Towards Fine-Grained Webpage Fingerprinting at Scale

Xiyuan Zhao*
INSC & BNRist, Tsinghua University
Beijing, China
zhaoxy23@mails.tsinghua.edu.cn

Xinhao Deng*
INSC & BNRist, Tsinghua University
Beijing, China
dengxh23@mails.tsinghua.edu.cn

Qi Li
INSC, Tsinghua University
Zhongguancun Laboratory
Beijing, China
qli01@tsinghua.edu.cn

Yunpeng Liu
INSC, Tsinghua University
Beijing, China
liuyp20@mails.tsinghua.edu.cn

Zhuotao Liu
INSC, Tsinghua University
Zhongguancun Laboratory
Beijing, China
zhuotaoliu@tsinghua.edu.cn

Kun Sun
IST, George Mason University
Fairfax, USA
ksun3@gmu.edu

Ke Xu
DCST, Tsinghua University
Zhongguancun Laboratory
Beijing, China
xuke@tsinghua.edu.cn

Abstract

Website Fingerprinting (WF) attacks can effectively identify the websites visited by Tor clients via analyzing encrypted traffic patterns. Existing attacks focus on identifying different websites, but their accuracy dramatically decreases when applied to identify fine-grained webpages, especially when distinguishing among different subpages of the same website. WebPage Fingerprinting (WPF) attacks face the challenges of highly similar traffic patterns and a much larger scale of webpages. Furthermore, clients often visit multiple webpages concurrently, increasing the difficulty of extracting the traffic patterns of each webpage from the obfuscated traffic. In this paper, we propose Oscar, a WPF attack based on multi-label metric learning that identifies different webpages from obfuscated traffic by transforming the feature space. Oscar can extract the subtle differences among various webpages, even those with similar traffic patterns. In particular, Oscar combines proxy-based and sample-based metric learning losses to extract webpage features from obfuscated traffic and identify multiple webpages. We prototype Oscar and evaluate its performance using traffic collected from 1,000 monitored webpages and over 9,000 unmonitored webpages in the real world. Oscar demonstrates an 88.6% improvement in the multi-label metric Recall@5 compared to the state-of-the-art attacks.

*Both authors contributed equally to this work.

CCS '24, October 14–18, 2024, Salt Lake City, UT, USA

ACM ISBN 979-8-4007-0636-3/24/10
https://doi.org/10.1145/3658644.3690211

CCS Concepts

• **Networks** → **Network privacy and anonymity**.

Keywords

Webpage fingerprinting; Tor; privacy; data augmentation; multi-label metric learning

ACM Reference Format:
Xiyuan Zhao, Xinhao Deng, Qi Li, Yunpeng Liu, Zhuotao Liu, Kun Sun, and Ke Xu. 2024. Towards Fine-Grained Webpage Fingerprinting at Scale. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security (CCS '24), October 14–18, 2024, Salt Lake City, UT, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3658644.3690211

1 Introduction

The Onion Routing (Tor) has millions of daily active clients and protects their online privacy through multi-layer encryption with multiple randomly selected relays [12]. However, it is vulnerable to Website Fingerprinting (WF) attacks, which can effectively deanonymize the communication. WF attacks identify the websites visited by Tor clients through analyzing the unique traffic patterns of the websites, *e.g.*, packet sizes, timestamps, and directions.

Prior WF attacks [11, 17, 22, 44, 45, 50] develop complex model structures to extract features of various websites from traffic. However, these attacks focus on identifying websites rather than fine-grained webpages. Since a single website often hosts multiple webpages, accurately identifying fine-grained webpages can provide additional valuable information. The performance of existing WF attacks significantly declines when tasked with webpage identification, as models trained with cross-entropy loss struggle to capture the subtle differences in webpage traffic.

To identify different webpages, a series of fine-grained WF attacks have been studied, *i.e.,* WebPage Fingerprinting (WPF) attacks [30, 46, 48, 49, 63], which leverage both coarse-grained and fine-grained traffic attributes. However, Tor clients often visit multiple webpages consecutively, a common behavior that significantly

complicates webpage fingerprinting and makes it challenging for existing attacks to extract the unique patterns of each webpage from the obfuscated traffic under the multi-tab setting [11]. Furthermore, evaluations of previous WF and WPF attacks generally involve a limited number of webpages. As the scale of webpages expands, these attacks experience degrading performance [7].

To fully relax the assumption of existing attacks on the browsing behaviors of Tor clients, the goal of this paper is to develop a fine-grained webpage fingerprinting attack that is capable of identifying multiple webpages concurrently visited. Generally, there are three main challenges in webpage fingerprinting. First, webpage traffic patterns share higher similarity [30, 56] due to analogous layouts, increasing the difficulty of distinguishing different webpages in the original feature space. Second, clients typically browse multiple webpages concurrently. This will generate obfuscated traffic from multiple webpages, resulting in irrelevant packets that can interfere with the identification of individual webpages. Additionally, multi-tab traffic patterns exhibit higher diversity due to the various combinations of webpages and the dynamic packet order. Third, the scale of webpages is approximately 50 times larger than that of websites [36], posing significant challenges to the performance of existing methods that primarily focus on website fingerprinting.

In this paper, we develop a multi-tab WPF attack framework called Oscar, which is capable of identifying webpages based on the subtle differences in their traffic patterns. The design of Oscar is based on the key observation that even though webpages within the same website share similar layouts, their contents and resources still exhibit differences, leading to subtle variations in their local traffic patterns. We utilize metric learning to transform the feature space, clustering traffic of the same webpages and distancing traffic of different webpages to extract the subtle differences among webpage traffic in the new feature space.

It is challenging to develop the metric-learning-based WPF attack under the multi-tab setting due to the class collapse problem, where the traffic of all webpages clusters to a single point [26]. To address this issue, Oscar utilizes multi-label metric learning to construct the WPF attack. First, it employs two data augmentation mechanisms based on the characteristics of multi-tab traffic to enhance the sample diversity. Second, it utilizes metric learning to transform the feature space to separate different webpages based on a multi-label loss, which includes a proxy-based loss and a sample-based loss. Specifically, to address the class collapse issue, Oscar clusters relevant webpage traffic by setting proxies for each webpage and separates irrelevant webpage traffic by isolating samples with low label correlation. Finally, Oscar achieves efficient and accurate multi-tab webpage identification on a large scale of monitored webpages by leveraging proxy-based and sample-based combined k-NN classifiers. The webpage identification relies on the distribution characteristics of different webpages in the transformed feature space.

We evaluate Oscar using datasets collected under both the closed-world and open-world settings, including 1,000 monitored webpages and more than 9,000 unmonitored webpages. To the best of our knowledge, these are the first multi-tab webpage traffic datasets regarding each webpage as a distinct class, with the number of concurrently accessed webpages being dynamic. We release the datasets and source code of Oscar[1]. Compared to state-of-the-art attacks, Oscar achieves an average improvement of 88.6% and 76.7% in Recall@5 under the closed-world and open-world settings, respectively. Moreover, Oscar significantly outperforms previous attacks under various scales of monitored webpages.

In summary, the contributions of this paper are four-fold:

- We develop a fine-grained WPF attack, Oscar, to identify webpages from obfuscated traffic under the multi-tab setting. To the best of our knowledge, Oscar is the first multi-tab attack to accurately identify fine-grained webpages at scale.
- We utilize data augmentation techniques both within and between traffic samples, leveraging the characteristics of multi-tab traffic. The data augmentation designed for multi-tab traffic enhances the generalization of Oscar in real-world WPF attacks.
- We develop a multi-label metric learning method for traffic samples to transform the feature space. The feature transformation combines the proxy-based loss and sample-based loss, effectively separating different webpages from multi-tab obfuscated traffic.
- We collect the first multi-tab webpage traffic datasets with 1,000 monitored webpages and over 9,000 unmonitored webpages, and validate the performance of Oscar under both the closed-world and open-world settings.

2 Background

2.1 WF and WPF Attacks

WF Attacks. Recently, encrypted traffic analysis has been extensively studied [27, 42]. WF attacks, a specific approach within the broader field of encrypted traffic analysis, aim to identify the unique traffic patterns of websites, including packet time intervals, sizes, and directions. WF attacks compromise the privacy of Tor clients by extracting the traffic patterns of different websites from packet sequences. ML-based WF attacks [19, 39, 57] utilize expert knowledge to construct features specific to websites for identification. With advanced deep learning (DL) algorithms, DL-based WF attacks [2, 50] enable automatic feature extraction and robust attacks. To apply WF attacks in the real world, existing works develop WF attacks for various real-world scenarios, such as under the multi-tab setting [11, 17, 22], limited training data [37, 51], dynamic network conditions [1], and various defenses [44, 47]. However, existing WF attacks focus on identifying different websites, and have mostly been evaluated on index pages. Despite certain attacks gathering subpage traffic samples [11, 37, 41], their identification targets remain restricted to websites. When applied to fine-grained webpage identification, these attacks become ineffective due to performance deterioration caused by the high similarity of webpage traffic patterns.

WPF Attacks. Existing ML-based WPF attacks [48, 49, 63] leverage global and local features to differentiate webpages. Existing DL-based WPF attacks [30, 46] apply powerful Convolutional Neural Networks (CNN) or Graph Neural Networks (GNN) to extract features from webpage traffic. However, these methods either consider distinguishing webpages within the same website or suffer from the limitation of small-scale webpages. Additionally, none of the above works address the challenge of multi-tab identification.

[1] https://zenodo.org/records/13383332

Table 1: Summary of Existing Attacks.

Attacks	Identification target[1]	Multi-tab[2]	Large scale[3]
k-FP [19]	Website	✗	✗
DF [50]	Website	✗	✗
Tik-Tok [44]	Website	✗	✗
TF [51]	Website	✗	✗
MWF [60, 61]	Website	✓	✗
BAPM [17]	Website	✓	✗
NetCLR [1]	Website	✗	✗
TMWF [22]	Website	✓	✗
ARES [11]	Website	✓	✗
FineWP [48, 49]	Webpage	✗	✗
BurNet [46]	Webpage	✗	✗
GAP-WF [30]	Webpage	✗	✗
Oscar	Webpage	✓	✓

[1] **Identification target** refers to if the attacks target on websites or webpages.
[2] **Multi-tab** refers to whether the attacks consider the setting where clients can visit multiple webpages and traffic from various webpages mixes.
[3] **Large scale** refers to whether the attacks are evaluated under the datasets with thousands of monitored webpages.

The existing attacks are summarized in Table 1. Different from existing attacks, Oscar implements multi-tab identification targeting specific webpages from various websites. Furthermore, Oscar expands the scale of monitored webpages to realize a real-world WPF attack.

2.2 Metric Learning

Metric learning aims to develop a new data representation to establish the similarity of samples, enhancing the ability to distinguish among samples from different classes [24]. This technique is widely applied in the image recognition domain to improve the prediction accuracy in both classification and clustering problems [14, 18, 21, 25, 33, 38, 52–54, 59]. The core concept of metric learning involves comparing anchor samples with positive samples or/and negative samples during the learning process, thereby pulling samples from the same classes closer together and pushing samples from different classes farther apart.

TF [51] and NetCLR [1] utilize metric learning and contrastive learning to identify the differences among website traffic. Metric learning is usually based on labeled data, while contrastive learning does not require samples to be labeled. However, these attacks depend on clean traffic under the single-tab setting to transform the feature space. In particular, the performance of TF is significantly impacted by the class collapse issue [26] when the traffic is obfuscated under the multi-tab setting. Moreover, NetCLR augments traces by analyzing single-tab traffic to pre-train a model using self-supervised learning. However, there is a significant gap between single-tab and multi-tab traffic patterns. Therefore, we design a multi-label metric learning method to distinguish webpages from obfuscated traffic.

3 Threat Model

The threat model of Oscar is illustrated in Figure 1. In our threat model, the clients are free to explore various webpages, moving beyond just the index pages. They are redirected to other webpages by clicking on links displayed on the current webpage. Clients are often interested in multiple webpages, therefore they may open multiple webpages concurrently, and the number of visited webpages is dynamic and unknown to the attacker a prior. Furthermore, a website typically hosts multiple webpages. Therefore, the scale of webpage monitoring is far larger than that of website monitoring. We broaden the scope of our monitoring to include more monitored webpages, accommodating the size of real-world webpages.

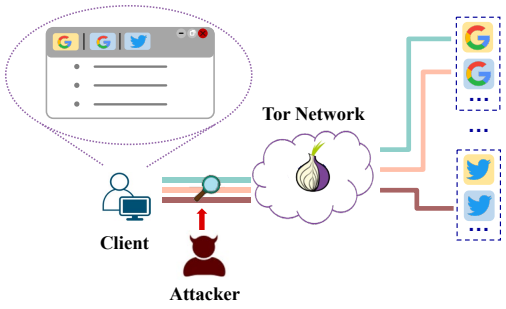


Figure 1: The threat model of Oscar, where clients can access both the index pages and other webpages under the multi-tab setting.

To identify the webpages visited by the client, the attacker has the ability to eavesdrop on the communication link between the client and the guard node of the Tor network to analyze the patterns of bidirectional communication packets [11, 22, 50, 51]. Though the connections between the client and the guard node are protected by encryption, the attacker can gather traffic metadata such as the direction, size, and time sequences of traffic packets to extract unique fingerprint features that can be used to identify the webpages visited by the client. In addition, the attacker does not require the ability to actively modify, delay, or decrypt these packets.

Similar to existing WF attacks [2, 11, 22, 44, 45, 50], we consider both the closed-world and open-world settings. Under the closed-world setting, Tor clients can only browse a limited number of webpages, *i.e.,* the monitored webpages. The attacker can collect traffic from all monitored webpages to train the model. Under the open-world setting, Tor clients can freely access a large number of webpages unknown to the attacker, *i.e.,* the unmonitored webpages. Given the vast number of webpages in reality, obtaining training samples for all of them is impractical, highlighting the realism of open-world identification. Note that Tor clients under the open-world setting still use multiple tabs to access both the monitored and unmonitored webpages.

4 System Overview

Oscar is a robust WPF attack that leverages the differences in webpage traffic to identify the webpages visited by Tor clients. Since different webpages from the same website are not identical, it leads to variations in local traffic patterns. Although it is difficult to directly distinguish webpage traffic in the original feature space, by extracting the differences in webpage traffic through metric learning, we can separate different webpages in the transformed feature space. Figure 2 shows the feature transformation of webpage traffic through metric learning, where traffic from different webpages is

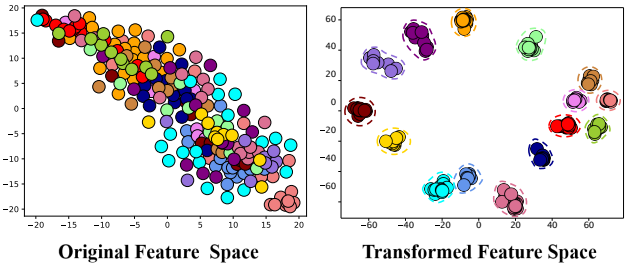


Figure 2: Feature transformation based on multi-label metric learning. Each color represents samples from a distinct webpage.

isolated. Specifically, in our study of 1,000 webpages, the similarity of traffic features across different webpages decreases by an average of 52.92% after feature transformation. This reduction in similarity highlights the effectiveness of Oscar in distinguishing webpage traffic and accurately identifying the visited webpages.

Formally, considering W monitored webpages, the problem is defined as follows: the sample set X=$\{x_1,..., x_N\}$ contains N samples, where each sample holds the dimension of $d_i * 2$, comprising both the direction sequences ds_i and time sequences ts_i extracted from the original packet sequences. The direction sequences are distinguished by +1 and -1 for outgoing and incoming packets, and the time sequences are calculated as the interval relative to the first packet. The label set is defined as Y=$\{y_1,... , y_N\}$, where each label y_i is a W-dimensional 0/1 vector. $y_{ij} = 1$ shows that the ith sample comprises the traffic from the jth webpage. Under our attack setting, the clients can visit multiple webpages, *i.e.*, $|y_i| \geq 1$. Consequently, our problem is characterized as a multi-label classification, where each sample may be associated with one or more labels. The objective is to precisely determine if each label is present in samples. This is more challenging than the multi-class classification, as the latter only considers the label with the highest probability.

Figure 3 shows the three modules in Oscar to achieve a robust WPF attack. First, the Data Augmentation module enhances the generalization of Oscar through inter-sample and intra-sample combined data augmentation operations based on the characteristics of multi-tab traffic in the real world. The inter-sample augmentation combines the traffic of two samples in chronological order to enhance the traffic diversity from different webpage combinations. The intra-sample augmentation exchanges packets within a single sample to accommodate variations in packet order. Second, the Feature Transformation module leverages both the original samples and augmented samples to train the DF-based feature transformation model based on two losses. The proxy-based loss clusters relevant samples, and the sample-based loss isolates irrelevant samples. Combining these two losses, this module obtains a feature transformation model responsible for transforming the feature space. Third, the Webpage Identification module achieves efficient multi-label webpage classification based on the distribution features of webpage traffic in the transformed feature space. It is composed of two k-NN classifiers, incorporating both the proxy-sample distance and the sample-sample distance, to enhance accurate webpage identification. The predicted webpages are determined based on the scores calculated from the two classifiers.

5 System Design

In this section, we present the design details of Oscar, including the Data Augmentation module, the Feature Transformation module, and the Webpage Identification module.

5.1 Data Augmentation

The Data Augmentation module employs inter-sample and intra-sample operations to generate simulated traffic based on the original traffic, thereby enhancing the generalization to diverse multi-tab traffic. To achieve the data augmentation, Oscar operates on the raw traffic samples and labels based on traffic characteristics. However, clients typically browse multiple webpages concurrently. Existing data augmentation operations [1, 5, 62] cannot be applied to multi-tab traffic, as they ignore the diversity introduced by traffic mixing under the multi-tab setting. To address this issue, we design two data augmentation operations specifically for multi-tab traffic.

Multi-tab traffic exhibits greater diversity compared to single-tab traffic as the clients may browse multi-tab webpages with different webpage combinations, resulting in various mixed traffic. Even when the same combination of webpages is accessed, packets from different webpages within a session are transmitted through different circuits, leading to a dynamic packet order [55]. Therefore, multi-tab traffic is more diverse, both for different webpage combinations and the same webpage combinations. To enrich the sample diversity under the multi-tab setting, we incorporate two data augmentation operations, as shown in Figure 4. First, we design an inter-sample data augmentation that combines traffic of different samples to adapt to the diversity arising from various webpage combinations. Second, we implement an intra-sample data augmentation that exchanges packets within a single sample to handle the variations in packet order, addressing the diversity within the same webpage combinations. We detail these two operations as follows.

Inter-Sample Augmentation. We design an inter-sample data augmentation based on traffic combining to enhance the variety of traffic from different webpage combinations. Under the multi-tab setting, traffic from various webpages is mixed, making traffic from the same webpage display completely different patterns when mixed with different webpages. Figure 5(a) illustrates the correlation between the edit distance of samples and the quantity of browsed webpages, where the edit distance reflects the similarity of two vectors by measuring the minimum operations required to transform one vector to another. Specifically, we select 500 samples from different webpage combinations. These samples are grouped based on the number of labels, ranging from 1 to 5, with 100 samples in each group. We then calculate the edit distance among samples within each group, where all samples share the same number of webpage labels. The results show that disparity in samples from different webpage combinations increases as the number of concurrently accessed webpages rises. Therefore, Oscar adopts an inter-sample augmentation to adapt to the diversity of various webpage combinations.

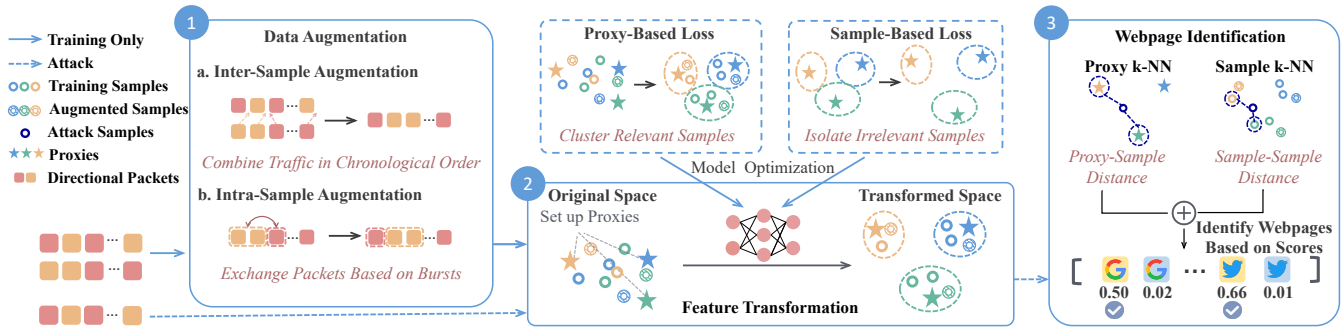
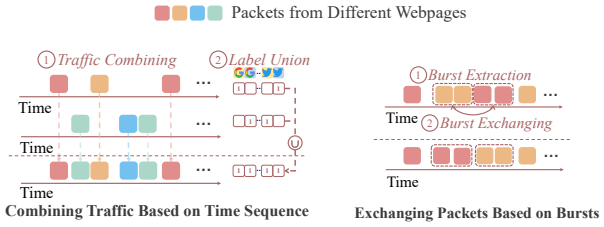
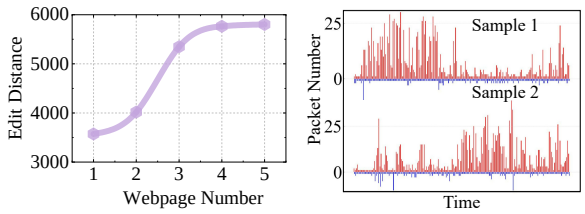


Figure 3: Design overview of Oscar, which consists of three modules: Data Augmentation module, Feature Transformation module and Webpage Identification module.



(a) Inter-Sample Augmentation (b) Intra-Sample Augmentation

Figure 4: Two operations in the Data Augmentation module: inter-sample and intra-sample augmentation. Packets from different webpages are shown in different colors.



(a) Edit distance of samples (b) Burst patterns when browsing the same webpages twice

Figure 5: Analysis of multi-tab obfuscated traffic collected from the real world.

Different from the intra-sample augmentation, the inter-sample augmentation involves two samples and operates on both the traffic samples and labels. Specifically, Oscar combines the original two traffic based on the time sequence of packets. Algorithm 1 details the inter-sample augmentation. Oscar first sets two pointers to indicate the packet indexes of the two samples respectively (line 1). Oscar then compares the times of the indexed packets in the two samples, and adds the packet with the earlier time to the newly generated packet sequence, with the index moving one step backward (line 2-8). In this way, Oscar integrates the packets of the two samples in a time-ordered manner. To generate the new label, Oscar unions the labels of the two samples since the newly generated traffic includes the packets of both the original samples (line 9). Following this operation, Oscar generates a new sample that contains traffic from a webpage combination that is not present during training. With the inter-sample augmentation, Oscar enriches the variety of webpage combinations, enhancing the generalization of Oscar when applied to multi-tab webpage identification.

Algorithm 1: Inter-Sample Data Augmentation.

input : original sample1: ds_i(direction sequence), ts_i(time sequence);
original label1: y_i;
original sample2: ds_j(direction sequence), ts_j(time sequence);
original label2: y_j;
input dimension: d_i;
output: generated sample: ds_g(direction sequence);
generated label: y_g;

1 $index_1 \leftarrow 0$ $index_2 \leftarrow 0$ // set two pointers
2 **for** $k \leftarrow 0$ **to** d_i **do**
3 **if** $ts_i[index_1] \leq ts_j[index_2]$ **then** // compare the times of the two indexed packets
4 $ds_g[k] \leftarrow ds_i[index_1]$ // add packet to the new sequence
5 $index_1 \leftarrow index_1 + 1$
6 **else**
7 $ds_g[k] \leftarrow ds_j[index_2]$ // add packet to the new sequence
8 $index_2 \leftarrow index_2 + 1$
9 $y_g \leftarrow y_i \cup y_j$ // union the labels of two original samples
10 return ds_g, y_g

Intra-Sample Augmentation. In addition to the inter-sample augmentation, we design an intra-sample data augmentation to adapt to the traffic diversity of the same webpage combinations under the multi-tab setting. Web browsing on various tabs is based on distinct Tor circuits, resulting in dynamic packet ordering when traffic mixes [55]. Figure 5(b) demonstrates the burst patterns of traffic when browsing the same webpages twice. It is evident that there is a distinct discrepancy between the burst patterns of the two samples. To accommodate this sequential diversity when multiple webpages are loaded concurrently, Oscar employs an intra-sample augmentation strategy based on packet exchanging.

Algorithm 2: Intra-Sample Data Augmentation.

input : original sample: ds_i(direction sequence);
exchanging ratio: m_e;
output : generated sample: ds_g(direction sequence);

1 $burst\_sequences \leftarrow extract\_bursts(ds_i)$
2 $ds_g \leftarrow ds_i$
3 $ex\_num \leftarrow len(burst\_sequences) * m_e$ // calculate the number of bursts to be exchanged based on the total burst number
4 $ex\_bursts \leftarrow sample(burst\_sequences, ex\_num)$ // sample ex_num bursts from the burst sequence
5 **for** $burst$ *in* ex_bursts **do**
6 $\quad exchange\_bursts(ds_g[burst], ds_g[burst+1])$ // exchange the selected bursts with their subsequent bursts
7 return ds_g

The intra-sample augmentation modifies traffic within a single sample without altering its label. Specifically, Oscar adopts an exchanging operation based on bursts. Bursts are consecutive packets of the same direction, often containing resources like texts and images [1]. Algorithm 2 details the exchanging operation. Oscar begins with identifying the bursts in the original direction sequence (line 1). Oscar then determines the number of bursts to be exchanged ex_num based on the total burst number of the sample with a ratio of m_e, and samples ex_num bursts in the original traffic (line 3-4). Finally, for each selected burst, Oscar exchanges it with the following burst, while leaving the remaining bursts unchanged (line 5-6). By setting the exchanging ratio advisedly, Oscar dynamically adjusts the number of exchanges according to the total burst number of different samples, thus ensuring that the modification of each sample is controlled within a manageable range. In summary, this operation improves the ability to cope with the dynamic and unpredictable patterns of packet order in multi-tab traffic.

Notably, our data augmentation is grounded in the analysis of multi-tab traffic characteristics. By combining inter-sample and intra-sample augmentation techniques, Oscar significantly enhances sample diversity, ensuring the generalization of the WPF attack under the multi-tab setting. After generating augmented samples, we blend them with the original samples for feature transformation.

5.2 Feature Transformation

The Feature Transformation module transforms traffic features to cluster traffic of the same webpages and separate traffic from different webpages. To realize the above feature transformation, Oscar needs to contrast the traffic of different webpages based on metric learning, so as to extract the subtle differences in the traffic patterns of different webpages. However, existing website fingerprinting attacks based on metric learning [51] cannot be applied to webpage identification. The reason is that Tor clients usually browse multiple webpages under the multi-tab setting, with each traffic having multiple labels. Traditional metric learning methods select positive and negative samples based on the single label, leading to a dramatic increase in the number of positive samples under the multi-tab setting, which causes the class collapse issue [26] (*i.e.,* traffic from all different webpages clusters together in the new feature space).

To address the above issue, we design a multi-label metric learning method to achieve feature transformation. The details of the feature transformation are shown in Figure 6. To effectively identify webpages corresponding to multi-tab obfuscated traffic, our feature transformation contains a feature transformation model to embed features to a lower-dimensional feature space and a multi-label metric learning loss function to aggregate traffic of the same webpages and separate traffic of different webpages in the transformed feature space.

The feature transformation model takes the original direction sequences of traffic as input, and outputs the transformed lower-dimensional vectors. We select DF as the feature transformation model for the following reasons: (i) DF has demonstrated effectiveness in WF attacks, achieving 98% accuracy in identifying different websites [50]; (ii) DF is built upon CNN, which can effectively extract the features regardless of the part in which the feature fragments appear. The shift-invariance characteristic of CNN can extract specific features with dynamic locations, which is particularly important under the multi-tab setting. DF contains four basic convolutional blocks and two fully connected layers. Each block contains two one-dimensional convolutional layers and one max pooling layer. We retain the original four convolutional blocks and replace the fully connected layers with a linear layer to embed the features to a low-dimensional feature space, enabling DF to function as a feature extractor.

Beyond the feature transformation model, the loss function is crucial for the effectiveness of feature transformation. Oscar utilizes a multi-label metric learning loss to aggregate traffic from the same webpages and separate traffic from different webpages. The loss function comprises two parts: a proxy-based loss and a sample-based loss.

Proxy-Based Loss. As discussed above, existing metric learning approaches calculate loss based on positive samples, which leads to class collapse under the multi-tab setting. Therefore, we develop a proxy-based loss to aggregate traffic from the same webpages under the multi-tab setting. The left part of Figure 6 illustrates the proxy-based loss. We first set up proxies as representatives for each webpage. Instead of pulling samples with the same labels closer together, the proxy-based loss directs samples to the correlated proxies, therefore effectively aggregating samples of the same webpages. Specifically, the positions of the proxies are dynamic and optimized together with the model's parameters in each epoch of model training. Such adaptability of proxies is crucial as it contributes to learning more accurate distributions of different webpages along with the model progressing through training.

To calculate the proxy-based loss, we first initialize the W proxies in the proxy set P = $\{p_1, ..., p_W\}$, where W is the number of webpages. Proxies hold the dimension of d_o, consistent with the embedded vector dimension after the feature transformation model. After proxy initialization, we excavate the proxy-sample relationship to cluster relevant samples.

The proxy-sample relationship can be divided into two types: positive proxy-sample pair and negative proxy-sample pair. In the case where the sample contains traffic from the webpage associated with the proxy, they are identified as a positive proxy-sample pair, otherwise, they are a negative proxy-sample pair, *i.e.,* if $y_{ij} = 1$, sample x_i and proxy p_j constitute a positive proxy-sample pair.

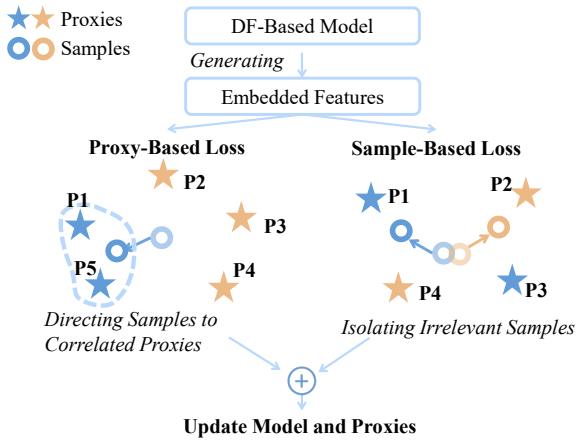


Figure 6: Details of the Feature Transformation module, which comprises a DF-based model and a multi-label metric learning loss function. We display the samples and their correlated proxies in the same color.

For positive proxy-sample pairs, the anticipated similarity between them should approach 1, indicating that the samples are close to the relevant proxies in the feature space. The positive loss is then defined as the difference between the cosine similarity and 1:

$$L_{pos\_proxy}\langle x_i, p_j\rangle = 1 - cos\_sim\langle x_i, p_j\rangle, \tag{1}$$

where cos_sim refers to the cosine similarity shown as follows:

$$cos\_sim\langle x_i, p_j\rangle = \frac{x_i \cdot p_j}{\|x_i\| \times \|p_j\|}. \tag{2}$$

On the other hand, negative proxy-sample pairs are anticipated to exhibit low similarity, indicating their separation in the transformed feature space. To prevent overfitting, Oscar sets a margin for the expected similarity. If the similarity is below this margin, indicating effective separation from unrelated proxies, the loss is set to 0. The negative loss is then defined as the maximum of the two terms:

$$L_{neg\_proxy}\langle x_i, p_j\rangle = max(cos\_sim\langle x_i, p_j\rangle - margin, 0), \tag{3}$$

where the margin is a hyperparameter preset.

We calculate the sum of all positive proxy-sample loss $L_{all\_pos\_proxy}$ and negative proxy-sample loss $L_{all\_neg\_proxy}$, and combine them for the total proxy-based loss L_{proxy}:

$$L_{proxy} = \frac{L_{all\_pos\_proxy}}{\Theta_{pos\_proxy}} + \frac{L_{all\_neg\_proxy}}{\Theta_{neg\_proxy}}, \tag{4}$$

where Θ_{pos_proxy} and Θ_{neg_proxy} refer to the total number of positive proxy-sample pairs and negative proxy-sample pairs. Note that we divide by the number of positive pairs and negative pairs to avoid quantity imbalance. The multi-label proxy loss effectively enhances the accuracy of webpage identification by ensuring that samples are tightly distributed around the relevant proxies. As a result, the patterns of different webpages can be extracted.

Sample-Based Loss. Different from the proxy-based loss, Oscar utilizes a sample-based loss to separate irrelevant webpage traffic in the transformed feature space. Specifically, under the multi-tab setting, the proxy-based loss might inadvertently bring traffic from unrelated webpages closer together. The right part of Figure 6 illustrates this conflict. The left sample, comprising traffic from webpage 1 and webpage 3, is expected to be approximately positioned between proxy 1 and proxy 3. Similarly, the right sample, containing traffic from webpage 2 and webpage 4, is likely to be located between proxy 2 and proxy 4. Despite that these two samples share no common labels, their positions in the transformed feature space can be notably close. Therefore, we design a sample-based loss to effectively identify and separate irrelevant webpage traffic by evaluating the similarity between samples in the feature space, ensuring that traffic from distinct webpages is separated.

The sample-based loss takes advantage of the relationship between samples. Due to large-scale monitored webpages, the number of sample pairs with different labels is very large. Therefore, we design sample mining based on the label coincidence degree to selectively separate samples with low correlation, *i.e.,* samples without identical labels. To mine irrelevant sample pairs, we first sift out the samples with at least two labels and build a new set $X' = \{x_i | x_i \in X \wedge |y_i| > 1\}$, where the total number of the filtered samples is N'. Then we find samples with no overlapping labels in X', *i.e.,* $y_i * y_j = 0$, and form the sample pairs. For irrelevant sample pairs, the cosine similarity between them is defined as:

$$cos\_sim\langle x_i, x_j\rangle = \frac{x_i \cdot x_j}{\|x_i\| \times \|x_j\|}. \tag{5}$$

We expect irrelevant sample pairs to show a lower degree of cosine similarity. Similar to the calculation of the negative proxy-sample loss, the margin is applied, and when the similarity is below the margin, the loss is set to 0:

$$L_{ir\_sample}\langle x_i, x_j\rangle = max(cos\_sim\langle x_i, x_j\rangle - margin, 0). \tag{6}$$

In each epoch, we collect all the irrelevant sample pairs in the batch and compute their sum $L_{all\_ir\_sample}$. The total sample loss is calculated as follows:

$$L_{sample} = \frac{L_{all\_ir\_sample}}{\Theta_{ir\_sample}}, \tag{7}$$

where Θ_{ir_sample} refers to the total number of irrelevant sample pairs. Compared with the vanilla sample-based metric learning methods, our approach focuses on samples with low label correlation, thus significantly reducing the pair number. Overall, the sample-based loss enhances the effectiveness of the feature transformation by isolating irrelevant samples.

After calculating the two parts of losses, we combine them for the total loss:

$$Loss = L_{proxy} + \beta \times L_{sample}, \tag{8}$$

where β is a hyperparameter that adjusts the weights of these two losses. When $\beta = 0$, the above loss simplifies to the multi-label proxy-based loss.

Note that the proxies are added to the parameters and updated with the model's parameters using the same optimizer. This dynamic adjustment refines the distributions of proxies, leading to more precise boundaries for each webpage. Following the feature transformation module, we strategically transform the feature space to distinctly separate different webpages. This separation is vital for effective webpage classification, as it ensures that each webpage

is represented in a unique and distinguishable manner within the feature space.

5.3 Webpage Identification

The Webpage Identification module integrates two k-NN classifiers to achieve robust multi-tab webpage identification. Traditional k-NN classifiers typically rely on the labels of the nearest samples for classification. However, the diversity of multi-tab traffic can lead to sample drift and performance degradation. Therefore, we integrate a proxy-based k-NN and a sample-based k-NN, as illustrated in Figure 7, to achieve robust webpage identification. The proxy-based k-NN benefits from constantly updated and precise representations of webpages, focusing on the uniform features of webpages. Meanwhile, the sample-based k-NN accounts for the diversity of multi-tab webpage traffic across various webpage combinations. Finally, we combine the results of these two classifiers to calculate the label scores, leveraging the strengths of both to improve the robustness of webpage identification under the multi-tab setting.

The proxy-based k-NN achieves classification based on the proxy-sample distance. Specifically, it retrieves the nearest b proxies and calculates the scores using the distances of the retrieved proxies with the target sample. Given that our transformed feature space is built upon cosine similarity, our k-NN classifiers adopt the cosine distance for score calculation:

$$cos\_dis\langle x_{target}, p_j\rangle = 1 - cos\_sim\langle x_{target}, p_j\rangle, \tag{9}$$

where cos_sim is defined as above. x_{target} is the sample to be identified and p_j is the retrieved proxy. Then the label scores based on proxies $score\_proxy$ are calculated as:

$$score\_proxy_j = \begin{cases} \frac{1}{cos\_dis\langle x_{target}, p_j\rangle} & p_j \in R_{proxy} \\ 0 & p_j \notin R_{proxy} \end{cases}, \tag{10}$$

where R_{proxy} is the set of retrieved proxies. Proxies closer to the target sample contribute higher scores, while proxies outside the retrieved set contribute a score of 0.

Similarly, the sample-based k-NN retrieves the nearest b samples and calculates the sample-sample distance:

$$cos\_dis\langle x_{target}, x_i\rangle = 1 - cos\_sim\langle x_{target}, x_i\rangle, \tag{11}$$

where x_i is the retrieved sample. The samples are associated with multiple labels, and each retrieved sample contributes the same score for the corresponding labels. Then the label scores based on samples $score\_sample$ can be calculated by summing the contributions from all retrieved samples:

$$score\_sample_j = \sum_{x_i \in R_{sample} \wedge y_{ij}=1} \frac{1}{cos\_dis\langle x_{target}, x_i\rangle}, \tag{12}$$

where R_{sample} is the set of retrieved samples.

At last, we combine the results of these two classifiers, *i.e.,* for label j, the total score is the weighted sum of these two terms:

$$score_j = score\_proxy_j + \theta \times score\_sample_j, \tag{13}$$

where θ is a hyperparameter that adjusts the weights of these two scores. By combining the results of these two classifiers, our webpage identification considers both the uniform characteristics of different webpages and the variability of multi-tab traffic samples.

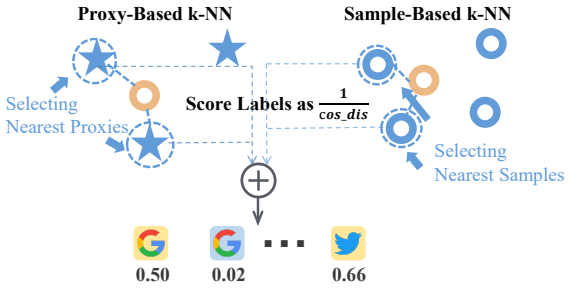


Figure 7: Details of the Webpage Identification module, which consists of two k-NNs to calculate label scores.

Table 2: Hyperparameter settings in our evaluation.

Module	Hyperparameters	Value
Data Augmentation	Input Dimension d_i	10,000
	Exchanging Ratio m_e	5%
Feature Transformation	Margin	0.1
	Loss Weight β	4.5
	Transformed Dimension d_o	512
Webpage Identification	Neighbor Number b	40
	Score Weight θ	2
	Threshold τ	0.3

The scores for different webpages are aggregated and ranked, with the identified webpages being output based on a preset threshold τ.

6 Evaluation

In this section, we evaluate Oscar with datasets collected in the real world. We compare the performance of Oscar with the state-of-the-art WF attacks.

6.1 Experimental Setup

Implementation. We prototype Oscar using Torch 1.9.0 and Python 3.8. We perform a random search of hyperparameters and set the optimal hyperparameters to default values as shown in Table 2. For the data augmentation module, we set the exchanging ratio m_e to 5% to sufficiently augment the original traffic without corrupting the critical traffic patterns. For the feature transformation module, we use the weight of the sample-based loss to a larger value (*i.e.,* $\beta = 4.5$) to enhance the quality of the transformed feature space under the multi-tab setting. For the webpage identification module, we set the value of the threshold τ to achieve the best F1-score. Further analysis of the impact of hyperparameters can be found in Section 6.6.

Dataset. Existing datasets [11, 45, 50] regard each website as a distinct class. To evaluate the performance of Oscar on realistic WPF attacks, we collect multi-tab webpage traffic datasets. To the best of our knowledge, these are the first datasets of real-world traffic from multi-tab webpages, where each webpage is regarded as a distinct class. To be specific, we collect two multi-tab datasets:

Table 3: Details of our datasets.

Dataset	Webpage Number	Label / Sample[1]	Sample / Comb.[2]	Sample Number
CW	1,000	1-5	10	81,284
OW	9,236	2-5	1	9,236

[1] **Label / Sample** represents the number of labels per sample.
[2] **Sample / Comb.** represents the number of samples per webpage combination.

closed-world dataset and unmonitored webpage dataset under the open-world setting, as shown in Table 3.

- **Closed-World Dataset *CW*:** We first build our monitored webpage set. Specifically, we select 115 websites from Alexa-top 20,000, visit the homepage of these websites and obtain 10 subpages by crawling the links on each website. Then we record the screenshots of these webpages during data collection, and filter out the invalid webpages by checking whether the screenshots exist and whether the webpage contents are successfully loaded. In this way, we acquire 1,000 webpages in total, and regard them as monitored webpages. We then collect samples of browsing the above 1,000 monitored webpages. The number of webpages visited in a session ranges from 1 to 5, with intervals between webpages randomly set between 3 and 10 seconds. 10 samples are collected for each webpage combination. We filter out samples with less than 1,000 packets and consider them as invalid accesses.
- **Unmonitored Webpage Dataset under the Open-World Setting *OW*:** For the unmonitored webpage set, we remove the websites that have been used by the closed world in Alexa-top 20,000 and keep the homepage of the rest websites. After filtering out invalid webpages, we get 9,236 webpages from distinctive websites and regard them as unmonitored webpages. We then collect samples of browsing a mixture of monitored webpages and unmonitored webpages. The number of webpages visited in a session is in the range of 2-5, with one from unmonitored webpages and the rest from monitored webpages. Each combination within this dataset consists of different unmonitored webpages and one sample is collected for each combination. In this way, we ensure that the open-world traffic in the training set, validation set, and testing set originates from different unmonitored websites.

The differences between our datasets and previous datasets are as follows: (i) We separate the traffic of different subpages from the same website and treat each webpage as a distinct class, which is different from existing works that merely collect traffic of index pages [45, 50] or utilize subpage traffic to identify websites [11, 31, 37, 41]. (ii) We expand the scale of monitored webpages. Most previous works only collect and monitor around 100 websites [11, 17, 50, 51, 61], whereas we monitor 1,000 webpages in total. Specifically, our datasets are the largest multi-tab webpage datasets in the wild. (iii) In the real world, the number of webpages visited by clients in a session is dynamic, which is not fully considered by existing works [17]. To adapt to this dynamism, the label number of samples in our datasets is not fixed.

Baselines. We compare Oscar with six state-of-the-art WF attacks, divided into two categories.

- **Single-Tab Attacks.** We select one machine-learning-based method k-FP [19] and three deep-learning-based methods DF [50], Tik-Tok [44] and NetCLR [1]. k-FP extracts 175 features from time and direction sequences and applies the Random Forest classifier for website classification. The original k-FP attack uses 1,000 trees, which cannot be realized due to the high resource overhead under the large-scale multi-tab setting. Therefore, we have to reduce the tree number to 100. DF and Tik-Tok adopt CNN-based architecture to achieve automatic feature extraction. NetCLR applies self-supervised learning to pre-train a DF model based on augmented traces and finetunes the model with labeled traces. Since the original cross-entropy loss adopted by the above three methods is suitable for multi-class classification, we follow [11] and change their loss function to the binary cross-entropy loss to achieve multi-label classification.
- **Multi-Tab Attacks.** We select two multi-tab attacks BAPM [17] and TMWF [22]. BAPM applies multi-head attention for multi-tab identification and each head predicts an individual website. TMWF applies the DETR algorithm in object detection and each query predicts a website. Since BAPM can only identify a fixed number of webpages, we follow [22] and set a "no-tab" class for these two methods. We set N attention heads for BAPM and N tab queries for TMWF where $N = 5$ since the maximum number of concurrently accessed webpages in our datasets is 5.

Due to various practical limitations, we do not compare Oscar with all previous attacks. Specifically, TF [51] mines samples based on single-label samples, leading to the problem of class collapse under the multi-label setting. MWF [60, 61] can only identify the first webpage under the multi-tab setting. ARES [11] builds a separate, complex Transformer-based classifier for each website, making it impractical for large-scale webpage fingerprinting due to its high overhead. Therefore, these attacks are excluded from our experiments.

Evaluation Metrics. We use two multi-label classification metrics for evaluation: Recall@k and AP@k [6, 28]. Rather than assess the result with the highest predicted probability, Recall@k evaluates the recall rate of the top k predicted webpages with the highest probabilities. Specifically, for sample i, assuming the real set of visited webpages is y_i, and the set of top-k predicted webpages is $\hat{y_i}$, Recall@k is calculated as follows:

$$Recall@k = \frac{\|y_i \cap \hat{y_i}\|}{\|y_i\|}. \tag{14}$$

AP@k is the average of Precision@k, which measures the proportion of correctly predicted webpages among the top-k results. Since the number of visited webpages in our datasets varies, AP@k can reflect the performance more precisely. Specifically, for sample i, assuming the real set of visited webpages is y_i, AP@k can be calculated as:

$$AP@k = \frac{\sum_{t=1}^{k} Precision@t}{min(k, \|y_i\|)}. \tag{15}$$

To compute Precision@t, we get the top-t predicted webpages with the highest probabilities $\hat{y_i}$. Precision@t is then defined as:

$$Precision@t = \frac{\|y_i \cap \hat{y_i}\|}{t}. \tag{16}$$

Table 4: Recall@5 of different attacks under the closed-world setting.

Attacks	k-FP	NetCLR	DF	Tik-Tok	BAPM	TMWF	Oscar
Recall@5	0.2331	0.1809	0.3354	0.3313	0.2106	0.3951	**0.4899**

Note that we do not calculate averaged Recall@k because the denominator of Recall@k is the number of the ground truth label of the sample, which is the same under different k values. We calculate Recall@k and AP@k for each sample in the dataset, and report the average as the final results.

6.2 WPF Attacks in the Closed World

We first evaluate the performance of Oscar under the closed-world setting, where clients only visit monitored webpages and the attacker can collect traffic samples of all the webpages to train the model. We use our closed-world dataset ***CW*** for evaluation. We divide the dataset into the training, validation and testing sets with the ratio of 8:1:1. Specifically, we utilize the augmented training set to train the feature transformation model and finetune parameters on the validation set. Then we use the trained model to transform samples in the testing set and achieve webpage identification based on the updated proxies and transformed training samples. For NetCLR, we use the training set to pre-train the model and the validation set to finetune. We calculate the multi-label classification metrics Recall@k and AP@k with the k values of Recall@k in $\{5, 10, 15, 20, 25, 30\}$, and AP@k in $\{1, 2, 3, 4, 5\}$. Since BAPM and TMWF predict webpages based on each attention head or tab query and are unable to determine the probability of all the webpages visited in a session, we can only compare with them on Recall@5.

We present the Recall@k and AP@k of five attacks in Figure 8 and Recall@5 of all methods in Table 4. Results show that Recall@k and AP@k increase as the k value increases and Oscar achieves the best performance in all metrics. Specifically, Recall@30 and AP@5 of Oscar are both over 0.73, while the best results of other methods remain around 0.52. Compared with k-FP, NetCLR, DF, Tik-Tok, BAPM and TMWF, Oscar improves by 110.2%, 170.8%, 46.1%, 47.9%, 132.6% and 24.0% on Recall@5. The performance superiority demonstrates that Oscar can identify the webpages both comprehensively and accurately. Although existing attacks achieve good performance in terms of website identification, their effectiveness significantly declines when applied to multi-tab webpage identification. This performance reduction is primarily due to their incapability of capturing the subtle distinctions hidden within the high-dimensional feature vectors of similar webpages. In contrast, Oscar concentrates on analyzing the differences among various webpages through webpage comparison. The proxy-based metric learning loss effectively clusters traffic from the same webpages and the sample-based metric learning loss isolates traffic from irrelevant webpages. In this way, we separate different webpages and extract their distinct characteristics in the transformed feature space. In addition, the improved multi-label webpage identification comprehensively takes into account the uniform characteristics of webpages and the diversity of multi-tab traffic, therefore achieving more accurate results under the multi-tab setting.

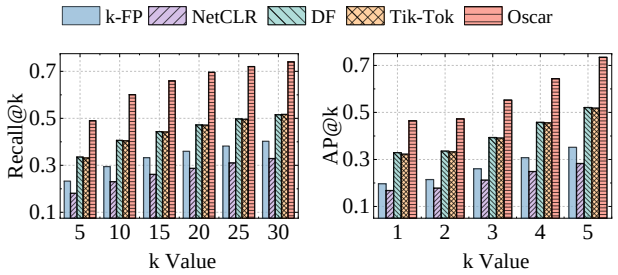


Figure 8: Results of the closed-world experiment. We report the Recall@k and AP@k with different k values.

Remark. In a nutshell, Oscar achieves the best performance in identifying multiple webpages from obfuscated traffic, which is credited to the effectively transformed features based on our method. Besides, the proxy-based and sample-based combined webpage identification can classify webpages more accurately.

6.3 WPF Attacks in the Open World

Under the open-world setting, the attacker can only collect samples from a subset of webpages, which does not cover all the webpages in the testing set. Typically, sensitive webpages constitute only a fraction of the entire webpages, and our primary objective is to precisely identify the particular monitored sensitive webpages despite the interference of unmonitored webpages. Under the open-world setting, following existing works[11, 50], we mix the closed-world dataset ***CW*** and the unmonitored webpage dataset in the open world ***OW*** to ensure that the sample number of monitored and unmonitored webpages is balanced. All the unmonitored webpages are treated as a single class, while each monitored webpage is still regarded as a distinct class. Note that samples under the open-world setting are still multi-labeled and the number of sample labels is also dynamic.

Figure 9 shows the performance under the open-world setting. Since BAPM and TMWF cannot distinguish between the unmonitored and the padding webpages under the open-world setting, we do not compare with them. Results show that Recall@30 of Oscar remains around 0.7 and AP@5 remains over 0.67. Specifically, Oscar improves by an average of 63.5% and 72.0% on Recall@30 and AP@5 respectively. Therefore, Oscar can still identify webpages more accurately than existing attacks in the presence of a large number of unmonitored webpages. This is mainly due to our feature transformation design. The proxy-based loss can individually cluster traffic from monitored webpages and unmonitored webpages, and the sample-based loss can further isolate irrelevant traffic.

Note that there is a slight performance drop compared to the closed-world setting. We believe this is mainly because traffic patterns of different unmonitored webpages exhibit obvious variations. When they are regarded as the same class in the training phase, it is challenging to extract the common feature pattern of all these webpages. In addition, the traffic of unmonitored webpages in the testing phase does not necessarily share similar patterns with the unmonitored webpages in the training phase, and may even resemble monitored webpages more closely, which brings confusion to the model. In spite of this, Oscar still outperforms previous works

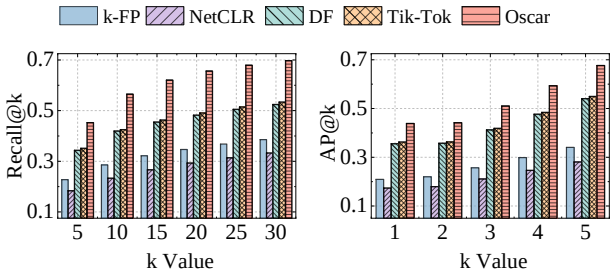


Figure 9: Results of the open-world experiment. We report the Recall@k and AP@k with different k values.

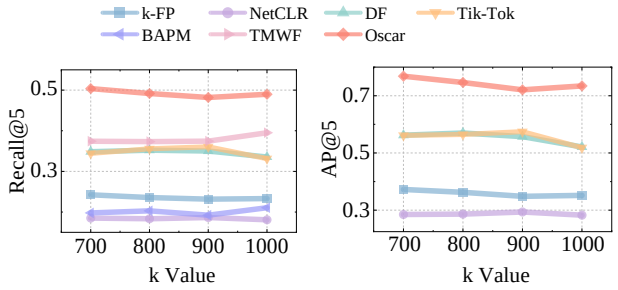


Figure 10: Results of the experiment on various scales of webpages, where the webpage number ranges from 700 to 1,000.

Table 5: Details of the datasets for the evaluation on different scales of monitored webpages.

Webpage Number	Sample Number
700	70,889
800	74,683
900	77,387
1,000	81,284

under the open-world setting by separating traffic from various monitored webpages and unmonitored webpages.

Remark. Overall, Oscar achieves the best performance under the open-world setting, where traffic of monitored webpages is mixed with that of unmonitored webpages. This demonstrates the superiority of Oscar in handling real-world attacks with a substantial number of webpages.

6.4 WPF Attacks on Various Scales of Webpages

We further evaluate the performance of Oscar on various scales of monitored webpages. We consider the large-scale evaluation and set the size of webpages as 700, 800, 900, 1,000. We use the closed-world dataset ***CW*** for this experiment and the details of datasets with different webpage numbers are shown in Table 5. The datasets are constructed as follows: we first randomly sample different numbers of webpages from the full set of monitored webpages, and then filter traffic samples from the original dataset. We keep the samples whose labels overlap with the selected webpages, and leave out the others. For those selected samples, the labels corresponding to the selected webpages are retained, and other labels are ignored. We use Recall@5 and AP@5 as metrics to evaluate the performance of different attacks.

The results are presented in Figure 10. As mentioned above, BAPM and TMWF cannot calculate AP@5, so we do not compare with them on this metric. Results demonstrate that Oscar consistently delivers superior performance across various scales of webpages. Specifically, AP@5 of Oscar is maintained over 0.72, while the best result of existing attacks is less than 0.58. Furthermore, Recall@5 of Oscar declines by only 2.76% and AP@5 declines by 4.41% as the webpage number increases from 700 to 1,000. This demonstrates the potential of Oscar to maintain effective across different scales of monitored webpages. As the number of webpages grows, the challenge to distinguish them in the original feature space increases. But Oscar focuses on contrasting webpages against each other to extract the subtle differences among them. As a result, Oscar can separate different webpages even if they are hard to distinguish in the original feature space. Furthermore, the multi-label classification based on k-NN is efficient and effective, allowing realistic large-scale attacks in the wild.

Remark. Oscar outperforms existing attacks in classifying various scales of webpages with the performance fluctuation kept within a manageable range. When the scale of monitored webpages further rises, Oscar is expected to sustain efficient and accurate attacks due to its ability to extract the differences among webpages.

Table 6: Ablation study results.

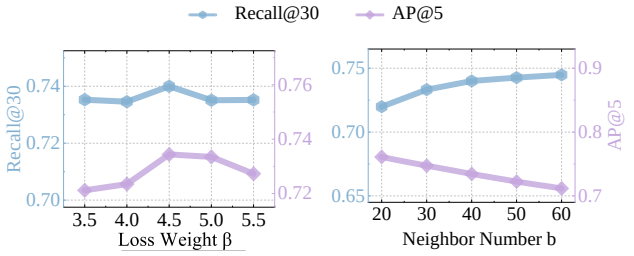
Method	Closed-World		Open-World	
	Recall@5	AP@5	Recall@5	AP@5
WI[1]	0.0155	0.0189	0.0238	0.0234
FT[3] (combined) +WI[1]	0.4511	0.6749	0.4206	0.6272
DA[2]+FT[3] (proxy-based) +WI[1]	0.3066	0.4450	0.2996	0.4340
DA[2]+FT[3] (sample-based) +WI[1]	0.0063	0.0070	0.0413	0.0826
DA[2]+FT[3] (combined) +WI[1]	**0.4899**	**0.7344**	**0.4527**	**0.6766**

[1] **WI** represents the webpage identification module.
[2] **DA** represents the data augmentation module.
[3] **FT** represents the feature transformation module.

6.5 Ablation Study

Next, we conduct the ablation study to analyze the impact of each module and the two loss functions on the attack performance. The experiment is conducted under both the closed-world and open-world settings and we report Recall@5 and AP@5. We consider the following five settings: (i) k-NN classifiers with the raw features; (ii) k-NN classifiers with proxy-based and sample-based combined feature transformation; (iii) k-NN classifiers with data augmentation and proxy-based feature transformation; (iv) k-NN classifiers with data augmentation and sample-based feature transformation; (v) k-NN classifiers with data augmentation and proxy-based and sample-based combined feature transformation.

The experiment results are detailed in Table 6. When identifying webpages based on the original feature space, it achieves a poor performance in distinguishing different webpages. This is due to the stochastic distributions of webpage traffic in the original feature space. The incorporation of the feature transformation module



(a) Loss weight (b) Neighbor number

Figure 11: Impacts of two critical hyperparameters on the performance: loss weight β in the Feature Transformation module and neighbor number b in the Webpage Identification module.

greatly improves the performance of Oscar, as it transforms the feature space to cluster samples of the same webpages and separate samples of different webpages. Additionally, the data augmentation module generates more samples and enhances the sample diversity, bringing further performance improvement.

Regarding the metric learning loss functions, the vanilla proxy-based loss underperforms the combined loss as it ignores the relationship among webpages under the multi-label setting. Besides, the vanilla sample-based loss aims at isolating irrelevant webpage traffic but cannot cluster relevant webpage traffic, making it ineffective when used alone. Nevertheless, it can significantly improve the performance when combined with the proxy-based loss, as it contributes to the optimization of webpage distributions in the feature space by isolating irrelevant webpage traffic.

Remark. To summarize, the feature transformation module plays a vital role in webpage identification by effectively separating traffic of different webpages in the transformed feature space. The data augmentation module boosts the performance by generating more samples and enhancing the sample diversity. The proxy-based loss contributes to aggregate traffic from the same webpages, and the sample-based loss is pivotal in separating irrelevant traffic. Therefore, combining these two losses results in the best performance.

6.6 Analysis of Hyperparameters

In this section, we analyze the impacts of critical hyperparameters on the performance of Oscar. We select two hyperparameters: loss weight β in the Feature Transformation module and neighbor number b in the Webpage Identification module. β adjusts the weights of the proxy-based and sample-based losses, and b decides the number of the retrieved neighbor proxies and samples. We evaluate the above two hyperparameters in the closed world, and all other hyperparameters are set to the default setting when evaluating each hyperparameter.

Figure 11(a) shows the performance with different β values. The fluctuations in Recall@30 and AP@5 are maintained within 0.015, demonstrating a stable performance with different settings. We note that Oscar achieves better performance when assigning a greater weight to the sample-based loss, demonstrating the importance of separating irrelevant traffic under the multi-tab setting. Through this separation, the sample-based loss contributes to extracting the relationship among webpages, therefore optimizing the distributions of different webpages in the transformed feature space.

Figure 11(b) demonstrates the impact of b values on the performance. When retrieving more proxies and samples, Recall@30 exhibits a slight increase while AP@5 shows a slight decrease. But both metrics stay within a narrow range from 0.71 to 0.77. Considering both the results of Recall and AP, we set the number of neighbor proxies and samples as 40.

Remark. Overall, the variation in hyperparameter settings for Oscar demonstrates only a modest impact on its performance, underscoring that its superior performance is due to the robust design instead of hyperparameter settings. This stability across different settings highlights the model's adaptability and reliability, making it well-suited for real-world applications.

7 Discussion

Larger-Scale of Monitored Webpages. We assess the performance of Oscar using 1,000 monitored webpages in the experiments, which is still quite limited compared to the number of webpages in the real world. The substantial size of webpages leads to a larger number of potential webpage combinations, which makes it impossible to achieve a comprehensive analysis that matches the number of real-world webpages. However, existing experiments still demonstrate the superiority of Oscar on various scales of webpages and the potential to maintain effective on a even larger scale of webpages. This effectiveness is largely attributed to our feature transformation design. Despite that the increase in the webpage number naturally reduces the disparities in the original feature space, Oscar employs comparisons among different webpages, enabling it to effectively discern and capture these subtle distinctions. Therefore, it is capable of separating webpages in the transformed feature space even when the scale is further up.

Deploying Complex Webpage Fingerprinting Defenses. Recently, an increasing number of works on webpage fingerprinting defenses have been proposed. Existing defenses can be divided into the following five categories: molding traffic into fixed patterns [3, 4, 13, 29], adding dummy packets [15, 23], creating collisions among webpages [35, 57, 58], splitting traffic into multiple streams [9, 20] and introducing adversarial noise [16, 34, 43]. However, many existing defenses incur high latency and bandwidth overhead, making them impractical for real-world deployment [31]. We will focus on improving the robustness of WPF attacks under complex defenses in future work.

Robustness under Concept Drift. Concept drift is incurred by the discrepancy between training data and testing data as the webpage properties change over time [11]. In reality, the contents of webpages are constantly changing, leading to variations in traffic patterns. Therefore, the trained model may not be well adapted to identify traffic a few years later, leading to a decrease in the identification accuracy. However, Oscar learns the differences among webpages through metric learning instead of mapping features to specific labels. Thus, the framework based on feature transformation remains effective in discerning the differences among webpages, even amidst content changes. In addition, we can finetune

the feature transformation model and update the proxies by collecting a modest amount of new samples to enhance the accuracy under concept drift.

8 Related Work

Single-Tab WF Attacks. WF attacks compromise the online privacy of Tor clients by extracting website fingerprints from Tor traffic. Early attacks [19, 39, 40, 57] extract website fingerprints based on expert knowledge and utilize ML models for website identification. Recently, DL has been widely applied to enhance the performance of WF attacks. AWF [45] utilizes DL models for automatic feature extraction and analysis. DF [50] develops an improved CNN model capable of robust WF attacks against the WTF-PAD defense [23]. Tik-Tok [44] and RF [47] improve traffic feature representations, further enhancing the robustness of WF attacks. However, the excellent performance of DL-based WF attacks depends on a large amount of training data. Var-CNN [2], TF [51] and GANDaLF [37] improve the model architecture and the training method to achieve effective WF attacks with a small number of training samples. NetCLR [1] augments traces and applies self-supervised and semi-supervised learning to enhance the robustness across different network conditions. Holmes [10] implements an early-stage WF attack by analyzing the spatio-temporal distribution features of traffic. Mitseva et al. [32] analyze traffic from multiple subpages of the same website to enhance website identification. Different from existing WF attacks that target on website identification, Oscar effectively achieves a fine-grained webpage fingerprinting attack.

WPF Attacks. Fine-grained WPF attacks present significant challenges because multiple subpages of a website often share similar templates and layouts, resulting in more similar traffic patterns [49, 56]. Existing WPF attacks mainly identify webpages by extracting packet-level features and flow-level features [30, 46, 48, 49, 63]. For example, BurNet [46] extracts features from unidirectional burst sequences based on CNN, and GAP-WF [30] utilizes GNN to extract flow-level features. However, existing WPF attacks assume that Tor clients only open a single tab to access webpages. Oscar relaxes the assumption of existing WPF attacks, achieving more realistic multi-tab WPF attacks. Even with the interference of noise packets from other webpages, Oscar still achieves a robust WPF attack.

Multi-tab WF Attacks. The obfuscated traffic under the multi-tab setting imposes challenges of extracting the pure traffic patterns of each website [8, 61]. Existing multi-tab WF attacks identify obfuscated traffic under the multi-tab setting by applying the attention mechanism [11, 17, 22]. For instance, BAPM [17] and ARES [11] use the multi-head attention mechanism, and TMWF [22] integrates powerful Transformer models for feature extraction. However, existing multi-tab attacks aim at differentiating websites rather than fine-grained webpages and are only applicable to a small scale of monitored websites. Oscar achieves a large-scale WPF attack under the multi-tab setting.

9 Conclusion

In this work, we propose Oscar, a fine-grained WPF attack designed for multi-tab webpage identification from obfuscated traffic. Constructing WPF attacks is more challenging than the existing WF attacks because the analyzed webpage traffic patterns exhibit a higher degree of similarity than website traffic patterns. Oscar utilizes metric learning to extract the differences among webpages, which combines proxy-based and sample-based losses to transform the feature space so that samples from the same webpages are clustered and samples from different webpages are separated. Moreover, we develop data augmentation mechanisms for Oscar, which allow Oscar to adapt to the diversity of multi-tab traffic in the real world. We prototype Oscar, and evaluate the performance on the collected datasets of multi-tab webpage traffic. The experiment results demonstrate that Oscar achieves 88.6% and 76.7% improvements of Recall@5 over the state-of-art attacks under both the closed-world and open-world settings, while maintaining a stable performance with various scales of monitored webpages.

Acknowledgment

We thank our anonymous reviewers for their helpful comments and feedback. The work is supported in part by NSFC under Grant 62132011, 62472247, and 62425201. Qi Li is the corresponding author of this paper.

References

[1] Alireza Bahramali, Ardavan Bozorgi, and Amir Houmansadr. 2023. Realistic Website Fingerprinting By Augmenting Network Traces. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*. 1035–1049.

[2] Sanjit Bhat, David Lu, Albert Kwon, and Srinivas Devadas. 2019. Var-CNN: A data-efficient website fingerprinting attack based on deep learning. *Proceedings on Privacy Enhancing Technologies* 2019, 4 (2019), 292–310.

[3] Xiang Cai, Rishab Nithyanand, and Rob Johnson. 2014. Cs-buflo: A congestion sensitive website fingerprinting defense. In *Proceedings of the 13th Workshop on Privacy in the Electronic Society*. 121–130.

[4] Xiang Cai, Rishab Nithyanand, Tao Wang, Rob Johnson, and Ian Goldberg. 2014. A systematic approach to developing and evaluating website fingerprinting defenses. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. 227–238.

[5] Mantun Chen, Yongjun Wang, Hongzuo Xu, and Xiatian Zhu. 2021. Few-shot website fingerprinting attack. *Computer Networks* 198 (2021), 108298.

[6] Ting Chen and Yizhou Sun. 2017. Task-guided and path-augmented heterogeneous network embedding for author identification. In *Proceedings of the tenth ACM international conference on web search and data mining*. 295–304.

[7] Giovanni Cherubin, Rob Jansen, and Carmela Troncoso. 2022. Online website fingerprinting: Evaluating website fingerprinting attacks on Tor in the real world. In *31st USENIX Security Symposium (USENIX Security 22)*. 753–770.

[8] Weiqi Cui, Tao Chen, Christian Fields, Julianna Chen, Anthony Sierra, and Eric Chan-Tin. 2019. Revisiting assumptions for website fingerprinting attacks. In *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*. 328–339.

[9] Wladimir De la Cadena, Asya Mitseva, Jens Hiller, Jan Pennekamp, Sebastian Reuter, Julian Filter, Thomas Engel, Klaus Wehrle, and Andriy Panchenko. 2020. Trafficsliver: Fighting website fingerprinting attacks with traffic splitting. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. 1971–1985.

[10] Xinhao Deng, Qi Li, and Ke Xu. 2024. Robust and Reliable Early-Stage Website Fingerprinting Attacks via Spatial-Temporal Distribution Analysis. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security*.

[11] Xinhao Deng, Qilei Yin, Zhuotao Liu, Xiyuan Zhao, Qi Li, Mingwei Xu, Ke Xu, and Jianping Wu. 2023. Robust Multi-tab Website Fingerprinting Attacks in the Wild. In *IEEE Symposium on Security and Privacy (SP)*.

[12] Roger Dingledine, Nick Mathewson, Paul F Syverson, et al. 2004. Tor: The second-generation onion router. In *USENIX security symposium*, Vol. 4. 303–320.

[13] Kevin P Dyer, Scott E Coull, Thomas Ristenpart, and Thomas Shrimpton. 2012. Peek-a-boo, i still see you: Why efficient traffic analysis countermeasures fail. In *2012 IEEE symposium on security and privacy*. IEEE, 332–346.

[14] Weifeng Ge. 2018. Deep metric learning with hierarchical triplet loss. In *Proceedings of the European conference on computer vision (ECCV)*. 269–285.

[15] Jiajun Gong and Tao Wang. 2020. Zero-delay lightweight defenses against website fingerprinting. In *Proceedings of the 29th USENIX Conference on Security Symposium*. 717–734.

[16] Jiajun Gong, Wuqi Zhang, Charles Zhang, and Tao Wang. 2022. Surakav: Generating Realistic Traces for a Strong Website Fingerprinting Defense. In *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1558–1573.

[17] Zhong Guan, Gang Xiong, Gaopeng Gou, Zhen Li, Mingxin Cui, and Chang Liu. 2021. BAPM: Block Attention Profiling Model for Multi-tab Website Fingerprinting Attacks on Tor. In *Annual Computer Security Applications Conference*. 248–259.

[18] Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, Vol. 2. IEEE, 1735–1742.

[19] Jamie Hayes, George Danezis, et al. 2016. k-fingerprinting: A Robust Scalable Website Fingerprinting Technique. In *USENIX security symposium*. 1187–1203.

[20] Sébastien Henri, Gines Garcia-Aviles, Pablo Serrano, Albert Banchs, and Patrick Thiran. 2020. Protecting against website fingerprinting with multihoming. *Proceedings on Privacy Enhancing Technologies* 2020, 2 (2020), 89–110.

[21] Elad Hoffer and Nir Ailon. 2015. Deep metric learning using triplet network. In *Similarity-Based Pattern Recognition: Third International Workshop, SIMBAD 2015, Copenhagen, Denmark, October 12-14, 2015. Proceedings 3*. Springer, 84–92.

[22] Zhaoxin Jin, Tianbo Lu, Shuang Luo, and Jiaze Shang. 2023. Transformer-based Model for Multi-tab Website Fingerprinting Attack. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*. 1050–1064.

[23] Marc Juárez, Mohsen Imani, Mike Perry, Claudia Dıaz, and Matthew Wright. 2015. WTF-PAD: toward an efficient website fingerprinting defense for tor. *CoRR, abs/1512.00524* (2015).

[24] Mahmut Kaya and Hasan Şakir Bilge. 2019. Deep metric learning: A survey. *Symmetry* 11, 9 (2019), 1066.

[25] Sungyeon Kim, Dongwon Kim, Minsu Cho, and Suha Kwak. 2020. Proxy anchor loss for deep metric learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 3238–3247.

[26] Elad Levi, Tete Xiao, Xiaolong Wang, and Trevor Darrell. 2021. Rethinking preventing class-collapsing in metric learning with margin-based losses. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 10316–10325.

[27] Qi Li, Xinhao Deng, Zhuotao Liu, Yuan Yang, Xiaoyue Zou, Qian Wang, Mingwei Xu, and Jianping Wu. 2022. Dynamic network security function enforcement via joint flow and function scheduling. *IEEE Transactions on Information Forensics and Security* 17 (2022), 486–499.

[28] Weiwei Liu, Haobo Wang, Xiaobo Shen, and Ivor W Tsang. 2021. The emerging trends of multi-label learning. *IEEE transactions on pattern analysis and machine intelligence* 44, 11 (2021), 7955–7974.

[29] David Lu, Sanjit Bhat, Albert Kwon, and Srinivas Devadas. 2018. Dynaflow: An efficient website fingerprinting defense based on dynamically-adjusting flows. In *Proceedings of the 2018 Workshop on Privacy in the Electronic Society*. 109–113.

[30] Jie Lu, Gaopeng Gou, Majing Su, Dong Song, Chang Liu, Chen Yang, and Yangyang Guan. 2021. GAP-WF: Graph attention pooling network for fine-grained SSL/TLS Website fingerprinting. In *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[31] Nate Mathews, James K Holland, Se Eun Oh, Mohammad Saidur Rahman, Nicholas Hopper, and Matthew Wright. 2023. Sok: A critical evaluation of efficient website fingerprinting defenses. In *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE, 969–986.

[32] Asya Mitseva and Andriy Panchenko. [n. d.]. Stop, Don't Click Here Anymore: Boosting Website Fingerprinting By Considering Sets of Subpages. ([n. d.]).

[33] Yair Movshovitz-Attias, Alexander Toshev, Thomas K Leung, Sergey Ioffe, and Saurabh Singh. 2017. No fuss distance metric learning using proxies. In *Proceedings of the IEEE international conference on computer vision*. 360–368.

[34] Milad Nasr, Alireza Bahramali, and Amir Houmansadr. 2021. Defeating DNN-Based Traffic Analysis Systems in Real-Time With Blind Adversarial Perturbations. In *USENIX Security Symposium*. 2705–2722.

[35] Rishab Nithyanand, Xiang Cai, and Rob Johnson. 2014. Glove: A bespoke website fingerprinting defense. In *Proceedings of the 13th Workshop on Privacy in the Electronic Society*. 131–134.

[36] NJ. [n. d.]. How Many Websites Are There in the World? https://siteefy.com/how-many-websites-are-there. Accessed April 20, 2024.

[37] Se Eun Oh, Nate Mathews, Mohammad Saidur Rahman, Matthew Wright, and Nicholas Hopper. 2021. GANDaLF: GAN for data-limited fingerprinting. *Proceedings on Privacy Enhancing Technologies* 2021, 2 (2021).

[38] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. 2016. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4004–4012.

[39] Andriy Panchenko, Fabian Lanze, Jan Pennekamp, Thomas Engel, Andreas Zinnen, Martin Henze, and Klaus Wehrle. 2016. Website Fingerprinting at Internet Scale. In *NDSS*.

[40] Andriy Panchenko, Lukas Niessen, Andreas Zinnen, and Thomas Engel. 2011. Website fingerprinting in onion routing based anonymization networks. In *Proceedings of the 10th annual ACM workshop on Privacy in the electronic society*. 103–114.

[41] Tobias Pulls. 2020. Towards effective and efficient padding machines for tor. *arXiv preprint arXiv:2011.13471* (2020).

[42] Yuqi Qing, Qilei Yin, Xinhao Deng, Yihao Chen, Zhuotao Liu, Kun Sun, Ke Xu, Jia Zhang, and Qi Li. 2023. Low-Quality Training Data Only? A Robust Framework for Detecting Encrypted Malicious Network Traffic. *arXiv preprint arXiv:2309.04798* (2023).

[43] Mohammad Saidur Rahman, Mohsen Imani, Nate Mathews, and Matthew Wright. 2020. Mockingbird: Defending against deep-learning-based website fingerprinting attacks with adversarial traces. *IEEE Transactions on Information Forensics and Security* 16 (2020), 1594–1609.

[44] Mohammad Saidur Rahman, Payap Sirinam, Nate Mathews, Kantha Girish Gangadhara, and Matthew Wright. 2019. Tik-Tok: The utility of packet timing in website fingerprinting attacks. *arXiv preprint arXiv:1902.06421* (2019).

[45] Vera Rimmer, Davy Preuveneers, Marc Juarez, Tom Van Goethem, and Wouter Joosen. 2017. Automated website fingerprinting through deep learning. *arXiv preprint arXiv:1708.06376* (2017).

[46] Meng Shen, Zhenbo Gao, Liehuang Zhu, and Ke Xu. 2021. Efficient fine-grained website fingerprinting via encrypted traffic analysis with deep learning. In *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*. IEEE, 1–10.

[47] Meng Shen, Kexin Ji, Zhenbo Gao, Qi Li, Liehuang Zhu, and Ke Xu. 2023. Subverting website fingerprinting defenses with robust traffic representation. In *32nd USENIX Security Symposium (USENIX Security 23)*. 607–624.

[48] Meng Shen, Yiting Liu, Siqi Chen, Liehuang Zhu, and Yuchao Zhang. 2019. Webpage fingerprinting using only packet length information. In *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 1–6.

[49] Meng Shen, Yiting Liu, Liehuang Zhu, Xiaojiang Du, and Jiankun Hu. 2020. Fine-grained webpage fingerprinting using only packet length information of encrypted traffic. *IEEE Transactions on Information Forensics and Security* 16 (2020), 2046–2059.

[50] Payap Sirinam, Mohsen Imani, Marc Juarez, and Matthew Wright. 2018. Deep fingerprinting: Undermining website fingerprinting defenses with deep learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. 1928–1943.

[51] Payap Sirinam, Nate Mathews, Mohammad Saidur Rahman, and Matthew Wright. 2019. Triplet fingerprinting: More practical and portable website fingerprinting with n-shot learning. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. 1131–1148.

[52] Kihyuk Sohn. 2016. Improved deep metric learning with multi-class n-pair loss objective. *Advances in neural information processing systems* 29 (2016).

[53] Evgeniya Ustinova and Victor Lempitsky. 2016. Learning deep embeddings with histogram loss. *Advances in neural information processing systems* 29 (2016).

[54] Jian Wang, Feng Zhou, Shilei Wen, Xiao Liu, and Yuanqing Lin. 2017. Deep metric learning with angular loss. In *Proceedings of the IEEE international conference on computer vision*. 2593–2601.

[55] Shaobu Wang, Xiangyu Meng, and Tongwen Chen. 2011. Wide-area control of power systems through delayed network communication. *IEEE Transactions on Control Systems Technology* 20, 2 (2011), 495–503.

[56] Tao Wang. 2020. High precision open-world website fingerprinting. In *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 152–167.

[57] Tao Wang, Xiang Cai, Rishab Nithyanand, Rob Johnson, and Ian Goldberg. 2014. Effective attacks and provable defenses for website fingerprinting. In *23rd {USENIX} Security Symposium ({USENIX} Security 14)*. 143–157.

[58] Tao Wang, Ian Goldberg, et al. 2017. Walkie-Talkie: An Efficient Defense Against Passive Website Fingerprinting Attacks. In *USENIX Security Symposium*. 1375–1390.

[59] Xun Wang, Xintong Han, Weilin Huang, Dengke Dong, and Matthew R Scott. 2019. Multi-similarity loss with general pair weighting for deep metric learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 5022–5030.

[60] Yixiao Xu, Tao Wang, Qi Li, Qingyuan Gong, Yang Chen, and Yong Jiang. 2018. A multi-tab website fingerprinting attack. In *Proceedings of the 34th Annual Computer Security Applications Conference*. 327–341.

[61] Qilei Yin, Zhuotao Liu, Qi Li, Tao Wang, Qian Wang, Chao Shen, and Yixiao Xu. 2021. An Automated Multi-Tab Website Fingerprinting Attack. *IEEE Transactions on Dependable and Secure Computing* 19, 6 (2021), 3656–3670.

[62] Yixi Zhang, Xueliang Sun, Xiang Qin, Chaoran Li, Siwei Wang, and Yi Xie. 2021. Tripod: Use data augmentation to enhance website fingerprinting. In *2021 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 1–7.

[63] Ziqing Zhang, Cuicui Kang, Gang Xiong, and Zhen Li. 2019. Deep forest with LRRS feature for fine-grained website fingerprinting with encrypted SSL/TLS. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 851–860.