Proceedings of 13th IOE Graduate Conference
Peer Reviewed
ISSN: 2350-8914 (Online), 2350-8906 (Print)
Year: 2023 Month: April Volume: 13

Traffic Classification and Load Balancing in SDN Environment

Gopal Karn [a], Binod Sapkota [b], Babu R. Dawadi [c]

[a, b] *Department of Electronics and Computer Engineering, Thapathali Campus, IOE, TU, Nepal*
[c] *Department of Electronics and Computer Engineering, Pulchowk Campus, IOE, TU, Nepal*
[a] gopal76msiise10@tcioe.edu.np, [b] replybinod@gmail.com, [c] baburd@gmail.com

Abstract
Software-Defined Networking (SDN) is a modern networking approach that is flexible, manageable, cost-effective, and adaptable. By decoupling the control plane from the data plane, SDN enables flexible network management. In a network, all traffic doesn't have same level of criticality and routing the traffic through different paths on basis of class of traffic helps in routing traffic among components of the network, consequently improving the overall Quality of Service (QoS). In this paper, a comparative analysis of KMeans Clustering, Logistic Regression, XGBoost, Classification and Regression Tree (CART) are done on basis of accuracy, precision, recall and f-1 score to find the best machine learning algorithm for classification of SDN traffic. The best model among all i.e., Classification and Regression Tree (CART) is then used for real time traffic classification along with load balancing module of Ryu controller for routing based on traffic class. The shortest path between source and destination is defined using Depth First Search (DFS) algorithm. Secondary controllers are implemented to handle the overload problem on main controller. The overall system improves the QoS of overall network and also solves the reliability issue existing in a single controller network.

Keywords
Decision Tree, Load balancing, Quality of Service (QoS), Software Defined Networking, Traffic Classification

1. Introduction

Categorizing network traffic is highly important within the field of computer science, as is managing network performance for Internet service providers (ISPs). The first step in identifying and classifying unknown types of network traffic is to perform traffic classification. Properly classifying network traffic is essential for the effective functioning of network security and management tools, such as load balancing and quality of service (QoS) [1].

1.1 Traffic Classification

The SDN controller can gather notification about the application layer from traffic categorization. Depending on the application being used, this information may be utilized to develop routing and/or access policies. However, this operation must be carried out in real-time to classify a flow based on just a few of its initial packets. Port-based traffic classification was effective in early days when each protocol used specific ports. However, this port-based method is ineffective in current scenario as most applications use dynamic port to avoid detection by intruders.

Payload-based method looks for the features and signatures of network applications while examining the packet contents in network traffic. The problem with this method is that it requires expensive hardware for recognition of pattern and is not feasible for encrypted network application traffic. Also, a continuous update of signature pattern is required for this method to work efficiently [2].

1.1.1 Machine Learning based classification

The use of traffic classification methodologies based on machine learning has lately solved the drawbacks of traditional traffic classification methods. ML-based approaches that are based on statistics, such as packet length distribution and packet inter-arrival duration, can identify between various applications. The ultimate goal of ML-based techniques is the classification of distinct applications or the grouping of traffic flows into corresponding groups with comparable patterns. Comparatively to solutions based on Deep Packet Inspection, ML-based algorithms for identifying traffic can categorize encrypted communication without the requirement to view packet contents making it less expensive computationally.

The paper [3] talks about various machine learning approaches like classification (supervised learning), clustering (unsupervised learning) and association that can be used for IP traffic classification. In our work, machine learning approaches like K-Means Clustering, Support Vector Machine (SVM), Logistic Regression, XGBoost, Ordering Points to Identify the Clustering Structure (OPTICS) and Classification And Regression Tree (CART) were tested to find best method for classification of SDN traffic.

K-Means Clustering is implemented to test if the datasets form the desired cluster for each protocol name after dropping the protocol from the dataset.

Other algorithms like SVM, Logistics Regression, XBoost, OPTICS and CART are used as these algorithms are found to be suitable to classify non-linear multi-class dataset in other research works.

1.2 Traffic type based Load balancing

Load balancers are tasked with continuously distributing the workload among all the components in the system in order to improve system performance and resource utilization. The load balancing algorithms discussed and analysed by the authors in [4] are: Round-Robin, Least connection based, Agent adaptive, Fixed Weighted, Weighted Response time, Source IP hash and

Software Defined Networking (SDN) adaptive.

In our work, SDN adaptive load balancing algorithm is implemented. This strategy makes use of both information about the status of the network at bottom layers and understanding of the upper network layers. The information of traffic type from classification module is used to create the flow and group tables which routes different traffic on different path leading to overall balance of resource utilization in network along with reducing the delay in transmission of traffic from source to destination.

1.3 Contribution

Most load balancing algorithms implemented in SDN are based on throughput or CPU utilization and do not take into account the incoming traffic type; as a result, they are unable to identify the type of traffic that is important and critical for a given network since not all incoming traffic has the same level of criticality. Being able to route the traffic on different paths based on its type allows to give the shortest path or path with minimum latency to the type of traffic with highest criticality given the criticality of traffic is pre-defined.

Thus, this study resolves real-time load balancing to enhance network performance and system QoS as well as traffic classification to identify vital traffic.

2. Background and Related Works

2.1 Need for traffic based load balancing

Due to increasing demand for large scale networks that can process huge amounts of data efficiently and with negligible latency, various attempts are made to improve network. One attempt among all that has been a huge breakthrough is implementation of SDN in place of traditional manual network for ease in network management.

In real world scenario, all network traffics don't have same level of criticality, for example, voice and video traffic can hold higher importance than other data traffic. Identification of type of traffic supports in managing the resources and allocating it to the critical application thus ensuring the improvement in QoS and stability of system. Most existing load balancing techniques are based on amount of incoming load or resource consumed but current scenario demands a load balancing algorithm which can balance and manage traffic based on traffic type dynamically.

2.2 Related Works

To speed up response time during the changing of rules in the SDN flow table, Liao et al. [5] presented a dynamic load-balancing technique. By using wildcard rules, the system creates a flow table and keeps track of the servers in the SDN network. To ensure that the server can execute the request by redirecting it to other servers with light loads, the load balancer switch notifies the controller to update the flow table if a target server is at capacity.

Hailong et al. [6] implemented Equal Cost Multipath (ECMP) routing as a successful load balancing strategy. When a packet arrives at a switch or router, this approach employs a hash algorithm to determine its header fields and then chooses one of the forwarding paths based on the hash value. The scheduler module directs traffic to underutilized links if the threshold is crossed.

Packets with comparable headers are consequently sent on the same routing. A major drawback of ECMP is that it will route multiple huge, drawn-out data flows to the same output port if they collide on their hash, which can result in a bottleneck. However, Curtis et al. [7] used external backend server to detect elephant flows. The Mahout scheduling mechanism technique reduces packet overhead and latency. Two priority levels are used in this process, and each level has a related rule in the flow table. While elephant flows (high priority) traffic is directed to a Mahout-based controller for optimal route calculation, low level priority traffic was sent using the normal ECMP approach.

Mekky et al. [8] suggested a per-application flow metering strategy utilizing the SDN architecture. Based on specific application tables, applications were recognized in the data plane and provided the relevant policies. The SDN control channel's overhead is reduced by this method. The investigation discovered a considerable reduction in overhead and an increase in application forwarding performance.

Jinghua et al. [9] compared various models that can be implemented to perform traffic classification in software defined networks and gave comments about them for future implementation. Santos de Silva et al. classified DDoS, FTP, and video traffic using support vector machines (SVM). Although the algorithm provided fine-grained categorization, it could not classify the traffic in real-time.

Similar to this, Pedro Amaral et al [10] implementation of stochastic gradient boosting, extreme gradient boosting, and random forest classifier to classify enterprise network traces was also successful in classifying the traffic from dataset effectively, but it was unsuccessful in doing so in real time. The author also makes the argument that developing a dynamic traffic classification algorithm that can adapt to the needs of future networks is necessary in order to classify traffic in the future with new incoming protocols.

3. Proposed Approach

In order to manage the load in the network, the traffic incoming on all the controllers need to be monitored and analyzed. The traffic is first monitored by the controller, features are extracted and then the trained classification model classifies the incoming data into various pre-defined classes. After the steps of traffic engineering, the packets are forwarded to respective controllers assigned for each traffic type and then the data is send to the destination address based on the priority and queuing policies defined in the controllers and switches.

The detailed SDN architecture with components of each layer and their functionalities is show in figure 2.

1. **Infrastructure layer:** This layer basically consists of OVS switches which are connected to the hosts. The switches are connected to a control layer to send the traffic to the controller and then receive commands from the controller. Using the OpenFlow protocols, the switches maintain the flow table and group table which

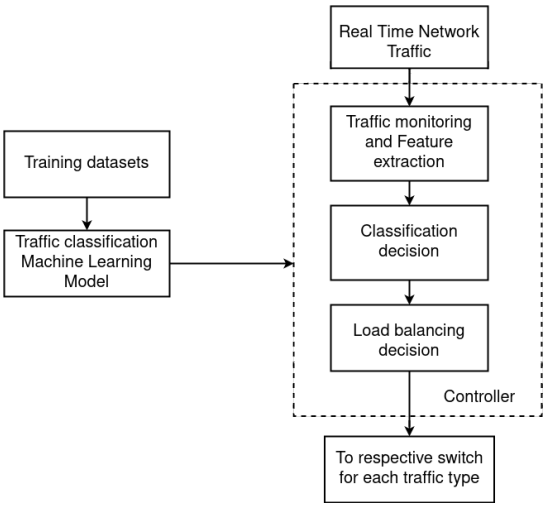

Figure 1: Proposed methodology

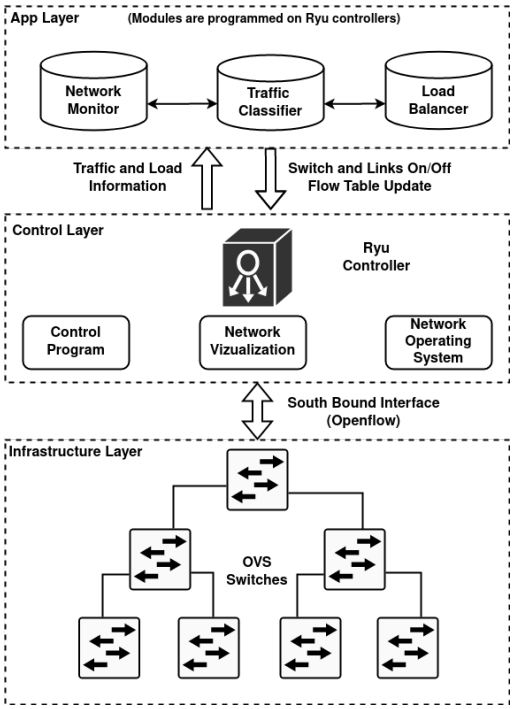

Figure 2: SDN System Architecture

are responsible for handling the known traffic incoming into the network from the hosts.

2. **Control layer:** The control plane of SDN, commonly known as the controller, manages and programs forwarding devices through the southbound interface. It determines the routing of traffic in the network based on application requirements and communicates the policies to the data plane. The controller acts as the central brain and network operating system in the network. It converts application requirements such as QoS, traffic prioritization, and bandwidth management into forwarding rules that are sent to the forwarding elements of the data plane network.

3. **App layer:** This layer implements various applications of the system like network monitor which monitors the incoming traffic along with bytes count, time delay in traffic and other parameters. Another application to be implemented is traffic classifier that will implement machine learning to classify the incoming traffic in real time into various categories. The third application is load balancing which re-routes the traffic incoming into the network to the controller of choice based on its type.

3.1 Explanation of training dataset

Data on traffic flow was produced using the Distributed Internet Traffic Generator (D-ITG) [11] application and utilized to train machine learning models. D-ITG is a platform that can generate both IPv4 and IPv6 traffic via precisely simulating the workload of existing Internet applications.

The following traffic categories were employed for traffic classification training dataset: Ping, Telnet, DNS, Voice, and Video. Three classes were used for classification; Ping, Telnet, and DNS traffic were placed in the ”Data” class, while voice and video traffic were placed in the ”Voice” and ”Video” classes, respectively. A sample of dataset is shown in table 1

Table 1: Training Dataset Sample

Forward Packets	Forward Average Packets/second	Reverse Packets	Traffic Type
454	1.166666667	507	Voice
790	0.923076923	1018	Voice
856	0.105691057	508	Data
856	0.10483871	508	Data
856	0.09352518	507	Video
1622	0.320512821	507	Video

The features of data that are extracted and used for training purpose are shown in table 2. Some of the features are dropped after feature selection during model training phase due to irrelevance of feature in traffic classification.

Table 2: Features used for traffic classification

Feature Label	Feature
0	Delta Forward Packets
1	Delta Forward Bytes
2	Forward Instantaneous Packets per Second
3	Forward Average Packets per second
4	Forward Instantaneous Bytes per Second
5	Forward Average Bytes per second
6	Delta Reverse Packets
7	Delta Reverse Bytes
8	Delta Reverse Instantaneous Packets per Second
9	Reverse Average Packets per second
10	Reverse Instantaneous Bytes per Second
11	Reverse Average Bytes per second

3.2 Working Principle

The detailed explanation on how the network traffic classifier and load balancer module of the controller work is described as follows. When a new packet arrives into the network, the first switch checks if there is a matched flow entry in the flow table. If there is, the packet is sent directly to destination without calling in the controller. This reduces overload on controller. If there is no flow entry, the packet is sent to controller, the controller extracts the features from the packet to predict the traffic type and returns a queue ID based on type. If a packet is not classified, it is given default less priority queue.
The queue ID is then associated with output port in flow entry by

the controller which determines, if the packet is sent through the same controller or sent through another controller. Lastly, flow is added into each switch's flow table for that particular data path. The switches have the ability to accept additional packets from the same flow and route them with a predefined protocol to their destination.

3.3 Experimental Setup

To evaluate the working and performance of the proposed model, a virtual network is emulated using mininet [12] as topology emulator, Ryu [13, 14] as the controller responsible for performing duties of traffic classification and take load balancing decision. Open vSwitch [15] is used as switching element which maintains flow tables and group tables of various rules implemented for routing specific type of traffic to specific path.
The network consists of a Ryu controllers, six switches and three hosts (one sender and two receiver as shown in figure 3.

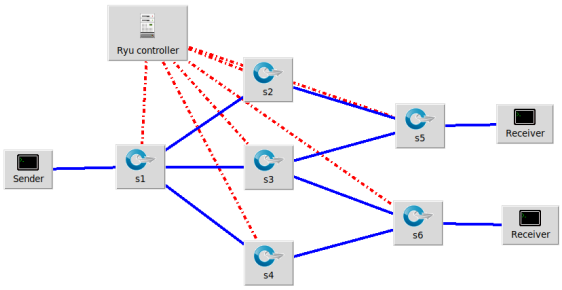

Figure 3: Topology used for network simulation

The Ryu framework, which serves as the SDN controller, is intended to operate on a host computer while having a TCP connection to the Mininet network topology that is being simulated. The three paths are used for video, voice and data traffic respectively.

4. Result and Analysis

4.1 Comparison of classification models on generated dataset

Table 3 discusses about evaluation of various machine learning algorithm on basis of different parameters while figure 4 demonstrates comparative plot of these algorithms.

Table 3: Measure of traffic classification on generated dataset

ML Algorithm	Accuracy (%)	Precision (%)	Recall (%)	F-1 Score (%)
K-Means Clustering	32	27	26	26
Support Vector Machine (SVM)	82	85	82	81
Logistic Regression	63	63	63	62
XGBoost	91	94	94	94
OPTICS	56	49	51	51
Decision Tree	98	99	99	99

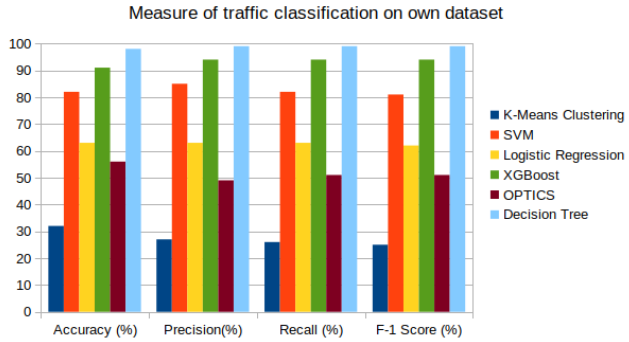

Figure 4: Evaluated result of different algorithms for training dataset

From above results, it is observed that K-means clustering has worst result on all parameters. This is because K-means clustering implies better on spherical and equal-sized cluster data, whereas our data is non-linear in shape and also different class does not form equal-sized cluster. As a result, during multi-class classification, K-means is unable to reflect the fundamental structure of the data as well as the class boundaries.

The best classification report is given by decision tree classifier (CART) algorithm. CART algorithm is most suitable for multi-class classification as shown by accuracy, precision, recall and F-1 score of all traffic types. Decision trees are a straightforward, quick, and efficient solution for multi-class classification issues.

The execution time of each machine learning algorithm in classification of dataset with almost 21000 data is shown in figure 5 From above graph, it is seen that decision tree (CART)

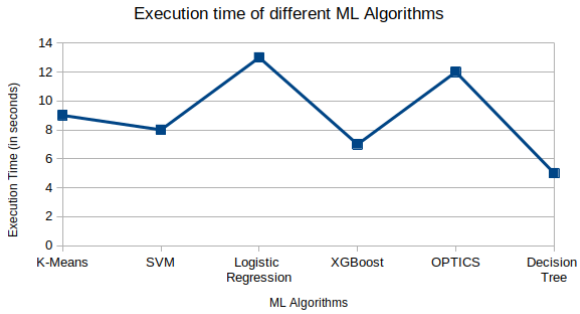

Figure 5: Execution time of different ML algorithms

has least execution time of 5 seconds whereas Logistic Regression has highest execution time of 13 seconds.

4.2 Comparison of classification models on kaggle dataset

While comparing the classification model on a dataset with 87 features and over 3.5 million instances [16], the following result were obtained. 5

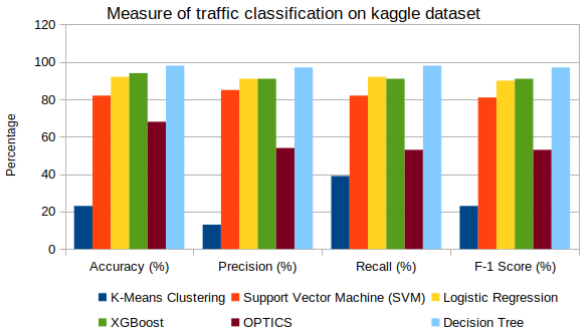

Figure 6: Evaluated results of algorithms for kaggle dataset

The graph 6 demonstrates that decision tree (CART) algorithm outperforms other algorithms in terms of accuracy, precision, recall and F-1 score which further justifies outcome of classification on generated dataset and validates the application of use of CART algorithm for classification purpose. On testing the CART algorithm on both the generated dataset and kaggle dataset while keeping the features set the same, it performs better than other algorithms which ensures that CART will always classify any multi-class non-linear dataset most efficiently.

4.3 Evaluation of load balancing

The figure 7 demonstrates the the advantage of load balancing in improving the performance of network by comparing bit-rate and packet rate.

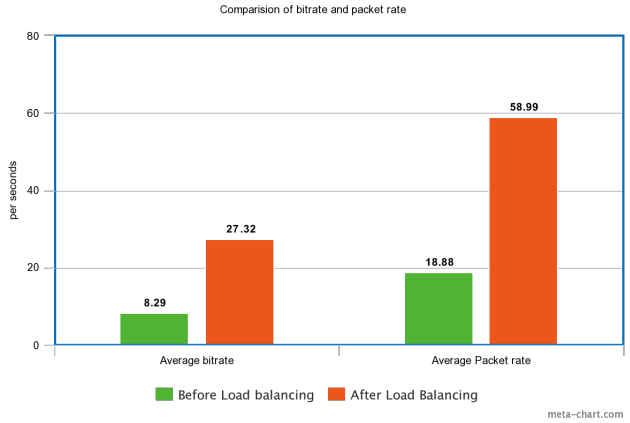

Figure 7: Evaluated result of different algorithms for training dataset

While running the classification and load balancing module in tandem for one minute and sending all three types of traffic, namely voice, video and data, it is observed that bit-rate and packet-rate is improved by almost 3 times suggesting large amount of traffic can be sent using same resource in short period of time by implementing traffic based load balancing.

Also, Jitter monitors the variance in packet latency as it travels over the network. A network's jitter is reduced with improved load balancing, which enhances the QoS for real-time applications like voice and video as seen in figure 8.

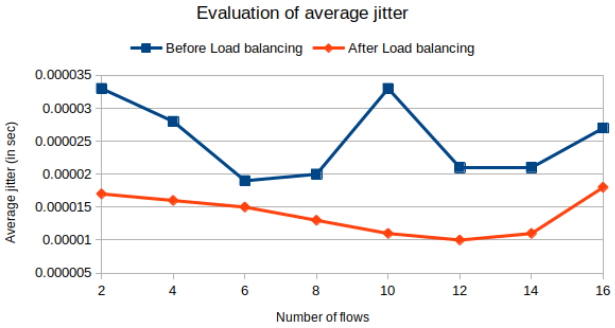

Figure 8: Comparison of jitter after load balancing

The average jitter is reduced from 0.000027 seconds to 0.000019 seconds after application of classification based load balancing for first 16 flows of traffic comprising voice, video and data. The trend shows that jitter will always be low for network with load balancer rather than network without load balancer.

5. Conclusion

The traffic classification algorithms were implemented on two datasets, i.e., generated dataset using D-ITG tool and kaggle dataset. For implementing the model for real-time traffic classification, the model was first trained using a quantitative multi-class dataset. The dataset is first prepared by removing null values and re-sampling the dataset to prevent from over-fitting to a particular traffic type.Then the relevant features were selected as shown in table 2. After selection of features, ML models were trained using the processed dataset and all the models were evaluated on accuracy, precision, recall and f-1 score. While comparing all algorithms, CART decision tree was found to be best algorithm. The execution time of CART algorithm was also found to be minimum. Also, after the application of traffic type based load balancing in network, the bit-rate, packet-rate was highly increased and latency was reduced enough to conclude that load balancing in software defined networking, especially traffic based, improves the QoS of a network. Hence, traffic classification based load balancing with application of CART decision tree for classification helps to improve performance of network.

6. Limitation and Future Work

The scope of this research was that the study was only on the application and evaluation of supervised and unsupervised machine learning algorithms. Also, the load balancing was only performed between the switches using single controller. The tool, D-ITG is only capable of generating few types of data. Further research and implementation can be done by application of semi-supervised and deep learning approach for traffic classification.

Also, load balancing between multiple controllers can be done to further improve QoS of the system. Also, a larger Mininet generated dataset with more traffic types would yield some enlightening results for categorization and traffic-based load balancing.

7. Acknowledgement

This work is carried out under the Department of Electronics and Computer Engineering, IOE, Thapathali Campus. The authors are grateful to the Coordinator of Masters' in Informatic and Intelligent System Engineering, Er. Dinesh Baniya Kshatri for his guidance and support.

References

[1] Muhammad Shafiq, Xiangzhan Yu, Asif Ali Laghari, Lu Yao, Nabin Kumar Karn, and Foudil Abdessamia. Network traffic classification techniques and comparative analysis using machine learning algorithms. In *2016 2nd IEEE International Conference on Computer and Communications (ICCC)*, pages 2451–2455, 2016.

[2] Jinghua Yan and Jing Yuan. A survey of traffic classification in software defined networks. In *2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN)*, pages 200–206, 2018.

[3] Thuy T.T. Nguyen and Grenville Armitage. A survey of techniques for internet traffic classification using machine learning. *IEEE Communications Surveys & Tutorials*, 10(4):56–76, 2008.

[4] Anish Ghosh and Mrs Manoranjitham. A study on load balancing techniques in sdn. *International Journal of Engineering & Technology*, 7:174, 03 2018.

[5] Wen-Hwa Liao, Ssu-Chi Kuai, and Cheng-Hsiu Lu. Dynamic load-balancing mechanism for software-defined networking. *2016 International Conference on Networking and Network Applications (NaNA)*, pages 336–341, 2016.

[6] Hailong Zhang, Xiao Guo, Jinyao Yan, Bo Liu, and Qianjun Shuai. Sdn-based ecmp algorithm for data center networks. In *2014 IEEE Computers, Communications and IT Applications Conference*, pages 13–18, 2014.

[7] Andrew Curtis, Wonho Kim, and Praveen Yalagandula. Mahout: Low-overhead datacenter traffic management using end-host-based elephant detection. pages 1629 – 1637, 05 2011.

[8] Hesham Mekky, Fang Hao, Sarit Mukherjee, Zhi-Li Zhang, and T.V. Lakshman. Application-aware data plane processing in sdn. In *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, HotSDN '14, page 13–18, New York, NY, USA, 2014. Association for Computing Machinery.

[9] Jinghua Yan and Jing Yuan. A survey of traffic classification in software defined networks. In *2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN)*, pages 200–206, 2018.

[10] Pedro Amaral, João Dinis, Paulo Pinto, Luis Bernardo, João Tavares, and Henrique S. Mamede. Machine learning in software defined networks: Data collection and traffic classification. In *2016 IEEE 24th International Conference on Network Protocols (ICNP)*, pages 1–5, 2016.

[11] Stefano Avallone, S. Guadagno, Donato Emma, Antonio Pescapè, and Giorgio Ventre. D-itg distributed internet traffic generator. pages 316–317, 01 2004.

[12] Rogério Leão Santos de Oliveira, Christiane Marie Schweitzer, Ailton Akira Shinoda, and Ligia Rodrigues Prete. Using mininet for emulation and prototyping software-defined networks. In *2014 IEEE Colombian Conference on Communications and Computing (COLCOM)*, pages 1–6, 2014.

[13] Shanu Bhardwaj and S. N. Panda. Performance evaluation using ryu sdn controller in software-defined networking environment. *Wirel. Pers. Commun.*, 122(1):701–723, jan 2022.

[14] Jehad Ali, Seungwoon Lee, and Byeong-hee Roh. Performance analysis of pox and ryu with different sdn topologies. ICISS '18, page 244–249, New York, NY, USA, 2018. Association for Computing Machinery.

[15] Ben Pfaff, Justin Pettit, Teemu Koponen, Ethan J. Jackson, Andy Zhou, Jarno Rajahalme, Jesse Gross, Alex Wang, Jonathan Stringer, Pravin Shelar, Keith Amidon, and Martín Casado. The design and implementation of open vswitch. NSDI'15, page 117–130, USA, 2015. USENIX Association.

[16] Juan Sebastián Rojas, Adrian Pekar, Álvaro Rendón, and Juan Carlos Corrales. Smart user consumption profiling: Incremental learning-based ott service degradation. *IEEE Access*, 8:207426–207442, 2020.