1. Luis Miguel CASTAÑEDA Herrera[1], 2. Wilmar Yesid CAMPO_MUÑOZ[2], 3. Alejandra DUQUE-TORRES[3]

University of Cauca (1), University of Quindío (2), University of Tartu (3)
ORCID: 1. 0000-0002-0500-8090; 2. 0000-0001-8585-706X; 3. 0000-0002-1133-284X

doi:10.15199/48.2022.08.17

Video Streaming Service Identification Using Incremental Learning on Software-Defined Network

***Abstract**. Software-defined networking (SDN) has emerged as a solution to the management challenges facing data networks today, including the identification of different types of services. Therefore, in this paper we present the classification of video streaming in SDN environments. Since, SDN enables the collection and extraction of patterns from traffic flows, through incremental ML algorithms to use classification models that identify video streaming. The results demonstrates that we can identify online video streaming traffic using the Adaptive Random Forest model (ARF).*

Streszczenie.** Sieć definiowana programowo (SDN) pojawiła się jako rozwiązanie problemów związanych z zarządzaniem, z jakimi borykają się współczesne sieci danych, w tym z identyfikacją różnych rodzajów usług. Dlatego w niniejszym artykule przedstawiamy klasyfikację strumieniowania wideo w środowiskach SDN. Ponieważ SDN umożliwia zbieranie i wyodrębnianie wzorców z przepływów ruchu za pomocą przyrostowych algorytmów ML w celu wykorzystania modeli klasyfikacji, które identyfikują strumieniowanie wideo. Wyniki pokazują, że możemy zidentyfikować ruch strumieniowy wideo online za pomocą modelu Adaptive Random Forest (ARF).* (Identyfikacja usługi przesyłania strumieniowego wideo za pomocą przyrostowego uczenia się w sieci definiowanej programowo***)

Keywords: Incremental Learning, SDN, Traffic Classification, Video streaming.
Słowa kluczowe: uczenie przyrostowe, streaming, przesyłanie danych.

Introduction

Due to the constant growth in the consumption of video streaming services, the arrival of 5G, and the future development of 6G, Internet Service Providers (ISP) face a challenge with the management of video traffic in their networks [1], [2], [3], [4], [5]. In addition, with the proliferation of Over-the-top communications (OTT) such as Disney plus, YouTube, and Netflix, the need to identify and distinguish streaming video traffic from other types of traffic is an urgent and necessary task to improve and ensure a good quality of experience to end-users by ISPs. Likewise, it is well known that identifying traffic is one of the first stages for numerous activities that concern network management. Therefore, for ISPs to provide a good quality of experience to their users and optimize their network resources, they must have efficient mechanisms to accurately identify the different types of traffic that navigate the network [6], [7]. In this vein, the Software-Defined Networking (SDN) paradigm has emerged as a solution to the management challenges that networks face today, including identifying different types of traffic. The main idea of SDN is to separate the control plane from the data plane [6], [8]. This separation moves the control logic of the forwarding devices in the data plane to the control plane, specifically on a logically centralized device called the Controller. SDN provides a generalized view of traffic through the control plane, allowing to discover the most efficient route for some traffic automatically; thus, improving overall network performance and optimizing device configuration time because, instead of making individual configurations in network equipment, only the desired configuration is required in the Controller.

In SDN, traffic can be viewed in network flows, allowing for greater granularity of the information navigating the network. A *network flow* can be defined as a set of packets passing an observation point during a specific time interval. Usually, packets that share common properties like source and destination IP addresses, port numbers (source and destination), and the protocol identifier belong to the same flow [9]. In SDN, it is possible to create flow records that retain information of flows in terms of their external or internal characteristics. For instance, an external characteristic is the total flow size, which is the sum of the size of each packet that contains a particular flow, and internal characteristics refer to the minimum, maximum, and average size of each packet belonging to a flow. There is a significant interest in the scientific community and industry in analyzing and exploring such flow records since they can contain trends and patterns in traffic, providing mechanisms to guarantee and maintain optimal network performance.

Regarding the identification of traffic in SDN scenarios, there are different studies where traffic analysis alternatives are presented [8], [10], [11], [12]. Furthermore, some research has taken advantage of stream logs to implement Machine Learning (ML) algorithms for traffic classification [13], [14]. However, these studies deal with cases of traffic in general aspects and do not focus on the study of video streaming services and the impact that this can generate on networks. Regarding the use of ML for the identification of video streaming services, studies such as [15], [16], [17], [18] are based on the characteristics of TCP/IP packets captured on a network interface. However, these investigations are used in simulated environments or traditional networks.

This article focuses on classifying video streaming traffic in SDN environments. We leverage the capabilities of SDN for collecting traffic information at flow level and the capability of ML techniques for discovering patterns in data to classify video streaming traffic. As a contribution We implement a real SDN network, generate different types of traffic, and evaluate different supervised ML approaches. Despite our positive results exploring supervised algorithms for streaming video classification, it is well known that one of the significant drawbacks of supervised approaches is the lack of resilience of the models concerning updates of new traffic patterns where our results show that there is a 50% probability that the supervised model misclassifies VoD flows when another service such as Livestream appears. Therefore, in this paper, we also explore the use of three incremental ML algorithms. Incremental ML approaches fit real-world scenarios in terms of resiliency to changes in network behaviour. This way helps the classifier adapt to new network traffic patterns not seen during the training stage. Our proof-of-concept demonstrates that our approach is applicable and that we can identify and classify online video streaming traffic with 99.8% accuracy using the Adaptive Random Forest model.

The rest of the paper is structured as follows: we start describing the experimental scenario's construction, then present the ML models for online testing. Subsequently, the results are exposed. We include different performance measures for the algorithms used, comparing online classification algorithms, and comparing ourselves with other related research in terms of performance parameters. Finally, the conclusions obtained are presented.

Experimentation scenario

For the development of this research, an SDN network is created where the video service packets are captured. The different flows requested from the internet by end-users pass through an SDN switch where their features are extracted and processed to generate datasets, which will then be used as input data to train ML models. Once these models have been trained, an ML mechanism is used that can work with online flows and classify them as video or non-video, see Fig. 1.

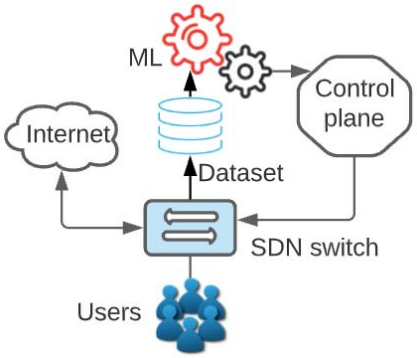


Fig. 1. Implemented scenario for online classification

Firstly, network traffic is captured, filtered, and stored using NFStream which is a framework developed in Python that provides flexible data structures to work with online or offline network data [19]. NFStream takes packets from the different flows navigating the network and extracts their statistical characteristics, which are organized to create two datasets named Stream1 and Stream2. The ML module takes as input the dataset to train and create several ML models, which will be initially evaluated offline comparing them with cross-validation to determine which models will be used in the online classification. The characteristics of the datasets are shown in Table 2 [12].

Once the models have been trained, NFStream starts working in online mode and extracts characteristics of the online traffic flows that come from the observation point; that is, the SDN switch interface that corresponds to a Zodiac FX SDN switch and delivers the information to the ML module. The ML Module performs the classification having the Stream1 dataset as a reference, labeling online flows with the number 1 for Livestream services, 2 for video on demand (VoD), and 3 for non-video flows. While, having the Stream2 Dataset as a reference, the flows of video streaming services are labeled with the number 1 and the non-video flows with the number 2. That is, the ML mechanism compares the flows obtained with each one of the previously trained models and then predicts which type of flow it belongs to. The code used to implement the ML mechanism can be seen in the script presented in Fig. 2 where the attributes used for classifying are those described in Table 2 [12].

ML Models for online testing

For having a point of comparison with the mechanism implemented in this paper, those works that are closest to this research in terms of classification methods were taken into account within the reviewed literature, highlighting their best results or results obtained in conditions similar to this investigation. Also, it was considered that, within the classes to be classified, there is some type of video or HTTP service, using the metrics of accuracy, TPR, and F1 score. These mechanisms and their performance are compared in the results section. For this paper, a Classification and Regression Tree (CART) algorithm was initially used. Then incremental ML algorithms like, Hoeffding Adaptive Tree (HAT); as a variation of decision trees, Naive Bayes in its online version, and Adaptive Random Forest (ARF), for its implementations we use the Scikit multi-flow tool [20]. Because unlike offline mechanisms, an update of the datasets and ML models is necessary since we are working with traffic that changes over time, and the mechanism must be dynamic. For this reason, algorithms designed for incremental learning should be used [21]. In regard to the attributes used for classification, the transmitted bytes variable is crucial to determine whether a flow belongs to a video or not. Since there is a clear difference in the size of the video streaming flow to the rest of the flows, the first ones are always in the order of MB (elephant traffic) while the control streams or other services are usually in the order of KB (mice traffic) [22].

Results

Initially, the CART model was used in the ML module in its online mode given the results shown in [6]. To evaluate its performance, once the mechanism starts working, the flows that come out of the ML module, already classified, are saved in a text file. Then, a sample is extracted every ten seconds until reaching thirty samples. Then, the performance of the mechanism is verified by extracting the true positives (TP), the true negatives (TN), the false positives (FP), and the false negatives (FN) from each sample to obtain the values of accuracy, precision, TPR, and F1 score. In Fig. 3, the data obtained from the 30 samples taken using the CART model in the ML module created with the Stream1 Dataset is presented. Fig. 4 shows the Stream2 Dataset where LS represents Livestream flows, VoD is video-on-demand flows, VD is video streaming flows, and NVD is non-video flows. On the other hand, LS_VoD, VoD_LS, VDF, NVF are the false negatives of each one of the previous flows respectively, which is i) LS_VoD: Livestream flows that were taken as VoD, ii) VoD-LS VoD flows that were taken as LS, iii) VDF video streaming flows that were taken as non-video and iv) NVF non-video flows that were taken as video service. Fig. 5 shows the accuracy values for stream1 and stream2 for each sample.

As can be seen for the two Datasets, in both cases there are average accuracy levels above 90%, obtaining an average accuracy value of 93.75% for stream1 and 96.14% for stream2. It could be thought that the CART model works well in classifying online traffic. However, when reviewing the confusion matrix, it can be concluded that this high accuracy value is because 80% and 74% of the flows for stream 1 and stream 2 respectively are not video streaming flows, see Table 1 and Table 2 where C1 is the Livestream class, C2 is the VoD class and C3 is the non-video class for stream1; and C1 is video streaming flows and C2 is non-video flows for stream2

```
from nfstream import NFPlugin, NFStreamer
from skmultiflow.trees import HoeffdingTreeClassifier    #  incremental Hoeffding Tree model
import pandas as pd
import numpy as np
from sklearn import metrics
import pickle as joblib       #------------libraries

with open('Htree2.pkl','rb') as f:   # The initial model is loaded
    model = joblib.load(f)
class ModelPrediction(NFPlugin):     # Classification class
    def on_init(self, packet, flow):
        flow.udps.model_prediction = 0
    def on_expire(self, flow):
        to_predict = np.array([flow.src2dst_packets,   # Attributes used for traffic classification process
                               flow.dst2src_packets,
                               flow.src2dst_bytes,
                               flow.dst2src_bytes,
                               flow.src2dst_mean_piat_ms,
                               flow.dst2src_mean_piat_ms,
                               flow.src2dst_mean_ps,
                               flow.dst2src_mean_ps,
                               flow.src_port,
                               flow.dst_port,
                               flow.protocol,
                               flow.src2dst_stddev_piat_ms,
                               flow.dst2src_stddev_piat_ms,
                               flow.src2dst_stddev_ps,
                               flow.dst2src_stddev_ps]).reshape((1,-1))#The model receives the attributes and returns the
        flow.udps.model_prediction = self.my_model.predict(to_predict) #classification belonging to each flow or stream

ml_streamer = NFStreamer(source="enp2s0", statistical_analysis=True, active_timeout=10, performance_report=1,
udps=ModelPrediction(my_model=model)) #NFStream function in which the capture point is determined, also the capture time per stream and
returns the streams with a new classification attribute
for flow in ml_streamer:
    print(flow) # Print each stream with its attributes and classification
```

Fig. 2. Mechanism script and its description

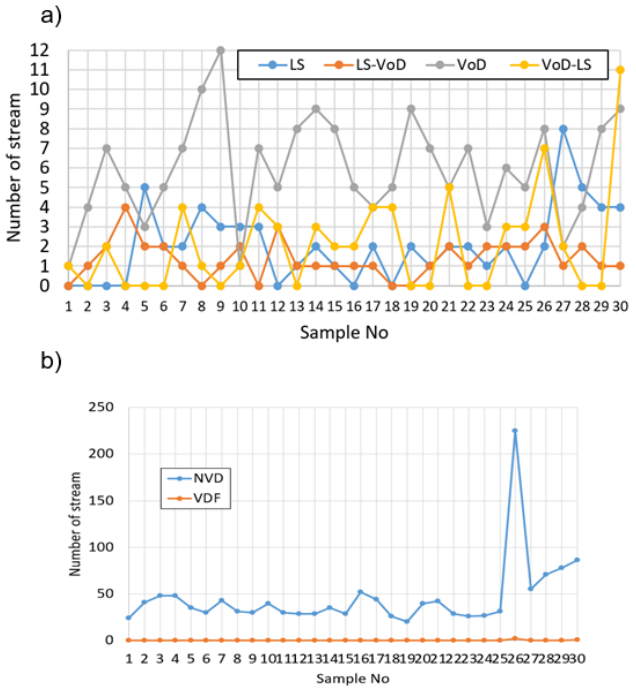


Fig. 3. Classification results for CART- Stream1 model. a) Stream1 LS, LS-VoD, VoD and VoD-LS. b) Stream1 NVD and NVF

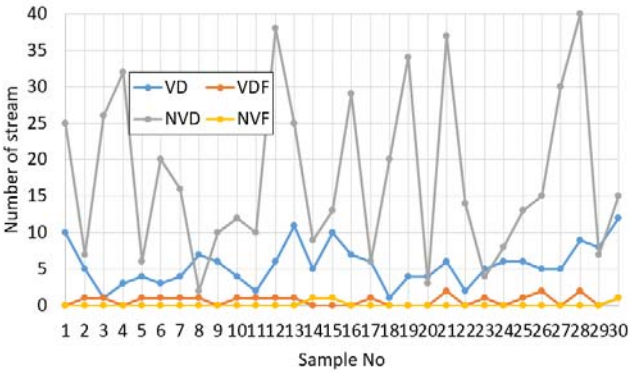


Fig. 4. Classification results for CART- Stream2 model a)

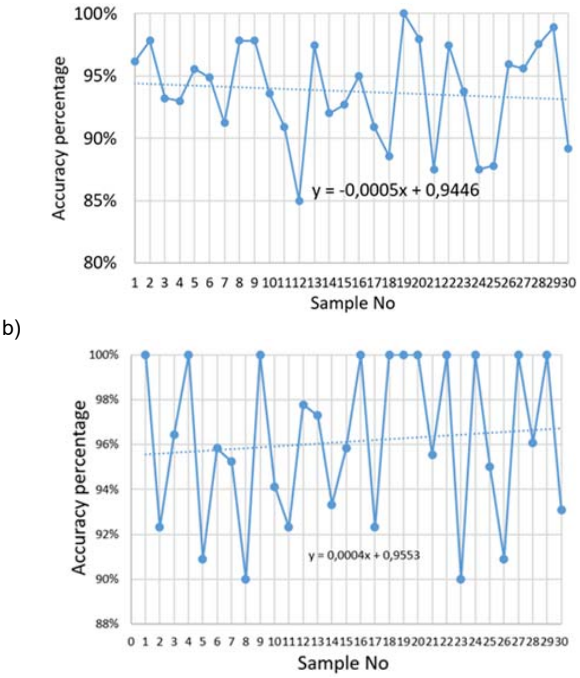


Fig. 5. CART model Accuracy a) Using Stream1. b) Using Stream2

If Table 1 is observed, the model, although it has no problem distinguishing a video streaming flow from a non-video flow, has shortcomings in differentiating between the two video services with 61 Livestream flows as true positives and 62 VoD flows classified as Livestream being false positives, resulting in a precision of 49%, see Table 2. That is, there is a 51% probability that the model is wrong in these two classes. Furthermore, when the model identifies a true positive with a TPR of 51% for C1, its value has a very low accuracy for Livestream services. Therefore, although the CART model gives good results in offline classification [7], its performance drops considerably in online classification with an F1 score of 54%, 75%, and 99% for classes C1, C2, and C3. respectively in Stream1, see Table 3. And although it improves notably in Stream2 with an F1 of 94%, see Table 4, this is because the video streaming flows over long periods are still considerably larger than the flows of no video services, making the model have a good performance. However, the idea of having a

model that allows classifying different types of video streaming services that can vary their characteristics over time, the use of incremental learning algorithms such as HATC, NB (Online), and ARF were chosen for the implementation of the ML module in its online mode.

Table 1: Confusion matrix fot CART-Stream1 model

Classes (Cs)	C1	C2	Cs	C2	C3	Cs	C1	C3
C2	61	41	C2	179	9	C1	61	0
C3	62	179	C3	4	1364	C3	0	1364

Table 2. Confusion matrix for CART-Stream2 model

Classes	C1	C2
C1	167	12
C2	7	526

Table 3. Performance scores for CART-stream1 model

Classes	C1	C2	C3
Precision	49%	79%	99%
TPR	59%	71%	99%
F1-Score	54%	75%	99%

Table 4. Performance scores for CART-stream2 model

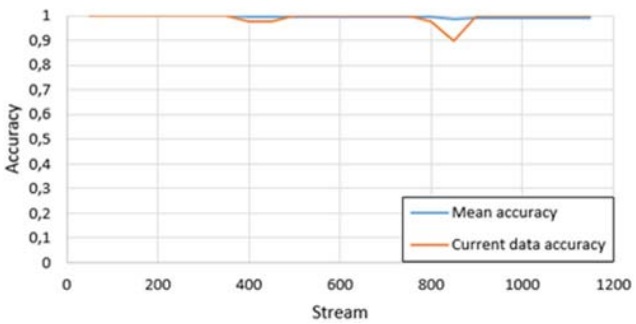
Classes	C1	C2
Precision	95%	97%
TPR	93%	98%
F1-Score	94%	98%

Online algorithm comparison

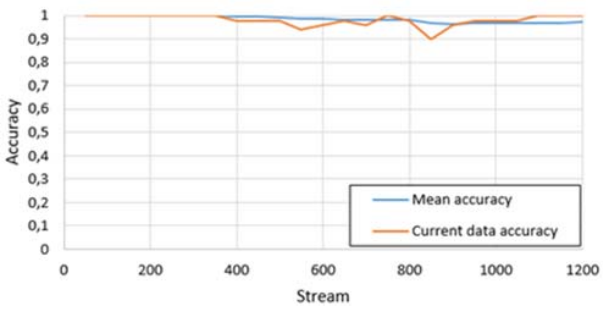
This section presents the results obtained with the ML module in its online mode. The HAT, NB (online), and ARF models were applied, using the Scikit multi-flow tool, which allows recreating an incremental learning process. A live stream broadcast is simulated, analysing 1200 flows for Stream1 and 1600 flows for Stream2. The results can be seen in Fig. 6 and Fig. 7 for Stream1 and Stream2 respectively.

Using Stream1 as the dataset, the accuracy results for the ARF, HAT and NB online models are 99.6%, 98.5%, and 97.9% respectively, see Fig. 6. Using Stream2 as the dataset, the accuracy results for the same models are 99.8%, 99.2%, and 99.3% see Fig. 7. However, the model that presents the best behavior in all cases is ARF. The accuracy metric is not enough to observe its performance; therefore, this model is used in the implementation of the ML module for validating it, but now with live and real-time traffic. The procedure is the same as the one described for CART. However, unlike the tests carried out with CART, Stream2 is used as the basis for ARF to create the initial model. This is due to the problems presented in the classification between Livestream and VoD found in the algorithms derived from the trees. Thus, it was determined that in the final implementation of the ML module, only two classes will be classified, video streaming (VD), non-video services (VDF), and their corresponding false negatives (NVD and NVF) see Fig. 8. Fig. 9 shows the accuracy values for each of the samples. Likewise, the results of the classification are presented in a confusion matrix and the performance parameters in Table 5 and Table 6.

a)



b)



c)

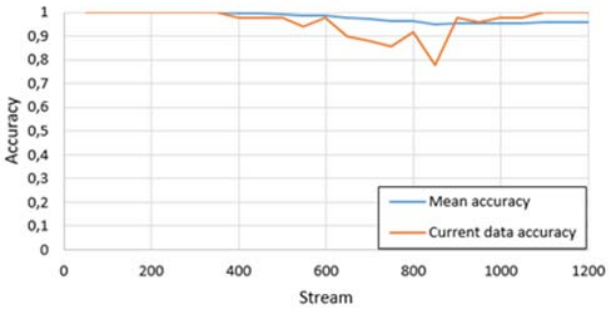
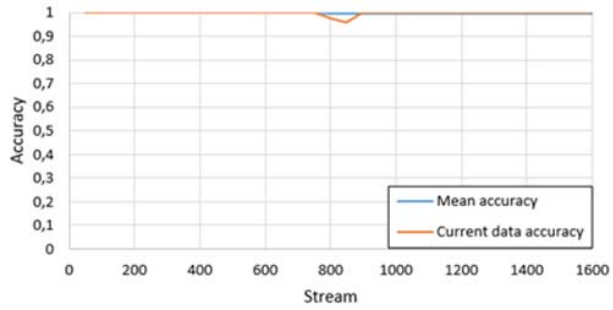
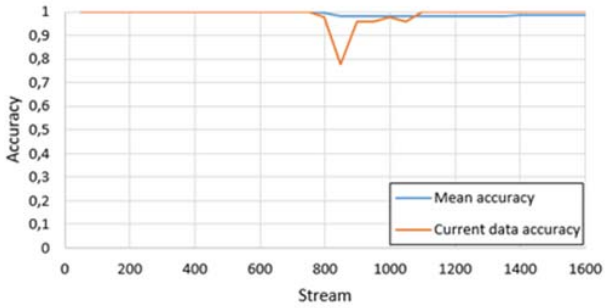


Fig. 6. Accuracy comparison of the online models using Stream1. a) Accuracy of ARF-Stream1. b) Accuracy of HAT-Stream1. c) Accuracy of NB- Stream1 Online

a)



b)



c)

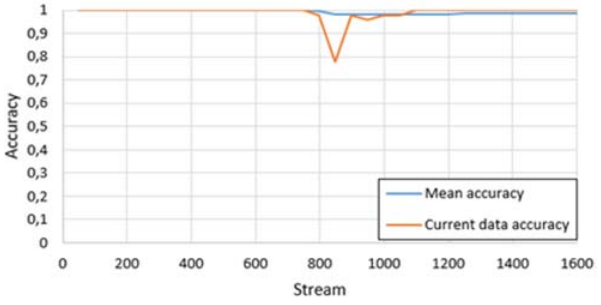


Fig. 7. Accuracy comparison of the online models using Stream2. a) Accuracy of ARF-Stream2. b) Accuracy of HAT-Stream2. c) Accuracy of NB- Stream2 Online

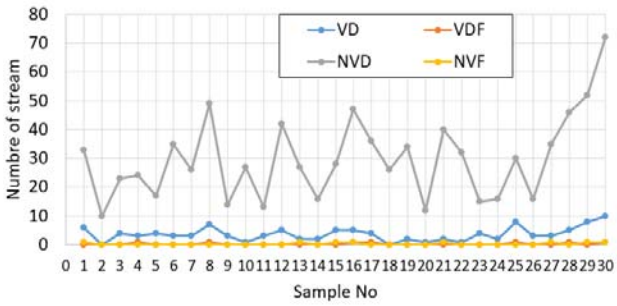


Fig. 8. Classification results for ARF-Stream2

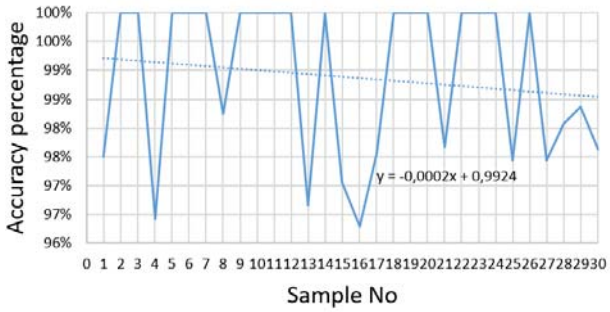


Fig. 9. Accuracy for ARF model

Table 5. ARF stream2 confusion matrix

	Prediction	
Stream2	Video	Non-video
Video	109	7
Non-video	8	893

Table 6. ARF stream2 Performance scores

Classes	C1	C2
Precision	93%	99%
TPR	94%	99%
F1-Score	94%	99%

In Table 5, it is determined that there is a better performance of ARF concerning CART in an online environment. Although, the overall performance for the video streaming class holds up with an F1 score of 94% just like CART with Stream2. The performance of ARF in the non-video class is higher with a value of 99% and has an average of 99.8% accuracy. This is because ARF is an adaptive algorithm and can continue learning from changes that may occur in the characteristics of flows over time, making it better than a static algorithm like CART. In addition, the results are compared with the most similar studies to this paper, see table 7, showing that the implemented mechanism presented a better performance with an accuracy of 99.8% and an F1 score of 94%. However, the TPR presented a lower value than those obtained in the other studies with a value of 94%. This result can be explained because the ML mechanism was implemented in a scenario with a real SDN switch classifying real live traffic, making the classification task more complex. However, the results are still good and show the feasibility of the proposed classification mechanism.

Table 7. Comparison of Classification Mechanisms

Mechanism	Classification type	ML algorithm	Accuracy	TPR	F1_score
Research on network traffic identification based on machine learning and deep packet inspection [13]	DPI y flows characteristics	NB	96.8%		
A New Semi-supervised Method for Network Traffic Classification Based on X-means Clustering and Label Propagation [14]	By flows characteristics	NB, j48	95%		
An innovative approach for real-time network traffic classification [15]	By flows characteristics	NB	98.8%	97%	
Supervised Machine Learning-based Classification of Video Traffic Types [16]	By flows characteristics	kNN, Tree, SVM, NB	97.7%		
Network Traffic Classification using Machine Learning Techniques over SDN [17]	By flows characteristics	NB	98%	96%	
An Intelligent Traffic Classification in SDN-IoT: A Machine Learning Approach [18]	By flows characteristics	Random Forest	88.3%		91.4%
This research	By flows characteristics	Adaptative Random Forest	99,8%	94%	94%

Conclusions

Considering the current trends in networks with the arrival of 5G and the future development of 6G, the traffic generated by video streaming services will continue to grow. For this reason, this paper proposes a video streaming identification approach based on incremental algorithms and network flow analysis, which allows the correct classification of video traffic in real-time. In this way, ISPs and content distributors can make better decisions when managing video traffic. In addition, this approach seeks to reduce the controller's workload by freeing it from classification tasks and thus improve network performance. For this reason, this paper evaluated several supervised ML models, resulting in better performance for incremental models in the ARM classifier.

The performance metrics of the models were derived from the confusion matrix where the results show that CART can differentiate between video streaming and non-video flows without any problem, but its performance begins to decrease when another service such as Livestream appears. This decrease in performance is accentuated when using the model with online traffic where an accuracy of 93.75% is presented using Stream1, but with a precision of 49% and a TPR of 59% for VoD, meaning that there is a 50% probability that the model misclassifies VoD flows. For this reason, in the final implementation of the ML module with online traffic, the use of incremental learning algorithms was chosen based on the information provided by Stream2 since it is complex to classify between video streaming services due to their similarity in the characteristics of its streams and the variability of the data that can arise in live traffic.

The results obtained by the ML model in its final implementation show that the model with the best performance is the adaptive random forest (ARF), with an average accuracy of 98.88%, a TPR of 94% for the video class, and a TPR of 99% for the non-video class. In addition, the model has high sensitivity levels, which means that the model correctly discriminates the positives from the negatives from the classified instances. Regarding the F1 score, the general performance of the model indicates that video streaming services are differentiated from non-video flows, yielding better results in accuracy than the most similar studies to this paper. The TPR presented a value lower than those obtained in the other research with a value of 94%. The ML module was implemented in a real SDN switch device that classified real live traffic, making the classification task more complex. However, the results remain optimal, verifying the usefulness and viability of the mechanism developed in this paper for the classification of video streaming services using incremental algorithms.

***Authors**: MS.c Luis Miguel Castañeda Herrera. E-Mail: luiscastaneda@unicauca.edu.co Faculty of Electronic Engineering and Telecommunications, PhD (c) program in Telematics Engineering. University of Cauca, Calle 5 Nº 4-70, Popayan, Cauca, Colombia.*
Ph.D Wilmar Yesid Campo Muñoz. E-Mail: wycampo@uniquindio.edu.co Prof. Faculty of Engineering, Electronic Engineering Program. University of Quindío, Carrera 15 con calle 12 norte, Armenia, Quindío, Colombia.
Alejandra Duque-Torres. E-Mail: duquet@ut.ee Junior Research Fellow and Ph.D Stutendt, Institute of Computer Science. University of Tartu, Narva mnt 18, Tartu, Estonia

REFERENCES

[1] Castaneda Herrera, L. M., Duque Torres, A., Campo Munoz, W. Y., An Approach Based on Knowledge-Defined Networking for Identifying Video Streaming Flows in 5G Networks, *IEEE LAT AM T.,* 19 (2021), No.10, 1737-1744.
[2] Alhassan-Gedel, I., Nwulu, N I., A techno-economic comparison of 5G centralized wireless network architectures, *Przegląd Elektrotechniczny.*, (2021), No. 5, 12–19.
[3] Araujo, H., de Freitas, A., Prata, D., Casella, I., Capovilla, C., A Multiband Antenna Design Comprising the Future 5G Mobile Technology, *Przegląd Elektrotechniczny.*, (2019), No.2, 108–111.
[4] Kamil, N., Al-Nakkash, A., Wadday, A., Khaleel, A-D., Fractal Vicsek MIMO Antenna for LTE and 5G Applications, *Przegląd Elektrotechniczny*, (2021), No.10, 53–57.
[5] Jansri, S. C., Phongcharoenpanich, C., Design of a Compact Wideband Circular Monopole Antenna for 5G Applications, *Przegląd Elektrotechniczny.*, (2021), No.4, 9–12.
[6] Bilski, T., New Challenges in Network Security, *Przegląd Elektrotechniczny.*, (2016), No.12, 228–232.
[7] Bagci, K. T., Sahin, K. E., Tekalp, A. M., Compete or Collaborate: Architectures for Collaborative DASH Video over Future Networks, *IEEE Trans. Multimed*., 19 (2017), No.10, 2152–2165.
[8] Estrada-Solano, F., Ordonez, A., Granville, L. Z., Caicedo Rendon, O. M., A framework for SDN integrated management based on a CIM model and a vertical management plane, *Comput. Commun*., 102 (2017), 150–164.
[9] Pekar, A., Duque-Torres, A., Seah, W.K.G. et al., Knowledge Discovery: Can It Shed New Light on Threshold Definition for Heavy-Hitter Detection?., *J Netw Syst Manage.,* 29 (2021), No 3, https://doi.org/10.1007/s10922-021-09593-w.
[10] Bakhshi, T., Multi-feature Enterprise Traffic Characterization in OpenFlow-based Software Defined Networks, *Int. Conf. Front. Inf. Technol. FIT 2017.*, (2017), 23–28.
[11] Li,Y., Li, J., MultiClassifier: A combination of DPI and ML for application-layer classification in SDN, *2014 2nd Int. Conf. Syst. Informatics ICSAI.,* (2014), 682–686, Jan. 2015.
[12] Castaneda Herrera, L. M., Duque Torres, A., Campo Munoz, W. Y., Video Streaming Service Identification on Software-Defined Networking, *Int. J. Comput. Commun. Control.*, 16 (2021), No. 5, 1-13.
[13] Yang, B., Liu, D., Research on network traffic identification based on machine learning and deep packet inspection, *Proc. 2019 IEEE 3rd Inf. Technol. Networking, Electron. Autom. Control Conf. ITNEC*., (2019), 1887–1891.
[14] Noorbehbahani, F., Mansoori, S., A new semi-supervised method for network traffic classification based on X-means clustering and label propagation, *2018 8th Int. Conf. Comput. Knowl. Eng. ICCKE.,* (2018), 120–125.
[15] Dias, K. L., Pongelupe, M. A., Caminhas, W. M., de Errico, L., An innovative approach for real-time network traffic classification, *Comput. Networks*, 158 (2019), 143–157.
[16] Grabs, E., Petersons, E., Ipatovs, A., Chulkovs, D., Supervised Machine Learning based Classification of Video Traffic Types, *Proc. 2020 24th Int. Conf. Electron. Electron.,* (2020).
[17] Parsaei, M. R., Sobouti, M. J., khayami, S. R., Javidan, R., Network Traffic Classification using Machine Learning Techniques over Software Defined Networks, *Int. J. Adv. Comput. Sci. Appl*., 8 (2017), No. 7.
[18] Owusu, A. I., Nayak, A., An Intelligent Traffic Classification in SDN-IoT: A Machine Learning Approach, *2020 IEEE Int. Black Sea Conf. Commun. Networking, BlackSeaCom.,* (2020).
[19] Aouini, Z., Pekar, A., NFStream: A flexible network data analysis framework, *Comput. Networks*., 204 (2022), p. 108719, https://doi.org/10.1016/j.comnet.2021.108719.
[20] Montiel, J., Read, J., Bifet, A., Abdessalem, T., Scikit-multiflow: A Multi-output Streaming Framework*, J. Mach. Learn. Res*., 19 (2018).
[21] Estrada-Solano, F., Caicedo, O. M., Da Fonseca, N. L. S., NELLY: Flow Detection Using Incremental Learning at the Server Side of SDN-Based Data Centers, *IEEE Trans. Ind. Informatics*., 16 (2020), No. 2, 1362–1372.
[22] Wang, W., Sun, Y., Salamatian, K., Li, Z., Adaptive Path Isolation for Elephant and Mice Flows by Exploiting Path Diversity in Datacenters, *IEEE Trans. Netw. Serv. Manag*., 13 (2016), No. 1, 5–18.