# A Classification Framework for IoT Network Traffic Data for Provisioning 5G Network Slices in Smart Computing Applications

Ziran Min*, Swapna Gokhale†, Shashank Shekhar‡, Charif Mahmoudi‡, Zhuangwei Kang*,
Yogesh Barve*, Aniruddha Gokhale*
*Dept. of CS, Vanderbilt University, Nashville, TN, USA.
Email: {ziran.min, zhuangwei.kang, yogesh.d.barve, a.gokhale}@vanderbilt.edu
†Dept. of CSE, University of Connecticut, Storrs, CT, USA. Email: swapna.gokhale@uconn.edu
‡Siemens Technology, Princeton, NJ 08540, USA. Email: {shashankshekhar, charif.mahmoudi}@siemens.com

***Abstract*—Existing massive deployments of IoT devices in support of smart computing applications across a range of domains must leverage critical features of 5G, such as network slicing, to receive differentiated and reliable services. However, the voluminous, dynamic, and heterogeneous nature of IoT traffic imposes complexities on the problems of network flow classification, network traffic analysis, and accurate quantification of the network requirements, thereby making the provisioning of 5G network slices across the application mix a challenging problem. To address these needs, we propose a novel network traffic classification approach that consists of a pipeline that combines Principal Component Analysis (PCA), with KMeans clustering and Hellinger distance. PCA is applied as the first step to efficiently reduce the dimensionality of features while preserving as much of the original information as possible. This significantly reduces the runtime of KMeans, which is applied as the second step. KMeans, being an unsupervised approach, eliminates the need to label data which can be cumbersome, error-prone, and time-consuming. In the third step, a Hellinger distance-based recursive KMeans algorithm is applied to merge similar clusters toward identifying the optimal number of clusters. This makes the final clustering results compact and intuitively interpretable within the context of the problem, while addressing the limitations of traditional KMeans algorithm, such as sensitivity to initialization and the requirement of manual specification of the number of clusters. Evaluation of our approach on a real-world IoT dataset demonstrates that the pipeline can compactly represent the dataset as three clusters. The service properties of these clusters can be easily inferred and directly mapped to different types of slices in the 5G network.**

***Index Terms*—Network traffic classification, Unsupervised Machine Learning, Clustering, 5G, Dynamic network slicing, Traffic analysis, Machine Learning.**

## I. Introduction

Our lives are increasingly dependent on various Internet of Things (IoT) devices, such as home voice controllers, doorbell cameras, smart light switches, etc. According to the State of IoT report [1], the number of global IoT connections grew by 8% from 2021 to 12.2 billion active endpoints by the end of May 2022. Moreover, the introduction of the 5G network enables the mobility, reliability, and scalability of IoT devices. Compared to 4G and wireless LoRaWAN, the 5G network is an ideal solution for a wide range of IoT applications, offering a balance between high-speed connectivity and cost-effectiveness, while promoting faster mobility of the connected devices and improving the bandwidth of the wireless network through the introduction of powerful base stations and the application of mmWave and Multiple-Input Multiple-Output (MIMO) technologies. The 5G network slicing technology, which enables virtual network services on top of shared physical infrastructures, helps network providers to increase flexibility by allowing customized network service. It improves resource utilization by sharing the physical infrastructure, as well as Quality of Service (QoS) by allowing the network providers to prioritize various traffic flows and allocate resources more effectively. Overall, enabling 5G in IoT offers a number of benefits for both the network service providers and IoT applications.

The heterogeneous network traffic from the 5G-enabled IoT devices, however, creates a number of challenges for the network providers in traffic classification and analysis, which must be addressed to improve the network performance and resource management, and enhance network planning, design, and security. A number of challenges are associated with classifying and analyzing heterogeneous network traffic in IoT, which include:

**Device characteristics and network requirements:** A number of end devices play different roles in IoT networks; thus, they could be using different protocols at the multiple layers of the network stack, and have different network requirements.

**Unavailability of labeled data:** The huge volumes of data and information gathered by IoT devices continuously is not only a challenge to the storage costs but also makes manual labeling of data infeasible.

**Dynamic and mobility-influenced traffic patterns:** 5G-enabled IoT environments often experience variations in network traffic patterns due to the movement and mobility of connected devices. This makes it challenging to accurately classify different types of network flows under such dynamic conditions.

979-8-3503-9730-7/22/$31.00 ©2022 IEEE

**Interpretability of classification results:** A network flow record contains a number of features, such as source IP, destination IP, port number, etc. Automatically identifying the most influential features that can impact the classification results is necessary. Furthermore, the interpretability of the classification results is crucial as the network slices are constructed using this traffic classification.

To address these challenges and satisfy the dynamic network requirements of 5G-enabled IoT devices, we propose a new network traffic classification approach for effective 5G network slicing. Considering the lack of labeled data and the complex nature of the IoT traffic data, we use unsupervised machine learning to discover underlying patterns and relationships. We term our classification approach as recursive KMeans, which while relying on the conventional KMeans algorithm, addresses the limitations of traditional KMeans clustering, such as assuming spherical clusters, sensitivity to initialization, and requiring manual specification of the number of clusters. In summary, we define our research problem as finding solutions to cluster heterogeneous IoT traffic without labels in a simple and interpretable manner to target 5G network slicing. To that end, this paper makes the following contributions:

- Design and implement an unsupervised and recursive clustering approach that applies to standard network traffic data sets.
- Analyze and explain the results of clustering towards 5G network slicing in simple and understandable terms.
- Visualize high-dimensional clusters in a clear and presentable manner.

The rest of this paper is organized as follows: Section II briefly summarizes related work; Section III presents our SDN-based 5G & IoT network architecture; Section IV introduces our approach; Section V presents and analyzes the results; and finally, Section VI provides concluding remarks and discusses potential future work.

## II. Related Work

This section provides a sampling of related work on IoT network traffic classification and compares it with ours. We discuss both traditional classification approaches as well as behavior-based classification methods.

Traditional network traffic classification aims at categorizing network traffic into different classes based on the network flow characteristics such as source/destination IP, source/destination port and payload, etc. A number of port-based and payload-based classification techniques are applied in traditional network traffic classification. However, traditional techniques have several limitations as discussed below.

**Port-based Classification:** A number of recent applications use dynamic or random port policy, such as peer-to-peer (P2P) applications. A previous study [2] indicates that port-based analysis is not able to identify 30%-70% of Internet traffic.

**Payload-based Classification:** Here, the classifier uses the packet content's syntax to classify the network traffic. This technique is not useful, however, for some encrypted data.

Therefore, recent studies have been focusing on behavior-based network traffic classification, which aims at identifying and analyzing the pattern of traffic by monitoring all the network traffic that is received by endpoints [3]. Due to the huge size of the datasets, the large variety of traffic patterns and the lack of labels, machine learning methods, especially unsupervised classification methods, such as KMeans, are widely used in a variety of scenarios.

A number of traffic clustering methods have been proposed, but the representative and scale of the dataset used in previous research may limit their generalizability. For example, Singh et al. [4] used the KMeans clustering algorithm to create clusters of sub-slices for similar application services while Sani et al. [5] divided the traffic into two clusters to reduce traffic and improve the QoS of the 5G network. However, due to the lack of description of the dataset used in their experiments, the generalizability of their approach cannot be determined. Aouedi et al. [6] used the KMeans algorithm to better understand and distinguish behaviors of traffic. Although they showed a good correlation among the network flows in the same resulting cluster, the dataset they used consists of only 6 days of data collected from a university in the morning and afternoon period, which is not representative of the general traffic pattern in an IoT network.

Another common limitation of previous studies is the method used to select features. Feature selection will directly influence the performance of the traffic clustering model. Although general IoT network traffic usually consists of numerous pieces of information that can be extracted as features, the clustering process will be difficult and less accurate if too much irrelevant information is included. Therefore, input features must accurately represent the traffic behaviors. Le et al. demonstrated that the KMeans clustering algorithm helps to implement the self-organizing network (SON) architecture [7], [8]. They manually selected the basis of average (in MB) and percentage values of the amount of different traffic in different time slots as their features. They showed that users had a better experience playing videos after applying the clustering algorithm to the network slicing deployment.

However, we surmise that the performance of the approach may be further improved by taking into account other valuable and useful information in the traffic, such as the protocol type and the flow duration. Moreover, such manually-engineered features cannot be extracted directly via standard network analyzers like Tcpdump or Wireshark and thus need extra effort to be prepared. Singh et al. [4] used the Support Vector Machine (SVM) for feature selection before running the KMeans. However, SVM is a supervised algorithm, which means that the quality of the model heavily relies on the number and quality of labels. Our work does not assume labeled datasets.

The predefined number of clusters for the KMeans algorithm is also a key factor that affects the performance and accuracy of clustering results. To achieve good clustering results, previous works either manually choose a specific number or use the cluster number corresponding to the elbow point

of the cost plot. The number of clusters usually corresponds to the number of slices used for the applications in the 5G network. Therefore, the cluster number used in prior work may far exceed the upper bound of the slice number in real life. Further, due to a lack of a metric to identify the best setting for the number of clusters, a decision needs to be made regarding the trade-off between network performance with more slices and efficiency with fewer slices assuming perfect clustering.

Overall, the limitations of previous Kmeans-based clustering algorithms include a) the representative nature and scale of the dataset, which limits the generalizability of the findings; b) the features of the network flow are limited, and the feature selection relies either on manual engineering or manual labels; c) requires the number of clusters to be specified ahead of time; d) a lack of systemic analysis of classification results in applying them to 5G network slicing. Therefore, to address the above limitations all at once, we propose a recursive KMeans algorithm, which 1) is applicable to the network flows generated by standard network analyzing tools; 2) is able to extract useful features automatically; and 3) decides the number of clusters appropriately without human intervention.

## III. SDN-based 5G & IoT Network Architecture

Our work assumes a flexible SDN-based 5G and IoT network architecture that is compatible with multiple slice types and customized slice configurations, comprising five main components: User Equipment(UE), Radio Access Network (RAN), Multi-access Edge Computing (MEC), Core Network (CN), and SDN Controller as shown in Figure 1. This architecture is capable of efficiently allocating network resources to customized and prioritized network slices by leveraging the results from clustering traffic.

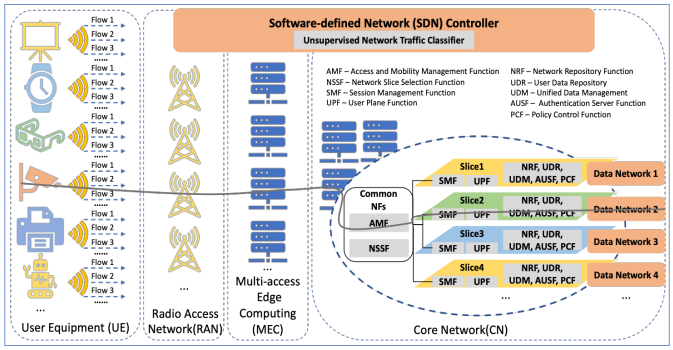


Fig. 1. SDN Based 5G&IoT System Architecture

**User Equipment (UE):** In a 5G network, the devices or equipment used by end users are called UEs. UEs can be smartphones, tablets, laptops, smart home devices, etc. Each UE can access the Internet first via the radio access network (RAN) and then via the network services provided by MEC, CN, and SDN Controllers. Each UE is identified by a unique identifier called the International Mobile Subscriber Identity (IMSI), which is authenticated by a mobile network operator and stored in the UE's Subscriber Identity Module (SIM) card. Moreover, a UE can generate multiple network flows simultaneously and each of them can be independently classified, managed, and scheduled.

**Radio Access Network (RAN):** The RAN in 5G is responsible for handling wireless communication between UE and CN. A RAN consists of **a)** base stations, such as gNodeB, which transmit and receive signals to and from the UE, **b)** radio network controllers (RNC), which are responsible for managing the base stations, and **c)** transport networks, which move the network data between base stations and CN.

**Multi-access Edge Computing (MEC):** The MEC server is widely adopted and integrated into 5G network for the deployment of computing and storage resources at the edge, which is closer to the UE. Therefore, the MEC server is able to support low-latency and high-bandwidth services. Recently, MEC technology has been integrated into many industrial applications such as manufacturing, healthcare, transportation, and smart cities.

**Core Network (CN):** The architecture in Figure 1 is a service-based (SB) standalone 5G network, which applies the new 5G RAN and CN technology instead of the legacy 4G network to manage and control network communications. The service-based architecture enables scalable, flexible and cost-effective deployment of network functions as micro-services. It also improves the programmability of the CN, thereby enabling automation and orchestration of network functions and helping the network to be more agile and provide real-time services. Several key functions in service-based 5G network are: **a)** Access and Mobility Function (AMF), which is responsible for managing the mobility of UEs within the network; **b)** Session Management Function (SMF), which is responsible for creating, managing, and terminating sessions between the UE and the network; **c)** User Plane Function (UPF), which is the data plane function responsible for forwarding and processing the network traffic between the UE and the network; **d)** Network Slice Selection Function (NSSF), which is responsible for selecting, provisioning, and managing the appropriate number of network slices for the UE based on their network requirements and other QoS needs.

**Software-defined Networking (SDN) Controller:** Compared to the traditional network, SDN separates the control plane, which is responsible for managing and controlling the network flows, from the data plane, which is responsible for forwarding and processing the network flows. This separation improves the flexibility and enables programmability of the network. In 5G networks, the SDN controller provides a centralized view of control and management for the different network functions and components. Specifically, the SDN controller can monitor the network traffic in real-time. It can also collect data from the 5G network and utilize machine learning techniques to make intelligent decisions for network traffic classification, prediction, and management. Additionally, the SDN controller can also adapt to varying network conditions, requirements, and user demands, making real-time decisions to improve network performance and deliver fine-grained Quality of Service (QoS). Our previous work [9], [10] deployed the SDN controller in both wired and 5G networks to improve

network performance by minimizing the queuing latency and service time.

The network slicing technology enables a single 5G network to be divided into multiple virtual network slices. Each network slice can have its own virtualized infrastructure (such as computing power and storage), network functions (such as SMF, UPF as shown in Figure 1), and security mechanisms (such as firewalls and intrusion detection). A UE can be associated with one or more network slices at the same time. Moreover, we can also deploy management and orchestration tools among the network slices for traffic monitoring, troubleshooting, and scaling the slice. Overall, network slicing in a 5G network provides flexibility, efficient resource allocation, improved network scalability, orchestration, QoS, increased security, and reduced costs.

Network slicing is a technique used in 5G networks to allocate resources to different services based on their specific requirements. However, determining the optimal number of network slices is a challenge. Having too many slices might lead to underutilization, while having too few could cause overcrowding and degrade the QoS. Therefore, it is crucial to find a balance that allows for efficient allocation of resources and maintains the desired QoS for each application. In this work, we build a novel framework for network data classification to determine the right number and type of network slices. In this way, our classification approach can contribute to efficient resource allocation and improved network performance.

The classification of network traffic plays a crucial role in effectively mapping different types of services to network slices based on their unique QoS requirements. The 3rd Generation Partnership Project (3GPP) has defined several types of services for 5G and IoT network slicing:

**Massive Machine Type Communications (mMTC):** Introduced in 3GPP Release 13 [11], mMTC is designed to support a large number of low-power, low-cost, and low-data-rate devices, such as industrial sensors and smart building devices. The main QoS requirements for mMTC are maximizing coverage and device density, with less stringent demands than other service types.

**Ultra-Reliable and Low-Latency Communications( URLLC):** Introduced in 3GPP Release 14 [12], URLLC provides extremely low-latency and high-reliability communications for devices with high data rate requirements, such as industrial robots, drones, autonomous vehicles, and virtual and augmented reality headsets. URLLC has more stringent QoS requirements than mMTC, focusing on high-reliability and low-latency network services.

**Enhanced Mobile Broadband (eMBB):** Introduced in 3GPP Release 15 [13], eMBB supports high-bandwidth, high-performance devices with data rates of up to several Gbps, such as smartphones, tablets, and cameras. The QoS requirements for eMBB include high bandwidth and throughput, among other metrics.

By accurately classifying network traffic, it becomes possible to map each type of service to an appropriate network slice, ensuring that the QoS requirements of each service are met. Classification results are also valuable for network function configuration, network slice management, orchestration, and troubleshooting. For effective decision-making and resource allocation, classification results must be easily interpretable and rationalized, allowing network administrators to allocate resources efficiently based on the specific requirements and priorities of different traffic classes.

## IV. Network Traffic Classification Approach

This section describes our network traffic classification approach to determine the most appropriate number of 5G network slices for IoT applications. We chose the unsupervised machine learning approach as the backbone of our classification framework for the following reasons. From the perspective of the data set, unsupervised methods do not require labeled data for training, which makes them more applicable for network traffic classification where data volume is large, and the labeling process is intensive and time-consuming. Thus, unsupervised learning helps to save the cost of manual labeling. Because of its ability to discover hidden patterns, unsupervised learning can also adapt to dynamic changes in traffic patterns. This is particularly useful in the 5G&IoT network, where it becomes possible to add new UEs or applications without retraining the model. Finally, unsupervised learning shows better performance in unbalanced data sets where the number of examples for each class is not equal. This property is relevant in a heterogeneous 5G&IoT network which is characterized by varying data rates, diverse QoS requirements, heterogeneous traffic patterns, and uneven distribution of devices. Additional benefits from the above characteristics can be leveraged even from a system perspective. Unsupervised learning can accommodate a large volume of data, which makes it easy and efficient to deploy while the IoT system is scaling up/down. Finally, it can be applied in a real-time manner, which satisfies the IoT communication requirements.

We propose a recursive KMeans-based classification approach, as shown in Figure 2. This classification approach can also be deployed as a network function placed in the SDN controller as presented in Figure 1. The steps in the pipeline are as follows:

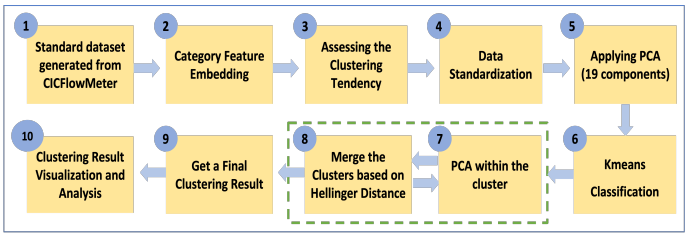


Fig. 2. Pipleine of the Classification Approach

In step 1, we prepare the data set, which can be generated by a standard network analyzer, CICFlowMeter. The data set [14] we used is generated by Télécom SudParis and composed of fifteen IoT devices, a Raspberry pi 3, an access point, and a server. A set of interactions with these devices were performed to allow the generation of real traffic.

In step 2, we pre-process the data set by converting the timestamp and IP features, such as Flow ID, Source IP, and Destination IP, to numerical values. This conversion streamlines the data representation and facilitates the clustering process. However, it is crucial to recognize that using numerical values may create a perception of ordering or distance between IP addresses that might not accurately reflect their actual relationships. To counteract the potential consequences of this representation, we meticulously analyzed the dataset and performed experiments to confirm that the selected representation does not substantially affect the clustering algorithm's performance.

In step 3, we use the Hopkins statistic [15] to assess the clustering tendency of the data. The Hopkins statistic is used in cluster analysis to determine the suitability of a data set for clustering. The value of Hopkins statistics ranges from 0 to 1. If the value is less than 0.5, then it is unlikely that the data set has statistically significant clusters. If the value is close to 1, we can conclude that the data set is significantly clusterable [16] The Hopkins statistic is especially useful when working with large data sets, as it can quickly determine the clustering tendency.

After computing the value of the Hopkins statistic, in step 4, we standardize the data and remove the features with zero variance, such as Bwd PSH Flags[1] and Fwd URG Flags[2]. These features need to be removed in order to prepare for the Principal Component Analysis (PCA).

PCA is a dimensionality reduction technique that reduces the number of features in the data set while maintaining as much of the original information as possible by finding the directions of maximum variance in the data and projecting the data onto a new lower-dimensional space [17]. In this process, if the variance of a feature is zero, its covariance with other features will be zero as well, which will affect the accuracy of PCA. In step 5, we apply PCA to reduce features in the data. We used the cumulative explained variance ratio to determine the number of principal components to retain, so that $90\%$ variance in the data can be explained.

After reducing the data set to its principal components, we normalize each feature to [0, 1] because the KMeans algorithm is sensitive to the scale of the features because of its distance-based nature. If the scale of the features is varying, the distance between observations and cluster centroids will not accurately reflect the similarity between the data points. Then in step 6, we **a)** run the KMeans algorithm on the normalized data for a range of values of $k$, which indicates the number of clusters, **b)** calculate the distortion score, which is a clustering index that reflects the sum of the squared distances between each data point in a cluster and the cluster centroid, and **c)** identify the elbow point on a distortion score curve by applying a kneed algorithm [18]. The elbow point indicates the optimal number of clusters in a dataset and determines the initial number of clusters in our approach, denoted as $best_k$.

In step 7, we apply PCA again to the corresponding original features in each cluster. By doing so, we can reveal how much each feature dominates the clustering result. More specifically, we let $n$ principal components that can be interpreted as linear combinations of $m$ traffic features. We use a matrix $C$ to store the principal components explained by variance. An element $c_{ij}$ in the matrix C indicates the correlation between $i_{th}$ principal component and $j_th$ feature. The sign of $c_{ij}$ indicates the direction of the correlation, and there are $m * n$ elements in $C$. Therefore, the contribution of the $j_th$ feature in the $i_{th}$ principal component can be represented by:

$$s_{ij} = \frac{|c_{ij}|}{\sum_{j=1}^{m}|c_{ij}|}, \forall i \in [1, n] \tag{1}$$

Moreover, we assume the proportion of variance explained by each principal component is stored in a matrix $R$. An element $r_i$ in the $R$ matrix represents the proportion of the $i_{th}$ principal component. We use matrix $P$ to store the contribution of each feature across all principal components, and any element $p_j, \forall j \in [1, m]$ can be calculated with:

$$p_j = s_{ij} ri, \forall i \in [1, n], \forall j \in [1, m] \tag{2}$$

In step 8, we use Hellinger distance [19] to evaluate the similarity between clusters. The distance between the two cluster distributions $P_1$ and $P_2$ can be represented by:

$$H(P_1, P_2) = \frac{1}{\sqrt{2}}\sqrt{\sum_{i=1}^{m}(\sqrt{p_{1i}} - \sqrt{p_{2i}})^2} \tag{3}$$

After calculating the Hellinger distance between every pair of clusters, we get a distance matrix $D$, and the element $d_{ij}$, $\forall i, j \in$ [1,best_k] indicates the distance between $i_{th}$ cluster and $j_{th}$ cluster.

Then, we apply KMeans again on matrix $D$ to merge similar clusters. The number of clusters is reduced using this merging strategy. Then, we repeat step 7 to step 8 until the average Hellinger distance of the matrix $D$ meets the given threshold.

In steps 9 and 10, we visualize the clustering results and analyze the traffic statistics within every cluster.

In summary, the novelty of our approach includes: 1) Applying PCA before running KMeans, which efficiently reduces the dimension of the features, thereby significantly decreasing the cost of KMeans algorithm; 2) Applying PCA after the first run of KMeans to calculate the contribution of original features to the clustering results, thereby helping to merge similar clusters and simplify the clustering results; and 3) Introducting Hellinger distance to quantify the similarity between every pair of clusters, thereby helping to identify the optimal number of clusters automatically.

## V. Results and Discussion

We conducted our experiments on the publicly available data set of real-world IoT traffic collected by Télécom Sud-Paris [14]. The data set consists of 164 files with a total size of 3.0G and contains more than 5 million instances collected

[1]Bwd PSH Flags indicate whether the Push flag is set in the TCP header of the packets in the backward direction of the flow.

[2]Fwd URG Flags indicate whether the Urgent flag is set in the TCP header of the packets in the forward direction of the flow.

over the past 3 years up to the present time. In this section, we present and analyze the results obtained at each step of our approach, shown in Figure 2.

### A. Dimensionality Reduction – PCA

Prior to applying the PCA, we pre-processed the data set and assessed the clustering tendency by computing the Hopkins statistic. The value of the Hopkins statistic is $0.9986$, which suggests that our data set has a high clustering tendency. Figure 3 shows the results of the initial PCA. It reveals that as the number of principal components increases, the cumulative explained variance ratio also increases; however, the marginal effect of the number of principal components on the cumulative explained variance ratio diminishes with the increase in the number of principal components increases. Note that the cumulative explained variance ratio indicates how much information retained in the principal components is used in the analysis. The blue dashed line in Figure 3 labels the elbow point of the cumulative explained variance ratio curve, which helps to select the appropriate number of principal components; here, 19 such components are able to explain $90.52\%$ of the variance in the original data and adding more does not significantly improve the cumulative explained variance ratio. Thus, the dimensionality of the data set is reduced by a fourth, from 87 to 19 features.

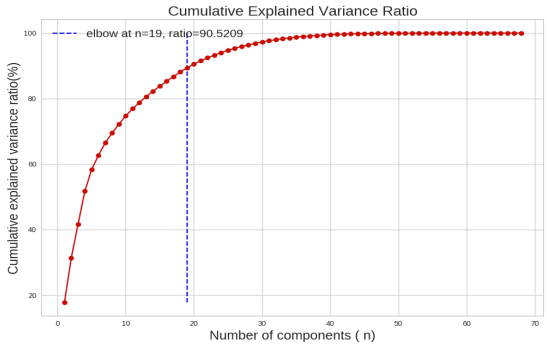


Fig. 3. Initial PCA – Cumulative Explained Variance Ratio

### B. Initial Clustering – KMeans

Figure 4 illustrates the distortion score elbow for KMeans clustering, with the number of clusters (k) plotted on the x-axis, and the distortion score plotted on the left y-axis. The distribution score shows a decreasing trend with increasing number of clusters, but the incremental gain in the distribution score gradually diminishes. The gray dashed line indicates the elbow point, which is located at $k = 25$. At this point, although increasing the number of clusters increases the distribution score, the impact of any additional clusters is negligible. The right y-axis indicates the KMeans fit time, which increases proportionally with the number of clusters.

### C. Recursive KMeans – Hellinger Distance

The KMeans clustering algorithm gave us $25$ clusters as the optimal number, each of which is described by 19 principal components. Then, we calculated the Hellinger distance between every pair based on Eq.(3). Figure 5 shows the Hellinger distance among $25$ clusters in a heat map. The

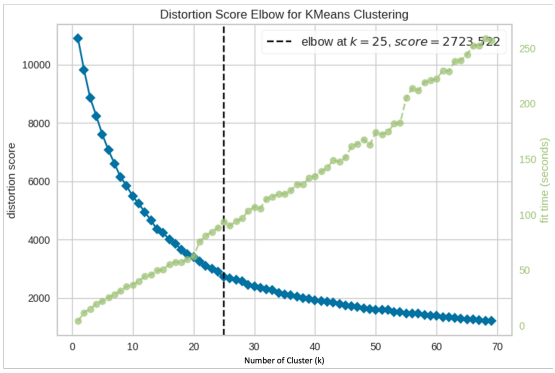


Fig. 4. Initial KMeans Elbow Curve after PCA

distances range from $0$ to $0.42$, where the distance between a cluster to itself is $0$. Heuristically, we assume that clusters with distances between $0$ and $0.2$ are highly similar. Thus, we can reduce the number of clusters to improve their interpretability by merging similar clusters. Therefore, applying the KMeans algorithm here is necessary to determine which clusters should be merged. In contrast to the previous KMeans application, the data used to drive the KMeans algorithm at this step is a matrix of Hellinger distances. Similar to Figure 4, we identified the elbow point using the Kneed algorithm, which indicates that the current optimal number of clusters is 7. Therefore, we recursively merge the 25 clusters into 7, and then repeat the calculation of the Hellinger distance and applying KMeans. Subsequent iterations merge the 7 clusters into the 3, and we re-calculate the Hellinger distance among them.

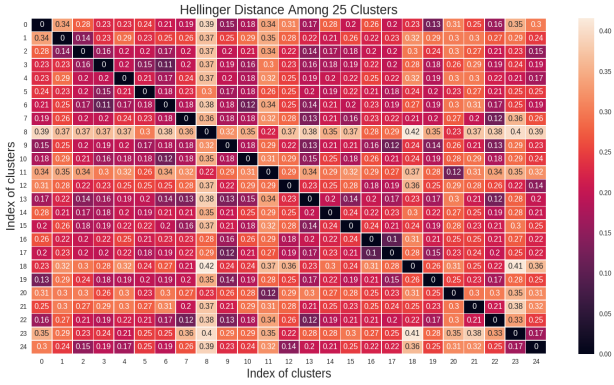


Fig. 5. Hellinger Distance – 25 Clusters

To evaluate the performance of our approach, we compare it with the traditional KMeans algorithm on the same data set, in terms of the number of clusters, variance, and standard deviation (std) of Hellinger distance and the Davies-Bouldin Index, as shown in I. The number of clusters is an important factor to consider in clustering analysis. While a larger number of clusters can provide a more detailed representation of the data, it may also lead to overfitting and reduced interpretability. In addition, reducing the number of clusters can simplify the network slicing design and resource allocation, making it more efficient to apply our clustering results to 5G network slicing. Compared to KMeans, our approach efficiently reduced the number of clusters from 25 to 3, taking into account the trade-off between cluster granularity and interpretability, as well as

the efficiency of network slicing.

TABLE I
TRADE-OFF METRICS

| Approaches | Number of Clusters | Variance of Hellinger Distance | Std of Hellinger Distance | Davies-Bouldin Index |
|---|---|---|---|---|
| KMeans | 25 | 0.0063 | 0.0798 | 1.0765 |
| Our Approach | 3 | 0.0100 | 0.1004 | 2.7652 |

The variance and standard deviation of the Hellinger distance are measures of how well the clusters are separated from each other. A lower variance and standard deviation indicate that the clusters are more compact and well-separated. In our evaluation, we observed that our approach achieved a lower variance and standard deviation of the Hellinger distance indicating better separation of the clusters compared to KMeans.

The Davies-Bouldin Index [20] is another widely used metric for evaluating clustering performance as it considers both intra-cluster and inter-cluster distances. A lower Davies-Bouldin Index indicates better clustering performance. However, it is worth noting that achieving a lower Davies-Bouldin Index may come at the expense of reduced interpretability, as it may result in overly complex or fine-grained clusters. In our evaluation, we found that our approach achieved a higher Davies-Bouldin Index compared to KMeans, indicating a trade-off in clustering performance and interpretability. By reducing the number of clusters from 25 to 3, our approach improved the efficiency of network slicing design and resource allocation while still achieving acceptable clustering performance. However, depending on the specific application and use case, a different trade-off between clustering performance and interpretability may be preferred.

### *D. Visualization of Clusters*

Figure 6 displays the two-dimensional PCA plot of the first two principal components. Each data point represents a network flow in the data set. The 2D plot reveals a rough separation of different clusters. However, there is still some overlap among clusters 1, 2, and 3 to some extent. Moving on to the three-dimensional PCA in Figure 7, cluster 1 is well separated from cluster 2, but there is still a small overlap between clusters 1, 2, and 3. Despite the overlap, the PCA results are still acceptable as they provide a visual representation of the data that can aid in understanding the relationships and patterns between the network flows. The three-dimensional PCA plot shows a better separation of clusters, but there is still a small overlap between them. The four-dimensional PCA, which includes the size of the data points as the fourth dimension, did not result in a significant difference among the three clusters. Overall, the PCA results provide valuable insights into the clustering of the network flows despite the presence of some overlap.

### *E. Interpretation of Clusters*

Table II displays the network features for the final three clusters produced by the iterative application of KMeans algorithm on the Hellinger distance. Some of these features

TABLE II
NETWORK FEATURES STATISTIC

| Cluster ID | Number of Flows | Average Bandwidth (Bytes/s) | Average Duration (s) | Burstiness (pkt) |
|---|---|---|---|---|
| Cluster 1 | 270353.00 | 19258.55 | 71563.33 | 24.08 |
| Cluster 2 | 3512893.00 | 17167.94 | 72204.80 | 23.27 |
| Cluster 3 | 1328511.00 | 19959.27 | 72076.13 | 23.70 |

can be directly extracted from the data set, such as the number of flows, while others have been calculated using existing features, including average bandwidth, average duration, and burstiness.

*1) **Cluster 1 - eMBB**:* Based on the information from Table II, compared to cluster 2, cluster 1 has relatively fewer flows, higher average bandwidth, smaller average duration, and slightly higher burstiness. This indicates that cluster 1 is suitable for a 5G network slice that prioritizes high-bandwidth applications with moderate to low latency requirements, such as eMBB (enhanced Mobile Broadband) as defined in Section III. The higher average bandwidth suggests that cluster 1 may consist of flows that require high-speed data transmission. The smaller total and average duration indicate that the flows in cluster 1 may not require sustained connectivity for long periods of time. The slightly higher burstiness suggests that these flows may have occasional bursts of traffic, which could benefit from the high bandwidth available in eMBB. In terms of QoS, cluster 1 requires high throughput and low latency to ensure that the high-bandwidth flows can be transmitted quickly and with minimal delay. The burstiness of the flows may also require low packet loss and low jitter to ensure that bursts of traffic can be handled efficiently and effectively. To support this cluster, a network slice requires sufficient bandwidth to handle the high-bandwidth flows, as well as low latency and low packet loss to meet the QoS requirements. Additionally, the network slice may require sufficient computing resources to process the bursts of traffic efficiently and sufficient storage to buffer the flows during bursts of traffic. One possible example application that uses this type of network service could be a video streaming platform that requires high-bandwidth connections to provide high-quality video playback to users. Another example could be a cloud storage service that requires high-speed data transmission for large file uploads and downloads.

*2) **Cluster 2 -URLLC**:* Compared to cluster 1, cluster 2 has relatively more flows, lower average bandwidth, longer average duration, and lower burstiness. For cluster 2, a slice such as URLLC may be suitable. URLLC aims to provide ultra-reliable and low-latency communications for critical applications. An application example that may use this type of network service is industrial automation, which requires real-time and reliable communication for safety-critical operations. The QoS requirements for URLLC include ultra-low latency, high reliability, and high availability.

*3) **Cluster 3 - mMTC**:* Cluster 3 has the largest bandwidth. This type of traffic is suitable for the massive machine-type communication (mMTC) network slice, which provides high-capacity communication services for a large number of low-

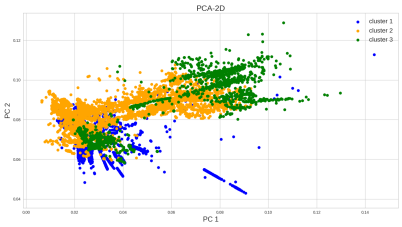


Fig. 6. Two-Dimensional PCA

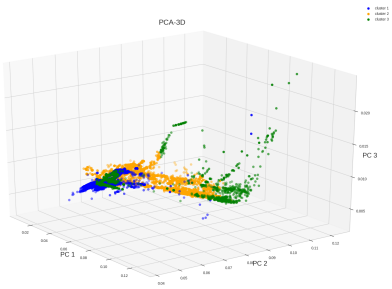


Fig. 7. Three-Dimensional PCA

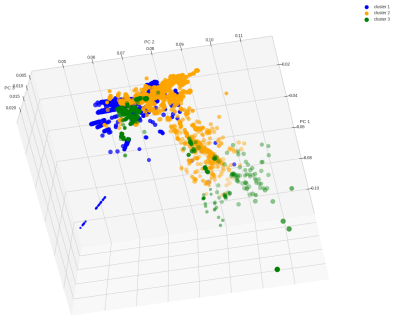

Fig. 8. Four-Dimensional PCA

power and low-rate devices. The QoS requirements for mMTC include high throughput, low power consumption, and high scalability. An example of an application that may use this type of network service is smart cities, which involve a large number of connected devices that transmit small amounts of data.

## VI. Conclusion and Future Work

This paper proposed an unsupervised IoT network traffic clustering approach, which automates determining the appropriate number of 5G network slices for IoT applications. By applying Principal Component Analysis (PCA) and the Hellinger distance-based recursive KMeans algorithm (rK-Means), we can substantially alleviate the problem of high dimensionality while maintaining the original information, simplify the clustering results, and identify the optimal number of clusters. The evaluation results show that the proposed approach can efficiently reduce the high dimensionality of the dataset while effectively visualizing the clustering result in 2D, 3D, and 4D plots. Additionally, the clustering results can provision the network resources in terms of 5G network slices.

Our future work will involve additional research to optimize the approach to handle large-scale IoT traffic datasets. Additionally, the work can be further tested with real-world traffic data and compared with existing traffic clustering methods. Finally, prototyping of the approach as a framework is needed to ascertain its feasibility in integrating with the existing 5G resource management control plane.

## VII. Acknowledgments

This work is supported in part by funding from Siemens Technology. Any opinions, findings, and conclusions or recommendations expressed in this material are of the author(s) and do not necessarily reflect the views of the sponsor.

## References

[1] K. L. Lueth, M. Hasan, S. Sinha, S. Annaswamy, P. Wegner, F. Bruegge, and M. Kulezak, "State of iot—spring 2022," *A comprehensive 114-page report on the current state of the Internet of Things, incl. market update & forecast, latest trends, IT spending update, and more*, 2022.

[2] A. Madhukar and C. Williamson, "A longitudinal study of p2p traffic classification," in *14th IEEE international symposium on modeling, analysis, and simulation*, pp. 179–188, IEEE, 2006.

[3] H. Tahaei, F. Afifi, A. Asemi, F. Zaki, and N. B. Anuar, "The rise of traffic classification in iot networks: A survey," *Journal of Network and Computer Applications*, vol. 154, p. 102538, 2020.

[4] 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.

[5] S. Saini, S. K. Garg, P. P. Singh, A. Ali, and A. Pandey, "Enhancing qos of network traffic based on 5g wireless networking using machine learning approaches," in *International Conference on Computational Intelligence and Smart Communication*, pp. 3–15, Springer, 2022.

[6] O. Aouedi, K. Piamrat, S. Hamma, and J. Perera, "Network traffic analysis using machine learning: an unsupervised approach to understand and slice your network," *Annals of Telecommunications*, vol. 77, no. 5, pp. 297–309, 2022.

[7] 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.

[8] 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.

[9] Z. Min, H. Sun, S. Bao, A. S. Gokhale, and S. S. Gokhale, "A self-adaptive load balancing approach for software-defined networks in iot," in *2021 IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS)*, pp. 11–20, IEEE, 2021.

[10] Z. Min, S. Shekhar, C. Mahmoudi, V. Formicola, S. Gokhale, and A. Gokhale, "Software-defined dynamic 5g network slice management for industrial internet of things," in *2022 IEEE 21st International Symposium on Network Computing and Applications (NCA)*, vol. 21, pp. 251–259, IEEE, 2022.

[11] 3rd Generation Partnership Project (3GPP), "Service requirements for machine-type communications (mtc); stage 2," Tech. Rep. TS 22.261, 3GPP, 3GPP Support Office, 650 Route des Lucioles, F-06921 Sophia-Antipolis Cedex, France, 2016.

[12] 3rd Generation Partnership Project (3GPP), "Technical specification group services and system aspects; study on new services and markets technology enablers," Tech. Rep. TR 22.874, 3GPP, 3GPP Support Office, 650 Route des Lucioles, F-06921 Sophia-Antipolis Cedex, France, 2018.

[13] 3rd Generation Partnership Project (3GPP), "Technical specification group services and system aspects; study on 5g phase 2," Tech. Rep. TR 22.874, 3GPP, 3GPP Support Office, 650 Route des Lucioles, F-06921 Sophia-Antipolis Cedex, France, 2019.

[14] T. SudParis, "Dataset of legitimate iot data," 2022.

[15] B. Hopkins and J. G. Skellam, "A new method for determining the type of distribution of plant individuals," *Annals of Botany*, vol. 18, no. 2, pp. 213–227, 1954.

[16] datanovia.com, "Assessing clustering tendency," 2018.

[17] A. Maćkiewicz and W. Ratajczak, "Principal components analysis (pca)," *Computers & Geosciences*, vol. 19, no. 3, pp. 303–342, 1993.

[18] V. Satopaa, J. Albrecht, D. Irwin, and B. Raghavan, "Finding a "kneedle" in a haystack: Detecting knee points in system behavior," in *2011 31st International Conference on Distributed Computing Systems Workshops*, pp. 166–171, 2011.

[19] E. Hellinger, "Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen.," *Journal für die reine und angewandte Mathematik*, vol. 1909, no. 136, pp. 210–271, 1909.

[20] A. K. Singh, S. Mittal, P. Malhotra, and Y. V. Srivastava, "Clustering evaluation by davies-bouldin index (dbi) in cereal data using k-means," in *2020 Fourth international conference on computing methodologies and communication (ICCMC)*, pp. 306–310, IEEE, 2020.