A time-series classification approach to shallow web traffic de-anonymization

Axel De Nardin[1], Marino Miculan[1], Claudio Piciarelli[1] and Gian Luca Foresti[1]

[1]*Department of Mathematics, Computer Science and Physics, University of Udine*

Abstract

Web traffic analysis and classification has been extensively studied, both with classical and deep learning techniques. Many of these systems analyse the entire packet to perform the classification task. Due to the increase of encrypted traffic in recent years, this approach has become problematic. Moreover, few works focus on the classification of the users themselves, also called *web traffic de-anonymization.* In the present paper we address this problem by proposing an approach focused on a *shallow, temporal analysis* of web traffic data packets. We show that it is possible to identify the users of a network just by analyzing their navigation patterns and without accessing the content of the TCP packets. Finally, we propose a comparison between the performance of our approach and a more classical feed forward neural network architecture to showcase the informational power of temporal data in this context.

Keywords

Temporal Analysis, User De-Anonymization, Shallow packet inspection, Network traffic analysis

1. Introduction

The importance of being able to identify the users accessing the Internet, both for commercial and forensic purposes, cannot be overstated. Being able to categorize the users at different levels and by different points of view (e.g. application used, OS, browser, demographic) is important both for commercial uses, where it can be used to perform a profiling of the identified users and offer them more personalized services, and for forensic purposes where it can be applied to identify individuals performing criminal actions.

A common approach to this problem, called *web traffic de-anonymization*, is to look for useful information (e.g. usernames, email addresses) inside the payloads of IP packets. This well known technique, called *deep packet inspection* (DPI), can be very effective [1, 2], but it can be applied only if the traffic is not encrypted. Nowadays almost all web traffic (especially that carrying identification data) is encrypted at the transport level by means of SSL and TLS protocols, and therefore DPI is rarely applicable. Moreover, DPI raises important privacy issues, because it allows the inspector to access the whole traffic content, not only the data needed for user identification [3].

Therefore, the ability to identify the users generating web traffic on a network without looking at the actual payload but only performing a *shallow packet inspection*, is gaining importance.

ITASEC'21: Italian Conference on Cybersecurity, April 07–09, 2021, Online

denardin.axel@spes.uniud.it (A. D. Nardin); marino.miculan@uniud.it (M. Miculan); claudio.piciarelli@uniud.it (C. Piciarelli); gianluca.foresti@uniud.it (G. L. Foresti)

However, while many works have already been proposed regarding the task of classifying (encrypted) web traffic, there is little research on web traffic de-anonymization through shallow packet analysis; see e.g. [4]. In particular, to our knowledge there is no previous work focusing on *temporal* analysis applied to this specific area.

In this paper we aim specifically to this type of analysis. To this end, we introduce the use of recurrent neural network models, specifically LSTMs, in order to gain a better insight regarding the impact of the temporal component in this scenario. To achieve a better understanding about the actual performance gain when using time series, instead of single web traffic logs, as instances of the dataset we also compare the temporal models with a classic Feed Forward neural network of similar size.

Synopsys. The rest of the paper is organized as follows. An overview of related work about network traffic classification is in section 2. Then, in section 3 we introduce the problem we focus on. In section 4 we outline the proposed approach, by describing the models used and the way data collection has been performed. The experimental results are reported in section 5 and, finally, in section 6 we summarize our work and propose some ideas for future work.

2. Related work

When tackling a classification problem it is important to determine which aspect we aim to classify. Many different approaches have been investigated for web traffic classification focusing on different categorization goals: identifying the protocol used [5], application classification [6, 7, 5], traffic type (e.g. browsing vs. video chat) [8, 9], OS classification [7] and intrusion detection [10].

Various network architectures and techniques have also been explored. One widely adopted idea in recent works is to try to map the traffic data to a bi-dimensional image and then use a CNN to extract the spatial information from it, as proposed in [11]. However it was noted than when trying to extract features from time series, temporal information loss occurs in the convolutional and pooling layers. The use of architectures such as C-LSTM ([12]) combining both convolutional and recurrent layers, specifically LSTMs, has also been investigated and was able to achieve state-of-the-art performance for web traffic anomaly detection on the Webscope S5 dataset by Yahoo. One thing to notice, however, is that many of the presented works do not rely uniquely on a shallow inspection of the transmitted packets but use also consider their payload to perform the classification task, according to the well known *deep packet inspection*. Moreover, only few of these techniques take into account the temporal aspect of the data alone and rely for the most part on datasets which have single logs as their instances. Finally, while previous works have explored the use of classification techniques for different purposes related to web traffic analysis, the area of user identification is still under investigated; see e.g. [4].

3. Problem description

In order to recognize users by means of shallow packet inspection, we adopt the architecture depicted in Figure 1. The whole network traffic is logged by a sniffer and subsequently filtered

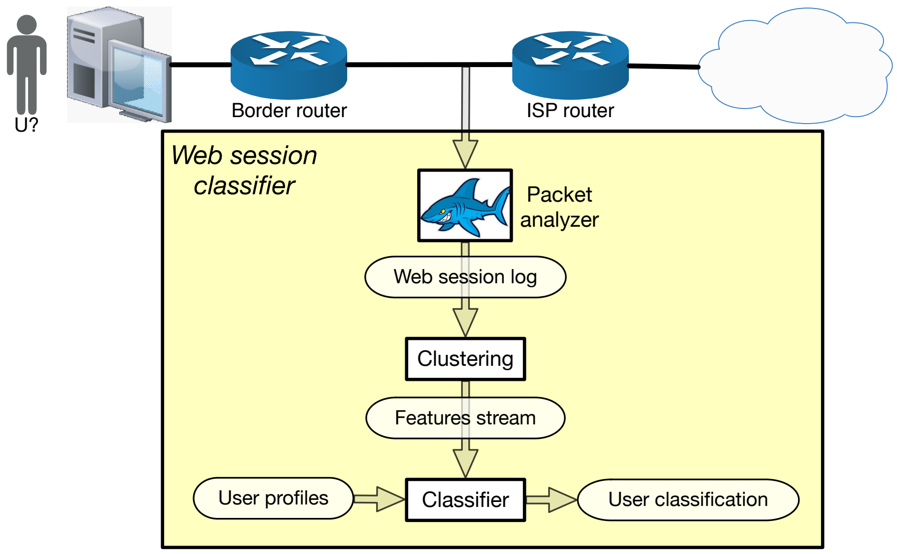

Figure 1: Architecture of the web session classifier (from [4]).

and pre-processed to collect only the data relevant for the system. In order to preserve user privacy, pre-processing also replaces source IP addresses with unique identifiers $(u_1, u_2, \ldots)$. Hence, despite the system internally stores the address/identifier associations (which are needed to guarantee a coherent labeling through time), the final data are pseudonymised.

Following the formalization given in [4] the problem can be defined as:

> given a training set TS for users $u_1, \ldots u_n$ and a web session log L generated by one of these users, is it possible to determine which user has generated L?

where the training set is defined as:

$$TS = \{\langle L_1, u_{i_1} \rangle, \ldots, \langle L_k, u_{i_k} \rangle\}. \tag{1}$$

Here, i_k is the index of the user generating log L_k, and each log L consists in a subset of data extracted from the TCP/IP packed header. In this work, since our main goal is investigating the role of the temporal aspect in the user classification task, we redefine the training set as:

$$TS = \{\langle S_1, u_{i_1} \rangle, \ldots, \langle S_k, u_{i_k} \rangle\} \tag{2}$$

where S_i now identifies a sequence of web traffic logs instead of a single one. Thus we obtain the following revisitation of the previous definition:

> given a training set TS for users $u_1, \ldots u_n$ and a temporal sequence of web session logs S generated by one of these users, is it possible to determine which user has generated S?

4. Proposed approach

In this section an approach for the aforementioned problem will be introduced. We start by describing the way data is collected and pre-processed, then we give an outline of the models and techniques used for the identification task.

4.1. Data collection

All the data used for the training and testing of the adopted models have been retrieved by shallow-sniffing a large WiFi network during a time window of a couple hours. This allowed to retrieve a dataset of around 6.1 million instances each represented by a tuple consisting of 6 elements:

- timestamp
- source MAC address
- destination IP address
- packet length
- TCP source port
- TCP destination port

The privacy of the users has been preserved by running a pseudonymization process on the dataset, which mapped each MAC address to a progressive id which has been kept consistent across the different instances. No data regarding the content of the packets has been sniffed.

4.2. Data preprocessing

Three major pre-processing operation have been performed on the collected data. The first one is represented, as already mentioned, by the mapping of the MAC address of each detected user to a progressive and consistent numerical ID which is then used as the label of the corresponding instance. The same process has been applied to destination IP addresses. The second one, on the other hand, is performed on the timestamps of the connection. Each timestamp has been replaced with a value Δ_t which is defined as the difference between the time at which was performed the current log and the the time of the previous log of the same user. The idea justifying this transformation is that, when analyzing the behavior of a user, we are more interested in the interval occurring between his actions rather than in the absolute time at which they have been performed. Finally the last pre-processing step consisted in the standardization of the feature values in order to speed up the convergence of the models during training.

4.3. Data selection

Through a preliminary analysis of the data distribution we noticed a significant imbalance in the number of instances belonging to each user, with a single class completely dominating the dataset, as shown in Fig. 2. For this reason we decided to use two different settings, regarding the dataset used, for all the experiments. In the first setting we used the whole dataset represented by 6.1M instances distributed over 151 classes, as already presented, while in the second one we

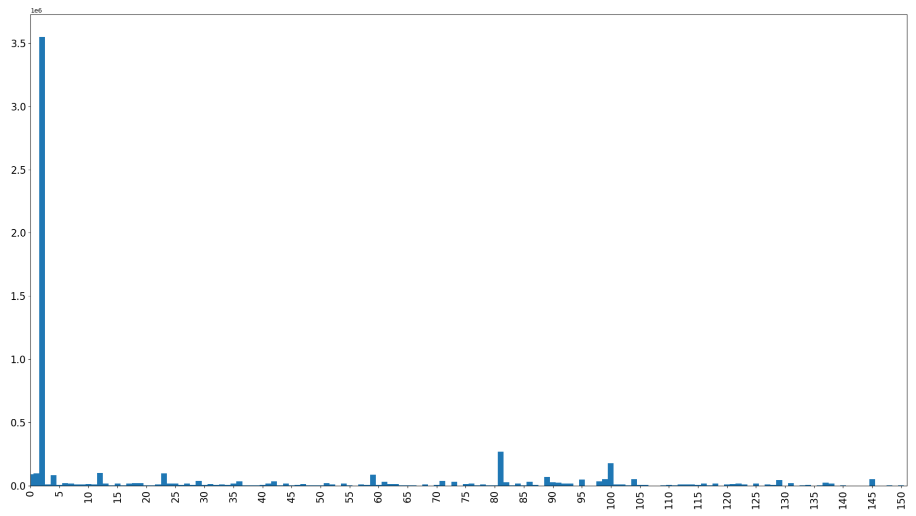

Figure 2: Instances distribution over the classes of the full dataset

decided to filter out the classes that contributed the most to the imbalance in the distribution, in particular we removed the classes consisting in more than 150k elements and those represented by less then 5k elements (Fig. 3). The resulting dataset consisted in 2M instances distributed over 90 classes. This allowed to analyze the performance of the proposed methods on problems of different scales and also to understand how class imbalance affected them. Another issue concerns, instead, the features describing the destination IP address. While this piece of data seems to be very valuable in terms of information provided to the classifier, due to its categorical nature it is more challenging to find a meaningful way to represent it in a consistent way and to deal with situations, which are likely to occur in a real world scenario, in which the training and test sets don't necessarily contain instances with the same subset of IP addresses. For this reason in the present work we decided to analyze the performance of the selected models in two different settings for both the previously presented datasets, one in which the IP address is used as an input feature (by simply mapping it to a consistent numerical ID) and one in which it is discarded. This choice allowed us to understand how much impact this particular feature has on the classification capabilities of the models and if it is possible to obtain good performance even without including it.

4.4. Models

All the collected data, organized in the settings described in the previous section, has been used to train a set of Deep Learning classifiers. Two different architectures have been used, the first one is represent by a 3-layer feed forward network which was used as a baseline for the performance that can be obtain without taking in consideration the temporal aspect of the data

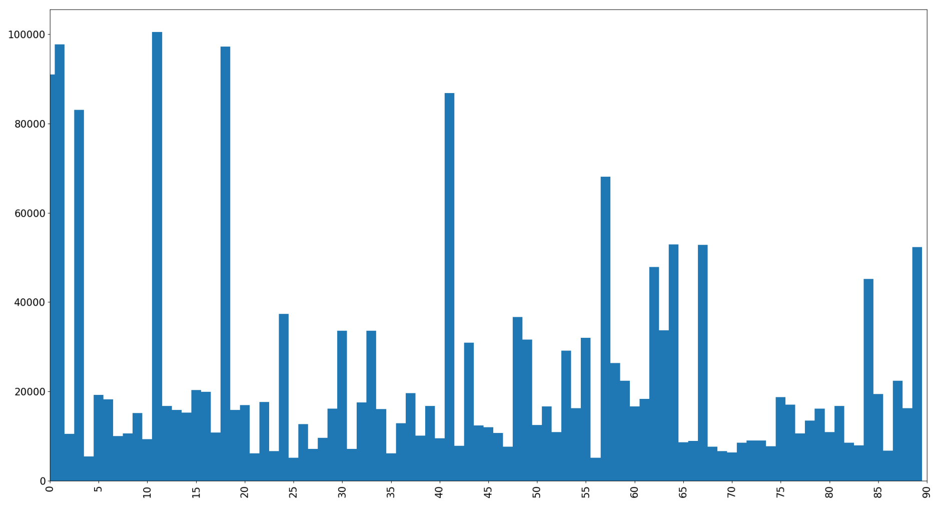

Figure 3: Instances distribution over the classes of the filtered dataset

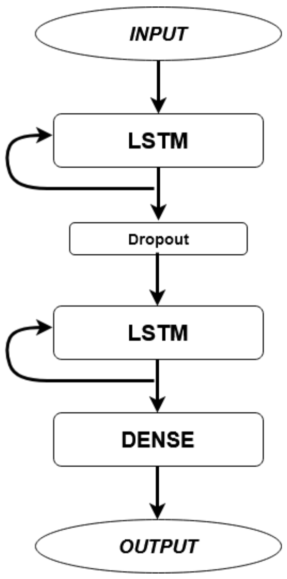

Figure 4: Diagram of the Recurrent Neural network used in the experiments

collected. The second architecture, which is the main focus of the present work, is a recurrent neural network composed by 2 stacked LSTM [13] layers, with a 50% dropout layer in between them, and ending with a fully connected layer used to perform the actual classification task. A diagram of this network is shown in Fig. 4.

Table 1
Dataset settings used in the different training scenarios

	# instances	# classes	destination ip
full_nodest	6.1M	151	no
full	6.1M	151	yes
reduced_nodest	2M	90	no
reduced	2M	90	yes

Table 2
models hyperparameters

	Learning rate	Batch size	# epochs	Sequence length
Feed_FW	1e^-4	1024	50	1
LSTM_5	1e^-4	1024	50	5
LSTM_10	1e^-4	1024	50	10
LSTM_20	1e^-4	1024	50	20

5. Experimental results

All the models have been trained on the 4 different dataset settings which have been described in the previous section and are summarized in table 1.

A summary of the hyper-parameters used during the training process is reported in table 2. While the main focus of the present work wasn't to find the optimal values for these parameters, we investigated different choices to make sure that they led to reasonable results while at the same taking into consideration the time needed for the training of the models.

We tried to preserve the consistency of the hyper-parameters selected for the different models analyzed in order to be able to make a more meaningful comparison between them. The only exception being the sequence length, which represents the temporal aspect we wanted to investigate with this work and therefore is changed across the different instances of the model to gain a better understanding on how it affects their performances. In all the scenarios a 60/40 split of the datasets was performed, using the 60% of the data to train the model and the remaining 40% for the testing process. The metrics used to evaluate the different models are 4:

- **Accuracy**
- **Recall**
- **Precision**
- **F-Score**

The results shown for these metrics (table 3), with the exception of the accuracy, have been calculated by macro-averaging the values obtained for the single classes.

As it can be seen, the classic feed forward network, trained on the single instances of the datasets, achieve a much worse performance than every recurrent neural network, with differences up to 50% between the values of some of the metrics when using the smaller, more balanced, data set. This seems to imply that the temporal aspect of the data has actually an impact on the ability of the models to recognize the different users. This idea is also supported by

Table 3
Classification results.

Classifier	Dataset	Accuracy (%)	Precision (%)	Recall (%)	F-Score (%)
Feed_FW	Reduced_nodest	49.15	30.64	27.66	26.00
	Reduced	63.54	49.20	52.25	46.60
	Full_nodest	77.18	14.82	16.95	12.80
	Full	84.26	29.92	31.93	32.90
LSTM_5	Reduced_nodest	78.82	71.91	73.74	71.80
	Reduced	90.77	87.04	88.01	87.20
	Full_nodest	86.63	38.55	38.55	38.00
	Full	95.35	67.00	75.55	68.20
LSTM_10	Reduced_nodest	86.51	82.00	82.96	82.10
	Reduced	95.21	93.29	93.65	93.40
	Full_nodest	90.39	49.52	49.53	49.80
	Full	97.98	82.04	88.51	83.90
LSTM_20	Reduced_nodest	89.00	84.88	85.68	85.00
	Reduced	97.00	**95.84**	**95.98**	**95.90**
	Full_nodest	94.98	66.75	72.40	67.00
	Full	**98.85**	87.95	92.34	89.10

the fact that the models performance is positively correlated with the length of the sequences in which the data is organized, reaching their peak when using the largest value for this parameter (Fig. 5). Another interesting thing to notice is that while the accuracy of all the models increases when using the full dataset, compared to when the reduced one is used, the values of the other three metrics seem to greatly deteriorate instead. This suggests that the models are heavily influenced by the imbalances between the number of instances belonging to the different classes of the full dataset and have the tendency to specialize towards the more populated ones, neglecting the others. This phenomenon seems to be partially mitigated by the introduction of the destination IP address as one of the features during training, which leads to an improvement of the performance over all the metrics adopted and also reduces the gap between the accuracy and the other metrics values, especially when the full dataset is used.

6. Conclusions

In this paper we investigated the impact of introducing a temporal component in the context of web traffic classification. More specifically we tried to determine if analyzing sequences instead of isolated web logs allows to make more accurate predictions when performing a shallow packet inspection with the goal of de-anonymizing users. The use of a shallow inspection is motivated by an ever increasing adoption of encrypted connections on a technological side and by privacy concerns on a human and legal one.

While the obtained results clearly point out that the temporal aspect has a great impact on

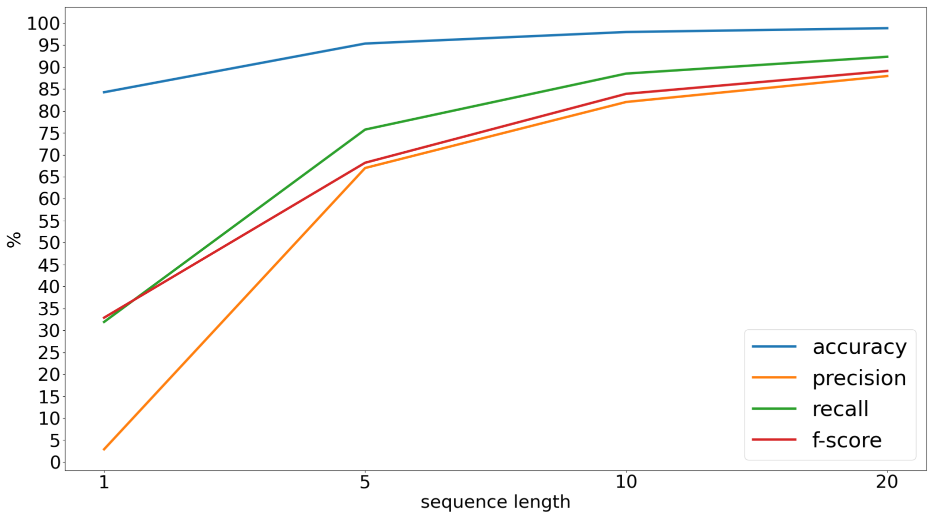

Figure 5: Results obtained for the different metrics when using different values of sequence length and training the model on the full dataset

the performance of web traffic classification models by also mitigating, in a certain measure, the negative effects of a highly unbalanced distribution of the instances of the dataset, further investigation is possible in the future since only a narrow category of network architectures has been considered, leaving out solutions that have been proven to be effective for other applications related to web traffic analysis, such as the combination of LSTMs and CNNs. Another aspect we plan to address in future work regards an improvement in the heterogeneity of the data. While the dataset we used for the experiments was reasonably large, we suspect it may not be too representative of a real world scenario as all the logs were collected during a single session spanning over just a few hours, therefore there is a high possibility that the retrieved data is highly redundant. Finally we plan to improve the way categorical features, such as the IP addresses, are represented since we believe that providing a more semantically meaningful embedding could improve further the performance of the models.

References

[1] S. Kumar, J. Turner, J. Williams, Advanced algorithms for fast and scalable deep packet inspection, in: Architecture for Networking and Communications systems, 2006. ANCS 2006. ACM/IEEE Symposium on, IEEE, 2006, pp. 81–92.

[2] C. Parsons, Deep Packet Inspection in Perspective: Tracing its lineage and surveillance potentials, Surveillance Studies Centre, Queen's University, 2008.

[3] A. Daly, The legality of deep packet inspection, International Journal of Communications Law & Policy (2011). doi:10.2139/ssrn.1628024.

[4] M. Miculan, G. L. Foresti, C. Piciarelli, Towards user recognition by shallow web traffic inspection, in: P. Degano, R. Zunino (Eds.), Proceedings of the Third Italian Conference on Cyber Security, Pisa, Italy, February 13-15, 2019, volume 2315 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2019.

[5] Z. Chen, K. He, J. Li, Y. Geng, Seq2img: A sequence-to-image based approach towards ip traffic classification using convolutional neural networks, in: 2017 IEEE International Conference on Big Data (Big Data), 2017, pp. 1271–1276. doi:10.1109/BigData.2017.8258054.

[6] G. Aceto, D. Ciuonzo, A. Montieri, A. Pescapé, Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges, IEEE Transactions on Network and Service Management 16 (2019) 445–458. doi:10.1109/TNSM.2019.2899085.

[7] S. Rezaei, X. Liu, How to achieve high classification accuracy with just a few labels: A semi-supervised approach using sampled packets, 2020. arXiv:1812.09761.

[8] W. Wang, M. Zhu, J. Wang, X. Zeng, Z. Yang, End-to-end encrypted traffic classification with one-dimensional convolution neural networks, 2017, pp. 43–48. doi:10.1109/ISI.2017.8004872.

[9] M. Lotfollahi, R. Shirali hossein zade, M. Jafari Siavoshani, M. Saberian, Deep packet: A novel approach for encrypted traffic classification using deep learning, Soft Computing 24 (2020). doi:10.1007/s00500-019-04030-2.

[10] W. Wang, Y. Sheng, J. Wang, X. Zeng, X. Ye, Y. Huang, M. Zhu, Hast-ids: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection, IEEE Access 6 (2018) 1792–1806. doi:10.1109/ACCESS.2017.2780250.

[11] W. Wang, M. Zhu, J. Wang, X. Zeng, Z. Yang, End-to-end encrypted traffic classification with one-dimensional convolution neural networks, in: 2017 IEEE International Conference on Intelligence and Security Informatics (ISI), 2017, pp. 43–48. doi:10.1109/ISI.2017.8004872.

[12] T.-Y. Kim, S. Cho, Web traffic anomaly detection using c-lstm neural networks, Expert Systems with Applications 106 (2018). doi:10.1016/j.eswa.2018.04.004.

[13] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (1997) 1735–1780. doi:10.1162/neco.1997.9.8.1735.