Deep Content: Unveiling Video Streaming Content From Encrypted WiFi Traffic

Ying Li,* Yi Huang,* Suranga Seneviratne,§ Kanchana Thilakarathna,§ Adriel Cheng,†
Guillaume Jourjon,‡ Darren Webb,† and Richard Xu*

* University Technology of Sydney, Australia; Email: {ying.li-11, yi.huang-3,yida.xu}@uts.edu.au
† Defence Science & Technology Group, Edinburgh, Australia; Email: {firstname.lastname}@dst.defence.gov.au
‡ Data61-CSIRO, Sydney, Australia; Email: guillaume.jourjon@data61.csiro.au
§ University of Sydney, Australia; Email: {firstname.lastname}@sydney.edu.au

***Abstract*—The proliferation of smart devices has led to an exponential growth in digital media consumption, especially mobile video for content marketing. The vast majority of the associated Internet traffic is now end-to-end encrypted, and while encryption provides better user privacy and security, it has made network surveillance an impossible task. The result is an unchecked environment for exploiters and attackers to distribute content such as fake, radical and propaganda videos.**

Recent advances in machine learning techniques have shown great promise in characterising encrypted traffic captured at the end points. However, video fingerprinting from passively listening to encrypted traffic, especially wireless traffic, has been reported as a challenging task due to the difficulty in distinguishing retransmissions and multiple flows on the same link. We show the potential of fingerprinting videos by passively sniffing WiFi frames in air, even without connecting to the WiFi network. We have developed Multi-Layer Perceptron (MLP) and Recurrent Neural Networks (RNNs) that are able to identify streamed YouTube videos from a closed set, by sniffing WiFi traffic encrypted at both Media Access Control (MAC) and Network layers. We compare these models to the state-of-the-art wired traffic classifier based on Convolutional Neural Networks (CNNs), and show that our models obtain similar results while requiring significantly less computational power and time (approximately a threefold reduction).

I. Introduction

More than 50% of global Internet traffic is currently end-to-end encrypted and HTTPS has become the norm of many forms of communication over the Internet [1]. While Transport Layer Security (TLS), the underlying security protocol behind HTTPS, provides confidentiality for the message that is being exchanged between two parties, wide adoption of end-to-end encryption opens up several issues in terms of network management. For example, end-to-end encryption eliminates the possibility of traffic analysis in the core network for intrusion detection and parental filtering. It hinders network optimisations carried out by telecommunication operators. Furthermore, end-to-end encryption makes network surveillance impossible impeding national security activities. As such, there is increasing interest in making inferences and predictions from encrypted traffic flows.

Previous research explored the possibility of making inferences based on encrypted traffic flows by capturing network data packets at the IP layer [2], [3], [4]. This is viable because despite the message content being encrypted, the statistical properties of traffic flows such as packet lengths, inter-packet times, burst sizes, and burst intervals can still reveal information about the underlying encrypted traffic that are in transit. Also, at the IP level there are other useful meta-data such as IP addresses, ports, and TLS header information.

In this paper, we investigate the possibility of making useful inferences from passively observed WiFi traffic that is encrypted at both transport layer (TLS) and MAC layer (WPA2). This is more challenging in comparison to making predictions from the IP layer traffic due to lack of any meta information. Specifically, we focus on identifying traffic flows from a set of known online videos. Videos are highly popular on the Internet, but are frequently misused in many ways that include distribution of fake news, hate speech, and radical and propaganda content.

Thus, especially in network protection, counter-intelligence and situational awareness applications, there is a strong need to identify whether certain known videos are being watched by certain individuals or in a certain area. On the other hand, the possibility of identifying videos by passive WiFi observation further reinforces the need to build protocols that do not leak any information by creating fingerprints. We make the following contributions in this paper.

- We demonstrate the possibility of making predictions from encrypted WiFi traffic by building deep learning-based classifiers that are able to identify specific videos from a closed set of videos when they are streamed from a popular video hosting service, i.e. YouTube via Dynamic Adaptive Streaming over HTTP (DASH).
- We show that a simple Multi-Layer Perception architecture is able to achieve 97% accuracy in identifying videos from a closed set of 10 videos purely based on passive measurements collected at the WiFi layer.
- We compare our model with a state-of-art CNN model proposed for the same task in the IP layer [2]. We show that while such a model is also suitable for predictions using WiFi traffic, it requires significantly more resources both in terms of CPU and time (approximately three times more resources are required for training the CNN model).

978-1-5386-7659-2/18/$31.00

- Finally, we evaluate the longevity of our classifier by making predictions two weeks apart and show that our classifier is still able to maintain the same level of accuracy.

The remainder of the paper is organised as follows. In Section II, we review related work that use deep learning models to perform traffic classification. Section III presents an overview of DASH streaming and our data collection process. Section IV introduces the three deep learning models tested in this paper and the features used for traffic classification. We present and discuss results of these models in Section V, and conclude and outline future work in Section VI.

II. Related Work

We review related work that has tackled network traffic classification over encrypted traffic flows. Traffic classification can be performed either in real-time (e.g. intrusion detection systems) or conducted posterior using network traces (e.g. in the case of network forensics).

Previous studies have investigated the feasibility and evaluated the performance of classical deep learning algorithms such as Recurrent Neural Networks, Convolutional Neural Networks, or Deep Neural Networks; using either the KDD cup [5] or the NSL-KDD cup datasets [6]. Niyaz et al. [7] proposed to use a sparse autoencoder combined with softmax regression [8] to re-identify flows from the NSL-KDD cup dataset. Ma et al. [9] proposed to parallelise the deep learning technique based on a spectral k clustering of the original data. This corresponds to building k independent deep neural networks (one per cluster). In [10], the authors only considered a 2-class problem with 6 features out of the 41 features proposed. Kim et al. [11] proposed to use Long Short-Term Memory recurrent neural networks in order to classify the traffic from the NSL-KDD dataset. They also modified the dataset to only contain 300 entries for every class. Overall, this technique achieves high true positive rates, but also introduces a very high level of false positives (up to 80%).

Dong et al. [12] compared Support Vector Machine (SVM), Decision Tree, Naïves Bayes and SVM with Boltzman Machines to perform traffic classification on the NSL-KDD dataset. Overall, SVM with Boltzman Machines obtained the best performance, but only marginally. In [13], Fiore et al. were first to propose Boltzmann Machines with a Recurrent Neural Network to classify network traffic. This technique belongs to the class of stochastic Energy-Based Models where an energy is associated to each configuration (state) of the system under analysis. Finally, Michael et al. [14] compared the performance of various Neural Networks models to more classical Bayesian models and found that these models obtain similar performances.

The techniques described detect possible intrusions based on full knowledge of the end-to-end communication flow such as length of the flow or total number of packets. In most of the solutions presented, the authors relied on either a full trace of the communications or a summary of every flow (as in the case of NSL-KDD cup dataset). In contrast, we do not assume any statistical knowledge of the flow to identify known videos, and we do not need to collect the entire flow to make predictions.

In [15], Kang et al. proposed a two-class classifier, based on MLP, in the particular context of a vehicular network as opposed to the work presented previously focusing only on enterprise network connected to the Internet. This classifier makes use of the "DATA" field in every packet using the Controller Area Network (CAN) protocol [16]. This field consists of a vector of 64 bits and the authors normalise these bits between 0 and 1 based on their probability. Overall, their method was promising but it appears to be limited to the specific field on vehicular communication due to its reliance on the CAN protocol.

More recently Schuster et al. presented the first ever use of CNNs to detect not only the type of traffic but also the content of encrypted traffic [2]. In their proposal, the authors were able to identify which videos were downloaded over HTTPS from several video providers using features that consist of temporal representations of the traffic. The authors obtain very high accuracies within each video provider. Similar to [2], the authors of [17] proposed a CNN model to identify the class of traffic based on the characteristics of the sequence of packets in a given flow.

Recent research has successfully identified video streaming content in encrypted traffic without using deep learning techniques [18], [19]. In particular [18] focused on identifying video streaming using a Variable Bit Rate algorithm within encrypted WiFi traffic using similarity metrics and statistical machine learning. In a similar manner, but with the addition of DASH streaming, [19] identified video streaming within WiFi traffic. In their latest work [19], the authors adopted an approach similar to [18] by modelling various video streaming traffic. They also applied multiple statistical machine learning techniques but did not proposed a generic method applicable to deep learning techniques. Overall, both methods obtained similar accuracies of 90%.

In contrast, we propose a video identification method that does not rely on a complete mathematical model of the video traffic, TCP/IP flow information, or complete flow statistics. Furthermore, our technique operates within wireless environments with less resource requirements than [2], can be deployed in a real life traffic environment, and achieves unprecedented accuracy above 97%.

III. Experimental Setup

A. DASH streaming

Video streaming over the Internet has shifted to what is commonly referred as HTTP-based Adaptive Streaming (HAS). HAS works by encoding multiple versions of an original content with different bit rates and resolutions. The encoding bit rates recommended by YouTube for video streaming are 8, 5 and 1Mbps for video resolutions of 1080p, 720p and 360p respectively for frame rates up to 30fps. Each version of the video is split into smaller chunks (or segments), typically of 2-4 seconds in length. These chunks are then stored on a web server which can be accessed by clients on demand using

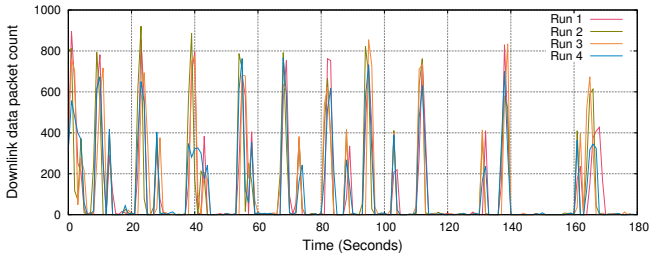


Figure 1. Different Traffic Flows of the Same Video

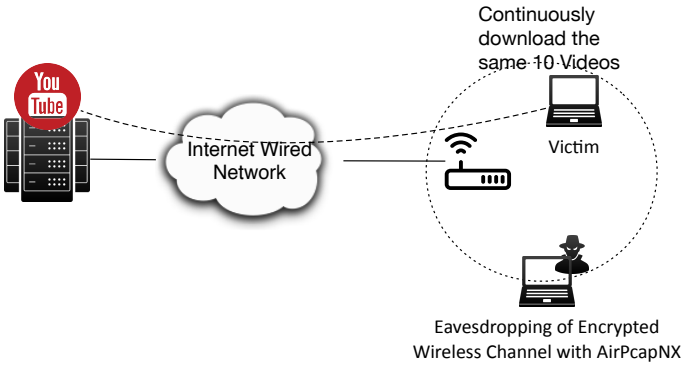


Figure 2. Experimental Setup

simple HTTP GET requests from a video streaming client running one of the HAS algorithms. Since HTTP operates on-top-of TLS, all data including HTTP headers are encrypted and can only be decrypted at the endpoints. The client video player (known as the DASH player) uses information from a server manifest file and current network conditions to dynamically adapt the stream.

As a result, streaming video with the DASH player creates a traffic profile which typically contains periodic spikes of downloads of potentially different magnitude based on the network condition (illustrated in Figure 1). These spikes are related to downloading the next series of chunks of the video followed by a waiting time until the user has played a certain percentage of the downloaded chunk. In particular, we can see in Figure 1 how the DASH player adapts to various network conditions by requesting different quality chunks, e.g. “Run 4" contains less packets compared to other runs. This traffic behaviour is exploited by our model described in Section IV.

B. Data Collection

To explore the feasibility of eavesdropping attacks over encrypted wireless network, we configured a laptop to connect to an 802.11n WiFi access point using channel 6 of the 2.4 GHz spectrum with WAP2 encryption. From this laptop, we repeatedly downloaded the same 10 videos from YouTube.[1] On a separate laptop, we used AirPcap Nx from Riverbed[2] to passively capture all the frames available on this channel regardless of the Ethernet address within the frame. This setup is illustrated in Figure 2.

Overall, we captured wireless traffic in a campus environment of 10 videos for more than 300 times for each video. For each video, we only captured the first three minutes of the stream. In addition, a registered YouTube Red account was used to avoid advertisements. These files were later post-processed using the *Scapy* Python library to extract frames associated with the targeted laptop Ethernet address. Using this data, we built the various features for our content classifier described in Section IV.

IV. *Deep Content* METHODOLOGY

In this section, we first describe data preprocessing and feature engineering procedures including data filtering, frame type identification and the key statistical characteristics gathered. We then present three different deep learning models and their corresponding network architectures.

[1]See appendix for the list of videos.

[2]https://www.riverbed.com

Table I
FEATURE SELECTION FROM WIRELESS TRAFFIC DATA

Traffic Direction	Feature Name
Uplink	F1. Number of Packets (data)
Downlink	F2. Number of Bytes (data)
Combination (up and down)	F3. Number of Packets (non-data)
	F4. Number of Bytes (non-data)
	F5. Minimum packet size
	F6. Maximum packet size
	F7. Average packet size
	F8. Variance packet size

A. Preprocessing & Feature Engineering

1) Data Filter: The WiFi captures include any data packet transmitted on our selected channel within proximity of the air environment. These packets were encrypted by IEEE 802.11 protocol using WPA-2, therefore it was impossible to extract any layer 3 and above protocol information such as port numbers or to apply deep packet inspection. Instead, we obtain several basic parameters from the MAC layer, for instance the frame size, frame type, frame duration time, radio information including signal strength and noise level, and MAC addresses of the source and destination. As explained in Section III, we also produced the same parameters for each direction of traffic flowing in and out of the target host.

2) Frame Type Identification: According to the IEEE 802.11 protocol, the MAC frames of the filtered target data are divided into three types: management frames, control frames, and data frames, of which data frames are most closely related to video classification. Hence, data packets are selected from the target laptop using the packet size parameter to only capture frames with a size greater than or equal to the minimum packet size in a wired network.

3) Feature Engineering: The captured data consists of the first three minutes of each video stream. This traffic is later grouped into up-link, down-link and combination (up and down) frames. For each group, we binned each feature in

500 bins of 0.36 seconds for ease of statistical computations. Features are generated from a sliding window over these bins.

In each temporal bin, we compute the features that characterise the dynamic aspect of the traffic: number of packets in data frames, number of bytes in data frames, number of packets in management and control frames, and number of bytes in management and control frames. In addition, the traffic waveform of MPEG-DASH video streaming from YouTube fluctuates depending on the video content. Accordingly, four additional features, namely the minimum size of packets, maximum size of packets, average size of packets, and variance of packet size, all within the studied time period, are constructed to further characterise video streaming traffic. These features are summarised in Table I.

B. Classifier Architectures

We implemented three neural network architectures: a Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) and Multi-Layer Perceptron (MLP). These classifiers are constructed using TensorFlow.[3] We shuffle the captured traffic and then split them into training and testing sets in ratios of 80% and 20% respectively.

1) Convolutional Neural Network Model: As explained in Section II, Schuster et al. [2] apply a CNN model to classify the content of the traffic. This is similar to the goal of our paper. The CNN model used in [2] reported excellent performance, and we implement this model as a reference architecture to evaluate the performance of our models. Note that the CNN model in our case is applied to data captured through sniffing WiFi wireless signal as opposed to wired traffic. The architecture of the CNN model in Figure 3(a) is cascaded by an input layer, three convolution layers, a max pooling layer and two fully connected layers. We apply the Adam optimiser to train this model on mini-batches with 64 samples.

2) Long Short-Term Memory Model: The behavior of the different video traffic exhibits different patterns with respect to time as illustrated in Figure 4. These graphs are exported from WireShark's statistic tools, where the X-axis is the time sequence with 1s interval and the Y-axis is the count of data packets on the down-link flow. The distribution of the spikes in different video traffic follows a specific time sequence. This indicates that time correlation of the feature values is crucial. Hence, we propose using RNNs because of its superiority in training time sequence data. Specifically, we utilise LSTM models which address the vanishing gradient problem in classical RNNs.

As shown in Figure 3(b), the input to the LSTM network is a fixed-size 500×1 array in which 500 denotes time steps and 1 denotes a single feature. This network has two hidden layers and each hidden layer has 32 neural nodes.

3) Multi-Layer Perceptron Model: During training, if just one feature was chosen the input to our network is a fixed-size 500 array. The array is passed through a stack of Fully-Connected (FC) layers. Through experiments, we acquired a high classification accuracy using two hidden layers and one dropout layer. The first hidden layer has 300 neural nodes and the second hidden layer has 100 neural nodes as shown in Figure 3(c). We train using the Adam optimiser on batches of 64 samples, with categorical cross-entropy as the loss function.

[3]Various parameters are listed in the appendix.

V. Results

In this section we first evaluate and demonstrate the effects of using our selected features in combination with the three neural network models outlined above. We then discuss their results in detail and shed light on the differences between these neural network models.

A. Classification Performance on CNN Model

As previously mentioned, during the data collection period about 3,198 video streaming samples of 10 videos were captured from YouTube Red. The input of the CNN model presented in Section IV is a 500×1 array where 500 denotes the number of temporal bins of 0.36 seconds and 1 denotes one feature. Each feature is utilised one by one to train the model and the corresponding test accuracy results are shown in Figure 5.

The performance of the CNN model, as shown in Figure 5 and Figure 6(a), is on par with results from Schuster et al. [2]. As mentioned in Section III, our data in this paper was captured by WireShark through AirPcap, not directly through the WLAN interface. Thus, there is much more noise in our captured data compared to the wired data from [2]. However, the accuracy of the trained model appears to be robust against such noise.

B. Classification Performance on LSTM Model

The LSTM model was built as described in Section III. We configured the LSTM model to select a single feature at a time. However, the input to the LSTM model is different from that of the CNN model because it is a sequence of 500 steps corresponding to the number of temporal bins of 0.36 seconds from the traffic traces.

Similarly, as with the CNN model, we applied each feature to train the LSTM model. As we can see in Figure 5, the LSTM model performs relatively well to detect and classify the video with an accuracy ranging from 72% for packet size variance to 96% for the number of packets in the sliding window in the uplink. In particular, we see that the LSTM model performs slightly worse than the state of the art CNN model [2]. To better understand these results we present in Figure 6(b) the confusion matrix of the LSTM model.

C. Classification Performance on MLP Model

Here, a video streaming flow is represented as a 500×1 array input to the MLP model with only one feature chosen. To seek the optimal MLP configuration scheme, hyper-parameters such as feature selection, number of hidden layers, number of nodes in each hidden layer and choice of activation function were evaluated by both empirical analyses and experimental

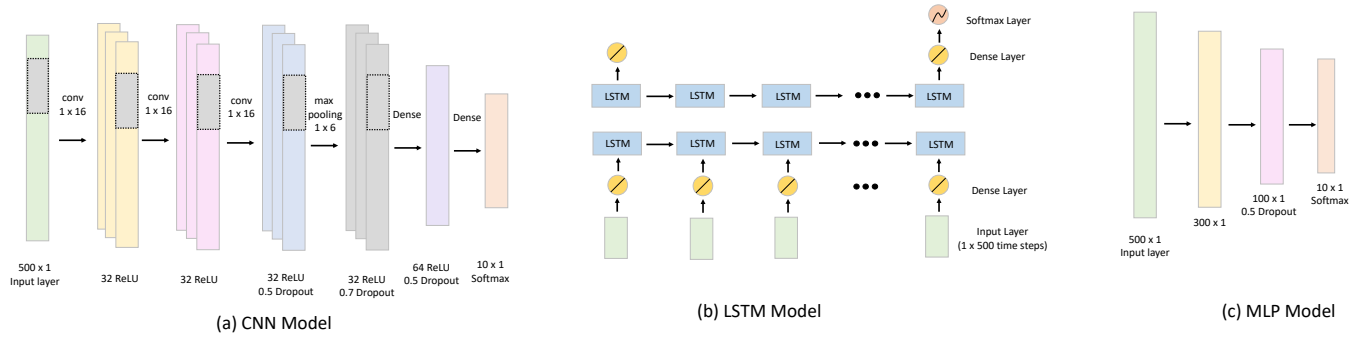


Figure 3. Architecture of Different Models

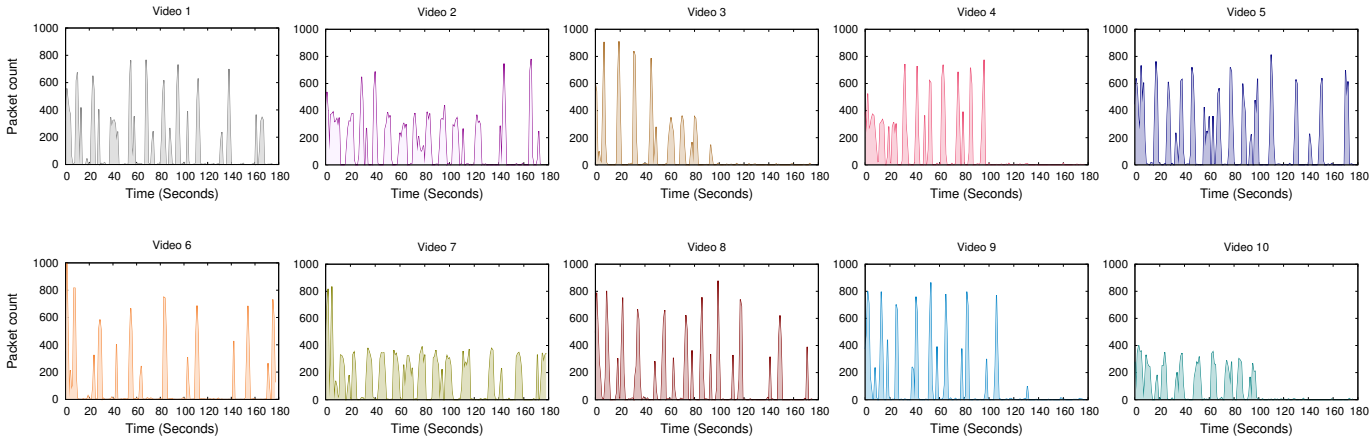


Figure 4. I/O Graphs of a single run for 10 Different Videos

results. In this paper, the features were evaluated by experiments and other hyper-parameters were selected from our experience.[4]

The results demonstrate that the MLP structure with 500×1 input (i.e. only one feature was selected) and 2 hidden layers could provide excellent performance (shown in Figure 5).

We observe from Figure 5 that, surprisingly, feature F3 "Number of Packets (non-data)" achieved the best performance not only on down-link traffic but also on bi-direction traffic. Examining up-link traffic, feature F2 "Number of Bytes (data)" performed best. Combining the traffic direction and feature type, the "Number of Packets (non-data)" on down-link traffic of video streaming trained the most accurate model (accuracy of 97.5%). Note that increasing the number of hidden layers and changing the position of the dropout layer did not achieve any further gain in performance.

In order to further understand the performance of the MLP classifier, we present in Figure 6(c) the confusion matrix for all ten videos based on the optimum model.

Other feature combinations were also validated using an MLP model with an input vector of $500 \times k$ entries where k denotes the number of feature. The input vector was constructed by splicing the 500×1 vectors of different features together. The performance was lower than those obtained using a single feature.

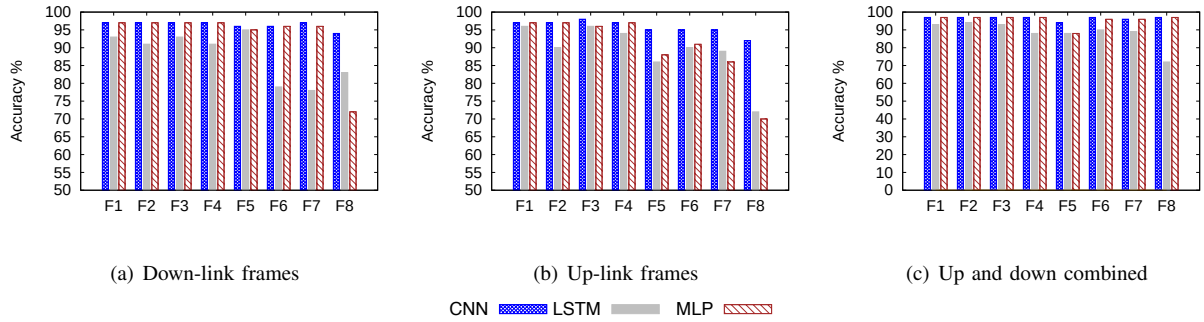
[4]The complete list of hyper-parameters is given in the Appendix.

D. Result Analysis

In this paper, three neural network architectures were implemented and used to train a video classification model. The results aforementioned showed that all models achieve similar performance. Next, we analyze the three deep learning architectures in detail and clarify limitations of our models.

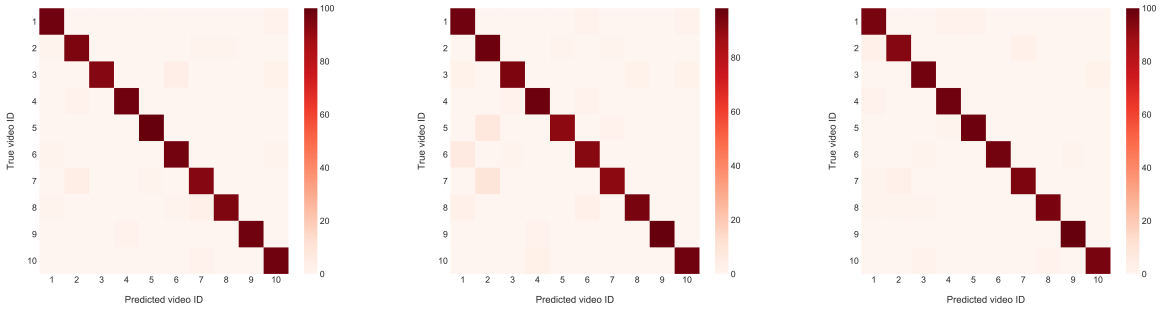
Firstly, we focus on the optimal model structure to explore the underlying factors for acquiring high MLP performance. In the MLP architecture, each of F1, F2, F3 and F4 which are directly obtained from the captured traffic files could achieve excellent performance on down-link, up-link and combination of bi-directional link.

Intuitively, it is expected that the number of packets in the data frame provides the best performance because the video contents are encapsulated in data frames. This result can be explained by the number of packets per second of different videos as shown in Figure 4. This figure validates that the fluctuation of "number of packets" along with time is a crucial



(a) Down-link frames (b) Up-link frames (c) Up and down combined

Figure 5. Accuracy of various Neural Network Models



(a) CNN (b) LSTM (c) MLP

Figure 6. Classification Performance Confusion Matrix for Models with F1 (number of packets (data) on down-link)

characteristic to distinguish between videos. This was also demonstrated in [2] for the case of a wired network capture. On the other hand, it is worth noting that non-data frames can also be exploited to classify video streaming accurately. This demonstrates that not only the bursts generated by video content, but also the interaction information between server and client carry information.

In addition, the performance of other features (F5, F6, F7 and F8) generated from the original captured data are not stable. From Figure 1, it is clear that the traffic waveforms of the same video captured at different times share many common characteristics, especially the time sequence of wave peak and wave trough. However, the amplitude variation of these waves is dependant on the WiFi signal environment due to the DASH rate adaptation. Therefore, the probable reason that the generated features could not provide steady or reliable performance is that the noise would be magnified by the generated features. This would result in overfitting during the model training.

Considering that our traffic information used is not video content but traffic attributes similar to those extracted from the convolutional layer, the MLP model requires only 2 hidden layers similar to the fully connected layers in CNNs. This is because the potential relationship between features and attributes do not need to be represented in a complex manner.

Next, we analyze the performance of both CNN and LSTM models. CNNs are advantageous in the image processing field [20]. However, seeking suitable and related features to generate meaningful convolution in other application domains is a challenge, in spite of prior work [2]. It is worth noting that the paper [2] stated significant noise in traffic as a limitation of their model. As we sniff the traffic signal using AirPcap, the same drawback applies.

Moving to the LSTM model, as stated in Sec IV, LSTM schemes can classify dynamic time sequence behavior and can process arbitrary input series of time sequence using its internal memory. Video streaming is by definition a time sequence, hence our consideration of LSTM. Even though we do not use video content but focus on fingerprints extracted from the original video, the performance of the LSTM model is considered satisfactory.

To demonstrate time independence of the MLP model, we captured the traffic flows for the same 10 videos after 2 weeks and tested it again on the MLP model. The results are shown in Table II. It is clear that the performance of the model on new test sets and old test sets is maintained.

In order to better visualise the optimal MLP classification, we apply t-SNE on the output of the last hidden layer of the original dataset and present the result in Figure 7. In this figure, we can see that some videos can overlap, in particular videos 2 and 7. This can be explained by their relatively close streaming pattern as shown in Figure 4.

There were two assumptions during data collection. First, the starting point of the video is supposed known; and second

Table II
MLP MODEL ACCURACY WITH NEW DATASET TWO WEEKS LATER

Feature Name	Down-link	Up-link	Combined
Number of Packets (data)	0.95667	0.96334	0.97334
Number of Bytes (data)	0.95667	0.96334	0.97000
Number of Packets (non-data)	0.95334	0.96334	0.96000
Number of Bytes (non-data)	0.97000	0.95667	0.96000
Minimum packet size	0.95000	0.85000	0.86334
Maximum packet size	0.95000	0.89667	0.95667
Average packet size	0.97000	0.81667	0.97000
Variance packet size	0.67334	0.67667	0.97334

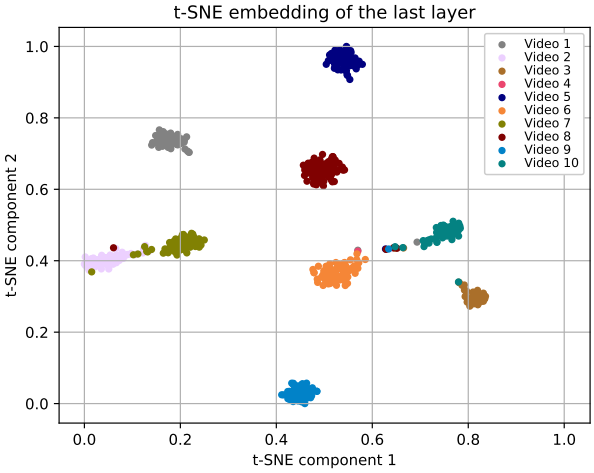


Figure 7. t-SNE Embedding of the Last Layer

the advertisements in the very beginning are deleted as a member of YouTube Red. Although it is essential to estimate and identify the video start point in a real scenario, to simplify the data collection, we ignored this procedure. In addition, data capture time is fixed to 3 minutes but the length of advertisement is variable. For this reason, we remove advertisements to minimise their potential interference. In future work, we aim to develop methods to address these issues, such as using traffic type classification techniques and increasing data capture time.

Finally, one question remains: which architecture should be used in a real-world scenario? To shed light to this question, we present in Figure 8 the training acceleration,[5] of MLP compared to the state-of-the-art, with similar performance, CNN model. As shown in this figure, the MLP model is at least three times faster to train compared to the CNN model.

To summarise, according to empirical analyses and experimental validation, we represent extracted traffic information from video streaming as a waveform-like signal and encode the waveform utilising a well trained MLP model with 2 hidden layers. This model is then applied to test datasets acting as a decoder to identify video streaming. The selected features together with the MLP neural network model achieve accuracy of 97% while consuming three time less computational resources to train.

[5]We defined the training time as the ratio of training time with CNN over the training time with MLP

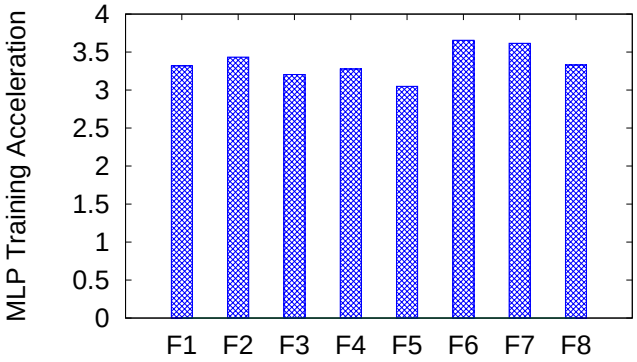


Figure 8. Computational Acceleration with MLP instead of CNN

VI. CONCLUSION

In this paper, we investigated the possibility of discovering video-streaming content from passively observed WiFi traffic that is encrypted at both transport layer (TLS) and MAC layer (WPA2). In order to unveil this WiFi video traffic, we proposed two types of neural network, namely a Recurrent Neural Network and a Multi-Layer Perceptron, and a reference implementation of a Convolutional Neural Network [2] to analyse the captured traffic. Overall, we have demonstrated that by leveraging the particular DASH pattern of each video, the MLP model was able to achieve 97% accuracy in identifying videos from a closed set of 10 videos in encrypted WiFi traffic. This high accuracy was later re-evaluated when, two weeks after the original data collection, we collected a new set of data and using the same original model we were able to obtain similar performance. Thus demonstrating the robustness of our approach.

Additionally, we have also shown that while the state-of-art CNN model proposed for the same task in the IP layer [2] was suitable for a wireless environment, it needs significantly more computational resources than our proposed architecture using MLP.

As future work, we are currently developing a prototype that leverages our proposed model, capable of identifying video traffic and thus video content within a live network containing a large number of concurrent flows.

ACKNOWLEDGEMENTS

This work was conducted in partnership with the Defence Science & Technology Group and Data61/CSIRO, through the Next Generation Technologies Program.

REFERENCES

[1] Gennie Gebhart, "We're Halfway to Encrypting the Entire Web," https://www.eff.org/deeplinks/2017/02/were-halfway-encrypting-entire-web, 2017, online; accessed 02-07-2018.

[2] 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)*. Vancouver, BC: USENIX Association, 2017, pp. 1357–1374. [Online]. Available: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/schuster

[3] R. Alshammari and A. N. Zincir-Heywood, "Investigating two different approaches for encrypted traffic classification," in *Sixth Annual Conference on Privacy, Security and Trust*. IEEE, 2008, pp. 156–166.

[4] S. E. Coull and K. P. Dyer, "Traffic analysis of encrypted messaging services: Apple imessage and beyond," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 5, pp. 5–11, 2014.

[5] R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham *et al.*, "Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation," in *DARPA Information Survivability Conference and Exposition, 2000. DISCEX'00. Proceedings*, vol. 2. IEEE, 2000, pp. 12–26.

[6] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009. IEEE Symposium on*. IEEE, 2009, pp. 1–6.

[7] Q. Niyaz, W. Sun, A. Y. Javaid, and M. Alam, "A deep learning approach for network intrusion detection system," in *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (Formerly BIONETICS), BICT-15*, vol. 15, 2015, pp. 21–26.

[8] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning: transfer learning from unlabeled data," in *Proceedings of the 24th international conference on Machine learning*. ACM, 2007, pp. 759–766.

[9] T. Ma, F. Wang, J. Cheng, Y. Yu, and X. Chen, "A hybrid spectral clustering and deep neural network ensemble algorithm for intrusion detection in sensor networks," *Sensors*, vol. 16, no. 10, p. 1701, 2016.

[10] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for network intrusion detection in software defined networking," in *Wireless Networks and Mobile Communications (WINCOM), 2016 International Conference on*. IEEE, 2016, pp. 258–263.

[11] J. Kim, J. Kim, H. L. T. Thu, and H. Kim, "Long short term memory recurrent neural network classifier for intrusion detection," in *Platform Technology and Service (PlatCon), 2016 International Conference on*. IEEE, 2016, pp. 1–5.

[12] B. Dong and X. Wang, "Comparison deep learning method to traditional methods using for network intrusion detection," in *Communication Software and Networks (ICCSN), 2016 8th IEEE International Conference on*. IEEE, 2016, pp. 581–585.

[13] U. Fiore, F. Palmieri, A. Castiglione, and A. De Santis, "Network anomaly detection with the restricted boltzmann machine," *Neurocomputing*, vol. 122, pp. 13–23, 2013.

[14] A. K. J. Michael, E. Valla, N. S. Neggatu, and A. W. Moore, "Network traffic classification via neural networks," 2017.

[15] M.-J. Kang and J.-W. Kang, "Intrusion detection system using deep neural network for in-vehicle network security," *PloS one*, vol. 11, no. 6, p. e0155781, 2016.

[16] M. Farsi, K. Ratcliff, and M. Barbosa, "An overview of controller area network," *Computing & Control Engineering Journal*, vol. 10, no. 3, pp. 113–120, 1999.

[17] M. Lotfollahi, R. S. H. Zade, M. J. Siavoshani, and M. Saberian, "Deep packet: A novel approach for encrypted traffic classification using deep learning," *CoRR*, vol. abs/1709.02656, 2017. [Online]. Available: http://arxiv.org/abs/1709.02656

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

[19] J. Gu, J. Wang, Z. Yu, and K. Shen, "Walls have ears: Traffic-based side-channel attack in video streaming," in *Proc. of IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, 2018.

[20] Y. LeCun *et al.*, "Lenet-5, convolutional neural networks," *URL: http://yann. lecun. com/exdb/lenet*, p. 20, 2015.

APPENDIX

We present in Table III the list of videos used in this article as well as their category as per YouTube metadata.

Table III
LIST OF YOUTUBE VIDEO

Video ID	YouTube ID	Video Category	Length	HD
Video 1	d853h-8rsPQ	Entertainment	11:11	Yes
Video 2	sywaPf021oE	Sports	13:27	Yes
Video 3	6rzlfG6Xkg0	Sports	5:31	Yes
Video 4	95M8W1JgH_0	Entertainment	5:03	Yes
Video 5	QU1byle-nmA	Sports	7:21	Yes
Video 6	gPHVLxm8U-0	Entertainment	5:12	Yes
Video 7	C3ap6S4mAco	Music	4:09	Yes
Video 8	kbMqWXnpXcA	Music	6:05	Yes
Video 9	XruwUFiK8YA	How-to & Style	10:03	Yes
Video 10	4bPezggGBa8	Comedy	4:53	Yes

Finally, we summarise in Table IV the various neural networks parameters that we used across the three models.

Table IV
LIST OF NEURAL NETWORKS PARAMETERS

Parameter Name	Value
learning rate	0.0001
batch size	64
activation	RELU
optimiser	Adam
Batch Normalisation decay	0.5
Batch Normalisation epsilon	0.001