Flow-Packet Hybrid Traffic Classification for Class-Aware Network Routing

Sayantan Chowdhury*, Ben Liang*, Ali Tizghadam†, Ilijc Albanese†
*University of Toronto, Canada †TELUS, Canada

***Abstract*—Network traffic classification using machine learning techniques has been widely studied. Most existing schemes classify entire traffic flows, but there are major limitations to their practicality. At a network router, the packets need to be processed with minimum delay, so the classifier cannot wait until the end of the flow to make a decision. Furthermore, a complicated machine learning algorithm can be too computationally expensive to implement inside the router. In this paper, we introduce flow-packet hybrid traffic classification (FPHTC), where the router makes a decision per packet based on a routing policy that is designed through transferring the learned knowledge from a flow-based classifier residing outside the router. We analyze the generalization bound of FPHTC and show its advantage over regular packet-based traffic classification. We present experimental results using a real-world traffic dataset to illustrate the classification performance of FPHTC. We show that it is robust toward traffic pattern changes and can be deployed with limited computational resource.**

I. Introduction

Traffic classification is critical to the operation of computer networks in many aspects, such as network management, quality-of-service guarantee, and security concerns. As a large segment of network traffic is encrypted in recent years, traditional methods of classification, e.g., the protocol-based approach and content comparison, are no longer appropriate [1]. In contrast, machine learning algorithms can identify traffic flows with high accuracy using statistical features [2]–[6]. Here, a *flow* is defined as a group of packets sharing the same 5-tuple: source and destination IP addresses, source and destination ports, and protocol used.

For class-aware routing in a network, the routers need to conduct traffic classification before forwarding the traffic. However, common flow-based machine learning algorithms are often too computationally expensive for the routers. For example, even though deep neural networks is known to provide accurate classification outcomes, they are computationally intensive to train, while they need to be updated frequently to adapt to the changing traffic pattern over time. More importantly, the statistical features required for flow-based traffic classification techniques often are not available for real-time classification. In general, statistical features such as the *variance of packet length* are extracted at the end of each flow, after the lengths of all packets are observed. Therefore, they are not useful for a router that must route the packets of a flow as they arrive, with as little delay as possible. Some authors have proposed early recognition of traffic classes by observing a subset of packets from each flow [7]–[9]. However, this still might cause significant delay as a router generally needs to process millions of packets within a fraction of a second.

A naive alternative is pure packet-based traffic classification, where each packet is observed and classified immediately, i.e., the classifier does not wait for a stream of packets from a flow. The router can look at the simple features embedded in the packet header and make a quick decision per packet. Packet-based traffic classification requires only fast lookup of the packet headers and thus is amenable to practical implementation in high-speed routers. However, a key drawback of this approach is that the classification performance can be poor due to the absence of detailed statistical features that are available in flow-based classification.

This motivates us to combine the advantages of both flow-based and packet-based traffic classification. We propose a novel Flow-Packet Hybrid Traffic Classification (FPHTC) method, where a low-complexity routing policy with packet classification at the router is designed with the assistance of a flow-based classifier that resides outside the router. To the best of our knowledge, there exists no prior work that considers this hybrid form between flow-based and packet-based methods for network traffic classification in the router.

Our contributions can be summarized as follows:

- We propose FPHTC, which generates a low-complexity routing policy to be applied to the incoming packets at a router. The routing policy enables class-aware routing using only simple features, e.g., those that can be directly read from the packet header. We generate the routing policy by exploiting the knowledge learned by a highly accurate flow-based classifier residing outside the router. The routing policy is constructed as a decision tree trained using the packets labeled by the flow-based classifier. In FPHTC, we can employ the flow-based classifier to label any number of packets, and thus, the resulting routing policy can be highly accurate.
- We show that FPHTC can be deployed in an online learning setting, where a new routing policy is updated at the router whenever the performance of the current routing policy falls below a certain threshold due to changes in the traffic pattern. This is achieved by adaptively retraining the flow-based classifier and then the routing policy, upon receiving the feedback that routing policy update is required.
- We provide theoretical justification for the performance advantage of FPHTC over regular packet-based traffic classification, in terms of the generalization bound. This

This work was funded in part by Telus Communications and the Natural Sciences and Engineering Research Council (NSERC) of Canada.

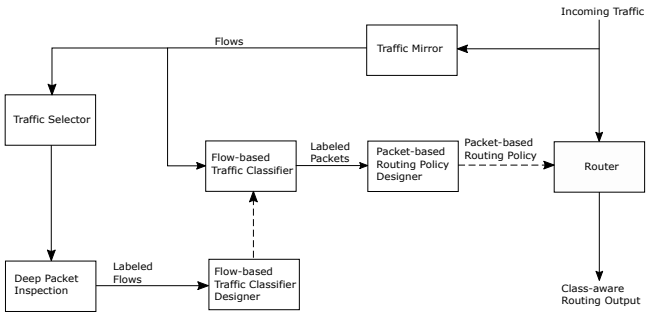

Figure 1: Diagram illustrating the FPHTC framework.

further enables exploring the trade-off between the cost of labeling data for training the flow-based classifier and the generalization bound of the routing policy.

- We conduct extensive experiments using an aggregate dataset of 43590 encrypted traffic flows from [10] and [11]. We train gradient boosted tree models XGBoost [12] and LightGBM [13] as the flow-based classifier and compare the performance of FPHTC with regular packet-based traffic classification for different training dataset sizes. We observe substantial performance gain under FPHTC.

The rest of this paper is structured as follows. The concept of FPHTC is presented in Section II, where we describe different components of FPHTC, routing policy design, and update procedure in detail. Section III provides an analytical comparison between FPHTC and regular packet-based traffic classification in terms of the generalization bound. In Section IV, we present our experimental setup and classification performance of FPHTC. Section V concludes the paper.

II. Flow-Packet Hybrid Traffic Classification

We propose FPHTC for a router that needs to conduct class-aware traffic processing. In this section, we provide a detailed description of our scheme. A diagram illustrating the overall framework of FPHTC is given in Fig. 1.

A. Core Components of FPHTC

1) Router: The router accepts an incoming stream of packets and processes them according to their service classes using the routing policy. The basic structure and function of such a routing policy are well-defined in prior works on packet classification [14], [15]. Throughout our work, we focus on how to generate routing policy rules by training a machine learning model for packet-based traffic classification, where the chosen header fields of each packet are its features, i.e., the inputs into the learning model, and the packet is classified by the learning model to determine its CoS. For example, the chosen header fields may be the source IP address, destination IP address, source port number, and destination port number, among others, and the possible actions may be to route a packet as delay sensitive, delay moderate, or delay tolerant.

2) Flow-based Traffic Classifier: The flow-based traffic classifier resides outside the router, in some powerful equipment that can handle the heavy computation required by sophisticated machine learning techniques. It is a complex and highly accurate machine learning model that can classify a traffic flow in terms of CoS for all of its packets. It is trained using a number of bidirectional TCP flows with a set of flow-level statistical features extracted from the raw dataset.

Various methods are possible to generate the training dataset for the flow-based traffic classifier. In this work, since we are ultimately interested in online classification to handle changing traffic pattern over time, we propose to use a continuously updated recording of the past traffic. Specifically, we use a traffic mirror and a traffic selector, as shown in Fig. 1, to separate a selected small portion of the incoming traffic flows. The selected flows are then labeled using a Deep Packet Inspection (DPI) module according to their CoS. The true CoS labels obtained by DPI are used to train the flow-based classifier. We note that DPI cannot be used to replace the role of the flow-based classifier for all flows, due to its prohibitive cost and delay for common encrypted traffic.

The role of the flow-based traffic classifier designer includes data preprocessing, hyperparameter selection, and finally, training the flow-based classifier. Once the flow-based classifier is trained, we use it to infer the CoS labels of all incoming flows captured by the traffic mirror. Then all packets belonging to a flow can be tagged by CoS label of the flow. We note that the CoS labels generated in this way, by a flow-based classifier, are too late to be used in the *routing* of the labeled packets. However, what this achieves is to create a packet-level dataset for *training* the packet-based routing policy as explained below.

3) Packet-based Routing Policy Designer: The packet-based routing policy designer takes labeled packets from the flow-based classifier as input, and it outputs a routing policy for the router. Specifically, the routing policy designer trains a packet-based classifier using the labeled packets as the training dataset.

In this work, we use the binary decision tree learning model for the packet-based classifier. In the decision tree, each path from the root to a node is a routing policy rule. Thus, to obtain routing policy rules that can be used in the router, the routing policy designer only needs to train a decision tree on the packet-level dataset. Furthermore, we note that the number of routing policy rules equals the number of leaf nodes in the decision tree. This provides an easy way to control the size of the routing policy, i.e., the routing policy designer can limit

the maximum number of leaf nodes while training the decision tree.

B. Construction of Routing Policy

The construction of the routing policy in FPHTC involves transferring learned knowledge from the flow-based classifier to the routing policy designer. In the machine learning literature, knowledge distillation [16], [17] is a technique where a simple student model is trained on the predictions supplied by a highly accurate and complex teacher model. In FPHTC, we train a decision tree at the routing policy designer using the predictions from the flow-based classifier as training targets. In essence, the routing policy designer tries to approximate the performance of the flow-based classifier.

The flow-based classifier is trained with flow-level statistical features whereas the routing policy designer uses only some features that can be read directly from the packet header. Therefore, it is clear that the learned routing policy will perform worse than the flow-based classifier given the same traffic data for training. However, since there are unlabeled training data available, i.e., those that have not been labeled by DPI, we can label those data samples using our flow-based classifier to substantially enlarge the training dataset for the routing policy designer. Since the decision tree at the routing policy designer is trained on a much larger dataset than that of the flow-based classifier, the performance of the routing policy can be close to that of the flow-based classifier. More importantly, since the routing policy created by FPHTC utilizes information learned from a more powerful flow-based classifier, it can substantially outperform a regular packet-based classifier trained using only the small amount of labels generated by DPI.

C. Routing Policy Update Procedure in Online Setting

In a practical system, the data pattern of the incoming traffic changes over time, e.g., due to new applications appearing in the network, or changing user behavior. Therefore, we design FPHTC to dynamically update the routing policy over time.

In Fig. 2, we illustrate how the modules sequentially function over a continuous stream of traffic. At any given time slot, we collect and label a small portion of the incoming traffic flows using DPI to train the flow-based classifier. Meanwhile, we continue to collect flows to be used in the training of the routing policy. Once the flow-based classifier is trained, we use it to label those collected flows not labeled by DPI. Then, the routing policy designer trains a decision tree to generate the routing policy, which is then updated to the router.

One important question is whether we should repeat these steps and update the routing policy at each time slot. If the traffic data pattern does not change too frequently, routing policy update at every time slot would be a waste of resources. To re-train the flow-based classifier, the labeling cost using DPI would also be expensive. A cost-effective solution is to update the routing policy only when the traffic pattern has altered significantly. This can be inferred by measuring the performance deterioration at the router. A feedback signal can be generated, for example, based on the increase in packet drop or congestion, to indicate that a routing policy update is necessary. We demonstrate the adaptiveness of FPHTC in the online setting in Section IV.

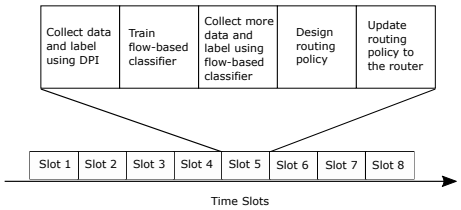

Figure 2: FPHTC in online setting.

III. Comparison With Regular Packet-based Traffic Classification

To highlight the benefit of combining flow-based and packet-based traffic classification in FPHTC, we compare it against regular packet-based traffic classification without the help from a flow-based classifier. In this section, we provide a theoretical justification of why FPHTC performs better than regular packet-based traffic classification.

In regular packet-based traffic classification, a decision tree at the routing policy designer is trained with the true labels of the packets. In contrast, in FPHTC only the flow-based classifier requires true labels. Hence, for fair comparison, we maintain that the training dataset size of the regular packet-based traffic classifier is equal to the training dataset size of the flow-based classifier in FPHTC, when both are measured in terms of the number of flows.

Let n be the number of flows in the training dataset for the routing policy designer in FPHTC. Recall that these flows are labeled by the flow-based classifier. The flow-based classifier is trained with a λ fraction of these flows, which have been labeled by DPI. If the cost of labeling each flow using DPI is c_{DPI}, then the total cost is $n\lambda c_{\text{DPI}}$. Increasing λ will lead to a more accurate flow-based classifier in FPHTC and ultimately a more accurate routing policy. However, this will also result in a greater cost of labeling flows. Thus, there exists a trade-off that should be carefully analyzed.

Suppose $\mathcal{H}$ is the hypothesis set of a classifier with some capacity measure $|\mathcal{H}|_C$. If $\hat{f} \in \mathcal{H}$ is the function learned by the classifier from n training samples, and f is the ground truth, i.e., the target function of interest, then the generalization bound can be expressed as follows [17]:

$$R(\hat{f}) - R(f) \leq O\left(\frac{|\mathcal{H}|_C}{n^r}\right) + \epsilon, \tag{1}$$

where $R(\cdot)$ is the expected loss, the $O(\cdot)$ term is the estimation error, and ϵ is the approximation error. The rate of learning is given by $O(n^{-r})$. For difficult or *non-separable* problems, $r = \frac{1}{2}$ and this represents a slow rate of learning. In contrast, for easy or *separable* problems, where the trained classifier makes no training error, $r = 1$ and this represents a fast rate of learning [17].

In FPHTC, the flow-based classifier and the routing policy designer play the role of the teacher and the student

respectively. Let $\mathcal{H}_{\text{fl}}$ be the hypothesis set for the flow-based classifier, with capacity measure $|\mathcal{H}_{\text{fl}}|_C$. Let $f_{\text{fl}} \in \mathcal{H}_{\text{fl}}$ be the function learned by the flow-based classifier and f be the ground truth. Since only a fraction λ of the flows are used for training, the generalization bound of the flow-based classifier is given by

$$R(f_{\text{fl}}) - R(f) \leq O\left(\frac{|\mathcal{H}_{\text{fl}}|_C}{n\lambda}\right) + \epsilon_{\text{fl}}, \tag{2}$$

where ϵ_{fl} is the approximation error of the flow-based classifier. Here we use a common assumption that the rate of learning for the more sophisticated teacher is inversely proportional to the size of the training dataset, i.e., $O((n\lambda)^{-1})$.

Similarly, let $\mathcal{H}_{\text{rp}}$ be the hypothesis set of the routing policy designer in FPHTC with capacity measure $|\mathcal{H}_{\text{rp}}|_C$, and let $f_{\text{rp}} \in \mathcal{H}_{\text{rp}}$ be the function determined by the routing policy designer. We have

$$R(f_{\text{rp}}) - R(f_{\text{fl}}) \leq O\left(\frac{|\mathcal{H}_{\text{rp}}|_C}{n^\alpha}\right) + \epsilon_{\text{rp}}, \tag{3}$$

where ϵ_{rp} is the approximation error of the routing policy. As the student learns using the teacher's predictions, the decision boundary of the original classification problem has been translated to a smoother one. Thus, the student, aided by the teacher's predictions, learns at a faster rate than with the true labels, so that its rate of learning is represented by the parameter $0.5 \leq \alpha \leq 1$.

Combining (2) and (3), we get

$$\begin{aligned} R(f_{\text{rp}}) - R(f) &= R(f_{\text{rp}}) - R(f_{\text{fl}}) + R(f_{\text{fl}}) - R(f) \\ &\leq O\left(\frac{|\mathcal{H}_{\text{rp}}|_C}{n^\alpha}\right) + \epsilon_{\text{rp}} + O\left(\frac{|\mathcal{H}_{\text{fl}}|_C}{n\lambda}\right) + \epsilon_{\text{fl}} \\ &\leq O\left(\frac{\lambda^\alpha|\mathcal{H}_{\text{rp}}|_C + |\mathcal{H}_{\text{fl}}|_C}{n^\alpha\lambda^\alpha}\right) + \epsilon_{\text{rp}} + \epsilon_{\text{fl}}. \end{aligned} \tag{4}$$

This gives the generalization bound for FPHTC. We note that this bound improves as λ increases.

As a further step for system optimization, we can consider a weighted sum of the generalization bound and the DPI labeling cost as a function of λ:

$$C(\lambda) = K \cdot \frac{\lambda^\alpha|\mathcal{H}_{\text{rp}}|_C + |\mathcal{H}_{\text{fl}}|_C}{n^\alpha\lambda^\alpha} + \epsilon_{\text{rp}} + \epsilon_{\text{fl}} + n\lambda c_{\text{DPI}}, \tag{5}$$

where K is a constant of proportionality. To minimize $C(\lambda)$, we differentiate (5) wrt λ and set the derivative equal to zero. Finally, we obtain the optimal λ for FPHTC as

$$\lambda^* = \left(\frac{\alpha K|\mathcal{H}_{\text{fl}}|_C}{n^{1+\alpha}c_{\text{DPI}}}\right)^{\frac{1}{1+\alpha}}. \tag{6}$$

In regular packet-based traffic classification, we have the same hypothesis set $\mathcal{H}_{\text{rp}}$, since it represents the capability of the same router. The function $f_{\text{pk}} \in \mathcal{H}_{\text{rp}}$ is chosen to approximate the ground truth f without the help of the teacher. Again, we can bound the regular packet-based traffic classifier as follows:

$$R(f_{\text{pk}}) - R(f) \leq O\left(\frac{|\mathcal{H}_{\text{rp}}|_C}{\sqrt{n\lambda}}\right) + \epsilon_{\text{pk}}, \tag{7}$$

where ϵ_{pk} is the approximation error of the regular packet-based traffic classifier. As the student is trained using true labels in this case, the classification problem is difficult and the rate of learning is slower at $O((n\lambda)^{-1/2})$. Comparing (7) with (4), we see that FPHTC outperforms regular packet-based traffic classification if the following inequality holds:

$$O\left(\frac{\lambda^\alpha|\mathcal{H}_{\text{rp}}|_C + |\mathcal{H}_{\text{fl}}|_C}{n^\alpha\lambda^\alpha}\right) + \epsilon_{\text{rp}} + \epsilon_{\text{fl}} \leq O\left(\frac{|\mathcal{H}_{\text{rp}}|_C}{\sqrt{n\lambda}}\right) + \epsilon_{\text{pk}}. \tag{8}$$

Let us now explain why it is reasonable for (8) to hold in our traffic classification problem. As the teacher is a highly complex flow-based classifier and the student is a simple decision tree trained at the routing policy designer, we have $|\mathcal{H}_{\text{fl}}|_C >> |\mathcal{H}_{\text{rp}}|_C$. Hence, FPHTC may be viewed as an instance of Hinton's knowledge distillation framework [16], except that in our case the feature spaces of the teacher and the student are different. Furthermore, the flow-based classifier is trained with many flow-level statistical features, whereas the routing policy is designed based on a much smaller number of packet-level features. Therefore, the approximation error of the routing policy is much larger than that of the flow-based classifier. Thus, in (8), $\epsilon_{\text{fl}} + \epsilon_{\text{rp}} << \epsilon_{\text{pk}}$. Furthermore, as $\alpha \geq 0.5$, we have $n^\alpha\lambda^\alpha \geq \sqrt{n\lambda}$ when $\lambda \leq 1$. Therefore, even though $|\mathcal{H}_{\text{fl}}|_C >> |\mathcal{H}_{\text{rp}}|_C$, the large value of $|\mathcal{H}_{\text{fl}}|_C$ can be balanced by the parameters α and λ.

Note that α is an intrinsic parameter that we cannot control, but we can control λ. On the one hand, if λ is close to 1, there is no benefit from using λ. On the other hand, if λ is too small, both $n^\alpha\lambda^\alpha$ and $\sqrt{n\lambda}$ approaches zero and the difference between α and 0.5 is lost. However, a moderate λ can satisfy (8) and allow FPHTC to outperform regular packet-based traffic classification.

IV. Experimental Evaluation

In this section, we first discuss our experimental setup. Then, we evaluate the performance of FPHTC and compare it against regular packet-based traffic classification to demonstrate the impacts of the training dataset size, and the online setting.

A. Dataset and Learning Models

We use the combined real-world traffic traces of ISCX VPN-nonVPN (2016) [10], [18] and ISCX Tor-nonTor (2016) [11], [19]. It contains pcap files for 43590 encrypted TCP bidirectional flows from 8 application types. We group these 8 application types into 3 CoS categories as shown in Table 1. For flow-based classification, we extract 268 features from each TCP flow. Our current set of features includes source and destination IP addresses in addition to the list of 266 features used in [6].

We remove the flows with no payload from our dataset and use 90% of the rest of the dataset as the full training dataset, plus 10% for testing. In various experiments that require different training dataset sizes, we use a randomly selected subset of the full training dataset. All training datasets are balanced by applying the *sklearn.utils.resample* function

CoS label	Application type
Delay Sensitive	CHAT, VOIP
Delay Moderate	AUDIO, VIDEO
Delay Tolerant	FTP, MAIL, P2P, WEB

Table 1: Application types and CoS labels.

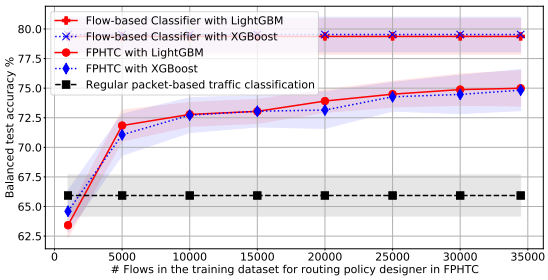

Figure 3: Balanced test accuracy vs. n, size of training dataset for the routing policy designer in FPHTC, with 90% confidence interval.

from scikit-learn v0.21.3 [20] so that the number of training samples in each CoS class is the same. All machine learning models are implemented in Python 3.8.3. We use balanced test accuracy as the main performance metric, which is the average of the proportion of correctly classified samples in each class. This performance metric is not affected by the imbalance in the dataset.

The routing policy designer takes labeled packets as input with 4 features: source IP address, destination IP address, source port number, and destination port number. The 32-bit IPv4 addresses are converted to decimal numbers to be used as feature values. Given a set of labeled flows, we obtain the corresponding set of unique packets having these 4 features. Note that these 4 features guarantee that all packets from a particular TCP flow are mapped to the same routing output.

We note that the traffic dataset is structured. For this type of data, it is known that the gradient boosted tree ensemble is appropriate as a learning model [21]. We use the state-of-the-art gradient boosted tree ensembles XGBoost [12] and LightGBM [13] as the flow-based classifier. In our experiments, the XGBoost model is trained with 100 trees in the ensemble. The learning rate is 0.3 and the maximum depth of each tree is limited to 6. For LightGBM, we use the gradient-based one-side sampling (GOSS) boosting method. The number of trees in the ensemble is 100 with the number of leaves and maximum depth restricted to 31 and unlimited, respectively. The learning rate remains 0.3.

The routing policy designer trains a single Classification and Regression Tree (CART) with the predictions from the flow-based classifier as training targets. We use a decision tree classifier with balanced class weights and entropy as the criterion for choosing the best split. The parameters such as maximum depth and maximum leaf nodes, which determine the structure of the tree, are kept unlimited.

B. Impact of Size of Training Datasets

The performance of FPHTC is shown in Fig. 3 with 1000 flows in the training dataset of the flow-based classifier.

# Flows in the training dataset	Flow-based classifier	FPHTC	Regular packet-based
1000	79.59%	74.99%	65.93%
5000	89.59%	84.85%	78.97%
10000	92.27%	87.13%	83.85%

Table 2: Balanced test accuracy vs. # flows in the training dataset for the flow-based classifier and the regular packet-based classifier.

We present the balanced test accuracy of FPHTC with 90% confidence interval. We observe that it increases as the training dataset size for the routing policy designer is increased. Recall that, since this training dataset is generated by the flow-based classifier, there is no theoretical limit to its size. In contrast, the training dataset size for the regular packet-based traffic classifier remains the same as that of the flow-based classifier. Thus, we observe that FPHTC outperforms regular packet-based traffic classification when the training dataset for the routing policy designer becomes large enough. The performance gain of FPHTC is larger when the flow-based classifier and the regular packet-based traffic classifier are trained using a small training dataset. For example, when the flow-based classifier with LightGBM and the routing policy designer are trained using 1000 flows and the maximum number of available flows, i.e., full training dataset containing 34473 flows, respectively, FPHTC is about 9% more accurate than regular packet-based traffic classification. Thus, FPHTC significantly improves accuracy in the low data regime.

We note that the performance of FPHTC is strictly ascending in Fig. 3. If we had more than 34473 unique flows in our available dataset for training, we could have further increased the training dataset size for the routing policy designer and observed a larger gain of FPHTC over regular packet-based traffic classification. Also, we see that using LightGBM as the flow-based classifier performs slightly better than using XGBoost. Therefore, we will present the results using LightGBM as the flow-based classifier for the rest of the experiments.

In Table 2, for various training dataset sizes, we present the balanced test accuracy of the flow-based classifier with LightGBM, regular packet-based traffic classifier, and FPHTC, where $n = 34473$. The experiment has been repeated over 10 randomly chosen training and test sets and then the average accuracy is listed. We note that there is a significant gap between the accuracy of the flow-based classifier and the regular packet-based traffic classifier. The gap is especially large when we have a small training dataset. This gap is due to the many more features observed by the flow-based classifier than the regular packet-based traffic classifier. This confirms our rationale for this work, that there is room for improvement for the routing policy if it can utilize the knowledge learned by the flow-based classifier. Furthermore, we observe that FPHTC can substantially reduce that gap, especially when the amount of training data is small.

C. Classification Performance of FPHTC in Online Setting

Finally, we implement the routing policy update procedure of FPHTC in an online setting. In this experiment, we simply

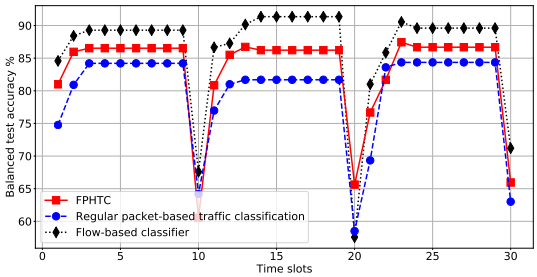

Figure 4: Performance of FPHTC in online setting with traffic pattern changing after every 10 time slots.

	Application types in the dataset
At time slot 0	AUDIO, FTP, VIDEO, VOIP, WEB
At time slot 10	FTP, MAIL, P2P, VIDEO, VOIP
At time slot 20	AUDIO, CHAT, FTP, MAIL, WEB

Table 3: Change of traffic pattern over time.

use the test accuracy of the routing policy as a feedback signal. If the accuracy is dropped below some accuracy threshold at the end of a time slot, re-training of the flow-based classifier and the routing policy update begin from the next time slot. Training stops when the accuracy crosses back above the accuracy threshold and the consecutive improvement in accuracy is less than some saturation threshold.

To simulate a never-ending stream of traffic, we keep shuffling our training dataset randomly. We run our experiment over 30 time slots, and we change the data pattern after every 10 time slots. To simulate traffic pattern change, we always use only a subset of 5 application types as the incoming traffic. After 10 time slots, we pick another subset of 5 application types and generate the incoming traffic. Thus, the test traffic pattern is changed at time slot 0, at time slot 10, and at time slot 20. The changing of applications over time is shown in Table 3.

At each time slot where re-training is needed, the first 1000 flows are selected and labeled by DPI to be used for training the flow-based classifier. Then, 10000 flows are labeled by the trained flow-based classifier and used for routing policy design. We use an accuracy threshold of 80% and a saturation threshold of 1% in our experiment. Fig. 4 illustrates how the balanced test accuracy drops after every 10 time slots due to the traffic pattern change. Then re-training begins, the routing policy is updated, and we observe gradual increase of the test accuracy. Once the test accuracy is saturated, the routing policy update stops, and we observe a flat line in the test accuracy. Again, we observe that FPHTC substantially outperforms regular packet-based traffic classification in the online setting.

V. Conclusion

In this paper, we propose Flow-Packet Hybrid Traffic Classification (FPHTC), which enables low-complexity but highly accurate packet-based classification at a network router. In FPHTC, a sophisticated flow-based classifier residing outside the router uses high-dimensional flow-level features to achieve high classification accuracy, while a routing policy designer generates a simple routing policy for the router based on a small number of packet-level features, utilizing the knowledge of the flow-based classifier. We discuss routing policy updates in an online setting, which keeps FPHTC robust towards traffic pattern change over time. Our experimental results confirms that the learned knowledge from the flow-based classifier can substantially improve the performance of the routing policy.

References

[1] M. Finsterbusch, C. Richter, E. Rocha, J. Muller, and K. Hanssgen, "A survey of payload-based traffic classification approaches," *IEEE Communications Surveys Tutorials*, vol. 16, no. 2, pp. 1135–1156, 2014.

[2] A. Moore and D. Zuev, "Internet traffic classification using Bayesian analysis techniques," in *Proc. ACM SIGMETRICS Int. Conf. on Measurement and Modeling of Computer Systems*, 2005.

[3] T. Auld, A. Moore, and S. F. Gull, "Bayesian neural networks for Internet traffic classification," *IEEE Transactions on Neural Networks*, vol. 18, no. 1, pp. 223–239, 2007.

[4] M. Roughan, S. Sen, O. Spatscheck, and N. Duffield, "Class-of-service mapping for QoS: A statistical signature-based approach to IP traffic classification," in *Proc. ACM SIGCOMM Conference on Internet Measurement*, 2004.

[5] P. Wang, S. Lin, and M. Luo, "A framework for QoS-aware traffic classification using semi-supervised machine learning in SDNs," in *Proc. IEEE International Conference on Services Computing (SCC)*, 2016.

[6] S. Chowdhury, B. Liang, and A. Tizghadam, "Explaining class-of-service oriented network traffic classification with superfeatures," in *Proc. ACM CoNEXT Workshop on Big DAta, Machine Learning and Artificial Intelligence for Data Communication Networks (Big-DAMA)*, 2019.

[7] L. Bernaille and R. Teixeira, "Early recognition of encrypted applications," in *Proc. Passive and Active Network Measurement*, 2007.

[8] T. T. T. Nguyen, G. Armitage, P. Branch, and S. Zander, "Timely and continuous machine-learning-based classification for interactive IP traffic," *IEEE/ACM Transactions on Networking*, vol. 20, no. 6, pp. 1880–1894, 2012.

[9] G. Xie, M. Iliofotou, R. Keralapura, M. Faloutsos, and A. Nucci, "SubFlow: Towards practical flow-level traffic classification," in *Proc. IEEE INFOCOM*, 2012.

[10] G. Draper-Gil, A. Lashkari, M. Mamun, and A. Ghorbani, "Characterization of encrypted and VPN traffic using time-related features," in *Proc. International Conference on Information Systems Security and Privacy (ICISSP)*, 2016.

[11] A. Lashkari, G. Draper-Gil, M. Mamun, and A. Ghorbani, "Characterization of Tor traffic using time based features," in *Proc. International Conference on Information System Security and Privacy (ICISSP)*, 2017.

[12] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[13] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. Neural Information Processing Systems (NIPS)*, 2017.

[14] P. Gupta and N. McKeown, "Packet classification on multiple fields," *SIGCOMM Comput. Commun. Rev.*, vol. 29, no. 4, p. 147–160, 1999.

[15] ——, "Algorithms for packet classification," *Netwrk. Mag. of Global Internetwkg.*, vol. 15, no. 2, p. 24–32, 2001.

[16] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *Proc. NIPS Deep Learning and Representation Learning Workshop*, 2015.

[17] D. Lopez-Paz, B. Schölkopf, L. Bottou, and V. Vapnik, "Unifying distillation and privileged information," in *Proc. International Conference on Learning Representations (ICLR)*, 2016.

[18] "ISCX VPN-nonVPN," 2016. [Online]. Available: https://www.unb.ca/cic/datasets/vpn.html

[19] "ISCX TOR-nonTOR," 2016. [Online]. Available: https://www.unb.ca/cic/datasets/tor.html

[20] F. Pedregosa et. al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[21] R. Bekkerman, "The present and the future of the KDD cup competition: an outsider's perspective," 2015. [Online]. Available: https://www.linkedin.com/pulse/present-future-kdd-cup-competition-outsiders-ron-bekkerman