2023 20th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON) | 979-8-3503-0052-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/SECON58729.2023.10287432

EVS2vec: A Low-dimensional Embedding Method for Encrypted Video Stream Analysis

Luming Yang, Yongjun Wang*, Shaojing Fu, Lin Liu, Yuchuan Luo
College of Computer, National University of Defense Technology, Changsha, China

***Abstract*—The rise in video streaming has led to an increase in network traffic, with encrypted video streams playing a significant role in illegal video detection. However, there are challenges in performing content analysis of encrypted video streams, including label limitations and complex calculations. In this paper, we proposed a low-dimensional embedding method based on Byte Rate Sequences (BRS), named EVS2vec (Encrypted Video Stream to Vector), to solve these problems effectively. It can represent the content of encrypted video streams with low-dimensional vectors by mapping the indefinite-length sequence into a low-dimensional Euclidean space. EVS2vec can thereby be applied for not only supervised analysis but also unsupervised analysis. Furthermore, using BRS can also save the time overhead on fine-grained network traffic parsing. In order to ensure the content-related distinguishability of the embedding result, inspired by contrastive learning, we designed a network structure based on Recurrent Neural Network (RNN) with self-attention mechanism in EVS2vec and trained it using siamese network. The experiments on a public dataset show that EVS2vec saves storage overhead while containing enough video content information. EVS2vec can achieve a high accuracy of similarity threshold, reaching 96.71%. An 8-dimensional fingerprint for each video is constructed. Moreover, classification and clustering analysis can also be performed with acceptable results.**

***Index Terms*—Encrypted Video Stream, Embedding, BRS, Side-channel Analysis, DASH**

I. Introduction

With the increasing popularity of video streaming, it now takes up a significant portion of Internet traffic and is growing rapidly. According to a Cisco report [1], by 2023, 88% of Internet traffic will be made up of video applications. Watching online videos on mobile devices has become a lifestyle for people. In order to enhance viewer Quality of Service (QoS), many video streaming websites such as YouTube use Dynamic Adaptive Streaming over HTTP (DASH) [2]. Additionally, to protect against sniffing attacks, almost all well-known websites have applied Transport Layer Security (TLS) to encrypt their network traffic. DASH combined with TLS is becoming a widely adopted trend in the video streaming industry.

The massive amount of encrypted video streams presents a significant challenge for detecting illegal content. Many studies [3]–[5] have shown that the application of DASH makes it possible to analyze the content of encrypted video streams, even with the widespread use of encryption techniques by video services. This is due to the Variable Bitrate (VBR) nature of DASH, where the size of encoded fragments is directly related to the complexity of the video content. By analyzing the side-channel information of encrypted video streams without decryption, it is possible to detect illegal videos and prevent the spread of harmful information, contributing to a safe and secure online environment.

* Yongjun Wang is the corresponding author.

Currently, there are several methods for mining video content through side-channel analysis, utilizing different analysis units such as BPPs (Bit-per-Peaks) [3], [6], [7], bursts [4], [8], video fragments [9]–[14], etc. These approaches mainly concentrate on video title classification and have achieved a satisfactory level of accuracy. However, there are still some limitations in these existing works that need to be addressed:

- **Label Limitation:** Current approaches in analyzing encrypted video streams mainly rely on supervised learning to identify video titles in a static dataset. However, they are created with a closed-world assumption, and their limited label space cannot adapt to the open-world scenario. Classifiers will misclassify the unknown types of samples as known labels, forcibly. Rebuilding a new model to address this issue will incur high costs.
- **Complex Computation:** In previous approaches, some requires TCP reassembly to determine the length of video segments, and others based on BPPs and bursts still require dividing and merging network packets transmitted over a period of time. Additionally, flow-splitting based on the 5-tuple is also a necessary step in these existing works, leading to additional computational costs.

The low-dimensional embedding method based on Byte Rate Sequences (BRS) offers a promising solution to these challenges. The method transform the encrypted video stream data into low-dimensional representations, which can effectively represent the content information of the encrypted video stream. As the embedding results are not limited by known labels, it allows for greater adaptability in various downstream analysis tasks. Moreover, the use of BRS eliminates the need for flow splitting and packet reassembly during traffic preprocessing, thus reducing computation costs and making the analysis process more efficient.

The design of a low-dimensional embedding method for encrypted video streams faces several practical challenges. Most importantly, the embedding method requires a content-related differentiated representation of encrypted video streams, so as to better adapt to various downstream analysis tasks. Besides, the representation results should be feature vectors, which are extracted from the indefinite-length BRS sequences.

To address these challenges, in this paper, we present a low-dimensional embedding method, named EVS2vec (Encrypted

Video Stream to Vector). Inspired by contrastive learning, we designed a network structure for EVS2vec based on Recurrent Neural Network (RNN) with self-attention mechanism and trained it using siamese network to ensure content-related distinguishability. EVS2vec is more suitable to the open-world scenario. The embedding results can be applied for both supervised and unsupervised analysis. Additionally, using low-dimensional vectors to represent encrypted video streams reduces storage overhead and makes it suitable for large-scale data analysis tasks. This is the first study that maps encrypted video streams from an indefinite-length sequence into a low-dimensional Euclidean space.

The paper's main contributions are:

1) We propose a low-dimensional embedding method named EVS2vec based on BRS to analyze encrypted video stream. This method overcomes the limitations of previous approaches in terms of label restrictions and complex calculations. To our best knowledge, it is the first work to apply embedding techniques in this field.
2) We apply the embedding results of EVS2vec to perform various downstream tasks such as thresholding, fingerprinting, classification, and clustering. These solutions are designed to demonstrate the versatility of EVS2vec in encrypted video stream analysis.
3) The effectiveness of EVS2vec is thoroughly evaluated on a public dataset. The experimental results show that the accuracy of the discrimination threshold can reach 96.71%. Furthermore, the fingerprinting, classification, and clustering tasks are also successfully performed, highlighting the validity of our approach and the effectiveness of our network design.

The remainder of this paper is organized as follows. Section II shows the threat model and the analysis framework. In Section III, we introduce the proposed the embedding methods (EVS2vec) in detail. Then, we introduce downstream analysis based on EVS2vec in Section IV. We explain our experiments in Section V. Then, we discusses the result of experiments in Section VI. We give a review of related work in Section VII. Finally, the conclusions are laid out in Section VIII.

II. Threat Model and Overview of EVS2vec

In this section, we first present the threat model. Afterwards, we provide a comprehensive overview of our proposed low-dimensional embedding method, EVS2vec.

A. Threat Model

The threat model is depicted in Fig. 1. It assumes that an adversary is able to sniff network traffic. The adversary could be an Internet Service Provider (ISP) or a middleman between a client and a server, who can eavesdrop on the client's communication when playing videos. Despite the use of TLS encryption for DASH streams, the adversary can still obtain the bitrate transmitted over the network through sniffing. The encryption technology does not have the capability to alter the bitrate generated during video playback, making encrypted video streams vulnerable to analysis. It can be applied to identify video while streaming, referred to as a side-channel video identification attack. [5] Network supervisors can deduce what video is being streamed by comparing traffic patterns.

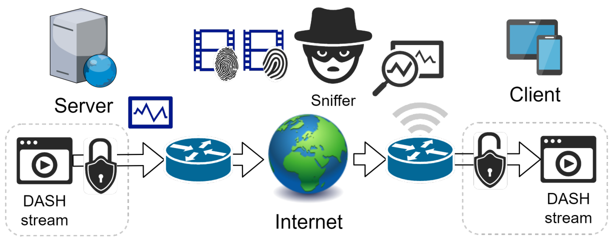


Fig. 1: The threat model of encrypted video stream analysis using side-channel attack.

B. Overview of EVS2vec

We present the EVS2vec, the low-dimensional embedding method for encrypted video stream analysis. The entire analysis framework of EVS2vec is detailed in Fig. 2 and consists of the following components:

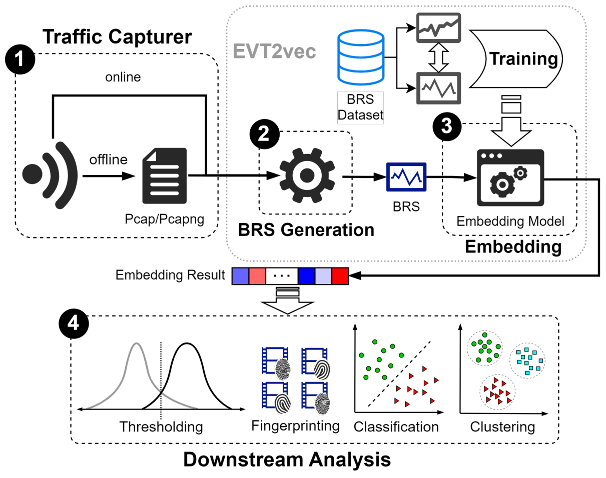


Fig. 2: The analysis framework of EVS2vec.

- **Traffic Capturer:** It involves sniffing network communication during video playback. The raw packets of encrypted video streams are acquired using an external library (e.g. tshark) and saved as pcap/pcapng files.
- **BRS Generation:** BRS is the analysis unit for EVS2vec and acts as the input for the embedding model. Byte rate is the number of bytes transferred per second. BRS is a sequence of changing byte rates over a period of time. The BRS sequence can be generated by either parsing the saved network traffic data or directly generated during sniffing without saving the original traffic data.
- **Embedding:** This component is the core of the framework and processes the preprocessed encrypted video streams to generate the embedding results. After preprocessing, BRS sequences are finally put into the embedding model, and all the downstream analysis methods are also based on the embedding results.
- **Downstream Analysis:** Based on the embedding results of EVS2vec, various downstream analysis tasks can be

performed. Analysts can perform both supervised and unsupervised analysis of encrypted video streams, including thresholding, fingerprinting, classification, and clustering.

III. Implementation of EVS2vec

In this section, we introduce the EVS2vec in detail. As the packet capturer is not the contribution of this paper, we will focus on the BRS generation and the embedding process.

A. BRS Generation

In order to avoid flow splitting and fine-grained network traffic parse and reassembly operations, we propose the use of BRS as a new unit for analysis. Compared with existing approaches, it can simplify the computational complexity.

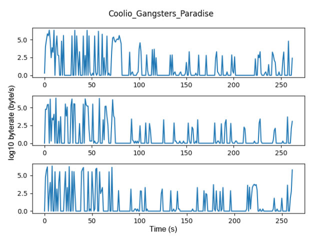
BRS can reflect the real-time complexity of video content. In DASH, video segments of roughly the same duration have different sizes, and the video segment size is content-related. Consequently, the byte rate (or bit rate, where bitrate $= 8 \times$ byterate) on the network is constantly changing in real-time as the video content changes. In general, the flow that provides video services is the main factor affecting the byte rate. During video playback, compared with other applications, the flow data from video services will occupy most of the transmission data. The use of BRS in the training process also enhances the robustness of the embedding model.

We have developed Algorithm 1 to generate BRS for video streams. This algorithm calculates the total payload length of all TCP packets captured per second, without the need to divide different flows based on the 5-tuple (i.e. srcIP, destIP, srcPort, destPort, and Protocol).

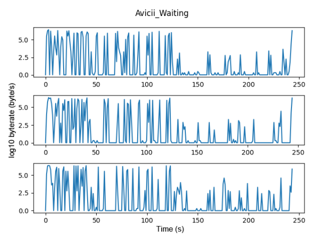
Algorithm 1 Generation of BRS.

```
Require: The packets set that sniffed when video playback
    $\mathcal{P} = \{(t_1, \boldsymbol{p}_1), (t_2, \boldsymbol{p}_2), \cdots, (t_N, \boldsymbol{p}_N)\}$;
    The time interval $\Delta t$, which usually set 1 second;
Ensure: $BRS$;
  $BRS = [\quad]$, $payload\_len = 0$, $t = 0$, $t_0 = 0$
  for $i = 1$ to $N$ do
    while $t_i - t_0 > t + \Delta t$ do
      $BRS.append(0)$
      $t \leftarrow t + \Delta t$
    end while
    if $t_i - t_0 > t$ then
      $BRS.append(payload\_len)$
      $payload\_len \leftarrow 0$, $t \leftarrow t + \Delta t$
    end if
    if $\boldsymbol{p}_i.proto = TCP$ then
      $payload\_len \leftarrow payload\_len + len(\boldsymbol{p}_i.data)$
    end if
  end for
  return $BRS$
```

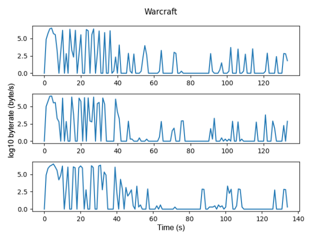
Fig. 3 illustrates the changes in BRS when different videos are played online (each video played 3 times). The X-axis displays the time sequence with an interval of 1 second, while the Y-axis represents the number of data bytes transmitted per second. As shown in the figure, video streams with the same title display similar BRS patterns during playback, while the BRS patterns differ when different videos are played.



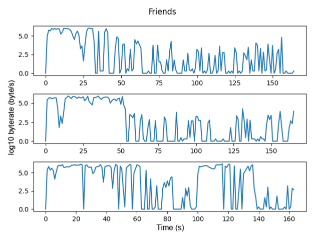
(a) V1: Coolio Gangsters Paradise



(b) V2: Avicii Waiting



(c) V3: Warcraft



(d) V4: Friends

Fig. 3: Example BRSs for encrypted video streams with different titles (V1-4). For ease of visualization, the Y-axis values are taken as base 10 logarithms. The figure shows that each video source exhibits its own unique pattern on BRS.

B. Design of Embedding Model

Representation learning provides a useful approach to derive embedding results from the BRS features. However, three key challenges should be addressed to effectively analyze encrypted video streams. (1) The indefinite-length sequence data needs to be mapped into low-dimensional Euclidean space. The embedding results should (2) retain as much sequence information as possible and (3) exhibit distinguishability in terms of video content.

To tackle these challenges, we propose a network structure, shown in Fig. 4, which serves as the embedding function $G(\cdot)$. (1) The output state of RNN is used to convert the indefinite-length BRS into a low-dimensional vector. (2) The representation of BRS information can be enhanced by incorporating a self-attention mechanism within the network. In the training process, (3) we apply contrastive learning to ensure the embedding results have content-related distinguishability. (It is explained in more detail in Section III-C of the paper.)

Then, we will elaborate on the details of the function $G(\cdot)$.

*1) **Mapping of BRS:*** The input of $G(\cdot)$ is a sequence with indefinite length, that is, a BRS generated by video playback. For a BRS input $X = \langle x_1, x_2, \cdots, x_T \rangle$, its elements x_i will be fed into the RNN module one by one. The RNN cell will simultaneously output $\boldsymbol{o}_i$ and hidden $\boldsymbol{h}_i$ state:

$$\boldsymbol{o}_i, \boldsymbol{h}_i = RNNcell(x_i, \boldsymbol{h}_{i-1}) \quad i = 1, 2, \ldots, T \tag{1}$$

where o_i consist of the output sequence $O = \langle \boldsymbol{o}_1, \boldsymbol{o}_2, \cdots, \boldsymbol{o}_T \rangle$. The output sequence can be calculated as follows:

$$O = RNN(X) \tag{2}$$

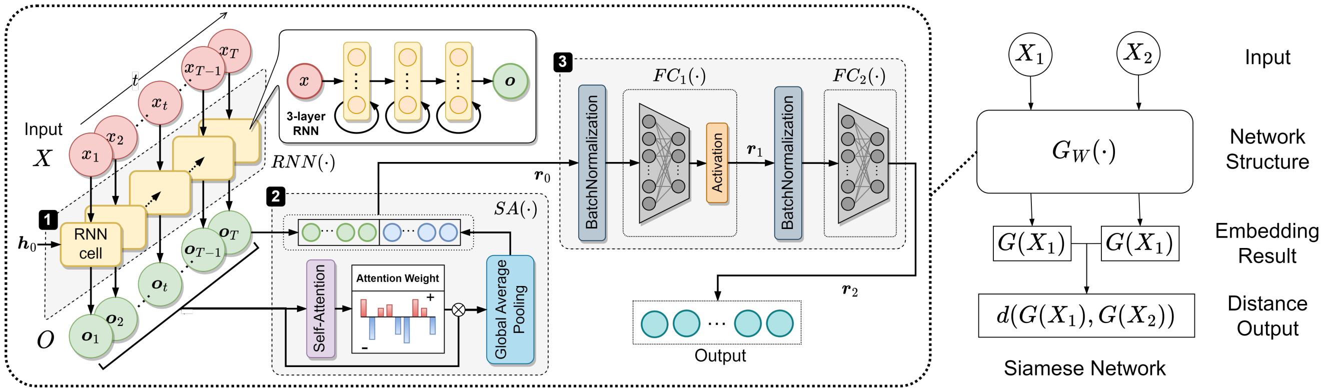


Fig. 4: The network structure of the embedding model based on RNN and self-attention mechanism.

The final output of the RNN cell is $\boldsymbol{o}_T$, which represent the original sequence data to some extent. In addition, through padding and masking techniques, the RNN module can effectively handle variable-length BRS in each training batch. As for the RNN module, we use a 3-layer RNN network, where each layer contains 32 RNN cells. The RNN cell here can be LSTM or GRU. Therefore, the output item $\boldsymbol{o}_i$ is a 32-dimension vector. (That is, $d_k = 32$.)

*2) **Enhancement of Representation**:* Considering the vanishing gradient problem of the RNN module, we propose to add a self-attention mechanism to the output sequence O in order to enhance the extraction ability of sequence information. We used scaled dot-product attention to implement the self-attention. It can produce an attention score matrix, which represents the relative importance of each element. Its output $\mathbf{O}^{att}$ is a $d_k \times T$ matrix that contains global information of output sequence O, which is calculated as follows:

$$\mathbf{O}^{att} = \mathbf{V}\text{Softmax}(\frac{\mathbf{K}^\top \mathbf{Q}}{\sqrt{d_k}}) = \mathbf{W}_v O \text{Softmax}(\frac{O^\top \mathbf{W}_k^\top \mathbf{W}_q O}{\sqrt{d_k}}) \tag{3}$$

Then, Global Average Pooling (GAP) is applied to sum out the spatial information of each output dimension, which makes it more robust. Its output is also a 32-dimension vector. We concatenate the output and o^T as follows:

$$\boldsymbol{r}_0 = [GAP(\mathbf{O}^{att})^\top || \boldsymbol{o}_T^\top]^\top = SA(O) \tag{4}$$

*3) **Dimensionality reduction**:* Fully connected layer is also need to compress the information of the output cells. We add two layers (FC_1 and FC_2) to compress the output into low-dimensional vector. The first fully connected layer is $FC(64, 32, \text{ReLU})$ including a ReLU activate layer, which is expressed as follows:

$$\boldsymbol{r}_1 = \text{ReLU}(\mathbf{W}_1 \boldsymbol{r}_0 + \boldsymbol{b}_1) = FC_1(\boldsymbol{r}_0) \tag{5}$$

The second fully connected layer is $FC(32, 8)$ without activate layer, which is expressed as follows:

$$\boldsymbol{r}_2 = \mathbf{W}_2 \boldsymbol{r}_1 + \boldsymbol{b}_2 = FC_2(\boldsymbol{r}_1) \tag{6}$$

So far, the indefinite-length BRS sequence generated by an encrypted video stream is embedded in the 8-dimensional Euclidean space. In addition, Batch Normalization (BN) layer is often set to avoid overfitting. Therefore, we also add two BN layers before FC_1 and FC_2.

The embedding function $G(\cdot)$ can also be expressed in the form of a composite function:

$$\boldsymbol{r}_2 = (FC_2 \circ FC_1 \circ SA \circ RNN)(X) = G(X) \tag{7}$$

C. Training of Embedding Model

In the training process, we apply the siamese network to train the embedding function $G(\cdot)$. Siamese network introduces the idea of contrastive learning into the training process of the embedding function so that the embedding results have content-related distinguishability. Unlike supervised learning, contrastive learning uses label similarity instead of original labels during training, leading to trained embedding models that are better suited for open-world scenarios.

The siamese neural network is composed of two twin networks whose output is jointly trained. There is a function above to learn the relationship between input data sample pairs. The two networks share the same weight and network parameters. As shown in Fig. 4, while working in tandem on two different input vectors X_1 and X_2, the siamese neural network G_W uses the same weights W and computes distance output. Weight tying guarantees that two extremely similar inputs could not possibly be mapped by their respective networks to very different locations in feature space because each network computes the same function.

During the training process, we randomly select a pair of BRSs from the training set and input them into the embedding model. The label data (y) indicates whether the selected BRS pair comes from the same video title. We apply the L2-norm to measure the distance of the embedding vectors:

$$D_W = d(G(X_1), G(X_2)) = ||G(X_1) - G(X_2)||_2 \tag{8}$$

The loss function applied in siamese network is contrastive loss, which is defined as follows:

$$\mathcal{L} = \mathcal{L}_s + \mathcal{L}_d = \frac{1-y}{2} D_W^2 + \frac{y}{2} \max\{0, m - D_W\}^2 \tag{9}$$

where m is called margin value. In this loss function, $\mathcal{L}_s$ reduce the distance between encrypted video streams of the same title as much as possible, while $\mathcal{L}_d$ control the distance between encrypted video streams of different titles around 1.

The trained embedding model is capable of transforming encrypted video streams into a low-dimensional representation within a Euclidean space. As a result, encrypted video streams with similar content will be mapped to close proximity, while those with differing content will be positioned further apart.

IV. Downstream Analysis Based on EVS2vec

In this section, we introduce four downstream analysis tasks based on EVS2vec, including thresholding, fingerprinting, classification, and clustering.

A. Threshold Discrimination

For the analysis of a limited number of encrypted video streams, we usually need to judge the homology, that is, whether the two unknown encrypted video streams belong to the same video title. This analysis task requires a threshold.

The random variable X_1 and X_2 indicate the embedding distance of two video streams belonging to the same title and different title. Their probability of density function are $f_1(\cdot)$ and $f_2(\cdot)$, and their probability distribution functions are $F_1(\cdot)$ and $F_2(\cdot)$, respectively. For a probability distribution, its probability distribution function is the primitive function of its probability density function. Then, the true positive rate (TPR) and the false positive rate (FPR) can be expressed as follows:

$$TPR = \frac{TP}{TP+FN} = F_1(x) \quad FPR = \frac{FP}{FP+TN} = F_2(x) \tag{10}$$

A Receiver Operating Characteristic (ROC) curve is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. The ROC curve is created by plotting the TPR against the FPR at various threshold settings. Youden's Index (J) [15] is a single statistic that captures the performance of a dichotomous diagnostic test, which is often used in conjunction with ROC analysis.It is related to sensitivity ($\frac{TP}{TP+FN}$) and specificity ($\frac{TN}{TN+FP}$), which is defined as follows:

$$J = sensitivity + specificity - 1 = TPR - FPR \tag{11}$$

Youden's Index is defined for all points of an ROC curve, and the maximum value of the index may be used as a criterion for selecting the optimum threshold for binary classification. To maximize it, the threshold can be calculated as follows:

$$Threshold = \arg\max_{x\in\mathbb{R}^+} J = \arg\max_{x\in\mathbb{R}^+} F_1(x) - F_2(x) \tag{12}$$

B. Fingerprint Construction

Based on the embedding technology of encrypted video streams, we can generate fingerprints for specific videos.

For a encrypted video stream set of the same video title $\mathcal{S} = \{\boldsymbol{s}_1, \boldsymbol{s}_2, \cdots, \boldsymbol{s}_m\}$, we need to convert $\boldsymbol{s}_i$ to BRS first. $\boldsymbol{s}_i$ is a packet set that sniffed when video playback. Then, input it into the neural network function $G(\cdot)$, and get the embedding result. Once the embedding result of each stream is obtained, the average embedding result of $\mathcal{S}$ can be calculated and regarded as the fingerprint of the video. The encrypted video stream fingerprint can be computed as follows:

$$\boldsymbol{f} = \frac{1}{m}\sum_{i=1}^{m} G(BRS(\boldsymbol{s}_i)) \tag{13}$$

where $BRS(\cdot)$ is a function that converts encrypted video streams into BRSs (Algorithm 1).

For an unknown encrypted video stream $\boldsymbol{s}$, we need to calculate its embedding result $G(BRS(\boldsymbol{s}))$. After that, compute the Euclidean distance between the embedding result and each fingerprint. According to the Maximum Likelihood Estimate (MLE), the judgement result for the unknown encrypted video stream is the video to which the fingerprint of its nearest neighbor belongs. In addition, this shortest distance needs to be compared to the threshold we calculated before. If the shortest distance $(\min ||\boldsymbol{f} - G(BRS(\boldsymbol{s}))||_2)$ is still greater than the threshold, the unknown encrypted video stream is considered to not belong to any known video.

C. Classification and Clustering

Classification and clustering based on the embedding vectors of encrypted video streams follow a similar process. The first step is to input the BRS data into the network structure $G_W(\cdot)$ to obtain the set of embedding vectors.

For classification tasks, the input data to the embedding model is the training data. The embedding vectors are then used to train a classifier such as k-NN, SVM, Random Forest, XGBoost, etc. The goal of this training is to improve the identification accuracy by making the classifier better suited to the spatial structure of the embedding vectors.

For clustering tasks, the data input to the embedding model is the encrypted video streams to be analyzed. The distance between the embedding vectors of encrypted video streams is related to their video content, making is the basis of cluster analysis. Common distance-based clustering algorithms include K-means, agglomerative clustering, GMM (Gaussian Mixture Model), etc.

V. Experimental Evaluation

In this section, we describe our experimental setup and the dataset used to evaluate the effectiveness of EVS2vec. We then perform various experiments, including thresholding, fingerprinting, classification, and clustering, to assess the efficiency.[1]

A. Experimental Setup

To evaluate our model, we conducted preliminary experiments. Our experiments are performed on a workstation, which is equipped with the following configuration: Intel(R) Core(TM) i9-10900K CPU @ 3.70GHz, NVIDA GeForce RTX 3090, 128GB of RAM.

The RNN cell applied in the embedding model is 3-layer GRU. In the siamese network, the margin value of the loss function is set to 1. All the neural network structures are

[1]The experimental source code of our implementation is available at https://github.com/David108/EVS2vec.

implemented by Tensorflow (version 2.9.0) [16] and trained by the Adam optimization algorithm with a learning rate of 0.001. The batch size is set to 256, that is, 256 pairs of BRS of encrypted video streams are randomly selected from the training data each time. We call the training of 100 batches an epoch, for a total of 200 epochs. During training, the model parameters that perform best on the test set will be saved.

B. Dataset

In this study, we utilize a public dataset developed by Dubin et al. that was previously utilized in [3]. This dataset contains 10,000 YouTube streams, each consisting of 100 video titles (100 streams per title). These video streams were collected via real-world Internet connections over different network conditions using a Selenium web automation tool with ChromeDriver. The Chrome browser was used to collect the video traffic due to its widespread popularity. The video titles in the dataset are popular YouTube videos from different categories such as sports, news, nature, etc.

The dataset is divided into a training set and a test set, with a ratio of 1:1. The training set consists of video streams from 50 titles, while the remaining video streams make up the test set. The embedding model for training is fed with the BRS pairs of encrypted video streams from the training set. The ratio of BPS pairs from the same video and different videos is controlled to be 1:1.

C. Downstream Task 1: Thresholding

In the experiment, the ROC threshold is applied to determine whether two unknown encrypted video streams belong to the same video title. We analyze the distance distribution of the sample pair and select the optimal threshold. Then, we evaluate its ability to distinguish between different video titles in terms of the ROC threshold accuracy.

Experimental Setting: From the test set of encrypted video streams, we selected 100,000 stream pairs, with roughly equal proportions of stream pairs from the same video and different videos. We then processed each stream pair by converting the encrypted video streams into BRSs and inputting them into our network structure to obtain their embedding results. Finally, we calculated the distance between each stream pair based on the embedding results.

Result: As shown in Fig. 5, the results of our experiments demonstrate a clear distinction between the distance distributions of encrypted video streams from the same video title and those from different video titles. After statistical analysis, the calculated ROC threshold is 0.30835 (the probability is 96.707%). It indicates a 96.707% probability of correctly determining whether two encrypted video streams are from the same video source.

D. Downstream Task 2: Fingerprinting

Based on the EVS2vec, any video title can have their own unique video fingerprints constructed. We demonstrate the effectiveness of fingerprints through a set of experiments.

Experimental Setting: We randomly selected 10 video titles as an instance. Then, we applied EVS2vec to each

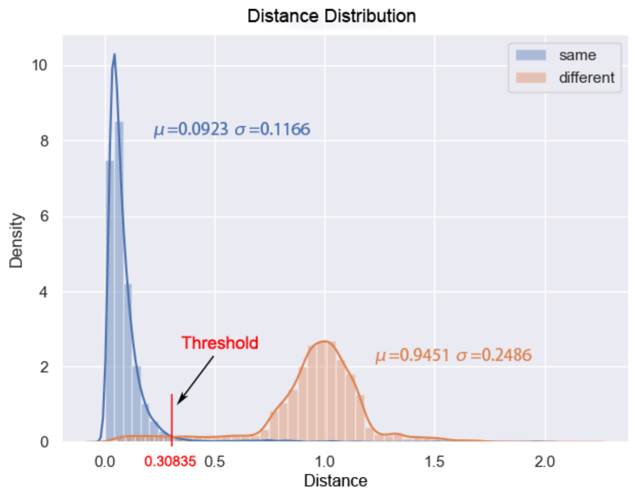


Fig. 5: The distance distribution of embedding results.

encrypted video stream and constructed an 8-dimensional video fingerprint for each video title. To validate the effectiveness of this fingerprint construction scheme, we compared the similarity between the fingerprints and the corresponding samples in the embedding space.

Result: The fingerprint construction based on EVS2vec relies on the spatial separability of the embedding results. It means that the embedding vectors of the same video should be clustered together, while those of different videos should be separated. To validate this, we utilized the t-distributed Stochastic Neighbor Embedding (t-SNE) to visualize the embedding vectors in a two-dimensional space. As shown in Fig. 6a, it is evident that the encrypted video streams of different titles can be well distinguished after embedding.

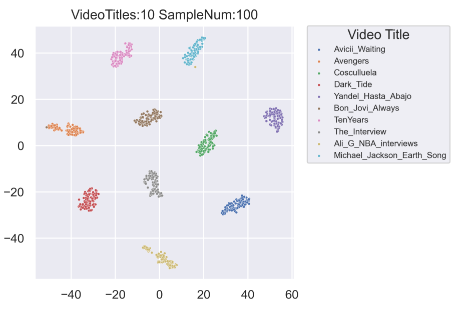
The embedding results of selected samples are shown in Fig. 6. Each video title has 30 samples. We generate the fingerprint for each videoFig. 6b displays the sample embedding results of encrypted video streams for each title in the form of a heat map. The fingerprints shown in Fig. 6c correspond to the embedding results of selected samples. The similarity between the two figures demonstrates the effectiveness of the fingerprinting method for encrypted video streams.

E. Downstream Task 3: Classification

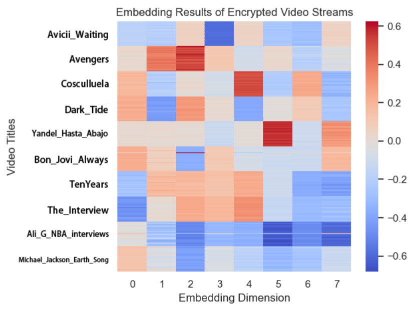
The EVS2vec can also be applied in supervised learning. We set experiments to test the classification results based on embedding results of the encrypted video streams.

Experimental Setting: In our experiments, half of the EVS2vec embedding results were utilized to train various classifiers including well-known machine learning algorithms such as k-NN, Ridge Regression, SVM, and Decision Tree. In addition, we also employed advanced ensemble learning methods including Random Forest and XGBoost. The remaining half of the embedding results were used for testing.

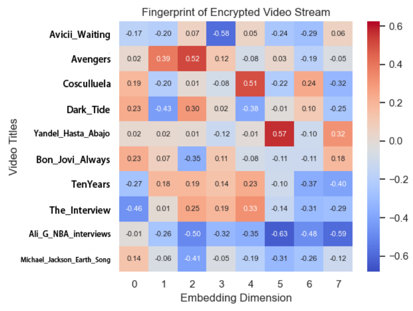
Result: The results of the classification accuracy can be seen in Table I. It demonstrates that the ensemble classifiers can achieve higher identification accuracy (over 95%). Among



(a) t-SNE Visualization



(b) Embedding Results



(c) Fingerprint

Fig. 6: The examples of video fingerprints based on EVS2vec.

traditional machine learning algorithms, the SVM produces the best results. Table I also provides the statistical results of F_1 values for different video titles. Both ensemble classifiers achieved good identification results for all video titles, with all F_1 values surpassing 0.8. The Random Forest classifier even achieved exceptional results, with all F_1 values exceeding 0.9. These results indicate that the classifier trained through supervised learning is better adapted to the spatial structure of the embedding results.

TABLE I: The metrics of classification based on EVS2vec using different algorithm.

Classifier	F_1 **score** ≥ 0.80	≥ 0.90	≥ 0.95	**Accuracy**
k-NN	17	10	6	0.6958
RidgeRegression	34	11	1	0.8140
DecisionTree	44	31	17	0.9085
SVM	48	41	21	0.9269
XGBoost	50	45	32	0.9554
RandomForest	50	50	40	0.9739

F. Downstream Task 4: Clustering

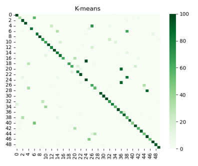
The results of the embedding model can also be applied to clustering analysis. To demonstrate this, we conduct clustering experiments using popular algorithms.

Experimental Setting: We utilized the entire test set consisting of 50 video titles, each with 100 encrypted video streams. Before performing clustering, the embedding results of all encrypted video streams were calculated. We evaluated the performance of three clustering algorithms including k-means, agglomerative clustering, and GMM.

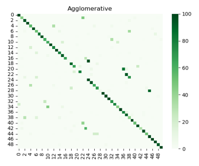
Result: Table II illustrate the results of clustering experiments. Approximately 20-30% of video titles can be accurately clustered ($F_1 > 0.95$). In terms of accuracy score, the agglomerative algorithm is found to have the best clustering performance. The confusion matrices of the clustering results are indicated in Fig .7. The confusion matrices indicate that most video titles can be effectively clustered, demonstrating the effectiveness of EVS2vec in the task of clustering encrypted video streams.

TABLE II: The accuracy of clustering.

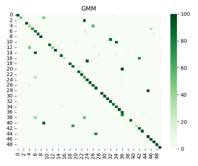
Clustering	F_1 **score** ≥ 0.80	≥ 0.90	≥ 0.95	**Accuracy**
K-means	22	16	12	0.6969
Agglomerative	24	17	13	0.7331
GMM	23	18	12	0.6703



(a) K-means



(b) Agglomerative



(c) GMM

Fig. 7: The confusion matrices of clustering.

G. Robustness Test

In order to test the robustness of the embedding model, we examine their ROC accuracy of threshold discrimination under a certain noise.

Experimental Setting: A positive random variable X represents the Bytes per second of traffic noise. We assume that the traffic noise follows a Rayleigh distribution, that is, $X \sim \text{Rayleigh}(\sigma)$ where σ is the scale parameter of the distribution. Its probability density function is

$$f(x;\sigma) = \frac{x}{\sigma^2}\exp(-\frac{x^2}{2\sigma^2}), \quad x \geq 0. \tag{14}$$

The mean and the standard deviation of the Rayleigh distribution are $\sqrt{\frac{\pi}{2}}\sigma$ and $\sqrt{\frac{4-\pi}{2}}\sigma$, respectively.

We test the accuracy of ROC threshold under Rayleigh noise with different means (μ). In the experiments, all embedding models contain a 3-layer RNN. The value set of μ is $\{10\text{B/s}, 100\text{B/s}, 1\text{KB/s}, \cdots, 100\text{MB/s}, 1\text{GB/s}\}$.

Result: We apply heatmap to show the effect of noise with different means on the ROC accuracy of embedding models. As evident from Fig. 8, the embedding models with self-attention mechanism is the most robust. When the mean noise is less than 100MB/s, Att-LSTM is almost unaffected

TABLE III: The comparison with existing works.

Existing Work	[9], [10]	[3], [6]	[4]	[17]	[18]	[19]	[5], [20]	[11]	[8]	[7]	[12]	[13], [14]	Our Work
Threshold							✓				✓		✓
Fingerprint	✓	✓	✓	✓	✓			✓				✓	✓
Classification		✓	✓	✓	✓	✓	✓		✓			✓	✓
Clustering										✓	✓		✓
Analysis Unit	Video Fragment	BPPs	Burst	Flow Feature	Packet Sequence	Flow Feature	Bit Rate	Video Fragment	Burst	BPPs	Video Fragment	Video Fragment	BRS
Base Model	KD-Tree	k-NN	1D-CNN	—	MLP/CNN /LSTM	RF/GDBT	P-DTW	—	k-NN	AE	Markov Chain	Levenshtein Distance	EVS2vec

by Rayleigh noise and maintain about 95% discrimination accuracy of ROC threshold. Att-GRU also has acceptable resistance to noise. It is remarkable that they can resist the vast majority of network noise in the real world.

On the contrary, embedding models without self-attention mechanisms are more sensitive to low mean noise. Their discrimination accuracy of ROC threshold will shown a trough in the noise range of 10B/s-1KB/s. Experimental results illustrated that the addition of the attention mechanism in the embedding models can effectively improve the model's resistance to network noise.

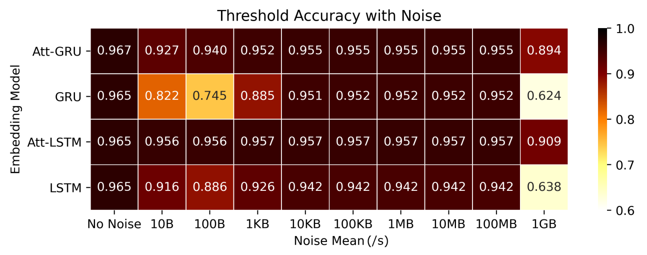


Fig. 8: The threshold accuracy of embedding models under Rayleigh noise with different means.

VI. Discussion

To the best of our knowledge, there is no existing work that implements encrypted video stream analysis through low-dimensional embedding. Currently, the mainstream approach is classification based on supervised machine learning. As described in Table III, we compare EVS2vec existing works and discuss the advantages as follows:

- Our method uses BRS as the analysis unit which can be directly obtained during network sniffing without the need to capture full network traffic data.As a result, complex computational processes such as TCP reassembly [11]–[14] can be avoided, reducing computational complexity.
- Compared with existing end-to-end methods based on deep learning [4], [18], our method only provides a normalized representation of encrypted video streams that is not limited to a specific analysis task or a known label set, allowing for greater scalability in analysis.
- Analysis methods based on traditional features [17], [19] require manual feature extraction of network flow. Our method uses representation learning based on RNN and self-attention mechanism to automate feature extraction, improving the representation ability.
- Compared with existing method based on sequence alignment [5], [13], [14], [20], EVS2vec maps encrypted video streams from sequential space to low-dimensional Euclidean space, making fingerprint construction easier. Consequently, our method is more suitable for many types of downstream analysis tasks.

While this work presents a promising approach to encrypted video stream analysis, there are still some limitations to be addressed. Firstly, the training process of the embedding model only uses rough similarity information in the form of matching video titles. The semantic information analysis of the embedding results is also limited. Last but not least, the training data for EVS2vec is restricted to one website, limiting the ability to perform cross-platform analysis of encrypted video streams. Other scenarios, such as online video calls or conferences that use the UDP protocol, also remain to be studied and discussed.

For the future work, researchers should explore incorporating additional information related to the video content to further improve the EVS2vec model. It would be beneficial to delve into the semantic information contained within the embedding results, which would enhance the interpretability of the model. To further improve accuracy and generalization, the embedding model in EVS2vec should be trained on a larger, cross-platform encrypted video stream dataset.

VII. Related Work

Encrypted video stream analysis is a vital area of research in the field of encrypted traffic analysis. Due to the VBR, the encoded size of video fragments is directly related to the complexity of the video content. Consequently, many video streams are uniquely characterized by their transmission patterns. [4]. BPP (Bit-per-Peaks) [3], [6], [8], burst [4], [7], video segments [9]–[14], packet sequence [21], [22] and bitrate sequence [5], [20] have been applied in the analysis of encrypted video streams. These works are essentially extracting the time series features of encrypted video streams and mining content-related information by machine learning algorithms. Dubin et al. [3], [6] presented several algorithms based on SVM (Support Vector Machine) and k-NN (k-Nearest Neighbors) for encrypted HTTP adaptive video streaming title classification and achieved high classification accuracy. Liu et al. [8] proposed a video flow identification method based on the intermittent traffic pattern-related features and the k-NN algorithm, called ITP-KNN. With the development of deep learning, many neural network structures have also been applied in encrypted video traffic analysis. Roei Schuster [4] classifiers based on CNN (Convolutional Neural Network) can accurately identify these patterns given very coarse network measurements. Inspired by Natural Language Processing, Dvir et al. [7] exhibited a clustering method for encrypted video streams based on Deep Encoder-based feature embedding algorithms. Shi et al. [22] converted each stream feature to a letter in the English alphabet, and then applied DNN-based classification methods.

In addition, some encrypted video stream identification methods [17]–[19] are based on traditional statistical features of flow rather than series features of encrypted video streams. Li et al. [17] presented a real-time, lightweight video classification method suitable for ISP middle-boxes, named Silhoustte, which used only flow statistics for tasks, making it payload-agnostic and effective. Garcia et al. [19] tested the identification performance of ensemble classifiers under massive encrypted video streams. Considering the development in the real environment, they pay more attention to the balance between the running performance and the identification effect.

VIII. Conclusion

The proposed EVS2vec method provides an effective solution for content analysis of encrypted video streams, overcoming the challenges of label limitations and complex calculations. Our method generates BRS for each encrypted video stream and uses its embedding result to represent the original traffic data. The embedding model, using RNN with self-attention mechanism and trained by siamese network, ensures content-related distinguishability. The experimental results also demonstrated its effectiveness. The embedding results of EVS2vec support many downstream analysis tasks of encrypted video streams, including such as classification and clustering threshold discrimination, fingerprint construction, classification, and clustering analysis.

Acknowledgment

This work is supported by National Natural Science Foundation of China (No.62072466, No.62102429, No.52177067, No.62102422, No.62102450, No.61872372, No.62102430,), Natural Science Foundation of Hunan Province, China(Grant No. 2021JJ40688), Science Research Plan Program by NUDT (Grant No.ZK22-50), and the Graduate Research and Innovation projects in Hunan Province (CX20210008).

References

[1] U. Cisco, “Cisco annual internet report (2018–2023) white paper,” *Cisco: San Jose, CA, USA*, 2020.

[2] T. Stockhammer, “Dynamic adaptive streaming over http– standards and design principles,” in *Proceedings of the second annual ACM conference on Multimedia systems*, 2011, pp. 133–144.

[3] R. Dubin, A. Dvir, O. Pele, and O. Hadar, “I know what you saw last minutełencrypted http adaptive video streaming title classification,” *IEEE transactions on information forensics and security*, vol. 12, no. 12, pp. 3039–3049, 2017.

[4] R. Schuster, V. Shmatikov, and E. Tromer, “Beauty and the burst: Remote identification of encrypted video streams,” in *26th USENIX Security Symposium (USENIX Security 17)*, 2017, pp. 1357–1374.

[5] J. Gu, J. Wang, Z. Yu, and K. Shen, “Walls have ears: Traffic-based side-channel attack in video streaming,” in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 2018, pp. 1538–1546.

[6] R. Dubin, A. Dvir, O. Hadar, and O. Pele, “I know what you saw last minute-the chrome browser case,” *Black Hat Europe*, 2016.

[7] A. Dvir, A. K. Marnerides, R. Dubin, N. Golan, and C. Hajaj, “Encrypted video traffic clustering demystified,” *Computers & Security*, vol. 96, p. 101917, 2020.

[8] Y. Liu, S. Li, C. Zhang, C. Zheng, Y. Sun, and Q. Liu, “Itp-knn: Encrypted video flow identification based on the intermittent traffic pattern of video and k-nearest neighbors classification,” in *International Conference on Computational Science*. Springer, 2020, pp. 279–293.

[9] A. Reed and B. Klimkowski, “Leaky streams: Identifying variable bitrate dash videos streamed over encrypted 802.11 n connections,” in *2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2016, pp. 1107–1112.

[10] A. Reed and M. Kranch, “Identifying https-protected netflix videos in real-time,” in *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy*, 2017, pp. 361–368.

[11] H. Wu, Z. Yu, G. Cheng, and S. Guo, “Identification of encrypted video streaming based on differential fingerprints,” in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2020, pp. 74–79.

[12] L. Yang, S. Fu, Y. Luo, and J. Shi, “Markov probability fingerprints: A method for identifying encrypted video traffic,” in *2020 16th International Conference on Mobility, Sensing and Networking (MSN)*. IEEE, 2020, pp. 283–290.

[13] L. Yang, Y. Zeng, S. Fu, and Y. Luo, “Unsupervised analysis of encrypted video traffic based on levenshtein distance,” in *International Symposium on Security and Privacy in Social Networks and Big Data*. Springer, 2020, pp. 97–108.

[14] L. Yang, S. Fu, and Y. Luo, “A clustering method of encrypted video traffic based on levenshtein distance,” in *2021 17th International Conference on Mobility, Sensing and Networking (MSN)*. IEEE, 2021.

[15] W. J. Youden, “Index for rating diagnostic tests,” *Cancer*, vol. 3, no. 1, pp. 32–35, 1950.

[16] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, “Tensorflow: a system for large-scale machine learning,” in *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, 2016, pp. 265–283.

[17] F. Li, J. W. Chung, and M. Claypool, “Silhouette: Identifying youtube video flows from encrypted traffic,” in *Proceedings of the 28th ACM SIGMM Workshop on Network and Operating Systems Support for Digital Audio and Video*, ser. NOSSDAV ’18. New York, NY, USA: Association for Computing Machinery, 2018, p. 19C24.

[18] Y. Li, Y. Huang, R. Xu, S. Seneviratne, K. Thilakarathna, A. Cheng, D. Webb, and G. Jourjon, “Deep content: Unveiling video streaming content from encrypted wifi traffic,” in *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*. IEEE, 2018, pp. 1–8.

[19] J. Garcia, T. Korhonen, R. Andersson, and F. Västlund, “Towards video flow classification at a million encrypted flows per second,” in *2018 IEEE 32nd International Conference on Advanced Information Networking and Applications (AINA)*. IEEE, 2018, pp. 358–365.

[20] J. Gu, J. Wang, Z. Yu, and K. Shen, “Traffic-based side-channel attack in video streaming,” *IEEE/ACM Transactions on Networking*, vol. 27, no. 3, pp. 972–985, 2019.

[21] Y. Shi and S. Biswas, “A deep-learning enabled traffic analysis engine for video source identification,” in *2019 11th International Conference on Communication Systems & Networks (COMSNETS)*. IEEE, 2019, pp. 15–21.

[22] Y. Shi, D. Feng, Y. Cheng, and S. Biswas, “A natural language-inspired multilabel video streaming source identification method based on deep neural networks,” *Signal, Image and Video Processing*, vol. 15, no. 6, pp. 1161–1168, 2021.