Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2022.022495

check for updates

ARTICLE

Dark-Forest: Analysis on the Behavior of Dark Web Traffic via DeepForest and PSO Algorithm

Xin Tong[1], Changlin Zhang[2,*], Jingya Wang[1], Zhiyan Zhao[1] and Zhuoxian Liu[1]

[1]People's Public Security University of China, Beijing, 100038, China

[2]Henan Police College, Zhengzhou, 450046, China

*Corresponding Author: Changlin Zhang. Email: zcl@hnp.edu.cn

Received: 13 March 2022 Accepted: 26 May 2022

ABSTRACT

The dark web is a shadow area hidden in the depths of the Internet, which is difficult to access through common search engines. Because of its anonymity, the dark web has gradually become a hotbed for a variety of cyber-crimes. Although some research based on machine learning or deep learning has been shown to be effective in the task of analyzing dark web traffic in recent years, there are still pain points such as low accuracy, insufficient real-time performance, and limited application scenarios. Aiming at the difficulties faced by the existing automated dark web traffic analysis methods, a novel method named Dark-Forest to analyze the behavior of dark web traffic is proposed. In this method, firstly, particle swarm optimization algorithm is used to filter the redundant features of dark web traffic data, which can effectively shorten the training and inference time of the model to meet the real-time requirements of dark web detection task. Then, the selected features of traffic are analyzed and classified using the DeepForest model as a backbone classifier. The comparison experiment with the current mainstream methods shows that Dark-Forest takes into account the advantages of statistical machine learning and deep learning, and achieves an accuracy rate of 87.84%. This method not only outperforms baseline methods such as Random Forest, MLP, CNN, and the original DeepForest in both large-scale and small-scale dataset based learning tasks, but also can detect normal network traffic, tunnel network traffic and anonymous network traffic, which may close the gap between different network traffic analysis tasks. Thus, it has a wider application scenario and higher practical value.

KEYWORDS

Dark web; encrypted traffic; deep forest; particle swarm optimization

1 Introduction

In the 1960s, no one could have predicted that a small computer communication network for military use, called the ARPAnet [1], would be able to connect almost all the computers on the planet in the following decades and become the largest internet—Internet. By the end of 2021, there had been 4.95 billion Internet users worldwide, accounting for 62.5% of the world's population [2]. The Internet has become one of the most important mediums of daily communication and has a positive impact on people's production and life. However, the websites that we can access through search engines such as Google only occupy a small part of the Internet, and more information is hidden in the depths

of the Internet, and the dark web is a shadow area that cannot be illuminated by the sunlight. The dark web refers to a method that provides users with anonymous access to Internet information using technologies such as encrypted transmission, peer-to-peer networks, and relay obfuscation, and its most prominent feature is anonymity. Since everyone's identity is invisible on the dark web, there is a large amount of private data, violence, pornography, criminal information and terrorism hidden in it. Additionally, it has become a haven for information theft, cyber-attacks, and ransomware, bringing new challenges for cyberspace security and public safety. Therefore, the analysis of dark web traffic has become one of the important research fields in cyberspace security.

Unfortunately, the complexity of dark web traffic data poses challenges for automated analysis methods, specifically:

(1) Packets of dark web traffic are always encrypted, so it is almost impossible for us to obtain the keys to decrypt the packets, which makes it hard to match appropriate feature engineering for this task as well.

(2) The network data packets are mixed, so a captured packet cannot be reliably judged as normal traffic or dark web traffic, and which specific encryption method is used. A unified analysis framework is needed to handle different situations in real scenarios.

(3) The scale of network traffic in daily life is large, and the related detection tasks require high real-time performance, so the detection may need to be done on edge devices such as gateways. This brings difficulties to feature extraction and analysis using large-scale deep learning models that need hardware acceleration. Therefore, how to explore efficient and reliable methods for dark web traffic detection has become a pain point in this field at present.

To solve the above challenges, we propose a dark web behavior analysis model based on deep learning, called Dark-Forest. Specifically, the main contributions of this paper include:

(1) **A generic traffic behavior analysis method.** In this paper, a model called Dark-Forest is proposed, which is able to describe Internet access behavior according to network traffic, whether normal traffic or dark web traffic. This model can close the gap between different network traffic analysis tasks.

(2) **Broader application scenarios and finer-grained analysis.** Dark-Forest uses DeepForest [3] as the backbone classifier, which can accurately and automatically analyze the behavior of dark web traffic based on VPN and Tor technology without decrypting the packets. So it not only takes into account the advantages of traditional statistical machine learning methods in analyzing structured data, but also has the multi-layer representation ability similar to deep neural networks. Compared with the traditional detection models for a few types of encryption methods, Dark-Forest has a wider range of application scenarios.

(3) **Higher analysis efficiency.** Considering the real-time requirements in the real network environment, the feature selection mechanism based on the particle swarm optimization (PSO) algorithm [4] is introduced, which can eliminate redundant features to a greater extent, reduce model training costs, and speed up its inference, while ensuring that the accuracy of detection is not negatively affected as much as possible. It makes it easier for Dark-Forest to be deployed to edge devices such as gateway and firewalls to realize the real-time analysis of dark web traffic data.

Experiments based on the public DIDarknet dataset [5] show that the classification accuracy of Dark-Forest can reach 87.84%. And this method has advantages in precision, F1 score and other

metrics, and its performance is also better than the current mainstream machine learning and deep learning detection methods.

2 Related Work

2.1 Introduction to the Dark Web

The dark web is mainly based on anonymous networks and tunnel networks. Users can access it anonymously only through encryption tools such as VPN and Tor browser. On the one hand, The Onion Router (Tor) is currently the most widely used dark web traffic obfuscation technology. As shown in Fig. 1, each access traffic passing through the Tor network will be encrypted and obfuscated by three Tor nodes: the entry node, the relay node, and the exit node before transmission, which can effectively avoid the user's real network address and identity information from being traced. On the other hand, dark web users also use virtual private networks (VPN) technology to encrypt traffic to break through firewall restrictions and hide identity information. Therefore, how to analyze the function of dark web data traffic packets encrypted by multiple methods has become an important prerequisite for dark web intelligence analysis and cybercrime forensics. Due to the nonlinear and complex characteristics of various types of encryption methods, it is difficult to analyze dark web traffic by traditional rule-based or manual-based methods, so researchers have tried to use artificial intelligence models to mine and classify packets without decrypting them, and have made some progress. Related research can be summarized as statistical machine learning based detection methods and deep learning based detection methods.

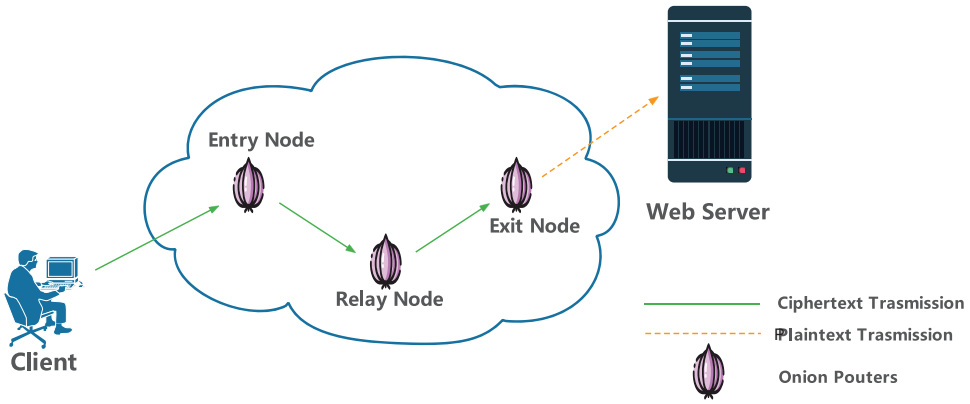

Figure 1: Encryption process during dark web traffic transmission

2.2 Machine Learning Based Detection Methods

Initially, the IP addresses of the Tor directory server and relay nodes are accessible, and analysts can directly build rule bases according to IP address and other identifiers to detect and block Tor anonymous traffic. However, with the emergence of obfuscation technologies such as Bridge and Meek, these rule-based filtering methods are no longer effective. Although some complex rule bases [6] are still helpful for classifying encrypted traffic, they may lead to increased maintenance costs. Additionally, the flexibility of these methods is also slightly insufficient, making it difficult to apply to real-world detection tasks. Consequently, some researchers try to use various machine learning models to build automated encrypted traffic analysis methods.

For detection methods based on statistical machine learning, it mainly includes two steps: feature engineering and classification, which are often used to analyze packet header information or spatio-temporal features of the encrypted traffic. The representative works on feature engineering mainly include: Islam et al. [7] tried to propose an effective feature engineering for VoIP traffic based on VPN encryption. They selected a set of flow spatio-temporal features (FSTF), which proved to be able to effectively improve the performance of six classic machine learning models, including KNN, Bagging, and Boosting in this classification task. To extract the features of Tor traffic, Xu et al. [8] proposed a feature engineering based on sliding windows, which can convert Tor traffic into 12 kinds of features. Then machine learning models such as XGboost and random forest are used as downstream classifiers, and the final precision and recall rate both can reach 99%.

The work on classifiers mainly includes: Zhioua [9] proposed a Tor traffic analysis model based on the hidden markov model (HMM), which can detect the local network traffic between the target Tor client and the first Tor relay node. The experimental analysis shows that the proposed HMM-based approach has a high precision (93% on average) and F1 Score (75% on average). This method can also be applied to attack the privacy of the Tor network [10]. Afuwape et al. [11] constructed an ensemble learning classifier using a random forest and gradient boosting machine, which achieved 93.80% classification accuracy in the task of distinguishing VPN traffic from Non-VPN traffic, outperforming single classifiers such as KNN, multilayer perceptron, and decision trees. Subsequent research [12] added a decision tree model based on this work, which improved the classification accuracy of VPN traffic by 0.6%. In addition, different from the above research based on classification models, Rao et al. [13] proposed a gravitational clustering algorithm for Tor network traffic analysis, which can use gravitational force and similarity samples are clustered. Experiments show that the average accuracy and F1 Score of the method both exceed 80%, while the accuracy of the K-means algorithm trained on the same dataset only reaches 50%.

The above shallow models have the advantages of the training process and resource consumption, and are very easy to implement in engineering. Researchers can train an effective model in a few minutes even by using a low-power CPU and limited memory. However, these models are still difficult to apply to real scenes. Firstly, these methods rely on complex manual feature engineering, but the encrypted traffic data is very different from the normal data, with strong nonlinear features and weak interpretability, which makes it difficult for researchers to rely on intuition and experience to judge the effectiveness of features. Secondly, the limited number of parameters of these models leads to the lack of representation ability, which makes the models converge faster on small-scale datasets, but there may be a learning bottleneck problem in the tasks based on large-scale datasets. Finally, most shallow models do not have the ability to analyze the correlation between features, which further restricts their fitting ability.

2.3 Deep Learning Based Detection Methods

Some researchers try to build an end-to-end dark web encrypted traffic recognition model based on deep learning to directly extract information from the raw traffic data and classify it. The works [14,15] proposed convolutional neural networks (CNN) based classification models for encrypted traffic. In these experiments, the raw traffic data is directly converted into a grayscale map, where each group of bytes corresponds to one pixel, and then one-dimensional or two-dimensional convolutional layers are used to learn specific patterns in the grayscale map of the traffic. Finally, the full connection layers are used to classify the features extracted from the convolution layers. The experimental results of these methods are shown to be superior to the mainstream shallow classifiers. Sequential models such as long short-term memory (LSTM) [16] networks and gated recurrent units (GRU) [17] are able

to extract the sequential information from encrypted traffic data, avoiding the problem of feature loss that exists in convolutional neural network-based approaches. Thus, the results are further improved. Lu et al. [18] tried to combine CNN and LSTM, and proposed a model named ICLSTM (Inception-LSTM) for the classification of encrypted traffic. In their study, the raw encrypted traffic data was first converted into a grayscale image. The grayscale image is then sent to the convolutional block-based Inception network and LSTM network for classification. The accuracy rate reached 98% in the test on the ISCX-VPN-2016 [19] dataset. To analyze VoIP traffic in tunneled and anonymous networks, the captured raw traffic is preprocessed based on the feature engineering of FSTFs [20], and then a hybrid deep learning model (MLP, 1D-CNN, and LSTM) is used for classification, which can classify traffic data into four categories: VPN VoIP, VPN Non-VoIP, TOR VoIP, and TOR Non-VoIP, and the classification accuracy can exceed 94%. In addition, experiments in work [21] show that the attention mechanism [22] can help the deep learning model focus on the more critical information in encrypted traffic and ignore the noise that is meaningless to the detection task, which can further improve the detection effect of the model. Based on this discovery, Yao et al. [23] proposed a model based on the combination of attention mechanism and LSTM, which can effectively detect VPN traffic, and achieved a classification accuracy of 91.2% on the ISCX-VPN-2016 dataset, and this rate is improved by 1.4% compared to the model without attention mechanism.

In addition to supervised learning methods, some semi-supervised and self-supervised deep learning methods have also been shown to have good results in encrypted traffic detection tasks, and provide inspiration for the analysis of dark web traffic. Guo et al. [24] proposed an encrypted traffic classification model based on an unsupervised training method. They preprocessed traffic samples into session images, and then used convolutional autoencoders (CAE) [25] for feature dimensionality reduction and classification. In the binary classification task for Tor traffic, the accuracy rate can reach 98%. Iliyasu et al. [26] proposed a VPN traffic detection method based on Deep Convolutional Generative Adversarial Network (DCGAN) [27], through alternating training between the generator and the discriminator, to achieve the detection method that relies on a small number of labeled samples. The function of generating new traffic samples can effectively improve the accuracy of downstream classifiers.

Although various encrypted traffic analysis methods provide references for the detection of dark web traffic, there are still some pain points in current research. On the one hand, the existing detection technologies can only analyze a few types of dark web traffic. For example, only one of the tunnel network traffic or anonymous network traffic can be detected, and cannot handle traffic with multiple encryption technologies coexisting in real scenarios. On the other hand, the prior methods have high recognition accuracy in simple coarse-grained detection tasks (such as the binary classification task of encrypted traffic), and the accuracy in more fine-grained multi-classification tasks still needs to be improved. In addition, these end-to-end models usually deal with the raw data directly, and the high-dimensional features produced by this process further increase the hardware cost, resulting in the limited application scenarios of these methods, so these models are difficult to be deployed in low computing devices on the edge of the Internet. Therefore, how to improve the detection speed and accessibility to meet the real-time requirements is also an urgent problem to be considered in the task of dark web traffic analysis.

3 Model

In order to solve the pain points existing in the field of dark web traffic analysis, inspired by deep learning methods based on tree models for normal traffic detection tasks [28], this paper proposes an

analysis method for the behavior of dark web traffic, called Dark-Forest. The framework of the method is shown in Fig. 2, which is mainly composed of a PSO-based feature selector and a backbone classifier based on the DeepForest model. Among them, the PSO algorithm is mainly used to solve the minimum set of features that keep the model effect without degradation in order to achieve data dimensionality reduction and meet the real-time requirement of this task; while DeepForest is continuously trained and evaluated based on the current feature subset, and finally the optimal feature sequence is selected and fitted by using the two together.

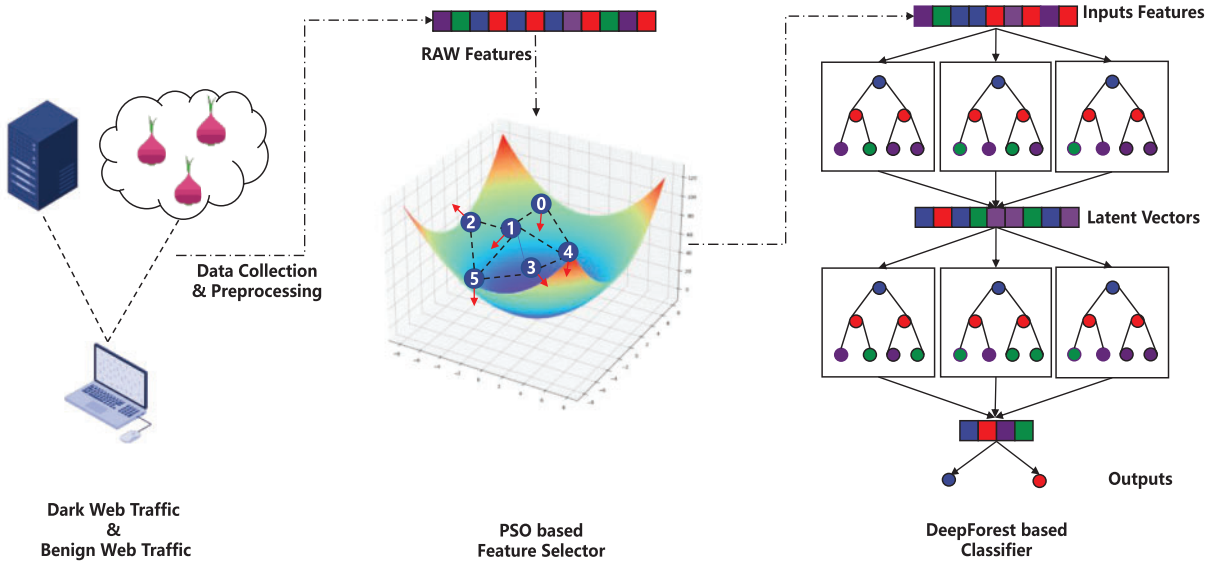

Figure 2: The overall framework of the Dark-Forest model

3.1 Feature Selection Algorithm Based on PSO

To eliminate redundant features in the raw data, reduce training costs, and improve the real-time performance of the model, a feature selection mechanism based on the PSO algorithm is introduced. PSO is a heuristic optimization algorithm that mimics the activity of bird clusters and is widely used in optimization problems for machine learning tasks [29,30]. The algorithm assumes that there are some randomly distributed particles in the problem search space and each particle has two attributes of position and velocity. In the optimization process, these attributes of the particles are continuously updated through iterations under the control of fitness evaluation metrics to determine the optimal point (pbest) of each particle. Finally, the global optimal position (gbest) of the whole particle swarm population is determined based on the pbest of all particles to obtain the optimal solution to the objective problem.

The overall process of feature selection using the PSO algorithm is illustrated in Fig. 3, which mainly includes the following steps:

(1) Particle initialization: According to the boundary of the problem search space, randomly initialize the position and velocity attributes of the particles.

(2) Particle swarm evaluation: In order to evaluate particle swarm optimization, first, a mask vector for features will be created. If the feature value is less than a certain threshold, the corresponding element in the mask vector is converted to 0; otherwise, it is converted to 1. After this process, irrelevant features can be eliminated. Then, the remaining features are evaluated.

In the experiments, in order to be able to filter as many redundant features as possible while taking into account the accuracy of the model, an evaluation metric function as shown in Eq. (1) was used, where a smaller score of fitness represents a better subset of features selected. Where *acc* is the accuracy of the model on the validation set of the non-redundant feature subset, $N_{selected}$ is the number of retained features, and N_{total} is the number of raw features. α is the hyperparameter used to balance the two types of evaluation metrics. A larger α means that PSO tends to eliminate fewer features that can enhance the effectiveness of the model, and a smaller α means that the main search objective of PSO is to filter out more features.

$$fitness = \alpha \times (1 - acc) + (1 - \alpha) \times \frac{N_{selected}}{N_{total}} \tag{1}$$

(3) Particle update: The velocity information and position information of the particle are updated according to the current fitness score. When updating the velocity in the iteration, three main influencing factors are considered: its current initial velocity (velocity after the last iteration), the velocity of its own historical best point, and the velocity of the global historical best point. The updated particle velocity can be obtained by weighted summation of the three, as shown in Eq. (2), where *w*, *c1*, and *c2* are weight hyperparameters, called inertia weights of velocity and learning factors, respectively.

$$v_i^{t+1} = w \times v_i^t + c_1 \times random(0, 1) \times (pbest - x_i^t) + c_2 \times random(0, 1) \times (gbest - x_i^t) \tag{2}$$

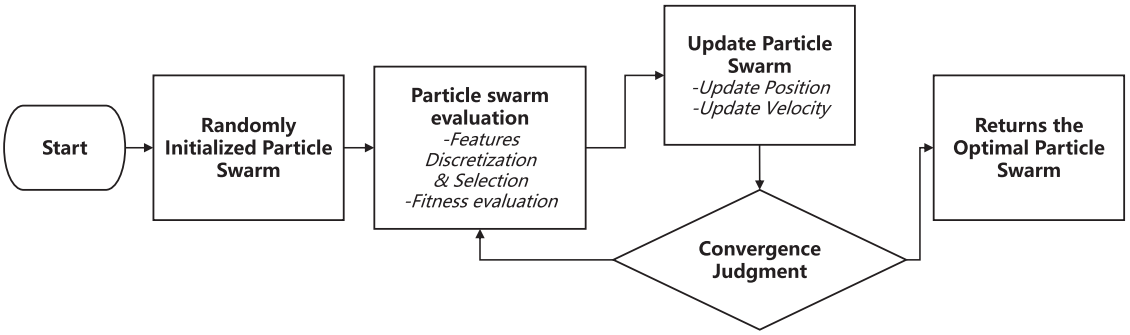

Figure 3: Feature selection process based on PSO algorithm

When updating the position information, the particles are assumed to be in a uniform linear motion, as shown in Eq. (3), where x_i^{t+1} is the current initial position, and v_i^{t+1} is the particle velocity after the previous step update.

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{3}$$

3.2 DeepForest Model

DeepForest is a deep learning method based on tree models. This learner integrates and connects the forests formed by trees to achieve the purpose of representation learning, thereby improving the classification performance. The model mainly includes two core algorithms of multi-grained scanning and cascade forest, and its overall structure is shown in Fig. 4.

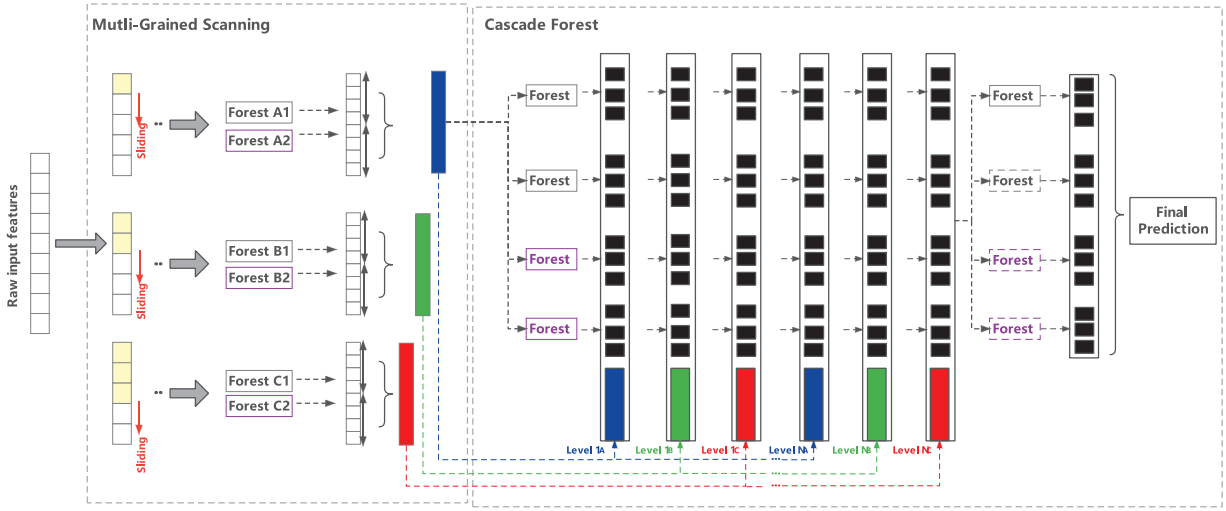

Figure 4: The overall structure of the DeepForest model

3.2.1 Multi-Grained Scanning

The powerful representational capabilities of deep neural networks are primarily due to their ability to correlate and reorganize features. Inspired by this idea, a mechanism called "multi-grained scanning" is used in DeepForest to preprocess the input features, which can help the model mine the correlations between the different features of the samples. On the one hand, this mechanism is similar to the sliding convolution kernel used by CNN, and features are scanned through sliding windows of multiple scales to achieve feature reuse. On the other hand, if the input feature dimension is too high, the multi-grained scanning also provides a downsampling function similar to the pooling operation to achieve data dimensionality reduction. The process of multi-grained scanning is shown in Fig. 5.

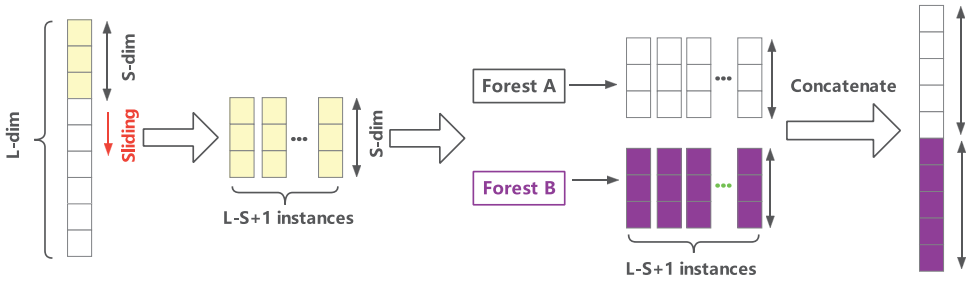

Figure 5: The process of multi-grained scanning

Specifically, for an input feature vector of dimension L, it is first scanned and sampled using a sliding window of length S to generate a subset of the feature vector, which contains $L - S + 1$ feature vectors of dimension S. These sub-features are then input to a normal random forest and a completely random forest and trained. These forest models output feature vectors with a dimension equal to the size of the sliding window. Finally, the output results of all forest models are concatenated together to obtain the final feature vector after multi-grained scanning. This process only does the scanning of features and does not involve the learning of model parameters, so it is faster compared to the convolution operation of CNN.

3.2.2 Cascade Forest

Hierarchical representation learning is another major advantage of deep neural networks, that is, through the design of stacked network layers, the complex mapping relationship from the raw data space to the target task space can be disassembled into a step-by-step nonlinear mapping transformation learning. Through the cooperation of hierarchical feature abstraction and transformation, the representation learning of different scales and levels can be realized.

From the perspective of the structure of the model, similar to the deep neural networks, a stacked model structure called cascade forest is also adopted in the DeepForest, which realizes layer-by-layer representation learning by integrating and concatenating forest models composed of trees. Deep neural networks use neurons as the basic unit of each layer. In the cascade forest, the composition basis of each layer is a random forest, and these random forests themselves are the result of the ensemble of multiple decision tree models, so the cascade forest is an "ensemble of the ensemble". At the same time, in order to improve the learning ability of model differentiation and diversity, cascade forest is similar to multi-grained scanning, and two different types of random forest substructures are also introduced, namely complete random forest and normal random forest.

As for the training process, the data stream inside the cascade forest is also from front to back. The previous layer applies a nonlinear transformation to the input data, and then takes the output result as the input of the back layer. Each forest unit in the cascade forest averages the classification probabilities of all its leaf nodes and uses them as the final output classification results. Also, in order to avoid information loss during the forward propagation, the cascade forest additionally introduces a shortcut connection mechanism. For each forest layer, its input is a mixed feature vector formed by concatenating the output result vector and the input data vector of the previous layer. In addition, a variety of training mechanisms to reduce the risk of overfitting have also been introduced into the cascade forest. On the one hand, the output vectors generated by each forest are generated by k-fold cross-validation; on the other hand, the model uses a combination of early stop control based on validation gain and hyperparameter control to constrain the depth of the entire cascade forest to avoid the model from being too deep. The algorithm description of the cascade forest is shown in Algorithm 1.

Algorithm 1: Cascade Forest Algorithm

Inputs: feature vectors x, number of layers of cascade forest *num_layers*.
Functions: multi-grained scanning processing function **MGS**, cascade forest layer processing function **Layers**, vector concatenate function **Concat**, Average Calculation Function **Avg**, Function to get the index corresponding to the maximum value **ArgMax**, Sequence Generation Function **Range**.
Outputs: predicted label y.

```
# Multi-grained scanning of raw dark web traffic data
x = MGS(x)
#Traverse each layer of the cascade forest
for idx in Range (num_layers):
        #result is the processed result of each layer
        result = Layers[idx](x)
        #Update the input variable, concatenating the output of the previous layer and
        the output of this layer to obtain a mixed vector
        x = Concat(x, result)
```

(Continued)

Algorithm 1: (Continued)

9. #Average the output of each forest unit in the last layer
10. $score = \textbf{Avg}(result)$
11. # Select the label corresponding to the maximum value in score as the prediction result
12. $y = \textbf{ArgMax}(score)$

4 Experiment and Analysis

4.1 Datasets and Evaluation Metrics

To verify the effectiveness of the Dark-Forest model in the dark web traffic detection task, this paper uses the DIDarknet dataset provided by Habibi et al. [5] to train and test the model. DIDarknet is a structured dataset based on the flow spatio-temporal features of dark web traffic data, and the feature in this dataset can be extracted quickly without decrypting the data packet by using tools such as CICFlowMeter. From the perspective of encryption methods, the dataset mainly includes three types of data: normal traffic, Tor-based encrypted traffic, and VPN-based encrypted traffic; in terms of application types, the dataset covers encrypted and non-encrypted traffic of eight common types of applications, including VoIP traffic, Video-Stream traffic, File-Transfer traffic, Chat traffic, Audio-Stream traffic, email traffic, Browsing traffic and P2P traffic. The distribution of this dataset is shown in Fig. 6. In the experiment, we prepared two types of datasets, a large-scale dataset, using 116481 samples as a training set, 5000 samples as a validation set, and 20,000 samples as a test set, to examine the model in large-scale performance under labeled samples. The other small-scale dataset contains only 1000 training samples to examine the performance of Dark-Forest when training data is scarce. The environment configuration of the server used in the experiment and the hyperparameter settings of the Dark-Forest model are shown in Tables 1 and 2.

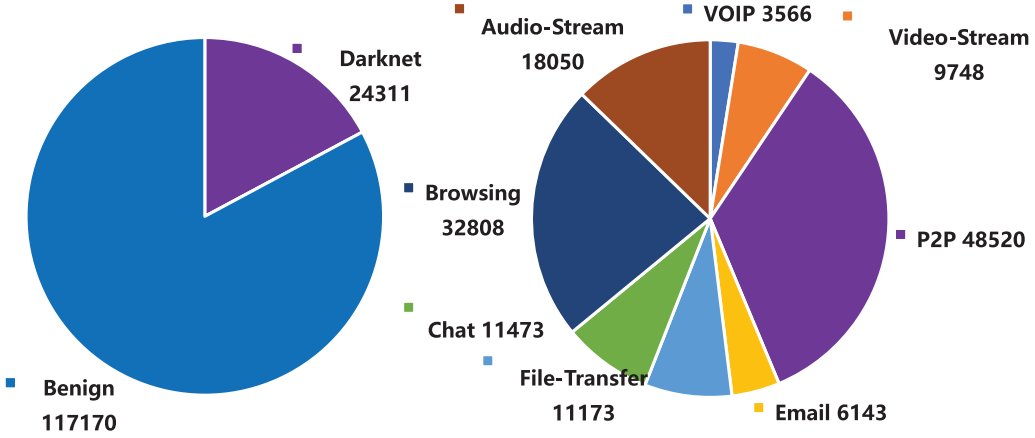

Figure 6: The composition of the DIDarknet dataset. **Left:** the composition of all types of traffic. **Right:** the composition of dark web traffic

Table 1: Experimental environment information

Device & Software	Information
CPU	Intel(R) Xeon(R) CPU E5-2690 v4
RAM	12 GB
External storage	512 GB SSD
Operating system	Ubuntu 18.04.3 LTS
Python version	3.8.8 (AMD64)
Machine learning library	numpy 1.19.5; pandas 1.2.4

Table 2: Hyperparameter settings

Part	Parameter name	Values
DeepForest	Criterion	gini
	max_depth	10
	Delta for early stopping	$1e-5$
	n_trees	100
PSO	Number of particles	5
	Max iteration	15
	w	0.9
	α	0.95
	c1	2
	c2	2

The main task objective of the experiment is to examine whether the model can distinguish the corresponding access behavior types of each sample in the dataset where both unencrypted traffic and two types of encrypted traffic are present. Therefore, Accuracy, Precision, Recall, and F1 Score are selected as the evaluation metrics, to achieve the purpose of measuring the recognition effect, leakage rate, and false alarm rate of the model in all aspects. The principles are shown in Eqs. (4)–(7). The TP and TN represent the sample numbers of true cases and true negative cases, and the FP and FN represent the sample numbers of false positive samples and false negative samples.

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + FP + TN} \tag{4}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{5}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{6}$$

$$F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{7}$$

4.2 Comparative Experiment

In the experiment, several representative statistical machine learning models, such as Logistic Regression (LR) model, Naive Bayesian (NB) model, Decision Tree (DT) model, Random Forest (RF) model, the Support Vector Machine (SVM) model based on the linear kernel function and the Gaussian kernel function, and the K-Nearest Neighbor (KNN) algorithm, were selected as baseline models. At the same time, the MultiLayer Perceptron (MLP) model and MLP with attention mechanism (MLP-Attention) are selected as the representative of the deep neural network model. In addition, to enhance the convincingness of the results, we also cite the experimental results of two deep learning-based methods, 1D CNN and Deep Image, on DIDarknet by Habibi et al. [5], and compare with them. The experimental results based on the large-scale dataset and small-scale dataset are shown in Tables 3 and 4, respectively.

Table 3: Experimental results based on the large-scale dataset

Methods	Models	Accuracy	Precision	Recall	F1 score
Statistical machine learning	LR	62.24	72.37	62.24	66.13
	NB	44.89	61.47	44.89	51.28
	KNN	81.81	82.73	81.81	82.17
	DT	81.93	85.56	81.93	83.09
	RF	84.36	87.07	84.36	85.10
	Linear-SVM	64.65	73.98	64.65	68.04
	RBF-SVM	69.25	76.88	69.25	71.92
Deep learning (Neural network)	MLP	80.58	84.80	80.58	81.98
	MLP-Attention	78.36	82.98	78.36	79.95
	1D CNN [5]	73.00	74.00	73.00	73.00
	Deep image [5]	86.00	86.00	86.00	86.00
Deep learning (Tree model)	DeepForest	87.51	87.78	87.51	87.59
	Dark-Forest (ours)	**87.84**	**88.34**	**87.84**	**88.02**

Table 4: Experimental results based on the small-scale dataset

Methods	Models	Accuracy	Precision	Recall	F1 score
Statistical machine learning	LR	55.53	68.32	55.53	60.83
	NB	41.03	62.39	41.03	48.21
	KNN	71.84	74.94	71.84	72.94
	DT	72.58	74.17	72.58	73.18
	RF	74.80	78.32	74.80	76.17
	Linear-SVM	57.78	72.35	57.78	63.42
	RBF-SVM	57.43	72.97	57.43	63.96

(Continued)

Table 4 (continued)

Methods	Models	Accuracy	Precision	Recall	F1 score
Deep learning	MLP	66.25	73.72	66.25	69.27
(Neural network)	MLP-Attention	60.28	76.96	60.28	67.14
Deep learning	DeepForest	75.46	78.22	75.46	76.60
(Tree model)	Dark-Forest (ours)	**77.06**	**79.90**	**77.06**	**78.20**

Analysis of the experiment shows that after the feature selection of the PSO algorithm, the number of input features of the DeepForest model is reduced from the original 75 to 32, the reduction rate reaches 57.3%, and the training time of the model is also shortened by 26.3%. At the same time, feature screening not only did not have a negative impact on the detection effect, but further improved the accuracy of the model. In the large-scale dataset-based experiments, Dark-Forest achieves 87.84% accuracy and 88.02% F1 score, which are 0.33% and 0.43% better than the original DeepForest model, and achieves state of the art in all evaluation metrics. In the learning task with the training set containing only 1000 labeled data, DeepForest also achieves an impressive accuracy and F1 Score of 77.06% and 78.2%, respectively, significantly outperforming baseline models such as DeepForest and Random Forest. Also, comparing the performance of Dark-Forest and MLP models reveals that tree-based deep learning methods are more easily trained on the small-scale dataset.

At the same time, it is found in the experiment that the LR model and NB method depend on the normalization preprocessing of data, and directly processing the raw data may lead to the non-convergence of the model. Similarly, in a large number of experiments such as image classification and text classification based on deep learning, it is also proved that deep neural network also highly depends on the normalization process, otherwise it will have a negative impact on the convergence of gradient descent algorithm. The two classical tree models, decision tree and random forest, have low requirements for data preprocessing, and can get better recognition performance without normalizing the raw data. The tree-based deep learning model DeepForest also inherits this feature well. Experiments show that the training results of DeepForest and Dark-Forest are consistent regardless of whether the data is normalized or not. This advantage makes it unnecessary for additional data preprocessing when applied to dark web traffic detection in real scenes, and can further reduce the computational overhead, and meet the real-time requirements.

To examine the classification robustness of Dark-Forest, we also visualized the micro-average ROC curves and micro-average PR curves of the five better-performing models (Dark-Forest, Deep-Forest, RF, KNN, and MLP), respectively, and shown in Figs. 7 and 8. It can be seen that the AUC scores and micro-AP scores of the Dark-Forest model are higher than those of the baseline model, which proves that the model has better robustness and is advantageous in handling the task of analyzing dark web traffic with unbalanced sample distribution in real scenarios.

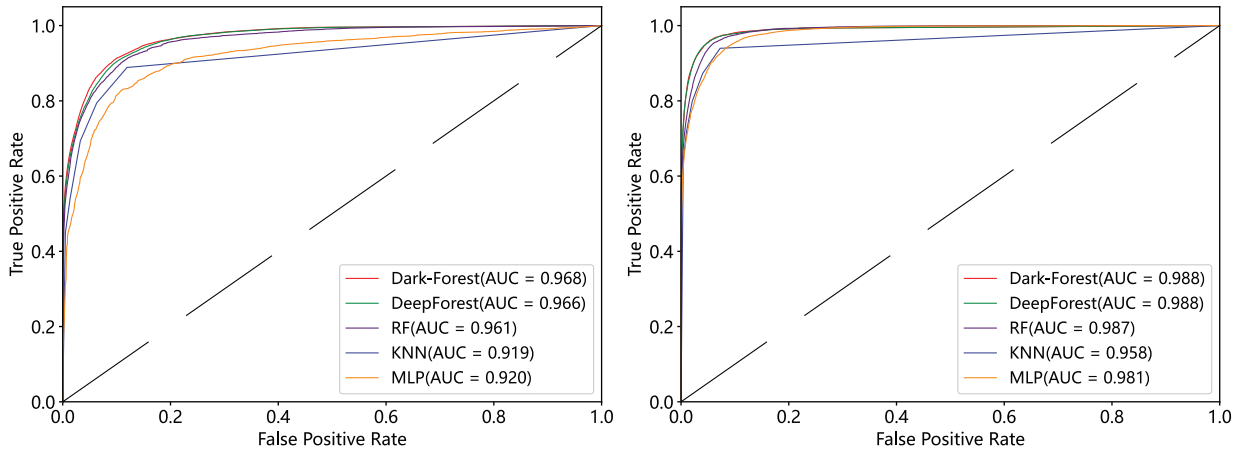

Figure 7: Micro-average ROC curves of 5 methods. **Left:** ROC curve of experiment based on the small-scale dataset. **Right:** ROC curve of experiment based on the large-scale dataset

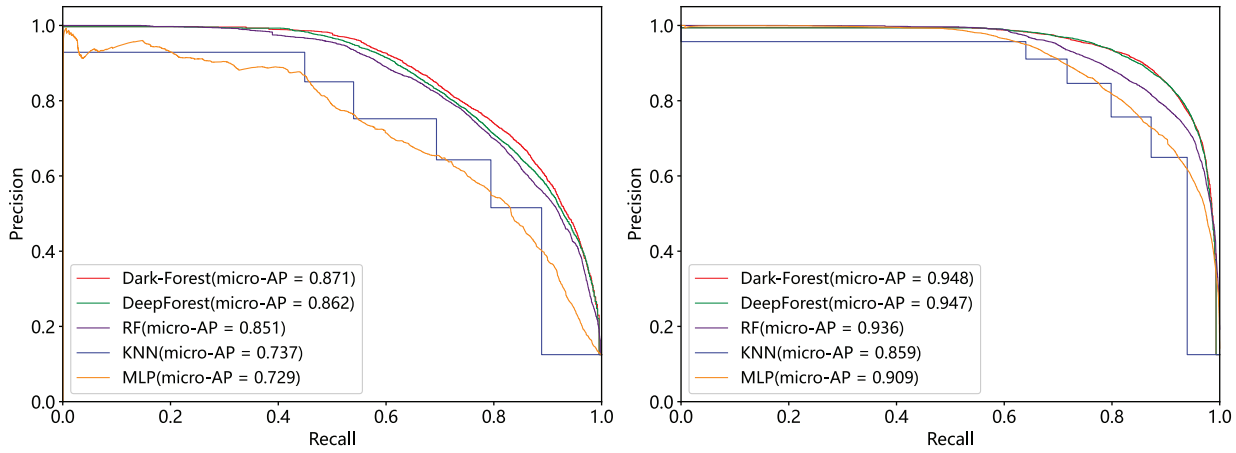

Figure 8: Micro-average PR curve of 5 methods. **Left:** PR curve of experiment based on the small-scale dataset. **Right:** PR curve of experiment based on the large-scale dataset

In order to further analyze the detection ability of the Dark-Forest model for different types of dark web traffic data, we also visualized the standardized confusion matrix of Dark-Forest, as shown in Fig. 9. It can be seen that the model has a better performance on the five types of traffic detection: Audio-Streaming, Browsing, Chat, File-Transfer, and P2P. However, due to the unbalanced distribution of data samples, the recognition accuracy of the model for Email and VoIP traffic still needs to be improved, and these two types of traffic are more likely to be mistakenly identified as Chat-type traffic by Dark-Forest.

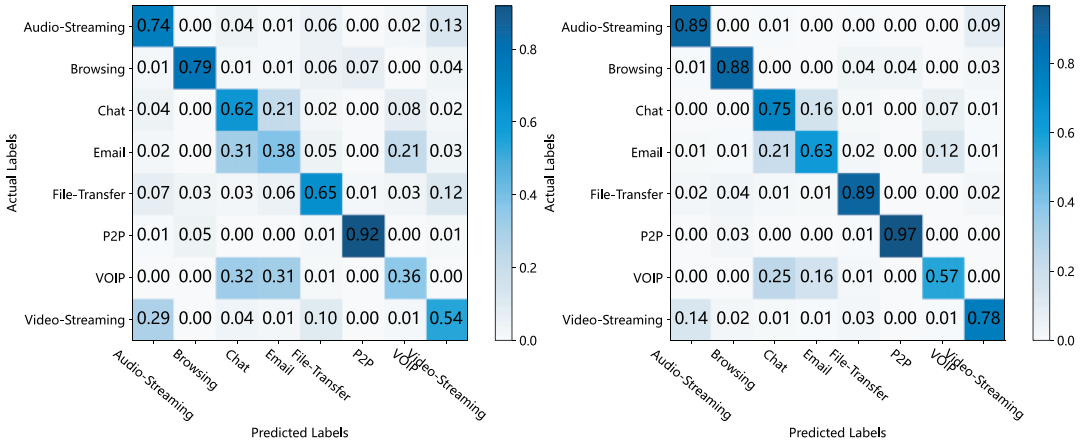

Figure 9: The standardized confusion matrix of Dark-Forest. **Left:** results based on the small-scale dataset. **Right:** results based on the large-scale dataset

In addition, we also compare Dark-Forest with several better-performing models in more detail and examine the detection effect of these models on each type of sample. In the large-scale dataset, we compare the performance of Dark-Forest and Deep Image, random forest, and the original DeepForest model, as shown in Figs. 10 and 11. In the small-scale dataset based training task, KNN was selected as an alternative model because the relevant results of the Deep Image model could not be obtained. It can be seen that the five models have their own advantages for the detection of different types of dark web traffic samples, and Deep Image is better at detecting Chat and Video-Streaming types of traffic. Dark-Forest and DeepForest detect Browsing and P2P-type traffic better than Deep Image. On the other hand, KNN has higher precision in the analysis of Email traffic whose identification precision is low in Dark-Forest. It can also be found that after the feature selection by the PSO algorithm, Dark-Forest is more effective than the original DeepForest model when dealing with the small-scale dataset based learning task.

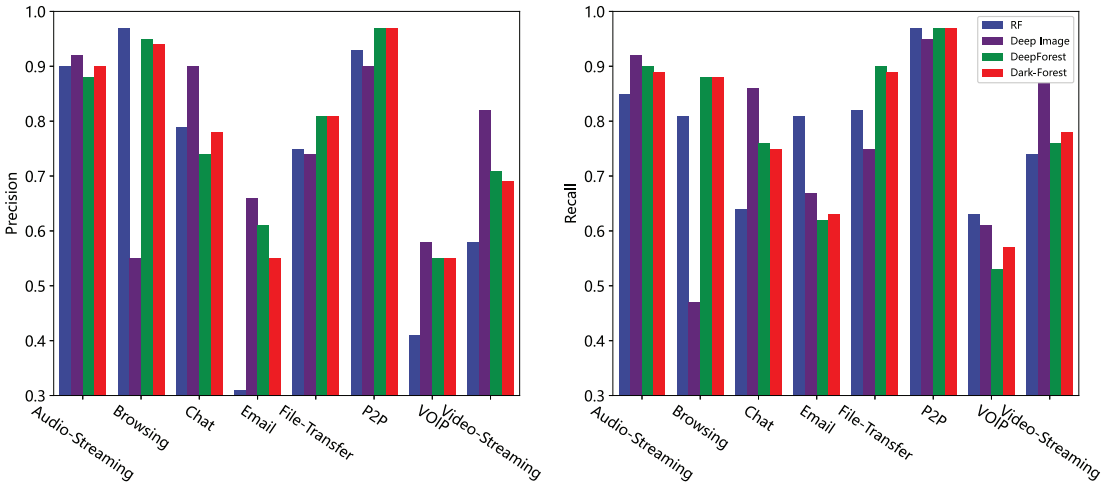

Figure 10: Comparison of the effects of 4 models trained on the large-scale dataset for detecting different types of traffic. **Left:** precision *vs.* traffic types. **Right:** recall *vs.* traffic types

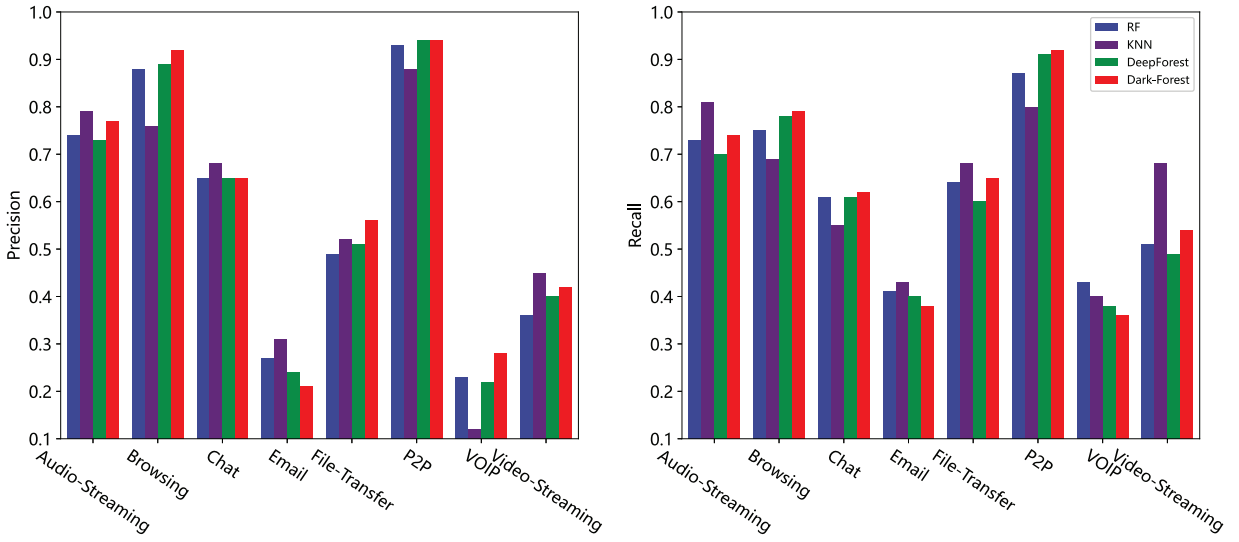

Figure 11: Comparison of the effects of 4 models based on the small-scale dataset for detecting different types of traffic. **Left:** precision *vs.* traffic types. **Right:** recall *vs.* traffic types

Considering that in real scenarios, it is possible to continuously collect and label dark web traffic data. Therefore, we also studied the changes in the effects of several models as the size of the training set continued to expand, to examine the potential of the model for long-term application in the real environment. The experimental results are shown in Fig. 12. It can be seen that increasing the number of training samples has a positive impact on the performance of the five models. Among them, the performances of deep learning-based models (Dark-Forest, DeepForest, and MLP) have been improved more significantly, and the value of the large-scale datasets can be more fully utilized, while the random forest model has prematurely entered a performance saturation period.

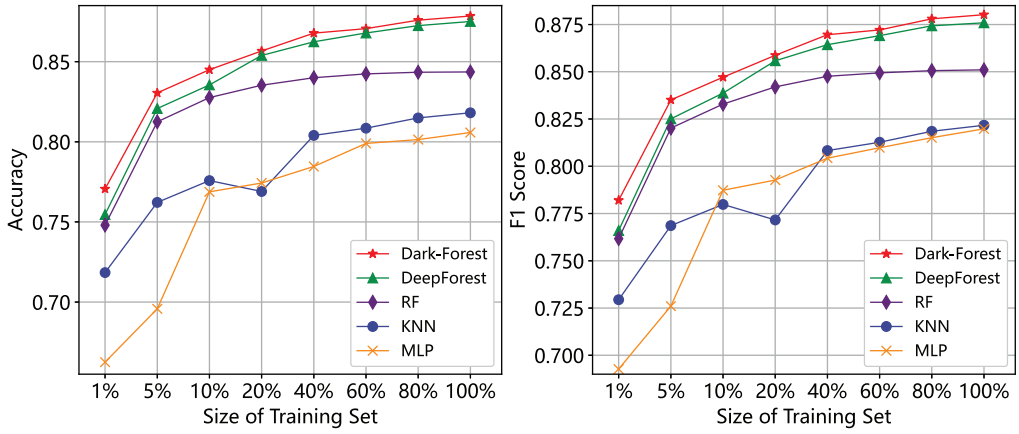

Figure 12: Performance changes of 5 models with the dataset size. **Left:** accuracy *vs.* dataset size. **Right:** F1 score *vs.* dataset size

4.3 Ablation Experiment

To verify the effectiveness and advantages of the PSO feature selection algorithm, additional ablation experiments are carried out in this paper. On the one hand, the original DeepForest model was set as the baseline method, and we compare the PSO-based feature selection algorithm with five common feature dimensionality reduction or filtering methods: mutual information-based feature selection (Mutual Info), chi-square-based feature selection (Chi2), principal component analysis (PCA) [31], autoencoder and genetic algorithms (GA) [32]. The experimental results on the large-scale dataset and the small-scale dataset are shown in Tables 5 and 6. It can be found that the performance of DeepForest has been further improved after feature selection by the chi-square algorithm, the PSO algorithm, and the GA algorithm, and the optimization based on the PSO algorithm has the most obvious improvement effect. However, the PCA algorithm, autoencoder and feature selection based on mutual information have a certain negative impact on the accuracy of the model. At the same time, it is also found in the experiment that the search time of the PSO algorithm is lower than that of the GA feature selection, and the process of convergence is more stable.

Table 5: Ablation experiments based on the large-scale dataset

Models	Features	Accuracy	Precision	Recall	F1 score
Baseline	75	87.51	87.78	87.51	87.59
+Mutual Info	**32**	2.54↓	2.30↓	2.54↓	2.45↓
+Chi2	**32**	0↑	0.20↑	0↑	0.09↑
+PCA	**32**	2.30↓	2.10↓	2.30↓	2.21↓
+Autoencoder	**32**	3.18↓	2.81↓	3.18↓	3.01↓
+GA	34	0.20↑	0.32↑	0.20↑	0.25↑
+PSO(ours)	**32**	**0.33**↑	**0.56**↑	**0.33**↑	**0.43**↑

Table 6: Ablation experiments based on the small-scale dataset

Models	Features	Accuracy	Precision	Recall	F1 score
Baseline	75	75.46	78.22	75.46	76.60
+Mutual Info	**32**	0.45↓	0.13↓	0.45↓	0.30↓
+Chi2	**32**	0.95↑	1.13↑	0.95↑	1.02↑
+PCA	**32**	2.99↓	1.14↓	2.99↓	2.34↓
+Autoencoder	**32**	6.08↓	3.26↓	6.08↓	5.07↓
+GA	34	0.68↑	1.31↑	0.68↑	0.92↑
+PSO(ours)	**32**	**1.60**↑	**1.68**↑	**1.60**↑	**1.60**↑

On the other hand, we also examine the transferability of the features selected by the PSO algorithm and conduct experiments using three models of KNN, random forest, and multilayer perceptron, based on the large-scale and small-scale dataset. The experimental results are shown in Tables 7 and 8, respectively. It can be seen that the selected features can be transferred to KNN and random forest models, but will harm the MLP model, which proves that the features selected in this paper are model-related and not applicable to all models.

Table 7: Experimental results on the transferability of features based on the large-scale dataset

Models	Accuracy	Precision	Recall	F1 score
KNN	**0.45**↑	**0.56**↑	**0.45**↑	**0.48**↑
RF	0.23↑	0.12↑	0.23↑	0.22↑
MLP	3.71↓	2.39↓	3.71↓	3.09↓
Dark-Forest(ours)	0.33↑	**0.56**↑	0.33↑	0.43↑

Table 8: Experimental results on the transferability of features based on the small-scale dataset

Models	Accuracy	Precision	Recall	F1 score
KNN	0.12↑	0.48↓	0.12↑	0.02↓
RF	**1.71**↑	1.65↑	**1.71**↑	**1.69**↑
MLP	4.33↓	0.09↓	4.33↓	2.70↓
Dark-Forest(ours)	1.60↑	**1.68**↑	1.60↑	1.60↑

In addition, considering that the hyperparameter α in the evaluation metric of the PSO algorithm has a high influence on feature selection, we also show the PSO feature selection process based on different α values, as shown in Fig. 13. It can be seen that although the evaluation score of the PSO algorithm decreases with the increase of the number of iterations when α is small, the PSO algorithm tends to compress the number of features rather than improve the detection accuracy to reduce fitness. In experiments, it is found that when the feature selection algorithm with α of 0.5 is completed, only 20 features are retained, resulting in Dark-Forest's test accuracy of only 87.39%, which was lower than the original baseline model. This shows that a too small α may bring about the problem of information loss.

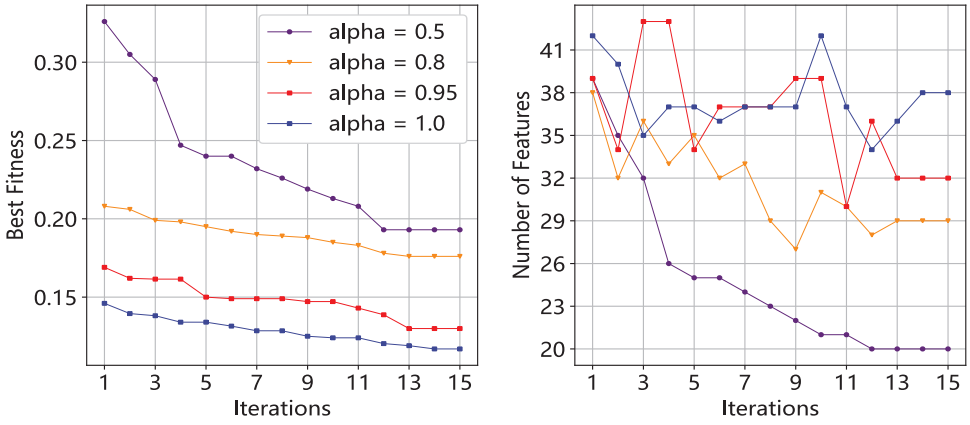

Figure 13: Influence of α on feature selector based on PSO algorithm. **Left:** fitness *vs.* the change of α. **Right:** number of features *vs.* the change of α

5 Conclusions

Affected by the characteristics of encryption and anonymity of dark web, how to realize automatic dark web traffic analysis has always been the research focus in the field of dark web forensics. Aiming at the current pain points in this field, this paper proposes a Dark-Forest model that combines the PSO feature selection algorithm and the tree deep learning method. The model can automatically analyze the behavior of normal traffic, tunnel network traffic and anonymous network traffic only according to the flow spatio-temporal features of traffic data without decrypting the data packets, which can eliminate the gap between different network traffic analysis tasks to a certain extent. On the one hand, from the perspective of detection effect, experiments on the public dataset DIDarknet show that Dark-Forest's accuracy, F1 score, and other evaluation metrics are better than those of Deep Image, random forest, and other baseline models. On the other hand, as for detection efficiency, Dark-Forest can filter out the most redundant features that are less meaningful for detection tasks, so the detection speed is faster than the original DeepForest model. In addition, from the perspective of accessibility, different from the existing deep learning methods based on neural networks, the training and inference process of the Dark-Forest model hardly depend on the accelerators such as GPUs. Therefore, this model can be deployed in firewalls, gateway and other edge computing devices with only CPUs installed, so as to realize the real-time analysis of dark web traffic, which has practical value.

It is also found in the experiments that Dark-Forest still needs to be improved for some specific types of dark web traffic data, while other models have advantages for the detection of these traffic data. Therefore, in addition to further expanding the dataset, the next work will focus on the fusion of multiple models to further improve the detection effect of Dark-Forest. In addition, transfer learning [33] can be used to solve the problem of insufficient samples, and also provides research ideas to improve the effectiveness of the Dark-Forest model in the task of dark web traffic detection based on few-shot learning or even zero-shot learning.

Acknowledgement: The authors wish to express their appreciation to the reviewers for their helpful suggestions which greatly improved the presentation of this paper.

Funding Statement: This research was funded by Henan Provincial Key R&D and Promotion Special Project (Science and Technology Tackling) (212102210165); National Social Science Foundation Key Project (20AZD114); Henan Provincial Higher Education Key Research Project Program (20B520008); Public Security Behavior Scientific Research and Technological Innovation Project of the Chinese People's Public Security University (2020SYS08).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Lukasik, S. (2010). Why the ARPANET was built. *IEEE Annals of the History of Computing, 33(3),* 4–21. DOI 10.1109/MAHC.2010.11.
2. Digital 2022: Another year of bumper growth (2022). https://wearesocial.com/us/blog/2022/01/digital-2022-another-year-of-bumper-growth-2/.
3. Zhou, Z., Feng, J. (2019). Deep forest. *National Science Review, 6(1),* 74–86. DOI 10.1093/nsr/nwy108.
4. Kennedy, J., Eberhart, R. (1995). Particle swarm optimization. *Proceedings of International Conference on Neural Networks*, pp. 1942–1948. Perth, Australia.

5. Habibi, L. A., Kaur, G., Rahali, A. (2020). DIDarknet: A contemporary approach to detect and characterize the darknet traffic using deep image learning. *Proceedings of the 10th International Conference on Communication and Network Security*, pp. 1–13. Tokyo, Japan.
6. Zain, ul A. M., Saleem, S., Ejaz, M. (2019). VPN traffic detection in SSL-protected channel. *Security and Communication Networks*. https://www.hindawi.com/journals/scn/2019/7924690/.
7. Islam, F. U., Liu, G., Liu, W. (2020). Identifying VoIP traffic in VPN tunnel via flow spatio-temporal features. *Mathematical Biosciences and Engineerin, 17(5),* 4747–4772. DOI 10.3934/mbe.2020260.
8. Xu, W., Zou, F. (2021). Obfuscated tor traffic identification based on sliding window. *Security and Communication Networks*. https://www.hindawi.com/journals/scn/2021/5587837/.
9. Zhioua, S. (2013). Tor traffic analysis using hidden markov models. *Security and Communication Networks, 6(9),* 1075–1086. DOI 10.1002/sec.669.
10. He, G., Yang, M., Luo, J. (2015). A novel application classification attack against Tor. *Concurrency and Computation: Practice and Experience, 27(18),* 5640–5661. DOI 10.1002/cpe.3593.
11. Afuwape, A. A., Xu, Y., Anajemba, J. H. (2021). Performance evaluation of secured network traffic classification using a machine learning approach. *Computer Standards & Interfaces, 78,* 103545. DOI 10.1016/j.csi.2021.103545.
12. Uğurlu, M., Doğru, İ. A, Arslan, R. S. (2021). A new classification method for encrypted internet traffic using machine learning. *Turkish Journal of Electrical Engineering & Computer Sciences, 29(5),* 2450–2468. DOI 10.3906/elk-2011-31.
13. Rao, Z., Niu, W., Zhang, X. S. (2018). Tor anonymous traffic identification based on gravitational clustering. *Peer-to-Peer Networking and Applications, 11(3),* 592–601. DOI 10.1007/s12083-017-0566-4.
14. Wang, W., Zhu, M., Wang, J. (2017). End-to-end encrypted traffic classification with one-dimensional convolution neural networks. *Proceedings of 2017 IEEE International Conference on Intelligence and Security Informatics*, pp. 43–48. Beijing, China.
15. Shapira, T., Shavitt, Y. (2021). FlowPic: A generic representation for encrypted traffic classification and applications identification. *IEEE Transactions on Network and Service Management, 18(2),* 1218–1232. DOI 10.1109/TNSM.2021.3071441.
16. Hochreiter, S., Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9(8),* 1735–1780. DOI 10.1162/neco.1997.9.8.1735.
17. Cho, K., Merrienboer, B. V., Gulcehre, C., Bahdanau, D., Bougares, F. et al. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv:1406.1078.
18. Lu, B., Luktarhan, N., Ding, C. (2021). ICLSTM: Encrypted traffic service identification based on inception-LSTM neural network. *Symmetry, 13(6),* 1080. DOI 10.3390/sym13061080.
19. Gerard, D. G., Arash, H. L., Mohammad, M. (2017). Characterization of encrypted and VPN traffic using time-related features. *Proceedings of the 2nd International Conference on Information Systems Security and Privacy*, pp. 407–414. Rome, Italy.
20. Islam, F. U., Liu, G., Zhai, J. (2021). VoIP traffic detection in tunneled and anonymous networks using deep learning. *IEEE Access, 9,* 59783–59799. DOI 10.1109/ACCESS.2021.3073967.
21. Liu, X., You, J., Wu, Y. (2020). Attention-based bidirectional GRU networks for efficient HTTPS traffic classification. *Information Sciences, 541,* 297–315. DOI 10.1016/j.ins.2020.05.035.
22. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L. et al. (2017). Attention is all you need. *Proceedings of the Advances in Neural Information Processing Systems*, pp. 6000–6010. Long Beach, CA, USA.
23. Yao, H., Liu, C., Zhang, P. (2019). Identification of encrypted traffic through attention mechanism based long short term memory. *IEEE Transactions on Big Data, 8,* 241–252. DOI 10.1109/TBDATA.2019.2940675.
24. Guo, L., Wu, Q., Liu, S. (2020). Deep learning-based real-time VPN encrypted traffic identification methods. *Journal of Real-Time Image Processing, 17(1),* 103–114. DOI 10.1007/s11554-019-00930-6.

25. Chen, M., Shi, X., Zhang, Y. (2017). Deep feature learning for medical image analysis with convolutional autoencoder neural network. *IEEE Transactions on Big Data, 7(4),* 750–758. DOI 10.1109/TBDATA.2017.2717439.
26. Iliyasu, A. S., Deng, H. (2019). Semi-supervised encrypted traffic classification with deep convolutional generative adversarial networks. *IEEE Access, 8,* 118–126. DOI 10.1109/Access.6287639.
27. Radford, A., Metz, L., Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv:1511.06434.
28. Dai, J., Wang, T., Wang, S. (2020). Network traffic classification method based on deep forest. *Journal of National University of Defense Technology, 42(4),* 30–34.
29. Tran, B., Xue, B., Zhang, M. (2017). A new representation in PSO for discretization-based feature selection. *IEEE Transactions on Cybernetics, 48(6),* 1733–1746. DOI 10.1109/TCYB.2017.2714145.
30. Mistry, K., Zhang, L., Neoh, S. C. (2016). A micro-GA embedded PSO feature selection approach to intelligent facial emotion recognition. *IEEE Transactions on Cybernetics, 47(6),* 1496–1509. DOI 10.1109/TCYB.6221036.
31. Wold, S., Esbensen, K., Geladi, P. (1987). Principal component analysis. *Chemometrics and Intelligent Laboratory Systems, 2(1–3),* 37–52. DOI 10.1016/0169-7439(87)80084-9.
32. Srinivas, M., Patnaik, L. M. (1994). Genetic algorithms: A survey. *Computer, 27(6),* 17–26. DOI 10.1109/2.294849.
33. Pan, S. J., Yang, Q. (2009). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering, 22(10),* 1345–1359. DOI 10.1109/TKDE.2009.191.