INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
Online ISSN 1841-9844, ISSN-L 1841-9836, Volume: 16, Issue: 5, Month: October, Year: 2021
Article Number: 4258, https://doi.org/10.15837/ijccc.2021.5.4258

Video Streaming Service Identification on Software-Defined Networking

L. M. Castañeda Herrera, W. Y. Campo Muñoz, A. Duque-Torres

Luis Miguel Castañeda Herrera*, Wilmar Yesid Campo Muñoz
Department of Electronic Engineering
University of Quindío, Colombia
Cra 15 #12N, Armenia, Quindío, Colombia
*Corresponding author: lmcastanedah@uqvirtual.edu.co
wycampo@uniquindio.edu.co

Alejandra Duque-Torres
Institute of Computer Science
University of Tartu, Estonia
Narva maantee 18, 51009 Tartu, Estonia
duquet@ut.ee

Abstract

It is well known that video streaming is the major network traffic today. Furthermore, the traffic generated by video streaming is expected to increase exponentially. On the other hand, Software-Defined Networking (SDN) has been considered a viable solution to cope with the complexity and increasing network traffic due to its centralised control and programmability features. These features, however, do not guarantee that network performance will not suffer as traffic grows. As result, understanding video traffic and optimising video traffic can aid in control various aspects of network performance, such as bandwidth utilisation, dynamic routing, and Quality of Service (QoS). This paper presents an approach to identify video streaming traffic in SDN and investigates the feasibility of using Knowledge-Defined Networking (KDN) in traffic classification. KDN is a networking paradigm in SDN, that utilises Artificial Intelligence (AI) using Machine Learning methods, allowing for integrating behavioural models to recognise patterns in SDN traffic, such as video streaming traffic detection. We extract important network traffic information in the form of flows statistics in our initial proof-of-concept. Then, we used such information to train six ML models that can classify network traffic into three types, Video on Demand (VoD), Livestream, and no-video traffic. Our proof-of-concept demonstrates that our approach is applicable and that we can identify and classify video streaming traffic with 97.5% accuracy using the Decision Tree model.

Keywords: livestream, KDN, SDN, Traffic Classification, VoD.

1 Introduction

It is well known that video streaming is the main network traffic today. According to the latest Cisco Visual Networking Index (VNI) forecast, in 2021, Internet video users will be nearly 1.9 billion [7], *i.e.,* the world will see 3 trillion minutes of Internet video per month. Consequently, the video will continue to dominate global Internet traffic, accounting for 80% of all Internet traffic in 2021 [30], [14]. On the other hand, Software-Defined Networks (SDN) has been considered a viable solution to cope with complexity and increase network traffic. SDN enables network programmability and provides a flexible architecture to manage more efficient computer networks [25].

The Control Plane (CP) and Data Plane (DP) are separated in SDN, allowing for logically centralised network device control. The network devices become simple forwarders that are programmable using a standardised protocol such as OpenFlow by transferring the forwarding devices' control logic to a logically centralised device, namely the Controller [33]. The centralised view of the network and its traffic flows is one of SDN's key advantages. On the other hand, those advantages do not ensure that network performance will not suffer as traffic volume grows. Video streaming services, in particular, have raised their quality of service (QoS) standards [5]. As a result, even with SDN benefits, controlling the traffic that flows across the network remains a big challenge [16].

Classification of traffic flows, which provides inputs for different network-related management tasks, is an important tool to ensure the reliable operation of networks [5], [16]. In particular, the classification of flows that belong to video services has been the object of study both in academia and in industry, as they dominate global Internet traffic and account for 80% of all Internet traffic [30], [14]. Most approaches to video traffic classification are based on port number, Deep Packet Inspection (DPI), and flow characteristics [2, 4, 10, 15, 17]. Port-based strategies are not as reliable as several applications or services use random or non-standard port numbers [10], [4]. DPI approaches are accurate in identifying traffic as they inspect the packet's payload [15]. Unfortunately, these approaches have a high processing cost and are generally not used because the packet is encrypted [4].

On the other hand, methods that use flow characteristics (*e.g.*, flow size, number of packets, flow duration, etc.) to perform flow classification have attracted attention in academia and industry. These methods take different statistical characteristics of the flows during the establishment, maintenance and release of a session [17], [2]. Then, by using data mining and analysis methods, it is possible to extract patterns that can distinguish the nature of the network flow and identify the type of traffic to which it belongs.

In this paper, we introduce a technique for identifying video streaming traffic in SDN and examine the possibility of utilising Knowledge-Defined Networking (KDN) for traffic classification. KDN is a networking concept that uses Machine Learning (ML) to improve a variety of network management activities and services [8], [3]. In our initial proof-of-concept, we derive the relevant information of network traffic in the form of flow statistics such as total flow size, the total number of packets, flow duration, etc. Then, we used such information to train six ML models that can classify network traffic into three types, Video on Demand (VoD), Livestream, and Non-video traffic.

Our approach follows some steps of the Cross-Industry Standard Process for Data Mining (CRISP-DM) methodology [18], [6]: *i)* creating a dataset from an SDN-based network, *ii)* cleaning and selecting features for the generated flows dataset, and *iii)* creating a data model. The application of various ML techniques to identify video streaming flows is referred to as data model.

To sum up, the contributions of this paper are: *i)* two dataset to model VoD, Livestream, and Non-video traffic in SDN environment, *ii)* a model that allows classifying VoD, Livestream traffic from Non-video traffic; *iii)* the evaluation of different classification techniques disclosing that Decision Tree (DT) model is the algorithm that most accurately classifies the targeted traffic. Our proof-of-concept demonstrates that our approach is applicable and that we can identify and classify video streaming traffic with 97.5% accuracy.

The remainder of this paper is organized as follows. We provided a detailed description of material for the SDN network implementation and the data mining method used in Section 2. In Section 3, we present the answers to our research questions and discuss the results obtained. Finally, we conclude the paper in Section 4.

2 Material and method

This Section introduces our approach materials and method to classify video streaming flows in an SDN environment. We start with the definition of KDN (Section 2.1), CRISP-DM methodology (Section 2.2), the data understanding (Section 2.3), the data preparation (Section 2.4), the construction of the models (Section 2.5), and the performance metrics used to evaluate our approach (Section 2.6)

2.1 Knowledge-Defined Networking (KDN)

Clark and Col [8] introduced the idea of an smart plane, known as Knowledge Plane (PN), to the conventional computer network architecture shaped by the control plane (CP) and the data plane (DP) [20]. The KP' key is to introduce artificial intelligence to offer the network the ability to discover, generalize or learn from past events [3]. To earn these skills, KP proposes the use of ML techniques. The main purpose of KP is to provide automation processes, recommendation systems and data prediction [22].

The addition of KP to the SDN architecture is called Knowledge-Defined Networking, KDN [22]. Data Plane (DP), Control Plane (CP), Management Plane (MP), and KP are the four planes that make up KDN. Through network device forwarding, DP is responsible for creating metadata. The CP interfaces receive the KP instructions, which are subsequently transmitted to the forwarding devices by the Controller. In addition, CP provides metadata about the network's state to the MP. Data and Control Plane metadata are gathered and kept in the Management Plane. The MP provides the KP with basic statistics data per-flow and per forwarding device. Finally, KP provides a set of instructions to the Controller regarding what the network should perform [19].

2.2 Cross-Industry Standard-Process for Data Mining (CRISP-DM)

We followed the CRISP-DM methodology to build a classification model to classify video streaming traffic into three categories VoD, Livestream, and no-video traffic. In particular, we perform three CRISP-DM phases: Data Understanding, Data Preparation, and Modelling. The traffic-related dataset is generated, collected, and defined in the data understanding phase. In particular, we generate, collect, and define a dataset of flows in the form of statistic information. Data preparation seeks to clean the data (removing null values or damaged data) and choose the relevant attributes to decrease computational cost and improve flow classification accuracy. Modelling is the process of classifying flows using various ML approaches. We examine several supervised learning algorithms in this stage to find the one that best identifies flows into VoD, Livestream, and no-video categories in an SDN scenario.

2.3 Data Understanding

This phase creates an initial dataset that represents flow/packet features in an SDN scenario. The following activities are carried out in order to create this dataset: i) created a controlled experimental scenario to re-create an SDN scenario, ii) deployed the controlled scenario, and iii) generated network traffic.

The scenario proposed is shown in Figure 1. We followed the traditional SDN architecture, *i.e.*, SDN Controller, n switches, and n host. The DP network devices can receive and send video and non-video traffic from CP. We run Opendaylight Controller [24] in a Virtual Machine (VM) with Linux Ubuntu 18.04. To get the network status and information about network traffic, *i.e.*, flows statistics, we developed an ofctl-rest-API. This API is located in the AP and the DP contains a tree topology network.

We used a Zodiac FX switch [37] and Ubuntu Server 18.04 VM. Zodiac FX is a 4-port 10/100M Ethernet switch driven by the need for a low-cost option for SDN experimentation and academic propose. Regarding traffic generation, we used two common severs, Wowza and VLC. Wowza server [36] generates the VoD traffic and VLC server [34] the Livestream traffic. To capture traffic generated, we used the Wireshark tool [35]. Wireshark is a well-known network analysis tool that captures packets

in real-time. We decided to use it since Wireshark is the most often-used packet sniffer in the world [35].

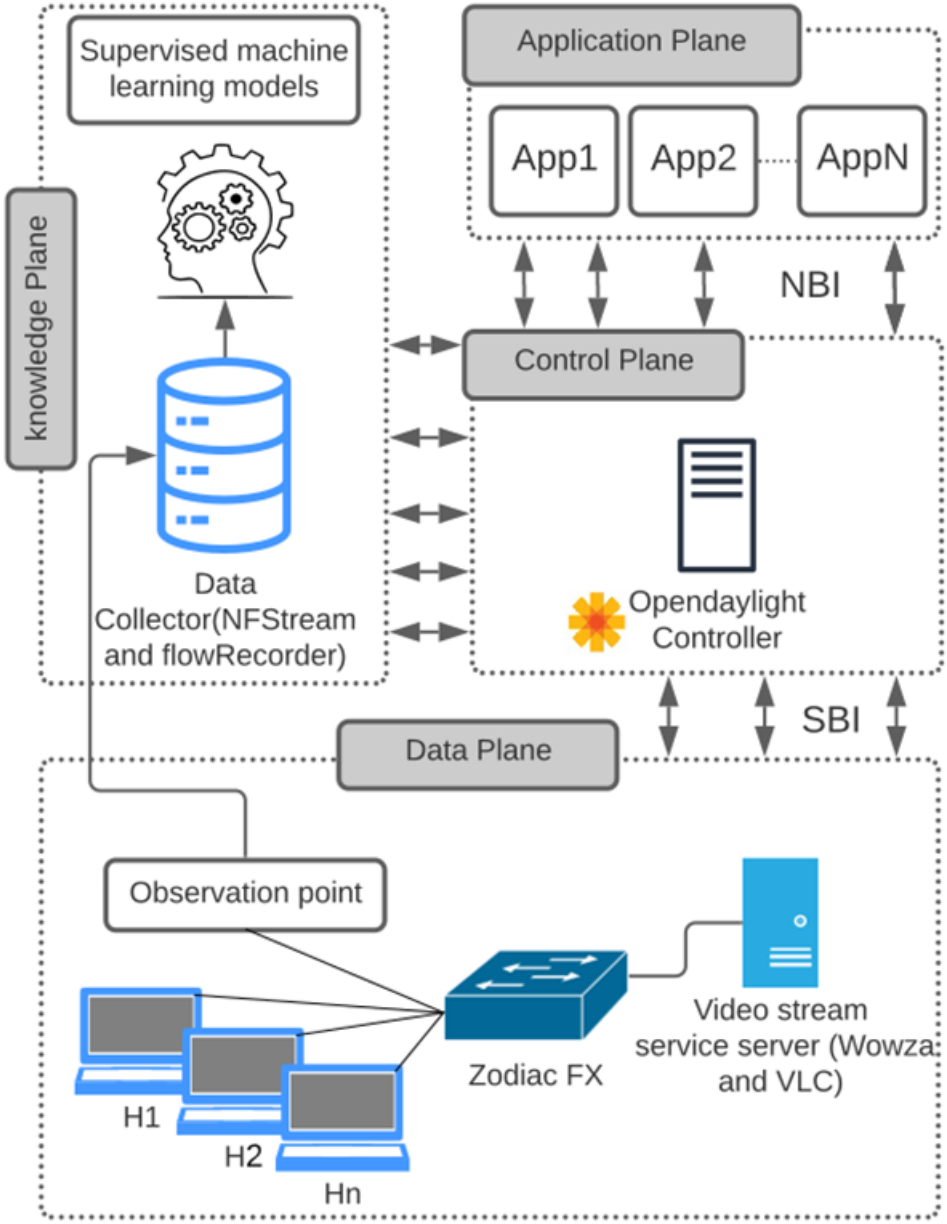

Figure 1: Test environment for dataset construction.

2.4 Data Preparation

The final flow dataset is built in this phase. The following activities were are carried out in order to get the final dataset: i) extract the PCAP (Packet CAPture) traces' packets and organise them into flows, ii) cleaned and examined the data, *i.e.*, add or remove instances; and iii) feature engineering, *i.e.*, detects and discards superfluous and unnecessary features. We execute the aforementioned tasks with Scikit-learn [31], a Python data mining package that includes a comprehensive set of ML algorithms and data preparation processes.

2.5 Data Modelling

This section covers the selection, evaluation, and modeling of ML models for use in constructing the classification model. We selected six machine learning models, which are detailed below:

- Logistic Regression (LR): LR is a supervised learning classification algorithm used to predict a target variable's probability. It can be used for various classification problems. LR is widely used in different fields because it is one of the simplest ML algorithms [38].
- Linear Discriminant Analysis (LDA): Belongs to the supervised learning algorithms. It's used to find a linear combination of features that separates two or more classes of data. The succeeding combination can be used as a linear classifier [32].
- K-Nearest Neighbours (KNN): Is widely used in ML for Classification and regression problems. KNN algorithms use data points and classify new data points based on similarity measures, *e.g.*, distance function. KNN belongs to supervised learning techniques [21].
- Decision tree classifiers (CART): Is a non-parametric approach that can be used for classification and regression. A model is represented over the entire input space and trained with full training data from its parameters in parametric estimation. It is then easy to use the same template and parameters for any test input [21].
- Support Vector Machine (SVM) or Support Vector Network (SVN): Is an ML algorithm for classification and regression tasks. SVM is a supervised learning method that looks at data and sorts it into one of two categories. An SVM outputs a map of the sorted data with the margins between the two as far apart as possible. SVMs are used in text categorisation, image classification, handwriting recognition and in the sciences [11].
- Naive Bayes (NB): NB is a classification technique based on Bayes' theorem assuming independence between the predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature [29], [9].

2.6 Performance Metrics

In our research, we employed performance metrics generated from the Confusion Matrix (See Table 1). Let B and B' denote a classification (*e.g.*, a Livestream flow). Then each reference standard measure is expressed as a function of the Confusion Matrix defined as follows:

True Positive (TP): If the current class of a case was B and the predicted class was B. This represents a successful prediction.

True Negative (TN): If the current class of a case was B' and the predicted class was B'. This represents a successful prediction.

False Positive (FP): If the current class of a case was B' and the predicted class was B. This represents a wrong prediction.

False Negative (FN): If the current class of a case was B and the predicted class was B'. This represents a wrong prediction.

Table 1: Confusion matrix

B / B'	C_n	$no - C_n$
C_n	TP	FP
$no - C_n$	FN	TN

The ratio of correct predictions made to both groups is known as accuracy, and it is represented as:

$$Accuracy = \frac{TN + TP}{FN + TN + TP + FP} \tag{1}$$

The ratio of correct predictions for class A is known as precision (or positive predictive value):

$$Precision = \frac{TP}{FP + TP} \tag{2}$$

The percentage of successful predictions made to class A instances is known as the True Positive Rate (or recall).

$$TPR = \frac{TP}{TP + FN} \tag{3}$$

The f-measure statistic (or F1 score) considers both the TPR and precision of a classifier to measure its quality:

$$f1 - score = \frac{2 * TPR * Precision}{TPR + Precision} \tag{4}$$

3 Result and discussion

This section presents the results and their analysis. We start with the data understanding (Section 3.1) and the data preparation (Section 3.2). Finally, the results of the performance metrics of the models created and the final discussion (Section 3.3).

3.1 Data understanding

The traffic captured by Wireshark was processed and organized into flow records by using the flowRecorder tool [27]. flowRecorder is a tool written in Python, that allows to turn IP packets into flow records stored in a CSV (Comma-Separated Values) file. Using packets either in the form of PCAP (Packet CAPture) files or sniffed live from a network interface. This tool supports the measurement of flow features in both unidirectional and bidirectional modes. Table 2 describes the flow features used.

The main idea of flowRecoder is to organize packets into flow by using the top 5-tuple packet header. Depending on the observed (incoming) packets' properties, either new flow records is created, or the existing ones' flow features are updated. For more information about flowRecoder we refer the reader to [12, 13, 26, 28]. Two datasets named "Datavideo1" and "Datavideo2" were generated using flowRecorder. These data sets were validated with another network tool called NFStream [23].

3.2 Data preparation

NFStream performs the same tasks as flowRecorder. Unlike flowRecorder, NFStream uses nDPI (open source library for deep packet inspection) to extract information from layer 7. Deep packet inspection allows verifying which application it belongs to, *i.e.*, it will enable identifying if the packet belongs to a Livestream, VoD or no video stream. We used NFStream to label the Datavideo1 and Datavideo2 datasets. Datavideo1 consists of 1600 instances. Each instance represents a flow. Thus, Datavideo1 has 1600 flows, 800 are VoD services, and 800 are non-video flows. Datavideo2 comprises 1200 flows; 400 are live stream video streams, 400 are VoD flows, and 400 are non-video flows.

We analyzed the data distribution of each feature to find bias in the data. Figure 2 illustrates an example of data distribution performed. It turned out that some features of Datavideo1 are not uniformly distributed. Several narrow peaks are in some features, *e.g.*, "f-pktTotalCount", "f-octetTotalCount", "f-avg-piat". This behaviour also happens in Datavideo2. We believe that the non-video flows to introduce the not uniform distribution since they only contain 1 or 2 packets. In Figure 2, we only show de forward flows since backward flows present the same behaviour.

As Figure 2 shows, the features "b-avg-ps", "src-port", and "f-std-dev-ps" have bimodal distributions, which means that those features tend to two values. We used a Yeo-Johnson transformation to obtain better performance from the models created for the classification traffic. Yeo-Jonson is a technique used to stabilize variance, make the data more normal distribution-like [1].

We performed an analysis to determine which set of the feature are imports and which not, *i.e.*, which variables provide the most information for classifying flows. We use the correlation matrix

Table 2: Dataset feature description

Feature name	**Description**
f-pkt_Total_Count	Nº of transmitted packets (Forward)
b-pkt_Total_Count	Nº of transmitted packets (Backward)
f-octet_Total_Count	Nº of transmitted packets (Backward)
b-octet_Total_Count	Nº of transmitted bytes (Backward)
f_avg_piat	Average packet inter arrival time (Forward)
b_avg_piat	Average packet inter arrival time (Backward)
f_avg_ps	Average packet size (Forward)
b_avg_ps	Average packet size (Backward)
src_port	Source port
dst_port	Destination port
protoc	Identifier of protocol. In particular, TCP/UDP
f-std-dev-piat	Stand. dev. packet inter-arrival times (Forward)
b-std-dev-piat	Stand. dev. of packet inter-arrival times (Backward)
f-std-dev-ps	Stand. dev. of packet sizes (Forward)
b-std-dev-ps	Stand. dev. of packet sizes (Backward)
Class	Classification of the sample

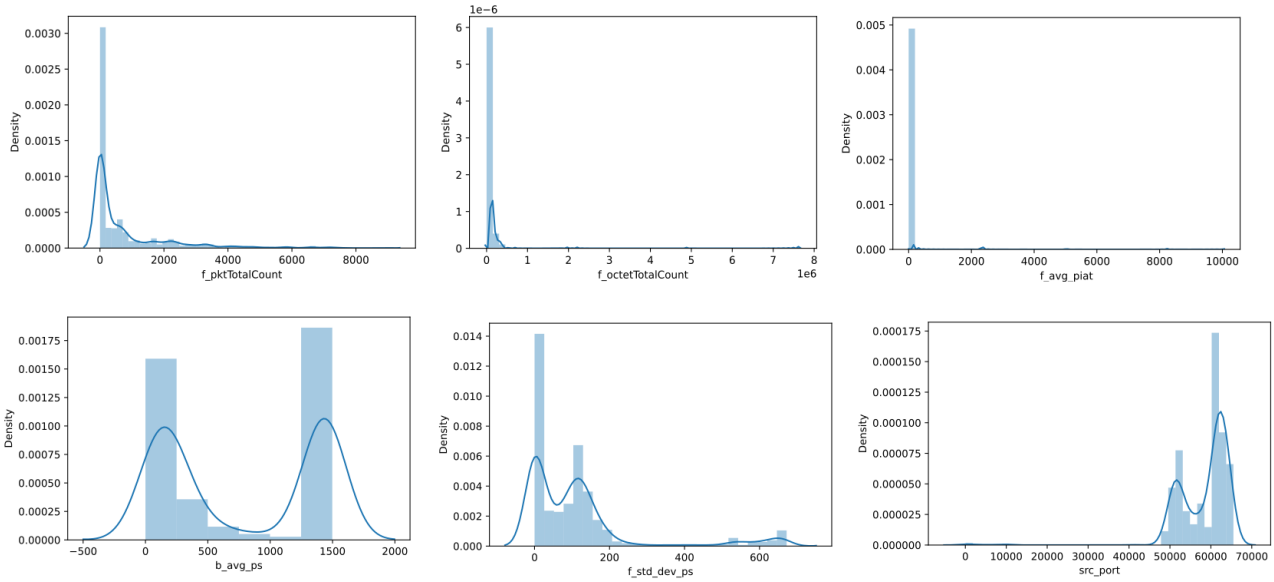

Figure 2: Data features distributions.

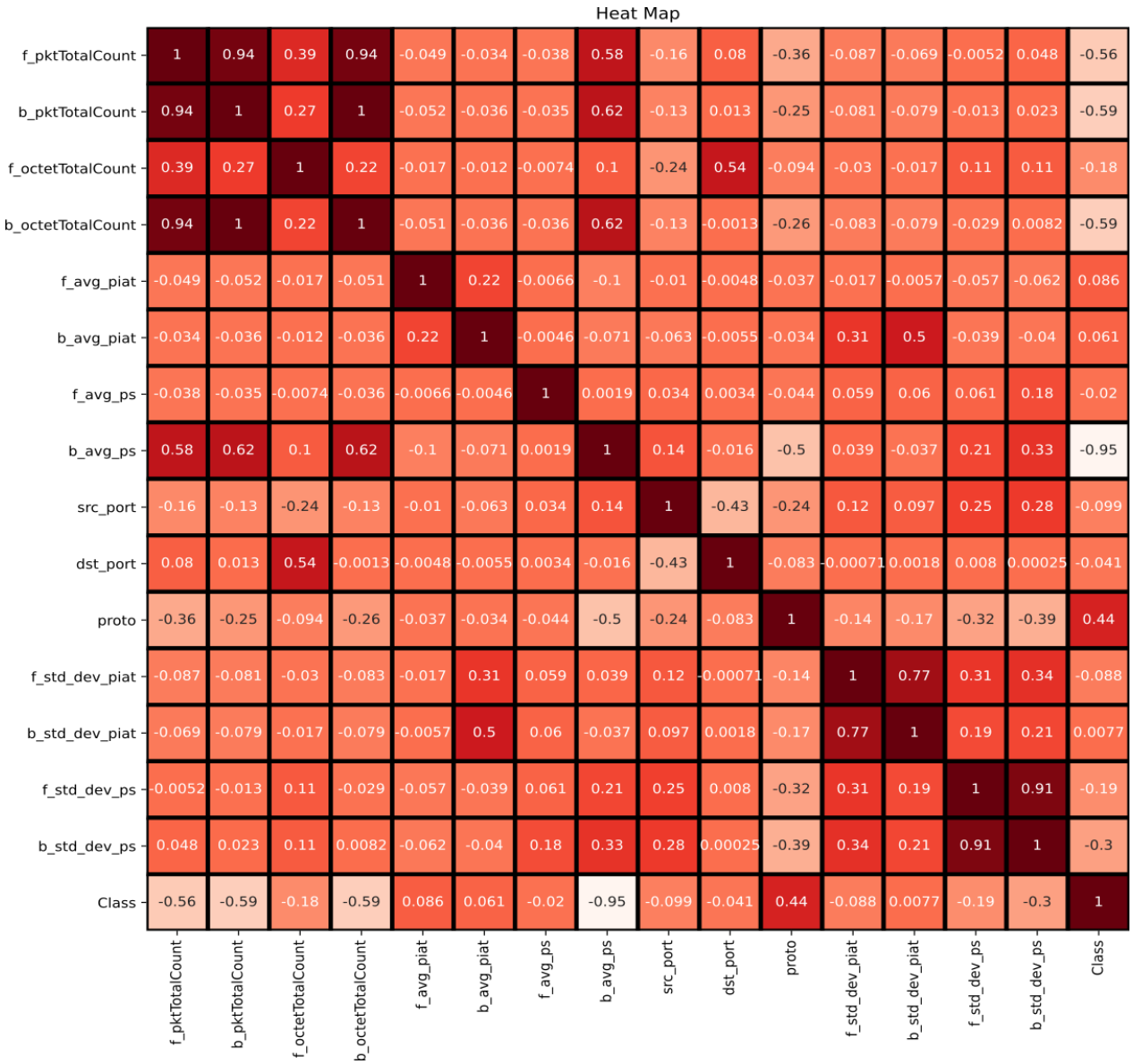

Figure 3: Heat map correlation matrix from Datavideo1.

approach. A correlation matrix is a table showing correlation coefficients between variables [9]. Each cell in the table shows the correlation between two or more variables, *i.e.*, provides information about how correlated the features are concerning others. Figure 3 shows the Datavideo1 correlation matrix. Overall, there is a high correlation of the data, highlighting a strong dependence between "PktTotal-Count" and "octelTotalCount" for both forward and backward flows; this means that the number of transmitted packets is closely related to the number of transmitted bytes.

As Figure 3 shows, there are also high correlations between "f-avg-piat" and "b-avg-piat", "f-std-dev-piat" and "b-std-dev-piat". Such a high correlation is expected since these features belong to the information given by the arrival time between packages. Another high correlation is presented between the "std-dev-ps" in forwarding and backward flows. Also, these features have a small dependence on packet sizes and the number of bytes transmitted.

We also applied a recursive elimination method to de data. The recursive elimination method builds a model on the features that remain to classify each characteristic according to its importance. In Datavideo1 this method showed that the attributes that provide the most information are "f-pktTotalCount", "b-pktTotalCount", "f-avg-piat", "b-avg-piat" and "f-std-dev-piat". Taking the correlation thrown by the matrix between "f-pktTotalCount", "b-pktTotalCount", with "f-avg-piat", "b-avg-piat" and the data thrown by the elimination method.

Table 3: Performance metrics of Datavideo1

Datavideo1	Without Feature Elimination		Normalized		Normalized and Feature Elimination	
ML Model	Accuracy	Standard Deviation	Accuracy	Standard Deviation	Accuracy	Standard Deviation
LoR	96.25	4.50	94.75	4.1	93.5	6.99
LDA	93.75	7.09	94.75	5.6	91.5	11.19
K-NN	95.0	5.36	61.25	13.6	96.5	1.65
CART	96.75	5.25	97.24	5.29	97.50	2.5
NB	89.49	10.65	86.5	15.54	88.0	9.53
SVM	85.75	15.53	84.3	15.5	91.25	7.26

Table 4: Performance metrics of Datavideo2

Datavideo1	Without Feature Elimination		Normalized		Normalized and Feature Elimination	
ML Model	Accuracy	Standard Deviation	Accuracy	Standard Deviation	Accuracy	Standard Deviation
LoR	78.88	28.63	83.91	16.40	70.16	21.71
LDA	72.66	24.30	90	14.13	68.75	21.06
K-NN	66.22	26.84	86	15.67	72.44	20.01
CART	88.66	16.83	91.11	16.80	71.25	20.3
NB	71.33	37.49	81.25	18.35	70.0	19.28
SVM	65.55	30.57	88.66	13.26	69.11	38.73

3.3 Data modelling and Performance Metrics

Tables 3 and 4 summarise the performance metrics for Datavideo1 and Datavideo2 dataset, respectively. Tables 3 and 4 show the best performance among the two datasets is for Datavideo1 since it only consisted of two types of classes, *i.e.*, video and non-video service applications.

In particular, the CART model was the best performance regarding true positive. This can be seen in the model's accuracy, which reached 96.75%, 97.24%, and 97.5% with low deviations. It is essential to notice that the CART algorithm is frequently used when for binary classification. Then, our results are expected to be precise due to the good overall performance in the CART models' binary classification. Table 3 shows that the models' accuracy improves when using feature elimination methods and a normalization of the data. The LDA reached a high accuracy, 94.7%, due to the normalization of the data. Also, K-NN accuracy improves without the redundant features, reaching 96.5% of accuracy.

As Table 3 shows, it is clear that the performance metrics of Datavideo1 are improved by applying data normalization and feature removal methods. Thus, one expected that the same behaviour happens in Datavideo2. However, as Table 4 shows, the overall accuracy in all model decrease; *e.g.*, in CART model, the accuracy reached is 91.1% only. We compared the importance of the features between Datavideo1 and Datavideo2, and it turned out that it is slightly different. The features that no longer are important in Datavideo2 are "octetTotalCount" or "scr-port". In Datavideo2, CART model can easily classify video streams and non-video flows but not classify different types of video services, *i.e.*, VoD and Livestream.

Table 5: Confusion Matrix Datavideo1

	Prediction			Class 1	Class 2
Datavideo1	Video	Non video	Precision	1	0.96
Video	134	6	Recall	0.96	1
Non video	0	160	F1-score	0.98	0.98

We wanted to dig deeper regarding CART model. Then, we provide a deeper analysis of the CART model confusion matrix. Table 5 summarises the confusion matrix for the CART model of Datavideo1, C1 represents video services, and C2 refers to non-video flows. As Table 5 shows, the CART model does not have any problem when classifying between video streams and non-video flows. There are a total of 134 true positives and 160 true negatives. In both classes, C1 and C2 reach an f1-score of

Table 6: Confusion Matrix Datavideo2

(a)

Datavideo2	Prediction	
C1 vs C2	Livestream	VoD
Livestream	99	19
VoD	7	133

(b)

Datavideo2	Prediction	
C2 vs C3	VoD	Non-video
VoD	113	0
Non-video	1	117

(c)

Datavideo2	Prediction	
C1 vs C3	Livestream	Non-video
Livestream	99	1
Non-video	3	117

Table 7: Performance Metrics per-class Datavideo2

	Class 1	Class 2	Class 3
Precision	0.91	0.83	0.99
Recall	0.83	0.94	0.97
F1-score	0.87	0.89	0.98

0.98.

It is important to note that video services have an intrinsic property concerning buffers' use. Video streams have a bandwidth dependency on the link. This fact simplifies the classification between video and non-video flows. Table 6 summarizes the confusion matrix of Datavideo2; C1 represents Livestream, C2 refers to VoD, and C3 represents Non-video flows. Similar to Datavideo1, Datavideo2 can easily distinguish between video services flows and non-video flows. Also, similar to Datavideo1, the model performance decreases when trained to classify into specific categories, *e.g.*, Livestream, VoD, etc.

Table 6(a) shows that 7 VoD flows were labelled as Livestream, indicating that the classification between video services is more complex. However, this result represents less than 10% of the total VoD flows. Table 7 summarizes the performance metrics of Datavide2 per-class. As Table 7 shows, the CART model can classify in a precise way C1 and C3, more than 90% of the samples. However, the precision for C2 is around 83%. Overall, the model results are promising, reaching f1-score around 0.87, 0.89, 0.98 in C1, C2 and C3, respectively.

4 Conclusion

Flow identification has become a prominent issue in academia and industry as a way to maintain network performance. The identification of video streaming streams is of particular relevance, as video streaming traffic is projected to skyrocket as more 5G-compatible devices are connected. As a result, we describe a method for detecting video streaming traffic in SDN in this article. Furthermore, we investigated the feasibility of using the KDN concept in traffic classification. In our initial proof of concept, we obtain the relevant information from the network traffic in the form of flow statistics, such as the total size of the flow, the total number of packets, the duration of the flow, etc. We then use that information to train six ML models to classify network streaming, and non-video traffic.

The main result of our proof of concept is that it is possible to implement an SDN scenario and implement different strategies for flow classification by following a KDN approach. Overall, our results show that ML models have no difficulty in differentiating between video and non-video flows. However, some models degrade their performance when the classification is multi-class classification, for example, when the models are trained to classify between different video services. We believe that there are similarities in their stream statistics as they fall into the category of "video flows." Thus, a more sophisticated strategy must be applied, when video service identification is needed. Another important result is that the CART model seems to be a good option for classifying video flows. In our analysis CART model reached an accuracy of 97.5% in Datavideo1 and an accuracy of 91% in Datavideo2.

Motivated by the results provided in this paper, future work will investigate Livestream services' impact on the flow classification mechanism. Also, we want to extend our investigation to the online

classification mechanism for video streaming services in SDN.

Author contributions

This was a collaborative effort by the writers.

Conflict of interest

There are no conflicts of interest declared by the authors.

References

[1] Arslan, A. K.; Tunç, Z.; Çolak, C. (2019). An Open Sourced Software for Data Transformation and an Application on Simulated Data, *International Artificial Intelligence and Data Processing Symposium*, 1–6, 2019.

[2] Bakhshi, T. (2017). Multi-feature Enterprise Traffic Characterization in OpenFlow-based Software Defined Networks, *International Conference on Frontiers of Information Technology*, 23–28, 2017.

[3] Bishop, C. M.; Nasrabadi, N. (2007). Pattern Recognition and Machine Learning, *J. Electronic Imaging*, 16, 049–901, 2007.

[4] Boutaba, R.; Salahuddin, M. A.; Limam, N. et al (2019). A comprehensive survey on machine learning for networking: evolution, applications and research opportunities, *Journal of Internet Services and Applications*, 9(1), 1–99, 2010.

[5] Callado, A. et al (2009). A Survey on Internet Traffic Identification, *IEEE Communications Surveys & Tutorials*, 11(3), 37–52, 2009.

[6] Cios, K. J.; Pedrycz, W.; Swiniarski, R. W.; Kurgan, L (2007). *Data Mining: A Knowledge Discovery Approach*, Springer-Verlag, 2007.

[7] Cisco, *Cisco Visual Networking Index: Forecast and Trends, 2017–2022*, pp. 147, 2019. [Online]. Available: https://davidellis.ca/wp-content/uploads/2019/05/cisco-vni-feb2019.pdf, Accesed on 18 April 2021.

[8] Clark, D. D.; Partridge, C.; Ramming J. C.; Wrocławski, J. T. (2003). A knowledge Plane for the Internet, *Proc SIGCOMM'03*, 3–10, 2003.

[9] Dias, K. L.; Pongelupe, M. A.; Caminhas, W. M.; de Errico, L. (2019). An innovative approach for real-time network traffic classification, *Computer Networks*, 158, 143-–157, 2019.

[10] Dhote, Y.; Agrawal S.; Deen, A. J. (2015). A Survey on Feature Selection Techniques for Internet Traffic Classification, *International Conference on Computational Intelligence and Communication Networks*, 1375–1380, 2015.

[11] Duan, M. (2018). Short-Time Prediction of Traffic Flow Based on PSO Optimized SVM, *International Conference on Intelligent Transportation, Big Data & Smart City*, 41–45, 2018.

[12] Duque-Torres, A.; Amezquita-Suárez, F.; Caicedo-Rendon, O. M.; Ordóñez A.; Campo, W. Y. (2019). An Approach Based on Knowledge-Defined Networking for Identifying Heavy-Hitter Flows in Data Center Networks, *Applied Sciences*, 9(22):4808, 2019. https://doi.org/10.3390/app9224808.

[13] Duque-Torres, A.; Pekar, A.; Seah, W. K. G.; Caicedo-Rendon, O. M. (2019). Heavy-Hitter Flow Identification in Data Centre Networks Using Packet Size Distribution and Template Matching, *IEEE 44th Conference on Local Computer Networks*, 10–17, doi: 10.1109/LCN44214.2019.8990807, 2019.

[14] Ericsson, *Ericsson Mobility Report*, June, pp. 36, 2020. [Online]. Available: https://www.ericsson.com/49da93/assets/local/mobility-report/documents/2020/june2020-ericsson- mobility- report.pdf, Accesed on 1 May 2021.

[15] Finsterbusch, M.; Richter, C.; Rocha, E.; Muller, J.; Hanssgen, K. (2014). A Survey of Payload-Based Traffic Classification Approaches, *IEEECommunications Surveys Tutorials*, 16(2), 1135–1156, 2014.

[16] Hayes, M.; Ng, B.; Pekar, A.; Seah, W. K. G. (2018). Scalable Architecture for SDN Traffic Classification, *IEEE Systems Journal*, 12(4), 3203–3214, 2018.

[17] Hou, Y.; Huang, H.; Shao, W. (2014). Traffic Classification Method by Combination of Host Behaviour and Statistical Approach, *Journal of Engineering Science and Technology Review*, 7, 151–157, 2014.

[18] Huber, S.; Wiemer, H.; Schneider, D.; Ihlenfeldt, S. (2019). DMME: Data mining methodology for engineering applications – a holistic extension to the CRISP-DM model, *12th CIRP Conference on Intelligent Computation in Manufacturing Engineering*, 79, 403–408, 2019.

[19] Hyun, J.; Tu, N. V.; Hong, J. (2017). Knowledge-defined networking using in-band network telemetry, *19th AsiaPacific Network Operations and Management Symposium*, 54–57, 2017.

[20] Kreutz, D.; Ramos, F. M. V.; Veríssimo, P. E.; Rothenberg, C. E.; Azodolmolky S.; Uhlig S. (2015). Software-Defined Networking: A Comprehensive Survey, *Proceedings of the IEEE*, 103(1), 14–76, 2015.

[21] Lopes Pereira, S.; De Castro e Silva, J. L.; Bessa Maia, J. E. (2014). Ntcs: A real time flow-based network traffic classification system, *10th International Conference on Network and ServiceManagement*, 368—371, 2014.

[22] Mestres, A.; Rodriguez-Natal, A.; J. Carner, J.; Barlet-Ros, P.; Alarcón E. et al (2017). Knowledge-Defined Networking, *SIGCOMM Comput. Commun*, 47(3), 2–10, 2017.

[23] NFStream, *Flexible Network Data Analysis Framework*, 2020. [Online]. Available: https://www.nfstream.org/, Accesed on 10 May 2021.

[24] OpenDaylight, *OpenDaylight Controller*, 2020. [Online]. Available: https://www.opendaylight.org/ Accesed on 10 May 2021.

[25] Ordonez-Lucena, J.; Ameigeiras, P.; Lopez, D.; Ramos-Munoz, J. J. et al. (2017). Network Slicing for 5G with SDN/NFV: Concepts, Architectures, and Challenges, *IEEE Communications Magazine*, 55(5), 80–87, 2017.

[26] Pekar, A.; Chovanec, M.; Vokorokos, L.; Chovancova, E.; Feciľak, P.; Michalko, M (2018). Adaptive Aggregation of Flow Records, *Computing and Informatics*, 37, 142–164, 2018, doi: 10.4149/cai_2018_1_142.

[27] Pekar, A.; Duque-Torres, A. (2018). A Network Traffic Flow Feature Measurement Tool – flowRecorder, version v1.1.2, Nov. 2018.

[28] Pekar, A.; Duque-Torres, A.; Seah, W. K. G. et al (2021). Knowledge Discovery: Can It Shed New Light on Threshold Definition for Heavy-Hitter Detection?, *J Netw Syst Manage*, 29, 24, 2021, https://doi.org/10.1007/s10922-021-09593-w.

[29] Reza, M.; Sobouti, M. J.; Raouf, S.; Javidan, R. (2017). Network traffic classification using machine learning techniques over software definednetworks, *International Journal of Advanced Computer Science and Applications*, 8, 2017.

[30] Sandvine, *Global Internet Phenomena Report 2019*, June, pp. 23, 2019. [Online]. Available: https://www.sandvine.com/global-internet-phenomena-report-2019, Accesed on 15 May 2021.

[31] Scikit-learn, *Machine Learning in Python*, 2020. [Online]. Available: https://scikit-learn.org/stable/, Accesed on 18 December 2020.

[32] Subasi, A. (2020). *Practical Machine Learning for Data Analysis Using Python // Machine learning techniquesr*, 91–202, doi:10.1016/B978-0-12-821379-7.00003-5, 2020.

[33] Van Asten, B. J.; van Adrichem, N. L. M.; Kuipers, F. A. (2014). Scalability and Resilience of Software-Defined Networking: An Overview, 1408.6760, 2014

[34] VLC, *VLC media player*, 2021. [Online]. Available: https://www.videolan.org/vlc/index.es.htmlAccesed on 11 September 2020.

[35] Wireshark, *Wireshark tool*, 2021. [Online]. Available: https://www.wireshark.org/#download, Accesed on 16 September 2020.

[36] Wowza, *Wowza Streaming Engine*, 2021. [Online]. Available: https://www.wowza.com/, Accesed on 13 April 2021.

[37] Zodiac, *The world's smallest OpenFlow SDN switch*, 2020. [Online]. Available: https://northboundnetworks.com/pages/zodiac-fx-faq., Accesed on 14 April 2021.

[38] Zou, X.; Hu, Y.; Tian, Z.; Shen, K. (2019). Logistic regression model optimization and case analysis, *IEEE 7th International Conference on Computer Science and Network Technology*, 135–139, 2019.

Journal's webpage: http://univagora.ro/jour/index.php/ijccc/

This journal is a member of, and subscribes to the principles of,
the Committee on Publication Ethics (COPE).
https://publicationethics.org/members/international-journal-computers-communications-and-control

Cite this paper as:

Castañeda Herrera, L. M.; Campo, W. Y.; Duque-Torres, A. (2021). Video Streaming Service Identification on Software-Defined Networkin, *International Journal of Computers Communications & Control*, 16(5), 4258, 2021.
https://doi.org/10.15837/ijccc.2021.5.4258