Network Traffic Analysis using Machine Learning: an unsupervised approach to understand and slice your network

Ons Aouedi, Kandaraj Piamrat, Salima Hamma, J K Menuka Perera

Noname manuscript No.
(will be inserted by the editor)

Network Traffic Analysis using Machine Learning: an unsupervised approach to understand and slice your network

Ons Aouedi[1] · **Kandaraj Piamrat**[1*] · **Salima Hamma**[1] · **J.K.Menuka Perera**[1]

Received: date / Accepted: date

Abstract Recent development in smart devices has lead us to an explosion in data generation and heterogeneity, which requires new network solutions for better analysing and understanding traffic. These solutions should be intelligent and scalable in order to handle the huge amount of data automatically. With the progress of high-performance computing (HPC), it becomes feasible easily to deploy machine learning (ML) to solve complex problems and its efficiency has been validated in several domains (e.g., healthcare or computer vision). At the same time, network slicing (NS) has drawn significant attention from both industry and academia as it is essential to address the diversity of service requirements. Therefore, the adoption of ML within NS management is an interesting issue. In this paper, we have focused on analyzing network data with the objective of defining network slices according to traffic flow behaviors. For dimensionality reduction, the feature selection has been applied to select the most relevant features (15 out of 87 features) from a real dataset of more than 3 million instances. Then, a K-Means clustering is applied to better understand and distinguish behaviors of traffic. The results demonstrated a good correlation among instances in the same cluster generated by the unsupervised learning. This solution can be further integrated in a real environment using network function virtualization.

Keywords Machine Learning · Feature Selection · Clustering · Unsupervised Learning · Network Traffic · Traffic analysis · Network Slicing

1 Introduction

Under the evolution of smart devices, the networks become increasingly heterogeneous, dynamic, and this has pushed network operators to search for new concepts of management. Consequently, designing network architecture that

*Correspondence to kandaraj.piamrat@univ-nantes.fr

can handle the heterogeneity and maximize resource utilization efficiency is a major challenge [1]. In this context, several solutions have been proposed including Network Slicing (NS) and Machine Learning (ML). Applying these concepts together can provide an autonomic and intelligent network resources management as well as can yield performance optimization on a large-scale environment, which will accommodate 5G quality of service (QoS) requirements [2]. In fact, ML is deployed to solve complex problems without explicit programming where the algorithm can model and learn the underlying behavior using a training dataset/environment. Its efficiency is validated and it has achieved promising results in several domains and network management is one of them [3]. On the other hand, NS provides the network as a service (NaaS) for different use cases, allowing infrastructure provider to unlock their physical network to several occupants [4] [5]. It partitions the network infrastructure into isolated network slices where each slice has its own resources and performance requirement. NS has various advantages for the 5G networks and beyond. First, it can support multi-tenancy sharing the same physical network infrastructure. Second, NS can handle the diversified services with stringent QoS and guarantee service level agreements (SLA). Third, as it addresses the diversified service requirements, instead of the "one-fit-all" principal [4], it leads to a flexible and efficient utilization of the limited resources.

However, since the network behaviors changes depending on several factors such as user mobility, location, social events, it is a challenging task to define explicitly the network state. In this context, the resource allocation to network slicing instances becomes more difficult. Moreover, the huge amount of new network traffic and applications made the creation of network slices manually time-consuming [6]. Consequently, the process should be flexible, answer to instantaneous requests, and depend on the traffic behavior of the end-user. Monitoring the performance of network slices and network resources optimization (e.g., auto-scaling) can take place via intelligent traffic management, which understands the behavior of connected smart devices and applications. Therefore, the intelligent definition of Network (sub)-Slicing according to the traffic behaviors will become an important issue to support different use cases in the heterogeneous environments and even within the slice (sub-slicing).

To efficiently understand the traffic and automatically generate network slicing, ML offers these benefits since it can inspect a large quantity of data and find a useful pattern from this data in a reasonable time. It allows the system to come up with rules for automating tasks. The most important advantage of ML is its capability to deal with complex problems. Moreover, analysis of these vast datasets using the machine learning approach promises novel discoveries. Therefore, integrating these tools into the traffic analysis and NS will enable network operators to implement self-configuring, self-healing, and self-optimizing networks [7], which in turn provide a *zero-touch* management. Within the field of ML, a broad distinction could be made between *supervised* and *unsupervised* learning. The main objective behind *supervised learning* is to identify a mapping from the input features to an output class, which requires a fully labeled dataset. On the other hand, the objective of *unsupervised learning*

is to find a structure (i.e., pattern) in the inputs without the need of an output class.

In practice, the learning ability of ML models is only as good as the given data and features. The increase in the data dimensionality may decrease the performance of an algorithm as well as cause an extra computational cost (e.g., storage and processing) [8] and curse of dimensionality problem. Consequently, to make the raw data suitable for analysis, preprocessing steps should be applied and feature selection is one of them.

In this study, an ML-based slice-defining solution is introduced. The proposed architecture uses network statistics and an offline process for understanding network traffic patterns with a clustering algorithm. Preliminary results of this work appeared in [9] where only 4 features (out of 87) were selected based on Ryu controller's default capability. However, this small number of features is not enough to recognize traffic behavior. In order to better define each cluster, more features have been added and analyzed in this work.

Therefore, the main contributions of this paper are:

- *Feature selection*: We include additional experiments involving features selection. Since the networks is a time-critical system and ML algorithms are as good as the quality of data, we show that the set of features needed to distinguish between applications can be reduced using feature reduction techniques, which can improve the performance of learning algorithms (i.e., classification or clustering) and decreases the overhead both for data collection and model computation.
- *Traffic clustering*: We apply an unsupervised-learning method to find different clusters of the traffic using the previously selected features. These clusters are constructed from traffic with similar behavior.
- *Cluster analysis*: We focus on the analysis of each cluster (i.e., future slice definition) and explain its behavior according to the property of selected features.

The rest of the paper is organized as follows: Section 2 presents related work of similar researches that use ML for traffic analysis and network slicing. Section 3 introduces the essential background in order to understand our proposition. Section 4 describes the proposed solution and presents the dataset used during this work. Section 5 presents the experimental results along with its analysis. Section 6 provides some inside discussions and finally Section 7 concludes the paper.

2 Related Work

The existing research with the most similar context to this paper is presented in [10] where the authors have discussed software defined networks (SDN), network function virtualization (NFV), Machine learning, and big data driven network slicing for 5G. In this work, they have proposed an architecture to

classify network traffic and used those decisions for network slicing. According to the paper, with the exponentially increasing number of applications entering the network, it is impossible to classify traffic by a single classification model. Therefore, they have used the K-means clustering algorithm to cope with this issue. By using this unsupervised algorithm, they have grouped the data set and labeled them. They have set the number of clusters (i.e. slices) k=3 associating three bandwidths. With this grouping and labeling, they have trained five classification models and have compared their accuracy. The results show that the user can play the video smoothly and with better quality after network slicing is deployed. However, the number of applications used in this work is limited (21 applications) and without an analysis of their behavior. There is no information about the choice of the attributes during the clustering and no metric used to select the best number of clusters.

In [11], the authors developed a proof-of-concept of network slicing in a real mobile virtual network operator (MVNO) network. They classify mobile traffic into fine grained slices, by identifying application types in order to apply QoS control and various network function per application. However, only five flow features are selected to train an 8-layer deep learning model. Also, there is no information about the behaviors of each slice.

Another interesting work is presented in [12] where the authors analyzed the traffic to capture its behavior of all the cells in each cluster. Firstly, a clustering model (K-means) explored the data to identify the traffic behavior of the clusters. Secondly, they propose a traffic utilization forecasting model for each cluster using several ML models. Based on their proposition, the authors demonstrate how clustering model can help to forecast the network traffic cells and provide self-organizing networks. However, the authors analyze the traffic utilization and not the network applications. Also, the framework is not dedicated to network slicing and no feature selection method has been used.

In [13], an SDN-based slicing prototype based on the home devices is presented. The authors present three types of slicing strategy. However, the ML technique was not used in this work where the authors have manually specified home users requirements according to the restrained resources of the home network. Thus, this solution can not scale well.

In [5], the authors proposed a hybrid learning algorithm in order to design efficient slicing classification. For the given data features, they classify the traffic to the exact network slices like enhanced Mobile Broadband (eMBB), massive Machine Type Communication (mMTC), and Ultra-Reliable Low-Latency Communication (URLLC). However, this solution classifies the data to the static service types (eMBB, URLLC, mMTC) defined in the 5G era. Moreover, it is based on supervised learning algorithms where the models tend to learn over-fit spaces to the labeled samples.

In [14], the authors proposed a machine learning-based network sub-slicing framework in a 5G environment where each slice is divided into a virtualized sub-slice of resources. To do so, support vector machine (SVM) algorithm is used for feature selection based on the smart applications in IoT devices, then the K-means algorithm is applied to assign the group based on similar types

of application services. However, there is no information about the data, the initial and selected features as well as a lack of analyzing the clusters (sub-slices) behaviors.

3 Background

In this section, we provide a brief overview of basic concepts, which are useful for understanding the rational behind the proposition.

3.1 Network-Slicing based architecture

Network slicing is one of the most popular technologies used to enhance resource utilization [15] [16]. It has been proposed with the aim of sharing the same physical network infrastructure to address the diversified service requirements, instead of "one-fit-all" principal [4]. Therefore, the network slicing concept offers reduced Capital Expenditure (CAPEX) and Operational Expenditure (OPEX) costs. According to the International Telecommunication Union (ITU), network slicing *"is recognized as a key enabler for the support of different types of services"*. It has several advantages such as supporting multi-tenancy, providing customized slices for different services with different QoS and resources requirements [1]. As shown in Figure 1, in order to meet the SLA of each slice, different virtualized network resources (via virtual network functions or VNFs) are jointly allocated based on the slice behaviors. The VNFs can be computing, communication, and caching resources that are running over the same physical infrastructure. Moreover, as slices can be created on-demand or modified as needed, network slicing improves the flexibility and adaptability in network management [17]. Therefore, it has a profound implications on resource management.

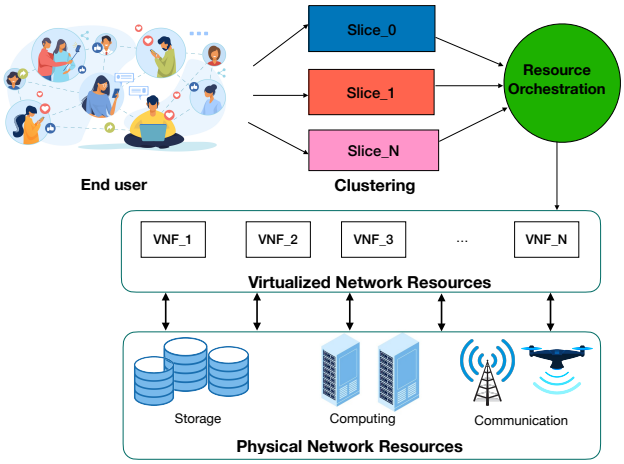


Fig. 1: Network Slicing using behavior clustering

3.2 Unsupervised Learning

With the rapid increases in the size and complexity of data, unsupervised learning will be the trend in the future [18]. It tries to find relationships between the inputs without having any prior knowledge of the outputs. In other words, the aim of unsupervised learning is processing data for knowledge discovery, and using the knowledge for reasoning and decision making. Clustering algorithms are one of the mostly used unsupervised learning processes. These algorithms are used to categorize the input data into distinctive clusters (i.e., groups) by examining the similarity between them. Each observation within the same cluster is having greater similarity as compared to the observation in other clusters [19]. Several clustering algorithms have been proposed and used over the past few years. However, in this section, we have selected to present only a few representative ones. Also, Table 1 presents the advantages and weaknesses of each algorithm.

– **K-means**

K-means algorithm is the oldest and popular partitional method [20]. It is well known for its efficiency in clustering large-scale data sets. It aims to partition the data into K clusters based on a similarity measure. In other words, the examples belong to the same cluster have high similarity as compare to those of other clusters. Each cluster has a category center μ_k. The Euclidean metric is selected as the criterion of similarity. The sum of squares of the distance between the points in each cluster to the center of the cluster μ_k is calculated to minimize the sum of squares within each cluster. The objective function is written as:

$$V = \sum_{i=1}^{K} \sum_{n=1}^{N} ||x_i - \mu_k||^2$$

where x is the sample to be clustered, N is the number of samples and K is the number of clusters. K points are randomly selected from the datasets as the initial clustering center. The Euclidean distance from each sample to the cluster center is calculated. The sample is returned to the nearest cluster center. The new clustering center is obtained by calculating the average value of the newly formed data objects of each cluster. If there is no change in two successive iterations, it shows that the sample adjustment is over and the clustering criterion function has been converged.

– **Hierarchical Clustering**

Unlike K-means, Hierarchical Clustering (HC) does not need to pre-specify the number of clusters. HC constructs the clusters by recursively partitioning the data objects in a top-down fashion or by merging them in a bottom-up fashion and this process can be visualized as a dendrogram [21]. It uses two popular approaches: *Agglomerative* and *Divisive* clustering. The agglomerative approach goes from bottom to top where all data objects are considered as different clusters and then at each step, it merges the objects by similarity

until it forms a single cluster. The divisive approach works in the top-down fashion where it assumes that all the objects are in a single cluster and then at each step, it splits the cluster until each cluster contains a single data object.

– **Density-Based Spatial Clustering of Applications with Noise (DBSCAN)**

DBSCAN is one of the most popular unsupervised learning algorithms. It aims to use the density in order to divide the samples of the dataset into various clusters [22]. Specifically, it partitions the dataset into different clusters using two parameters, ϵ and MinPts (the minimum number of points to create a cluster). Any data object that does not fit into any cluster is considered an outlier. That is why unlike K-means and Hierarchical Clustering, DBSCAN can handle the outliers objects (i.e., noise) in the dataset. Its performance depends on the parameters settings (i.e., MinPts, ϵ).

Table 1: Comparison of the popular unsupervised algorithms [23] [24]

Method	**Advantages**	**Weaknesses**
K-means	– Fast and simple – Less complex – Easy to implement – Highly scalable	– Requires the number of clusters in advances
Hierarchical Clustering	– No need to specify the number of clusters in advance – Good visualisation capabilities	– High computational cost – Can not handle the outliers – Not scalable – Requires choosing the decomposition process
DBSCAN	– Does not need the number of cluster in advance – Handle the outliers within the dataset	– Needs a careful selection of its parameters (i.e., MinPts, ϵ)

In this study, we have selected k-means as our clustering algorithm since it eases further implementation. To estimate the optimal number of clusters, several clustering validity indicators have been proposed in the literature. In our study, we will use two popular clustering validation methods, which are *Davies-Bouldin Index* and *Silhouette Coefficient*.

– **Davies-Bouldin Index**

The basic principle of validity clustering is minimizing the intra-cluster distance or maximizing the inter-cluster distance. This study used Davies-

Bouldin Index (DBI) to measure inter and intra-cluster distance and find the ideal number of clusters [25], which is calculated as follows:

$$DBI = \frac{1}{K}\sum_{i=1}^{K}\max_{i\neq j}\frac{(\hat{d}_i + \hat{d}_j)}{d_{i,j}}$$

where K is the number of clusters, $\hat{d}_i$ is the average distance between each object in the ith cluster and the centroid of the ith cluster, $\hat{d}_j$ is the average distance between each object in the jth cluster and the centroid of the jth cluster, and $d_{i,j}$ is the Euclidean distance between centroids of the ith and jth clusters. A lower DBI means a better cluster configuration.

– **Silhouette Coefficient**

The silhouette Coefficient (SC) measures how similar an object is to its own cluster (cohesion) compared to other clusters (separation) [26]. It evaluates the intra-cluster and nearest-cluster distance for each object. In contrast to DBI, the closer SC is to 1, the better the clustering result. In other words, SC ranges from -1 to 1. The coefficient equals -1 means clustering is wrong; 0 means indifference (same); and 1 means both clusters are far away. However, it is more computationally demanding than DBI. SC is calculated as follows:

$$SC(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}}$$

where $a(i)$ is the average distance of the object x_i to other objects in the same cluster as i, and $b(i)$ is the minimum average distance from the object x_i to objects in a different cluster (i.e., x_i is not a member).

3.3 Feature Selection

As the amount of high-dimensional data has increased in recent years, ML tasks have many benefits. However, this data can consist of irrelevant (a feature that provides no useful information) and redundant (a feature with predictive ability covered by another), which increase search space and decrease predictive quality [27]. Thereby, with the increase in the dimensionality (i.e., features) of data, the predictive ability of an algorithm decreases as well as causing an extra computational cost for both storage and processing [8]. These challenges are referred to as *curse of dimensionality*, which is one of the most important challenges in ML, first introduced by Bellman in 1961 to indicate that ML algorithms can work fine in low dimensions and become intractable with high-dimensional data [28]. In practice, each algorithm is only as good as the given features. Also, the learning ability of the model depends not only on the quality of collected features but also on the number of features considered. The model will perform poorly if the number of variables is larger than the number of observations (over-fitting) [29]. In this situation, the model can easily separate the training data but it will be inaccurate on the test data. In

fact, these make dimensionality reduction difficult tasks not only because of the higher size of the initial dataset but because it needs to meet two challenges, which are to maximize the learning capacity and to reduce the number of features. Further, to make the raw data suitable for analysis, feature selection steps should be applied.

4 Proposed Solution

The proposed solution consists of four steps as shown in Figure 2. In the first step, the dataset is cleaned and prepared for the second step, which is the feature selection. Then, an unsupervised ML algorithm is applied to cluster the applications that have similar traffic behavior. Finally, we analyze network data to define network slices according to the traffic flow behaviors of each cluster.

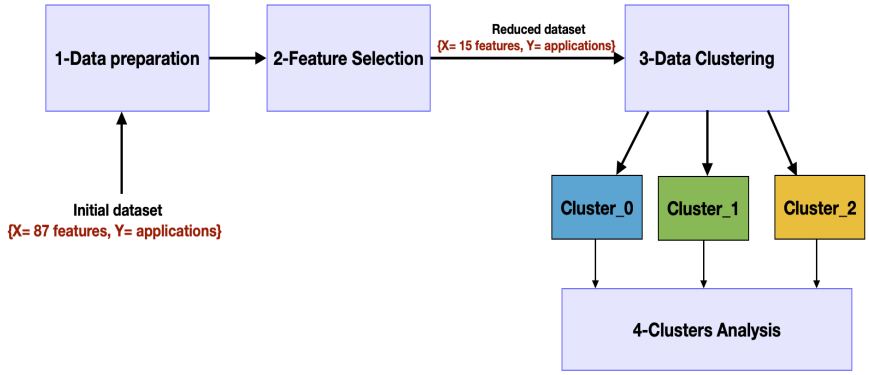


Fig. 2: Steps of our proposition

4.1 Dataset description

For this paper, "IP Network Traffic Flows, Labeled with 75 Apps" dataset from Kaggle[1] database was used. This dataset is a good match for our objectives and satisfies all the three main components of a good dataset, which are real-world, substantial and diverse. This dataset was created by collecting network data from Universidad Del Cauca, Popayán, Colombia using multiple packet capturing and data extracting tools [30]. This dataset is consisting of 3,577,296 instances (observations), 78 applications (Facebook, Google, YouTube, Yahoo, Dropbox, and so on), and 87 features. It is originally designed for application

[1] https://www.kaggle.com/jsrojas/ip-network-traffic-flows-labeled-with-87-apps

classification. Each row represents a traffic flow from a source to a destination and each column represents features of the traffic data.

4.2 Data Preparation

In the data cleaning process, several operations need to be done before it is ready for machine learning model training. If there are duplicate instances in the dataset, it will cause bias in the machine learning algorithm. So to avoid the biasing, those duplicates need to be identified and remove from the dataset. Also, the main issue is that a real dataset can have redundant and irrelevant features. Consequently, preprocessing of data is a necessary task, which can be done with several techniques among which the feature selection.

Moreover, in this dataset, there are several features containing different data types; however, some ML models can only work with numeric values. To use those data types for the ML models training, it is necessary to convert or reassign numeric values to represent its correlations with other features. In this work, we have converted *timestamp* and $IPaddresses$ values to numerical values. After this cleaning process, the dataset is ready for feature selection.

4.3 Feature Selection

The feature selection methods try to pick a subset of features that are relevant to the target concept. This step has become an indispensable component of the ML process. They can be grouped into two broad categories (filter and wrapper) based on their dependence on the inductive algorithm that will finally use the selected subset [31]. We based our analysis on the obtained features by Recursive Feature Elimination (RFE) method from [32]. As a reminder, RFE is a wrapper methods that recursively evaluates alternative sets by running some induction algorithms on the training data. It uses the estimated accuracy of the resulting classifier as its metric. Starting from all the feature sets, the method recursively removes features in order to maximize accuracy. Then it ranks the features based on the order of their elimination. The algorithm used here is the classification and regression tree (CART).Using RFE, we have derived a method to identify the best 15 features out of 87 features in our dataset. These features are summarized in Table 2.

4.4 Data Clustering

Data clustering is another part of this proposition, wherein the selected features are used to cluster the traffic according to their similar properties. For this, K-means unsupervised learning model has been deployed as it is effective, fast, scalable, and less complex. The number of clusters has been determined using the *Davies-Bouldin Index* and *Silhouette Coefficient* as previously described in Section 3. From the K-means clustering results, the number of clusters (k value) has been varied from 2 to 14, and then the DBI and SC are

Table 2: Best fifteen features of our data set

Features Name	**Description**	**Rank**
Flow.IAT.Max	The maximum value of the inter-arrival time of the flow (in both directions).	1
DestinationIP	The destination IP address.	2
SourcePort	The source port number.	3
SourceIP	The source IP address of the flow.	4
Fwd.IAT.Total	The total Inter-arrival time in the forward direction.	5
Init_Win_Bytes_Backward	The total number of bytes sent in the initial window in the backward direction.	6
DestinationPort	The destination port number.	7
Fwd.Packet.length.Max	The maximum value in bytes of the packets length in the forward direction.	8
Timestamp	The instant the packet was captured stored.	9
Subflow.Fwd.Bytes	The average number of bytes in a subflow in the forward direction.	10
Init_Win_Bytes_Forward	The total number of bytes sent in the initial window in the forward direction.	11
Bwd.Packet.Length.Mean	The mean value in bytes of the packets length in the backward direction.	12
Bwd.Packet.Length.Max	The maximum value in bytes of the packets length in the backward direction.	13
Fwd.Packet.Length.Std	The standard deviation in bytes of the packets length in the forward direction.	14
Flow.Duration	The total duration of the flow.	15

calculated for each k value. As can be seen from Fig. 3 and Fig. 4, $k = 3$ has the lowest score with the DBI and highest score with the SC, which reflects that there are three types of traffic behaviors that can be identified from this dataset. The three types of network traffic behaviors recognized by the K-means algorithm will be analyzed in order to understand their characteristics.

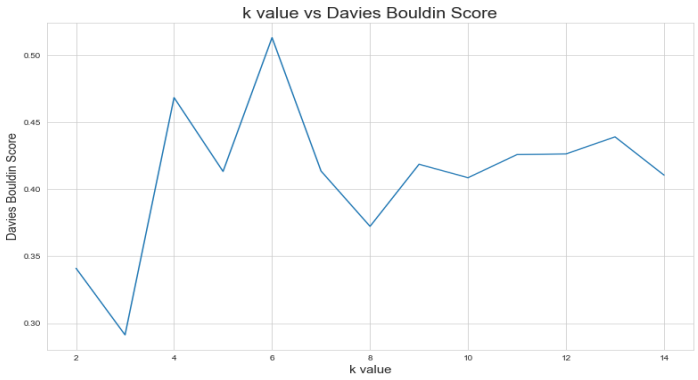


Fig. 3: k value vs Davies Bouldin Score

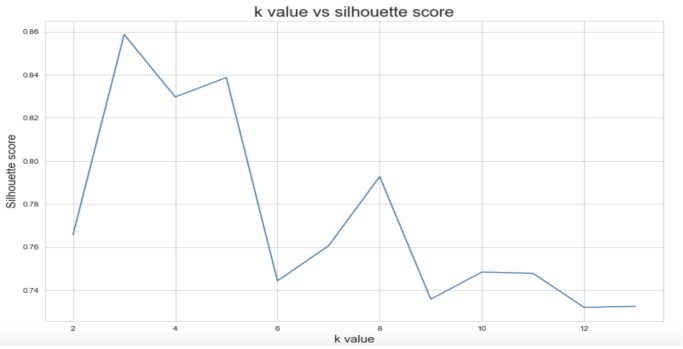


Fig. 4: k value vs Silhouette Coefficient

5 Results and analysis

In this section, we analyse and discuss the obtained results (via the selected features in each obtained cluster). We separate the analysis into three parts: *application-related*, *time-related*, and *bandwidth-related*.

5.1 Application-related features analysis

After running K-means algorithm, three clusters were obtained with different applications distribution as presented in Fig. 5. For the sake of clarity, we limit the plot to the majority applications (those with at least 10 000 instances).

It can be noticed that the majority of the traffic comes from web browsing as we can see the huge numbers of HTTP related flows in all the clusters, following by Google, SSL, and YouTube. Please note here that the SSL flows are encrypted connections (and ports); therefore, we cannot discuss about the final applications but we can infer some behaviors via time and bandwidth related features in the next sections. Concerning the cluster 0, we can observe that it contains the most significant amount of traffic from HTTP Proxy and Google (as HTTP Connect flows are related to TCP connection initiation process, we will not emphasize it in this paper). HTTP Proxy means that the browsing flows have the campus proxy server as the destination before going outside. This is also shown in the Fig 6. We limit the plot to the majority destination ports number where the port number 3128 (proxy) represents most of the traffic **(92%)** especially for cluster 0. Concerning the cluster 1, we can observe that most of the traffic are HTTP (direct connections via port 443 and 80, seen in Fig. 6), follow by Google. As for cluster 2, no trend can be concluded except that there are also many direct connections as seen in Fig. 6.

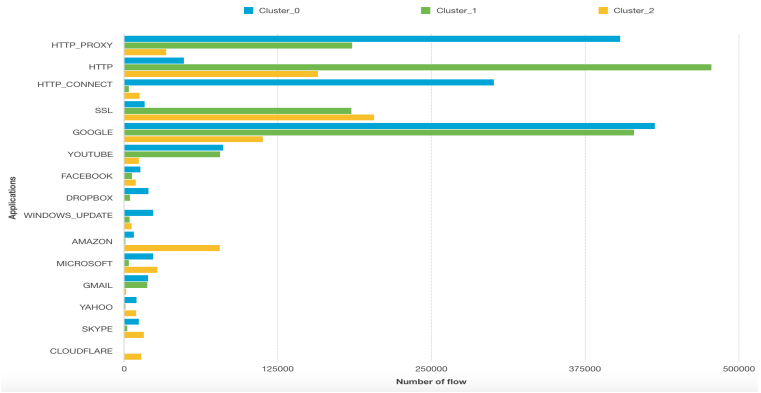


Fig. 5: Application distribution of each cluster

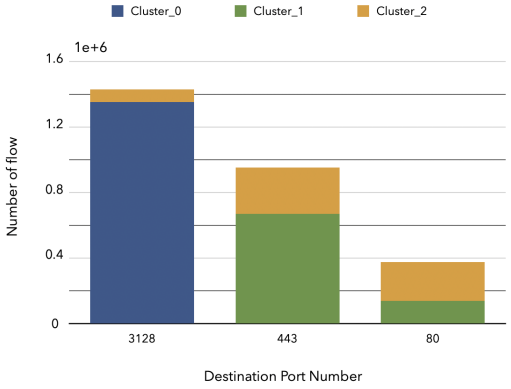


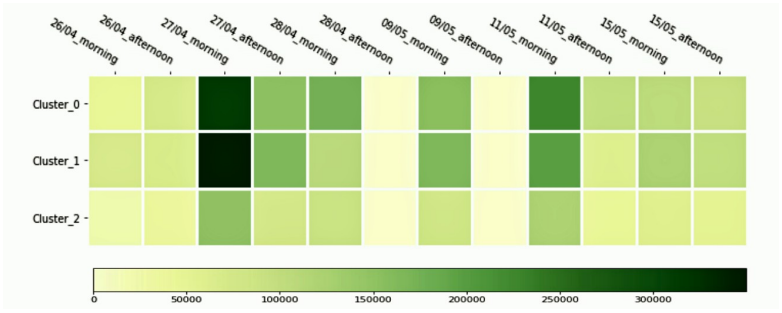
Fig. 6: Distribution of destination ports

5.2 Time-related features analysis

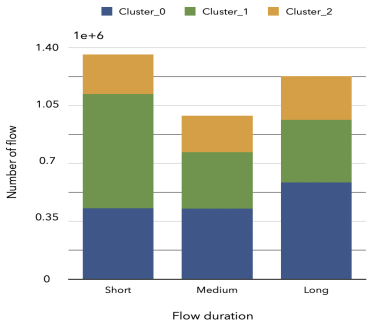
In this subsection, we analyze the time-related features in the resulting clusters. The dataset used in this study consists of 6 days of data collected from university network traffic in the morning and afternoon period, the distribution is presented in the Fig. 7 (a). Colors present the distribution of traffic for the morning/afternoon for each day. The darker the color the more dominant the traffic within that period.

Fig. 7 (b) presents the distribution of flow duration among different clusters categorized by duration as *short* (less than 100 milliseconds), *medium* (between 100 milliseconds until 10 seconds), and *long* (longer than 10 seconds). It can be observed that the majority of flows **(50%)** in cluster 1 have very short duration (less than 100 milliseconds). On the contrary, the majority of the flows **(40%)** in cluster 0 have a duration of more than 10 seconds.

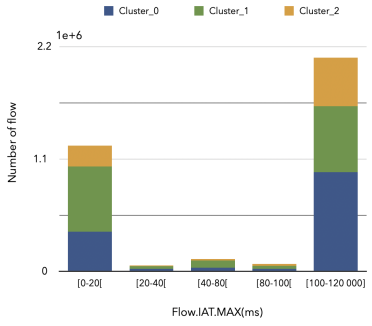
Moreover, Fig. 7(c) presents the distribution of the maximum inter-arrival time (IAT) among clusters. It can be noticed that for the cluster 0 and 2, most of the flows arrived with high inter-arrival time with **67%** and **65%** of the flows respectively. As for the cluster 1, we can observe that approximately half the flows **(45%)** arrived with IAT less than 20 milliseconds and the other half of the flows **(46%)** arrived with IAT more than 100 milliseconds. These time-related features can help operators to manage network resources; for example, in terms of scheduling and reservation, they can fine tune the related parameters according to user behavior in the same cluster (slice) and the available resources.



(a)



(b)



(c)

Fig. 7: Distribution of time related behavior

5.3 Bandwidth-related feature analysis

Figure 8 shows the flow distribution with respect to the maximum packet length and the number of bytes sent in the initial window size in both directions (backward and forward). The majority of flows in the backward (Fig. 8 (a)) are of small size, mostly control or acknowledgement packets. Approximately **64%** (cluster 0) and **76%** (cluster 1) are less than 753 bytes. This is correlated to the

number of bytes sent in the initial window size in the backward direction (Fig. 8 (c)), which shows that **84%** (cluster 1) and **89%** (cluster 0) remain below 1310 bytes. We note, however, that for a window size greater than 5242 bytes, cluster 1 sent more bytes than the other two. Therefore, the resource reservation for this cluster must be particularly adapted. In the forward direction, about **75%** of the transmitted packets are small, i.e., less than 657 bytes. For cluster 0, these packets are mostly sent when the initial window size exceeds 6553 bytes **(65%)**. For cluster 1, **58%** of the packets are sent when this window size less than 1310 bytes and about a third are sent when this size exceeds 6553 bytes. Whereas for Cluster 2, we observe an almost equal distribution between small **(45%)** and large **(41%)** window sizes. In summary, special attention will be given to resource reservation for outbound Cluster 0 traffic.

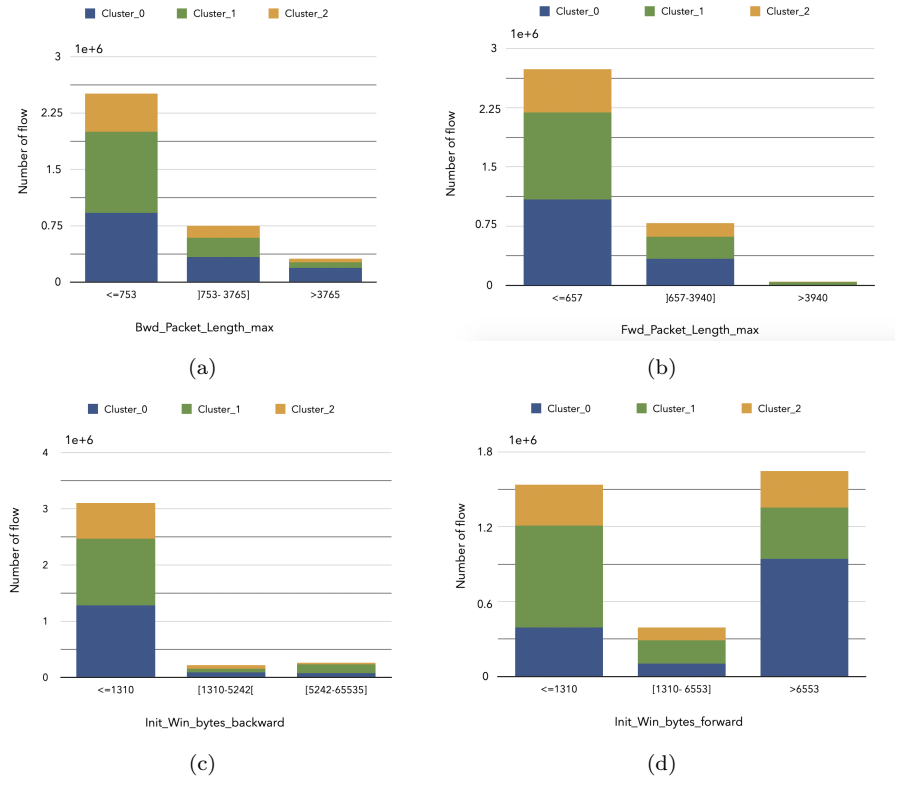


Fig. 8: Distribution of bandwidth related behavior

6 Discussion

This work aims to design an efficient network slicing using an unsupervised ML algorithm. We have demonstrated that the analysis of the massive quantity of data in a short duration, and discovering the complex traffic behavior in an efficient way could be made possible by the use of ML algorithms. Also, in this work, we have used unlabeled data, which is often abundant and easily available. Moreover, instead of building dedicated networks for all services, NS can be used to allow the operators to create a customized network and allocate the optimal resources for each cluster created through an unsupervised ML algorithm (K-means in our case). In other words, the slice specifications can be applied to the aggregate traffic of certain flows (clusters) in order to maximize flexibility and guaranteeing the QoS to the end-users. In addition, the default service types (e.g., eMBB, URLLC, mMTC) defined in the 5G era are static [33], while new types of services continually evolve. If the behaviors of these new services are not integrated efficiently, it may make the network less reliable and less optimized. To eliminate these issues, our proposition can help to automatically recognize a new type of service and establish the required network slices. Moreover, it can be used to discover different behavior into common slices (i.e., sub-slicing).

Pros and cons of the proposed network slicing solution

- Labeling all the traffic to their particular slice requires a huge effort of human annotators sometimes with a specific domain of expertise. Therefore, the proposed solution helps to find traffic pattern without human intervention and avoids time-wasting as maximum as possible.
- However, in order to save and process the network data with ML models a high amount of storage and computational resources are needed, as well as there are some threats on information security.

7 Conclusion and future work

This study has been carried out as a proof of concept while combining ML with traffic analysis, in particular, for defining network slicing. This study demonstrates the powerful capacity of machine learning to analyze traffic behaviors and hence can create intelligent network slices. Feature selection (RFE) method and K-means algorithm are deployed in this proposition to find the most relevant features and clustering, respectively. An experimental analysis of these clusters (future slices) using the selected features in order to find useful behaviors has been conducted in order to be utilized easily for network slicing and resource management. These features are classified according to application-related, time-related, and bandwidth-related features. The obtained results have demonstrated some clear behaviors that need special attention. For example, on the morning of 04/27, cluster 0 and cluster 1 (future slices) need the infrastructure provider to allocate more resources because the

traffic load increases during this time. Also, the majority traffic of cluster 0 is proxy, thus special attention should be given in terms of resources (e.g., more CPU and caching resource to the proxy server) for this cluster (future slice). Consequently, this helps the infrastructure provider to anticipate the network state and thereby avoid the violation of SLA.

For future work, other clustering algorithms such as DBSCAN (Density-based spatial clustering of applications with noise) or hierarchical clustering should also be applied to the dataset in order to compare the performance to the K-mean clustering used in this paper. Implementation of network slicing infrastructure should also be done in order to bench-marking the signaling and overhead generated by this solution.

References

1. X. Shen, J. Gao, W. Wu, K. Lyu, M. Li, W. Zhuang, X. Li, and J. Rao, "Ai-assisted network-slicing based next-generation wireless networks," *IEEE Open Journal of Vehicular Technology*, vol. 1, pp. 45–66, 2020.
2. R. Fantacci and B. Picano, "When network slicing meets prospect theory: A service provider revenue maximization framework," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 3179–3189, 2020.
3. R. Boutaba, M. A. Salahuddin, N. Limam, S. Ayoubi, N. Shahriar, F. Estrada-Solano, and O. M. Caicedo, "A comprehensive survey on machine learning for networking: evolution, applications and research opportunities," *Journal of Internet Services and Applications*, vol. 9, no. 1, pp. 1–99, 2018.
4. X. Li, M. Samaka, H. A. Chan, D. Bhamare, L. Gupta, C. Guo, and R. Jain, "Network slicing for 5g: Challenges and opportunities," *IEEE Internet Computing*, vol. 21, no. 5, pp. 20–27, 2017.
5. M. H. Abidi, H. Alkhalefah, K. Moiduddin, M. Alazab, M. K. Mohammed, W. Ameen, and T. R. Gadekallu, "Optimal 5g network slicing using machine learning and deep learning concepts," *Computer Standards & Interfaces*, p. 103518, 2021.
6. V. P. Kafle, Y. Fukushima, P. Martinez-Julia, and T. Miyazawa, "Consideration on automation of 5g network slicing with machine learning," in *2018 ITU Kaleidoscope: Machine Learning for a 5G Future (ITU K)*, pp. 1–8, IEEE, 2018.
7. A. Mestres, A. Rodriguez-Natal, J. Carner, P. Barlet-Ros, E. Alarcón, M. Solé, V. Muntés-Mulero, D. Meyer, S. Barkai, M. J. Hibbett, *et al.*, "Knowledge-defined networking," *ACM SIGCOMM Computer Communication Review*, vol. 47, no. 3, pp. 2–10, 2017.
8. A. L'heureux, K. Grolinger, H. F. Elyamany, and M. A. Capretz, "Machine learning with big data: Challenges and approaches," *Ieee Access*, vol. 5, pp. 7776–7797, 2017.
9. M. P. J. Kuranage, K. Piamrat, and S. Hamma, "Network traffic classification using machine learning for software defined networks," in *In-*

ternational Conference on Machine Learning for Networking, pp. 28–39, Springer, 2019.
10. L.-V. Le, B.-S. P. Lin, L.-P. Tung, and D. Sinh, "Sdn/nfv, machine learning, and big data driven network slicing for 5g," in *2018 IEEE 5G World Forum (5GWF)*, pp. 20–25, IEEE, 2018.
11. A. Nakao and P. Du, ""Toward in-network deep machine learning for identifying mobile applications and enabling application specific network slicing"," *IEICE Transactions on Communications*, pp. 1536–1543, 2018.
12. L.-V. Le, D. Sinh, B.-S. P. Lin, and L.-P. Tung, "Applying big data, machine learning, and sdn/nfv to 5g traffic clustering, forecasting, and management," in *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, pp. 168–176, IEEE, 2018.
13. S. Wang, X. Wu, H. Chen, Y. Wang, and D. Li, "An optimal slicing strategy for sdn based smart home network," in *2014 International conference on smart computing*, pp. 118–122, IEEE, 2014.
14. S. K. Singh, M. M. Salim, J. Cha, Y. Pan, and J. H. Park, "Machine learning-based network sub-slicing framework in a sustainable 5g environment," *Sustainability*, vol. 12, no. 15, p. 6250, 2020.
15. X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network slicing in 5g: Survey and challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 94–100, 2017.
16. I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing and softwarization: A survey on principles, enabling technologies, and solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2429–2453, 2018.
17. Q. Ye, J. Li, K. Qu, W. Zhuang, X. S. Shen, and X. Li, "End-to-end quality of service in 5g networks: Examining the effectiveness of a network slicing framework," *IEEE Vehicular Technology Magazine*, vol. 13, no. 2, pp. 65–74, 2018.
18. M. Usama, J. Qadir, A. Raza, H. Arif, K.-L. A. Yau, Y. Elkhatib, A. Hussain, and A. Al-Fuqaha, "Unsupervised machine learning for networking: Techniques, applications and research challenges," *IEEE Access*, vol. 7, pp. 65579–65615, 2019.
19. D. Tomar and S. Agarwal, "A survey on data mining approaches for healthcare," *International Journal of Bio-Science and Bio-Technology*, vol. 5, no. 5, pp. 241–266, 2013.
20. A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM computing surveys (CSUR)*, vol. 31, no. 3, pp. 264–323, 1999.
21. S. C. Johnson, "Hierarchical clustering schemes," *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.
22. M. Ester, H.-P. Kriegel, J. Sander, X. Xu, *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise.," in *kdd*, vol. 96, pp. 226–231, 1996.
23. D. Tomar and S. Agarwal, "A survey on data mining approaches for healthcare," *International Journal of Bio-Science and Bio-Technology*, vol. 5, no. 5, pp. 241–266, 2013.

24. A. Ahmad and S. S. Khan, "Survey of state-of-the-art mixed data clustering algorithms," *Ieee Access*, vol. 7, pp. 31883–31902, 2019.
25. D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE transactions on pattern analysis and machine intelligence*, no. 2, pp. 224–227, 1979.
26. P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.
27. A. Janecek, W. Gansterer, M. Demel, and G. Ecker, "On the relationship between feature selection and classification accuracy," in *New challenges for feature selection in data mining and knowledge discovery*, pp. 90–105, PMLR, 2008.
28. P. Domingos, ""A few useful things to know about machine learning"," *Communications of the ACM*, vol. 55, no. 10, pp. 78–87, 2012.
29. I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, ""Gene selection for cancer classification using support vector machines"," *Machine learning*, vol. 46, no. 1-3, pp. 389–422, 2002.
30. J. S. Rojas, Á. R. Gallón, and J. C. Corrales, "Personalized service degradation policies on ott applications based on the consumption behavior of users," in *International Conference on Computational Science and Its Applications*, pp. 543–557, Springer, 2018.
31. P. Langley *et al.*, "Selection of relevant features in machine learning," in *Proceedings of the AAAI Fall symposium on relevance*, vol. 184, pp. 245–271, 1994.
32. O. Aouedi, K. Piamrat, and B. Parrein, "Performance evaluation of feature selection and tree-based algorithms for traffic classification," in *2021 IEEE International Conference on Communications (ICC) DDINS Workshop*, (Montreal, Canada), June 2021.
33. R. Li, Z. Zhao, X. Zhou, G. Ding, Y. Chen, Z. Wang, and H. Zhang, "Intelligent 5g: When cellular networks meet artificial intelligence," *IEEE Wireless communications*, vol. 24, no. 5, pp. 175–183, 2017.