
(15)       }
(16) }
(17) Apriori_gen (Lk−1)
(18){
(19)       for each itemset m1 in Lk−1
(20)             for each itemset m2 in Lk−1
(21)                if ((m1[1]! = m2[1])|| ... ||(m1[k−1]! = m2[k − 1]))
(22)                   {
(23)                      c = m1 ∪ m2;
(24)                      //If the subset c already exists in the k − 1 itemset, prune
(25)                      if (has_infrequent_subset (c, Lk−1))
(26)                         delete c;
(27)                      else
(28)                         add c to Ck;
(29)                   }
(30) }
```

5. Experimental Evaluation

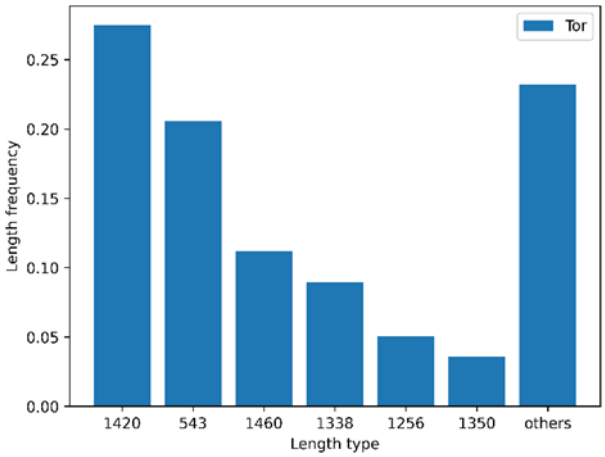
To verify the proposed method, this paper selects the UNB public data set ISCX-Tor2016 [16,43] as the experimental data. The author imitates the workstation and the gateway, respectively, through two virtual machines. Users operate different applications on the workstation to generate flow, and the workstation proxies the flow through the gateway to the Tor network. The author captures the normal encrypted flow generated at the exit of the workstation, judges the flow application label, and marks the Tor flow at the gateway exit. Finally, we collect eight categories of Tor flow dataset ISCX-Tor and common encrypted flow dataset ISCX-nonTor, and the eight categories are Browsing, Chat, Audio, Video, File Transfer, VoIP, VoIP, and Mail so that it can meet the experimental requirements of Tor flow discovery and application identification in Tor flow. According to the Tor flow packet length distribution research in Section 3.1 of this paper, ISCX Tor and ISCX non-Tor are labeled as dataset A (Tor) and dataset B (non-Tor), respectively.

The classification performance needs to be evaluated during the experiment. For the binary classification problem, each sample can be divided into true positive (TP), false positive (FP), true positive (TP), false positive (FP), true negative (TN), and false

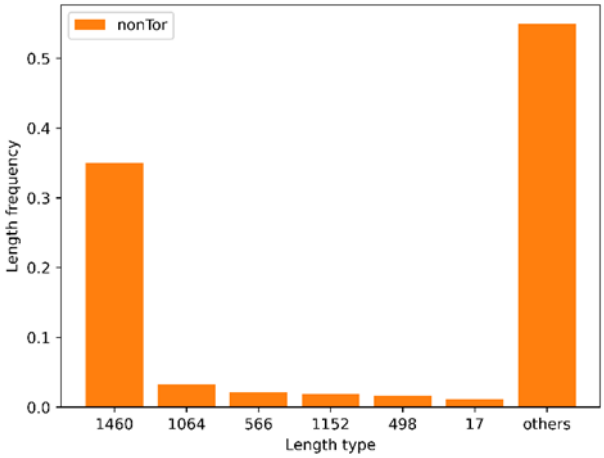
negative (FN). Based on the above concepts, this paper uses precision, recall, and accuracy to evaluate the performance of the proposed method, where precision = TP/(TP + FP), recall = TP/(TP + FN), and accuracy = (TP + TN)/(TP + TN +FP + FN).

5.1. Package Feature Probability Distribution

This section investigates the probability distribution of packet lengths for datasets A (Tor flow dataset) and B (Non-Tor flow dataset). First, the frequent multinomial sets of each data set are extracted based on the Apriori algorithm. The length value frequency is counted for the packet lengths that appear in the frequent sets, and the frequencies are sorted from the large to the small. Due to the limitation of chart space, this paper section only accurately calculates the probability of the top 6 packet lengths. The lengths of the packets ranked after the 6th are uniformly calculated according to other lengths. As seen from Figure 6, in dataset B, except for the first ranked packet lengths, in addition to the frequency exceeding 0.3, the packet lengths of ranks from 2–6 are all below 0.05 with little difference. There are many packets of other sizes, while in dataset A, the length of the top 6 packets occurs more frequently than packets of other sizes. It means that data packets are concentrated in a specific size in the Tor network. In contrast, the frequency of data packets of different sizes is relatively discrete in the non-Tor environment. Therefore, the features of the data packet can be converted into fixed-length intervals to improve computing speed.



(**A**)Tor



(**B**)non-Tor

Figure 6. Tor dataset and non-Tor dataset packet length distribution (seven types of data length and time distribution of dataset (**A**,**B**)).

Moreover, the frequencies of different types of flow within the fixed-length interval are significantly different. As shown in Figure 7, when the packet size is greater than or equal to 1057, the frequency of the Tor flow is considerably higher than that of the non-Tor flow. When the packet size is less than 1057, the non-Tor flow appears more frequently because the encapsulation strategy used by the Tor protocol makes most of the Tor flow packets by at least 543 bytes or more.

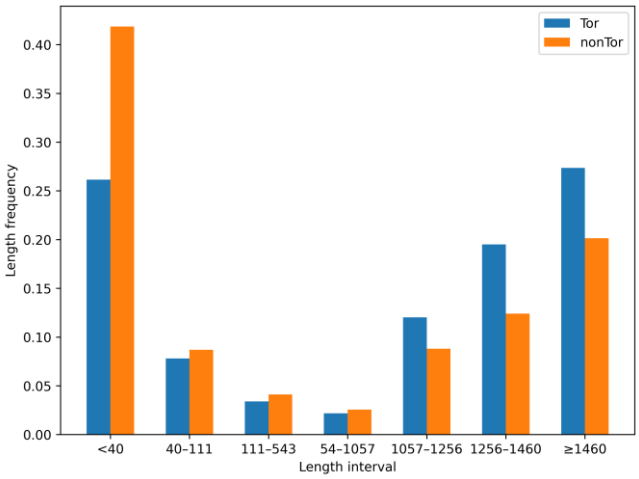


Figure 7. The frequency distribution of Tor dataset and non-Tor dataset packets within fixed-length intervals.

Second, the Tor communication link is constructed by three Tor nodes worldwide, and the number of nodes is limited. In order to prevent the congestion of the Tor network, Tor network node operators can customize the bandwidth that is provided according to their capabilities. Therefore, the time interval between two data packets in the Tor network is more significant than that in the non-Tor network. All packet intervals can be converted into a fixed time interval to improve the calculation speed. Moreover, the frequency of different types of flow within a fixed-time interval is slightly different. As demonstrated in Figure 8, when the packet interval is less than 0.75 ms, the proportion of non-Tor flow is higher than that of Tor flow. At the same time, the proportion of Tor flow in some intervals is higher than that of non-Tor flow when the packet interval is greater than or equal to 0.75 ms.

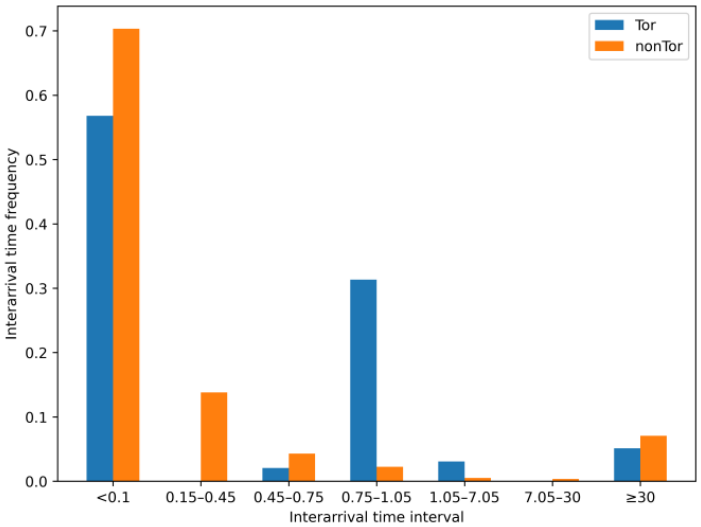


Figure 8. The frequency distribution of Tor dataset and non-Tor dataset packets in the fixed time interval.

5.2. Detection and Classification of Tor and Non-Tor Flow Based on Relative Entropy of Packet Features

In this paper, we use two measures: packet size and inter-arrival time. According to the analysis of the length of Tor flow data packets in Section 3.1, all the collected data packets are divided into 7 categories according to the length of the packets (<40, 40–111, 111–543, 543–1057, 1057–1256, 1256–1460, ≥1460) and also divided into 7 intervals according to the packet interval time (<0.15 ms, 0.15–0.45 ms, 0.45–0.75 ms, 0.75–1.05 ms, 1.05–7.05 ms, 7.05–30 ms, ≥30 ms). Using training set data calculation packet length measurement theory distribution P_1, packet interval time measurement theory distribution P_t, then calculate corresponding actual distribution Q_1, and Q_t on the test set, respectively. According to training data, a threshold value L is preset and multi-measure accumulation relative entropy D is compared; If $D < L$, the flow is Tor flow, otherwise it is a different flow.

According to Section 3.3, the time window W_t in the experimental network is set to 10, 20, 50, and 100 s, $\Delta t = \{W_t/20, W_t/10, W_t/5, W_t/2\}$ for comparative experiments. The final results are shown in Table 2. The initial effect of changing the window size is limited because the number of data packets distributed in the window is small at first, which is not enough to extract useful distribution information. As the window gradually increases, the calculated distribution value becomes larger. Close to the true value, when the window is increased to 20 s, continuing to introduce more data packets will not bring about changes in the distribution value, so the accuracy will not change. In addition, as the step size increases to a certain size of $W_t/10$, the detection accuracy shows a downward trend. Although the number of data packets will not change at this time, the number of overall samples will decrease, and the method is more overfitting, reducing the accuracy of detecting Tor flow. Finally, the detection effect is the best when the sliding window size is 20 s and the step size is $W_t/10$. At this time, the number of samples and distribution information are kept in a relatively stable state.

Table 2. Experimental results of different windows and step sizes.

	W_t = 10	W_t = 20	W_t = 50	W_t = 100
$\Delta t = W_t/20$	0.9533	0.9633	0.9628	0.9620
$\Delta t = W_t/10$	0.9540	0.9850	0.9711	0.9721
$\Delta t = W_t/5$	0.9515	0.9621	0.9623	0.9600
$\Delta t = W_t/2$	0.9518	0.9616	0.9517	0.9618

This paper uses datasets A and B to test the classification of Tor and non-Tor flow. It uses different similarity calculation standards such as relative entropy, Euclidean distance, and cosine similarity to conduct a classification experiment of Tor flow. It can be found from Table 3 that the identification accuracy of Tor flow is higher than that of the other two methods by calculating the similarity of relative entropy.

Table 3. Experimental results of using different calculation standards.

Method	Accuracy TP (%)	False Alarm Rate FP (%)
Relative entropy	93.50	6.50
Euclidean distance	89.33	10.67
Cosine similarity	90.33	9.67

Rao [25] and He [44] also used Tor's packet length as a feature. To collect enough non-cell data, the time to judge a data stream is approximately 43 s. The length of the time window can be reduced to 20 s through the time dimension, which effectively reduces the judgment time. At the same time, the relative entropy-based scheme is characterized by the distribution trend of the length of data packets in a fixed interval, which is more efficient than the simple method that used the top 90% data packet length ratio as a feature. Our method contains a broader range of data packets because low frequency is used, which can effectively reduce the false positive rate.

5.3. Tor Application Flow Classification Based on Bidirectional Flow Characteristics

According to the method in Section 4.2, the given data set A is firstly extracted according to different types of flow, and the session sets of two-way flow are firstly extracted, and then 80% of the data is selected as the training set, and the Apriori algorithm extracts the two-way statistical feature sets of different types of flow. The support degree is set to be 0.6, that is, the itemsets with the support degree greater than 0.6 are frequent candidate sets. In this paper, the longest candidate itemset is selected as the frequent set, as presented in Table 4, and then the remaining 20% of the data is used as the test set, and extract frequent bidirectional sets from it. It is worth noting that the Apriori algorithm requires the number of itemsets to be greater than 1, and the frequent itemset cannot be extracted when the number of itemsets is small. Therefore, this paper uses 15 candidate itemsets as one group; the frequent itemsets are taken from this group and compared with Table 4 to determine the type of flow. In addition, two factors, length and number of coincidences, need to be considered when comparing frequent sets, because the frequent sets of different types of flow have partial overlap.

Table 4. Bidirectional statistical feature mining results for different applications.

Type	Feature
Browsing	−543, 543, 1057
Chat	−543, 543
Audio	−1338, −1086, −543, −291, 543
Video	−1420, −1350, −1280, −1240, −1210, −80, −40, 543
Mail	−543, 198, 543, 1181, 1460
File Transfer	−1338, −1086, −543, 111, 543, 1460
P2P	−1420, −1338, −1256, −1174, −1092, 543, 1057, 1460
VoIP	−1350, −1086, −543, −279, 543

For example, the frequent set of Chat type is <−543, 543>, which is included in Browsing, Audio, File Transfer, VoIP, and Mail either, but the frequent set with a length of 2 only exists in the flow of this Chat type. Therefore, this paper uses $len\_cou = thre * count - (1 - thre) * lenc$ as the judgment indicator and selects the traffic type corresponding to the maximum value of len_cou as the prediction label, where $thre = 0.6$, the count represents the number of items intersected by the two frequent sets, and $lenc$ represents the length of the two frequent sets' difference. The identified confusion matrix and results are shown in Figure 9 and Table 5. The overall accuracy of the method is 91% for the test set. The method has a high classification accuracy for most traffic types except for the lower recall rate for Browsing; the possible reason is that the Browsing type flow includes other types of flow. Additionally, the frequent set calculated by this method has a high probability of overlapping with other types of flow (such as Mail and File flow), which results in a lower recall rate, and lower recognition accuracy of effective File Transfer and Mail.

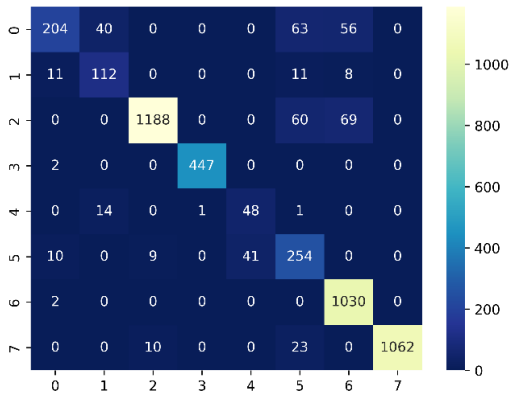


Figure 9. Confusion Matrix for Bidirectional Statistical Feature Recognition for Different Applications.

Table 5. Bidirectional statistical feature recognition results for different applications.

Flow Type	Precision (%)	Recall (%)
Browsing	89.1	56.2
Chat	67.5	78.8
Audio	98.4	90.2
Video	99.8	99.5
Mail	53.9	75
File Transfer	61.7	80.9
P2P	88.6	99.8
VoIP	100	96.9
Accuracy (%)	91	

5.4. Performance Comparison of Different Tor Flow Classification Models

In the final part of the evaluation of this paper, we analyze the classification performance and specific applicability of the three models. The three models are as follows: Lashkari et al. [16] trained a random forest using time-based features extracted from 15 s of Tor traffic, Shapira et al. [21] trained a CNN model from 2D histograms of packet size distributions extracted from 60 s of Tor traffic, and the classification model based on relative entropy proposed in this paper. All three models are trained under the ISCXTor dataset, and we use the first two models as representatives of machine learning methods and deep learning methods, respectively, to demonstrate the effectiveness and applicability of our method. The same training and testing sets are used, and the results are shown in Table 6. Except for Shapira et al., the accuracy rates obtained by other methods are above 80%. In particular, the method's accuracy in this paper reaches 91%, indicating that it can detect application traffic types with a lower error rate. Due to the fact that the method proposed here does not use any computationally intensive deep learning algorithms and extract complex features but only relies on the similarity between features, it takes less time to classify each flow, only 0.05 s, and is more applicable to real-time traffic detection scenarios compared to other methods. In addition, with sufficient computing resources, our method can also use other machine-to-learn models, such as KNN and CNN, to replace the similarity as a classifier, thereby further improving the classification performance.

Table 6. Performance of different Tor flow classification methods.

Method	Precision (%)	Recall (%)	Accuracy (%)	Classification Time (s/per Stream)
Lashkari et al. [16]	84.3	83.8	-	1
Shapira et al. [21]	-	-	67.8	0.1
This article	91	91	91	0.05

6. Conclusions

This paper proposes an efficient and real-time identification method for classifying Tor traffic and its application by dividing the Tor flow. The real-time and efficient classification results are achieved by designing a sliding window model to split the real-time flow and reorganizing it in the time window into a bidirectional flow.

Specific packet and payload information are no longer used as features to avoid long-time overhead and easy interference of traditional feature extraction methods. Relative entropy is used to measure traffic data in the two feature dimensions (i.e., packet length and time interval distribution) to classify Tor and non-Tor traffic. In the experiment, the time window method is used to divide the traffic, and the influence of different windows and steps on the traffic classification is determined through the experiment. Compared with other identification methods that do not use the time feature, this method effectively reduces the time to judge the type of data flow; this paper uses the distribution trend of the

packet length in the fixed interval as the feature, and does not filter the packets with low frequency, which can effectively reduce the false alarm rate.

The network-layer-specific traffic pattern based on the bidirectional flow is analyzed to classify the Tor application. The bidirectional statistical features are extracted by using the sequential pattern mining method in the time window containing the Tor traffic. This paper makes full use of the features of the protocol interaction, which is more interpretable than the deep learning method. In addition, the feature extracted in this paper is low complexity and high recognition efficiency compared with the deep learning method, which is more suitable for scenarios requiring high-speed classification.

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

Funding: This research was supported in part by the National Natural Science Foundation of China under Grant No. U1736216.

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

References

1. Dingledine, R.; Mathewson, N.; Syverson, P. *Tor: The Second-Generation Onion Router*; Naval Research Lab.: Washington, DC, USA, 2004.
2. Sirinam, P.; Imani, M.; Juarez, M.; Wright, M. Deep fingerprinting: Undermining website fingerprinting defenses with deep learning. In Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS), Toronto, ON, Canada, 15–19 October 2018; pp. 1928–1943.
3. Hardegen, C.; Pfülb, B.; Rieger, S.; Gepperth, A. Predicting network flow characteristics using deep learning and real-world network flow. *IEEE Trans. Netw. Serv. Manag.* **2020**, *17*, 2662–2676. [CrossRef]
4. Lotfollahi, M.; Jafari Siavoshani, M.; Shirali Hossein Zade, R.; Saberian, M. Deep packet: A novel approach for encrypted flow classification using deep learning. *Soft Comput.* **2020**, *24*, 1999–2012. [CrossRef]
5. Gu, Y. Research on network traffic classification based on machine learning and deep learning. *Telecommun. Sci.* **2021**, *37*, 105–113.
6. Saghezchi, F.B.; Mantas, G.; Violas, M.A.; Duarte, A.M.O.; Rodriguez, J. Machine learning for DDoS attack detection in industry 4.0 CPPSs. *Electronics* **2022**, *11*, 602. [CrossRef]
7. Qin, T.; Wang, L.; Liu, Z.L.; Guan, X.H. Robust application identification methods for P2P and VoIP flow classification in backbone networks. *Knowl.-Based Syst.* **2015**, *82*, 152–162. [CrossRef]
8. Cover, T.M.; Thomas, J.A. Entropy, relative entropy and mutual information. *Elem. Inf. Theory* **1991**, *2*, 12–13.
9. Hegland, M. The apriori algorithm—A tutorial. *Math. Comput. Imaging Sci. Inf. Process.* **2007**, 209–262. [CrossRef]
10. Zhang, J.; Yu, F.R.; Wang, S.; Huang, T.; Liu, Z.; Liu, Y. Load balancing in data center networks: A survey. *IEEE Commun. Surv. Tutorials* **2018**, *20*, 2324–2352. [CrossRef]
11. Wang, L.; Dyer, K.P.; Akella, A.; Ristenpart, T.; Shrimpton, T.; Assoc Comp, M. Seeing through Network-Protocol Obfuscation. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, 12–16 October 2015; pp. 57–69.
12. Draper-Gil, G.; Lashkari, A.H.; Mamun, M.S.I.; Ghorbani, A.A. Characterization of encrypted and vpn flow 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.
13. Wagner, A.; Plattner, B. Entropy based worm and anomaly detection in fast IP networks. In Proceedings of the 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise (WETICE'05), Linkoping, Sweden, 13–15 June 2005; pp. 172–177.
14. Nicolazzo, S.; Nocera, A.; Ursino, D.; Virgili, L. A privacy-preserving approach to prevent feature disclosure in an IoT scenario. *Future Gener. Comput. Syst.* **2020**, *105*, 502–519. [CrossRef]
15. Almubayed, A.; Hadi, A.; Atoum, J. A model for detecting tor encrypted flow using supervised machine learning. *Int. J. Inf. Secur.* **2015**, *7*, 10–23.
16. Lashkari, A.H.; Draper-Gil, G.; Mamun, M.S.I.; Ghorbani, A.A. Characterization of Tor flow using time based features. In Proceedings of the 3rd International Conference on Information Systems Security and Privacy (ICISSP), Porto, Portugal, 19–21 February 2017; pp. 253–262.
17. Wang, L.; Mei, H.; Sheng, V.S. Multilevel identification and classification analysis of Tor on mobile and PC platforms. *IEEE T. Ind. Inform.* **2020**, *17*, 1079–1088. [CrossRef]

18. Bonifazi, G.; Cauteruccio, F.; Corradini, E.; Marchetti, M.; Terracina, G.; Ursino, D.; Virgili, L. Representation, detection and usage of the content semantics of comments in a social platform. *J. Inf. Sci.* **2022**. [CrossRef]
19. Cauteruccio, F.; Corradini, E.; Terracina, G.; Ursino, D.; Virgili, L. Extraction and analysis of text patterns from NSFW adult content in Reddit. *Data Knowl. Eng.* **2022**, *138*, 101979. [CrossRef]
20. Lan, J.H.; Liu, X.D.; Li, B.; Li, Y.N.; Geng, T.T. DarknetSec: A novel self-attentive deep learning method for darknet flow classification and application identification. *Comput. Secur.* **2022**, *116*, 102663. [CrossRef]
21. Shapira, T.; Shavitt, Y. Flowpic: Encrypted internet flow classification is as easy as image recognition. In Proceedings of the IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops, Paris, France, 29 April–2 May 2019; pp. 680–687.
22. Okonkwo, Z.; Foo, E.; Li, Q.; Hou, Z. A CNN Based Encrypted Network Traffic Classifier. In *Australasian Computer Science Week 2022*; Association for Computing Machinery: New York, NY, USA, 2022; pp. 74–83.
23. Wang, T.; Goldberg, I.; Assoc, U. Walkie-Talkie: An Efficient Defense Against Passive Website Fingerprinting Attacks. In Proceedings of the 26th USENIX Security Symposium, Vancouver, BC, Canada, 16–18 August 2017; pp. 1375–1390.
24. Cuzzocrea, A.; Martinelli, F.; Mercaldo, F.; Vercelli, G. Tor flow analysis and detection via machine learning techniques. In Proceedings of the 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, USA, 11–14 December 2017; pp. 4474–4480.
25. Rao, Z.; Niu, W.; Zhang, X.; Li, H. Tor anonymous flow identification based on gravitational clustering. *Peer Peer Netw. Appl.* **2018**, *11*, 592–601. [CrossRef]
26. Petagna, E.; Laurenza, G.; Ciccotelli, C.; Querzoni, L. Peel the onion: Recognition of android apps behind the tor network. In Proceedings of the International Conference on Information Security Practice and Experience, Kuala Lumpur, Malaysia, 26–28 November 2019; Volume 11879, pp. 95–112.
27. Korczynski, M.; Duda, A. Markov chain fingerprinting to classify encrypted flow. In Proceedings of the IEEE INFOCOM 2014—IEEE Conference on Computer Communications, Toronto, ON, Canada, 27 April–2 May 2014; pp. 781–789.
28. Panchenko, A.; Lanze, F.; Pennekamp, J.; Engel, T.; Zinnen, A.; Henze, M.; Wehrle, K. Website Fingerprinting at Internet Scale. In Proceedings of the NDSS, San Diego, CA, USA, 21–24 February 2016.
29. Johnson, C.; Khadka, B.; Ruiz, E.; Halladay, J.; Doleck, T.; Basnet, R.B. Application of deep learning on the characterization of tor flow using time-based features. *J. Internet Serv. Inf. Secur.* **2021**, *11*, 44–63.
30. Lingyu, J.; Yang, L.; Bailing, W.; Hongri, L.; Guodong, X. A hierarchical classification approach for tor anonymous flow. In Proceedings of the 2017 IEEE 9th International Conference on Communication Software and Networks, Guangzhou, China, 6–8 May 2017; pp. 239–243.
31. Papadogiannaki, E.; Ioannidis, S. A survey on encrypted network traffic analysis applications, techniques, and countermeasures. *ACM Comput. Surv.* **2021**, *54*, 1–35. [CrossRef]
32. Wang, P.; Wang, Z.X.; Ye, F.; Chen, X.J. Bytesgan: A semi-supervised generative adversarial network for encrypted traffic classification in SDN edge gateway. *Comput. Netw.* **2021**, *200*, 108535. [CrossRef]
33. Li, M.; Han, D.Z.; Yin, X.M.; Liu, H.; Li, D. Design and implementation of an anomaly net-work traffic detection model integrating temporal and spatial features. *Secur. Commun. Netw.* **2021**, *2021*, 7045823.
34. Tian, Z.D.; Song, P.f. A novel network traffic combination prediction model. *Int. J. Commun. Syst.* **2022**, *35*, e5097. [CrossRef]
35. Lin, K.; Xu, X.; Gao, H. TSCRNN: A novel classification scheme of encrypted traffic based on flow spatio-temporal features for efficient management of IIoT. *Comput. Netw.* **2021**, *190*, 107974. [CrossRef]
36. Thijs, V.E.; Bortolameotti, R.; Continella, A.; Ren, J.J.; Dubois, D.J.; Lindorfer, M.; Choffnes, D.; Steen, M.V.; Peter, A. Flowprint:semi-supervised mobile-app fingerprinting on encrypted network traffic. In Proceedings of the 2020 Network and Distributed System Security Symposium, San Diego, CA, USA, 23–26 February 2020.
37. Towhid, M.S.; Shahriar, N. Encrypted network traffic classification using self-supervised learning. In Proceedings of the 2022 IEEE 8th International Conference on Network Softwarization (NetSoft), Milan, Italy, 27 June–1 July 2022; pp. 366–374.
38. Li, Y.; Huang, Y.; Seneviratne, S.; Thilakarathna, K.; Cheng, A.; Jourjon, G.; Webb, D.; Smith, D.B.; Xu, R.Y.D. From traffic classes to content: A hierarchical approach for encrypted traffic classification. *Comput. Netw.* **2022**, *212*, 109017. [CrossRef]
39. Špaček, S.; Velan, P.; Čeleda, P.; Tovarňák, D. Encrypted web traffic dataset: Event logs and packet traces. *Data Brief* **2022**, *42*, 108188. [CrossRef] [PubMed]
40. Nowakowski, P.; Zórawski, P.; Cabaj, K.; Mazurczyk, W. Detecting network covert channels using machine learning, data mining and hierarchical organisation of frequent sets. *J. Wirel. Mob. Netw. Ubiquitous Comput. Dependable Appl.* **2021**, *12*, 20–43.
41. Huo, Z.; Zhu, W.; Pei, P. Network traffic statistics method for resource-constrained industrial project group scheduling under big data. *Wirel. Commun. Mob. Comput.* **2021**, *2021*, 1–9. [CrossRef]
42. Gowtham Akshaya Kumaran, P.; Amritha, P.P. Real-time segregation of encrypted data using entropy. In Proceedings of the Congress on Intelligent Systems, Bengaluru, India, 4–5 September 2021; Springer: Singapore, 2022; pp. 835–844.
43. Zhao, Y.; Chen, J.; Wu, D.; Teng, J.; Yu, S. Multi-task network anomaly detection using federated learning. In Proceedings of the Tenth International Symposium on Information and Communication Technology, Ha Long Bay, Vietnam, 4–6 December 2019; ACM: New York, NY, USA, 2019; pp. 273–279.
44. He, G.F.; Yang, M.; Luo, J.Z.; Zhang, L. Online identification of tor anonymous communication traffic. *J. Softw.* **2014**, *24*, 540–546. [CrossRef]