Lecture Notes in Networks and Systems 1421

Pit Pichappan
Martín López-Nores
Duong Van Hieu *Editors*

2024 Real-Time Intelligent Systems

Proceedings of the Sixth International Conference on Real Time Intelligent Systems RTIS 2024

Springer

Deep Learning for Encrypted Traffic Classification and Unknown Data Detection

Ramakrishnan Bakthavatchalu[1(✉)] and Meenakshi Chockalingam[2]

[1] Computer Science, VISTAS, Chennai, India
caliber.ram@gmail.com
[2] Department of Computer Application (PG), VISTAS, Chennai, India
meenasi.c@gmail.com

Abstract. Irrespective of the growing usage of encryption techniques to maintain privacy during Internet communications, users of mobile devices continue to face privacy and security issues. This research proposes a unique Deep Neural Network (DNN) based on a user activity detection framework to recognize certain user actions from an encrypted Internet traffic stream that is sniffed, which are called in-app activities. Maintaining the security and privacy of data transfer requires encrypted communication. Because it keeps hackers from intercepting private data that they could access without permission, it is crucial to maintain the security of our networks. Modern communication systems are seeing an increase in encrypted network traffic, which makes network management and security more difficult. To overcome this obstacle, Due to the inability to examine the content of the packets and the lack of clear visibility into their contents, machine learning models have been used to classify encrypted communication with varying degrees of success. More productive research has started on creating machine learning models for identifying encrypted payloads without explicitly analyzing their contents to address this problem. One of the difficulties is the vast number of apps; it is nearly hard to gather and train a DNN model with all the data that might come from them. As a result, in this work, we utilize the DNN output layer's probability distribution to filter out data from applications that weren't considered during model training (i.e., unknown data). The suggested approach uses a time-based strategy to partition the activity's traffic flow to identify in-app actions by analyzing a small portion of the activity's traffic. Our experiments reveal that the DNN-based framework can recognize previously learned in-app behaviors with an accuracy of 90% or higher and can identify them with an average accuracy of 79%.

Keywords: deep neural network · encrypted traffic classification · wireless networks · mobile applications · network analysis

1 Introduction

The rapidly rising needs of society lead to the development of more technology. Examples of these cutting-edge technologies that researchers are concentrating on are IOT and network traffic categorization. Recognizing and classifying network traffic is crucial

P. Pichappan et al. (Eds.): RTIS 2024, LNNS 1421, pp. 340–347, 2025.
https://doi.org/10.1007/978-3-031-92545-0_31

for preserving cyberspace security. The Internet is growing rapidly, and with it, new protocols and network applications increase in complexity and diversity in the kinds of network traffic that flow over it. This presents several challenges to the efficient administration of these networks. To properly manage various forms of network traffic, it is necessary to employ sophisticated algorithms for precisely classifying the traffic.

Classification of traffic flow in today's Internet Protocol (IP) network has become an important research area, especially with the recent adoption of Machine Learning (ML) techniques and Software-Defined Networking (SDN) principles. ML algorithms are pivotal in minimizing packet loss and achieving exact classification outcomes.

In recent years, smartphones have become the primary device users use to access the Internet. A mobile device can be used to view mobile websites and applications, sometimes known as apps. Due to the greater personalization, speed, and ease of use of mobile applications, most people prefer using them over visiting comparable websites through web browsers. Therefore, mobile applications have largely used web browsers to interact with most internet services, including social networking, banking, and streaming video. Research indicates that users of mobile devices spend about 90% of their time on mobile apps instead of web pages.

2 Literature Review

This section compares recently published studies that are relevant to our work. A technique developed by Conti et al. uses packet sizes and their layout to infer user actions on mobile apps. Park and Kim attack the mobile instant messaging provider KakaoTalk. These methods make use of traditional machine learning algorithms such as k-nearest neighbor and random forest. However, human-generated features, which require a lot of work and have limited generalizability, play a significant role in their performance.

For a visual comparison, Table 1 summarizes the corpus of research that has been done in this area and was covered above. The following are some noteworthy observations.

Table 1 provides a visual comparison of the body of research in this field that was discussed previously. Here are a few observations that are worth noting. Despite the fact that this is a typical situation, delivering network traffic analysis in the presence of noise from unknown traffic has not been particularly covered in any of the numerous studies in the literature.

CNN has been investigated recently for classification tasks involving converting network traffic flows to picture formats.

3 Proposed Methodology

This paper proposes a framework for in-app activity recognition using deep learning techniques, specifically DNN.

Table 1. Shows how the planned work is positioned with the literature's corpus of existing research.

Classification type	Encrypted traffic	Unknown data detection	Classification based on the subset of traffic	Results compared with the state-of-the-art methods	Dataset type	Accuracy
Traffic characterization and application	✓	✗	✗	✓	ISCX	90%
Service (HTTP, DNS, Telnet, etc.)	✓	✗	✓	✗	RedIRIS	93%
Service (HTTP, DNS, Telnet, etc.)	✓	✗	✓	✓	ISCX	80%
Traffic characterization	✓	✗	✗	✗	ISCX	90%–93%
Application	✓	✗	✓	✗	Private	82.30%
SNI	✓	✗	✗	✓	3 Private datasets	85.54%
Application	✓	✗	✗	✓	ISCX	90.75%
Traffic type and application	✓	✗	✗	✓	MaMPF	94.14%
Application	✓	✗	✗	✗	Private	95%
QUIC based services	✓	✗	✗	✓	Private	75.55%
Application and Activity	✓	✓	✓	✓	Openly shared	90% to 95%

3.1 Data Collection

Eight mobile apps were used to record a sequence of user actions, resulting in a comprehensive dataset whose attributes are listed in Table 2.

Data was collected four times for each action to create enough traffic flows for inference. More samples were collected for Facebook, Skype, and YouTube.

This is because Facebook offers more activities than other apps; on the other hand, Skype offers activities like making and receiving audio and video calls and sending and receiving video clips.

YouTube primarily offers activities connected to watching, posting, and downloading videos.

There is a lot of traffic from these events.

The recorded network traffic was stored in *pcap* files.

Table 2. Features of the app that were noted when collecting data.

App	Category	Number of	App
Facebook	Social networking	20	15,844
Instagram	Photo and video	18	5,718
YouTube	Photo and video	7	13,201
WhatsApp	Social networking	7	346
Viber	Social networking	7	560
Gmail	Productivity	4	836
Skype	Social networking	7	14,501
Messenger	Social networking	7	5,041
Total: 8 apps		77	56,047

To acquire the ground truth, each activity's generated network traffic was gathered independently, and each activity's name was appended to the network trace.

3.2 Data Pre-processing

It must be prepared before training the model to appropriately feed the network traffic data to the neural network. To do this, the prepared dataset underwent a pre-processing phase, including the following actions.

(a) **Filtering datasets**
IEEE 802.11 frames come in three flavors: data, control, and management. Only the data frames are utilized to transmit data during communication; the other two may make analysis more difficult.

(b) **Acquiring frames free of errors**
Packet losses in wireless traffic usually result in high retransmission rates. Retransmission of packets has the potential to alter an application's traffic pattern.

(c) **Normalization of data**
One-step that is essential to DL performance is data normalization. Standardization, or feature scaling, is used to normalize the data. There are scale-inconsistent feature values in the dataset.

(d) **Segmenting traffic**
It is not feasible to guarantee that every transaction of user activity may be seen during in-app activity detection. This is due to the possibility that the eavesdropper could begin recording traffic while the intended recipient is acting

3.3 Neural Network Architecture

A DNN has been utilized in this study to carry out the essential classifications During the neural network formation process, many important decisions must be taken, such as the number of hidden layers, the number of nodes in each hidden layer, the activation

function to employ, etc. The performance of a model can vary significantly based on the selection of hyperparameters. The hyperparameters of a DNN model can be chosen arbitrarily. Consequently, multiple tests were conducted with different hyperparameter configurations. The model was considered to avoid both under fitting and overfitting.

Few models are trained and tested with varying amounts of layers and neurons. The model that outperformed the other models assessed and trained in accuracy values is selected for every potential combination of characteristics to be learnt by the fully connected layers.

Table 3. Parameter values of the proposed DNN model.

Parameter	Values
Number of hidden layers	4
Number of nodes in each layer	[1014, 412, 226, 116]
Activation functions Loss function	Hidden layers – Tanh Output layer - Softmax
Optimizer	Adam
Batch size	2033
Number of epochs	100

The four hidden layers of the suggested DNN model architecture are composed of 1014, 412, 226, and 116 neurons. The dropout mechanism randomly resets several neurons to zero during the training phase. As a result, a random group of neurons are active at each repetition.

3.4 A Method for Categorizing Untrained In-App Activities

One of this work's key contributions is identifying untrained in-app activities as unknown traffic. A class probability-based methodology for detecting unknown in-app activities.

The output layer of the model may forecast the likelihood that a given data instance belongs to each class when input traffic is delivered.

4 Experimental Results and Discussions

The performance assessment of the suggested framework is provided in this section. There are two components to the performance evaluation:

The model should be able to correctly identify previously trained in-app behaviors when they are fed into it.

The model should be able to recognize previously untrained in-app activities as unknown data when they are fed into it.

4.1 Identification of In-App Activity

To identify in-app activities, the DNN model architecture suggested in Table 3 was employed. They are 0.05, 0.02, 0.2, and 0.5 s. Table 4 presents the classification accuracy results for the 92 in-app actions that were compared to these window sizes.

Table 4. In-app activity classification performance of the DNN model at different time window sizes.

Window size	Total samples	Training accuracy	Validation accuracy
0.5 s	56,047	95%	91%
0.2 s	230,624	95%	93%
0.05 s	1,666,524	92%	90%
0.02 s	6,057,263	91%	90%

4.2 Identification of Unknown In-App Activities

Eight apps in all are taken into consideration in this investigation. The test dataset including the previously untrained in-app events was fed into the model to see how well it could identify "unknown" traffic. Table 5 displays the values acquired at each test instance (Table 6).

Table 5. Noise detection rates of the DNN model when tested with previously untrained in-app activities.

Test no.	The apps used to train the DNN model	The app is not used in model training	Noise
T1	I, Y, G, M, S, W, V	F	85%
T2	F, I, G, M, S, W, V	Y	60%
T3	F, Y, G, M, S, W, V	I	71%
T4	F, I, Y, M, S, W, V	G	90%
T5	F, I, Y, G, S, W, V	M	71%
T6	F, I, Y, G, M, W, V	S	61%
T7	F, I, Y, G, M, S, W	V	90%
T8	F, I, Y, G, M, S, V	W	82%

4.3 A Review of Performance Against the State-of-the-Art

The Bayat et al. model and the Deep Packet model are the best-matching models that have been considered for our framework's performance comparison with the state-of-the-art.

Table 6. Misclassification matrix (% values) of the DNN model when tested with previously untrained in-app activities.

Test app		F	Y	I	G	M	S	V	W
	F		30	12	2	11	30	2	1
	Y	31		10	2	10	25	1	1
	I	44	22		2	10	30	2	1
	G	30	20	10		10	25	1	1
	M	32	22	10	2		22	1	1
	S	40	27	12	2	10		2	1
	V	30	20	10	2	8	20		1
	W	30	20	10	2	8	20	1	

The traffic classification used in these chosen DL models was likewise performed using the same dataset created in our study for a fair comparison (Table 7).

Table 7. Performance comparison of the proposed DNN model with the state-of-the-art DL models.

Architecture	Accuracy with the original dataset	Accuracy with our dataset
1D CNN and RNN	81.3%	80%
ID CNN	95%	30%
SAE	93%	32%
DNN	91%	91%

5 Conclusions

Using a DNN architecture for fine-grained in-app activity identification, this work adds to the literature on encrypted traffic classification. Experiments were conducted to determine the hyperparameters that optimize the DL model's performance. The suggested model successfully distinguished previously untrained in-app activity traffic as unknown data with an average accuracy of 75% while also identifying previously trained in-app activities with an accuracy of 90% or higher, according to the results. Thanks to the suggested windowing technique, the model can do categorization based on a few frames of the traffic flow rather than the full transaction flow. This is a far more complex process since there is more information in the full network flow than in just a portion of it. This function enables.

We intend to use cutting-edge machine learning concepts like transfer learning in our upcoming work to improve training speed and accuracy. It typically takes a few hours to many days or weeks to train a model from the beginning, depending on the size, architecture, and dataset. Thus, a model trained on a larger dataset for a source task can be applied to a different target task through transfer learning. Because the pre-trained

model has previously been trained to capture valuable characteristics, retraining on the target task requires a substantially smaller amount of labeled data and training time.

References

1. Flurry Analytics, ComScore, Pandora, Facebook, NetMarketShare (2015). https://www.marketingcharts.com/industries/media-and-entertain-ment58693/attachment/flurry-share-time-spent-mobile-devices-sept. Accessed 15 Dec 2021
2. Pathmaperuma, M.H., Rahulamathavan, Y., Dogan, S., Kondoz, A.M.: In-app activity recognition from Wi-Fi encrypted traffic in intelligent computing. In: Arai, K., Kapoor, S., Bhatia, R. (eds.) SAI 2020. AISC, vol. 1228. Springer, Cham (2020)
3. Aceto, G., Ciuonzo, D., Montieri, A., Pescapé, A.: DISTILLER: encrypted traffic classification via multimodal multitask deep learning. J. Netw. Comput. Appl.Netw. Comput. Appl. **183**, 102985 (2021)
4. Zhao, M., Chang, C.H., Xie, W., Xie, Z., Hu, J.: Cloud shape classification system based on multi-channel CNN and improved FDM. IEEE Access **8**, 44111–44124 (2020)
5. Jin, B., Cruz, L., Goncalves, N.: Pseudo RGB-D face recognition. IEEE Sens. J. (2022)
6. Aircrack-ng. https://www.aircrack-ng.org/. Accessed 1 Dec 2021
7. Chollet, F.: Keras. https://github.com/fchollet/keras. Accessed 1 Apr 2022