IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 28 December 2023, accepted 9 January 2024, date of publication 19 January 2024, date of current version 8 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3356073

RESEARCH ARTICLE

Classifying Tor Traffic Encrypted Payload Using Machine Learning

PITPIMON CHOOROD[1], (Member, IEEE), GEORGE WEIR[1], (Senior Member, IEEE), AND ANIL FERNANDO[1,2]

[1]Department of Computer and Information Sciences, University of Strathclyde, G1 1XH Glasgow, U.K.
[2]Centre for Vision, Speech and Signal Processing, Department of Electrical and Electronic Engineering, University of Surrey, GU2 7XH Guildford, U.K.

Corresponding author: Anil Fernando (anil.fernando@surrey.ac.uk)

ABSTRACT Tor, a network offering Internet anonymity, presented both positive and potentially malicious applications, leading to the need for efficient Tor traffic monitoring. While most current traffic classification methods rely on flow-based features, these can be unreliable due to factors like asymmetric routing, and the use of multiple packets for feature computation can lead to processing delays. Recognising the multi-layered encryption of Tor compared to nonTor encrypted payloads, our study explored distinct patterns in their encrypted data. We introduced a novel method using Deep Packet Inspection and machine learning to differentiate between Tor and nonTor traffic based solely on encrypted payload. In the first strand of our research, we investigated hex character analysis of the Tor and nonTor encrypted payloads through statistical testing across 8 groups of application types. Remarkably, our investigation revealed a significant differentiation rate of 94.53% between Tor and nonTor traffic. In the second strand of our research, we aimed to distinguish Tor and nonTor traffic using machine learning, based on encrypted payload features. This proposed feature-based approach proved effective, as evidenced by our classification performance, which attained an average accuracy rate of 95.65% across these 8 groups of applications. Thereby, this study contributes to the efficient classification of Tor and nonTor traffic through features derived solely from a single encrypted payload packet, independent of its position in the traffic flow.

INDEX TERMS Network traffic classification, Tor network, machine learning, encrypted payload features, character analysis.

I. INTRODUCTION

Tor [1] is an anonymous network that offers a significant advantage in providing privacy to Tor users by concealing their identities. Tor achieves anonymity by routing its traffic through a series of relays within the Tor network, which is maintained and operated by volunteers worldwide. This complicated process makes it difficult to trace the origin of Tor traffic because the multi-layered process of relaying traffic conceals users' actual IP addresses before reaching their destination. This anonymity protects a wide range of individuals, from regular internet users who wish to evade ISP tracking to journalists and activists who seek a secure connection without revealing their identities. However, Tor's benefits also attract criminals for illicit activities such as accessing child pornography or conducting cyberattacks, without fear of detection. While Tor effectively shields users' identities, locations, and activities, it cannot entirely hide the network traffic generated during its usage.

To date, several methods have been proposed for Tor traffic classification. Many studies have emphasised flow-based features since Tor traffic has distinctive latency patterns [1], [2]. However, asymmetric routing could make such a method less reliable [3], and computing multiple packets for feature extraction can introduce processing delays. An alternative approach focuses on the dominant packet properties of Tor fixed-cell size to detect Tor traffic [4], but the effectiveness of this technique could be compromised with changes in the Tor configuration [5]. Our current study aims to address this gap by introducing a novel method that overcomes these limitations.

The associate editor coordinating the review of this manuscript and approving it for publication was Jose Saldana.

Interestingly, the unique characteristic of multi-layered encryption in Tor, which maximises its privacy compared to the single-layer encryption found in nonTor networks, has captured our attention. The main goal of encryption is to ensure that the encrypted data (ciphertext) remains secure, revealing no details of the original data (plaintext), even if adversaries have prior knowledge about encryption methods [6]. The encrypted data should provide no information on the original content. Therefore, the encrypted payloads from the two types of networks should appear identical and indistinguishable. A study from Choorod and Weir [7] explored this hypothesis and shed light on this matter. This leads us to investigate further, leading to our primary research question ***RQ1: Can we differentiate Tor from nonTor encrypted traffic based on their encrypted payloads?***

In addressing this question, previous studies investigated by [24], [25], and [26] highlighted the effectiveness of character analysis via statistical computational methods in text classification. These proven statistical methods are now adapted for application in the context of computer networks. In line with this, our analysis focuses on the frequency-related features of hexadecimal characters present within encrypted payloads without the need for decryption. Our Deep Packet Inspection (DPI) approach ensures privacy preservation. If our observations through the statistical technique indicate that we cannot differentiate between Tor and nonTor traffic based on their encrypted payloads, it suggests that the encrypted payloads in both networks appear identical. Conversely, discovering differentiation leads to a second line of investigation.

At this point, certain traditional methods, particularly those relying on flow-based features, require multiple packets spanning from the start to the end of a flow to compute relevant features. This dependency can sometimes lead to unexpected timeouts, causing delays in the process. In light of these challenges, our aim shifts towards devising an approach that circumvents the need for multiple packets, focusing on effectively differentiating between Tor and nonTor traffic. This leads us to our second research question: ***RQ2: Can we distinguish Tor traffic using the encrypted payload in a data-efficient manner?*** Considering that collecting an extensive set of packets can be either impractical or intrusive, a method that reduces the data requirement to just a single packet proves to be more data-efficient. To address this, we employ machine learning techniques, specifically extracting character statistical-based features from the encrypted payload. We then conduct classification and predictions on unseen data using three supervised learning algorithms including J48, Random Forest (RF) and IBk. Ultimately, our goal is to demonstrate the efficiency and accuracy of our proposed method in differentiating Tor traffic from nonTor traffic.

Our research offers a significant contribution to the field of network security by presenting a novel and effective approach for detecting Tor traffic. Here are the primary contributions from our research:

1) Our novel approach requires only one packet to determine whether the traffic originated from Tor or nonTor networks. This represents a significant improvement compared to conventional methods, which usually require multiple packets for accurate classification.
2) Our novel approach requires only a single packet to determine whether the traffic originated from Tor or nonTor networks. This independence from sequential packet analysis allows us to examine packets from any location within network traces, eliminating the necessity to analyze packets in a specific order. Such flexibility enhances the versatility of our method in detecting Tor activity.
3) Our novel approach is based on the DPI of encrypted network traffic to differentiate packets between Tor and nonTor networks without decryption or accessing the actual content of the encrypted payload, thereby ensuring privacy preservation. Our approach accommodates the growing concern for user privacy in the digital age.

The rest of the paper is organised as follows: In Section II, we present related work on Tor traffic classification. In Section III, we provide relevant background information related to Tor traffic classification. Section IV explains the key elements involved in Tor traffic classification. In Section V, we introduce the proposed method. Section VI covers the two analysis techniques - statistical and machine learning. Discussion is provided in Section VII, with our conclusion presented in Section VIII.

II. RELATED WORK

In the domain of Tor traffic classification, numerous studies have successfully classified Tor traffic using machine learning algorithms, leveraging a variety of features. As encryption becomes more prevalent, many researchers now emphasise flow-based features or metadata from packet-based features, ensuring that they do not compromise the integrity of encrypted data.

As depicted in Table 1, various approaches to Tor traffic classification have been taken in past research. These approaches range from emphasizing flow-based features to exploring payload-based techniques. Many of these studies have chosen flow-based features because Tor traffic exhibits unique latency patterns [1], [2], allowing for fine-grained classification. However, these methods face significant challenges. For instance, asymmetric routing can reduce the reliability in classifying traffic flows [3], and the reliance on multiple packets for feature extraction decreases data efficiency and introduces processing delays. Additionally, while deep learning algorithms are known for their capability in automated feature extraction and often achieving high accuracy, their model interpretability remains a challenge. Our proposed method addresses these weaknesses by introducing a novel approach: handcrafting features from the frequency of hexadecimal characters within encrypted payloads. This method requires only a single

TABLE 1. Comparison of tor traffic classification methods.

Study	Features	Packets	Approach	Method
Lashkari et al. [8]	Time-based features from Flow packets	Multiple	Detecting Tor traffic (a) Categorising Tor-based apps (b)	C4.5, RF, *k*NN, ZeroR
Cuzzocrea et al. [9]	Time-based features from Flow packets	Multiple	Detecting Tor traffic (a) Categorising Tor-based apps (b)	JRip, J48
Hodo et al. [12]	Time-based features from Flow packets	Multiple	Anomaly detection	SVM, ANN
Kim et al. [13]	Features based on Hexadecimal raw packet header	Single	Detecting Tor traffic (a) Categorising Tor-based apps (b)	1D-CNN
Our Study	Frequency-based features of hexadecimal characters in encrypted payload	Single	Detecting Tor traffic	J48, RF, *k*NN

packet for accurate classification, significantly enhancing data efficiency compared to existing methods that depend on multiple packets. Moreover, our approach utilises models that are transparent and easily interpretable, addressing the common issue of opaqueness associated with deep learning models.

Looking into the specifics, previous studies that rely on flow-based features usually focus on two scenarios: 1) detecting Tor traffic (scenario (a)) and 2) categorising Tor-based applications (scenario (b)). The widely used UNB-CIC dataset, introduced by Lashkari et al. [8], comprises both Tor and nonTor traffic and serves as a foundational resource in this research domain. The cited authors employed this dataset to perform Tor traffic classification at two levels, achieving high performance in both scenarios. Specifically, for scenario (a) they achieved precision and recall above 0.9 using the C4.5 algorithm. For scenario (b), the RF algorithm yielded notable results with above 0.8 for the same metrics.

This dataset has also been utilised in other studies, Cuzzocrea et al. [9] showed that the JRip classifier delivered the best performance in scenario (a), achieving 1 for the weighted average of both recall and precision. Meanwhile, the J48 classifier demonstrated the best performance in scenario (b) with a score of 0.998 for the same metrics. Similarly, other researchers have showcased successful experimentation with flow-based features to achieve similar objectives, employing various classification techniques [2], [10], [11].

Hodo et al. [12] pursued different objectives aimed at improving the efficiency of the Tor network by identifying anomalous or nonTor traffic that could compromise users' privacy. They utilised a learning system to establish a regular traffic profile and detected deviations from this profile as outlier traffic. Their findings demonstrated that a CFS-ANN hybrid classifier outperformed SVM in detecting nonTor traffic, achieving respective overall accuracies of 99.8% and 94%.

As previously discussed, certain network performance factors, such as asymmetric routing can impact the reliability of features extracted from flow packets [3]. Additionally, the need to compute multiple packets for feature extraction can lead to processing delays. Recently, Kim and Anpalagan [13] sidesteps the constraints of time-based features in flow packets. Their focus on payload-based features closely aligns with our approach. They utilised the UNB-CIC dataset to compare results with the dataset creators [8]. Their proposed method employed a 1D-CNN model with raw packet headers as input, specifically utilising the first 54 bytes of TCP packets, including the TCP/IP header and Ethernet II header. These hexadecimal values were then converted to decimal values. The CNN model achieved packet classification without the need for manual feature extraction or engineering. The results indicated that the CNN outperformed the C4.5 algorithm in terms of precision and recall for classifying Tor and nonTor traffic in scenario (a). Additionally, in scenario (b), where application-level Tor traffic was classified, the CNN model consistently outperformed the C4.5 algorithm for all applications. These findings highlight the effectiveness of the proposed approach in classifying Tor traffic without hand-crafted feature extraction or engineering, showcasing the advantages of deep learning techniques. However, while this payload-based approach overcomes the inherent limitations of time-based features by leveraging deep learning technology, it still tends to obfuscate the insights derived from the features.

Until now, as far as we're aware, no studies have specifically concentrated on classifying Tor traffic through an engineering approach that emphasises manual intervention on its encrypted payload. We are the pioneers in this domain, extracting features exclusively from a single encrypted payload using a transparent model.

III. BACKGROUND TECHNOLOGY

In this section, we provide essential background technology information on Tor traffic classification, which is crucial for understanding the essence of our work. This will detail the methodology employed in our research and its significance in effectively classifying Tor traffic.

A. AN OVERVIEW OF TOR NETWORK

Tor, known as *The Onion Router*, enables Internet users to maintain privacy and anonymity. It is an overlay anonymous network operated in the Application Layer of the TCP/IP

protocol stack, Tor employs *onion routing* to encrypt packets multiple times, resembling peeling onion layers. This process ensures user identities remain concealed while accessing the Internet. Tor serves the purpose of preserving privacy for both individuals and groups. For instance, it assists general Internet users in preventing their online information from being viewed by ISPs. Additionally, Tor offers secure communication for various entities, including reporters, whistleblowers, and NGO workers. These individuals can establish secure and anonymous remote connections to organisational websites without revealing their identities. Moreover, activist groups utilise Tor to advance civil liberties online while evading government monitoring. However, the benefits of Tor can be misused for illicit purposes, highlighting the need for monitoring who is accessing the Tor network. This emphasises the critical importance of Tor traffic detection.

The Tor network comprises around 7,000 volunteer nodes contributing bandwidth to form virtual circuits. This elaborate network structure serves as the foundation for the Tor network's operation. The Tor network is based on the concept of onion routing, which involves routing connections through multiple nodes instead of direct client-to-server communication. In a direct connection scenario, participants are aware of each other's IP addresses, which impairs anonymity. In contrast, employing a non-direct connection within the Tor network involves encrypting data and relaying it through several nodes before reaching the destination. Each node only possesses knowledge of the preceding and succeeding nodes, rather than the entire path. This makes it significantly harder to trace your online activity back to your original IP address, providing a higher level of privacy and anonymity. Figure1 provides a visual comparison of the Tor network and the conventional Internet. When a user accesses a website on the standard Internet, the browser establishes a direct connection to the destination (a). In contrast, within the Tor network, packets are routed through multiple nodes, with each relay modifying the packets along the communication path (b. This multi-layered approach ensures increased privacy and anonymity for Tor users.

The Tor network comprises four main components:

Tor Client: Launches the Tor browser to connect to the Tor network.

Onion Routers (ORs): Carry data between the client and destination through entry, middle, and exit nodes. At each router, packets are modified through encapsulation and decapsulation using the TLS protocol.

Destination: Refers to any servers accessed by the Tor client.

Directory Servers: Deliver signed documents containing OR information, enabling clients to receive real-time updates on the Tor network.

Tor employs a distinctive approach in contrast to conventional communication methods. Instead of transmitting traffic in its original packet format, Tor uses fixed-sized cells of 512 bytes for data transmission, ensuring enhanced anonymity. The process of transmitting data across the

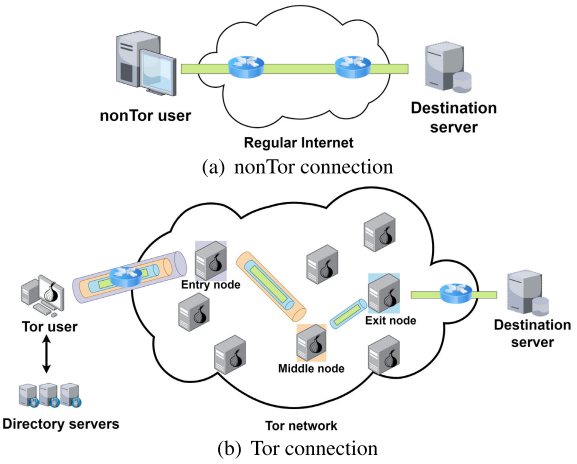


FIGURE 1. Comparison of connections in normal Internet (a) and Tor network (b).

Tor network involves the initial establishment of circuits, which are formed by randomly selecting three ORs from the Tor directory service. Constructing these circuits involves encryption and decryption operations. Tor utilises a combination of symmetric and asymmetric encryption techniques similar to the TLS protocol in principle, but Tor's approach involves more complex operations.

Tor strengthens its communication security by incorporating the TLS protocol and employing a combination of cryptographic techniques. For the secure exchange of public keys, Tor employs asymmetric cryptography using algorithms like Diffie and Hellman [14] and RSA (Rivest-Shamir-Adleman) [15]. Historically, Tor has utilised a 1024-bit key encryption for these algorithms. However, it's worth noting that, as of today, a security parameter of 1024-bit is considered insufficiently robust for Diffie-Hellman. For bulk data transmission, it utilises symmetric cryptography, specifically 128-bit AES (Advanced Encryption Standard) in Counter (CTR) mode, to encrypt and decrypt cells. As a result, data captured at a Tor client is encrypted with AES three times. Additionally, Tor integrates various other cryptographic schemes for diverse operations, ranging from path selection to congestion management [1].

B. UNDERSTANDING ENCRYPTED PAYLOADS IN NETWORK TRAFFIC

In this study, our primary focus lies on extracting features from the encrypted payload of network packets, which encompasses both headers and actual data or payload. The headers contain plaintext metadata crucial for communication, while the payload may consist of either plaintext or ciphertext, a product of encrypted payload. This encryption process occurs at the Application Layer before transmission to the Transport Layer. Several encryption protocols play a role in this process, but we will exclusively focus on the protocols present in the dataset utilised for this study. These protocols include TLS, Secure Shell (SSH), and proprietary protocols.

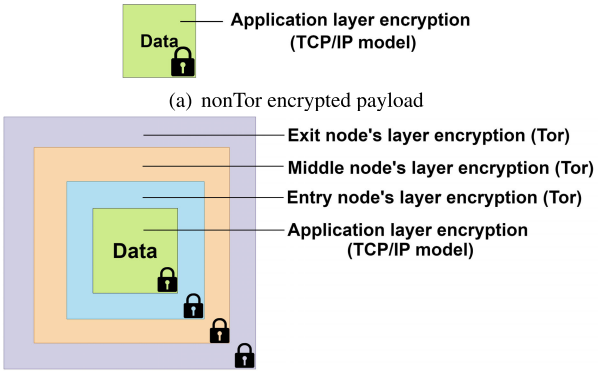


FIGURE 2. Comparison of encrypted payload structures in nonTor (a) and Tor networks (b).

Considering the encrypted payloads generated by Tor and nonTor networks, Tor stands out for its distinctive encryption methodology. In addition to the payload encryption at the application layer of the TCP/IP model, Tor incorporates a multi-layered encryption methodology before transmitting data to its intended destinations. In contrast, nonTor packets typically don't incorporate any additional encryption layers. Figure 2 provides a comparison of the number of encryption layers in outgoing and incoming packets between nonTor and Tor networks. In Tor, the packets are encrypted three times at relay nodes, which results in three additional encryption layers. In contrast, nonTor packets remain without these extra layers, as they lack supplementary encryption. As emphasised earlier, the primary objective of encryption is to ensure that the encrypted data does not reveal any details about its original content, even if adversaries have knowledge about the encryption methods [6]. Ideally, the encrypted payloads from both network types should look identical and be indistinguishable. This aspect prompts us to investigate further.

- **Encrypted Payload Transmission** Data is transmitted on the Internet via TCP/IP. To ensure efficient network communication, application data including encrypted payload needs to be sent in smaller pieces. This is one of the responsibilities of TCP that lies on the Transport Layer, TCP breaks down the application data received from the upper layers into smaller, transmittable units called segments. Each segment contains a sequence number, acknowledgement number, and other control information in the header. The size of these chunks is determined by the Maximum Segment Size (MSS) parameter, which specifies the maximum size of packets that can be sent over a network. MSS can vary based on various factors, dependent on the specific TCP stack implementation. Typically, MSS is calculated based on the Maximum Transmission Unit (MTU) value managed at the Network Layer. While many implementations use segments of 512 or 536 bytes [16], the actual value of MSS can vary depending on the specific TCP stack and configuration in use. This study utilises the segments of the encrypted payload from Tor and nonTor networks for analysis.

C. A BRIEF OVERVIEW OF MACHINE LEARNING IN NETWORK TRAFFIC CLASSIFICATION

Older and limited conventional methods, such as payload-based and port-based approaches, have led to the emergence of machine learning as a crucial tool in network traffic classification (NTC). In NTC, the general process of machine learning involves collecting raw network traffic data from various sources, pre-processing the data to clean and format it appropriately, extracting relevant features to represent the traffic patterns, labelling the data with corresponding classes, selecting an appropriate machine learning algorithm, training the model on the labelled data to learn patterns and relationships, evaluating the model's performance using validation metrics, tuning hyperparameters for optimisation, testing the model on unseen data to assess real-world performance, and finally, deploying the trained model into the network infrastructure for real-time traffic classification and management.

The key techniques of the machine learning process utilised in this study are presented as follows:

1) SUPERVISED LEARNING ALGORITHMS

In our experiment, we employed the Weka software suite [17], a popular and versatile platform for machine learning tasks. For our binary classification task, we selected three supervised learning algorithms to compare their performance. We utilised the J48 algorithm, a decision tree classifier available in Weka. Decision trees (DTs) are widely recognised for their simplicity, interpretability, and robustness in handling missing values. We complemented J48 with the RF algorithm known for enhancing performance and preventing overfitting. Additionally, we considered the IBk algorithm for its simplicity, effectiveness, and high performance alongside the other chosen algorithms.

- **J48** is a Java implementation of the C4.5 algorithm, which is a widely employed DT algorithm used for regression and classification tasks [18]. DTs are created using a divide-and-conquer approach, wherein nodes are recursively split into sub-nodes based on data attributes, forming an inverted tree-like structure. J48 employs the gain ratio as the splitting criterion, enhancing its decision-making process during tree construction. In this structure, the root node represents the topmost attribute, branches correspond to decision rules, child nodes represent attributes, and leaf nodes indicate decision outcomes or classes.
- **RF** is an ensemble learning method that creates a more robust and accurate model by combining multiple DTs. Each DT in the RF is trained on a different random subset of the data, achieved through bootstrapping [19]. Additionally, random feature selection is applied to

form each decision tree, introducing diversity among the trees. The RF model leverages aggregation, where the predictions from multiple trees are combined to make the final decision. By aggregating the results, the RF model achieves better generalisation and higher accuracy compared to a single decision tree. The combination of bootstrapping and aggregation is often referred to as bootstrap aggregating or bagging, which ensures the creation of diverse decision trees, effectively mitigating overfitting [20], a common issue with individual DTs, and ultimately enhancing the performance of the RF model. However, one drawback of the RF approach is its reduced interpretability compared to a single decision tree. This is because the RF model operates as an ensemble of multiple trees, making it challenging to discern the internal logic behind its predictions.

In addition to the RF model, which employs decision trees within the bagging framework, there are other ensemble methods such as boosting and stacking. All three can enhance performance robustness across various scenarios, but they come with their challenges, such as computational costs and difficulties in interpreting results. Boosting builds models sequentially where each new model corrects the errors of its predecessor. However, this approach may sometimes lead to overfitting, especially in the case of GBM. Stacking, on the other hand, involves training multiple models and then using another model (a meta-learner) to combine their predictions. One challenge with stacking is the complexity involved in defining appropriate models [21]. A study [22] highlighted that the accuracy gains using bagging, boosting, or stacking might not always be significant. In our context, we chose bagging due to its ability to reduce variance without increasing bias, making it particularly effective for complex models like decision trees that are prone to overfitting. Bagging, especially when implemented via techniques like RF, offers a balance between performance and model robustness.

- **IBk** refer to the *k*NN algorithm, which operates by identifying the k most similar instances to a new instance and predicting its label based on the majority label among these neighbours. To prevent ties when multiple neighbours share the same distance, an odd value of k is typically selected. The choice of k is a crucial hyperparameter that significantly impacts the *k*NN algorithm's performance. A larger value of k enhances the algorithm's robustness to noise, but it may also reduce its accuracy. On the other hand, a smaller value of k increases accuracy but may make the algorithm more sensitive to noise. The selection of an appropriate k-value depends on the specific dataset and problem at hand. The *k*NN classifier commonly employs the Euclidean distance function to calculate the distance between two points or nearest neighbours. This distance function measures the straight-line distance in an Euclidean space, which is suitable for a wide range of applications [23].

2) DATA SPLITTING

Data splitting involves dividing the available dataset into subsets for training and testing. Our study used two techniques to split data as follows.

- **Percentage split:** The dataset is divided into two portions based on a specified percentage. Our study employed a 70-30 percentage split, with 70% of the data allocated for training and the remaining 30% reserved for testing.
- **Cross-Validation (CV)**: This is a widely-used technique that divides the dataset into training and testing portions to evaluate predictive models. It involves splitting the data into n equally sized, random folds (often n=5 or n=10). In each iteration, one fold serves as the testing set while the remaining folds are used for training, and this process is repeated n times. The objective is to reduce bias and variance by thoroughly training and testing the model on different subsets of data. For this study, a 10-fold CV approach was utilised.

We used both data-splitting techniques. The percentage split suffices for reliable performance estimates in small or simple datasets, whereas a 10-fold CV is preferred for larger or complex datasets to prevent overfitting. It is essential to balance between robust performance and efficient use of computational resources. In this study, for datasets with about 10,000 instances or fewer, we applied CV. Conversely, for datasets larger than approximately 10,000 instances, we employed a percentage split.

3) PERFORMANCE METRIC EVALUATION

We evaluate classification model performance using the following standard evaluation metrics. These metrics are derived from the confusion matrix, which summarises the model's performance based on true positive (TP), false positive (FP), true negative (TN), and false negative (FN) predictions.

- **Accuracy:**

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

This metric provides an overall assessment of the model's correctness in terms of how well it can correctly classify Tor and nonTor traffic.

- **Precision:**

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

Precision gauges the accuracy of the positive predictions made by the model. Precision measures the model's ability to correctly classify Tor traffic among those instances it predicts as Tor.

- **Recall:**

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

Recall indicates the model's ability to correctly identify positive instances. Recall measures the model's ability to capture all actual instances of Tor traffic.

- **F1-score:**

$$F1\text{-}score = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (4)$$

The F1-score is the harmonic mean of precision and recall, providing a balanced metric that's especially useful for imbalanced datasets. In the context of classifying Tor and nonTor traffic, it measures the model's efficiency in accurately detecting Tor traffic, factoring in both false positives and false negatives.

IV. KEY COMPONENTS OF TOR TRAFFIC ANALYSIS

This section introduces the essential elements employed in our study's analysis of Tor traffic.

A. DATASET DESCRIPTION AND SOURCE

In this study, we utilised UNB-CIC or ISCXTor2016 dataset [1] that has been widely used in the research area of Tor traffic. The dataset encompasses both Tor and nonTor traffic data, gathered from real-world interactions via Whonix—a Tor-integrated open-source operating system based on Linux. Whonix consists of two Debian GNU/Linux virtual machines: a workstation and a gateway. Traffic captured by the Whonix workstation pertains to *regular nonTor traffic*, while traffic captured by the Whonix gateway corresponds to *Tor traffic*, as it is routed through the Tor network. The dataset was collected by simultaneously recording network activities on both virtual machines using a packet sniffing tool (e.g., Wireshark) and storing regular traffic and Tor traffic as separate PCAP files in Tor and nonTor directories. The UNB-CIC dataset contains 8 application types (Audio, Browsing, Chat, Email, FTP, P2P, VDO and VoIP) that were generated from 18 software applications. These applications employ several encrypted Internet protocols, including TLS, SSH, and proprietary protocols.

B. DATA GROUPING AND LABELLING

This dataset is categorised and labelled into eight application types, aligning with the application types in the original dataset. This categorisation results in eight binary classifications for statistical and machine learning purposes. These eight groups of data ensure consistency and validity in the quality of the results.

C. ENCRYPTED PAYLOAD FEATURES

Cryptography generates encrypted payloads through encryption, forming sequences of uniform hexadecimal characters according to rigorous principles [6]. However, real-world cryptography in computer networks can be vulnerable, potentially leading to information leaks. Statistical analysis, which has proven successful in text classification [24], [25], [26], is being leveraged in the field of computer networks.

The content of encrypted payloads appears as unreadable data and is displayed by network analyser tools in various formats, such as binary, hexadecimal, and ASCII. Among these formats, the binary representation comprises only 0s and 1s, making it challenging to comprehend and manipulate for larger values. In contrast, the hexadecimal format employs a base-16 numbering system with 16 characters (0 - 9, a - f), facilitating the interpretation and manipulation of larger values. Furthermore, encrypted data frequently contains non-printable characters that don't map well to ASCII characters, adding complexity to the analysis. In comparison, the hexadecimal representation accommodates both printable and non-printable characters, allowing for a more accurate and comprehensive analysis of encrypted data. Consequently, the study focuses on the character analysis of hex character representation in 1-hex form, as a suitable method. Two sets of features are considered:

- **Set 1: *F_0 - F_f***: This feature measures the frequency of each hexadecimal character distributed in encrypted payloads. While encryption renders the payload unreadable, the character frequencies remain observable. By examining the frequency of each character, it is possible to identify which characters occur more frequently and which are rarer.
- **Set 2: *R_0 - R_f***: This feature calculates the ratio of each hexadecimal character's frequency to the total frequency in the encrypted payload. Normalising the frequency distribution ensures length normalisation and independence, reducing biases or inaccuracies that may occur when analysing encrypted payloads based solely on absolute packet length.

To ensure the independence of features in Sets 1 and 2, we employed the Pearson correlation coefficient. The Pearson correlation coefficient, typically represented as r, quantifies the relationship between two variables, X and Y. It is computed using the following formula:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \quad (5)$$

where:

- n represents the number of data points.
- X_i and Y_i denote the individual data points for the variables X and Y, respectively.
- $\bar{X}$ and $\bar{Y}$ are the means of X and Y, respectively.

A value of r close to 1 indicates a strong positive relationship, while a value close to -1 signifies a strong negative relationship. A value near 0 indicates that there is no significant relationship between the variables.

V. PROPOSED METHOD

The proposed approach in this study comprises three main steps. Initially, the raw data is subjected to pre-processing, which includes tasks such as categorising the data into different types, eliminating irrelevant packetsparticularly those with unencrypted payloads or those tied to connection

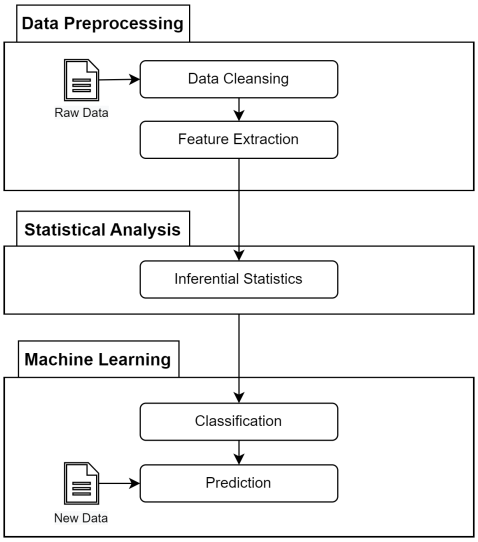


FIGURE 3. Diagram of the proposed approach.

phases—and extracting features based on the frequency of hexadecimal characters suited for both statistical and classification analyses. Secondly, inferential statistics are applied to the pre-processed data to address *RQ1*, focusing on differentiating Tor from nonTor traffic based on their encrypted payload. Lastly, the pre-processed data is used for classification and prediction to answer *RQ2*, evaluating the efficiency of the proposed approach in distinguishing traffic originating from Tor and nonTor networks.

A. DATA PREPROCESSING

1) DATA CLEANSING

In the digital era, numerous applications employ dynamic ports, often as a strategic measure to evade specific network restrictions or monitoring efforts. However, even widely recognised ports do not guarantee the identification of a particular application. For example, while Tor traffic can use port 443, this port is conventionally associated with HTTPS. Therefore, to ensure accurate data extraction, we closely adhere to the dataset description utilised in this study. This description, detailed within the corresponding paper [8], pertains to data captured from the control environment of each application within both the Tor and nonTor networks. Following the data grouping and labelling discussed in Section IV-B, two pivotal criteria are considered to filter only the relevant packets during data extraction: (1) obtaining packets containing payload data while disregarding network connection-related packets, and (2) retaining only payloads encrypted by the specific secure communication protocols. Based on the security protocols presented in the dataset, TLS, SSH, and secure proprietary protocols are taken into account.

Algorithm 1 (shown below) presents a systematic approach for extracting relevant packets from the network traces during the data cleansing process. For each packet, the algorithm checks if the topmost layer protocol is TLS, SSH, TCP, or a proprietary protocol. If the payload is non-empty, it extracts the relevant data using specific commands (‘`tls.app_data`’ for TLS, ‘`ssh.encrypted_packet`’ for SSH, and ‘`tcp.data`’ for TCP or proprietary protocols). The extracted payload data is then added to the encrypted payload set (E). The algorithm iterates through all packets in the network flow, applying these steps to each. After processing all packets, the resulting encrypted payload set (E) is obtained as the output. This procedure ensures the extraction of only relevant encrypted payload data from both Tor and nonTor networks for further analysis while excluding unnecessary packets from the dataset.

Algorithm 1 Data Cleansing Algorithm

Require: F: Network flow in PCAP format
Ensure: E: The encrypted payload

```
i ← 0 //Initialise with the first packet in the PCAP file
while i < F do
    Read packet i
    if topmost_layer_protocol = "TLS" then
        Read payload i
        if payload i != "0" then
            E ← Get tls.app_data //Extract non-empty TLS payloads and append to the encrypted payload set E
        end if
    else if topmost_layer_protocol = "SSH" then
        Read payload i
        if payload i != "0" then
            E ← Get ssh.encrypted_packet //Extract non-empty SSH payloads and append to the encrypted payload set E
        end if
    else if topmost_layer_protocol = "TCP" or proprietary protocol then
        Read payload i
        if payload i != "0" then
            E ← Get tcp.data //Extract non-empty TCP payloads and append to the encrypted payload set E
        end if
    end if
    i ← i + 1
end while
return E
```

For the imbalanced extracted data, we applied the *under-sampling* technique to balance the data to improve the quality of results for statistical analysis and machine learning. Table 2 presents the number of Tor and nonTor encrypted payloads before and after data balancing across all application types.

However, the number of Tor and nonTor encrypted payloads after data balancing is divided further into 90% for classification analysis and 10% for the unseen data prediction phase in the machine learning approach.

2) FEATURE EXTRACTION

We extracted our proposed features into two sets of hexadecimal statistical calculations, referred to as the frequency set and ratio set. These statistics are obtained using the

TABLE 2. Number of Tor and nonTor encrypted payloads before and after data balancing across all application types.

Application types	Before undersampling		After undersampling	
	Tor	nonTor	Tor & nonTor (each)	Total
Audio	13,727	148,335	13,727	27,454
Browsing	85,840	37,868	37,868	75,736
Chat	3,423	6,066	3,423	6,846
Email	28,559	6,474	6,474	12,948
FTP	271,027	512,339	271,027	542,054
P2P	228,300	1,339,363	228,300	456,600
VDO	103,603	16,923	16,923	33,846
VoIP	877,700	388,096	388,096	776,192

mathematical equations presented below.

$$F_{c_i}(p) = \sum_{k=1}^{|p|} \delta(c_i, p(k))$$
$$\forall i \in \{0, 1, \ldots, F\}$$
$$\delta(c_i, p(k)) = \begin{cases} 1, & \text{if } c_i = p(k) \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

Equation 6 calculates the frequency of each hexadecimal character in a single payload, $F_{c_i}(p)$ represents the frequency of hex character c_i in payload p. The hex character c_i ranges from 0 to F. The payload p has a length denoted by $|p|$, and k is an index variable iterating through each character in the payload. The indicator function $\delta(c_i, p(k))$ takes the value 1 when hex character c_i matches the k-th character of payload p and 0 otherwise.

$$R_{c_i}(p) = \frac{F_{c_i}(p)}{T(p)} \quad (7)$$

Equation 7 calculates the ratio of each hexadecimal character in a payload, $R_{c_i}(p)$ represents the ratio of hex character c_i in payload p. The hex character c_i ranges from 0 to F. $F_{c_i}(p)$ denotes the frequency of hex character c_i in payload p, as calculated using Equation 6. The payload p has a total character count represented by $T(p)$.

Regarding the equations, we implemented the feature extraction process outlined in Algorithm 2.

This algorithm processes each packet in the extracted encrypted payload (P), splitting the packet if it contains a comma (","). For each packet, it counts the character frequency in the payload (F) and normalises it to calculate the ratio of individual hex characters (R). The algorithm then returns the extracted features: F and R which will be utilised for statistical analysis and classification tasks.

Using Algorithm 2, along with Equation 6 and Equation 7, we derived 32 features which are comprehensively listed in Table 3. These features were utilised in the classification process. For easier reference, they are categorised into two sets: Set 1 comprises the 16 hex character frequency features

Algorithm 2 Feature Extraction Algorithm

Require: P: Extracted encrypted payloads
Ensure: F: Frequency of individual hex characters in the encrypted payloads
R: Ratio of the frequency of individual hex characters in the encrypted payloads

```
for all packet in P do
    if packet contains "," then
        Split and append a new row
    end if
    F ← Frequency count //calculated using Eq. 6
    R ← Ratio of F //calculated using Eq. 7
end for
return F, R
```

(F_0 - F_f) and Set 2 includes the 16 hex character frequency ratio features (R_0 - R_f).

TABLE 3. List of features used in the analyses.

Set Name	List of Features	Total Number of Features
Set 1: Frequency of hex characters	*F_0, F_1, F_2, F_3, F_4, F_5, F_6, F_7, F_8, F_9, F_a, F_b, F_c, F_d, F_e, F_f*	16
Set 2: Ratio frequency of hex characters	*R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_a, R_b, R_c, R_d, R_e, R_f*	16

B. STATISTICAL METHOD

Inferential statistics serve to draw conclusions about populations based on sample data. A prominent application of inferential statistics is determining whether two independent samples come from populations with identical distributions [27]. In this study, we employed the Mann-Whitney U test, a nonparametric test, to determine if there's a significant difference in the specific characteristics of Tor and nonTor encrypted payloads, especially when data is not normally distributed.

For a valid application of the Mann-Whitney U test, there are essential assumptions to consider. The two datasets being compared should be independent, ensuring that observations in one group are not influenced by those in the other. The data should be either ordinal, meaning rankable, or numerical. Additionally, the distributions of both groups should be similar in shape; while this doesn't necessitate identical distributions, the general shape or pattern should be consistent between the two. The dispersion or spread of data in both groups should be relatively comparable, ensuring that one group doesn't have significantly more variability than the other. The test yields a p-value ranging from 0 to 1. A p-value less than 0.05 indicates statistical significance, suggesting that there's less than a 5% probability of observing the given data (or more extreme) if the null hypothesis is true. If the *p-value* is less than 0.05, we reject the null hypothesis and accept the alternative hypothesis.

Equation 8 presents the calculation of the Mann-Whitney U test statistic as follows:

$$U = \min(U_1, U_2) = \min\left(\sum_{i=1}^{n_1} R_i, \sum_{j=1}^{n_2} R_j\right) \quad (8)$$

where:

- U is the test statistic.
- U_1 and U_2 represent the sum of ranks for Tor and nonTor encrypted payloads, respectively.
- n_1 and n_2 are the sample sizes of the Tor and nonTor payloads, respectively.
- R_i and R_j denote the ranks of the observations when both sample sets are combined.

The resulting U statistic provides a basis to compute the p-value, which is then used to determine the statistical significance of the observed differences. A p-value less than 0.05, in our context, suggests that the characteristics of the Tor and nonTor encrypted payloads are statistically different, justifying the acceptance or rejection of our null hypothesis.

C. MACHINE LEARNING APPROACH

The machine learning approach was employed to evaluate the performance of our proposed method. The problem in our study fell under binary classification, needing labelled data with extracted features for machine learning analysis. This analysis included both the classification and prediction stages.

In our study, the machine learning models were constructed using WEKA version 3.8.3 with default hyperparameters. For the J48 model, we used a confidence factor for pruning set at 0.25, ensuring less aggressive pruning. The minimum number of instances per leaf was set to 2. For the RF algorithm, the forest was generated with 100 trees, and each tree was grown to an unlimited depth. At each node, the square root of the number of attributes was considered for splits, and bootstrap sampling was used with a sample size set at 100% of the training set. The IBk employed a default of 1 nearest neighbor. The distance between instances was calculated using the Euclidean metric, with no distance weighting applied. A linear search algorithm was used to identify the nearest neighbours.

1) CLASSIFICATION

The preprocessed data contains a binary classification of eight application types with two sets of features: frequency and ratio frequency. Ensuring the data is balanced helps circumvent any biases and lays the foundation for consistent performance across all classifiers. In the model training phase, the balanced dataset is trained using three supervised learning algorithms: J48, RF, and IBk as discussed in Section III-C1. We proceeded with default parameters to ensure a baseline performance that is generally acceptable. While in-depth hyperparameter optimization could potentially enhance model performance, it also demands significant computational resources. Using default parameters allowed us a straightforward comparison across algorithms without the risk of introducing biases or overfitting that specialized tuning might occur. After training, we evaluated the models against a validation set, using performance metrics such as accuracy, precision, recall, and F1-score.

2) PREDICTION

Following the training in the classification phase, the finalised model was applied to new, previously unseen data to prevent overfitting and to assess the model's generalisation ability. This is a crucial step, ensuring that our model does not only fit our training data but also understands underlying patterns and can generalize its learning to new instances. The model's performance during this phase gives us an indication of its real-world applicability and robustness.

VI. STATISTICAL AND MACHINE LEARNING ANALYSES

The preprocessed data, divided into eight binary classes, each containing two sets of features (frequency and ratio), were analysed separately: first in the statistical analysis and then in the machine learning analysis.

A. STATISTICAL ANALYSIS

The analysis aimed to address the question of whether we can differentiate Tor from nonTor encrypted traffic based on their encrypted payloads in *RQ1*. We compared statistically encrypted payloads based on individual hex character statistics using two sets of features, each containing 16 features, for both Tor and nonTor traffic across eight application types. We used the Mann-Whitney U test to determine the significance of feature differences between the two traffic types. In accordance with the requirements of the Mann-Whitney U test, it's important to recognize certain foundational assumptions. The two groups of data we analyzed are independent of each other. The data under study is discrete in nature making it rankable. Our inspection of the distributional shapes and variances, done through histogram visualizations, revealed obvious differences in the shapes of distributions and their variances for feature set 1, influenced by the payload size. In contrast, feature set 2, which is independent of the payload size, displays no significant differences.

The summarised results are presented in Table 4, highlighting features with p-values greater than 0.05, indicating that they did not meet the conventional threshold for significance, suggesting non-significant differences between Tor and nonTor encrypted payloads. We can infer certain resemblances between Tor and nonTor encrypted payloads across different application types. Specifically, in the Audio application type, the features *R_5*, *R_c*, and *R_e*, were found to be indistinguishable between Tor and nonTor payloads. This resulted in a similarity rate of 9.38% for both Tor and nonTor Audio encrypted payloads. In the case of Chat, ten features were identified, (*R_1, R_5, R_6, R_7, R_8, R_9, R_a, R_b, R_d and R_e*) that led to a similarity rate of 31.25% for both Tor and nonTor Chat encrypted payloads. Conversely,

TABLE 4. Features with *p*-values > 0.05 in Tor and nonTor encrypted payloads.

Application types	Features	*p*-value (Mann-Whitney U)
Audio (3)	*R_5*	0.077
	R_c	0.066
	R_e	0.803
Chat (10)	*R_1*	0.368
	R_5	0.156
	R_6	0.964
	R_7	0.741
	R_8	0.481
	R_9	0.770
	R_a	0.064
	R_b	0.225
	R_d	0.185
	R_e	0.051
P2P (1)	*R_0*	0.380

P2P had only one identical feature (*R_0*), resulting in a resemblance rate of 3.13% for both Tor and nonTor P2P encrypted payloads.

In conclusion, considering all features from all application types, the Mann-Whitney U test findings revealed that out of the 256 features analysed, 242 features demonstrated significant differences between Tor and nonTor encrypted payloads, resulting in a high differentiation rate of 94.53%. These results highlight the distinct characteristics of encrypted payloads between the two traffic types, suggesting inherent differences in their distributions. Leveraging these differences could offer the method for developing automated tools to distinguish between the two traffic types. The following section provides the results of classification and prediction using machine learning techniques.

B. MACHINE LEARNING

To address the question of whether we can efficiently distinguish Tor traffic using the proposed approach in *RQ2*, we conducted a machine learning analysis. For this purpose, we employed three classification algorithms: J48, RF, and IBk in Weka to perform classification and prediction across eight binary classes. To ensure the robustness and reliability of our experimental outcomes and account for potential biases, we adopted a combination of percentage split and 10-fold CV techniques for dataset partitioning in all experiments.

1) CLASSIFICATION RESULTS

The accuracy scores of different feature sets and classification algorithms, applied to eight application types, are presented in Table 5. For Set 1 features, J48 achieved an average accuracy score of 94.71% while RF and IBk demonstrated higher scores of 96.53% and 96.42%, respectively. In contrast, for Set 2 features, J48 showed a higher performance with a score of 95.65%, while RF and IBk reached 92.62% and 73.77% on average, respectively.

TABLE 5. Accuracy comparison of two feature sets using J48, RF, and IBk algorithms.

Application types	J48(%)		RF(%)		IBk(%)	
	Set 1	Set 2	Set 1	Set 2	Set 1	Set 2
Audio	90.01	88.99	91.97	88.99	92.04	65.80
Browsing	94.35	99.03	96.56	94.26	96.86	70.44
Chat	91.34	93.36	95.94	86.37	96.88	60.62
Email	97.08	91.85	98.20	92.85	97.64	89.24
FTP	93.92	96.43	95.52	91.58	94.40	72.41
P2P	95.00	98.44	96.53	96.82	96.23	69.14
VDO	96.38	97.46	97.78	91.18	97.57	68.53
VoIP	99.62	99.75	99.77	98.92	99.77	93.98
Average	**94.71**	**95.65**	**96.53**	**92.62**	**96.42**	**73.77**

The table reveals that Chat and Audio generally demonstrated lower accuracy scores across most feature-algorithm combinations. VoIP, on the other hand, consistently showed the highest accuracies, nearing 100% for both feature sets across all algorithms.

Overall, RF achieved the highest accuracy for classifying Tor traffic using Set 1 features at 96.53%, while J48 led with Set 2 features at an average accuracy of 95.65%.

Despite RF achieving the highest average accuracy score for classifying Tor traffic using Set 1, the examination of correlation coefficient values between Set 1 and packet size reveals a significant association of Set 1 with packet size. In contrast, Set 2 shows a very weak relationship with packet size. As a result, we decided to disregard Set 1 due to its susceptibility to the influence of packet size and instead, concentrate our analysis on Set 2.

TABLE 6. Precision, Recall and F1 score results of J48 algorithm with Set 2.

Application types	J48		
	Avg. Precision	Avg. Recall	Avg. F1 score
Audio	0.89	0.89	0.89
Browsing	0.99	0.99	0.99
Chat	0.94	0.93	0.93
Email	0.80	0.80	0.80
FTP	0.97	0.97	0.97
P2P	0.99	0.99	0.98
VDO	0.93	0.93	0.93
VoIP	0.96	0.97	0.96
Average	**0.93**	**0.93**	**0.93**

Table 6 presents the precision, recall, and F1 score results of the J48 classifier employing Set 2 features. The classifier displayed high performance across various application types, achieving average precision, recall, and

F1 scores of 0.93. This uniformity in scoring underscores the classifier's effectiveness in accurately identifying Tor traffic. The individual application type results are consistent, with scores ranging from 0.80 to 0.99, indicating some variability in performance depending on the application type. On average, the J48 classifier demonstrates overall effectiveness with an aggregate average score of 0.93 across all metrics, strengthening its reliability in classifying Tor traffic across diverse applications.

TABLE 7. Comparative analysis of model performance in related studies.

Approach	Model	Performance
Time-based features using multiple packets	C4.5 with IG+RK	PR and RC >0.9 [8]
	JRip	PR and RC = 1 [9]
	CFS-ANN	Acc = 99.8% [12]
Hex-based of packet header using a single packet	1D-CNN	Acc = 100% [13]
Hex frequency-based of encrypted payload using a single packet	J48	Acc = 95.65% [Our study]

Table 7 provides a comprehensive overview of various studies on Tor traffic classification, focusing on different approaches and their model performances. The approach involving time-based features obtained from multiple packets has demonstrated exceptional scores in precision, recall, and accuracy, surpassing the results of our method. However, this approach has its challenges. Factors such as network sensitivities, including asymmetric routing, can undermine the reliability of time-based features. Additionally, generating features from multiple packets may lead to complexities in real-time processing and increased computational load.

On the other hand, the approach that utilises hex-based features from packet headers of a single packet exhibits promising results, Its deep learning algorithms may face interpretability issues. Our study adopts a novel strategy, focusing on hex frequency-based analysis of encrypted payloads from a single packet. This method achieved an average accuracy of 95.65%, which, while slightly lower than some of the other methods, offers significant advantages. Our approach addresses the limitations of the aforementioned methods by enhancing reliability in diverse network conditions and ensuring more straightforward interpretability of the model's decision-making process. This balance of accuracy and practical usability positions our method as a viable alternative in the domain of Tor traffic classification.

2) PREDICTION RESULTS

To prevent overfitting—a crucial aspect of developing robust machine learning models—it's essential to evaluate the finalised model using new, unseen data, ensuring it isn't merely memorising the training data. In our study, due to a lack of additional unseen data, we allocated 5% of the balanced data as unseen data to assess the model's performance as discussed earlier in the Data Pre-processing step.

TABLE 8. Unseen dataset testing results with finalized model.

Application Types	#Unseen Instances	Accuracy (%)
Audio	1,372	90.59
Browsing	3,786	99.15
Chat	342	93.57
Email	648	93.36
FTP	27,102	96.99
P2P	22,830	98.99
VDO	1,692	98.29
VoIP	38,810	99.80
Average		**98.06**

Table 8 presents the results of testing the finalised model with an unseen dataset. Each application type was evaluated on the number of unseen instances and the corresponding accuracy. The model demonstrated high accuracy across all application types, with the lowest accuracy achieved for Email at 93.36% and the highest for VoIP at 99.80%. The average accuracy across all application types was 98.06%. These results demonstrate the reliability and generalisability of the finalised model, as it successfully classified previously unseen data with high accuracy, ensuring its capability to classify new instances effectively.

VII. DISCUSSION

This section provides various dimensions that underpin our study essential for a comprehensive understanding. We'll address threats to validity, discuss potential success factors, detail the key role of feature selection, and directions for future research.

A. THREATS TO VALIDITY

To ensure the credibility and validity of our research, several precautionary measures were taken during the data analysis process:

- *Data Sets and Sample Size:* We utilized large data sets to strengthen the robustness of our results. The employment of multiple datasets ensures that our findings are not restricted to just a single data source but have broader applicability and generalizability.
- *Statistical Analysis:* The assumptions requisite for our inferential tests were met, ensuring the reliability of our statistical conclusions.
- *Machine Learning:* We integrated cross-validation techniques for the validity of our results. Model robustness was assessed through performance metrics like precision, recall, and the F1-score. Additionally, the

model's capability was further validated by evaluating its performance on new, unseen data sets during the prediction phase.

Through these rigorous measures, we aimed to minimize threats to the validity of our research, ensuring our findings are both dependable and replicable.

B. POTENTIAL SUCCESS FACTORS

While our results challenge the encryption theory's assumption that ciphertexts shouldn't be distinguishable from random strings of the same length, our method refrains from disclosing message content. Instead, it differentiates Tor and nonTor traffic through distinctive attributes. Various factors contribute to our hex character statistics-based method in identifying Tor traffic type efficiency.

- *Packet size and data splitting process:* Within the Tor network, packets of varying sizes undergo an initial split into 512-byte fixed-size cells before encryption at the application layer. Conversely, in nonTor networks, packets are first encrypted, followed by a data-splitting process. This splitting can occur as either segmentation (at the TCP layer) or fragmentation (at the IP layer).
- *Three-Layer Encryption Scheme in Tor Network:* A significant variation between Tor and nonTor traffic is their encryption schemes. Typically, nonTor networks employ single-layer encryption at the application layer. In contrast, the Tor network adds a sophisticated multi-layered encryption scheme to the originally encrypted payload.
- *Distinct Encryption Algorithms and Parameters:* Tor and nonTor networks implement different encryption algorithms and parameters, leading to distinctive characteristics in their encrypted traffic.
- *Homogeneous Traffic Pattern in Tor:* In the Tor network, packets are generated by a uniform encryption algorithm, while nonTor packets originate from a multitude of applications, each using distinct encryption protocols. This creates a more homogeneous traffic pattern in Tor compared to nonTor networks.

While our proposed explanations for the effectiveness of our hex character statistics analysis approach are grounded in plausible theories, they are currently assumptions. Further research is required to validate these hypotheses and enhance our understanding of the differences between Tor and nonTor traffic.

C. SELECTION OF FEATURES

The results demonstrate that both sets of features are effective in efficiently classifying Tor and nonTor traffic using the selected algorithms, especially J48 and RF. While Set 1 performed well in classification, it was found to be influenced by packet size. On the other hand, Set 2 achieved slightly lower classification performance but was not affected by packet size. However, it required more computational resources for feature computation compared to Set 1. As a result, it is important to consider the trade-off between classification accuracy and computational resource requirements when choosing between the two feature sets.

D. FUTURE WORK

Despite the promising results, there are opportunities for future research to further refine and enhance the performance of our classification model. One vital area for improvement is feature selection. In this study, we used a comprehensive set of features from Set 2. However, not all features might contribute equally to the classification task; some might only have a limited impact or could introduce noise. By employing feature selection techniques, such as Recursive Feature Elimination (RFE), Mutual Information, or Feature Importance from tree-based models, we can systematically prune the less relevant features. This could not only reduce the computational resource requirements but may also improve classification accuracy, offer a safeguard against overfitting, and enhance the model for better interpretability.

VIII. CONCLUSION

This study has provided insights into the distinct characteristics of encrypted payload of Tor and nonTor networks from two key perspectives. Firstly, by employing the Mann-Whitney U test for statistical analysis, the research unveiled a remarkable differentiation rate of 94.53% among the 256 analysed features. This effectively highlights the substantial variations of character analysis between encrypted payloads of Tor and nonTor traffic, leading to the second aspect: the classification analysis. This phase demonstrated robust accuracy scores across diverse feature sets and algorithms with the highest average accuracy for Tor traffic classification reaching 96.53% using Set 1 with RF and 95.65% using Set 2 with J48. The evaluation of the model with an unseen dataset further affirmed its reliability, achieving the highest average accuracy of 98.06%. Our findings challenge conventional encryption theory assumptions that the encrypted data should leak no information, emphasising the efficiency of our approach in differentiating between traffic types based on encrypted payload while maintaining message content confidentiality. While our justification is theoretically grounded, our explanations need further exploration for validation. Future research could extend this methodology to diverse encryption protocols within various applications. Overall, this work introduces a novel approach to Tor traffic analysis, focusing on a single encrypted payload, independent of its position and flow features within the traffic flow. It marks a substantial advancement in network security enhancement and monitoring techniques, overcoming the limitations of conventional methods that rely on the computation of flow features from multiple packets.

REFERENCES

[1] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second generation onion router," in *Proc. 13th USENIX Secur. Symp. (SSYM)*, San Diego, CA, USA, Aug. 2004, pp. 303–320.

[2] C. Johnson, B. Khadka, E. Ruiz, J. Halladay, T. Doleck, and R. B. Basnet, "Application of deep learning on the characterization of tor traffic using time based features," *J. Internet Serv. Inf. Secur.*, vol. 11, no. 1, pp. 44–63, 2021.

[3] O. Salman, I. H. Elhajj, A. Kayssi, and A. Chehab, "A review on machine learning-based approaches for Internet traffic classification," *Ann. Telecommun.*, vol. 75, nos. 11–12, pp. 673–710, Dec. 2020.

[4] J. Barker, P. Hannay, and P. Szewczyk, "Using traffic analysis to identify the second generation onion router," in *Proc. IFIP 9th Int. Conf. Embedded Ubiquitous Comput.*, Melbourne, VIC, Australia, Oct. 2011, pp. 72–78.

[5] S. Rezaei and X. Liu, "Deep learning for encrypted traffic classification: An overview," *IEEE Commun. Mag.*, vol. 57, no. 5, pp. 76–81, May 2019.

[6] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*, 2nd ed. New York, NY, USA: CRC Press, 2020.

[7] P. Choorod and G. Weir, "Tor traffic classification based on encrypted payload characteristics," in *Proc. Nat. Comput. Colleges Conf. (NCCC)*, Mar. 2021, pp. 1–6.

[8] A. H. Lashkari, G. D. Gil, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of Tor traffic using time-based features," in *Proc. 3rd Int. Conf. Inf. Syst. Secur. Privacy (ICISSP)*, Porto, Portugal, Feb. 2017, pp. 253–262.

[9] A. Cuzzocrea, F. Martinelli, F. Mercaldo, and G. Vercelli, "Tor traffic analysis and detection via machine learning techniques," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Boston, MA, USA, Dec. 2017, pp. 4474–4480.

[10] D. Sarkar, P. Vinod, and S. Y. Yerima, "Detection of tor traffic using deep learning," in *Proc. IEEE/ACS 17th Int. Conf. Comput. Syst. Appl. (AICCSA)*, Nov. 2020, pp. 1–8.

[11] N. Rust-Nguyen, S. Sharma, and M. Stamp, "Darknet traffic classification and adversarial attacks using machine learning," *Comput. Secur.*, vol. 127, Apr. 2023, Art. no. 103098.

[12] E. Hodo, X. Bellekens, E. Iorkyase, A. Hamilton, C. Tachtatzis, and R. Atkinson, "Machine learning approach for detection of nonTor traffic," in *Proc. 12th Int. Conf. Availability, Rel. Secur. (ARES)*, New York, NY, USA, 2017, pp. 1–6.

[13] M. Kim and A. Anpalagan, "Tor traffic classification from raw packet header using convolutional neural network," in *Proc. 1st IEEE Int. Conf. Knowl. Innov. Invention (ICKII)*, Jeju Island, South Korea, Jul. 2018, pp. 187–190.

[14] W. Diffie and M. E. Hellman, "New directions in cryptography," in *Democratizing Cryptography: The Work Whitfield Diffie Martin Hellman*. New York, NY, USA: ACM, 2022, pp. 365–390.

[15] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 26, no. 1, pp. 96–99, Jan. 1983.

[16] W. Goralski, *The Illustrated Network: How TCP/IP Works in a Modern Network*, 2nd ed. Cambridge, MA, USA: Morgan Kaufmann, 2017.

[17] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: An update," *ACM SIGKDD Explorations Newslett.*, vol. 11, no. 1, pp. 10–18, 2009.

[18] N. Bhargava, G. Sharma, R. Bhargava, and M. Mathuria, "Decision tree analysis on J48 algorithm for data mining," *Proc. Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 3, no. 6, pp. 1114–1119, 2013.

[19] A. Cutler, D. R. Cutler, and J. R. Stevens, "Random forests," in *Ensemble Machine Learning: Methods and Applications*. New York, NY, USA: Springer, 2012, pp. 157–175.

[20] V. K. Ayyadevara, "Random forest," in *Pro Machine Learning Algorithms: A Hands-On Approach to Implementing Algorithms in Python and R*. Berkeley, CA, USA: Apress, 2018, pp. 105–116.

[21] M. H. D. M. Ribeiro and L. dos Santos Coelho, "Ensemble approach based on bagging, boosting and stacking for short-term prediction in agribusiness time series," *Appl. Soft Comput.*, vol. 86, Jan. 2020, Art. no. 105837.

[22] I. Syarif, E. Zaluska, A. Prugel-Bennett, and G. Wills, "Application of bagging, boosting and stacking to intrusion detection," in *Proc. 8th Int. Conf. Mach. Learn. Data Mining Pattern Recognit. (MLDM)*. Berlin, Germany: Springer, 2012, pp. 593–602.

[23] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z.-H. Zhou, M. Steinbach, D. J. Hand, and D. Steinberg, "Top 10 algorithms in data mining," *Knowl. Inf. Syst.*, vol. 14, pp. 1–37, Jan. 2008.

[24] X. Luo, "Efficient English text classification using selected machine learning techniques," *Alexandria Eng. J.*, vol. 60, no. 3, pp. 3401–3409, Jun. 2021.

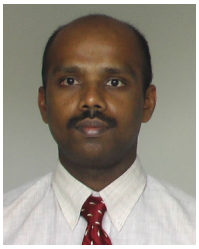
[25] F. Falck, J. Marstaller, N. Stoehr, S. Maucher, J. Ren, A. Thalhammer, A. Rettinger, and R. Studer, "Measuring proximity between newspapers and political parties: The sentiment political compass," *Policy Internet*, vol. 12, no. 3, pp. 367–399, Sep. 2020.

[26] M. Singh, A. K. Jakhar, and S. Pandey, "Sentiment analysis on the impact of coronavirus in social life using the BERT model," *Social Netw. Anal. Mining*, vol. 11, no. 1, p. 33, Dec. 2021.

[27] P. E. McKnight and J. Najab, "Mann–Whitney U test," in *The Corsini Encyclopedia of Psychology*, 1st ed. Hoboken, NJ, USA: Wiley, 2010, p. 1.

PITPIMON CHOOROD (Member, IEEE) received the B.Sc. degree in computer science from the Prince of Songkla University, Thailand. She is currently pursuing the Ph.D. degree with the Department of Computer and Information Sciences, University of Strathclyde, U.K. Her research interests include the areas of computer networking, cybersecurity, and machine learning.

GEORGE WEIR (Senior Member, IEEE) received the Postgraduate Diploma degree from the University of Strathclyde, the Master of Arts degree from the University of Glasgow, and the Ph.D. degree from the University of Edinburgh. He has taught computer science at the University of Strathclyde for more than 30 years. During this time, he has supervised many successful Ph.D. students and authored more than two hundred publications in the areas of human–computer interaction, education, applied language technology, information security, cybercrime, and digital forensics. Recently, he has worked closely with colleagues in the U.K., Canada, and Saudi Arabia on the application of textual analysis to content classification. He is currently an Adjunct Professor with the School of Criminology, Simon Fraser University, Canada, and a Visiting Lecturer in information security with the University of Colombo School of Computing, Sri Lanka. He is a fellow of the Sir Winston Churchill Memorial Trust and the Higher Education Academy and an Associate Member of the Association of Certified Fraud Examiners. In addition, he was elected as a Local Government Councillor in South Ayrshire.



ANIL FERNANDO is currently a Video Coding and Communications Professor with the Department of Computer and Information Sciences, University of Strathclyde, where he also leads the research team. He has worked on major national and international multidisciplinary research projects and led most of them. He has published more than 350 papers in international journals and conference proceedings and published a book on 3D video broadcasting. He has supervised more than 56 Ph.D. students and 30 research fellows. He has contributed to a large number of international research projects and coordinated a few international and national projects. His main research interests include video coding and communications, machine learning (ML), artificial intelligence (AI), signal processing, networking and communications, interactive systems, resource optimizations, distributed technologies, media broadcasting, quality of experience (QoE), and intelligent systems.