# MGEL: A Robust Malware Encrypted Traffic Detection Method Based on Ensemble Learning with Multi-Grained Features *

Juncheng Guo[1,2], Yafei Sang[1(✉)], Peng Chang[1], Xiaolin Xu[3], and Yongzheng Zhang[1,2]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{guojuncheng,sangyafei,changpeng,zhangyongzheng}@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3] National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing, China
xuxiaolin@cert.org.cn

**Abstract.** As the use of encryption protocols increase, so does the challenge of identifying malware encrypted traffic. One of the most significant challenges is the robustness of the model in different scenarios. In this paper, we propose an **ensemble learning** approach based on **multi-grained** features to address this problem which is called MGEL. The MGEL builds diverse base learners using multi-grained features and then identifies malware encrypted traffic in a stacking way. Moreover, we introduce the self-attention mechanism to process sequence features and solve the problem of long-term dependence. We verify the effectiveness of the MGEL on two public datasets and the experimental results show that the MGEL approach outperforms other state-of-the-art methods in four evaluation metrics.

**Keywords:** Malware Encrypted Traffic Detection · Ensemble Learning · Multi-Grained Features · Self-Attention.

## 1 Introduction

With the widespread use of encryption technology, the privacy, freedom, anonymity of Internet users have been greatly protected, but also it has allowed attackers to evade the anomaly detection system. For example, an attacker invades and attacks the system by encrypting malware traffic. Besides, criminals penetrate the darknet through tools such as Tor [12] to trade illegally. That is to say, the abuse of encryption technology poses many challenges to network anomaly detection and secure management. Therefore, the identification of malware encrypted traffic has aroused great concern in academia and industry.

* Supported by the National Natural Science Foundation of China (Grant No.U1736218). The corresponding author is Yafei Sang.

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77964-1_16

There are traditional rule-based methods such as port numbers [18] and deep packet inspection(DPI) [11]. Port numbers are a quick and easy approach. However, its accuracy has declined because of the use of well-known port numbers. Deep packet inspection(DPI) is to find patterns or keywords in the payloads. But this method can only handle plaintext information and the matching process is computationally costly.

Recently, some researchers have been using machine learning methods to solve such problems. They extract packet features from plaintext messages of the SSL/TLS handshake packets [1, 2] or use statistical features at the flow level [9]. However, TLS/SSL handshake information may be incomplete in real scenarios(*e.g.*, the client reconnects to the server via session ID without the process of certificate exchange and negotiate the secret key again), resulting in a low accuracy rate of identification. The statistical features, such as byte distribution, transition probabilities of lengths and times are often used as features. However, statistical features require additional preprocessing, and the statistical method is based on experience, which weakens the generalization ability of the model.

Deep learning has achieved great success in the areas of image process and natural language process, and are also gradually being applied to the field of traffic identification. In some recent literature, researchers mainly use AutoEncoder [10], CNN [16, 17], LSTM [20] and other network structures to learn representations from raw data. Although deep learning methods have high accuracy, there are some limitations in practical use:

(1) Deep learning methods are overly dependent on data size, and when there is not enough data for the real scenario, deep learning methods cannot learn a good representation.
(2) Some research only uses encrypted payloads of a few packets in a flow for identification, which can fail in some application-level issues. Moreover, flow sequence features mainly face the challenge of the long-term dependence problem.
(3) Deep learning methods typically have millions of parameters, and model performance is greatly influenced by hyperparameters. Moreover, Deep learning methods cannot adapt well to the new scenario.

The ability to generalize in different scenarios is critical because there are many types of malware encrypted traffic. Drawing the idea of ensemble learning that two heads are better than one, we use diverse base learners to enhance the model's robustness in different scenarios. In order to obtain diverse base learners, we use multi-grained features to learn separately. We first define packet features and flow features. Packet features refer to key fields in certain packets, such as port number, validity time of certificate, cipher suite, *etc.* Using the discrepancy in these key fields, we can directly identify malware encrypted traffic. Flow features refer to sequence features that reflect the communication process between the client and the server, such as packet length sequence, message type sequence, and time interval sequence. By learning these sequence features, we can get the pattern of communication of different types of traffic, so as to distinguish between normal and malware encrypted traffic. After getting diverse base

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77964-1_16

learners, we use the idea of stacking to ensemble each base learner to get the final identification result. This idea of ensemble learning based on multi-grained features enhances the robustness of the model in different scenarios.

In this paper, We propose an ensemble learning approach based on multi-grained features(MGEL) for malware encrypted traffic identification. The MGEL first obtains diverse base learners using models suitable for different grained features. Specifically, we use Xgboost to learn from packet features and use a model based on Bi-LSTM+Self-Attention to learn from flow features. Then, we introduce the idea of stacking to ensemble each base learner. We verify the effectiveness of the MGEL on two public datasets and the experimental results show that our MGEL approach outperforms other state-of-the-art methods in four evaluation metrics.

Our contributions can be briefly summarized as follows:

(1) We propose an ensemble learning approach based on multi-grained features for malware encrypted traffic identification. The attributes of multi-grained features and the characteristics of ensemble learning enhance the robustness of the model in different scenarios.
(2) For sequence features, we introduce a self-attention mechanism to solve the long-term dependency problem. Meanwhile, self-attention runs faster than RNN because its computation does not depend on the previous state.
(3) Our MGEL achieves excellent results on the two public datasets for malware traffic encrypted identification and outperforms other state-of-the-art methods.

## 2 Background and Related Work

### 2.1 SSL/TLS Encrypted Protocols

The Secure Sockets Layer(SSL) [6] and its successor Transport Layer Security(TLS) [5] protocol are popular encryption protocols used to protect client-server sessions. Fig. 1 shows the process of an SSL/TLS session. It mainly includes the handshake process and communication process. The handshake process is used to negotiate secret key and verify identification. Specifically, the client and server first exchange Client Hello and Server Hello messages to establish the session. Then both sides negotiate the secret key through Certificate, Server Key Exchange, Client Key Exchange messages, and finally the client sends ChangeCipherSpec and Finished messages to complete the handshake process. Then the subsequent communication process encrypts the communication payloads using the negotiated secret key and encryption algorithm.

### 2.2 Mainstream Method

**Conventional Methods** Conventional methods generally identify malware encrypted traffic by port number and DPI(Deep packet Inspect). The port-based

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77964-1_16

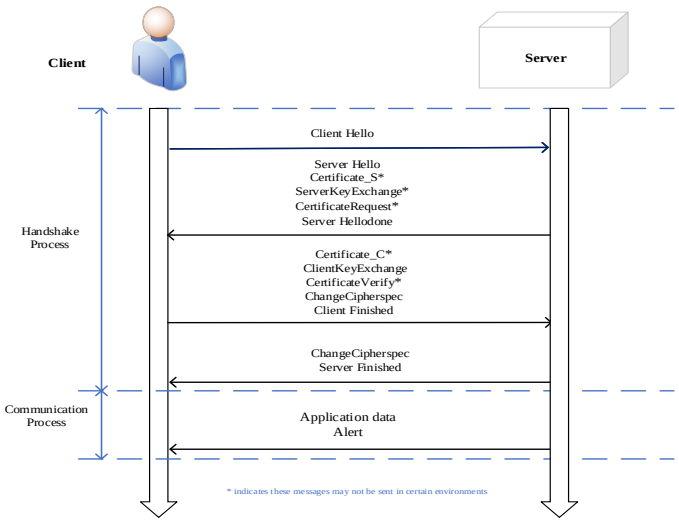



**Fig. 1.** A Communication Session of SSL/TLS Protocol.

approach is very efficient as it identifies the port number extracted from the packet and matches it with IANA TCP/UDP. However, the accuracy of port-based methods has dramatically decreased due to the widespread use of port obfuscation and dynamically assigned ports [4]. Deep Packet Inspect(DPI) technology is based on the analysis of information available in the payload of packets. This method can only handle plaintext information and the matching process is computationally costly.

**Machine Learning Methods** The machine learning method is mainly based on the assumption that there are some distinguishable statistical features between normal and malware encrypted traffic. From millions of flows, Anderson *et.* analyze the discrepancy in TLS key fields and summarize a number of distinguishing features such as Cipher Suite, Extension, Client's Public key Length, Validity of Certificate, *etc* [2]. In addition, some flow-level statistical features such as byte distribution, transition probabilities of lengths and times [9, 7, 13] are often used as features. However, since the statistical features are designed empirically, they may be valid only for data in certain scenarios and have weak generalization ability.

**Deep Learning Methods** Deep learning methods have been widely used in image process and natural language process, but are still new to the problem of encrypted traffic identification. Currently, the research of encrypted traffic identification based on deep learning mainly uses flow sequences or raw byte

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77964-1_16

data as input to learning a good representation by AutoEncoder(AE), Convolutional Neural Neteork(CNN), Recurrent Neural Network(RNN), *etc.* Wang *et.* transform the raw data into images and learn feature representations using 2D-CNN [17] and 1D-CNN [16]. Liu *et.* proposed an AutoEncoder model based on flow sequences and improve performance by adding classification loss and reconstruction loss [10]. Li *et.* proposed a byte segment neural network where payloads are divided into multiple segments [8]. They use the Attention mechanism to further select significant representations as to the input of the softmax layer.

## 3 The MGEL Framework

Our ensemble learning model with multi-grained features(MGEL) is shown in Fig. 2. The MGEL consists of three parts which are extracting multi-grained features, first stage of MGEL, second stage of MGEL. In the first part, we use the open source tool flowcontainer[1] to obtain multi-grained features from the raw flow data, and in the second part, we learn three base learners(MGEL-1, MGEL-2, MGEL-3) using multi-grained features. In the third part, we use the output of the three base learners to learn the meta learner. In the following section, we will show the details of the last two parts.

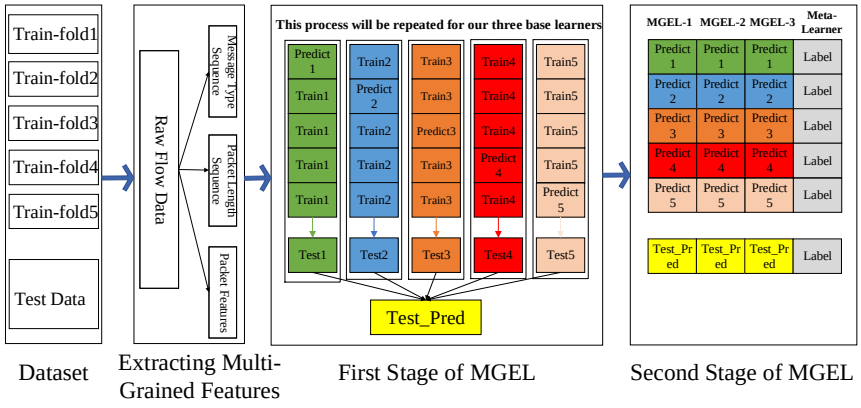



**Fig. 2.** The flow diagram of identifying malware encrypted traffic by MGEL. We perform a five-fold cross-validation on the model, which is represented by 5 colors.

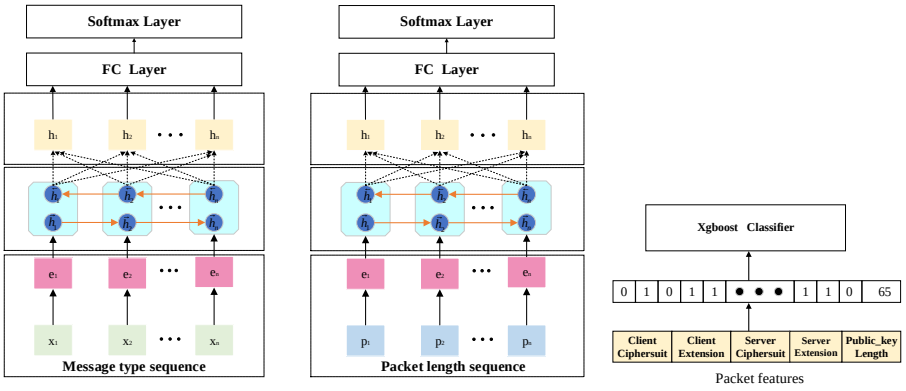
[1] https://github.com/jmhIcoding/flowcontainer

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77964-1_16

### 3.1 The First Stage of MGEL

In the first stage, we use the training dataset to learn three base learners by multi-grained features. Three base learners are based on flow features and packet features as shown in Fig. 3. Suppose the size of the training dataset and test dataset are $M_{train}$, $M_{test}$ , then after the first stage of learning, we will get the prediction matrix of $P_{train} \in \mathbb{R}^{M_{train} \times 3}$,$P_{test} \in \mathbb{R}^{M_{test} \times 3}$.

**Base Learner Based on Flow Feature** Flow features reflect the communication process between the client and the server. As shown in Fig. 3(a) and Fig. 3(b), MGEL-1 and MGEL-2 are based on message type sequences and packet length sequences, respectively. They consist of three layers, which are Embedding Layer, Bidirectional LSTM Layer and Self-Attention Layer.




(a) MGEL-1 (b) MGEL-2 (c) MGEL-3

**Fig. 3.** The architecture of three base learners. Subgraph (a) and (b) are base learners based on flow features(message type sequence and packet length sequence). From bottom to top are Embedding Layer, Bi-LSTM Layer, and Self-Attention Layer. Subgraph (c) is base learner based on packet features, and the packet features will be mapped to one-hot vectors to input into the Xgboost Claffifer.

*Embedding Layer* Due to the large range values of flow features, the traditional one-hot method can cause high-dimensional and sparse vectors. Drawing on the idea of embedding in natural language processing, we map each value of flow features to a fixed-length vector. We also introduce pad and unk tokens in the dictionary to deal with unknown values and sequence length inconsistency.

Assume that the size of the dictionary is $V$, and the embedded vector dimension is $d$. The embedding layer can be viewed as a matrix $E \in \mathbb{R}^{V \times d}$, where each

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77964-1_16

row of the matrix represents a $d$-dimensional vector corresponding to a certain value. The matrix $E$ is learnable so that we can get embedding vectors that are more context-sensitive.

*Bi-directional LSTM Layer* A bi-directional *lstm* runs a forward and backward *lstm* on a sequence starting from the left and the right ends, respectively. A forward language model computes the probability of the sequence by modeling the probability of tokens $s_t$ given the history$(s_1, ..., s_{t-1})$:

$$p(s_1, ..., s_N) = \prod_{t=1}^{N} p(s_t | s_1, ..., s_{t-1}) \tag{1}$$

A backward language model is similar to a forward language model, except it runs over the sequence in reverse, predicting the previous token given the future context:

$$p(s_1, ..., s_N) = \prod_{t=1}^{N} p(s_t | s_{t+1}, ..., s_N) \tag{2}$$

After going through bi-directional *lstm* Layer, we can get two context-dependent representations $\overrightarrow{h}_t$ and $\overleftarrow{h}_t$ at each position $t$. we concatenate them and get the final output for each position: $o_t = [\overrightarrow{h}_t, \overleftarrow{h}_t]$. With this setting, the output of bi-directional *lstm* can be expressed as $o = [o_1, o_2, ..., o_n]$, where $o$ contains bi-directional information about the the whole sequence.

*Self-Attention Layer* In order to improve the speed of the model and solve the problem of long-term dependencies, we introduce a self-attention mechanism [15], which has recently been widely used in natural language processing and computer vision. The self-attention mechanism generally adopts the form of query-key-value($Q$, $K$, $V$), and its calculation equation can be expressed as follows:

$$\begin{aligned} H &= softmax(\frac{K^T Q}{\sqrt{d_k}})V \\ Q &= XW^Q, K = XW^K, V = XW^V \end{aligned} \tag{3}$$

where $X = [x_1, x_2, ..., x_N]$ is the input sequence, $H = [h_1, h_2, ..., h_N]$ is the output sequence. $d_k$ is the dimension of $Q$ and $K$. $W^Q, W^K, W^V$ are learnable parameters. Through the self-attention mechanism, we can learn the parts that should be focused on in the sequence. At the same time, because the calculation of each step does not need to depend on the previous step, the computational speed is faster than the RNN model.

**Base Learner Based on Packet Feature** Packet features refer to some key fields in the packet such as cipher suite, extension, length of public key, validity time of certificate, *etc.* According to the analysis in [2], there is a clear distinction between these key fields in normal and malware encrypted traffic, which can be

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77964-1_16

learned by a machine learning model. Through a comprehensive comparison, we choose Xgboost as the base learners and the structure of the model is shown in Fig. 3(c).

*Xgboost* Xgboost [3] is a scalable machine learning system for tree boosting methods, which achieves state-of-the-art results in many areas. We choose Xgboost as the base learner based on two main considerations. Firstly, gradient tree boosting combines the advantages of bagging and boosting and has been shown to give state-of-the-art results in many applications. Secondly, by proposing a novel tree learning algorithm, Xgboost can handle sparse data and the lack of packet features. For a given sample , a Xgboost model uses $M$ additive functions to predict the output:

$$\hat{y}_i = \phi(x_i) = \sum_{k=1}^{M} f_k(x_i) \tag{4}$$

where $f_k$ is a regression tree (also known as CART). Since the parameters of the model are functions, we cannot use optimization algorithms like SGD. Instead, the model is trained in an additive manner. Formally, let $\hat{y}_i^{(t-1)}$ be the prediction of the $i^{th}$ instance at the $(t-1)^{th}$ iteration, we will need to add $f_t$ to minimize the following objective function.

$$\begin{aligned} &\mathcal{L}^t(\phi) = \sum_{i=1}^{N} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \\ &\text{where } \Omega(f_t) = \gamma T + \frac{1}{2}\lambda \left\| w \right\|^2 \end{aligned} \tag{5}$$

where $l$ is a convex loss function that measures the difference between the target $y_i$ and prediction $\hat{y}_i^{(t-1)}$, regularizer $\Omega(f_t)$ is used to penalize the complexity of the $t$th tree model. Equation 4 means we should add the $f_t$ that most minimizes $\mathcal{L}^t(\phi)$. In practice, Second-order approximation can be used to quickly optimize the objective. By performing a second-order Taylor approximation expansion of the loss function, we can approximate the objective function as following.

$$\mathcal{L}^t(\phi) \simeq \sum_{i=1}^{N} [g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t) \tag{6}$$

where $g_i = \partial_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$,$h_i = \partial^2_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$.

Therefore, we can transform packet features into one-hot vectors as the input of Xgboost and set the parameter $M$ for training. Moreover, the Xgboost model is interpretable, and we can evaluate the importance of features based on the number of times they are selected as split points.

### 3.2 The Second Stage of MGEL

In the second stage, we first train the meta learner using the output $P_{train} \in \mathbb{R}^{M_{train} \times 3}$ of the three base learners on the training dataset. For simplicity, we

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77964-1_16

choose logistic regression as the meta learner, which can be represented as:

$$Y = \sigma(f(P_{train})) \tag{7}$$

where $\sigma(\cdot)$ is activation function, $f$ is the parameter to be learned, $Y \in \mathbb{R}^{M_{train} \times 1}$

After the parameters are trained, we use $P_{test} \in \mathbb{R}^{M_{train} \times 3}$ as input to predict the performance of our MGEL model on the test dataset.

## 4 Experment and Results

### 4.1 Experiment Settings

**Dataset** The first dataset(*CIC-InvesAndMal2019*) is regenerated from *CIC-InvesAndMal2019* [14] which includes 10 different families of Ransomware, such as Charger family, Jisut family, Koler family, *etc.* A total of 3,797,000 data packets and 63,953 flows are included in the malware sample. The benign sample comes from normal Android applications, including 13,474,342 data packets and 69,670 flows.

The second dataset(*MTA*) is regenerated from malware-traffic-analysis.net, which is a website that provides real-time updates on current malware prevalent in Europe and the United States. We collected malware samples from 2019 to 2020, totaling 10,793,979 data packets and 50,289 flows. Since the normal sample is not provided on the website, we captured 9,990,438 data packets and 50,243 flows on the normal applications of Android.

**Evaluation Metrics** We evaluate and compare our model with the state-of-the-art methods using four metrics. Namely, Accuracy , Precision , Recall , and F-Measure:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{8}$$

$$Recall = \frac{TP}{TP + FN} \tag{9}$$

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

$$F - Measure = \frac{2Precision * Recall}{Precision + Recall} \tag{11}$$

Where TP(True Positive) represents malware traffic is correctly identified as malware traffic; FP(False Positive) represents benign traffic is incorrectly identified as malware traffic; TN(True Negative) represents benign traffic is correctly identified as benign traffic; FN(False Negative) represents malware traffic is incorrectly identified as benign traffic.

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77964-1_16

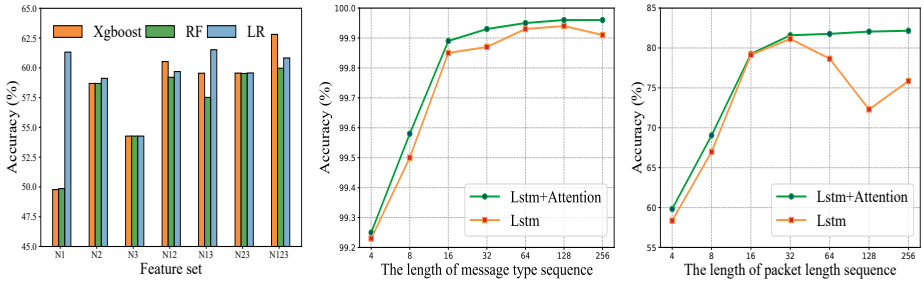
**Experimental Setings and Baselines** For the base learner based on flow features, We use the Adam optimizer with a batch size of 128, for training where the learning rate was set to 5e−3. For the base learner based on Xgboost , we choose the maximum depth of the tree to be 3 and the number of trees to be 160.

We compare our methods with other state-of-the-art methods as follows:

(1) FS-Net uses a multi-layer encoder-decoder structure and reconstruction mechanism to identify malware flows from flow sequences(message type sequences or packet length sequences) [10].
(2) MAMPF uses Random Forest to identify malware flows with features learned from message type and length block Markov models [9].
(3) FoSM constructs a first-order Markov model using message sequences to discriminate the class of traffic based on the maximum likelihood probability [7].
(4) SoSM is similar to FoSM, but utilizes a second-order Markov model [13].
(5) 2D-CNN converts raw traffic data into an image and use two-dimensional CNN for classification [17].
(6) 1D-CNN views raw traffic data as an article and extracts features using a 1D CNN and Max Pooling layer, and finally classifies them using a softmax layer [16].

### 4.2 Analysis on MGEL Component

We explore several key points of each base learner in this section. For the base




(a) (b) (c)

**Fig. 4.** The Experimental Results of MGEL Key Component

learner based on packet features, there are two main problems. The first one is which packet features should be selected, and the second one is which classifier should be chosen.

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77964-1_16

In Fig. 4(a), We compare the performance of Xgboost, Random Forest(RF) and Logistic Regression(LR) on the *CIC-InvesAndMal2019* with different combinations of features. Due to space constraints, we use *N1*, *N2*, *N3* accordingly to denote the client-side cipher suite and extension fields, the server-side cipher suite and extension fields, and the public key length. The combination of numbers indicates the combination of features, *e.g.*, *N12* means we use both client-side and server-side cipher suites and extension fields. According to the experimental results, we can see that the Xgboost method can obtain the highest accuracy rate. It can also be seen that as more features are added, the accuracy of the model is improved except for Logistic Regression. This is attributed to the fact that the tree-based model is more adept at dealing with missing features. Therefore, we choose Xgboost as the base learner and *N123* as the packet features.

For the base learner based on flow features, we focus on exploring the effect of sequence length and the self-attention mechanism on the accuracy of the model.

In Fig. 4(b) and Fig. 4(c), we take the length of the message type sequence and packet length sequence as [4, 8, 16, 32, 64, 128, 256], respectively, and compare the accuracy under the *CIC-InvesAndMal2019* dataset with and without attention layer. From the experimental results, it can be seen that the introduction of the self-attention layer can improve the accuracy of the model. This becomes more obvious as the sequence length increases, which is attributed to the fact that the self-attention mechanism is stronger than LSTM in solving the long-term dependence problem. Moreover, we can find that when the sequence length exceeds 128, the accuracy improvement of the model is no longer obvious. Therefore, we set the sequence length of both MGEL-1 and MGEL-2 to 128.

### 4.3 Comparison Results

We conducted comparative experiments on two public datasets using the six state-of-the-art methods mentioned in Section 4.1, and the experimental results are shown in Table 1 and Table 2. From Table 1 and Table 2, we can obtain the following conclusions:

(1) MGEL achieves the best performance and outperforms all the other methods on all the overall metrics. In addition, we can see that MGEL can reach more than 99% of F1 values on both two public datasets.
(2) Models using a single feature (FS-Net, FoSM, SoSM) have unstable performances on the two public datasets. For example, FS-Net has higher accuracy on *MTA* dataset, while FoSM and SoSM have better performance on *CIC* dataset. This is due to the different sensitivity of data to features, further confirming the importance of using multi-grained features.
(3) Deep learning models using raw byte data (1D-CNN, 2D-CNN) and multi-attribute features models(MAMPF) have a robust performance on both public datasets, but in general worse than MGEL. This is because of the advantage of self-attention and stacking ensemble method.

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77964-1_16

**Table 1. Comparison Results on *CIC-InvesAndMal2019* Dataset**

| Method | Accuracy | Precision | Recall | F-Measure |
|---|---|---|---|---|
| **MGEL** | **0.9988** | **0.9998** | **0.9978** | **0.9988** |
| FS-Net | 0.7953 | 0.7760 | 0.8859 | 0.8273 |
| MAMPF | 0.9332 | 0.9184 | 0.9613 | 0.9394 |
| FoSM | 0.933 | 0.9765 | 0.8976 | 0.9354 |
| SoSM | 0.9587 | 0.9558 | 0.9680 | 0.9619 |
| 2D-CNN | 0.744 | 0.7429 | 0.7463 | 0.7446 |
| 1D-CNN | 0.7424 | 0.694 | 0.7684 | 0.7293 |

**Table 2. Comparison Results on *MTA* Dataset**

| Method | Accuracy | Precision | Recall | F-Measure |
|---|---|---|---|---|
| **MGEL** | **0.9999** | **0.9999** | **1.0** | **0.9999** |
| FS-Net | 0.9998 | 0.9998 | 0.9997 | 0.9998 |
| MAMPF | 0.9984 | 0.9989 | 0.9977 | 0.9983 |
| FoSM | 0.9772 | 0.9904 | 0.9599 | 0.9749 |
| SoSM | 0.9780 | 0.9955 | 0.9566 | 0.9577 |
| 2D-CNN | 0.9949 | 0.9958 | 0.9934 | 0.9946 |
| 1D-CNN | 0.9970 | 0.9947 | 0.9991 | 0.9969 |

### 4.4 The Advanced of Ensemble Learning

In order to verify the advanced of the ensemble method, we compared the performance of a single model on two public datasets. Table 3 is our results on the *CIC-InvesAndMal2019* dataset, and Table 4 is our results on the *MTA* dataset. It can be seen from Table 3 and Table 4 that a base learner trained using only a single feature cannot achieve the best performance on different datasets. With the stacking ensemble method, the adaptability of features on different datasets can be adjusted adaptively, so that the best performance can be achieved on different datasets.

**Table 3. The Results of Advanced Analysis on *CIC-InvesAndMal2019* Dataset**

| Method | Accuracy | Precision | Recall | F-Measure |
|---|---|---|---|---|
| **MGEL** | **0.9988** | **0.9998** | **0.9978** | **0.9988** |
| base-learner 1 | 0.9988 | 0.9998 | 0.9978 | 0.9998 |
| base-learner 2 | 0.7403 | 0.7634 | 0.6964 | 0.7284 |
| base-learner 3 | 0.6221 | 0.6406 | 0.5502 | 0.5963 |

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77964-1_16

**Table 4. The Results of Advanced Analysis on *MTA* Dataset**

| Method | Accuracy | Precision | Recall | F-Measure |
|---|---|---|---|---|
| MGEL | **0.9999** | **0.9999** | **1.0** | **0.9999** |
| base-learner 1 | 0.9672 | 0.9661 | 0.9683 | 0.9672 |
| base-learner 2 | 0.9980 | 0.9984 | 0.9969 | 0.9977 |
| base-learner 3 | 0.9908 | 1.0 | 0.9817 | 0.9907 |

## 5 Conclusion and Future Work

In this paper, we propose an ensemble learning approach with multi-grained features for malware encrypted traffic identification. It jointly trains three base learners using flow features and packet features, and finally ensembles the three base learners by a stacking way. Since the three base learners use multi-grained features for learning, they can focus on different parts of the data. Moreover, our model is more robust on different scenarios by ensembling base learners through stacking. We validate the effectiveness of the MGEL on two public datasets, and the experimental results demonstrate that the MGEL can achieve an excellent identification performance and outperform other state-of-the-art methods. In the future, we would like to explore the performance of the model on multi-classification tasks. In addition, we would like to apply for the recent advances in the field of deep learning to improve the performance of traffic classification.

## References

1. Anderson, B., McGrew, D.: Identifying encrypted malware traffic with contextual flow data. In: Proceedings of the 2016 ACM workshop on artificial intelligence and security. pp. 35–46 (2016)
2. Anderson, B., Paul, S., McGrew, D.: Deciphering malware's use of tls (without decryption). Journal of Computer Virology and Hacking Techniques **14**(3), 195–211 (2018)
3. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. pp. 785–794 (2016)
4. Constantinou, F., Mavrommatis, P.: Identifying known and unknown peer-to-peer traffic. In: Fifth IEEE International Symposium on Network Computing and Applications (NCA'06). pp. 93–102. IEEE (2006)
5. Dierks, T., Rescorla, E.: The transport layer security (tls) protocol version 1.2 (2008)
6. Freier, A., Karlton, P., Kocher, P.: The secure sockets layer (ssl) protocol version 3.0. Tech. rep., RFC 6101 (2011)
7. Korczyński, M., Duda, A.: Markov chain fingerprinting to classify encrypted traffic. In: IEEE INFOCOM 2014-IEEE Conference on Computer Communications. pp. 781–789. IEEE (2014)
8. Li, R., Xiao, X., Ni, S., Zheng, H., Xia, S.: Byte segment neural network for network traffic classification. In: 2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS). pp. 1–10. IEEE (2018)

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77964-1_16

9. Liu, C., Cao, Z., Xiong, G., Gou, G., Yiu, S.M., He, L.: Mampf: Encrypted traffic classification based on multi-attribute markov probability fingerprints. In: 2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS). pp. 1–10. IEEE (2018)
10. Liu, C., He, L., Xiong, G., Cao, Z., Li, Z.: Fs-net: A flow sequence network for encrypted traffic classification. In: IEEE INFOCOM 2019-IEEE Conference on Computer Communications. pp. 1171–1179. IEEE (2019)
11. Park, J.S., Yoon, S.H., Kim, M.S.: Performance improvement of payload signature-based traffic classification system using application traffic temporal locality. In: 2013 15th Asia-Pacific Network Operations and Management Symposium (APNOMS). pp. 1–6. IEEE (2013)
12. Rezaei, S., Liu, X.: Deep learning for encrypted traffic classification: An overview. IEEE communications magazine **57**(5), 76–81 (2019)
13. Shen, M., Wei, M., Zhu, L., Wang, M., Li, F.: Certificate-aware encrypted traffic classification using second-order markov chain. In: 2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS). pp. 1–10. IEEE (2016)
14. Taheri, L., Kadir, A.F.A., Lashkari, A.H.: Extensible android malware detection and family classification using network-flows and api-calls. In: 2019 International Carnahan Conference on Security Technology (ICCST). pp. 1–8. IEEE (2019)
15. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. arXiv preprint arXiv:1706.03762 (2017)
16. Wang, W., Zhu, M., Wang, J., Zeng, X., Yang, Z.: End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In: 2017 IEEE International Conference on Intelligence and Security Informatics (ISI). pp. 43–48. IEEE (2017)
17. Wang, W., Zhu, M., Zeng, X., Ye, X., Sheng, Y.: Malware traffic classification using convolutional neural network for representation learning. In: 2017 International Conference on Information Networking (ICOIN). pp. 712–717. IEEE (2017)
18. Zejdl, P., Ubik, S., Macek, V., Oslebo, A.: Traffic classification for portable applications with hardware support. In: 2008 International Workshop on Intelligent Solutions in Embedded Systems. pp. 1–9. IEEE (2008)
19. Zheng, W., Gou, C., Yan, L., Mo, S.: Learning to classify: A flow-based relation network for encrypted traffic classification. In: Proceedings of The Web Conference 2020. pp. 13–22 (2020)
20. Zou, Z., Ge, J., Zheng, H., Wu, Y., Han, C., Yao, Z.: Encrypted traffic classification with a convolutional long short-term memory neural network. In: 2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS). pp. 329–334. IEEE (2018)

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77964-1_16