2022-05-16

ML-Based Online Traffic Classification for SDNs

Mohammed Nsaif
University of Debrecen, mohammed.nsaif@mailbox.unideb.hu

Gergely Kovasznai
Karoly Catholic University, kovasznai.gergely@uni-eszterhazy.hu

Mohammed Abboosh
University of Debrecen

See next page for additional authors

Recommended Citation

M. Nsaif, G. Kovásznai, M. Abboosh, A. Malik and R. d. Fréin, "ML-Based Online Traffic Classification for SDNs," 2022 IEEE 2nd Conference on Information Technology and Data Science (CITDS), Debrecen, Hungary, 2022, pp. 217-222, doi: 10.1109/CITDS54976.2022.9914138.

Funder: SFI

ML-Based Online Traffic Classification for SDNs

M. Nsaif and G. Kovásznai and M. Abboosh and A. Malik and Ruairí de Fréin

Ollscoil Teicneolaíochta Baile Átha Cliath,
Campas na Cathrach,
Ireland.
web: https://robustandscalable.wordpress.com

in: IEEE 2nd Conference on Information Technology and Data Science (CITDS). See also BibTeX entry below.

BibTeX:

```
@article{defrein22MLBased,
  author={M. Nsaif and G. Kovasznai and M. Abboosh and A. Malik and
  Ruair\'{i} de Fr\'{e}in},
  journal={IEEE 2nd Conference on Information Technology and Data Science (CITDS)},
  title={ML-Based Online Traffic Classification for SDNs},
  year={2022},
  pages={217-222},
  doi={10.1109/CITDS54976.2022.9914138}
  }
```


document created on: January 16, 2023
created from file: rdefreinITDS22.tex
cover page automatically created with CoverPage.sty
(available at your favourite CTAN mirror)

ML-Based Online Traffic Classification for SDNs

Mohammed Nsaif[1], Gergely Kovásznai[2], Mohammed Abboosh[3], Ali Malik[4], Ruairí de Fréin[4]
[1]Department of Information Technology, University of Debrecen, Hungary, [2]Department of Computational Science, Eszterházy Károly Catholic University, Hungary, [3]Department of Data Science and Visualization, University of Debrecen, Hungary, [4]School of Electrical and Electronic Engineering, Technological University Dublin, Ireland (e:ruairi.defrein@tudublin.ie).

***Abstract*—Traffic classification is a crucial aspect for Software-Defined Networking functionalities. This paper is a part of an on-going project aiming at optimizing power consumption in the environment of software-defined datacenter networks. We have developed a novel routing strategy that can blindly balance between the power consumption and the quality of service for the incoming traffic flows. In this paper, we demonstrate how to classify the network traffic flows so that the quality of service of each flow-class can be guaranteed efficiently. This is achieved by creating a dataset that encompasses different types of network traffic such as video, VoIP, game and ICMP. The performance of a number of Machine Learning techniques is compared and the results are reported. As part of future work, we will incorporate classification into the power consumption model towards achieving further advances in this research area.**

***Index Terms*—Machine learning, classification, Software Defined Networks, OpenFlow, Traffic Classification**

I. Introduction

Precise traffic classification is one of the essential networking functions that allows network operators to manage the network resources in an efficient manner. Traffic classification is particularly beneficial for solving Quality of Service (QoS) related problems, which are associated with a wide range of computer networks issues such as power consumption in data centers. Nowadays, data center networks are considered to be the backbone of many Internet applications and services in different forms such as multimedia and gaming. According to [1], the energy consumption of data center networks accounts for 1.4% of global electrical energy consumption, with a compound annual growth rate (CAGR) of 4.4% for $2007 - 2012$. In fact, this is substantially higher than the forecasted 2.1% percent rise in world consumption from 2012 to 2040.

As a part of the on-going project on minimizing the power consumption of data center networks using Software-Defined Networking (SDN), we recently introduced the Fill Preferred Link First (FPLF) routing protocol[2]. In FPLF, we designed an integer programming model and heuristic algorithms for energy-aware routing which was implemented using SDN. Since the data and control planes are decoupled in SDN, a global view of the data plane forwarding elements is possible, which makes network management and monitoring easier. In essence, FPLF reduces the usage of data plane links by fitting the incoming traffic flows to the smallest number of links. Although FPLF is capable of reducing the power consumption in data center networks, it is expected that such gains would come at the cost of reduced QoS for delay sensitive flows like video, VoIP and online game. Therefore, traffic classification is crucial for FPLF so that the class of service information can be identified and become available to the routing and forwarding scheme. Traffic classification can be achieved by three techniques, these are: port-based, deep packet inspection and Machine Learning (ML) [3].

The traffic of modern services is often encrypted or not even associated with specific ports where that makes both port-based and deep packet inspection approaches not very effective [4]. ML-based techniques have been widely explored to surpass the limitation of the traditional classification methods. In this paper, we contribute a lightweight real-time traffic classification model based on ML techniques. We have implemented and evaluated the proposed method in an SDN environment and shown its effectiveness in accurately classifying a wide range of traffic applications in an online manner.

The paper is organized as follows. Section II introduces various flow classification techniques from the literature. Traffic generation and data collection are described in Section III. The features selection and classification models are presented in Section IV. The online implementation is presented in Section V. The discussion of the results, including the performance limitations, is presented in Section VI. Finally, we summarize the work, the outcomes, and suggest future research suggestions in Section VII.

II. Related Work

This section summarizes recent research with a focus on supervised ML methods, including Support Vector Machines (SVM), Random Forests (RF), Gradient Boosting (GB) and Neural Networks (NN). In[5], the authors developed a system for collecting and selecting a set of flow metrics in SDN for accurate traffic categorization. The Principal Component Analysis (PCA) and a Genetic Algorithm (GA) were employed to find the best features for traffic categorization, SVM was used as a classification algorithm. To obtain a dataset for validation, the authors install the experiment with three layers of switches and hosts. The study showed that PCA and GA have classification accuracy rates over 91% and 88%, respectively. However, the study focused on finding the optimal subset of flow features and tested one type of ML algorithm, and there is no real-time testing, instead, it can be used for analyzing DDoS, FTP and Video Streaming traffic. The authors in[6] introduced an SDN-based traffic categorization architecture.

The OpenFlow protocol is used to collect the statistics from the forwarding elements for the purpose of feature extraction where that includes the first incoming five packets, interval arrival time, packet size, source and destination MAC/Port/IP, and flow duration. PCA was used to determine the best components in order to eliminate feature redundancy. An ensemble learning approach is applied in[6], in which three essential classifiers cast a weighted vote on the classification outcome. Random Forests (RF), Extreme Gradient Boosting (EGB), and Stochastic Gradient Boosting (SGB) were adopted. The reported accuracy of the experimental results was 90%. The main limitation of this approach is that it relies on port number and payload inspection, This approach is not effective due to the dynamics of modern traffic. The authors in [7] suggested to use ML algorithms to recognize network flows. Four Neural Network estimators were employed, these are: Feedforward, Multilayer Perceptrons, Levenberg-Marquardt and Naive Bayes. This method used full-flow data to estimate features such as five-tuple information, the average number of packets, and the average bytes in a flow. Data was collected through instant chatting, video, FTP, HTTP, streaming and peer-to-peer protocols. These four categorization approaches achieved an accuracy of 95.6%, 97%, 97%, and 97.6%, respectively. This approach must capture data from the whole flow, and such limitations may make it impossible to use early traffic classification. The authors in [8] introduced DeepSDN for predicting traffic flows in short response time. DeepSDN has been tested by using Moore dataset and not in an online fashion. The approach in [9] demonstrated how features for video traffic can be constructed by explicitly modeling traffic time series. In this case, a QoS metric is modelled as a series of falling exponentials. This type of passive feature modelling approach could easily be extended to traffic classification instead of congestion detection, which was the target application in the paper. Finally, due to the dynamic and encrypted nature of today's communication, traditional approaches such as identifying traffic based on port number and payload inspection are no longer effective. This work attempts to categorize the traffic flow based on its size and rate using supervised ML algorithms. It is worth to mention here that we have collected the data according to the accumulative value of packets and bytes that pass through the forwarding elements.

III. D-ITG Dataset

For the purpose of evaluating the proposed classification methods, we built a dataset using a synthetic traffic generator. First, a small-scale network topology was created using the Mininet emulator[10], which consisted of two hosts, one switch and one central controller. Each host in the network was connected to an OpenvSwitch (OVS) [11]. The OVS was connected to a Ryu controller by a secure OpenVFlow channel. As shown in Fig. 1, we developed two models in the application layer: (1) a monitoring model, which captures the instant flows and (2) a collecting model, which archives the traces that will be divided into training and testing datasets. Traffic flows were generated by the Distributed Internet Traffic

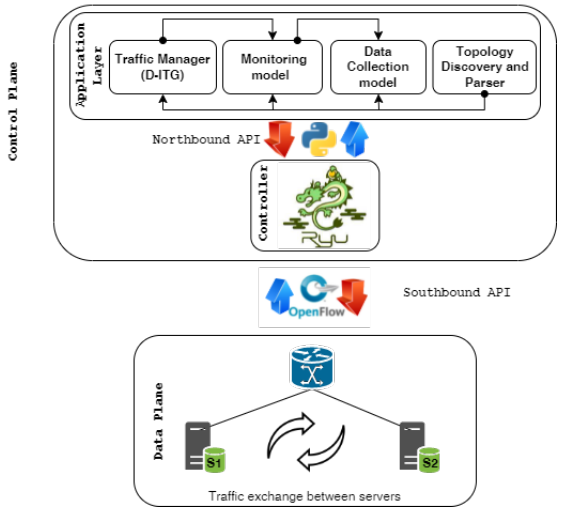

Fig. 1. The structure of the SDN network in the data collection phase, including the monitoring and the collecting models. Python is used on the northbound interface, and OpenFlow is used on the southbound interface.

Generator (D-ITG) tool[12]. We used the D-ITG to replicate statistical properties of 6 traffic applications including DNS, telnet, ping, gaming, quake-3 and VoIP. In addition, we simulated a video traffic by using VLC to generate real-time video streaming between a VLC server and client. When traffic passed through the network switch, the monitoring model sent data to the collecting model that extracted and archived the statistics of the arrival flows. The collected statistics consists of the accumulative packet and byte values to forward flows i.e., data frames, and reverse flows i.e., control frames, from which we derived other features e.g., average forward/reverse packets, delta forward/reverse packets, and forward/reverse instantaneous packets per second. We ran each traffic individually for approximately 150 minutes. We reported up to 17000 observations and 16 features for each of the experimental flows.

IV. Classification Models

This section presents the implementation of three ML models, namely logistic regression, SVM and neural network with different types of solvers. We used Python 3.8 and Windows 10 with Core i7 and 8 GB RAM to implement these models. The test of the models is performed on a portion of the dataset (i.e., test data). It is worth mentioning here that we dropped the features of "forward-byte", "forward-packets", "reverse-byte" and "reverse-packet". This is due to the cumulative nature of those features. They may have significantly different values from those that occur in the training phase. For instance, while those features start counting from zero in the training phase, they may start from another initial value in real traffic, which depends on the dynamic behavior of the network. Therefore, ML models might be negatively affected by the misleading information of such features.

A. Logistic Regression model

To construct a Logistic Regression (LR) model, the input is the dot product between a weight vector $\theta \in \mathbb{R}^n$ and an

TABLE I
DESCRIPTION OF THE LOGISTIC REGRESSION MODELS

LR model number	Penalize level	Solver type	Iteration number	Accuracy (%)	Time (sec) of convergence
1	Penalty function/ L2	saga	25	75.31	03.56
2	Penalty function/ L1	liblinear	40	28.23	11.24
3	Penalty function/ L2	newton-cg	18	75.29	08.49
4	Penalty function/ L2	sag	19	75.60	01.67

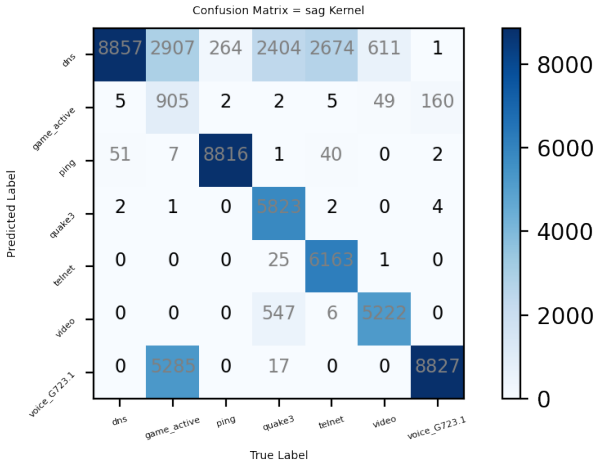

Fig. 2. Confusion matrix for the LR-SAG model.

input vector $\mathbf{x}$ (i.e., input training data $x_k \in \mathbb{R}^n$) plus a bias b_k. And the output is the probability $\hat{y}_k \in \mathbb{R}$ to select the class k that maximizes (1), see [13]:

$$p\left(\mathbf{y}_k = 1 \mid \mathbf{x}\right) = \frac{\exp\left(\theta_k \cdot \mathbf{x} + b_k\right)}{\sum_{j=1}^{N} \exp\left(\theta_j \cdot \mathbf{x} + b_j\right)} \tag{1}$$

where, N is the number of classes.

For the purposes of regularization, feature selection, and providing better long-term predictions, we use Ridge Regression (i.e., ℓ_2 Regularization) and Lasso Regression (i.e., ℓ_1 Regularization)[14], [15]. The inverse of regularization strength (C) set default value is 1.0.

Table I shows the results of the models' training based on the training time and accuracy value. We observed that models with solvers sag, saga, and newton had approximately the same accuracy, but different convergence times. On the other hand, the liblinear solver has inferior accuracy compared to the above ones. In Fig. 2, the confusion matrix of the sag model helps to identify where the model is unsuccessful in accurately determining the target. We observed an apparent misclassification of game, telnet, quake-3, video, and ping traffic as DNS traffic. Furthermore, there is a significant number of the game traffic overlap with voice traffic. This is reasonable as both of their traffic sources are interactive types of traffic. This is reflected in the precision value for each class of traffic. In the first and second columns of the classification report in Table II. The report shows the precision and recall values, respectively. The minimum precision value is observed

TABLE II
CLASSIFICATION REPORT FOR THE LR-SAG MODEL.

Traffics	Precision	Recall	F-score	Support
DNS	0.99	0.51	0.67	17830
game	0.15	0.80	0.25	1684
ping	0.97	0.98	0.98	8843
quake-3	0.66	1.0	0.80	5880
telnet	0.69	0.99	0.81	6184
video	8.88	0.98	0.89	5817
voice	0.97	0.64	0.77	13450
Accuracy			0.75	59688
Macro avg	0.76	0.83	0.74	59688
Weighted avg	0.89	0.75	0.78	59688

in the case of game traffic, and the maximum in the case of DNS traffic, since a lot of the game's flow are classified as DNS or voice traffic. On the other hand, the maximum recall provided for quake-3 traffic, and the minimum is for DNS traffic, due to the fact that many flows are misclassified as DNS traffic. The F-score (2) combines precision and recall to have an overall score that depicts our model's performance.

$$\text{F-score} = \frac{2}{1/\text{ precision } + 1/\text{ recall}} \tag{2}$$

Ping traffic gets the highest F-score, because it has the highest precision and recall scores.

B. SVM model

Like in LR model, we train the SVM model with input training data $x_k \in \mathbb{R}^n$ and output label data $y_k \in \mathbb{R}$ with $(N = 7)$ classes mentioned in the Table II to construct the Support Vector Classifier (SVC) in the form [16]:

$$y(x) = \text{sign}\left[\sum_{k=1}^{N} \alpha_k y_k \psi\left(x, x_k\right) + b_k\right] \tag{3}$$

where α_k is a positive real constant, $\psi\left(x, x_k\right)$ represents the kernel, and b_k is the bias. In order to get the best decision boundary between classes, we use four different kernels as follows:

1) Linear kernel:
$$\psi\left(x, x_k\right) = x_k^T x$$
2) Polynomial-kernel:
$$\psi\left(x, x_k\right) = \left(x_k^T x + r\right)^d$$
3) Radial Basis Function (RBF) kernel:
$$\psi\left(x, x_k\right) = \exp\left\{-\left\|x - x_k\right\|_2^2 / \sigma^2\right\}$$
4) Sigmoid kernel:
$$\psi\left(x, x_k\right) = \tanh\left[\gamma x_k^T x + r\right]$$

where r, d, σ and γ are the kernel parameters used to maintain the separation line between classes.

According to Table III, the sigmoid kernel reported significantly less accuracy compared to the other kernels. Despite that the linear kernel achieved 80.64% accuracy, the performance in online real test cannot be guaranteed due to the possibility of nonlinearity of the data. Thus, the use of

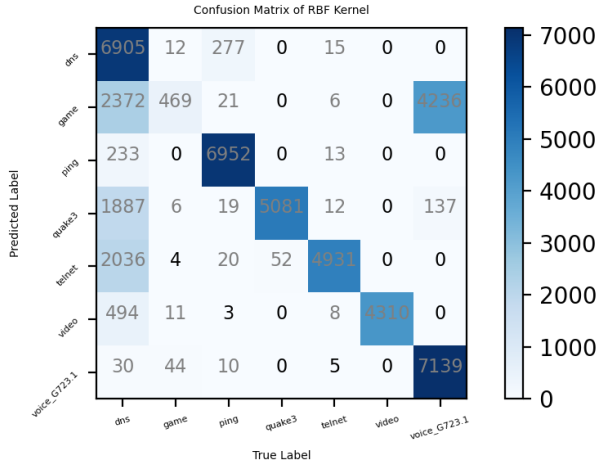

Fig. 3. Confusion Matrix for the SVM (RBF kernel) model.

TABLE III
CLASSIFICATION REPORT FOR THE SVM MODELS

SVM model number	Kernel type	Trade-off parameter (C value)	Accuracy (%)	Training Time (sec)
1	Linear	1	80.64	82
2	Polynomial	3	72.77	144
3	RBF	1	85.35	48
4	Sigmoid	1	62.36	107

sigmoid and the linear kernel is not the best choice. Although the polynomial and RBF kernel are preferred for non-linear hyper-plane and separation of high dimensional classes, the polynomial has achieved less accuracy compared to RBF kernel and consumed more time than other kernels. The C value is used to preserve the regularization and to control the trade-off between margins and misclassification term. We believe that the misclassification in polynomial kernel is coming from the higher trade-off C parameter, which leads to larger margin hyper-plane and makes the classification even more difficult in this model. Interestingly, the RBF achieved the highest accuracy in shorter time compared to the rest of the models. This can be clearly seen in the confusion matrix for the RBF kernel, that is illustrated in Fig. 3, in which the most predicted classes were labelled correctly.

It is apparent from Fig. 3 that game, DNS and voice traffic are overlapping with each other, which leads to false labeling as Table IV shows that DNS, game and voice resulted the lowest precision and F-score. In summary, the results indicate that for certain parameters, the linear kernel can get higher accuracy as the RBF kernel, however, this comes at an additional cost in training time.

C. Neural Network model

In order to construct a Neural Network (NN) model for the classification task, we stack 7 sequential layers as shown in Table V. The model consists of the following layers:

(1) Layer normalization [17] to scale and shift input values with zero mean and unit variance. The layer normalization statistics are computed over all the input values $x_1, \ldots, x_n$ as follows:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2}$$

(2-3) Densely connected layers of size 32 and 64, respectively, using the Rectifier Linear Unit (ReLU) activation function $f(x) = \max(0, x)$.

(4) A dropout layer with 50% dropping rate, for the sake of regularization and to avoid overfitting.

(5-6) Densely connected layers of size 28 and 14, respectively, using ReLU.

(7) A densely connected layer of size 7, followed by a softmax activation (4), which outputs one of the $N = 7$ class labels.

$$\sigma(\mathbf{x})_i = \frac{\exp(x_i)}{\sum_{j=1}^{N}\exp(x_j)} \tag{4}$$

During training, we use the sparse categorical cross entropy loss function to cope with this multi-class classification problem where labels are provided as integers. As an optimization technique for gradient descent, we run the experiments with the Stochastic Gradient Descent (SGD) method as well as the Adaptive Moment Estimation (Adam) algorithm.

Table VI shows the results of the NN models' training and validation, including the accuracy values and training time.

TABLE IV
CLASSIFICATION REPORT FOR SVM (RBF KERNEL) MODEL.

Traffics	Precision	Recall	F-score	Support
DNS	0.49	0.96	0.65	7209
game	0.86	0.07	0.12	7104
ping	0.95	0.97	0.96	7198
quake-3	0.99	0.71	0.83	7142
telnet	0.99	0.70	0.82	7043
video	1.00	0.89	0.94	4826
voice	0.62	0.99	0.76	7228
Accuracy			0.75	47750
Macro avg	0.84	0.75	0.73	47750
Weighted avg	0.83	0.75	0.72	47750

TABLE V
DESCRIPTION OF THE NN MODEL.

Layer type	**Output Shape**	**Param #**
Normalization	(None, 12)	25
Dense	(None, 24)	312
Dense	(None, 48)	1200
Dropout	(None, 48)	0
Dense	(None, 28)	1372
Dense	(None, 14)	406
Dense	(None, 7)	105
Total params: 3420		
Trainable params: 3395		
Non-trainable params: 25		

TABLE VI
CLASSIFICATION REPORT FOR THE NN MODELS.

NN model number	Dropping rate	Optimizer type	Epoch number	Accuracy (%)	Time (sec) of convergence
1	0.5	Adam	10	89	31.00
2	0.5	SGD	10	79.94	32.00

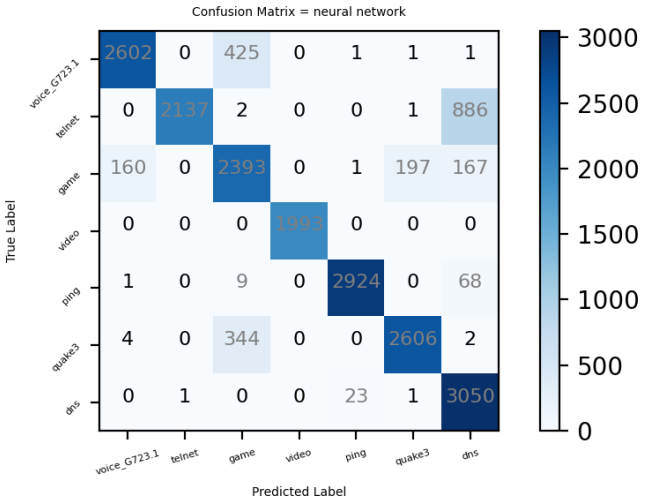

Fig. 4. Confusion Matrix for the NN model.

TABLE VII
CLASSIFICATION REPORT FOR THE NN (ADAM OPTIMIZER) MODEL.

Traffics	Precision	Recall	F-score	Support
voice	0.94	0.86	0.90	3030
telnet	1.00	0.71	0.83	3026
game	0.75	0.82	0.79	2918
video	1.00	1.00	1.00	1993
ping	0.99	0.97	0.98	3002
quake-3	0.93	0.88	0.90	2956
DNS	0.73	0.99	0.84	3075
Accuracy			89	20000
Macro avg	0.91	0.89	0.89	20000
Weighted avg	0.90	0.89	0.89	20000

According to Table VI, we observed that both SGD and Adam optimizers had approximately the same convergence time, but different accuracy. Therefore, we employ the Adam optimizer to test our approach in real-time action in Section V. Fig. 4 shows the confusion matrix for the model using Adam optimizer. We observed a significant number of games flows categorized as telnet, quake-3 and DNS traffic. Furthermore, there is a slight overlap between telnet and DNS traffic. This is reflected in the F-score values for each class of traffic, as shown by the classification report in Table VII.

On one hand, the minimum precision value was reported in the case of DNS traffic. On the other hand, the minimum recall value was reported in the case of game traffic. This is due to many flows being misclassified as DNS, quake-3, voice, and telnet traffic. Therefore, game, telnet, and DNS traffic resulted in a lower F-score value than other types of traffic did.

V. ONLINE TESTING

In this section, we discuss the evaluation the NN model in the context of predicting the labels of the newly arrival flows in a real-time network scenario. To do so, we integrated and incorporated the NN model with the SDN controller. Then, we simulated each traffic type for approximately 10 minutes following a burst of 1000 flows of each traffic type. The testbed was built using a Linux 20.0.4 LTS, 8 GB RAM, Core i7 machine. We created the network topology based

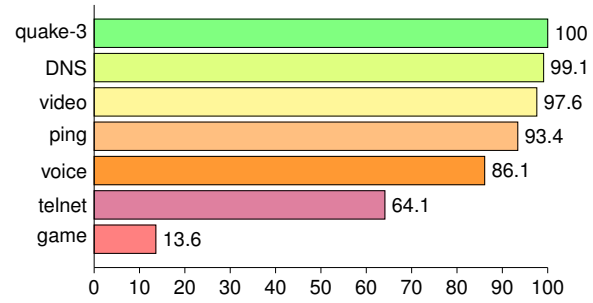

Fig. 5. Result of SDN testing

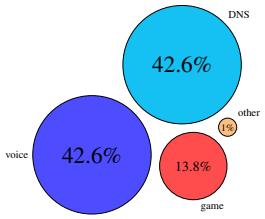

Fig. 6. SDN testing on game traffic

on the Mininet emulator with some other modules including monitoring and topology discovering. We adopted Ryu as a network controller.

Fig. 5 shows the percentage of accurate labels for each traffic class. It can be clearly seen that the classification model has 100% accuracy when predicting quake-3 traffic. Similarly, the model performs well to predict the traffic class of DNS, video and ping. According to Fig. 5, the accuracy of predicting DNS is up to 99%, while the accuracy is up to 97% for video and up to 93% for ping. For VoIP traffic, the classification model fails to classify up to 14% from the injected traffic. In the same vein, the model does not perform well in case of telnet traffic since up to 35% of the generated telnet traffic has been wrongly classified. Finally, the performance of the classification model was reported as the lowest in terms of predicting game traffic since the accuracy was up to 13%.

VI. DISCUSSION

In this section, we compare the results that were obtained from the training and validation phase with the results of the online testing for the NN model (with Adam optimizer). On one hand, during the online testing, the worst case was provided by the class of game traffic, as illustrated by Fig. 6, while, in the validation phase, game traffic provided the least F-score of 79%. On the other hand, during the online testing, we reported the best results for both quake-3 and video traffic approximately without any misclassification. By considering the validation phase, we can see that both quake-3 and video traffics had a high F-score value, namely 0.90 and 1, respectively. Furthermore, we observed the same impact and correlation in the case of telnet traffic, where online testing resulted in 64.1% and validation resulted in an F-score of 0.83.

In this work, the lightweight approach to classify/predict the network arrival traffic based on the packets bit rate only without the need of any further inspection such as decapsulation

the packet's header to retrieve additional data. This is a big challenging problem due to the fact that all the derived features are based on the size and the speed of flow packets. Although the proof of concept design works well, it is limited in some cases since some of the features measure the average packet and the size. Scenarios in which flows are interrupted due to network incidents might decrease the values of those features and, consequently, reduce the accuracy of the proposed model.

VII. Conclusion

This paper demonstrated the promise of applying machine learning approaches to the problem of traffic classification for SDNs. Three machine learning models LR, SVM and NN were implemented and evaluated in the context of classifying network traffic of different classes based on 16 selected features. We used the D-ITG in order to built a synthetic traffic dataset for this purpose. After having the three models tested, we incorporated only the NN model in a SDN implementation so that further measurements could be conducted in an online manner. Four metrics were used in the process of evaluation, these are: accuracy, precision, recall and F-measure. The experimental findings revealed that the NN model was capable of identifying wide range of traffic classes with high accuracy, however, it reported low accuracy for some classes. Since in this work we depend on MAC addresses as identifiers of hosts in the SDN, hence we did not consider parallel traffics between the same hosts. However, as future work, we can adopt new functionality in the upper network layers, i.e., transport and network layers. We used LASSO/L1 regression, which performs feature selection by driving certain weights to zero. Furthermore, we could apply more future investigations to the features to derive other features which might improve performance. As a future plan, we intend to combine the proposed model with our FPLF algorithm [2] in order to optimize the power consumption of software-defined data center networks and wireless networks [18], [19].

VIII. Acknowledgement

The authors extend gratitude and appreciation to the Information Technology department, University of Debrecen, to support the work and to fund it. This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under the Grant Number 15/SIRG/3459.

References

[1] Tatchell-Evans, Morgan and Kapur, Nik and Summers, Jonathan and Thompson, Harvey and Oldham, Dan., "An experimental and theoretical investigation of the extent of bypass air within data centres employing aisle containment, and its impact on power consumption," Applied energy, vol. 186, pp. 457–469, January 2017.

[2] Nsaif, Mohammed, Gergely Kovásznai, Anett Rácz, Ali Malik, and Ruairí de Fréin, "An Adaptive Routing Framework for Efficient Power Consumption in Software-Defined Datacenter Networks," Electronics, vol. A10, pp. 3027, December 2021.

[3] Al Khater, Noora, and Richard E. Overill. "Network traffic classification techniques and challenges." 2015 Tenth international conference on digital information management (ICDIM). IEEE, 2015.

[4] Chiu, Kai-Cheng, Chien-Chang Liu, and Li-Der Chou. "CAPC: Packet-Based Network Service Classifier With Convolutional Autoencoder." IEEE Access 8 (2020): 218081-218094.

[5] Da Silva, Anderson Santos and Machado, Cristian Cleder and Bisol, Rodolfo Vebber and Granville, Lisandro Zambenedetti and Schaeffer-Filho, Alberto., "Identification and selection of flow features for accurate traffic classification in SDN," 2015 IEEE 14th International Symposium on Network Computing and Applications, pp. 134–141, 2015.

[6] Amaral, Pedro and Dinis, Joao and Pinto, Paulo and Bernardo, Luis and Tavares, Joao and Mamede, Henrique S., "Machine learning in software defined networks: Data collection and traffic classification" 2016 IEEE 24th International conference on network protocols (ICNP), pp. 1–5, 2016.

[7] Parsaei, Mohammad Reza and Sobouti, Mohammad Javad and Khayami, Seyed Raouf and Javidan, Reza., "Network traffic classification using machine learning techniques over software defined networks" International Journal of Advanced Computer Science and Applications,vol. 8, pp. 220–225, 2017.

[8] Malik, Ali and de Fréin, Ruairí and Al-Zeyadi, Mohammed and Andreu-Perez, Javier., "Intelligent SDN traffic classification using deep learning: Deep-SDN" 2020 2nd International Conference on Computer Communication and the Internet (ICCCI), pp. 184–189, 2020.

[9] R. de Fréin, O. Izima and A. Malik., "Detecting Network State in the Presence of Varying Levels of Congestion," 2021 IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP), pp. 1-6, 2021.

[10] Lantz, Bob and Heller, Brandon and McKeown, Nick., "A network in a laptop: rapid prototyping for software-defined networks" Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks, pp. 1–6, 2010.

[11] McKeown, Nick and Anderson, Tom and Balakrishnan, Hari and Parulkar, Guru and Peterson, Larry and Rexford, Jennifer and Shenker, Scott and Turner, Jonathan., "OpenFlow: enabling innovation in campus networks" ACM SIGCOMM computer communication review, vol.38,pp. 69–74, 2008.

[12] Walter de Donato; Alberto Dainotti; Stefano Avallone; and Pescapde Donato, M. *D-ITG 2.8.1 Manual*; COMICS (COMputer for Interaction and CommunicationS). Available online: http://www.traffic.comics.unina.it/software/ITG/ (accessed on 1 August 2021).

[13] Daniel Jurafsky, James H. Martin,"Speech and Language Processing",pp. 91–92, Draft of December 30, 2020.

[14] Baraniuk, Richard G., "Compressive sensing [lecture notes]" IEEE signal processing magazine, vol.24,pp. 118–121, 2007.

[15] McDonald, Gary C., "Ridge regression" Wiley Interdisciplinary Reviews: Computational Statistics, vol.1,pp. 93–100, 2009.

[16] Huang, Xiaolin and Maier, Andreas and Hornegger, Joachim and Suykens, Johan AK, "Indefinite kernels in least squares support vector machines and principal component analysis", Applied and Computational Harmonic Analysis,vol. 43,pp. 162–172,Elsevier, 2017.

[17] Ba, Jimmy Lei and Kiros, Jamie Ryan and Hinton, Geoffrey E, "Layer normalization", arXiv preprint arXiv:1607.06450, 2016.

[18] Rabee, Furkan and Al-Haboobi, A and Nsaif, Mohammed Ridha, "Parallel three-way handshaking route in mobile crowd sensing (PT-MCS)," J Eng Appl Sci, vol. 14, pp. 3200–3209, 2019.

[19] Mohammed Ridha Nsaif, Ali S. A. Al-Haboobi, Furkan Rabee, and Farah A. Alasadi, "Reliable Compression Route Protocol for Mobile Crowd Sensing (RCR-MSC)," Journal of Communications, vol. 14, pp. 170-178, March 2019.