Explaining Class-of-Service Oriented Network Traffic Classification with Superfeatures

Sayantan Chowdhury
Dept. of Electrical and Computer Engineering
University of Toronto
Canada
sayantan.chowdhury@mail.utoronto.ca

Ben Liang
Dept. of Electrical and Computer Engineering
University of Toronto
Canada
liang@comm.utoronto.ca

Ali Tizghadam
Technology Strategy and Business Transformation
TELUS Communications
Canada
ali.tizghadam@telus.com

ABSTRACT

Recent studies have demonstrated that machine learning can be useful for application-oriented network traffic classification. However, a network operator may not be able to infer the application of a traffic flow due to the frequent appearance of new applications or due to privacy and other constraints set by regulatory bodies. In this work, we consider traffic flow classification based on the class of service (CoS), using delay sensitivity as an example in this preliminary study. Our focus is on direct CoS classification without first inferring the application. Our experiments with real-world encrypted TCP flows show that this direct approach can be substantially more accurate than a two-step approach that first classifies the flows based on their applications. However, without invoking application labels, the direct approach is more opaque than the two-step approach. Therefore, to provide human understandable interpretation of the trained learning model, we further propose an explanation framework that utilizes groups of superfeatures defined using domain knowledge and their Shapley values in a cooperative game that mimics the learning model. Our experimental results further demonstrate that this explanation framework is consistent and provides important insights into the classification results.

CCS CONCEPTS

• **Networks** → *Network management*; *Network monitoring*.

KEYWORDS

Traffic classification, class of service, machine learning, explanation framework, Shapley values

ACM Reference Format:
Sayantan Chowdhury, Ben Liang, and Ali Tizghadam. 2019. Explaining Class-of-Service Oriented Network Traffic Classification with Superfeatures. In *3rd ACM CoNEXT Workshop on Big DAta, Machine Learning and Artificial Intelligence for Data Communication Networks (Big-DAMA '19), December 9, 2019, Orlando, FL, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3359992.3366767

Big-DAMA '19, December 9, 2019, Orlando, FL, USA

ACM ISBN 978-1-4503-6999-2/19/12.
https://doi.org/10.1145/3359992.3366767

1 INTRODUCTION

Traffic classification is an essential service to ensure proper network management, achieve efficient resource allocation, and resolve critical security issues. Since classical methods of identifying Internet traffic based on ports and protocol content often fail for modern encrypted traffic, many researchers have proposed machine learning approaches instead [3, 10, 20]. A wide range of techniques have been studied, including Bayesian estimation [4, 18], unsupervised clustering [8, 30], support-vector machine (SVM) [9], and neural network (NN) [5, 14, 15, 28]. There are also several hybrid [29] and online variants [11, 21, 31].

Most existing works focus on application-oriented classification, assigning labels to traffic flows in terms of their applications. However, in practice, the network operator often only needs to know the required class-of-service (CoS) of a flow, e.g., delay sensitivity, to make decisions on traffic priority and resource allocation. For such CoS-oriented classification, a *two-step approach* can be used, where we first estimate a flow's application using any of the aforementioned techniques and then translate the application label to a CoS label based on some pre-determined mapping.

However, it is often difficult to keep up with the many new applications appearing daily, particularly to identify zero-day applications [32]. Furthermore, a network operator may not wish to infer the application of a flow, e.g., for compliance with privacy or network-neutrality regulation. Therefore, a better alternative is to directly assign CoS labels to traffic flows without learning their application. Such a *direct approach* has been studied in [24] using nearest-neighbor clustering and in [27] using semi-supervised SVM. Since the number of service classes generally is much smaller than the number of applications, and the service classes change far less frequently than applications do, the direct approach can potentially achieve higher efficiency. Indeed, in this paper, we will demonstrate with real-world traffic that the direct approach is substantially more accurate than the two-step approach.

Nevertheless, without invoking application labels, the direct approach is more opaque than the two-step method. However, the interpretability of a classifier is important in many scenarios [6]. One cannot rely on a black-box model for traffic classification in high-stake applications, such as network security, self-driving cars, and emergency services. Furthermore, erroneous performance analysis might give us false confidence in the learning model. There are many causes for a classifier to give false confidence or incorrect results, such as data leakage and unintentional bias in the dataset,

and these causes can be challenging to detect and rectify. More fundamentally, without an explanation for how a learning model arrives at its decisions, our scientific understanding would be limited. Therefore, in addition to building a learning model that is highly accurate, we also need a means to understand the rationale behind its predictions. To the best of our knowledge, no such work yet exists for traffic classification. Furthermore, this is a particularly challenging problem for the direct approach, since it omits the intermediate stage of application identification.

In the case of linear classifiers, one may infer the amount of contribution of each feature to the predicted label by inspecting the weight associated with the feature in the learning model. However, this is impossible for more complex models that achieve higher accuracy, such as SVM and NN. Thus, we arrive at a dilemma where the more accurate models are less interpretable. The authors of [23] proposed local interpretable model-agnostic explanations (LIME), which computes a linear approximation of a complex classifier by solving a fidelity-interpretability trade-off problem in the vicinity of a given sample. Thus, LIME provides insights about the classifier's perception using a local surrogate model. However, a global surrogate model to explain a classifier can be developed borrowing from game theory, by formulating the learning model as an equivalent cooperative game with the features as players and the Shapley values of features indicating their contribution [16, 26].

It is non-trivial to extend the interpretation approach of [16, 26] to traffic classification. First, each traffic flow can have hundreds of features, while the exact computation of all Shapley values requires exponential time in the number of features. Second, it is difficult to understand the meaning of each feature with respect to the overall CoS classification, so that even if we could obtain Shapley values for them, that is still far from human understandable interpretation. Therefore, in this work, we first group the flow features into several *superfeatures* based on domain knowledge and consider them as players. Since the superfeatures are much fewer than the original features, exact calculation of their Shapley values is now possible. Furthermore, since the superfeatures correspond to flow characteristics that are meaningful to the network operator, their Shapley values improve human understanding on how the learning model arrives at its classification results.

The contributions of this work are summarized as follows:

- Using an aggregate dataset of 43590 encrypted TCP flows from [7] and [13], we extract 266 features for each flow and train learning models for CoS-oriented classification, using binary delay sensitivity as an example for the CoS labels. We show that the direct approach has more than three times lower false-negative rate than the two-step approach for delay sensitive traffic.
- We develop an efficient explanation framework based on *superfeatures* grouped by easily-understood feature characteristics such as packet information and rate information. The Shapley values of these superfeatures, in a cooperative game that represents the learning model, indicate their contribution toward the predicted CoS label.

CoS Label	Application Type	Applications
Delay Sensitive	CHAT	Facebook chat, Skype chat, Hangouts chat, ICQ chat, AIM chat
	VOIP	Facebook voice, Skype voice, Hangouts voice
Delay Tolerant	AUDIO	Spotify
	FTP	Skype file transfer, FTPS, SFTP
	MAIL	SMPTS, POP3S, IMAPS
	P2P	uTorrent, Transmission (Bittorrent)
	VIDEO	Vimeo, YouTube
	WEB	Firefox, Chrome

Table 1: Applications and their types

- We propose several analytical methods and present further experimental results to demonstrate that the proposed explanation framework is reliable and informative. Correlation analysis on the Shapley values shows superior consistency of the proposed explanations. Thus, the direct approach and the explanation framework combine to provide a highly accurate yet explainable means to traffic classification.

The rest of this paper is structured as follows. Section 2 summarizes the flow features and our learning model. Section 3 details the explanation framework and the proposed analytical methods. In Section 4, we present the experimental results and demonstrate the performance of both the direct approach and the explanation framework. Section 5 concludes the paper.

2 FEATURES AND LEARNING MODEL

The proposed learning model and explanation framework are general. However, as a real-world example for illustration, we consider a dataset that combines ISCX VPN-nonVPN (2016)[2][7] and ISCX Tor-nonTor (2016) [1][13]. It contains pcap files for 43590 encrypted TCP flows of various applications that are grouped into 8 application types as shown in Table 1. We extract 266 features for each bidirectional flow to construct the dataset, which are loosely based on those used in [19] from older captures. We show some examples of these features in Table 2.

More generally, let $\mathcal{X}$ be our training dataset containing a number of flows with K features per flow. Each flow in $\mathcal{X}$ is associated with an application and a CoS label. Let $\mathcal{A}$ be the set of application types, e.g., as shown in Table 1, and C be the set of CoS labels. We assume a deterministic mapping $D : \mathcal{A} \rightarrow C$. For simplicity of illustration, here we consider only two CoS classes: delay sensitive, which contains all flows in the VOIP and CHAT application types; and delay tolerant, which contains all other flows. This CoS designation aligns with the practices of some systems of Internet service provided by TELUS in Canada, but the applicability of our work is not limited by it.

In CoS-oriented classification, we wish to build a machine learning model h, such that given some test flow with feature vector $\mathbf{x} = [x_1, ..., x_K]^T \in \mathbb{R}^K$ as input, $h(\mathbf{x})$ returns the predicted

Superfeatures	Features (# Features)
Packet Information	Mean packet length, variance of packet length, mean payload length, variance of payload length, etc. (**55**)
Protocol Information	Source port, destination port, # SYN packets, # FIN packets, max segment size requested, max and min window sizes, etc. (**51**)
Rate Information	Throughput, mean frame rate, variance of frame rate, mean data rate, variance of data rate, etc. (**55**)
Stochastic Information	# Bursts, mean # packets in bursts, variance of # packets in bursts, mean burst length, variance of burst lengths, etc. (**52**)
Time Information	Mean interarrival time (IAT), variance of IAT, FFT of IAT, etc. (**53**)

Table 2: Superfeatures and corresponding features

probability that the flow is delay sensitive. To train this model in the *direct approach*, we use the true CoS labels of the flows in the training dataset $\mathcal{X}$. Many variants of learning models are applicable and interpretable using the proposed explanation framework, but in this paper we focus only on NN as an example for illustration. For comparison in Section 4, we also consider the *two-step approach*, where we first training an NN model to categorize the flows into application types and then use the mapping $D : \mathcal{A} \rightarrow C$ to obtain the corresponding CoS labels.

3 EXPLANATION FRAMEWORK

3.1 Explanation Based on Superfeatures

In [16, 26], classification is modeled as a cooperative game, where the set of K features are players and the prediction probability is the outcome. Then, the Shapley value [25] measures the average marginal contribution of each feature to the output prediction. However, this requires computation of the prediction probability for all coalitions of the players, i.e., all 2^K possible subsets of the features. This is intractable in flow classification, since we have hundreds of features. Furthermore, the Deep SHAP approximation approach proposed in [16] loses accuracy. Finally, even if the contribution of each feature could be computed, it would still be challenging for human inspection.

Hence, we propose to separate the features into M groups, where $M << K$, based on domain knowledge specific to network traffic flows. Each group is called a *superfeature*. For our dataset $\mathcal{X}$, we group the 266 flow-level features into 5 disjoint superfeatures as shown in Table 2, where *packet information* includes features related to packet and payload lengths, *protocol information* covers TCP-specific features, *rate information* contains throughput and data rate-related features, and *stochastic information* and *time information* contain features associated with traffic burstiness and interarrival times, respectively.

Thus, we define a cooperative game with the set of M superfeatures $\tilde{\mathbf{x}} = \{\tilde{\mathbf{x}}_1, ..., \tilde{\mathbf{x}}_M\}$ as players and the probability of being delay sensitive as payoff. The m-th superfeature $\tilde{\mathbf{x}}_m$ denotes a set of features included in this superfeature or a vector with those features

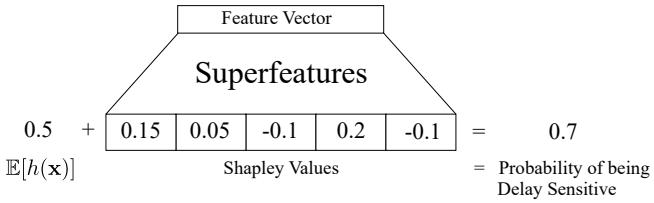

Figure 1: Example Shapley values of superfeatures

as its components. We will switch between these two interpretations of $\tilde{\mathbf{x}}_m$ for notational simplicity. Given a sample $\mathbf{x}$, the Shapley value of the m-th superfeature for the black-box model h can be obtained as

$$\phi_m(h, \mathbf{x}) = \sum_{\mathcal{S} \subseteq \tilde{\mathbf{x}} \setminus \{\tilde{\mathbf{x}}_m\}} \frac{|\mathcal{S}|!(M - |\mathcal{S}| - 1)!}{M!} (v(\mathcal{S} \cup \{\tilde{\mathbf{x}}_m\}) - v(\mathcal{S})), \tag{1}$$

where function $v(\cdot)$ computes the contribution of any coalition of superfeatures $\mathcal{S}$ marginalized over the superfeatures not in $\mathcal{S}$ as follows:

$$v(\mathcal{S}) = \mathbb{E}_{\_\mathcal{S}}[h(\mathbf{x})] - \mathbb{E}[h(\mathbf{x})], \tag{2}$$

where $\mathbb{E}_{\_\mathcal{S}}$ indicates expectation taken over the values of features included in the superfeatures not in $\mathcal{S}$ and $\mathbb{E}$ indicates expectation taken over the values of all features. Since the probability distribution from which the dataset is drawn remains unknown, we take the sample average over the training dataset to approximate the above expectations. Note that $\mathbb{E}[h(\mathbf{x})]$ gives the prior probability of being delay sensitive. Hence, if the dataset is balanced, i.e., it contains equal numbers of delay sensitive and delay tolerant flows, then $\mathbb{E}[h(\mathbf{x})] = 0.5$ for a well-trained model on this dataset.

An explanation framework based on Shapley values has several useful properties that other explanation methods do not possess [17], such as *null player*, *symmetry*, and *efficiency*. In particular, the last property suggests that, given a feature vector $\mathbf{x}$, the sum of $\mathbb{E}[h(\mathbf{x})]$ and the Shapley values of all superfeatures is equal to the payoff, i.e., the predicted probability of being delay sensitive:

$$h(\mathbf{x}) = \mathbb{E}[h(\mathbf{x})] + \sum_{m=1}^{M} \phi_m(h, \mathbf{x}). \tag{3}$$

As an example, Fig. 1 illustrates the Shapley values obtained for the 5 superfeatures of some flow sample. Added to the prior probability of being delay sensitive $\mathbb{E}[h(\mathbf{x})]$, some superfeatures have positively contributed and some have negatively contributed. As the sum of all Shapley values contributes positively over $\mathbb{E}[h(\mathbf{x})]$, the sample is predicted as delay sensitive. Therefore, we may view the Shapley value of a superfeature as the amount of probability that it contributes to our prediction. Hence, the set of M Shapley values $\{\phi_m(h, \mathbf{x})\}$ for the superfeatures of a sample $\mathbf{x}$ provides an explanation for the prediction of that sample by the classifier h.

3.2 Analysis of the Explanation Framework

In this section, we introduce several novel analytical methods to establish the reliability of the proposed explanation framework and to obtain insights about CoS-oriented classification. We note that available theoretical understanding about the features of Internet

traffic is limited, so it is challenging to establish the prior knowledge required for human understandable explanation. Therefore, we will demonstrate the reliability of the proposed explanation by evaluating the consistency of Shapley values across different feature vectors and between different approaches of classification. Experimental results of these methods will be presented in Section 4.

Let $\mathcal{E} = [\mathbf{x}^{(1)}, ..., \mathbf{x}^{(N)}]^T \in \mathbb{R}^{N \times K}$ be our explanation set that contains N correctly labeled samples for which we have obtained explanation. The set of superfeatures of the n-th sample $\mathbf{x}^{(n)}$ is given by $\{\tilde{\mathbf{x}}_1^{(n)}, ..., \tilde{\mathbf{x}}_M^{(n)}\}$. For brevity, we will rewrite $\phi_m(h, \mathbf{x}^{(n)})$, the Shapley value of the m-th superfeature of the n-th sample in the explanation set, as $\phi_m^{(n)}$. We evaluate the consistency of explanation by checking whether similarity in a particular superfeature for different samples results in similar contribution. Given the explanation set $\mathcal{E}$, let $\mathbf{x}$ be a new test sample with M superfeatures $\tilde{\mathbf{x}} = \{\tilde{\mathbf{x}}_1, ..., \tilde{\mathbf{x}}_M\}$. By comparing it with the flows in the explanation set, we expect the contribution of its m-th superfeature should be close to ψ_m defined as follows:

$$\psi_m = \left\{ \phi_m^{(n^*)} : n^* = \underset{n \in \{1,\ldots,N\}}{\arg\min} ||\tilde{\mathbf{x}}_m - \tilde{\mathbf{x}}_m^{(n)}|| \right\} \tag{4}$$

where $||\cdot||$ denotes the Euclidean distance. The index n^* in (4) may not be unique, and ties are broken uniformly randomly. Now, we define an *explained value* for $\mathbf{x}$ as $\mathbb{E}[h(\mathbf{x})] + \sum_{m=1}^{M} \psi_m$ and use it to perform two experiments. Firstly, we check the correlation coefficient between the explained value of a test sample and its probability of being delay sensitive predicted by the learning model. Note that the correlation coefficient is normalized between -1 and 1, and a high correlation indicates that the learning model is well-explained by the Shapley values. Secondly, we define an *explained label* to each test sample such that it is delay sensitive if $\sum_{m=1}^{M} \psi_m \geq 0$, i.e., the sum of the assigned Shapley values has positive contribution; and it is delay tolerant otherwise. We measure how accurately the explanations obtained from the classifier reflect the ground truth by comparing the explained labels against the true labels of the test samples.

Furthermore, to show that the explanations between the direct and two-step approaches are consistent, we consider their correlation as follows. For the direct approach, we have the vector of Shapley values of the m-th superfeature for the N samples in the explanation set $\mathcal{E}$, $\Phi_{\text{direct}}^{(m)} = [\phi_m^{(1)}, ..., \phi_m^{(N)}]^T \in \mathbb{R}^N$. In the two-step approach, for any flow, the probability of being delay sensitive equals the probabilities of being in the application types that are mapped to delay sensitive. We denote its vector of Shapley values for the m-th superfeature by $\Phi_{\text{two-step}}^{(m)}(D) \in \mathbb{R}^N$. We measure the correlation coefficient between $\Phi_{\text{direct}}^{(m)}$ and $\Phi_{\text{two-step}}^{(m)}(D)$, denoted by $\rho(\Phi_{\text{direct}}^{(m)}, \Phi_{\text{two-step}}^{(m)}(D))$. A high correlation would show that the explanations are consistent between the direct and two-step approaches. Furthermore, there are 2^8 possible choices of the mapping D for the 8 application types in our traffic flow dataset to be mapped to two CoS labels. Let D^* be the mapping such that

$$D^* = \arg\max_D \frac{1}{M} \sum_{m=1}^{M} \rho(\Phi_{\text{direct}}^{(m)}, \Phi_{\text{two-step}}^{(m)}(D)). \tag{5}$$

If D^* is the same as our input mapping of VOIP and CHAT to delay sensitive, then we understand that the best average correlation of superfeatures is achieved when the explanations correctly recognize the delay sensitivity of the application types.

Finally, some high-level insights can be drawn from the explanation set $\mathcal{E}$. Let σ be a function such that the k-th feature is grouped into the $\sigma(k)$-th superfeature. To determine how the k-th feature value varies along with the Shapley value of its parent superfeature $\sigma(k)$, we obtain the correlation between the k-th feature column of $\mathcal{E}$ and $\Phi_{\text{direct}}^{(\sigma(k))} = [\phi_{\sigma(k)}^{(1)}, ..., \phi_{\sigma(k)}^{(N)}]^T \in \mathbb{R}^N$. If it is positive, then we understand that as the value of the k-th feature increases, the probability of being delay sensitive increases; and vice versa. For each feature, the higher the absolute value of correlation, the more we are confident about our conclusion. In the next section, we will summarize some of these conclusions based on our explanation framework.

4 EXPERIMENTAL RESULTS

4.1 Dataset and Experimental Setup

We combine the aforementioned two ISCX datasets in our experiments. Our learning models are implemented in Python and trained on 80% of this dataset, plus 10% for validation and another 10% for testing. For data balancing of the training set, we apply the *sklearn.utils.resample* function from scikit-learn v0.21.3 [22], so that in the direct approach the training set contains the same number of delay sensitive and delay tolerant flows, while in the two-step approach the training set contains the same number of flows for each application type. We have experimented with logistic regression, SVM, and NN, but here we present results on NN only for brevity. The other learning models give similar conclusions but are less accurate in general.

We use RELU activation and batch normalization in each hidden layer in addition to a drop-out rate of 0.5. After experimenting with various configurations of NNs, we observe that using two hidden layers of 30 nodes each gives sufficient accuracy. To train the NNs, we use the ADAM optimizer along with the cross-entropy loss function. In the direct approach we use learning rate 5×10^{-5}, batch size 1000, and 400 epochs, while in the two-step approach we use learning rate 10^{-4}, batch size 500, and 600 epochs. Both settings have been observed to give superior performance for their respective approaches.

4.2 Classification Performance

In Table 3, we compare the classification performance of the two approaches. We observe that the test accuracy of 92.5% by the direct approach is superior to that of the two-step approach. More importantly, to a network service provider, misclassifying delay sensitive traffic as delay tolerant can impose a high penalty. Therefore, we are more interested in the false-negative rate on delay sensitive samples. We find it to be only 6% for the direct approach, more than three times lower than that of the two-step approach. This demonstrates the significant benefits of the direct approach. Furthermore, even though the two-step approach appears to give a lower false-negative rate on delay tolerant flows, its false-negative rate for classifying individual applications is

Metric	Direct Approach	Two-step Approach
Test accuracy	92.5%	87.5%
False-negative rate for delay sensitive flows	6%	21%
False-negative rate for delay tolerant flows	9%	4%

Table 3: Comparison of classification performance

CoS Label	Application Type	False-negative Rate
Delay Sensitive	CHAT	0.27
	VOIP	0.36
Delay Tolerant	AUDIO	0.35
	FTP	0.27
	MAIL	0.14
	P2P	0.04
	VIDEO	0.22
	WEB	0.13

Table 4: False-negative rates for different application types for two-step approach

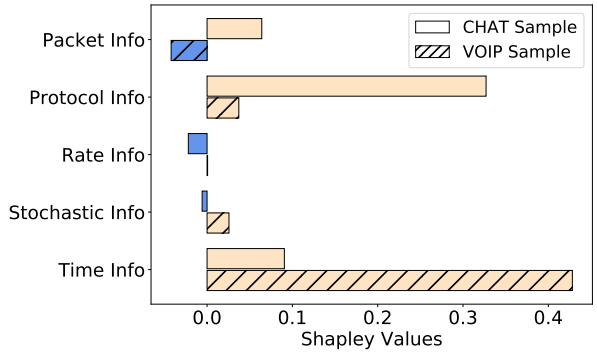

Figure 2: Explanations for the CoS classification of two delay sensitive flows

much higher, as shown in Table 4. Indeed, we observe that it often mis-classifies one application as another application of the same delay class, which does not appear as erroneous in delay classification but nevertheless calls into question its reliability.

4.3 Evaluating Explanations

As illustrative examples, in Fig. 2, we show the superfeature explanations $\{\phi_m\}$ for a CHAT flow and a VOIP flow, both of which are delay sensitive as explained in Section 2. One can observe that *protocol information* provides the greatest evidence for the CHAT flow being classified as delay sensitive, while *time information* provides the greatest evidence for the VOIP flow being classified as delay sensitive. For comparison, the explanations for two delay tolerant samples, a FTP flow and a MAIL flow, are shown in Fig. 3. Here we see that *packet information* provides the greatest evidence for both the FTP flow and the MAIL flow being classified as delay tolerant.

Next, we obtain the *explained values* of all samples in the test set by assigning Shapley values according to (4), using 2000 randomly

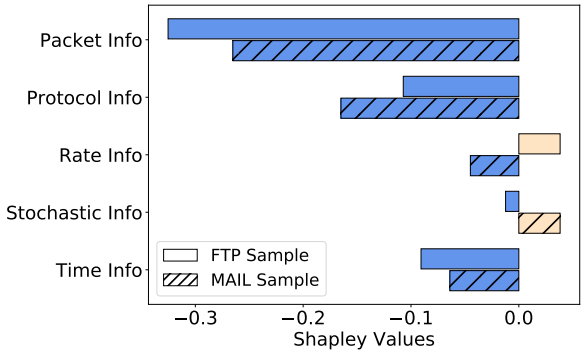

Figure 3: Explanations for the CoS classification of two delay tolerant flows

Test Samples	Our Work	Deep SHAP
All samples	0.83	0.66
Delay sensitive samples	0.67	0.48
Delay tolerant samples	0.75	0.56

Table 5: Correlation between explained value and predicted probability for test samples

Metric	Our Work	Deep SHAP
Accuracy	88%	83%
False-negative rate for explaining delay sensitive samples	15%	21%
False-negative rate for explaining delay tolerant samples	9%	13%

Table 6: Accuracy of explained labels against true labels for test samples

selected, correctly classified samples to form our explanation set $\mathcal{E}$. Table 5 records the correlation coefficient between the explained values and the predicted probabilities by the learning model. We compare the proposed explanation approach with Deep SHAP [16], where we approximate the Shapley values all 266 features and treat each as a superfeature to obtain the explained values. We observe that the proposed method has substantially higher correlation for the same learning model.[1] Next, we obtain the *explained labels* and present in Table 6 the overall accuracy and error rates of the explained labels when compared with the true labels. We observe higher overall accuracy by the proposed approach, indicating superior consistency of explanations over different feature vectors.

We then calculate the correlation between the explanations obtained from the direct and two-step approaches. Table 7 shows the correlation coefficients of each superfeature for the true mapping $D : \mathcal{A} \rightarrow C$ where VOIP and CHAT are mapped to delay sensitive. We observe that the correlation is high. In fact, the true mapping gives an average correlation of 0.82, which is the highest among

[1] As a rule of thumb to interpret correlation values, the ranges $(0.5, 0.7)$, $(0.7, 0.9)$, and $(0.9, 1)$ are commonly recognized as indicating moderate, high, and very high correlation, respectively [12].

Packet Info	Protocol Info	Rate Info	Stochastic Info	Time Info
0.81	0.89	0.78	0.72	0.92

Table 7: Correlation of superfeatures for the mapping of VOIP and CHAT as delay sensitive traffic

Top-ranked Features	Correlation Found
(1) FFT of data rate (arctan of frequency corresponding to the largest magnitude)	-0.72
(2) Time-to-live (dest. to src.)	0.71
(3) Mean pkt length	-0.62
(4) Mean payload length	-0.62
(5) # Bursts (dest. to src.)	0.62

Table 8: Top five features that impacts the probability of being delay sensitive

all 2^8 possible mappings as found by (5). This suggests that explanations between the two approaches are consistent, and the explanations reliably recognize the applications as the source of delay sensitivity.

Finally, Table 8 summarizes the top five features ranked by the absolute values of their correlation with their parent superfeatures, as described in Section 3.2. A large absolute value indicates that the feature strongly impacts the probability of a flow being delay sensitive, either positively or negatively. This shows that despite the dramatic reduction of complexity in computing Shapley values by using superfeatures, we can still obtain new insights about the contribution of individual features.

5 CONCLUSION

Through experimentation with real-world traffic flows, we have shown that the direct approach for CoS-oriented traffic classification can be substantially more accurate than the two-step approach that first infers the application types. We further present an efficient explanation framework based on Shapley values to interpret the classification results, using superfeatures defined based on domain knowledge. Our experimental results further demonstrate the consistency and usefulness of the proposed explanations.

6 ACKNOWLEDGMENT

This work is supported in part by grants from TELUS and from the Natural Sciences and Engineering Research Council (NSERC) of Canada.

REFERENCES

[1] 2016. ISCX TOR-nonTOR. Available online: https://www.unb.ca/cic/datasets/tor.html.
[2] 2016. ISCX VPN-nonVPN. Available online: https://www.unb.ca/cic/datasets/vpn.html.
[3] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé. 2018. Mobile Encrypted Traffic Classification Using Deep Learning. In *the Network Traffic Measurement and Analysis Conference (TMA)*. 1–8.
[4] T. Auld, A. Moore, and S. F. Gull. 2007. Bayesian Neural Networks for Internet Traffic Classification. *IEEE Transactions on Neural Networks* 18, 1 (2007), 223–239.
[5] S. Chabaa, A. Zeroual, and J. Antari. 2010. Identification and Prediction of Internet Traffic Using Artificial Neural Networks. *J. Intell. Learn. Syst. Appl.* 2, 3 (July 2010), 147–155.
[6] F. Doshi-Velez and B. Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608* (2017).
[7] G. Draper-Gil, A. Lashkari, M. Mamun, and A. Ghorbani. 2016. Characterization of Encrypted and VPN Traffic using Time-related Features. In *the 2nd Int. Conf. on Inf. Sys. Security and Privacy*. 407–414.
[8] J. Erman, A. Mahanti, M. Arlitt, and C. Williamson. 2007. Identifying and Discriminating Between Web and Peer-to-peer Traffic in the Network Core. In *the 16th Int. Conf. on World Wide Web*. 883–892.
[9] A. Este, F. Gringoli, and L. Salgarelli. 2009. Support Vector Machines for TCP traffic classification. *Computer Networks* 53, 14 (2009), 2476–2490.
[10] M. Finsterbusch, C. Richter, E. Rocha, J. Muller, and K. Hanssgen. 2014. A Survey of Payload-Based Traffic Classification Approaches. *IEEE Comm. Surveys Tutorials* 16, 2 (Second Quarter 2014), 1135–1156.
[11] L. Grimaudo, M. Mellia, E. Baralis, and R. Keralapura. 2014. SeLeCT: Self-Learning Classifier for Internet Traffic. *IEEE Transactions on Network and Service Management* 11, 2 (June 2014), 144–157.
[12] D. Hinkle, W. Wiersma, and S. Jurs. 2003. *Applied statistics for the behavioral sciences* (5th ed.).
[13] A. Lashkari, G. Draper-Gil, M. Mamun, and A. Ghorbani. 2017. Characterization of Tor Traffic using Time based Features. In *the 3rd Int. Conf. on Inf. Sys. Security and Privacy*.
[14] C. Liu, L. He, G. Xiong, Z. Cao, and Z. Li. 2019. FS-Net: A Flow Sequence Network For Encrypted Traffic Classification. In *the IEEE INFOCOM Conference on Computer Communications*. 1171–1179.
[15] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret. 2017. Network Traffic Classifier With Convolutional and Recurrent Neural Networks for Internet of Things. *IEEE Access* 5 (2017), 18042–18050.
[16] S. Lundberg and S. Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *the 30th Advances in Neural Information Processing Systems (NIPS)*. 4765–4774.
[17] C. Molnar. 2019. *Interpretable Machine Learning*. https://christophm.github.io/interpretable-ml-book/.
[18] A. Moore and D.Zuev. 2005. Internet Traffic Classification Using Bayesian Analysis Techniques. In *the ACM SIGMETRICS Int. Conf. on Measurement and Modeling of Computer Systems*. 50–60.
[19] A. Moore, D. Zuev, and M. Crogan. 2005. *Discriminators for use in flow-based classification*. Technical Report.
[20] T. T. T. Nguyen and G. Armitage. 2008. A survey of techniques for internet traffic classification using machine learning. *IEEE Comm. Surveys Tutorials* 10, 4 (Fourth Quarter 2008), 56–76.
[21] T. T. T. Nguyen, G. Armitage, P. Branch, and S. Zander. 2012. Timely and Continuous Machine-Learning-Based Classification for Interactive IP Traffic. *IEEE/ACM Trans. on Networking* 20, 6 (Dec 2012), 1880–1894.
[22] F. Pedregosa et. al. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.
[23] M. Ribeiro, S. Singh, and C. Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *the 22nd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*. 1135–1144.
[24] M. Roughan, S. Sen, O. Spatscheck, and N. Duffield. 2004. Class-of-service Mapping for QoS: A Statistical Signature-based Approach to IP Traffic Classification. In *the 4th ACM SIGCOMM Conf. on Internet Measurement*. 135–148.
[25] L. Shapley. 1953. A value for n-person games. *Contributions to the Theory of Games (AM-28)* 2 (1953), 307–318.
[26] E. Štrumbelj and I. Kononenko. 2014. Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems* 41, 3 (Dec 2014), 647–665.
[27] P. Wang, S. Lin, and M. Luo. 2016. A Framework for QoS-aware Traffic Classification Using Semi-supervised Machine Learning in SDNs. In *the IEEE International Conference on Services Computing (SCC)*. 760–765.
[28] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang. 2017. End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In *the IEEE Int. Conf. on Intelligence and Security Informatics (ISI)*. 43–48.
[29] Y. Wang, Y. Xiang, J. Zhang, W. Zhou, G. Wei, and L. T. Yang. 2014. Internet Traffic Classification Using Constrained Clustering. *IEEE Trans. on Parallel and Distributed Sys.* 25, 11 (Nov 2014), 2932–2943.
[30] S. Zander, T. Nguyen, and G. Armitage. 2005. Automated traffic classification and application identification using machine learning. In *the 30th IEEE Conference on Local Computer Networks (LCN)*. 250–257.
[31] J. Zhang, C. Chen, Y. Xiang, W. Zhou, and Y. Xiang. 2013. Internet Traffic Classification by Aggregating Correlated Naive Bayes Predictions. *IEEE Trans. on Inf. Forensics and Security* 8, 1 (Jan 2013), 5–15.
[32] J. Zhang, X. Chen, Y. Xiang, W. Zhou, and J. Wu. 2015. Robust Network Traffic Classification. *IEEE/ACM Transactions on Networking* 23, 4 (Aug 2015), 1257–1270.