CM-UTC: A Cost-sensitive Matrix based Method for Unknown Encrypted Traffic Classification

Zhiyuan Gao, Jinguo Li*, Liangliang Wang, Yin He and Peichun Yuan

College of Computer Science and Technology, Shanghai University of Electric Power, Shanghai 201306, China
*Corresponding author: lijg@shiep.edu.cn

Deep learning has been widely adopted in the field of network traffic classification due to its unique advantages in handling encrypted network traffic. However, most existing deep learning models can only classify known encrypted traffic that has been sampled and labeled. In this paper, we propose CM-UTC, a cost-sensitive matrix-based method for classifying unknown encrypted traffic. CM-UTC explores the probability distribution of the DNN output layer to filter out the unknown classes and further designs a cost-sensitive matrix to address the class imbalance problem. Additionally, we propose the utilization of the Harris Hawk optimization algorithm to modify the model parameters and improve its performance. The experiments are validated on two different datasets, and the results demonstrate that CM-UTC not only outperforms existing methods in terms of overall performance but also exhibits superior capability in correctly identifying samples from the minority class.

Keywords: cost-sensitive learning; Harris Hawk optimization algorithm; encrypted traffic classification

1. INTRODUCTION

With the advancement of technology, there is a growing shift towards more secure network connections [1]. This includes the adoption of secure encrypted transport protocols, such as SSL/TLS [2], to encrypt transmitted data, and the use of anonymous communication networks (Tor) for internet access [3]. While encryption is instrumental in safeguarding user privacy and ensuring secure data transmission, it also presents opportunities for malicious users to conceal their activities and engage in illicit actions [4].

Consequently, effective management of encrypted traffic [5] has become crucial for maintaining network security. Encrypted traffic detection technology plays a vital role in enabling network administrators to identify encrypted traffic and monitor its behaviour, facilitating the timely detection of unusual activities or potential security threats.

Through the analysis and detection of encrypted traffic, network administrators are better equipped to safeguard their networks against the risks of attacks and data breaches. This evolving landscape necessitates higher levels of accuracy and effectiveness in traffic classification. Notably, classifying encrypted traffic presents a significant challenge, as it requires the classification of encrypted packets [6], often demanding greater computational and storage resources [7].

The main techniques used to detect encrypted traffic include traditional methods of traffic classification, machine learning-based methods and deep learning-based methods.

Traditional methods of traffic classification are primarily reliant upon ports and payloads. Port-based classification techniques [8] were the initial approach applied to traffic analysis. However, they are known to be less precise for applications that utilize unusual port numbers, such as port masquerading [9] and dynamic port allocation [10]. Experimental tests conducted by Madhukar et al. [11] have confirmed that approximately 70% of network traffic cannot be accurately classified solely using port-based identification methods. In contrast, payload-based techniques such as Deep Packet Inspection (DPI) utilize patterns including signatures and regular expressions of predefined protocols to identify differences between protocols [12]. These methods excel at identifying protocols, and can effectively classify complex encrypted traffic. Although these methods perform well in identifying protocols and categorising intricate encrypted traffic, DPI is not suitable for detecting encrypted traffic because it necessitates examining the traffic content [13]. Additionally, both port- and payload-based approaches prove to be inefficient and imprecise, owing to the evasion techniques employed by several peer-to-peer (P2P) applications, such as dynamic port numbers, masquerading techniques and data encryption, which are used to avoid detection.

Machine learning methods can overcome these issues. For example, Niu et al. [7] proposed an approach based on Improved Adaptive Random Forest for adaptive online analysis, which allows for the adaptive updating of parameters when dealing with new types of malicious traffic in traffic flows. Orsolic et al. [14] developed a system called YouQ to estimate customer-perceived Quality of Experience (QoE) based on network traffic characteristics at the transport layer. However, there are still certain limitations that need to be considered. The manual extraction and feature selection process, coupled with the reliance on domain experts’ knowledge, introduce considerable uncertainty when applying machine learning to the classification of encrypted traffic [15]. Moreover, the effectiveness of these features tends to degrade over time, necessitating regular updates and adjustments to maintain accurate classification.

In contrast, deep learning models, such as convolutional neural networks (CNNs) and LSTMs, offer the advantage of automatically extracting meaningful features from raw data, avoiding the complexity and subjectivity of feature engineering. This gives
deep learning models a significant advantage in traffic classification tasks. This inherent capability of deep learning models makes them well suited for traffic classification tasks, leading to their increasing adoption by researchers and yielding promising results. For example, Lu et al. [16] proposed the Inception-LSTM (ICLSTM) method for encrypted traffic classification. This method transforms traffic data into grey-scale images and leverages the constructed ICLSTM neural network to extract essential features for effective traffic classification. Shen et al. [17] proposed Deep-QoE, a real-time video QoE measurement system that operates on encrypted traffic. DeepQoE utilizes a CNN with a complex input structure and architectural design. Wang et al. [18] developed a multi-task learning model, named MTC, which combines Transformer and 1D-CNN to classify traffic classification. The MTC model captures long-range feature dependencies through Transformer while considering local feature details.

These examples have demonstrated the efficacy of deep learning methods in traffic classification, as they automatically learn discriminative features directly from the data. Researchers have achieved notable advancements in encrypted traffic classification and video QoE measurement by leveraging CNNs, LSTMs and hybrid architectures. Continued exploration and refinement of deep learning approaches is crucial for further improving the accuracy and robustness of traffic classification systems.

It is worth noting that the above studies still face the following challenges:

1. Data collection represents a crucial stage in the network traffic classification process [7, 16]. Nonetheless, labelling data is a challenging undertaking in the field of network traffic analysis. However, inadequate data collection or an unrepresentative dataset can impair the classifier’s efficacy. Additionally, traffic classification necessitates a considerable number of labelled samples for supervised learning.

2. These methods lack the ability to identify unseen categories in the dataset, which becomes a hurdle as the variety of traffic types continues to expand, including unknown types, along with the growing demand for finer granularity classification.

3. Network traffic classification demands higher accuracy and effectiveness [19]. While supervised learning and DL models are widely used in traffic classification, there is a need for improved interpretability and scalability. Encrypted traffic classification presents a challenging task that requires higher computational and storage resources. As the range of network traffic types widens and the need for greater precision increases, there is an increasing requirement for ongoing exploration and innovation. These endeavours are imperative in enhancing our understanding and analysis of network traffic patterns and behaviours.

To address the aforementioned issues, we propose CM-UTC, a Cost-sensitive Matrix and deep learning based method for Unknown Encrypted Traffic Classification. Our framework aims to handle the identification and classification of unknown encrypted traffic. It combines a deep neural network (DNN) with cost-sensitive learning to address the class imbalance problem, and introduces the Harris Hawk optimization algorithm to optimize the parameters for improving performance. In this approach, a cost-sensitive matrix is utilized, assigning a cost to each misclassification based on a confusion matrix. During the training phase, this cost matrix is applied to the network to effectively improve the classification of unknown traffic.

The traffic classification method proposed in this paper offers several advantages over other classification schemes:

- We propose the CM-UTC method for unknown traffic classification, which address the challenges such as slow convergence and falling into local minima, incurred by the gradient-based DNNs. CM-UTC is based on DNNs, eliminating the need for experts to manually extract network traffic features, which can be automatically extracted from the hidden layers of the DNN.

- To the best of our knowledge, there has been no prior investigation into the integration of cost-sensitive learning with DNNs for the classification of unknown encrypted traffic, specifically aimed at addressing the challenge posed by class imbalance. In this study, we propose a novel approach that involves generating a cost matrix derived from the class distribution. This cost matrix is then utilized during the model training process and weight updates. The incorporation of cost-sensitive learning effectively mitigates the class imbalance problem encountered in this context.

- To assess the effectiveness of the proposed approach in classifying encrypted traffic and identifying unknown traffic, we used the ISCX VPN 2016 dataset [20] and dataset2 [19] in our experiments. The results obtained demonstrate that CM-UTC not only outperforms existing methods in terms of overall performance but also exhibits superior capability in correctly identifying samples from the minority class. Furthermore, we compare our proposed method with CNN [21], DNN [11] and the latest MTC [18] methods using this dataset. The empirical findings clearly indicate that CM-UTC achieves superior classification performance compared to deep learning-based methods.

The remainder of this paper is structured as follows. In Section 2, we provide a summary of related research in the field. Section 3 presents the foundational knowledge necessary for understanding the proposed method. The detailed description of the CM-UTC method is provided in Section 4. Section 5 outlines the experimental setup and presents the evaluation of the obtained results. Finally, Section 6 concludes the paper.

2. RELATED WORK

In recent years, encrypted traffic classification has remained a prominent research area, with continuous generation of novel research findings. In recent years, the classification of encrypted traffic has remained a prominent area of research, with a continuous flow of new research results.

2.1. Traditional encrypted traffic classification

Extensive research has been conducted on traditional encrypted traffic classification, encompassing methodologies for both port-based and payload-based traffic classification.

The port-based method relies on the TCP/UDP header’s port assignment to match the default port number specified in a document published by the Internet Assigned Numbers Authority [22]. This method is simple, fast [23] and is commonly employed in firewalls and access control lists for traffic classification. Nevertheless, it confronts various difficulties, such as port misuse, port hopping, network address translation and arbitrary port usage [24], resulting in a lessened classification effectiveness. The existing classification method based on ports [11, 22] can only accurately categorize a limited percentage, between 30 and 70%,
of online data. Consequently, catering to the comprehensive and multifaceted classification requirements of modern networks is problematic.

Payload-based traffic classification methods are commonly known as DPI [25]. DPI is a cryptographic traffic classification method that employs application layer information for categorization. It utilizes patterns [12], including signatures and regular expressions of predefined protocols, to differentiate between them. This technique demonstrates strong performance in protocol identification and can effectively classify intricate encrypted traffic. However, deep packet inspection (DPI) is inappropriate for detecting encrypted traffic as it necessitates inspecting the content of the traffic [13]. Callegati et al. [26] suggested a Man-In-The-Middle technique to decipher encrypted traffic data, thereby enabling conventional detection methods to remain valid for decrypted data. Nevertheless, this approach conflicts with the privacy-preserving feature of traffic encryption, and the process of decryption and re-encryption consumes considerable resources. Sherry et al. [27] put forth a system to tackle the issue of privacy in deep packet inspection by examining the encrypted payload without decryption. This approach can efficiently safeguard user privacy during deep inspection, leading to an improved network security. However, it only applies to HTTP secure traffic and cannot be implemented for other forms of traffic. More efficient and secure algorithms must be developed to address the privacy issue for other forms of encrypted traffic.

2.2. Machine learning-based classification of encrypted traffic

Detection methods based on machine learning algorithms show unique advantages in detecting encrypted traffic compared to detection methods that decrypt ciphertext. These methods leverage machine learning algorithms to analyze features extracted from traffic without the need for decryption. These features can be categorized into three main types: metadata features, statistical features and unencrypted Transport Layer Security (TLS) header features. Metadata features provide essential information about the data flow, such as a five-tuple. Statistical features are derived from statistical analysis of the traffic flow and require calculations to extract meaningful information. When traffic is encrypted using the TLS protocol, unencrypted TLS header features come into play. These features encompass SSL connection details, certificate information, and more. McGrew et al. [28, 29] extract various features, including metadata features, packet length, byte distribution and TLS header information from encrypted traffic. They employ a logistic regression-based classifier to identify encrypted traffic. Additionally, Anderson et al. [30, 31] introduce additional learning features by incorporating background traffic, such as DNS context flows linked to encrypted flows, and headers of HTTP context flows from the same source IP within a 5-min window. Torroledo et al. [32] focus on extracting self-signed or freely generated certificates from encrypted network sessions and employ an RNN + LSTM model to classify the certificates.

Machine learning methods have been widely adopted for traffic classification by many researchers, such as Bayesian neural networks [33], Naive Bayesian classifiers [22] and artificial neural networks [34]. The two most well-known methods applied to the ISCX VPN 2016 traffic dataset are also based on machine learning methods, namely C4.5 decision trees [20] and k-nearest neighbor techniques [35]. These methods have achieved high accuracy and recall rates, demonstrating good performance and scalability. They achieve an average classification accuracy of 90%, but the extraction of stream features requires a relatively high computational overhead. However, machine learning-based classifiers typically exhibit lower classification accuracy and require manual feature selection [12].

2.3. Deep learning-based classification of encrypted traffic

Deep learning-based traffic classifiers have gained significant attention in various domains, including computer vision and natural language processing [36-40]. For instance, Li et al. [38] proposed an attention-inclusive byte segment neural network that extracts features from payload segments and utilizes a softmax classification layer for classification. Aceto et al. [41] introduced Mimetic, which classifies mobile encrypted traffic using modular deep learning. These aforementioned methods require complex feature extraction and challenging model training, which can limit their potential for generalization.

In addition, several deep learning-based end-to-end classification models have been proposed in the literature, aiming to enhance the accuracy and efficiency of traffic classification. For instance, Liu et al. [39] developed FS-Net, which integrates recurrent neural networks and self-encoders for traffic classification and packet feature mining. Wang et al. [42] presented an integrated framework for feature extraction, feature selection and classifier sets, demonstrating its effectiveness through comparative experiments. Moreover, Lotfollahi et al. [43] proposed a deep packet framework that combines stacked autoencoders (SAE) with one-dimensional convolutional neural networks (1D-CNN) for network traffic classification, achieving superior classification results on various datasets.

Clearly, most prior cryptographic traffic classification has centred around the closed-world assumption, which hinders the classification of packets in static datasets. However, under the open world assumption, these deep learning-based classifiers must be regularly updated to ensure accurate traffic classification and support network measurement and management.

Therefore, our research endeavours to improve the state of the art by accurately identifying previously trained traffic classes and detecting as well as classifying previously unknown classes as unidentified data.

3. METHODS

3.1. Cost-sensitive learning

Cost-sensitive learning is a type of machine learning that takes into account the costs associated with different types of misclassification when training a model. In traditional machine learning, misclassification is considered equally expensive. However, in cost-sensitive learning, misclassification may incur different costs depending on the context or application. By incorporating these costs into the learning process, cost-sensitive learning can produce more accurate and cost-effective models. Some common techniques for cost-sensitive learning include adjusting decision thresholds, modifying classification algorithms and re-weighting training data.

Cost-sensitive loss is a loss function used in cost-sensitive learning that penalizes misclassification to varying degrees depending on the type of classification error.

In cost-sensitive learning, different types of errors may lead to different costs. For example, misdiagnosing a serious disease...
as healthy can be very costly. Therefore, cost-sensitive loss takes into account different types of errors and costs, and minimizes the overall cost by adjusting the parameters of the model.

For example, for a binary classification task, the cost-sensitive matrix can be expressed as Table 1:

<table>
<thead>
<tr>
<th>Cost-sensitive matrix</th>
<th>True label is 1</th>
<th>True label is 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predicted label is 1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Predicted label is 0</td>
<td>10</td>
<td>0</td>
</tr>
</tbody>
</table>

In this context, the cost-sensitive matrix is utilized to quantify the model’s incurred cost for misclassifying a given category. Each element of the matrix represents the cost incurred by the model when it incorrectly predicts the category shown in the column as the category shown in the corresponding row, which represents the true label. For example, if the true label is 0 and the model predicts it as 1, the cost would be 10; similarly, if the true label is 1 and the model predicts it as 0, the cost would be 0. Depending on the specific values in the cost-sensitive matrix, the cost-sensitive loss function can penalize different types of errors to varying degrees.

### 3.2. Deep neural networks

A DNN is a multi-layer unsupervised neural network that uses the output features from one layer as input to the next layer for feature extraction. It aims to enhance the representation of the input data by progressively mapping spatial samples onto a different feature space across successive layers. DNNs comprise multiple layers with non-linear mappings, enabling them to model highly complex functions. If the deep structure is considered as a network of neurons, the fundamental concept of a DNN can be summarized in the following three points:

- Each layer of the network undergoes pre-training using unsupervised learning techniques.
- Unsupervised learning is applied layer by layer, utilizing the previous layer’s output as the input for the subsequent layer.
- Fine-tuning of all layers, along with the inclusion of a classifier for classification tasks, is accomplished through supervised learning.

DNNs and traditional neural networks differ primarily in their training mechanisms. To address the limitations of traditional neural networks, such as the tendency to overfit and slow training speed, DNNs employ a layer-by-layer pre-training mechanism instead of the traditional neural networks’ back-propagation training mechanism as a whole.

DNNs offer several advantages for cryptographic traffic classification. These advantages include the following:

- Automating feature design: DNNs automatically learn and extract features from raw data, eliminating the need for researchers to manually design features.
- Layer-wise pre-training: DNNs can be pre-trained layer by layer, allowing each layer to learn primary features from the data. This hierarchical representation of features can help identify unique patterns and reduce the similarity between papers by capturing different aspects of the problem at each layer.
- Exponential distributed data learning: DNNs excel at learning from large amounts of data. By leveraging distributed computing and parallel processing, DNNs can effectively process and learn from vast datasets. This enables researchers to train models on diverse datasets.

- Enhanced representation of complex problems: DNNs have the capability to capture complex non-linear relationships in the data. They can represent intricate interactions between features, resulting in a more detailed and efficient representation of cryptographic traffic classification. This enables researchers to explore different aspects of the problem and generate diverse solutions.

Our study findings validate this assertion and provide evidence of the effective feature extraction capabilities of the proposed model in network traffic classification.

### 3.3. Harris Hawks Optimization

Harris Hawks Optimization (HHO) is an emerging optimization algorithm inspired by the hunting behaviour of groups of Harris Hawks during predation. The algorithm is based on the behavioural characteristics of Harris hawks, which include searching and exploitation. By modeling these behaviours, the HHO algorithm can efficiently and quickly find the global optimal solution.

The basic steps of HHO are as follows:

**Step 1**: Population initialization. Initialize each individual within the search space’s upper and lower bounds.

**Step 2**: Fitness calculation. Evaluate the fitness of each individual’s position and select the one with the best fitness as the current prey position.

**Step 3**: Position update. Update individual positions based on the prey escape energy and a randomly generated number, employing either search or exploitation behaviours:

- Search behaviour: randomly select another eagle and move the current individual towards that eagle’s position.
- Exploitation behaviour: calculate and update the next position using the current individual’s position and the global optimal position.

**Step 4**: Fitness calculation. Calculate the fitness of each individual after the position update and compare it with the current prey’s fitness. If an individual’s fitness surpasses the prey’s, replace the prey position with the individual’s position.

**Step 5**: Stopping condition check. If the stopping condition is not met, return to step 3. Once the maximum number of iterations is reached, output the current prey position as the estimated target position.

By continuously updating individual positions and fitness, the HHO algorithm is capable of quickly finding the global optimal solution in the search space. Additionally, by incorporating strategies such as escape energy, search behaviour and exploitation behaviour, it effectively avoids being trapped in local optimal solutions. Compared to other optimization algorithms, the HHO algorithm offers advantages such as fast convergence, strong global search capability and ease of implementation. As a result, it has been successfully applied to various optimization problems.

### 4. PROPOSED METHODOLOGY

As imbalanced data significantly affects the efficiency of DNNs and can lead to overfitting during training [44], various methods have been proposed to address this issue. Existing methods [43] typically employ resampling techniques that rely on expert knowledge to identify majority and minority categories. However, these techniques are time-consuming, costly and have drawbacks.
They often result in the removal of numerous data patterns and the generation of incorrect data patterns. Therefore, we introduce a method called CM-UTC for cryptographic stream classification tasks. The proposed method is depicted schematically in Fig. 1.

The proposed method comprises four distinct phases. In the first phase, a pre-processing step is performed to convert the raw traffic data into the necessary input format for training purposes. The second phase involves training the CM-UTC model using the input data to identify unknown classes. The third phase computes the loss for the model and updates the weights. The final phase uses the SoftMax function to classify the data.
Table 2. List of captured protocols and applications.

<table>
<thead>
<tr>
<th>Traffic</th>
<th>Content</th>
<th>Number of samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chat</td>
<td>ICQ, AIM, Skype, Facebook, and Hangouts</td>
<td>4973</td>
</tr>
<tr>
<td>VoIP</td>
<td>Facebook, Skype, and Hangouts voice calls (1-h duration)</td>
<td>19 358</td>
</tr>
<tr>
<td>P2P</td>
<td>uTorrent and Transmission (Bittorrent)</td>
<td>447</td>
</tr>
<tr>
<td>Streaming</td>
<td>Vimeo and YouTube</td>
<td>3810</td>
</tr>
<tr>
<td>File Transfer</td>
<td>Skype, FTPS, and SFTP using Filezilla and an external service</td>
<td>1373</td>
</tr>
<tr>
<td>Email</td>
<td>SMTPS, POP3S, and IMAPS</td>
<td>875</td>
</tr>
</tbody>
</table>

Table 3. App features recorded during data collections.

<table>
<thead>
<tr>
<th>Traffic</th>
<th>Content</th>
<th>Number of samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facebook</td>
<td>Social networking</td>
<td>19 944</td>
</tr>
<tr>
<td>Instagram</td>
<td>Photo and video</td>
<td>6818</td>
</tr>
<tr>
<td>WhatsApp</td>
<td>Social networking</td>
<td>436</td>
</tr>
<tr>
<td>Viber</td>
<td>Social networking</td>
<td>690</td>
</tr>
<tr>
<td>Gmail</td>
<td>Productivity</td>
<td>1036</td>
</tr>
<tr>
<td>Messenger</td>
<td>Social networking</td>
<td>6061</td>
</tr>
</tbody>
</table>

In the third phase, the unknown classes are analysed, and labels are assigned to them. Finally, in the fourth stage, the effectiveness of the proposed framework in classifying unknown classes is evaluated through testing. Additional details regarding the dataset used and the proposed method are presented below.

4.1. Data collection and processing

To select a representative dataset, we utilized the ISCVVPN2016 dataset [20] and dataset from paper [19] (In this paper, it is termed Dataset2).

A crucial and commonly applied pre-processing step in DNNs is data normalization. Hence, the pre-processing of the dataset is performed to normalize and standardize the data.

4.1.1. ISCVVPN2016 dataset

The original dataset consists of 14 traffic types, including Chat, VPN Chat, VoIP, VPN VoIP and others. However, for this paper, we focused on distinguishing between VPN and non-VPN traffic, excluding Web Browsing as well. Consequently, we selected six classes for analysis: Chat, VoIP, P2P, Streaming, File Transfer and Email. The details of the selected classes are provided in Table 2.

This dataset comprises 23 characteristics for every flow instance. These statistical features of the flow’s packets include packet count, packet size, interarrival time and packet size distribution, among others. Packet size and inter-arrival time were taken into account in this study.

4.1.2. Dataset2

A comprehensive dataset [19] was created by performing a series of actions on apps such as Facebook, Instagram, WhatsApp, Viber, Gmail, Messenger, Skype and YouTube. We selected the top six categories for analysis. Table 3 provides detailed information about the selected categories.

In this work, the study examines two traffic features: the length of frames and the inter-frame arrival time, measured in bytes and seconds, respectively.

4.2. Cost-Sensitive Matrix based method for Unknown Encrypted Traffic Classification

Based on prior research, it has been inferred that employing cost-sensitive learning and training neural networks yield improved effectiveness compared to data-level approaches. Cost-sensitive learning refers to a specific area within machine learning that considers costs associated with prediction errors during model training. Consequently, as Fig. 1 shows, we introduce a novel method called CM-UTC. The proposed method aims to enhance the efficiency of traffic classification by automatically learning suitable features for both minority and majority classes. Additionally, it utilizes the meta-heuristic optimization algorithm known as HHO to discover optimal parameters.

Cost-sensitive learning is a strategy specifically developed to tackle class imbalance issues by prioritizing the misclassification of a small number of class instances, as these misclassifications have a more significant impact on the model. Unlike traditional cost functions, cost-sensitive learning employs an activated cost function to increase the surrogate value and give higher priority to a small number of class instances. Throughout the learning process, the neural network continuously adjusts its parameters to fit the training data. However, in the case of class imbalance, the parameter update process of the neural network becomes insensitive to the minority class, leading to reduced classification performance for these instances. To enhance the classification performance of minority classes, the cost-sensitive loss functions can be adjusted. This adjustment ensures that the parameter updating process of the neural network focuses more on the training of minority classes. As a result, this approach enhances the classification accuracy of minority class instances while also improving the overall performance of the classifier. Unlike previous approaches that relied on user-defined matrices, the proposed method automatically adjusts the cost per classification based on the proportion of misclassified samples in the confusion matrix, as shown in the following equation:

\[ \omega_i = \frac{N_{\text{max}}}{N_i} \]  

Here, \( N_i \) is the number of samples in class \( i \) traffic, \( N_{\text{max}} \) is the maximum number of samples in \( M \) type flows and \( \omega_i \) is the class weight, in order to assign higher costs to the few classes of misclassified samples:

\[ \text{mis}_i = \frac{\sum_{j=0}^{M} C_{ij}}{N_i}, i \neq j \]  

Here, \( C_{ij} \) is the number of samples misclassified from class \( i \) into class \( j \) and \( \text{mis}_i \) is the proportion of misclassified traffic from...
class i:
\[
    s_i = \frac{\text{mis}_i}{\sum_{i=0}^{N} \text{mis}_i} \tag{3}
\]

Here, \( s_i \) is the proportion of all misclassified traffic of category i:
\[
    \text{Cost}_{ij} = \begin{cases} 
        C_{ij} \cdot s_i \cdot \omega_{ij} & i \neq j \\
        0 & , i = j 
    \end{cases} \tag{4}
\]

Here, \( \text{Cost}_{ij} \) is the updated cost matrix.

After that, weighted losses are calculated based on the cost matrix. The cost-sensitive loss function is defined as follows:
\[
    L(y, f(x)) = \sum_{i=1}^{N} \text{Cost}_{ij} \cdot L_i(y_i, f_i(x)) \tag{5}
\]

Here, \( y_i \) is the true label of the instance for class i. \( f_i(x) \) is the predicted probability that the instance belongs to class i. \( L_i(y_i, f_i(x)) \) is the overall cost-sensitive loss function. \( N \) is the number of instances. \( L_i(y_i, f_i(x)) \) is the standard loss function (e.g. cross-entropy loss for classification problems) for predicting class i when the true class is i.

Once the model with the cost-sensitive loss function and matrix has been established, the HHO algorithm can be used to explore the optimal hyperparameters of the model. This procedure optimizes the hyperparameters to improve the performance of the model with respect to the chosen evaluation metric, taking into account the cost-sensitive effects resulting from the modified loss function.

In summary, the cost-sensitive loss function is implemented during the training phase of the model to guide the learning process, taking into account the costs associated with misclassification. The HHO algorithm fine-tunes the model’s hyperparameters, indirectly influencing the model’s performance under cost-sensitive learning. These two techniques can complement each other, resulting in a reliable and optimized model for handling imbalanced data sets or situations with varying misclassification costs.

4.3. Unknown category identification

One of the significant contributions of this research is the ability to classify untrained traffic as unknown traffic. To achieve this goal, the proposed method was employed to train the model, and the probability distribution of the output layer was utilized for detecting previously untrained traffic types.

Let C represent the output of the Softmax layer, and \( C_{\text{max}} = \max(C) \) denote the confidence score obtained by the classifier when processing a single piece of data. \( C_{\text{max}} \) represents the node with the highest probability. The decision to convert a category label based on the predicted probability relies on a parameter known as the threshold, denoted as \( \epsilon \). If \( C_{\text{max}} \) is below \( \epsilon \), the input traffic is considered to belong to an unknown category. This is because the model’s confidence level is not sufficiently high. By establishing a threshold for the positive category, it makes a determination as to whether the input data corresponds to any of the traffic types that have been trained.

To achieve a reliable and well-calibrated assessment of the model’s performance, we conducted an analysis of the confidence distribution for both known and unknown data instances. A histogram is constructed by recording the frequencies of known and unknown instances within each confidence value, as shown in Fig. 2. It is observed that the majority of known data instances concentrate near a confidence value of 1. On the other hand, the distribution of unknown data instances is more uniform. This suggests that by setting the threshold \( \epsilon \) to a probability close to 1, most of the unknown data can be effectively filtered out. It should be noted that the classifier is trained solely on samples from existing classes and does not have exposure to the unknown classes.

Through testing with multiple values of \( \epsilon \), it is observed that the model achieved the highest training accuracy, validation accuracy and detection of unknown data when \( \epsilon = 0.97 \).

While the discriminator is able to identify known classes of traffic data, the unknown data that are filtered out still need to be labeled which enables the creation of a fresh training dataset for model updates.

5. EXPERIMENTAL RESULTS

This section begins by describing the evaluation metrics used in this study, followed by an evaluation of the traffic classification performance of the CM-UTC model using the ISCX VPN2016 dataset [20] and dataset2 [19]. To showcase the effectiveness of our proposed approach, we conducted various experiments in traffic classification, as well as the identification and classification of unknown traffic. All implementations were carried out in Python, utilizing the Tensorflow framework [45] and the Keras library for the Python [46]. The hardware used for the experiments consisted of a laptop with NVIDIA GeForce RTX 3060 Laptop GPU, 12th Gen Intel Core i9-12900H 2.50 GHz, Windows 11 operating system and 16 GB of RAM in dual channel configuration. This section covers the discussion of the dataset, model evaluation methods, analysis of results and related issues.

5.1. Evaluation metrics

The traffic classification results can be categorized into four outcomes, namely True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN). TP represents the classifier correctly identifying Positive class samples as Positive, while FP indicates the classifier incorrectly classifying Negative samples as Positive. TN represents the classifier correctly identifying the negative class as negative, and FN indicates the classifier incorrectly classifying the positive class as negative.
These four metrics, TP, FP, TN and FN, are utilized in this study to evaluate the performance of CM-UTC.

\[
Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)}
\]

\[
Recall = \frac{TP}{TP + FN}
\]

\[
Precision = \frac{TP}{TP + FP}
\]

\[
F1\text{-score} = \frac{2 \times (Precision \times Recall)}{Precision + Recall}
\]

The four evaluation metrics mentioned above has gained significant popularity in prior studies. Accuracy signifies the classifier’s ability to accurately predict samples by calculating the ratio of correctly classified samples to the total number of samples. Recall measures the classifier’s performance in correctly identifying samples of the positive class, while precision quantifies the classifier’s accuracy in correctly classifying samples as positive. The F1-score, a harmonic mean of recall and precision, provides a single value ranging from 0 to 1, with higher scores indicating superior performance.

5.2. Experimental results

This section aims to verify the effectiveness of the model and compare it with other deep learning and machine learning algorithms using two separate approaches:

1. When previously trained traffic types are input into the model, it should accurately classify them.
2. When previously untrained traffic types are input into the model, it should recognize them as unfamiliar information.

5.2.1. Encrypted traffic classification

This section presents the outcomes obtained from employing the method for network traffic classification. The classification of known encrypted traffic is carried out using the model architecture proposed in the previous section.

The results of applying the methods proposed in this paper, as well as CNN [21], DNN [11] and the latest MTC [18], to the two different dataset for traffic identification and classification, are presented respectively in Tables 4 and 5. Multiple repetitions of the experiments were conducted, and the average classification accuracy was calculated based on precision, recall, and F1-score metrics, considering the available categories and instances within each category. For the six-category classification problem, the proposed method achieved an average accuracy of 92 and 90% on both datasets. As can be seen from Tables 4 and 5, the proposed method outperformed CNN, DNN and MTC significantly in terms of classification results.

In Table 4, we observe that, across all classes, both the DNN and MTC methods exhibit lower precision, especially for the P2P and Streaming classes. The CNN method, on the other hand, shows even lower precision for the P2P, File Transfer and Email classes. This decrease in accuracy can be attributed to the limited availability of training data for these specific classes. Consequently, both CNN and DNN fail to acquire sufficient features to effectively identify and classify the traffic. In contrast, our proposed method achieves remarkable accuracies of 98, 86, 8 and 95% for these respective classes. Table 5 further substantiates the effectiveness of our approach in addressing the challenge of class imbalance, as evident in dataset2 [19]. Our method leverages cost-sensitive learning to assign a higher loss to the minority class, resulting in improved accuracy in identifying these minority classes. Additionally, we optimize our method using the HHO algorithm, which aids in circumventing local optima and efficiently finding global optimal solutions. This comprehensive approach consistently demonstrates high accuracy and commendable performance when it comes to classifying various traffic types.

5.2.2. Unknown traffic classification

This section showcases the outcomes achieved through the application of the method for the identification and classification of unidentified traffic.

In this study, we address six distinct categories, each from two separate datasets. Each dataset is divided into two subsets: one for training and another for testing purposes. In each scenario, we select one category as the test dataset, while the remaining five categories are used for training. This process is replicated for all six categories, generating six unique scenarios. Subsequently, the model undergoes training using the designated training dataset. To assess the model’s capability to identify unknown traffic, we employ a separate test dataset comprising categories that were not included in the training data. This evaluation ensures that the model can effectively detect and classify traffic types it has not been explicitly trained on.

In Table 6, we use abbreviations to represent various applications, with C for Chat, V for VoIP, P for P2P, S for Streaming, F for File Transfer and E for Email applications. In Table 7, we similarly use abbreviations for popular apps, such as F for Facebook, I for Instagram, W for WhatsApp, V for Viber, G for Gmail and M for Messenger. Across all the tests outlined in the tables, our model demonstrates an average accuracy rate of 83 and 86% in effectively classifying unknown traffic. Notably, Table 6 reveals that when VoIP and P2P are utilized as test traffic, the majority of their data are correctly identified as unknown, resulting in a high detection rate. However, a significant portion of the Streaming and Email test traffic is inaccurately categorized as one of the pre-trained classes, leading to a lower detection rate. In Table 7, it is evident that most of the traffic from WhatsApp and Facebook is correctly identified as unknown, with any misclassified data predominantly attributed to Facebook.

Figures 3 and 4 present the tabular representation of the confusion matrix, showcasing the experimental outcomes obtained for the proposed method across six distinct scenarios, separately. The rows of this matrix correspond to the actual categories of each scenario, while the columns represent the predicted labels for each scenario.

In accordance with the data presented in Tables 6 and 7, our proposed approach achieved accuracy rates of 83 and 86%. These figures demonstrate a slight superiority over MTC, which achieved 75 and 81%, and CNN, which achieved 73 and 84%. Notably, our method also outperformed CNN significantly, with CNN achieving 59 and 55%. Moreover, our proposed approach exhibited substantial superiority over CNN, DNN and MTC in accurately identifying minority classes.

To summarize, the proposed method demonstrated promising outcomes compared to CNN, DNN and MTC approaches in detecting unknown traffic, particularly in recognizing the minority classes.

It is evident that CM-UTC yields superior results, particularly in scenarios involving class-imbalanced datasets. The proposed method exhibits noteworthy advantages in the following scenarios:

- Anomaly Detection Scenario: when enterprises, anomalies such as fraud or equipment malfunctions are infrequent occurrences, comprising a small fraction of the dataset.
Table 4. Encrypted traffic classification of CM-UTC, CNN, DNN and MTC (ISCX VPN2016 Dataset).

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy</td>
<td>Precision</td>
<td>Recall</td>
<td>F1-score</td>
</tr>
<tr>
<td>Chat</td>
<td>0.92</td>
<td>0.96</td>
<td>0.89</td>
<td>0.92</td>
</tr>
<tr>
<td>VoIP</td>
<td>0.92</td>
<td>0.97</td>
<td>0.95</td>
<td>0.96</td>
</tr>
<tr>
<td>P2P</td>
<td>0.98</td>
<td>0.97</td>
<td>0.97</td>
<td>0.95</td>
</tr>
<tr>
<td>Streaming</td>
<td>0.86</td>
<td>0.77</td>
<td>0.81</td>
<td>0.87</td>
</tr>
<tr>
<td>File Transfer</td>
<td>0.83</td>
<td>0.84</td>
<td>0.84</td>
<td>0.84</td>
</tr>
<tr>
<td>Email</td>
<td>0.95</td>
<td>0.71</td>
<td>0.80</td>
<td>0.81</td>
</tr>
</tbody>
</table>

Table 5. Encrypted traffic classification of CM-UTC, CNN, DNN and MTC (Dataset2).

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy</td>
<td>Precision</td>
<td>Recall</td>
<td>F1-score</td>
</tr>
<tr>
<td>Facebook</td>
<td>0.90</td>
<td>0.92</td>
<td>0.95</td>
<td>0.93</td>
</tr>
<tr>
<td>Instagram</td>
<td>0.77</td>
<td>0.74</td>
<td>0.75</td>
<td>0.75</td>
</tr>
<tr>
<td>WhatsApp</td>
<td>0.96</td>
<td>0.89</td>
<td>0.93</td>
<td>0.93</td>
</tr>
<tr>
<td>Viber</td>
<td>0.86</td>
<td>0.76</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>Gmail</td>
<td>0.82</td>
<td>0.80</td>
<td>0.81</td>
<td>0.81</td>
</tr>
<tr>
<td>Messenger</td>
<td>0.91</td>
<td>0.88</td>
<td>0.90</td>
<td>0.90</td>
</tr>
</tbody>
</table>

Table 6. Unknown traffic classification of CM-UTC, CNN, DNN and MTC (ISCX VPN2016 Dataset).

<table>
<thead>
<tr>
<th>Scenario</th>
<th>The categories used to train</th>
<th>The categories used to test</th>
<th>Unknown traffic detection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scenario A</td>
<td>V,P,S,E</td>
<td>Chat</td>
<td>0.87</td>
</tr>
<tr>
<td>Scenario B</td>
<td>C,P,S,E</td>
<td>VoIP</td>
<td>0.95</td>
</tr>
<tr>
<td>Scenario C</td>
<td>C,V,S,E</td>
<td>P2P</td>
<td>0.92</td>
</tr>
<tr>
<td>Scenario D</td>
<td>C,V,P,F,E</td>
<td>Streaming</td>
<td>0.72</td>
</tr>
<tr>
<td>Scenario E</td>
<td>C,V,P,S,E</td>
<td>File Transfer</td>
<td>0.81</td>
</tr>
<tr>
<td>Scenario F</td>
<td>C,V,P,S,F</td>
<td>Email</td>
<td>0.74</td>
</tr>
<tr>
<td>Avg</td>
<td>0.83</td>
<td>0.59</td>
<td>0.73</td>
</tr>
</tbody>
</table>

CM-UTC’s proficiency in managing class imbalances makes it an ideal solution for detecting and preventing these rare anomalies.

- Analysis of Rare Events: rare events, akin to anomalies, occur sporadically but lack predefined rules or logical patterns.
- Low-Frequency Events: these anticipated yet infrequent events necessitate precise identification. CM-UTC’s aptitude...
In conclusion, CM-UTC’s robust performance in addressing class-imbalanced data renders it an indispensable asset in scenarios involving the detection and categorization of rare, low-frequency and unpredictable events. This significantly contributes to bolstering operational integrity and security.

6. CONCLUSION

In this paper, we present CM-UTC, a DNN-based model designed for traffic classification. CM-UTC leverages cost-sensitive learning to mitigate the effects of imbalanced data. To evaluate and analyze the efficacy of our model, we conduct two sets of experiments. First, we evaluate the classification performance of known encrypted traffic using two different datasets. Second, we create six scenarios each to simulate unknown classes, with the objective of evaluating the model’s capability to detect and classify previously untrained traffic types. The experimental outcomes demonstrate the strong performance of our proposed method. Furthermore, we compare CM-UTC with CNN, DNN and MTC models. The results highlight the significant advantages of CM-UTC in handling data imbalance and classifying unknown traffic.

However, it is noteworthy that our study focuses solely on six classes and may have limitations when it comes to accurately identifying unknown encrypted traffic in a broader open-world setting, especially when dealing with a larger number of more finely grained classes. Therefore, further research is required to address this limitation.

FUNDING

National Natural Science Foundation of China (No. U1936213).

DATA AVAILABILITY

The first data underlying this article is available in IMPACT at https://doi.org/10.23721/100/1478793. The dataset was derived from sources in the public domain: Canadian Institute for Cybersecurity, https://www.unb.ca/cic/datasets/vpn.html. The second dataset was derived from sources in the public domain: https://www.dropbox.com/s/9tihcj9wx2xia1t/Dataset.7z?dl=0.

REFERENCES


