Hindawi
Security and Communication Networks
Volume 2021, Article ID 6070451, 14 pages
https://doi.org/10.1155/2021/6070451

Research Article

Website Fingerprinting Attacks Based on Homology Analysis

Maohua Guo and Jinlong Fei

State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou 450002, China

Correspondence should be addressed to Jinlong Fei; feijinlong_2021@163.com

Received 11 June 2021; Revised 31 August 2021; Accepted 24 September 2021; Published 4 October 2021

Academic Editor: Weiwei Liu

Website fingerprinting attacks allow attackers to determine the websites that users are linked to, by examining the encrypted traffic between the users and the anonymous network portals. Recent research demonstrated the feasibility of website fingerprinting attacks on Tor anonymous networks with only a few samples. Thus, this paper proposes a novel small-sample website fingerprinting attack method for SSH and Shadowsocks single-agent anonymity network systems, which focuses on analyzing homology relationships between website fingerprinting. Based on the latter, we design a Convolutional Neural Network-Bidirectional Long Short-Term Memory (CNN-BiLSTM) attack classification model that achieves 94.8% and 98.1% accuracy in classifying SSH and Shadowsocks anonymous encrypted traffic, respectively, when only 20 samples per site are available. We also highlight that the CNN-BiLSTM model has significantly better migration capabilities than traditional methods, achieving over 90% accuracy when applied on a new set of monitored sites with only five samples per site. Overall, our experiments demonstrate that CNN-BiLSTM is an efficient, flexible, and robust model for website fingerprinting attack classification.

1. Introduction

With the continuous development of Internet technologies, privacy protection has become one of the most critical concerns. Thus, a continuously increasing number of users protect their anonymity while browsing the Internet by utilizing anonymous network communication systems. However, current research [1–10] shows that privacy can be compromised even though clients use privacy-enhancing technologies such as Shadowsocks [11], I2P [12], Tor [13], Anonymizer [14], SSH, and VPN. Among several cyber-attacks compromising anonymity, the website fingerprinting attack is one of the most representative ones. The core idea of this type of attack is that although the user's communication content is encrypted when visiting different websites, the traffic characteristics generated by each website are unique due to each web page content, e.g., web code, images, scripts, and style sheets. Therefore, the attacker can analyze the anonymous traffic and infer the user's network behavior by passively extracting the traffic between the user and the anonymous network portal using the WF attack.

Current literature [1–5, 7, 8, 10, 15–17] considers website fingerprinting attacks a classification problem. Indeed, the attacker first builds a unique fingerprint model for each website and trains a suitable classifier using the fingerprint features, which can then be used to classify the collected user traffic. Early researchers used machine learning models such as Support Vector Machines [16] (SVM), k-Nearest Neighbors (k-NN) [10], and Random Forests [8], managing an attack accuracy of up to 90%. Nevertheless, in these techniques, the model performance mainly depends on handcrafted features. With the wide application of deep learning techniques in the field of traffic identification, attackers have applied deep learning models to website fingerprinting attacks [1–5, 7, 9], dramatically increasing the attack accuracy and effectively solving the challenging problem of feature extraction and selection. Although the advent of deep learning models has improved the attack accuracy, researchers need to collect hundreds of training samples for each website to enable the neural network to extract high-dimensional fingerprint features. Involving a large training dataset is crucial because when the training sample size is small, the model suffers significantly from

overfitting affecting the model's training process. Simultaneously, the traditional deep models are less flexible, with their performance dropping dramatically when applied to an entirely new classification task.

Spurred by the drawbacks of current deep learning methods, we propose a homology analysis-based approach for website fingerprinting attacks that employ a Siamese Networks [18] structure. Our deep learning architecture analyzes the homology relationship between website fingerprinting features and significantly reduces the training samples required for model training and managing an improved migration capability for the model. The main contributions of our work are as follows:

(i) We study and propose a homology analysis-based website fingerprinting attack model, relying on a Convolutional Neural Network-Bidirectional Long Short-Term Memory (CNN-BiLSTM), which achieves 94.8% and 98.1% attack accuracy in a closed world composed of encrypted traffic from SSH and Shadowsocks anonymity networks, respectively, with only 20 training samples per site. The performance of the proposed architecture is significantly better compared to traditional methods.

(ii) We innovatively construct one-hot matrices by sequence symbolization to represent the direction, size, and time interval attributes exposed in the traffic sequences. This strategy improves the data feature dimensionality and the fault tolerance for sample burst features.

(iii) Compared to previous studies, we design a more challenging scenario to evaluate the model's migration capability. Specifically, we complete training using SSH anonymous network encrypted traffic and then utilize the trained model to classify Shadowsocks anonymous network encrypted traffic. The results demonstrate that, with only five sample attacks per site, our technique exceeds 90% classification accuracy.

The remainder of this paper is organized as follows. Section 2 summarizes and reviews previous approaches to website fingerprinting attacks. In Section 3, we present the threat model for website fingerprinting attacks and the design of the CNN-BiLSTM model. Section 4 summarizes the datasets used and the data processing methods, while Section 5 provides the results of our experiments and the corresponding analysis. The limitations of our work and directions for future research are discussed in Section 6. Section 7 concludes this work.

2. Background and Related Work

Website fingerprinting attacks use a passive traffic analysis technique. The attacker first configures a network environment similar to the monitored target, exploits the same anonymous network encryption proxy to access each site in the monitored set, and collects adequate training samples. After that, the attacker builds a fingerprint library for each monitored site and identifies the actual address of the user's communication counterpart by analyzing, extracting, and comparing the features of the communication traffic obtained during monitoring.

In 1998, Cheng et al. [19] were the first to apply the website fingerprinting attack to traffic identification by using the feature of file size to identify some specific SSL-protected files. With the rise of anonymous networks, Herramnn et al. [17] in 2009 performed website fingerprinting to identify JAP, Tor, OpenSSH, OpenVPN, Stunnel, and CiscoIPsec-VPN. In 2011, Panchenko et al. [16] introduced a unique traffic burstiness combined with an SVM algorithm that achieved a 54% identification rate for Tor traffic. In subsequent studies, Wang et al. [10] extracted over 3000-dimensional feature vectors to model website fingerprinting and employed a weighted distance-based metric and a k-NN classifier to measure the similarity of website fingerprinting. Panchenko et al. [15] proposed the CUMUL method that exploited the feature of cumulative packet size, while Hayes et al. [8] proposed a random forest-based attack method (k-FP) to describe website fingerprinting by selecting 150 important features from the total 4,000 dimensions. Current methods are implemented by handcrafted feature sets combined with machine learning algorithms for website fingerprinting attacks and managing an accuracy exceeding 90%.

With the development of deep learning techniques in image, speech, and video, researchers have extended using deep learning schemes for website fingerprinting attacks. In 2016, Abe et al. [9] first succeeded using a Stacked Denoising Autoencoder (SDAE) for website fingerprinting attacks. In 2017, Rimmer et al. [7] extensively evaluated the performance of deep learning methods such as SDAE, CNN, and LSTM in a dataset consisting of 900 sites (each with 2500 samples). The reported results revealed that CNN provided the best results, achieving 96.66% accuracy in a closed world, while in an open environment, it achieves a TPR of 71.3% and an FPR of 3.4%. In 2018, Sirinam et al. [5] designed a more complex deep learning model (DF) with a deeper network structure that involves more convolutional and Batch Normalization layers. It eventually achieved 98.3% accuracy in a closed world consisting of 95 websites and 99% accuracy in an open world with 94% recall.

However, using deep learning for website fingerprinting attacks requires a large number of training samples per site. Hence, to solve this problem, Sirinam et al. [3] in 2019 first designed a Triplet Fingerprinting (TF) method for website fingerprinting attacks using a small-sample technique [18, 20–22], which involved a triplet network including an anchor (A), positive (P), and negative (N) as subunits of the triplet network. This method employs the cosine distance algorithm to measure the relationship between A-P and A-N, so that A and P are close to each other, while A and N are far away in the embedding space generated by the model. This means that the feature vectors generated by the same website sample traffic are close to each other, and the feature vectors generated by different website sample traffic are far apart. After training, the trained model is used as a feature extractor for website traffic, and then k-NN is used as a

classifier to finally achieve 95% accuracy requiring a small number of samples per website. Oh et al. [1], in 2021, first proposed another highly representative fingerprinting attack technique for low data sites, entitled GANDaLF, based on generative adversarial networks (GAN). This method uses a small number of labeled data and a more extensive unlabeled set to train the generator and the discriminator. The generator is trained to convert random seeds into pseudotraces with the same statistical distribution as the training data. The discriminator is trained to correctly exploit data for classification while distinguishing between the generator's true traces and pseudotraces output. This approach uses the generator as an additional data source to help improve the performance of the discriminator, making the website fingerprinting attack effective even in low data settings.

3. Attacks Based on Homology Analysis

3.1. Attack Threat Model. The website fingerprinting attack aims to disrupt the user's anonymity while visiting a website by utilizing traffic analysis; that is, the eavesdropper can infer the target websites visited by users from the encrypted anonymous traffic, with the primary attack model presented in Figure 1.

In this paper, we adopt the important assumptions of a website fingerprinting attack; that is, the attacker can only obtain the network packets on the communication link passively and cannot modify, delete, or insert any packet and encrypt, decrypt, or analyze the packets directly. The attacker collects the traffic, compares it with previously known traffic characteristics such as packet size, direction, and time interval, and finally finds the best match to the targeted website data stream record. In this way, the attacker is informed about the websites visited by a user and thus compromises the user's anonymity.

3.2. Website Fingerprinting Homology Analysis. The essence of the website fingerprinting attack is matching traffic characteristics, which is essentially the same goal as the homology detection of proteins and DNA in biology. Both scenarios aim to find similar segments between sequences, so we consider homology analysis feasible for the website fingerprinting attack. The homology analysis methods are commonly used in biology and are divided into three categories [23]: comparison-based, ranking-based, and discriminative-based methods. The most commonly used comparison-based methods are sequence, sequence spectrum, and HMM comparison, i.e., comparing sequences by dynamic programming and scoring functions. For example, in 2017, Zhuo et al. [6] implemented a website fingerprinting attack using a PHMM model. The core idea of the sorting-based approach is to regard homology detection as an information retrieval problem and sort the known sequences in the database and the unknown query sequences according to the homology relationship from near to far. The critical process of this method is the design of the sorting algorithm. According to the closeness of homology relationship, the discriminative-based approach involves dividing the

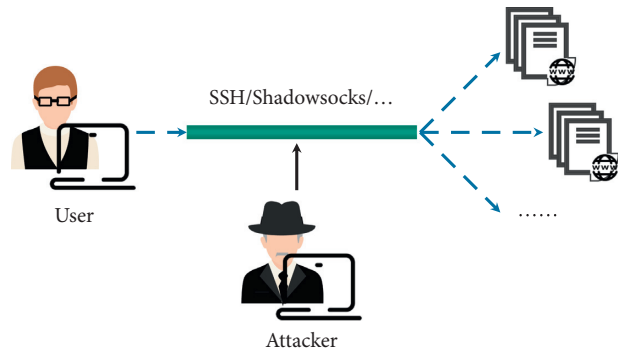

Figure 1: Threat structure model for website fingerprinting attacks.

sequences into positive and negative sample training and test sets. Then use the sequences in the training set to train the classification model based on machine learning and deep learning, and the test set evaluates the classifier's performance.

Traditional website fingerprinting attacks using deep neural networks require a large amount of data, and when the training data is insufficient, the model is less effective during classification. Additionally, the website content changes significantly over time, and these changes affect the website fingerprinting features. Therefore the model needs to be retrained after a while. At the same time, the migration ability of the model is weak, and the classification accuracy will drop significantly when the trained model is applied to a new classification task.

In this paper, we adopt a discriminative approach for website fingerprinting homology analysis. Unlike the traditional direct classification of website fingerprinting using machine learning and deep learning models, we adopt the structure of Siamese Networks [18]. During training, the purpose of our model is to change from directly attributing traffic sequences to corresponding website categories and train the network to learn the correlation between website traffic features, that is, the homology between website fingerprinting. This is achieved by using less data for model training to achieve a higher accuracy rate of website fingerprinting attacks.

3.2.1. Siamese Networks. Siamese Networks are a particular type of neural network structure, which, unlike a network model that learns to classify inputs directly, aim to learn the similarities and the correlations between the two inputs. The model selects the most likely identical category for a classification task by comparing each example in the test set with the training set. The Siamese Networks consider two samples on the input simultaneously and finally output the probability that they belong to the same category.

As shown in Figure 2, the Siamese Networks have two inputs X_1 and X_2, in each cell structure, where X_1 and X_2 are input into the neural networks Network_1 and Network_2 with shared weights (in the usual case, it can be considered that Network_1 and Network_2 are two identical neural network structures). Then, a similarity measure algorithm is used to calculate the distance between the high-dimensional

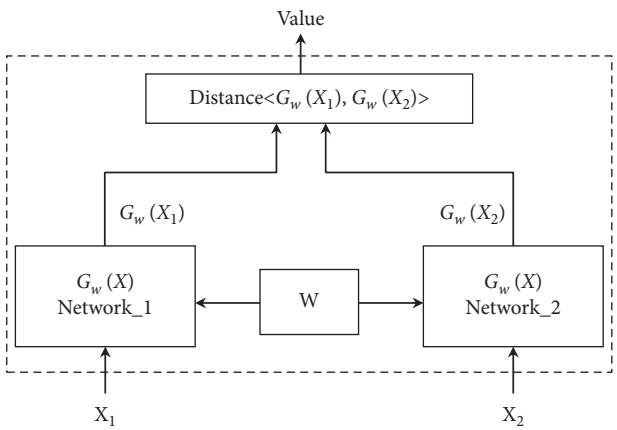

Figure 2: The structure model of Siamese Networks unit.

features $G_w(X_1)$ and $G_w(X_2)$ extracted by the neural network and the output value as the correlation measure of X_1 and X_2.

The training test of the Siamese Networks contains multiple Siamese Network units, and each twin unit accepts two input data. Figure 3 illustrates the training test structure of the Siamese Networks, including the input, network, distance, and output layers. The input layer combines the input data, and the two inputs are logically symmetric because the network layer weights are shared, and the network structure is consistent. The network layer uses deep neural networks to extract high-dimensional features from the input data, commonly used as a CNN. The distance layer calculates the correlation between the high-dimensional features, and the typically used distance metrics are the cosine and sine. The output layer uses the results of the distance layer to get the probability that two inputs belong to the same category.

3.2.2. Dataset Construction Method. Each unit of a Siamese Network requires two inputs, and therefore the dataset needs to be correctly reconstructed. Assuming that N websites are to be classified, the training and test sets are defined by

$$\begin{cases} S_{\text{train}}(k) = S^{+}_{\text{train}}(k) \cup S^{-}_{\text{train}}(k) \\ S_{\text{test}}(k) = S^{+}_{\text{test}}(k) \cup S^{-}_{\text{test}}(k) \end{cases} \quad (k = 1, 2, 3, \ldots, N), \tag{1}$$

where k denotes the dataset for the k-th website prediction classification, $S^{+}_{\text{train}}(k)$ and $S^{-}_{\text{train}}(k)$ are the positive and negative sample training set for the k-th website, respectively, and $S^{+}_{\text{train}}(k)$ and $S^{-}_{\text{train}}(k)$ together constitute the training set $S_{\text{train}}(k)$ for the k-th website. $S^{+}_{\text{test}}(k)$ denotes the positive sample test set of the k-th website, $S^{-}_{\text{test}}(k)$ denotes the negative sample test set of the k-th website, and $S^{+}_{\text{test}}(k)$ and $S^{-}_{\text{test}}(k)$ together constitute the test set $S_{\text{test}}(k)$ of the k-th website:

$$\begin{cases} S^{+}_{\text{train}}(k) = k^{+}_{i} \cup k^{+}_{j} \\ S^{-}_{\text{train}}(k) = k^{+}_{i} \cup k^{-}_{l} \end{cases} \quad (1 \le i < j \le P,\ 1 \le l \le P). \tag{2}$$

We assume that each website provides P samples for model training (equation (2)), k^{+}_{i} and k^{+}_{j} denote any two training samples from the k -th website, and the two inputs of the Siamese Networks unit are logically symmetric. Then, $S^{+}_{\text{train}}(k) = k^{+}_{i} \cup k^{+}_{j}$ denotes that $S^{+}_{\text{train}}(k)$ consists of any two training samples from the k-th website, with k^{-}_{l} referring to the training samples of other sites than the training samples of the k-th site. To balance the number of samples of $S^{+}_{\text{train}}(k)$ and $S^{-}_{\text{train}}(k)$ in the training set, we select only one random sample as k^{-}_{l} for each site other than the training samples of the k-th site, and $S^{-}_{\text{train}}(k) = k^{+}_{i} \cup k^{-}_{l}$ indicates that the two combinations of k^{+}_{i} and k^{-}_{l} together form a negative sample training set for the k-th website.

$$\begin{cases} S^{+}_{\text{test}}(k) = k^{+}_{i} \cup k^{+}_{j} \\ S^{-}_{\text{test}}(k) = k^{+}_{i} \cup k^{-}_{l} \end{cases} \quad (1 \le i < j \le Q,\ 1 \le l \le Q). \tag{3}$$

We also assume that each site provides Q samples for model test evaluation (equation (3)), and then under the same principle, we obtain the positive sample test set $S^{+}_{\text{test}}(k)$ and the negative sample test set $S^{-}_{\text{test}}(k)$ for the k-th site.

3.3. CNN-BiLSTM-Based Siamese Networks Attack Model Construction

CNN. A convolutional neural network has four significant features, that is, the local perceptual domain, shared weights, pooling, and multilayer network, which can capture the complex features in the original data, and therefore it is widely used to process serial and image data. The original data is convolved with the local perceptual domain, and shared weights are utilized to form a feature map composed of local features. These are then passed through the pooling layer for integration and to perform data dimensionality reduction. The in-depth features involve high-dimensional, complex, and abstract features created after several convolutional and pooling layers. In previous studies [3, 5, 7], CNNs have been widely used as the dominant feature extraction method for website fingerprinting attacks.

LSTM. The long short-term memory network dynamically processes the input sequence according to the time series, and the output processed in the previous time step is used as the input on the next time step. At the same time, LSTM achieves the purpose of blocking irrelevant information, absorbing relevant information, and maintaining information in a cell state through the collaboration among input gates, forgetting gates, and output gates, which solves the problem of gradient disappearance and gradient explosion often encountered in the training process of recurrent neural networks (RNN). Therefore, LSTM is widely used in sequence information processing. The possibility of using LSTM for website fingerprinting attacks was also discussed in [8].

As shown in Figure 4, our deep learning architecture uses a combined network comprising a CNN and a

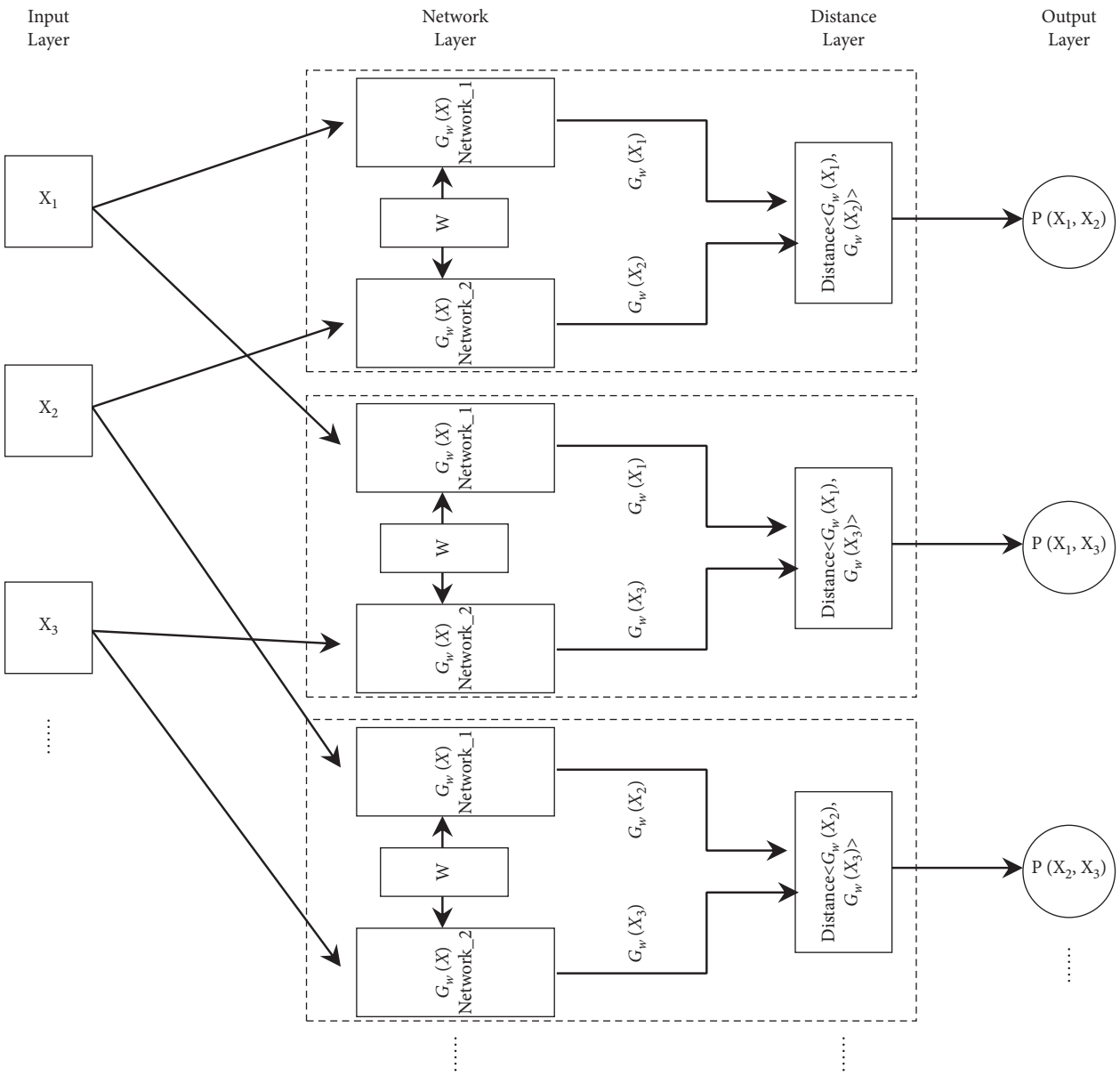

Figure 3: The structure model of Siamese Networks training and testing.

bidirectional LSTM (BiLSTM) as the base model of the Siamese Networks. Firstly, the CNN is used to extract the high-dimensional features of the two original input sequences, and then the dependencies in the high-dimensional features of the sequences are extracted through the BiLSTM layer. However, due to the long sequences generated by the network traffic, the output of LSTM at the last time step cannot represent the dependencies containing all subsequences, so we consider using the intermediate output of LSTM at each time step to better handle the local and global dependencies between the traffic sequences and the captured subsequences. At the same time, we choose a BiLSTM to replace the commonly used unidirectional LSTM. The forward LSTM in the BiLSTM model can extract the dependencies between the current input subsequence and its left subsequence, while the backward LSTM can extract the dependencies between the current input and its right subsequence. Hence, the concatenation of these two intermediate outputs allows for more comprehensive information on the dependencies between the sequences. In the distance layer of twin networks, traditional distance measurement metrics such as cosine, sine, Euclidean, or other linear ones often underperform in evaluating the correlation between the high-dimensional features of the sequences. Thus, in this paper, we consider using fully connected neural networks as the distance measuring function. The features extracted from two original sequences are spliced, combined, and input to the fully connected layer to evaluate the homology relationship between the traffic sequences.

3.4. Model Parameters. To select the optimal hyperparameters for our model, we evaluate several CNN-BiLSTM model structures and parameters using the

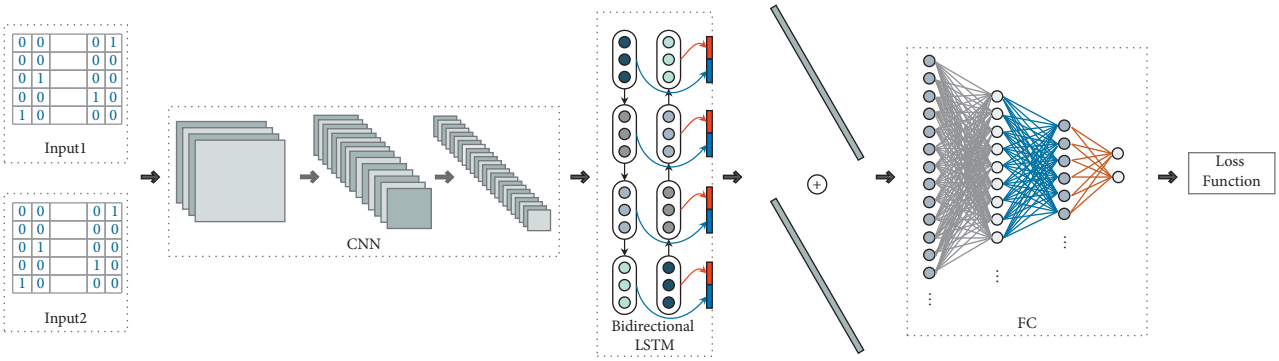

Figure 4: Structure of CNN-BiLSTM attack mode.

extensive candidate search method. Table 1 presents some of the critical parameter search spaces and the final selection.

We use Layer Normalization [24] for the Batch Normalization layer because the number of training samples we exploit is small, and Batch Normalization [24], which uses the mean and variance of the samples, does not reflect the global statistical distribution. Nevertheless, the Layer Normalization algorithm is independent of the batch size, and its statistics depend on the number of nodes in the hidden layer. For the network's activation function layer, we choose LeakyReLU [25], which presents the advantage of avoiding the neuron "death" faced by ReLU during training, reduces the parameters that need to be debugged, and improves training speed.

4. Dataset

4.1. Data Collection. The datasets used in this experiment are Liberatore's open dataset [26] and the Shadowsocks [6]. As shown in Table 2, we also exploit two open datasets to construct the closed world and open-world datasets required for the experiment.

4.1.1. Closed World

SSH-200 Dataset. Built from the Liberatore open dataset, this dataset contains encrypted network traffic data from 2000 different sites accessed using SSH tunnels. However, this dataset involves many empty packets due to various failures during the collection process. For consistency, this experiment screens out sites with average instance SSH packet sequence length greater than 100 (based on the original dataset) and randomly selects 200 sites from them, with 25 instances selected for each site to generate the SSH-200 dataset.

Shadowsocks-200 Dataset. 200 different domains were randomly selected from the top 1000 Alexa rankings, and each domain was accessed 25 times each using Shadowsocks tunneling encryption to generate the Shadowsocks-200 dataset.

4.1.2. Open World

SSH-2000 Dataset. One randomly selected instance from Liberatore's open dataset generates the SSH-2000 dataset for each site.

Shadowsocks-2000 Dataset. It includes randomly 2000 selected websites from Alexa top 1000 to 10000 and uses Shadowsocks tunnel to visit each website only once to generate Shadowsocks-2000 dataset.

4.2. Data Processing. We process packets to filter out fragmented packets that do not provide reliable information in transmission, including missing, retransmitted, ACK loss, duplicate answers, and transmission packets with zero data segment length. Since the subject of this paper is SSH and Shadowsocks anonymous network encrypted traffic without restrictions on the size of transmission units and packet delays like Tor [10], we extract the size, transmission direction, and time interval from each payload packet as the original sequence features.

This paper uses a one-hot matrix [23] to represent the original feature data, which requires sequence symbolization and construction of one-hot matrix processing for the original direction, size, and time interval feature sequences. After processing, we extend the feature dimension and the homology relationship between website fingerprinting features to enhance the measured feature distance.

4.2.1. Sequence Symbolization. Algorithm 1 describes the symbolization steps of the packet size and feature data direction, where the first two lines input the original packet sequence into the algorithm and extract them in order. Lines 3 to 7 merge the two attributes of size and direction, and lines 8 to 10 maximize the highlighted direction and size attributes in the form of double-symbol bits based on the maximum transmission unit MTU and the standard number of symbols Num. Finally, the double symbols are filled in cyclically to obtain the symbolized sequence *S&D* Seq.

Algorithm 2 describes the symbolization step of the packet time interval feature data with the input of the standard number of symbols Num and the maximum symbolization time interval Maxtime. The first two lines

Table 1: Model hyperparameter search space and final selection.

Hyperparameters	Search space	Selected value
Number of filters		
Conv2d1	[8 . . . 32]	16
Conv2d2	[16 . . . 64]	32
Normalization methods	[Batch Normalization, Layer Normalization]	Layer Normalization
Activation functions	[ReLU, ELU, LeakyReLU]	ReLU
Pooling layers	[Average, max]	Max
BiLSTM	[64 . . . 256]	128
Number of FC layers	[1 . . . 4]	3
[Filter, pool, stride] sizes	[2 . . . 8]	[3, 3, 1]
Loss function	[Cross-entropy loss]	Cross-entropy loss
Optimizer	[SGD, adam, Adamax, RMSProp]	Adam
Learning rate	[0.0001 . . . 0.01]	0.001
Training epochs	[10 . . . 50]	30
Minibatch size	[16 . . . 64]	48

Table 2: Dataset used in the experiment.

Name	Anonymous method	Training set	Test set	Purpose
SSH-200	SSH	200×20	200×5	Close world
Shadowsocks-200	Shadowsocks	200×20	200×5	Close world
SSH-2000	SSH	N/A	2000×1	Open world
Shadowsocks-2000	Shadowsocks	N/A	2000×1	Open world

Input: Packets Sequence Seq, Number of symbols Num
Output: Size and direction symbol sequences *S*&*D* Seq
Steps:
(1) *S*&*D*Seq ⟵ Null
(2) **for** packet $a \in$ Seq **do**
(3) **if** *a*.Direction = " + " **then**
(4) *a*.size ⟵ MTU.size + *a*.size
(5) **else**
(6) *a*.size ⟵ MTU.size − *a*.size
(7) **end if**
(8) *a.S*&*D*.interval ⟵ $(2 \times \text{MTU.size})/\text{Num}^2$
(9) *a.S*&*D*.symbol[0] ⟵ Symbol (*a*.size/ (*a.S*&*D*.interval × Num))
(10) *a.S*&*D*.symbol[0] ⟵ Symbol (*a*.size% (*a.S*&*D*.interval × Num))
(11) *S*&*D* Seq.append (*a.S*&*D*.symbol)
(12) **end for**

Algorithm 1: Size and direction symbolization algorithm.

indicate that the original packet sequence is input to the algorithm, and the average symbolization time interval Time.interval is calculated based on the standard number of symbols Num and the maximum symbolization time interval Maxtime. The time of the first packet is also set as the base time. Lines 3 to 8 symbolize the time interval characteristics of the original packet sequence by first calculating the sequential two packet time interval ΔTime, which is set as a fixed character if the time interval ΔTime is greater than the maximum symbolization time interval. If the time interval ΔTime is less than the maximum symbolization time interval, each time interval Time.interval corresponds to a symbol. Finally, the symbols are filled in cyclically to obtain the symbolization sequence *T*Seq.

4.2.2. Building a One-Hot Matrix. The original sequence is symbolized and can be expressed by

$$\text{Seq} = S_1, S_2, S_3, \ldots, S_L, \tag{4}$$

where S_i denotes the i-th character of the symbolized sequence Seq and L denotes the length of the sequence. In this paper, the one-hot matrix, commonly used to represent DNA, RNA, and protein sequences in biology, represents the

Input: Packets Sequence Seq, Number of symbols Num, Max Time interval Maxtime,
Output: Time interval symbol sequences *T*Seq
Steps:
(1) *T*Seq ⟵ Null, Time.base ⟵ First packet.Time.now
(2) Time.interval←Maxtime/Num
(3) **for** packet $a \in$ Seq **do**
(4) ΔTime ⟵ *a*.Time.now − Time.base
(5) **if** ΔTime ≥ Maxtime **then**
(6) *a*.Time.symbol ⟵ Symbol (Max)
(7) **else**
(8) *a*.Time.symbol ⟵ Symbol (ΔTime/Time.interval)
(9) **end if**
(10) Time.base ⟵ *a*.Time.now
(11) *T*Seq.append (*a*.Time.symbol)
(12) **end for**
(13) **return** *T*Seq

Algorithm 2: Time interval symbolization algorithm.

symbolized sequences. For a sequence Seq, its one-hot matrix can be expressed as

$$M = \begin{bmatrix} e_{1,1} & \cdots & e_{1,L} \\ \vdots & \ddots & \vdots \\ e_{\text{num},1} & \cdots & e_{\text{num},L} \end{bmatrix}, \quad e_{i,j} = \begin{cases} 1, & S_j = \text{Symbol}_i, \\ 0, & \text{otherwise}, \end{cases} \tag{5}$$

where num denotes the number of standard characters and Symbol_i denotes the i-th standard character $(1 \le i \le \text{num})$. Intuitively, each character of the symbolized sequence can be represented by a num-dimensional vector, and only this character is activated in this vector. The value of this dimension is one, and the rest of the dimensions are zero.

To facilitate the training of the neural network model, we normalize the length L of the symbolized sequence. When the sequence length is greater than the preset normalized value L, we truncate the sequence, and if the length does not satisfy L, we complement it with zero (the num dimensional vector corresponding to zero in constructing a one-hot matrix is the zero-vector). Finally, all the original sequences are processed into num × L matrices.

5. Experimental Evaluation

5.1. Assessment Indicators. To evaluate the experimental results, we use the following evaluation metrics: accuracy, true positive (TP), false positive (FP), true negative (TN), false positive (FP), precision, and recall. Accuracy indicates the ratio of the number of website categories correctly identified to the total number of websites in the same test set and is calculated by

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} = \text{FN}} \times 100\%, \tag{6}$$

where TP is the number of monitored websites correctly classified, FP is the number of unmonitored websites incorrectly classified as monitored, TN is the number of unmonitored websites correctly classified, and FN is the number of monitored websites incorrectly classified as different monitored or unmonitored websites. Recall refers to the percentage of monitored sites among the sites correctly classified by the classifier, and precision and recall are calculated by

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \qquad \text{recall} = \frac{\text{TP}}{\text{TP} + \text{TN}}. \tag{7}$$

5.2. Closed World Assessment. We evaluate the proposed model in the closed world case using SSH-200 and Shadowsocks-200 and demonstrate the parameter's interplay with the overall model's performance.

The accuracy of the model tested in the dataset SSH-200 is shown in Table 3. In a closed world and given some parameter setting conditions, our proposed CNN-BiLSTM model requires only 20 training samples and achieves up to 94.8% accuracy, performing significantly better than the traditional machine learning k-FP, k-NN, and PHMM models. Moreover, compared to the recently emerging small-sample website fingerprinting attack methods, the test results are slightly better overall than TF, the small-sample website fingerprinting attack model first proposed by Sirinam et al. in 2019 [3]. Additionally, our method's optimal test accuracy is equal to that of GANDaLF, the current state-of-the-art and data fingerprinting attack model proposed by Oh et al. [1].

In this section, we design comparative experiments to investigate the impact of using different combinations of traffic features and data representations on the accuracy of fingerprinting attacks. In the closed world, we employ the original direction and size features, that is, Raw Size&Direction, and the original direction, size, and packet spacing combination features, that is, Raw Size&Direction, ΔTime, the one-hot processed

Table 3: Test accuracy of dataset SSH-200 (%).

Method name	*Test methods*	*L = 200*	*L = 300*	*L = 400*	*L = 500*
CNN-BiLSTM	Raw Size&Direction	87.2	88.5	**89.3**	88.6
	Raw Size&Direction, ΔTime	85.6	87.9	88.3	**89.7**
	Directional Timing	86.9	88.9	**90.2**	89.8
	S&D Seq, one-hot	92.8	**93.4**	93.1	92.2
	S&D Seq, *T*Seq, one-hot	93.7	**94.8**	94.1	93.9
TF	*S&D* Seq, *T*Seq, one-hot	92.9	**94.1**	93.5	93.2
GANDaLF	*S&D* Seq, *T*Seq, one-hot	94.3	94.6	**94.9**	94.7
PHMM	*S&D* Seq, *T*Seq	85.9	87.3	**88.2**	86.5
Method name	*Test methods*		*K = 1*	*K = 2*	*K = 3*
k-FP	*S&D* Seq, *T*Seq		90.6	**91.2**	91.1
k-NN	*S&D* Seq, *T* Seq		**90.8**	86.4	82.3

S&D Seq matrix, and the one-hot processed *S&D* Seq and *T*Seq combined matrix. Also, we compare our technique with the newly proposed directional timing-based attack (Tik-Tok attack) by Rahman et al. [2] in 2020. Table 3 highlights that the attack accuracy of the model can be improved by 4-5 percentage points using our proposed data representation technique compared with the direct use of raw traffic features and is significantly higher than the Tik-Tok approach using the combination of packet direction and timestamp features.

Meanwhile, we count the packet sequence lengths of the visited sites in the SSH-200 dataset. Figure 5 highlights that more than 75% of the sites have sequence lengths within 500, and thus, we set $L = 200$, 300, 400, and 500. It can be seen from Table 3 that the highest accuracy of the model classification, when tested directly using the original feature sequences of size and direction, is 89.3%, and the model classification accuracy decreases slightly because of the feature increment introduced in the dimension of the time interval. The latter is due to exploiting only 20 training samples and the subtle perturbation brought by the change of time interval affects the model's final training effect.

Additionally, due to the introduction of packet size and time symbolization interval, the original feature sequence after data processing presents for the same site multiple sample collections, imposing data changes in a particular range that does not change the symbol but improves the stability of the site fingerprint data features, making these statistical features uniquely representing a site. Therefore, after data processing, adding the dimensional feature of time interval improves the classification accuracy by 1.5%, and the model's highest attack accuracy is achieved at $L = 300$. Using the combined sequence of *S&D* Seq and *T*Seq after the one-hot matrix processing, the accuracy increases to 94.8%. The test results in Table 3 also indicate that, after data processing, as the normalized sequence length L increases, the model reaches the peak classification accuracy earlier. This is because the one-hot matrix introduces more zero elements in the vector while expanding the feature dimension, and the increase of the normalized sequence length L leads to more and more traffic sequences generated by the sites needing to be zero-complemented, making the sequences look more similar to each other after data processing.

The test results in Table 3 reveal that the highest classification accuracy is improved by nearly 5% after symbolizing the original feature data and constructing the one-hot matrix. We designed the following validation experiments to analyze the interplay between the number of standard symbols (packet size symbolization interval and time symbolization interval) and the accuracy during the symbolization process.

Figure 6 presents the model attack accuracy curves, where the number of standard symbols Num involves sequence lengths of $L = 200$, 300, and 400. It is clear that the accuracy rate keeps improving with the increase of Num (for $0 \leq \text{Num} \leq 20$), and the attack performance of the model reaches the optimum when the standard number of symbols is Num = 20. After that, the performance of the model starts to gradually decrease (for $\text{Num} \geq 20$). Hence, we conclude that the model's performance is related to the size of the symbolized interval division. When the standard number of symbols Num is small, the symbolization interval is large. The serialization process is more fault-tolerant to minor variations in packet size and time intervals in different samples from the same site. These features allow the model to categorize the samples originating from the same site, but a too-large interval will lead to the sequence not being obvious enough. The sequence generated by the samples of different sites varies less, which is not conducive to the differentiation of different sites, thus affecting the model's overall performance. When the number of standard symbols Num is larger, the symbolization interval is smaller. After symbolizing the original data, the samples from different sites will have apparent differences, which is beneficial to classify samples from different sites. However, for the different samples generated by multiple visits to the same site, the perturbations generated by the packet size and time interval change will show more apparent differences in their symbolization sequences, which is not conducive to the homology analysis. This is because samples from the same site will affect the classification ability of the model.

The tested accuracy of the CNN-BiLSTM model on the dataset Shadowsocks-200 is shown in Table 4. The model remains efficient in classifying and identifying Shadowsocks anonymous encrypted traffic, achieving a maximum attack accuracy of 98.1% with only 20 training samples per site when classifying against SSH anonymous encrypted traffic.

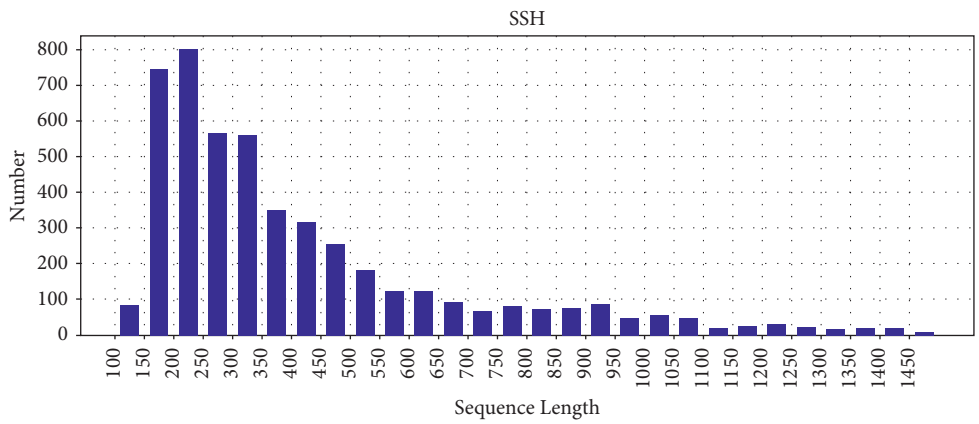

Figure 5: Statistics of site sequence length in SSH-200 dataset.

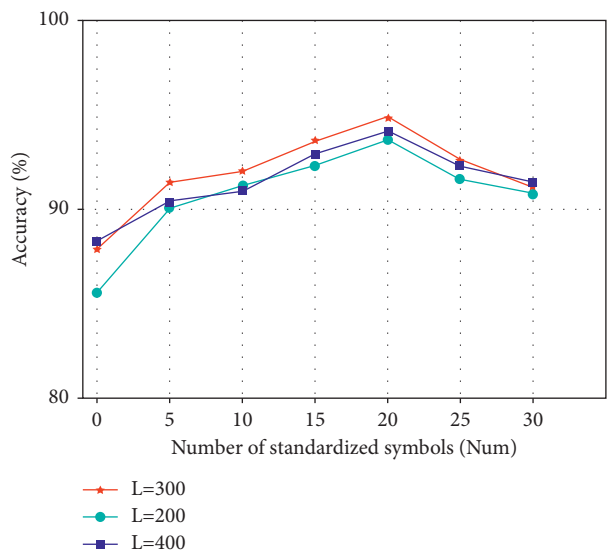

Figure 6: Curve of model accuracy with an increasing number of standard symbols.

Table 4: Test accuracy of dataset Shadowsocks-200 (%).

Method name	$L = 200$	$L = 300$	$L = 400$
CNN-BiLSTM	97.6	**98.1**	96.3

This shows that each site's packet direction, size, and time interval in the Shadowsocks anonymous environment are more prominent, while each site's traffic has fewer burst features and a smoother state, making it easier for eavesdroppers to perform website fingerprinting attacks.

5.3. Migration Capability Assessment. Transfer learning [27] is a deep learning-related technique, where an already trained CNN is partially retrained on an entirely new classification task. The performance of the newly trained model involves measuring its migration ability. Deep learning models can automatically extract data features from large amounts of data by semisupervised or unsupervised feature learning algorithms and hierarchical feature extraction schemes and manage a higher classification accuracy than traditional machine learning methods. However, traditional website fingerprinting classification methods that employ deep learning, such as DF and AWF, require the training and test data to be independent and codistributed. If a model trained in the monitored website dataset collection A is used to classify fingerprint data in the untrained monitored website collection B, the attack accuracy of the deep learning model will drop drastically. Additionally, much time is required to collect the monitored website data in collection B and retrain the attack model, which is unacceptable to the attacker.

To evaluate the migration capability of the model, we consider a more challenging scenario and conduct experiments using the SSH-200 and Shadowsocks-200 datasets.

SSH and Shadowsocks are two completely different anonymous communication systems producing very different fingerprint data characteristics and collect significantly different site information. Our model is trained using one dataset, and the trained model is retrained by randomly selecting $R\,(R \leq 10)$ samples for each site in the other dataset, with the latter dataset also exploited as a testing dataset to evaluate the model's classification accuracy. Considering our trials, we evaluate the classification accuracy of the CNN-BiLSTM, TF, AWF, DF, and GANDaLF models with SSH anonymous fingerprint data as the training set and employ the Shadowsocks anonymous fingerprint data as the test set. The corresponding results are illustrated in Figure 7.

As seen in Figure 7, the TF, GANDaLF, and CNN-BiLSTM models significantly outperform the traditional deep learning models. Since the test set and the training set are different types of traffic data, the data distribution is weakly correlated, and the trained model is directly applied to the classification task of the Shadowsocks dataset. The accuracy of the traditional deep learning AWF, DF, and GANDaLF models based on the GAN network is less than 10%. In comparison, the attack accuracy of both TF and CNN-BiLSTM models exceeds 70%. As the number of samples (R) involved in the transfer learning process (secondary training) increases, the model's attack accuracy gradually improves with TF and CNN-BiLSTM's accuracy when $1 \leq R \leq 3$, but in principle, this improvement effect remains the same. The accuracy of TF and CNN-BiLSTM stabilizes above 90%, and when $R = 10$, the CNN-BiLSTM model accuracy is close to 92%, which is a 6% improvement over the TF method. The GANDaLF model has a significant improvement in attack accuracy as the sample number R increases due to its robust data generation capability, managing a close to the TF model performance for $R = 10$, and the accuracy curve still maintains a slow upward trend. The accuracy improvement effect of the traditional methods AWF and DF as the sample number R increases is more evident than TF and CNN-LSTM methods but much lower than GANDaLF model. The accuracy rate is already close to 50% at $R = 10$, but still, 40% lower compared with the CNN-LSTM method. This indicates that traditional deep learning models have limitations in adapting to new classification tasks and that CNN-LSTM, TF, and GANDaLF models can all better mitigate the adverse effects of data mismatch. However, the CNN-LSTM method has better migration ability in environments where samples are lacking.

5.4. Open-World Assessment. The performance of classifiers in the open world is another essential evaluation metric in website fingerprinting attacks. The goal is to assess the ability of the model to distinguish traffic generated by monitored websites from traffic generated by any other unknown websites. We use precision and recall to evaluate the CNN-BiLSTM model in an open-world scenario by plotting the precision-recall curve.

This section evaluates the model's performance in the SSH and Shadowsocks anonymous communication systems. To balance the number of monitored site samples with the number of monitored samples, we randomly select 10 samples for each site from the SSH-200 and Shadowsocks-200 datasets to construct a monitored test sample set. The latter is then combined with the SSH-2000 and Shadowsocks-2000 datasets to form the SSH and the Shadowsocks open-world test set. At the same time, to better distinguish the monitored and unmonitored sites, we use the standard model during training and treat the unmonitored sites as an additional label.

Figure 8 presents the precision-recall curves of the CNN-BiLSTM model for sequence lengths of $L = 200$, 300, and 400 in the SSH and Shadowsocks open world. This figure highlights that the accuracy and recall rates are better in Shadowsocks than in SSH, which indicates that the model is more suitable for Shadowsocks' open-world environment for website fingerprinting attacks. As the recall rate increases, the classification accuracy rate significantly decreases for SSH and Shadowsocks but is still between 0.7 and 0.8. Also, in both environments, the model performance is optimal for a sequence length of $L = 300$.

Under small-sample conditions, we further evaluate two extremely optimal models for website fingerprinting attacks in the open world: TF and GANDaLF. We test the performance of each model for sequence length $L = 300$ and plot the precision-recall curves with the corresponding results shown in Figure 9. All three models perform better in the open-world environment of Shadowsocks, indicating that the individual characteristics of Shadowsocks anonymous traffic data sites are more prominent and easier for model classification. The CNN-BiLSTM model performs significantly better than the TF model in both open-world environments. Furthermore, compared with the GANDaLF model in both open environments, each has its advantages and disadvantages.

The model's performance is appropriately optimized for precision or recall at $L = 200$, 300, and 400 (Table 5). When the model is tuned for optimum precision rate, SSH reaches the highest precision rate of 0.889 at a sequence length of $L = 400$ with the corresponding recall rate being 0.831. Shadowsocks reaches the highest precision rate of 0.912 at $L = 300$, with the recall rate being 0.899. Accordingly, when the model is optimized for the recall rate, both SSH and Shadowsocks reach the highest performance at $L = 300$, managing the highest recall rates of 0.934 and 0.963, respectively, while the corresponding precision rates are 0.742 and 0.789.

Figure 8 and Table 5 highlight that the CNN-BiLSTM model is still highly usable in the open-world scenario, and the attacker can tune the model in the open world utilizing the task target. If the goal is to identify the traffic of monitored websites in the network data, then the recall rate should be of more concern to the attacker, and the accuracy rate can be appropriately sacrificed to improve the recall rate. Furthermore, when the attacker's goal is to accurately monitor the websites' visitors, the accuracy rate is more critical, and the recall rate needs to be appropriately reduced.

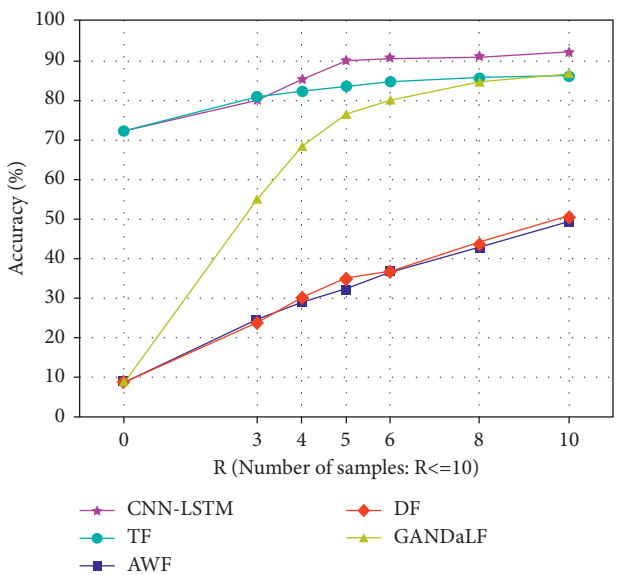

Figure 7: Accuracy curve with the number of "secondary training" samples.

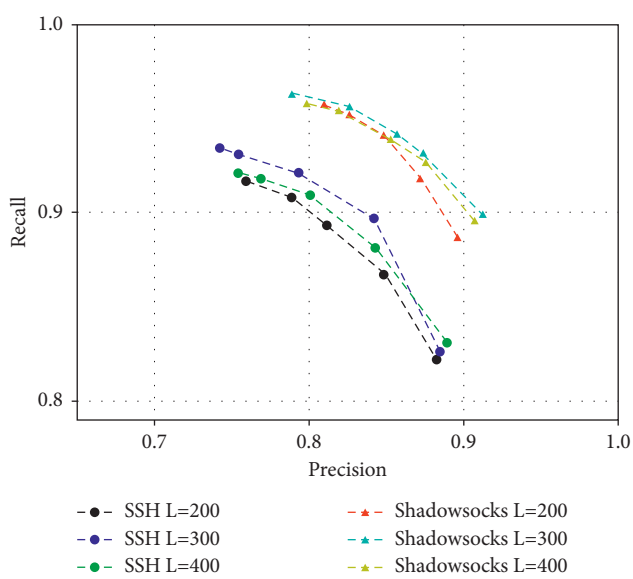

Figure 8: Precision-recall curves of the CNN-BiLSTM models in an open-world scenario.

6. Discussion

In this section, we discuss the possible limitations of this work and directions for future work.

6.1. Segmentation of Anonymized Web Data. In our experiments, we use previously collected representative datasets to ensure the purity of the data assuming that users open only one web page at a time during data collection. However, in a real-world attack scenario, users will open web pages accompanied by a lot of background traffic. Therefore, efficiently splitting the anonymous traffic from the background traffic is an important research topic.

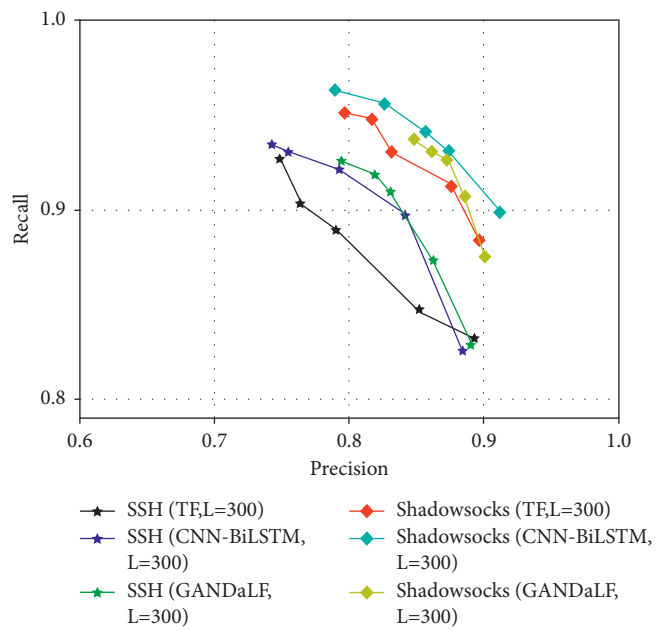

Figure 9: Precision-recall dynamic curves of CNN-BiLSTM, TF, GANDaLF models in the open world.

6.2. The Definition of Website Fingerprinting Attack. Our work is consistent with most current studies that only identify single-page website fingerprinting classification and do not include the hyperlinks and other subpages on the website homepage. The next step is to focus on how to characterize the overall fingerprint of the website.

6.3. Model Breakthroughs on Website Fingerprinting Defense Technology. This paper identifies and classifies the SSH and Shadowsocks single-agent anonymous encrypted traffic and employs the packet size, direction, and time interval as the essential features to achieve better attack results. To defend against website fingerprinting attack techniques that compromise user privacy, Tor, the currently best anonymous network communication system, was designed to transmit data in units in units of 512 bytes, called cells, and always pad all data transfers up to a cell boundary, with targeted defense against the important feature of packet size. Subsequent researchers have further defended against other features. Examples are the WTF-PAD based on adaptive padding proposed by Juarez et al. [28], Walkie-Talkie based on half-duplex communication and burst traffic proposed by Wang et al. [29] in 2017, Traffic Silver presented at USENIX Security 2020 proposed by Cadena et al. [30], zero-delay proposed by Gong and Wang et al. [31], and Mockingbird based on GAN techniques proposed by Rahman et al. [32]. These anonymity network defense techniques change the original direction, transmission time, and other characteristics of website traffic, blurring the differences between website traffic characteristics and increasing the difficulty for attackers to implement website fingerprinting attacks. Therefore, the model will have predictable degradation in attack effectiveness when applied to this more challenging anonymous network environment. A deeper analysis is

Table 5: Tuning results of CNN-BiLSTM model in the open world.

Environment	Length of sequence	Tuned for precision		Tuned for recall	
		Precision	Recall	Precision	Recall
SSH	$L = 200$	0.883	0.822	0.759	0.917
	$L = 300$	0.885	0.826	0.742	**0.934**
	$L = 400$	**0.889**	0.831	0.754	0.921
Shadowsocks	$L = 200$	0.896	0.887	0.809	0.957
	$L = 300$	**0.912**	0.899	0.789	**0.963**
	$L = 400$	0.907	0.896	0.798	0.958

needed on how to achieve a highly accurate small-sample website fingerprinting attack under such more complex conditions.

7. Conclusion

This paper proposes a website fingerprinting attack method based on homology analysis and designs a CNN-BiLSTM website fingerprinting attack model using a Siamese Network structure. Our architecture manages a high accuracy rate with only a small number of training samples per website. At the same time, we innovatively propose a data processing method to increase the data feature dimension and increase the fault tolerance of the sample's burst features.

We train our model with SSH anonymous network encrypted traffic and then exploit it to classify the Shadowsocks anonymous network encrypted traffic, managing over 90% accuracy with only five samples per site, which is significantly higher than current methods. Additionally, this experimental setup (training versus testing datasets are of different nature) highlights that the proposed model has a very appealing migration capability. Finally, the experimental results indicate that attackers can still achieve effective website fingerprinting attacks with fewer resources.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The National Key Research Projects supported this topic (nos. 2019QY1302 and 2019QY1305). The authors would like to express their gratitude to EditSprings (https://www.editsprings.com/) for the expert linguistic services provided.

References

[1] S. E. Oh, N. Mathews, M. S. Rahman, M. Wright, and N. Hopper, "GANDaLF: GAN for data-limited fingerprinting," *Proceedings on Privacy Enhancing Technologies*, vol. 2021, no. 2, pp. 305–322, 2021.

[2] M. S. Rahman, P. Sirinam, N. Mathews, K. G. Gangadhara, and M. Wright, "The utility of packet timing in website fingerprinting attacks," vol. 2020, no. 3, pp. 5–24, 2020, https://arxiv.org/abs/1902.06421.

[3] P. Sirinam, N. Mathews, M. S. Rahman, and M. Wright, "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*, pp. 1131–1148, London, United Kingdom, November 2019.

[4] S. Bhat, D. Lu, A. Kwon, and S. Devadas, "Var-CNN: a data-efficient website fingerprinting attack based on deep learning," *Proceedings on Privacy Enhancing Technologies*, vol. 2019, no. 4, pp. 292–310, 2019.

[5] P. Sirinam, M. Imani, M. Juarez, and M. Wright, "Deep fingerprinting: undermining website fingerprinting defenses with deep learning," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1928–1943, Toronto, Canada, October 2018.

[6] Z. Zhuo, Y. Zhang, Z. L. Zhang, X. Zhang, and J. Zhang, "Website fingerprinting attack on anonymity networks based on profile hidden Markov model," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1081–1095, 2017.

[7] V. Rimmer, D. Preuveneers, M. Juarez, T. V. Goethem, and W. Joosen, *Automated Website Fingerprinting through Deep Learning*, arXiv, preprint arXiv:1708.06376, 2017.

[8] J. Hayes and G. Danezis, "k-fingerprinting: A robust scalable website fingerprinting technique," in *Proceedings of the 25th {USENIX} Security Symposium ({USENIX} Security 16)*, pp. 1187–1203, Austin TX USA, August 2016.

[9] K. Abe and S. Goto, "Fingerprinting attack on Tor anonymity using deep learning," *Proceedings of the Asia-Pacific Advanced Network*, vol. 42, pp. 15–20, 2016.

[10] T. Wang and I. Goldberg, "Improved website fingerprinting on tor," in *Proceedings of the 12th ACM workshop on Workshop on privacy in the electronic society*, pp. 201–212, Berlin Germany, November 2013.

[11] J. Cheng, Y. Li, C. Huang, A. Yu, and T. Zhang, "ACER: detecting Shadowsocks server based on active probe technology," *Journal of Computer Virology and Hacking Techniques*, vol. 16, no. 3, pp. 217–227, 2020.

[12] B. Zantout and R. Haraty, "I2P data communication system," in *Proceedings of ICN*, pp. 401–409, ICN, Springer, Singapore, 2011.

[13] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: the second-generation onion router," in *Proceedings of the The 13th conference on USENIX Security Symposium*, CA, USA, August 2004.

[14] J. Boyan, "The anonymizer-protecting user privacy on the web," *Computer-Mediated Communication Magazine*, vol. 4, no. 9, 1997.

[15] A. Panchenko, F. Lanze, J. Pennekamp, T. Engel, A. Zinnen, and M. Henze, *Website fingerprinting at internet scale*, NDSS, NY, USA, 2016.

[16] A. Panchenko, L. Niessen, A. Zinnen, and T. Engel, "Website fingerprinting in onion routing based anonymization networks," in *Proceedings of the 10th annual ACM workshop on Privacy in the electronic society*, pp. 103–114, Chicago Illinois USA, October 2011.

[17] D. Herrmann, R. Wendolsky, and H. Federrath, "Website fingerprinting: attacking popular privacy enhancing technologies with the multinomial naïve-bayes classifier," in *Proceedings of the 2009 ACM workshop on Cloud computing security*, pp. 31–42, 2009.

[18] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *Proceedings of the ICML deep learning workshop*, Lille, France, July 2015.

[19] H. Cheng and R. Avnur, *Traffic Analysis of Ssl Encrypted Web Browsing*, University of Berkeley, CA, USA, 1998.

[20] C. Zhang, Y. Cai, G. Lin, and C. Shen, "DeepEMD: few-shot image classification with differentiable earth mover's distance and structured classifiers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Article ID 12203, Seattle, WA, USA, June 2020.

[21] W. Li, L. Wang, J. Xu, J. Huo, Y. Gao, and J. Luo, "Revisiting local descriptor based image-to-class measure for few-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7260–7268, Long Beach, CA, USA, June 2019.

[22] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Advances in Neural Information Processing Systems*, pp. 4077–4087, MIT Press, Cambridge, MA, USA, 2017.

[23] S. Li, J. Chen, and B. Liu, "Protein remote homology detection based on bidirectional long short-term memory," *BMC Bioinformatics*, vol. 18, no. 1, pp. 443–448, 2017.

[24] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *Proceedings of the International conference on machine learning*, pp. 448–456, Lille, France, July 2015.

[25] X. Zhang, Y. Zou, and W. Shi, "Dilated convolution neural network with LeakyReLU for environmental sound classification," in *Proceedings of the 2017 22nd International Conference on Digital Signal Processing (DSP)*, pp. 1–5, London, UK, August 2017.

[26] Q. Sun, D. R. Simon, Y. M. Wang, W. Russell, V. N. Padmanabhan, and L. Qiu, "Statistical identification of encrypted web browsing traffic," in *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, pp. 19–30, Berkeley, CA, USA, May 2002.

[27] L. Torrey and J. Shavlik, "Transfer learning," in *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*, pp. 242–264, IGI global, PA, USA, 2010.

[28] M. Juarez, M. Imani, M. Perry, C. Diaz, and M. Wright, "Toward an efficient website fingerprinting defense," in *Proceedings of the Computer Security - ESORICS 2016 European Symposium on Research in Computer Security*, pp. 27–46, Guildford, UK, September, 2016.

[29] T. Wang and I. Goldberg, "Walkie-talkie: an efficient defense against passive website fingerprinting attacks," in *Proceedings of the 26th {USENIX} Security Symposium ({USENIX} Security 17)*, pp. 1375–1390, Vancouver BC, Canada, August 2017.

[30] W. D. L. Cadena, A. Mitseva, J. Hiller, J. Pennekamp, S. Reuter, and J. Filter, "Trafficsliver: Fighting website fingerprinting attacks with traffic splitting," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1971–1985, Virtual Event USA, November 2020.

[31] J. Gong and T. Wang, "Zero-delay lightweight defenses against website fingerprinting," in *Proceedings of the 29th {USENIX} Security Symposium ({USENIX} Security 20)*, pp. 717–734, Boston, MA, USA, August 2020.

[32] M. S. Rahman, M. Imani, N. Mathews, and M. Wright, "Mockingbird: defending against deep-learning-based website fingerprinting attacks with adversarial traces," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 1594–1609, 2020.