# Based on deep-learning apporach to real-time identify traffic of VoIP applications

*Note: Sub-titles are not captured in Xplore and should not be used


1st Given Name Surname
*dept. name of organization (of Aff.)*
*name of organization (of Aff.)*
City, Country
email address

2nd Given Name Surname
*dept. name of organization (of Aff.)*
*name of organization (of Aff.)*
City, Country
email address

3rd Given Name Surname
*dept. name of organization (of Aff.)*
*name of organization (of Aff.)*
City, Country
email address

4th Given Name Surname
*dept. name of organization (of Aff.)*
*name of organization (of Aff.)*
City, Country
email address

5th Given Name Surname
*dept. name of organization (of Aff.)*
*name of organization (of Aff.)*
City, Country
email address

6th Given Name Surname
*dept. name of organization (of Aff.)*
*name of organization (of Aff.)*
City, Country
email address



***Abstract*—With their high service quality and low price cost, VoIP applications win most of the users' favor. However, owing to the flexible of VoIP services, it will causes incalculable damage if used improperly. In order to make VoIP applications serve human better, it is important to keep VoIP applications under supervision. The upgrading of VoIP technology makes the traditional identification method inefficient, it becomes more difficult to identify VoIP traffic. For the challenge of identifying VoIP traffic, the deep learning technology under the era of big data provides a new solution. With supervised learning method, we can build various feature libraries for the categories of VoIP traffic. This way is not only able to find the most useful features, but also it can get rid of human effort in exploring features. In this paper, we adopt CNN(Convolutional Neural Networks) to extract features for accurate VoIP application identification. In addition, we design a real-time identification system to capture VoIP traffic in a large-scale network and identify their application types with the features we trained. The evaluation results verify that our system can identify VoIP traffic timely and accurately.**



***Index Terms*—VoIP, application identification, CNN, real-time.**


## I. Introduction

VoIP(Voice over Internet Protocol) is an internet technology for the delivery voice communications and multimedia sessions over IP networks, it provides an environment supports connection of all kinds of terminals via internet. Recently, with their price advantage, VoIP applications have been used more and more among the people. Most of VoIP applications can provide convenient services, such as anonymous calls, multi-line calls, call forwarding. They are so flexible that many unruly elements use VoIP services make fraud calls. The results of VoIP identification researches can help network administrators to enhance network security, for this reason, VoIP identification based on application type has been an active research topic.

VoIP consists of the protocols that used in signaling and channel setup. H.323, SIP, MGCP and MEGACO are the most commonly used protocol for call signaling and controlling, RTP/RTCP are the most commonly used transmission protocol. Some researches [1]–[3] are devoted to analyze the behavior of the VoIP traffic generated in signaling phase, some [4], [5] are devoted to analyze the behavior of the voice traffic that RTP carries. There are also some large manufacturers using their own VoIP protocol, such as the Skype protocol. Therefore, there are some researches [6]–[8] specifically identify Skype traffic. Unfortunately, encryption technologies such as SSL/TLS, SIPS, WEP and WAP/WAP2 may be used to encrypt signaling connections, when SRTP/SRTCP may be used to encrypt voice transmissions. The utilization of encryption technology has an impact on both identification directions, identification task of VoIP traffic is getting harder and harder.

In this paper, we aim to identify the flows used for voice transmission rather than signaling. Encryption technologies are so ripe that most VoIP applications already adopt them into signaling phase. Compared to signaling phase, SRTP/SRTCP require more computing resources than RTP/RTCP, considering calculation ability and transmission quality, most of the existing VoIP applications still adopt RTP/RTCP in voice transmission phase. For encrypted voice traffic that few VoIP applications generate, we still can extract more features than signaling traffic because of there are more traffic generated in voice transmission phase. For these reasons, we adopt CNN to extract features for flows generated during the voice transmission phase. Considering identification real-time performance, we separate the VoIP entire flows into sub flows to train identification model.

Our contributions of this paper are summarized as fol-

low: 1) we adopt CNN to extract features for voice flows, these extracted features are not only more reliable than the features extracted by humans, but also greatly improve the identification efficiency. 2) we can identify VoIP flows in an acceptable time, we train identification model with sub flows instead of entire flows, it is a necessary condition for real-time identification. 3) These extracted features can accurately identify the application types of VoIP flows, in backbone network, administrators can intercept VoIP traffic according to specify application type.

The remainder of this paper is structured as follows. In section II, we describes the previously published related work. In section III, there is a brief introduction to several methods used in this paper. Then we present the architecture of our real-time identification system in section IV. In sections V, VI and VII, we describe the dataset and model used for deep learning, and how to perform real-time identification. The performance evaluation is shown in section VIII. The conclusion is shown in section IX.

## II. Related Work

Traffic identification has been an active research topic in the past decade. The identification of VoIP has attracted the attention of many researches and there are lots of worthy results obtained.

Since early VoIP applications use the ports that IANA designated, some previous works identify VoIP traffic based on ports. With the development of VoIP applications, most of them use dynamic ports, the traditional identification method based on port numbers is not effective enough. Some researches [1], [2] identify specify protocol with their regular characteristics, such SIP, RTP. These kinds of works are too limited to identify private like Skype protocol. To overcome these difficulties, [9] uses DPI (Deep Packet Inspection) way to extract application-level signatures to detect VoIP applications. However, the above methods cannot identify VoIP applications adopt encryption technologies.

Statistical analysis method is another most frequently used method to detect VoIP traffic. Some characteristics such as packet-inter arrival time, packet size and rate of packet exchange are analyzed and used to detect VoIP traffic in [10]. Authors in [11] combine traffic flow statistic analysis with host behavior estimation. They set a parameter D to filter non-VoIP traffic and calculate the R by EL (large inter-packet time) and ES (small inter-packet time) to identify VoIP traffic. There are still some latest researches [12], [13] adopt rule-based and statistical analysis-based solution to detect VoIP traffic. These ways require many thresholds to be set for corresponding rules to identify VoIP traffic, it's powerless for a wide variety of VoIP applications. With the development of machine learning, this method has gained its new vitality. In [14], authors propose the machine learning approach to identify encrypted VoIP traffic. They build flow-based feature set and employ 3 learning algorithms for training dataset, the results show a good performance to classify VoIP traffic. Usually this type of statistical analysis methods aim at detecting VoIP traffic, which means identify VoIP traffic in VoIP/non-VoIP level. With machine learning method, it can achieve further classification in application level. The method proposed in [14] trains the classifier with entire flows, and it requires a lot of works to build feature set. So they hardly could be used for real-time identification.

There are several researches trying to identify VoIP traffic in real time. For detecting VoIP traffic quickly, the packets whose length in range of 60 to 150 bytes are treated as VoIP packets in [15]. This method only aims at improving VoIP service quality, it cannot be used for accurate VoIP identification. A method that identifies peer-to-peer VoIP sessions using entropy and codec properties is proposed in [4], they consider several applications and their corresponding codecs so that the classifier can identify a VoIP session with the specific codec it uses. Moreover, they focus on the relation between the different lengths of N packets and explore their level of heterogeneity using entropy. In order to guarantee the real-time characteristic of the classifier, they adopt a sliding window with a constant size of N packets to access the heterogeneity in real time. Same real-time identification method used in [16], authors extract the PSD (Packet Size Distribution) of the first few packets of the entire Bi-flow. Results show this method can identify Bi-flow quickly after its establishment. All such methods that real-time identify VoIP traffic are based on features of sub-flows, but the limited features is too powerless to identify VoIP traffic in a tiny time period.

Standing on the shoulders of prior researches and focusing on the difficulties we found, in this paper we try to extract features for huge amounts of data with deep learning technology. Based on the technology proposed in [17], we employed CNN model to help us extract features automatically. These features are generic for non-encrypted and encrypted VoIP applications. The identification system we build can preliminarily detect VoIP traffic based on some rules, then the classifier can classify VoIP traffic into accurate application type.

## III. Methodology

Most of the VoIP applications use RTP/RTCP to transfer data, some others use SRTP/SRTCP, QUIC, etc. For a large amount of similar voice packets, the ability to identify a single packet is low. In this paper, we extract features for sub-flows (we use k-packet to denote one in the following) which consists of several continuous VoIP packets from a VoIP flow. In real-time identification phase, we build a k-packet after a connection established and apply the features to identify the k-packet. In order to verify that our method is feasible, we did some basic researches from 2 aspects. 1) Whether the number of packets generated by VoIP applications in a limited time can support the identification. 2) Whether RTP/RTCP flows carry the features that support identification. We collected traffic and did analysis for 10 VoIP applications. According to our investigation, the number of packets generated by VoIP applications can reach tens or even hundreds in a second, which means we can use hundreds of packets for feature identification. Moreover, we investigate 2 features, PSD [16]

and codec [4]. For PSD, there are 8 VoIP applications (AltCall, Jumblo, Xlite, Zoiper UUCall, Eyebeam, ExpressTalk and Bria) have constant PSD for voice transmission, 2 VoIP applications (Skype and KCCall) use variable-length packets. The length number used by Skype is completely random but limited to a range. For codecs, different routes of VoIP connections will use different codecs. The codecs we found list in Table I, there are many subtypes for every codec. Except these 2 main features, We expect deep learning will extract more features such as inter-arrival time, total duration, etc. And more features cannot be observed by humans.

TABLE I
CODECS USED BY VOIP APPLICATIONS.

| Application | Codecs |
|---|---|
| Skype | dynamic (96-127), SILK, G.729, PCM A/U |
| UUCall | H.263, G.723, G.728 |
| KCCall | PCM A/U, ILBC, GSM, G.729 |
| Jumblo | PCM A/U, dynamic (118) |
| Zoiper | PCM A/U, GSM |
| Xlite | PCM A/U, iLBC, GSM, Speex |
| Eyebeam | PCM A/U |
| ExpressTalk | PCM A/U, Speex |
| Bria | PCM A/U, Speex, GSM |

We apply supervised learning method to train the dataset. Constructing and labeling dataset is troublesome for researchers. In order to get raw VoIP traffic, we deploy several computers installed with VoIP applications in real network. To label them accurately, we adopt some traffic capture tools which are process-based. They can help us get pure raw traffic for each application, so we can label traffic using the name of VoIP application. Next problem is how to construct our training dataset with raw traffic data and ensure the dataset is uniform as soon as possible. We design a sampling method that uses sliding window to select k-packet into training dataset, the next step is randomized in [1, k]. This method not only promises the dataset is uniform but also reduces the training works. In order to identify non-VoIP traffic, we collected non-VoIP flows including video flows and real-time game flows, etc. Actually, non-VoIP traffic can also be classified more detailed, in our works, they will all be classified as non-VoIP traffic.

In order to design a suitable CNN model for training phase, we studied several of the most popular deep learning models. We considered factors such as hardware requirements and time-consuming and so on, and based experimental results, we designed a 8-layer CNN model. The training is carried out on a machine with 24G RAM and a quad-core Intel processor of 3.6GHz. The machine is equipped with NVIDIA GeForce GTX 1070(8GB) to accelerate computing. CUDA Toolkit 8.0 is used in our experiment to support parallel computing.

Collecting raw traffic data, constructing and labeling dataset and training dataset are offline, they are the basis for identifying traffic online in real time. In real-time identification phase, we parse the {srcIP, dstIP, srcPort, dstPort} in IP and UDP header, the quad-tuple will be used to separate flows as a key. We generate a k-packet for each flow with their specific key, then we identify the k-packet with the offline identification model. For each flow with specific key, the identification result will be saved as the value of the key. The {key, value} is saved into our database, we can identify the following traffic using their quad-tuple.

## IV. ARCHITECTURE

In this section, we introduce the architecture of our real-time identification system. There are 2 main phases in our architecture, offline training and online identification. We introduce the architecture of these 2 phases separately. Our architecture is shown in Figure 1.

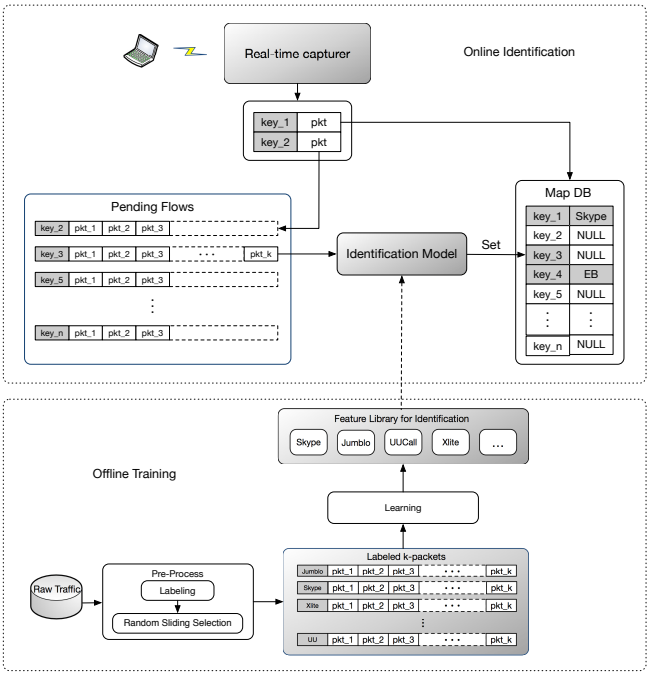


Fig. 1. Architecture of identification system.

### A. Offline Training

In offline training phase, we extract features for 11 kinds of flows, which include 10 kinds of VoIP application flows and non-VoIP flows. Our goal is to maximize the accuracy of training and validation data. There are mainly 2 procedures, data preprocessing and model training. In preprocessing, we label our dataset firstly, random select k-packets with sliding window secondly. In the procedure of converting k-packets into matrices, we remove IP header and UDP header, then add arrival timestamp for every packet. Then we input these matrices and label vector into CNN model. The training process is minimizing the loss function to find an optimal function for identification. The output of training process is the feature set that we will use in real-time identification phase. In evaluation procedure, we evaluate the identification model by calculating loss and accuracy for evaluation data which needs to be normalized too.

### B. Online Identification

Online identification is the final part of our system. In this phase, we build flows for traffic in large-scale network. We parse IP address and UDP port as its key for each packet. We maintain a map database for these keys. If the value of the corresponding key has been set, the value is the identification result of the packet. If the value is null, we search the key in pending flows database and add this packet into corresponding pending flow. When a pending flow is ready to be identified, it will be identified by identification model and set the identification result in map database.

As the Figure 1 shows, the number of pending flow key_3 reaches k, it is ready to be identified. We show 2 captured packets whose keys are key_1 and key_2, packet corresponding to key_1 will be identify with map database in real time, packet corresponding to key_2 will be appended into its pending flow.

## V. Traffic Collection and Processing

Traffic collection is the beginning of all of the works. In order to get raw traffic of 10 VoIP applications mentioned above, we deploy 8 computers installed 10 kinds of VoIP applications. We installed 6 computers with Windows operating system, the others 2 computers installed with Linux operating system, because there are only 4 applications can be installed on Linux platform. We can establish ordinary VoIP calls among them, and all of them can dial the phone number directly. There are 4 cell phones ready to be called. We try our best to make calls from various places in the country to improve the credibility of our experiment.

For the purpose of labeling data easier, we use the process-based capture tool QPA under Windows operating system; We use tcpdump to capture traffic based on port, which can be known after process occupied. QPA provide GUI application under Windows OS, it is easy to use for capturing traffic for a single process. Under Linux platform, we need to get pid of a process, then we use command like "tcpdump -i eth_name trans_protocol port pid" to capture traffic of a specific process. Because of process-based and port-based capture, it is easy to label raw traffic with their process name. Traffic from various video and game software will be labeled as non-VoIP traffic.

We need to construct our training dataset with captured raw traffic. We process the raw traffic to k-packets, we use sliding window to select k-packet into our dataset. The length of the window is k, and the next step is random. It ensures that our dataset isn't overly similar or vague. Figure 2 shows how we select 4 k-packets for $k = 10$. In this paper, a variety of identification models are trained according to different k, including 2, 4, 6, 8, 10, 20, 40 and 100. We have 4 steps to convert a k-packet to matrix. 1) Remove IP and UDP header; 2) Add 2 timestamps to each packet, one is the time difference from the first packet, the other one is the time difference from the previous packet; 3) Convert the k-packet to matrix on the basis of ASCII code; 4) Normalize the matrix. The length of VoIP packets is generally small around 50$\tilde{2}$10, so we set column of the matrix to 256. The row of the matrix is k, indicating a k-packet includes k packets.

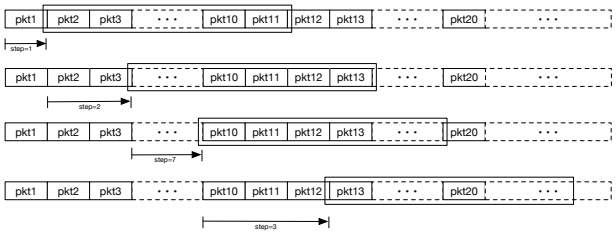


Fig. 2. How to process raw traffic to get k-packets(k=10).

## VI. Learning using Deep-Learning model

### A. Deep Learning model

With the rise of artificial intelligence technology, more and more excellent deep learning models appear. The oldest is CNN (Convolutional Neural Network) proposed by [18]. The LeNet is a development of the CNN proposed by [19]. [20] published an incredible neural network called AlexNet, this model get the best performance in the ImageNet competition held in 2012. [21] proposed VGG. The GoogleNet make a daring attempt in the design of the network, which proposed by [22]. [23] proposed ResNet, which has a deeper network structure up to 152 layers.

We did comprehensive consideration about accuracy, efficiency, hardware requirements, data set size and other conditions, we employed a 8-layer CNN model to train our data. Like many existing deep learning models, most of our ideas come from AlexNet. The CNN model includes 8 layers, consists of 5 convolution layers and 3 fully connected layers. We adopts ReLU (Rectified Linear Units) function as its activation function, and applies dropout in learning phase.

In our experiment, the inputs are matrices with size of k×l, where k is number of packets used for k-packet, l is the column of matrix, we use 256. We adjust the filter kernel to 5×5, and we change the padding type to same in each layer. The first 7 layers in the structure dropout the neural according to a probability of 0.5, the first 2 fully connected layers are activated using the ReLU activation function, and the last fully connected layer is activated using Softmax function. We only keep 1 max-pooling operation in the last convolution layer, the pooling window set to 2×2.

### B. Learning

We use Softmax function as the activation function of the last layer. There are 10 types of VoIP traffic in our experiment, we calculate the posterior probability for unknown VoIP traffic. Softmax function shows as

$$\hat{P}(V_i) = softmax(V_i) = \frac{e^{V_i}}{\sum_{j=1}^{n} e^{V_j}} \quad (1)$$

where $(V_1, V_2, \cdots, V_i, \cdots, V_n)$ denotes the matrix to be inputted into Softmax in the 8th layer;

$(\hat{P}(V_1), \hat{P}(V_2), \cdots, \hat{P}(V_i), \cdots, \hat{P}(V_n))$ is the output of the 8th layer, it is posterior probability distribution of k-packet, it is a vector with size of $1 \times n$; n is 10 in our experiment, it denotes the number of training VoIP application classes.

During the training, we train data to find the most optimal weights to maximize the likelihood estimate of k-packet. To minimize the CCE (categorical cross-entropy), we have the formula as follow,

$$CCE(V_i) = -\sum_{i=1}^{n} P(V_i) \times \log(\hat{P}(V_i)) \tag{2}$$

where $P(V_i)$ denotes the real probability $V_i$, it is the target matrix generated according to the labels corresponding to the training data. $P(V_j) = 1$, where j is the label corresponding to VoIP application that this k-packet belongs.$P(V_i) = 0$,where $i \neq j$.

We use SGD (Stochastic Gradient Descent) optimizer to minimize the loss function in this paper, and apply nesterov momentum to update gradient after every iteration. The formula of gradient increment shown as follow,

$$\Delta X_t = \tau M_{t-1} - \eta \nabla f(X_{t-1} + \tau M_{t-1}) \tag{3}$$

where $\tau$ denotes the factor of momentum, $\eta$ denotes learning rate. $g_t = \nabla f(X_{t-1} + \tau M_{t-1})$ is the gradient of transition point, where $\Delta X_t$ denotes the real decreasing displacement, $X_t$ denotes the position at time t, $M_t$ denotes the momentum ate time t.

In our experiment, learning rate will decay after each epoch, decay rule shown as:

$$\eta_i = \eta_{i-1} \times \frac{1}{1 + \rho \times i} \tag{4}$$

where $\rho$ denotes the decay factor, i denotes the number of epoch.

### C. Feature Set

Our method is to get a more precise feature set for identifying VoIP traffic. The feature set extracted by deep learning is unreadable. We use these feature set to train SVM, random forest, decision tree and naive bayes. With this feature set, they all achieved good accuracy. We compare the accuracy of using our feature set and the feature set used in some previous research in Section VIII-C.

## VII. Real-time identification

In this section, we introduce how to real-time identify VoIP traffic in our identification system. There are 3 components in real-time identification module. Capturer capture UDP packets and separate them into corresponding flows by IP address and UDP port. Filter follows several rules to filter out obvious non-VoIP flows. Classifier classifies flows into corresponding application type with extracted features.

The capturer maintain a pending flows queue and map packet into its matching pending flow using its IP address and UDP port. Figure 3 shows the details to separate flows by IP address and UDP port. If a packet cannot find the corresponding pending flow, it creates a new pending flow and adopt IP address and UDP port as its key. If a packet is appended into pending flow, and the number of packet reaches k, this k-packet will be sent to next component filter. The capturer guarantees that all captured packets are UDP packets, and the capturer can remove dead flows from pending flows queue. Flows whose number of packets doesn't reach k within limited time will be considered as dead flows.

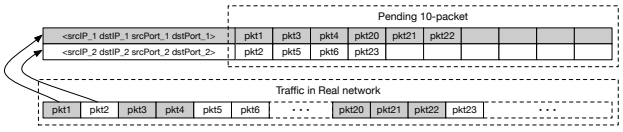


Fig. 3. To separate flows by IP address and UDP port in real network.

The second component is a rule-based filter, it receives k-packets sent by capturer. Because non-VoIP flows are no value to us, it will be filtered out to a certain extent. The remaining flows will be sent to next component to do further identification. We select benchmark values from our dataset and set several rules based on these benchmark values. Related algorithm and parameters are shown in Algorithm 1 and Table II

TABLE II
Benchmark value and threshold used in Algorithm 1.

| | Symbol | Description |
|---|---|---|
| 1. | min_mpl | Minimum of mean packet length in datatset |
| 2. | max_mpl | Maximum of mean packet length in datatset |
| 3. | min_pl | Minimum of packet length in datatset |
| 4. | max_pl | Maximum of packet length in datatset |
| 5. | min_miat | Minimum of mean inter-arrival time in datatset |
| 6. | max_miat | Maximum of mean inter-arrival time in datatset |
| 7. | min_iat | Minimum of inter-arrival time in datatset |
| 8. | max_iat | Maximum of inter-arrival time in datatset |
| 9. | $\lambda_1$, $\lambda_2$ | 2 thresholds for upper and lower limits |
| 10. | $\varrho$ | Proportional threshold for number of packets |

We input the matrix into identification model, it will output a vector whose size is $1 \times n$, n denotes the number of VoIP applications that our identification model can identify. The vector indicates that this k-packet's likelihood estimation for every VoIP application. The index of the maximum value in the vector is the identification result. We will set the value in map database and remove this flow in pending flows. We do not have a way to identify unknown VoIP applications for now, but we want to improve our system. So we will research the packets whose maximum likelihood is less than 0.9.

In order to detect VoIP voice traffic in real time, we have some rules. We only monitor the packets whose transport layer's protocol is UDP, and their packet length is less than 256. We parse the UDP packet that meets our requirements by RTP or RTCP protocol. If the UDP packet can be parsed correctly, we will process this packet with 2 operations, append it into corresponding pending flow or search identification result in map database. If it cannot be parsed, we ignored it.

**Algorithm 1** Algorithm to identify VoIP/non-VoIP Flows

**Input:** Flow: k-packet
**Output:** VoIP/non-VoIP

**if** $\lambda_1$min_mpl $<$ mean packet length of current k-packet $<$ $\lambda_2$max_mpl **and** $\lambda_1$min_miat $<$ mean inter-arrival time of current k-packet $<$ $\lambda_2$max_miat **then**
  **for** packet in k-packet **do**
    **if** (packet length $<$ $\lambda_1$min_pl ) **then**
      $ct\_pl\_l \leftarrow ct\_pl\_l + 1$
    **end if**
    **if** (packet length $>$ $\lambda_2$max_pl ) **then**
      $ct\_pl\_u \leftarrow ct\_pl\_u + 1$
    **end if**
    **if** (packet inter-arrival time $<$ $\lambda_1$min_iat ) **then**
      $ct\_iat\_l \leftarrow ct\_iat\_l + 1$
    **end if**
    **if** (packet inter-arrival time $>$ $\lambda_2$max_iat ) **then**
      $ct\_iat\_u \leftarrow ct\_iat\_u + 1$
    **end if**
  **end for**
  **if** ct_pl_l $<$ $\varrho$k **and** ct_pl_u $<$ $\varrho$k **and** ct_iat_l $<$ $\varrho$k **and** ct_iat_u $<$ $\varrho$k **then**
    **return** $VoIP$
  **else**
    **return** $non - VoIP$
  **end if**
**else**
  **return** $non - VoIP$
**end if**

Moreover, we ignore packet that is alone, it makes no sense to monitor it. It is easy but it works better than the way based on call control signaling.

## VIII. Performance evaluation

In this section, we first show the dataset used in our experiment. Secondly, we give the specific parameters used in the training of deep neural networks. Finally, we show the experimental results and give the evaluation of the system performance.

### A. Dataset

We capture traffic and divide it into 2 dataset for these 10 VoIP applications. We use dataset1 to train identification models, and we use dataset2 to examine these models. The details of dataset1 used for training shown in Table III.

The forth column shows the num. of k-packet where k is 10, of course, we also process the dataset according to k is 2, 4, 6, 8 and so on. 80% of the k-packet are used for training when 20% used for validating. It decides the first dimension of the matrix that will be input into AlexNet model.

### B. Parameter Setting

The optimization algorithm SGD used in training phase is not adaptive for learning rate, so the setting of learning rate has a direct impact on the experimental results. We learn from the general experience of deep learning, and we adjust it according to our experimental results. The initial learning rate is set to 0.01, it decays after 2 epochs following the rules mentioned in section 6.3. Nesterov updates based on the gradient, we set momentum factor to 0.9. We train 20 epochs using the dataset for the above 8 models, we set batch size to 100. With these parameters, we get satisfactory results for every model. Table IV shows the number and storage size of weights identification model.

TABLE III
THE DETAILS OF DATASET1.

| VoIP | Platform | Size(MB) | Num.(k=10) |
|---|---|---|---|
| Skype | Windows, Linux | 3908.8 | 669984 |
| UUCall | Windows | 2709.4 | 691566 |
| KCCall | Windows | 3128.8 | 808224 |
| ALTCall | Windows | 2795.2 | 692002 |
| Jumblo | Windows, Linux | 3704.6 | 871468 |
| Zoiper | Windows, Linux | 4418.1 | 677114 |
| Xlite | Windows, Linux | 5165.3 | 638288 |
| Eyebeam | Windows | 4524.7 | 616773 |
| ExpressTalk | Windows | 4633.1 | 602637 |
| Bria | Windows | 4476.0 | 598724 |

TABLE IV
THE DETAILS OF ALEXNET PARAMETERS.

| Model | Input_Shape | Num. of Parm | Size(MB) |
|---|---|---|---|
| k=2 | $2 \times 256 \times 1$ | 21827434 | 87.3 |
| k=4 | $4 \times 256 \times 1$ | 21827434 | 87.3 |
| k=6 | $6 \times 256 \times 1$ | 38604650 | 154.5 |
| k=8 | $8 \times 256 \times 1$ | 38604650 | 154.5 |
| k=10 | $10 \times 256 \times 1$ | 55381866 | 221.6 |
| k=20 | $20 \times 256 \times 1$ | 88936298 | 355.8 |
| k=40 | $40 \times 256 \times 1$ | 172822378 | 691.3 |
| k=100 | $100 \times 256 \times 1$ | 424480618 | 1619.3 |

### C. Experiment Results

The dataset listed in Section VIII-A is used to train 8 models. With the parameter setting in Section VIII-B, we get 8 identification models. The training and validating accuracies are shown in Table V.

For the above 8 identification models, we use TPR (True Positive Rate) and FPR (False Positive Rate) to evaluate them. TPR reflects the probability of positive samples are identified correctly, and FPR reflects the probability of negative samples

TABLE V
THE TRAINING AND VALIDATING ACCURACY OF 8 IDENTIFICATION MODELS.

| Model | Loss | Train Acc. | Validate Acc. |
|---|---|---|---|
| k=2 | 0.015865 | 0.995560 | 0.996130 |
| k=4 | 0.007670 | 0.997700 | 0.997860 |
| k=6 | 0.004933 | 0.998520 | 0.999340 |
| k=8 | 0.004393 | 0.998690 | 0.999130 |
| k=10 | 0.001159 | 0.999650 | 0.999650 |
| k=20 | 0.000546 | 0.999880 | 0.999880 |
| k=40 | 0.002920 | 0.999120 | 0.999460 |
| k=100 | 0.006689 | 0.997620 | 0.998730 |

TABLE VI
THE TIME CONSUMED TO IDENTIFY 100 FLOWS.

| Model | k=2 | k=4 | k=6 | k=8 | k=10 | k=20 | k=40 | k=100 |
|---|---|---|---|---|---|---|---|---|
| Time(s) | 27.55 | 20.12 | 29.09 | 30.53 | 48.00 | 73.62 | 141.33 | 309.51 |

are identified as positive samples incorrectly.TPR can be computed by Equation 5.

$$TPR = \frac{TP}{TP + FN} \tag{5}$$

And FPR can be computed by Equation 6.

$$FPR = \frac{FP}{FP + TN} \tag{6}$$

In our experiment, the identification rate for dataset1 is close to 1. When we examine these 8 models with dataset2, there are some outliers. They can be seen in the first 4 charts shown on Figure 4, model-20 for Skype, model-4 for Xlite and so on. This is overfitting, the identification model that we trained is overfitting the dataset. In the future research, we first consider add the L1-norm and L2-norm into our model, we will merge dataset2 into dataset1 to enlarge dataset to avoid overfitting. Apart from this, these charts show good performance to most of VoIP applications.

Figure 5 shows the probability distribution of true result and false result of dataset1 and dataset2, which true result means a k-packet is identified correctly and false result means a k-packet is identified in correctly. It shows the maximum probabilities of true results are generally close to 1, and they are disperse from 0.2 to 0.98. We use the feature set we extracted by neural network to train 4 machine learning models, the accuracy shows in Figure 6. The previous research we mentioned in Section II, they use machine learning and feature set extracted by human to identify VoIP traffic. With our feature set extracted by deep learning, the accuracy is higher than 98%.

The identification efficiency of SVM is better than original AlexNet model, we deploy identification model trained by SVM into real network. We capture 100 flows for each VoIP application and identify them by these models. Time consumed by these models is shown on Table VI. From this table we can see that it costs 3 seconds when k is 100, it is tolerate for a long call which lasts for a few minutes. For fraud calls which always end quickly, identification time needs to be further reduced, we have to consider using the smaller k.

## IX. CONCLUSION

In our work, we propose a generic approach to identify VoIP application in realtime. Then we deploy this approach into a real-time identification system, and the system shows good performance. Our approach cannot identify unknown VoIP applications, we will go a step further. We only deploy this system in a small network to test its real-time performance, we will deploy our system into real network and apply Storm in our system.

## REFERENCES


[1] J. Curtis, J. Cleary, A. McGregor, and M. Pearson, "Measurement of voice over ip traffic," in *Passive and Active Measurement conference*, 2000.

[2] M. Chen, G.-X. Zhang, and J.-P. Bi, "Research of sip-based voip traffic identification methodology," *Appl. Res. Comp*, vol. 24, 2007.

[3] J. Lee, K. Cho, C. Lee, and S. Kim, "Voip-aware network attack detection based on statistics and behavior of sip traffic," *Peer-to-Peer Networking and Applications*, vol. 8, no. 5, pp. 872–880, 2015.

[4] J. V. Gomes, P. R. M. Inacio, M. Pereira, M. M. Freire, and P. P. Monteiro, "Identification of peer-to-peer voip sessions using entropy and codec properties," *IEEE Transactions on Parallel & Distributed Systems*, vol. 24, no. 10, pp. 2004–2014, 2013.

[5] T. Qin, L. Wang, Z. Liu, and X. Guan, "Robust application identification methods for p2p and voip traffic classification in backbone networks," *Knowledge-Based Systems*, vol. 82, pp. 152–162, 2015.

[6] D. Bonfiglio, M. Mellia, M. Meo, and D. Rossi, "Detailed analysis of skype traffic," *IEEE Transactions on Multimedia*, vol. 11, no. 1, pp. 117–127, 2009.

[7] D. Adami, C. Callegari, S. Giordano, M. Pagano, and T. Pepe, "Skype-hunter: A real-time system for the detection and classification of skype traffic," *International Journal of Communication Systems*, vol. 25, no. 3, pp. 386–403, 2012.

[8] S.-H. Lee, Y.-H. Goo, J.-T. Park, S.-H. Ji, and M.-S. Kim, "Sky-scope: Skype application traffic identification system," in *Network Operations and Management Symposium (APNOMS), 2017 19th Asia-Pacific*. IEEE, 2017, pp. 259–262.

[9] S. Sen, O. Spatscheck, and D. Wang, "Accurate, scalable in-network identification of p2p traffic using application signatures," in *Proceedings of the 13th international conference on World Wide Web*. ACM, 2004, pp. 512–521.

[10] F. Idrees and U. Aslam, "A generic technique for voice over internet protocol (voip) traffic detection," 2008.

[11] B. Li, Z. Jin, and M. Ma, "Voip traffic identification based on host and flow behavior analysis," in *Wireless Communications Networking and Mobile Computing (WiCOM), 2010 6th International Conference on*. IEEE, 2010, pp. 1–4.

[12] M. Rathore, A. Paul, A. Ahmad, M. Imran, and M. Guizani, "High-speed network traffic analysis: detecting voip calls in secure big data streaming," in *2016 IEEE 41st Conference on Local Computer Networks (LCN)*. IEEE, 2016, pp. 595–598.

[13] N. A. Saqib, Y. SHAKEEL, M. A. Khan, H. MEHMOOD, and M. Zia, "An effective empirical approach to voip traffic classification," *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 25, no. 2, pp. 888–900, 2017.

[14] R. Alshammari and A. N. Zincir-Heywood, "Identification of voip encrypted traffic using a machine learning approach," *Journal of King Saud University - Computer and Information Sciences*, vol. 27, no. 1, pp. 77–92, 2015.

[15] T. Yildirim and P. J. Radcliffe, "Voip traffic classification in ipsec tunnels," in *International Conference on Electronics and Information Engineering*, 2010, pp. V1–151 – V1–157.

[16] T. Qin, L. Wang, Z. Liu, and X. Guan, "Robust application identification methods for p2p and voip traffic classification in backbone networks," *Knowledge-Based Systems*, vol. 82, pp. 152–162, 2015.

[17] Z. Wang, "The applications of deep learning on traffic identification," 2015.

[18] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[19] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors." *Readings in Cognitive Science*, vol. 323, no. 6088, pp. 399–421, 1988.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *International Conference on Neural Information Processing Systems*, 2012, pp. 1097–1105.

[21] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Computer Science*, 2014.

[22] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

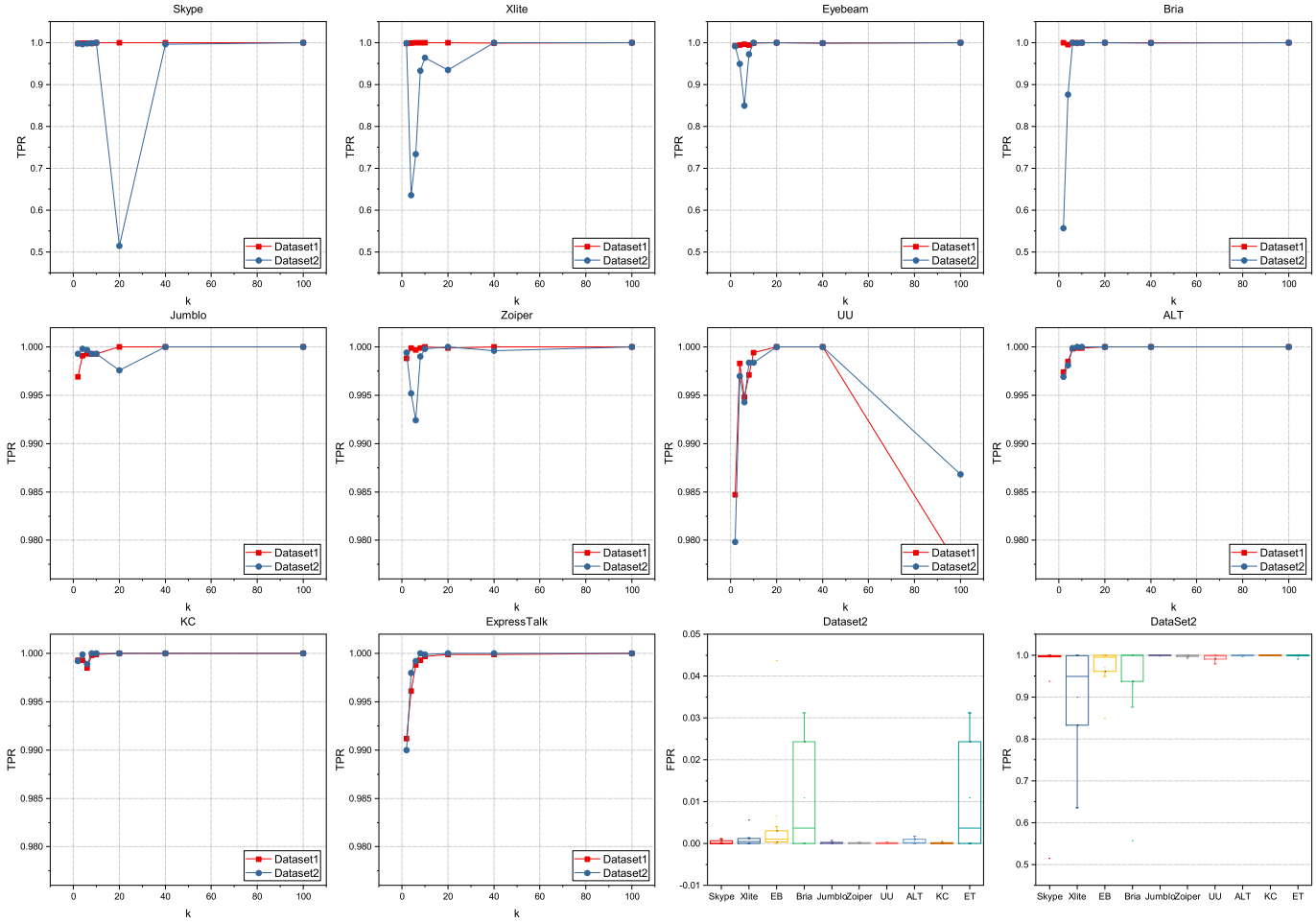


Fig. 4. The performance of the 8 models shown by TPR and FPR.

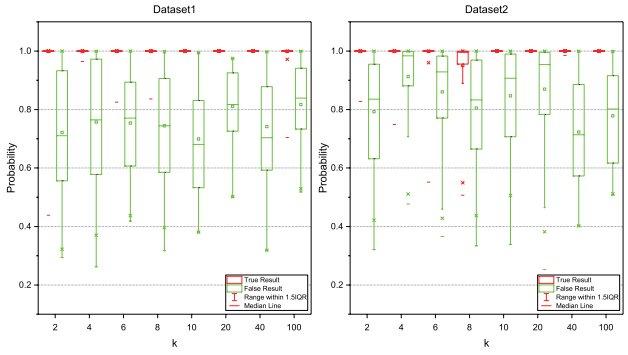


Fig. 5. The probability distribution of true result and false result.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

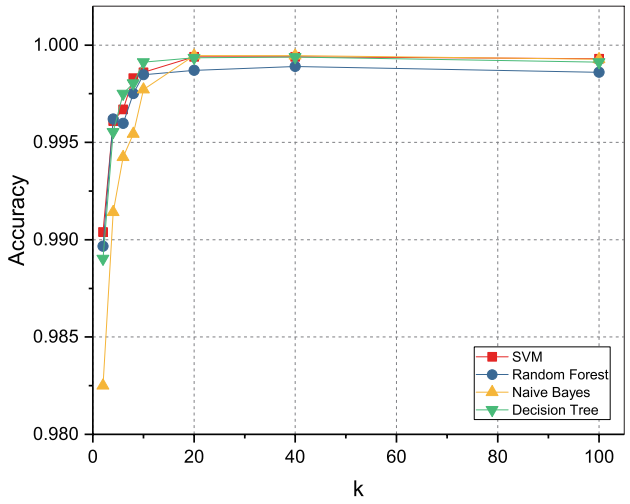


Fig. 6. The accuracy of 4 machine learning methods using extracted feature set.