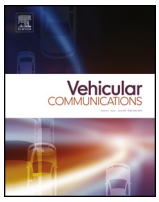
Vehicular Communications 31 (2021) 100370

Contents lists available at ScienceDirect

Vehicular Communications

www.elsevier.com/locate/vehcom



# A traffic data clustering framework based on fog computing for VANETs

M.L.M. Peixoto [a,b,*], A.H.O. Maia [a], E. Mota [a], E. Rangel [a], D.G. Costa [c], D. Turgut [d], L.A. Villas [b]

[a] Computer Science Department, Federal University of Bahia (UFBA), Salvador, Bahia, Brazil
[b] Institute of Computing, University of Campinas (UNICAMP), Campinas, Sao Paulo, Brazil
[c] Departament of Technology, State University of Feira de Santana, Feira de Santana, Brazil
[d] Departament of Computer Science, University of Central Florida, Orlando, United States of America

ARTICLE INFO

*Article history:*
Received 1 December 2020
Received in revised form 2 April 2021
Accepted 30 April 2021
Available online 11 May 2021

*Keywords:*
Data reduction
Data stream
Data clustering
VANET
Fog computing

ABSTRACT

Vehicular Ad-hoc Networks (VANETs) are based on vehicle to infrastructure communications in which the vehicles periodically broadcast information to update a Road Side Unit (RSU). The traffic data is forwarded from all RSUs to a cloud or a central server for global analysis and detection of congestion levels on the roads. However, communication costs may considerably increase when a large amount of data is transmitted to such cloud-like service providers. In this paper, we propose a data clustering framework to perform traffic information reduction at the edge of vehicular networks by exploiting fog computing. The proposed data clustering framework defines two methods for the reduction of the traffic data stream: (i) Baseline method, which is an ordinary traffic congestion detection approach, and (ii) two adapted clustering methods for a data stream; namely, the Ordering Points to Identify the Clustering Structure (OPTICS) and the Density-Based Spatial Clustering of Applications with Noise (DBSCAN). The results have shown that the proposed traffic data framework using clustering methods is accurate even when the vehicular traffic condition is highly congested, potentially reducing the communication costs and bringing significant results for the development of VANETs.



## 1. Introduction

Nowadays, traffic congestion is one of the biggest challenges in major cities worldwide. People stuck in their vehicles for hours during traffic jams eventually leading to economic and time losses and excessive air pollution in urban areas. According to [27], drivers lose an average of 97 hours a year due to traffic congestion in the United States, resulting in $87 billion annually in money-wasting. Therefore, better traffic management and efficient urban mobility have been primary objectives when fostering the development of smart cities.

Besides initiatives to promote alternative transportation to relieve heavy traffic and pollution in modern cities, the adoption of Vehicular Ad-hoc Networks (VANETs) can be an essential resource for more efficient and safer mobility. In fact, alongside other integrated services such as intelligent traffic lighting and urban-related emergency management, VANETs are expected to support the creation of Intelligent Transportation Systems (ITS) [38]. In this context, new developments are expected to profoundly transform the automotive industry in this century, significantly supporting new solutions for the biggest challenges of our time [11]. In this scenario, VANETs have been considered to provide innovative solutions for smarter transportation, but many challenges remain [21].

Although VANETs can bring significant results to modern cities, an important concern is a considerable increase in the volume of data produced by technological advances in ITS solutions. The International Data Corporation (IDC) has mapped some expectations regarding the number of connected devices, reporting an increase in the order of 50 billion to 1 trillion devices in 2020. Hence, that year accounts for 110 million connected cars with 5.5 billion sensors and 1.2 million connected houses with a total of 200 million sensors [16] [13]. This complex scenario will inherently put pressure on the operation and management of intelligent transportation systems, which have demanded optimizations to the way VANETs exchange data.

Roughly speaking, VANETs will provide information that can support better automated decisions by the vehicles, which may be exploited to avoid traffic and reduce the probability of accidents. As a result, different approaches have emerged for communication among vehicles, with different particularities [18]. Due to the complexity resulting from inter-vehicle communications, which may be

* Corresponding author at: Computer Science Department, Federal University of Bahia (UFBA), Salvador, Bahia, Brazil.
*E-mail address:* maycon.leone@ufba.br (M.L.M. Peixoto).

https://doi.org/10.1016/j.vehcom.2021.100370
2214-2096/

hard to achieve due to the inherent movement patterns in such scenarios, it is common to employ some fixed communication infrastructure to support data transmissions. A Road Side Unit (RSU) has been frequently employed as a unique device designed to act as a gateway, receiving and providing information to moving vehicles. However, although potentially benefiting when supporting communications among vehicles, high transmission demands may jeopardize the entire efficiency of the system and the performance of the employed RSUs, demanding proper planning and eventual optimizations.

VANETs have been leveraged to support the construction of Traffic Congestion Detection Systems (TCDS) [8]. TCDS have exploited the RSUs and direct communications among vehicles to detect traffic congestions and act to cease their causes, potentially alerting about accidents and slow-movement zones. For such systems, data is the most critical asset, which may be retrieved not only from vehicles and roads but also from any complementary systems such as social media and open databases maintained by governmental agencies [9,33]. Regardless of the case, it is natural to expect that a large amount of data is transmitted and processed to achieve efficient traffic detection, alerting, and mitigation.

Depending on the characteristics of the deployed VANET, such as the number of connected vehicles, the traffic data transmitted by the RSUs to the cloud may be massive, adding a high communication cost for the entire system. Since it may become critical, especially considering the expected increase in the number of active VANETs in modern cities, this article proposes a new traffic data clustering framework to process traffic data streams.

The proposed framework leverages the fog computing paradigm when receiving traffic information from RSUs, reducing the amount of data forwarded to the cloud. Our proposed framework defines two different approaches: (i) a traffic congestion detection approach without data reduction (Baseline method), and (ii) a traffic congestion detection approach based on two clustering methods: Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and Ordering Points to Identify the Clustering Structure (OPTICS). These methods are appropriately specified and simulated in different conditions, presenting promising results. The performance analysis shows that the clustering methods achieve a reduction in the amount of transmitted data in different ways. We believe that such clustering methods, which were not described before in the literature to the best of our knowledge, can potentially support important developments and research in the considered areas.

The remainder of this article is structured as follows. Section 2 discusses the related works within fog computing and vehicular ad hoc networks. The system model and problem formulation are presented in Section 3. Section 4 describes the conducted experiments and the results. Finally, Section 5 concludes the paper.

## 2. Related works

Various models of traffic congestion detection [40,7,3,2], with generic solutions, can be found in the literature. However, some of those proposals do not consider the issues related to the impact of traffic data generated on a network link between the RSUs and cloud-like services.

In order to reduce the communication delay between the vehicles and the network server, the study developed by Wahid et al. [40] used a congestion detection method that considers that all vehicles on the network operate as sensor nodes. The vehicles send data to the server and receive traffic information to calculate the road congestion. In the following, responses can be transmitted to all vehicles and road users. The selection of data to be sent is performed according to a transmission policy, which does not use a criteria of randomization or the cost of information. The defined policy considers that all vehicles have the same opportunity to transmit data to the server.

Chen and Li [7] proposed a traffic control that uses an estimation of the traffic flow phase derived from data obtained between VANET vehicles. That estimation uses a prediction algorithm based on Fuzzy logic, making it possible to classify the traffic flow in three states: free flow, synchronized flow, and wide movement congestion flow. The SUMO, MOVE, and NS2 frameworks were used for VANET simulation and data collection.

Bauza et al. [3,2] proposed the use of the CoTEC (COperative Traffic congestion detECtion) technique, which seeks to optimize the detection of vehicle traffic without the installation of infrastructure sensors, using only V2V communication. To detect congestion conditions, CoTEC uses a Fuzzy logic mechanism based on signal messages received from neighboring vehicles. Bauza et al. [3] conducted and evaluated a congestion simulation (traffic simulator - SUMO) with CoTEC in a highway environment. In that work, the speed and location parameters were analyzed. However, the employed communicator range was 300 meters, transmitting messages periodically at 1 Hz, signaling the vehicle's location and speed. For the highest precision of the data and adequate measurement of the traffic intensity, it is necessary to compute an optimal amount of re-transmitter vehicles. The higher the number of transmitted messages, the higher the congestion detection quality and the lower the RMSE (Root Mean Square Error).

Another aspect that needs to be considered is the problem of data collection. The study proposed by He and Zhang [15] investigated the rapid evolution of this problem in VANETs, aiming to minimize network communication overheads by choosing to transport or forward data packets based on the current VANET traffic information. To this end, they formulated data collection as a scheduling optimization problem, using an ideal dynamic programming algorithm and heuristics based on genetic algorithms for small and large scale data collection, respectively. To reduce the transmission cost, the scheduling policy considers forwarding only data based on some parameters, including the delay in delivering the message from the vehicle to the base station.

In the works presented in [7,3] and [2], the Fuzzy logic was used to perform the identification/prediction of congestion. Already related to the communication cost of VANETs, the study [15] proposed a data collection solution using a message scheduling policy, aiming to reduce the cost of transmitting these messages to a base station.

Data aggregation and reduction are characteristics present in the context of VANETs. Using the data recovery property in the message recovery signature (MRS), a real-time traffic aggregation scheme was proposed by Shen et al. [32]. In this model, data security features are taken into accounts, such as resistance to attacks, preservation of privacy, and data confidentiality. The authors considered authentication in the transmission of messages, performing the validation and verification of the vehicles' signatures in the network to guarantee security and prevent data from being exposed to possible attackers. They also proposed the application of batch operations, increasing the efficiency in multiple vehicle signature verification.

Guedes and Campos [12], the authors presented a data aggregation scheme, which seeks to reduce the amount of redundant data in a VANET to mitigate problems related to scalability. The proposed scheme is based on aggregated data, following fixed-length paths, which are used as a parameter for the decision mechanism. In order to validate the application of the scheme, NCTUns, a vehicle network simulator was employed.

In this same perspective, the work presented by Kaur and Kad [19] implemented a data aggregation model that uses the Ad-hoc on Demand Distance Vector Reliability (AODV-R) protocol based on ant colony optimization. The used protocol is reactive; that is, a

**Table 1**
Comparison of related works.

| Authors | Application domain | Application benefit* | Execution environment | Data stream |
|---|---|---|---|---|
| **This work** | TCDS | Communication cost reduction; Traffic data reduction | Fog and Cloud | yes |
| Wahid *et al.* [40] | TCDS | Server communication cost reduction | - | no |
| Chen and Li [7] | TCDS | Deployment cost reduction | - | no |
| Bauza *et al.* [3] | TCDS | Minimize the network communication overhead | - | no |
| Bauza and Gonzalez [2] | TCDS | Communication overhead reduction | - | no |
| He and Zhang [15] | Data collection | Communication cost reduction; Data reduction | - | no |
| Shen *et al.* [32] | Data aggregation | Secure messaging | cloud | no |
| Guedes and Campos [12] | Data aggregation | Data reduction | - | no |
| Kaur and Kad [19] | Data clustering | Communication cost reduction | - | no |
| Aadil *et al.* [1] | Data clustering | Communication cost reduction; Data reduction | - | no |
| Keramatian *et al.* [20] | Data clustering | Data reduction | Fog | yes |
| Havers *et al.* [14] | Data clustering | Data reduction | - | yes |
| Najdataei *et al.* [23] | Data clustering | Data reduction | - | yes |

link between the sending and receiving nodes is established only when necessary. To improve the process of choosing the shortest path, the authors proposed the use of the Ant Colony optimization algorithm, replacing the Djikstra and hop by hop algorithms implemented in the AODV-R protocol. However, that work did not clarify the conceptual level where the proposal is located at, whether in fog or the cloud.

A clustering algorithm based on ant colony optimization for vehicle networks denominated CACONET, proposed by Aadil et al. [1], aimed to optimize the formation of clusters to obtain a robust communication between the components of the VANETs. The central idea is to minimize the number of cluster heads in the network to reduce the communication cost. To validate the use of the algorithm, the authors compared other optimization techniques based on minimizing the number of cluster heads, namely Multi-objective Particle Swarm Optimization (MOPSO) and Comprehensive Learning Particle Swarm Optimization (CLPSO).

The study proposed by Havers et al. [14] proposed a tool to collect data on vehicular networks, in addition to grouping them based on distance. For that, they used a technique of linear approximation by parts (PLA) with the purpose of compacting the volume of collected data (drastically reducing it), avoiding the collection of raw data from vehicles that depend on an approach based on streaming limited by error. On the other hand, this approach performs the grouping of data online at the moment when they are being recovered from the devices. Thus, as we propose in this work, our approach uses continuous data instead of working with static databases.

Keramatian et al. [20] proposed a method of combining distributed multi-stage approximate clustering in order to detect and locate obstacles, efficiently exploiting the decentralized processing capacity that is available at the edge nodes, also avoiding network saturation of communication. To this end, the authors made use of point clouds from various LIDAR sensors, employing an efficient summarizing method. That method identifies the local clusters, then transforms the information collected from each object into a continuously computed summary of data. The authors evaluated the proposed method both in a simulated environment and in an IoT test bank, which contained representative fog/edge devices.

The work elaborated by Najdataei et al. [23] proposed applications of big data analysis, using efficient methods capable of processing the raw data obtained from high-rate flows. For that, two approaches were used, LISCO and LISCO-P, which were compared to the PCL E. LISCO approach represents a streaming approach to process LiDAR points while they are being collected. Its parallel counterpart, LISCO-P explores the parallelism of Lisco's processing pipeline in an architecture-independent manner.

As discussed, TCDS [40,7,3,2] can generate a large amount of data, leading to a significant increase in the cost of communication [15,19,1] in the network link between fog and cloud. Thus, for traffic congestion in a continuous data stream environment, aiming to minimize the traffic data [19] and consequently the network communication cost, the Table 1 compares such aspects with the works covered in this section.

Most of the listed works focused on TCDS do not consider any reduction of traffic data to mitigate the impact of high data generation on the network link between RSU/fog and the cloud. Actually, to the best of our knowledge, no work has been found in the literature providing data reduction in the TCDS problem for a continuous data stream environment since static databases have been mostly exploited by such approaches. Therefore, in order to cope with the limitations of the previous works, we propose a framework to detect road congestion that works incrementally and in an adaptive way, clustering the traffic data stream in order to reduce it, potentially decreasing the communication costs when avoiding to handle with large amounts of offline data.

## 3. Proposed approach

After defining the considered problem scope and the state-of-the-art on related research areas, the proposed approach is described in this Section. The fundamental concepts are described, as well as the required procedures. In order to support such discussions, Table 2 summarizes the adopted notations in next subsections.

### 3.1. Target scenario

Smart cities are subject to huge amounts of data, flowing through different applications in an urban scenario. Among them, TCDS will also produce massive data to be processed. This is, in fact, the target scenario to be addressed by the proposed framework, as depicted in Fig. 1.

We assume a smart city environment with active urban mobility. In this environment, we have RSU/fog continuously collecting packets from vehicles within its coverage area through IEEE 802.11p. Orange thick arrows indicate the positional data flow from moving vehicles towards the RSU/fog. RSU/fog layer is responsible for routing the traffic data to the LTE/5G network. However, when the traffic data is clustered in the fog, considerably less data is sent on the router's outgoing interface that connects the LTE/5G network.

### 3.2. Fundamental definitions

Considering the defined problem scope, it is assumed timestamped positional updates from moving vehicles, which are transmitted in a streaming fashion. Therefore, let $D(B,t)$ be the set of continuous traffic data $(B)$ arriving in the RSU/fog at time $t$, $\tau$ be

**Table 2**
Defined notations.

| Notation | Description |
|---|---|
| $G$ | Directed and weighted graph |
| $\tau$ | Current time |
| $NB$ | Network bandwidth $\rightarrow (SB/T)$ |
| $LB$ | Represents the traffic data: sum of all packet size |
| $SB$ | Traffic data (bps) $\rightarrow (LB + \lambda_B)$ |
| $\lambda_B$ | Average arrival rate |
| $T$ | Channel bandwidth (bps) |
| $a$ | Accuracy obtained |
| $c$ | Minimum accuracy acceptable |
| $V$ | Set of intersections $\rightarrow \{v_1, v_2, ..., v_{\|V\|}\}$ |
| $E$ | Set of road segments $\rightarrow \{e_1, e_2, ..., e_{\|E\|}\}$ |
| $N$ | Set of vehicles $\rightarrow \{n_1, n_2, ..., n_{\|N\|}\}$ |
| $U$ | Set of roads segments $\rightarrow \{e_1, e_2, ..., e_{\|e\|}\}$ |
| $W$ | Set of weights $\rightarrow \{(w_1, t_1), ..., (w_2, t_2), ..., (w_{\|W\|}, t_{\|W\|})\}$ |
| $v_o$ | Source intersection |
| $v_p$ | Destination intersection |
| $D$ | Traffic data stream $\rightarrow \{(B_1, t_1), (B_2, t_2), ..., (B_{\|B\|}, t_{\|t\|})\}$ |
| $B$ | Beacon message $\rightarrow (\tau, P^{x,y}, id)$ |
| $t$ | Timestamp |
| $P^{x,y}$ | Position of a vehicle on the road |
| $id$ | Unique identification code of a vehicle |
| $x$ | Latitude |
| $y$ | Longitude |
| $\Delta t$ | Interval of time |
| $\Omega$ | Traffic data reduction factor achieved |
| $R$ | Traffic reduction achieved |
| $s$ | Speed |
| $s^{avg}$ | Average speed in a road segment |
| $s^{max}$ | Maximum allowed speed in a road segment |
| $d$ | Traffic density |
| $MinPts$ | Minimum number of vehicles in a cluster |
| $\varepsilon$ | Neighborhood radius DBSCAN parameter |
| $w$ | Traffic flow weight in a road segment |
| $\lambda$ | Set of points spread across in each cluster |

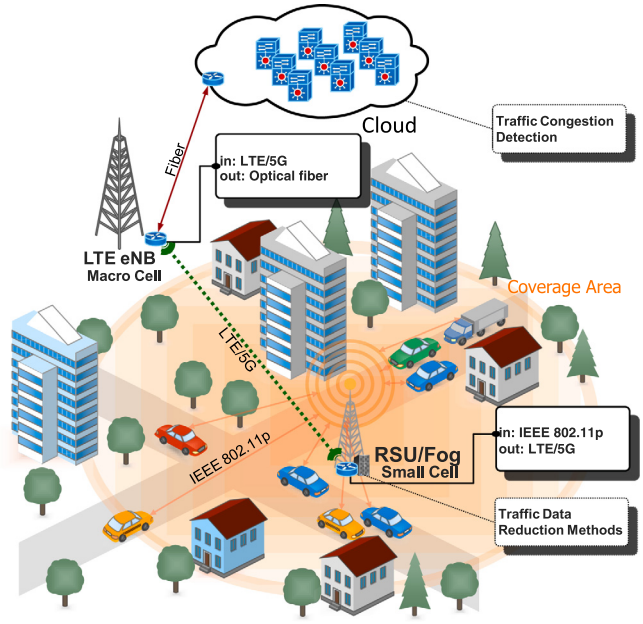



**Fig. 1.** Data Flow Framework Overview for Traffic Congestion Detection System. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

the current time, and 0 be the starting time. Thus, for each new traffic data $B_i$ obtained in the RSU, it is attached to the tuples newly arrived in the set. Thus, the continuous traffic data set $B$ at time $\tau$ is given by $D(B, \tau) = \cup_{t=1}^{\tau}(D(B, t) - D(B, t-1)) \cup D(B, 0)$ which is characterized by a spatial trajectory. A spatial trajectory is given by the movement of a vehicle in geographic spaces, usually represented by a series of chronologically ordered points, for example, $p_1 \rightarrow p_2 \rightarrow \ldots \rightarrow p_n$, where each point $p$ consists of a *timestamp* $(t)$ and a pair of geographic coordinates $(x, y)$. The $p$ trajectory is considered as a discrete sample but the movement of a set of points is a huge amount of sample, so a trajectory is considered as data flow. The spatial trajectory was applied by the [5,34,6,17,22] to identify traffic congestion over the time for several roads.

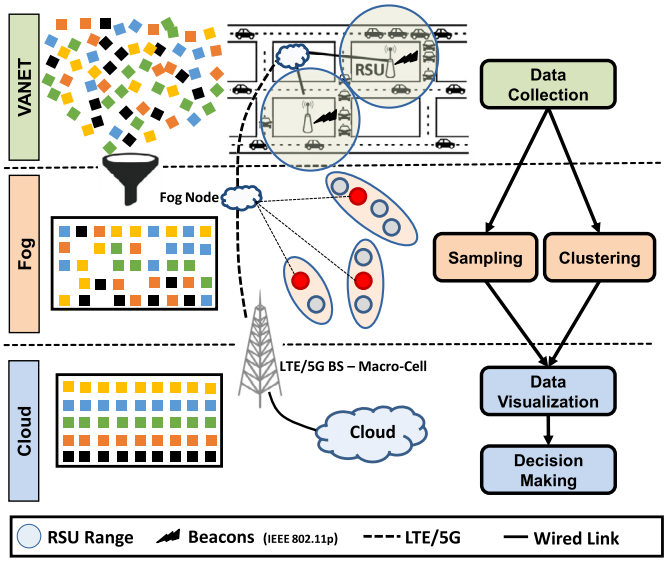



**Fig. 2.** Framework Data Pipeline.

This continuous traffic data flow generated over time by vehicles is usually large, leading to network congestion, packet losses, higher communication expenses, waste of bandwidth, and increased delay. For that, $NB = S_B/T$ gives the use of network bandwidth, with $S_B$ representing the traffic data (bps), for $T$ as the channel bandwidth (bps). Moreover, $S_B$ is also given by $(L_B + \lambda_B)$, where $L_B$ is the sum of all packet size and $\lambda_B$ is the average arrival rate. The greater the number of vehicles, the greater is $NB$. One way to approach this problem is to minimize $L_B$ subject to the constraint $a \geq c$, where $a$ is the accuracy obtained, and $c$ is the minimum accuracy acceptable to detect traffic congestion.

### 3.3. Streaming data pipeline

The proposed framework uses a pipeline scheme based on a set of connected components, allowing the continuous processing of the traffic data flow for detection of the congestion level on the roads. Thus, it is composed of a series of streaming data processing stages in which each step delivers an output that is the input to the next one. However, our framework can process in parallel some independent steps. For example, while the clustering method is being applied, the data gathering process remains in operation. Fig. 2 shows the pipeline of our data reduction framework.

We highlight the main framework pipeline components: data collection, sampling, clustering, data visualization, and decision-making, which are presented as follows:

- **Data Collection**: The data collection phase occurs when vehicles periodically send a beacon message (IEEE 802.11p) to the nearest RSU. VANETs use Wireless Access in Vehicular Environments (WAVE), a service provided by the IEEE 802.11p standard. The main infrastructure consists of RSUs engaging vehicles via WAVE to collect traffic information such as time, speed, and vehicle location. WAVE is dedicated to short-distance communication and uses beacon packets to exchange

**Table 3**
Beacon Payload.

| SenderID | TabSize | VehiID | PcktID | Lat | Lon | Speed | Head | Time | ... |
|---|---|---|---|---|---|---|---|---|---|
| 1B | 1B | 1B | 2B | 4B | 4B | 4B | 4B | 8B | |

information between vehicles and infrastructure. A beacon message, Table 3, is an IEEE 802.11p 2-layer periodic message of situational information used by RSUs and vehicles to exchange knowledge in the Control Channel (CCH). Such beacon has the situational information with 27 bytes but can reach approximately 100 bytes [28]. Depending on the beacon rate, the channel load can increase in scenarios of a high number of vehicles, overcharging the network. When CCH-network is overloaded, beacons packets are lost, causing unpredictable behavior [6,31,35];

- **Sampling**: There is a simple sampling approach designed in the framework, which does not take the data itself into account, e.g., when using a rate of [1:2], the second given traffic data is always routed from each RSU to the fog;
- **Clustering**: The framework uses a sliding time window to contain the traffic data stream that is processed by the Clustering approach. Within this context, we use two clustering algorithms: **a)** an adapted Density-Based Spatial Clustering of Applications with Noise (Adapted-DBSCAN) (see Algorithm 2) and **b)** the Ordering Points To Identify the Clustering Structure (Adapted-OPTICS) (see Algorithm 3). Both algorithms are responsible for implementing different grouping techniques that will be compared herein. DBSCAN and OPTICS are used due to their density-based way of conducting the cluster. Besides, DBSCAN and OPTICS do not require the number of clusters (*k-number*) a priori, which is essential for the dynamic environment of a vehicular network;
- **Data Visualization**: In the Data Visualization stage, a Multidimensional Projections is enabled to map (or project) traffic data into lower dimensional embedding space, typically two or three-dimensional. Formally, given a VANET environment containing $n$-dimensional points, a dataset $\mathbf{D}$ is denoted as $D^n = \{\mathbf{p}_i^{x,y} \in \mathbb{R}^n\}_{1 \leq i \leq N}$, which can be seen as a function $f : \mathbb{R}^n \times \rho \rightarrow \mathbb{R}^m$ mapping each point $\mathbf{p}_i^{x,y} \in D^n$ to a point $\mathbf{q}_i \in D^m$, where $m$ is the projected (visual) space and $\rho$ denotes the *parameter space* of $f$;
- **Decision-Making**: The results generated by data reduction techniques are sent to the cloud layer (LTE/5G) in order to support the desired traffic congestion detection. Therefore, it is possible to use such information in the Decision-Making Process, providing better vehicle routes or re-routing.

Next, we describe the three main layers of this environment and how they relate to our framework.

### 3.3.1. VANET layer

The VANET Layer offers a set of weighted graphs using the road information, which is based on spatial and temporal analysis. The definition of the road network is described as follows:

**Definition 1.** Let a VANET be modeled as a directed and weighted graph $G = (V, E)$, where $V = \{v_1, v_2, ..., v_{|V|}\}$ is the set of intersections and $E = \{e_1, e_2, ..., e_{|E|}\}$ is the set of road segments connecting the intersections. Also, let $N = \{n_1, n_2, ..., n_{|N|}\}$ represent the vehicles on the road. When a vehicle $n_i$, where i represents the $i^{th}$ vehicle $\in N$, is driving from a source intersection $v_o$ to a destination intersection $v_p$, an ordered set of road segments in the route is defined as $U_{o,p} = \{e_1, e_2, ..., e_{|e|}\}$, where $|e|$ is number of road segments in the route.

The TCDS process starts in the urban mobility environment and involves architecture and communication capacity between its elements. In this work, this layer is responsible for producing a massive continuous data flow containing essential information about traffic state identification at a specific moment. In a VANET environment, traffic data is spread out in a beacon way. Each beacon must be encapsulated, forming a metadata set that describes the current state of a specific vehicle. The beacons sent from different sources are gathered to form a landscape representing the traffic state in the scenario.

A Road Side Unit (RSU) is a processing unit placed along the road to receive data emitted for certain vehicles. Thus, a RSU provides part of the connectivity features in a VANET scenario, the V2I communication. The V2I communication consists of connectivity between vehicles and infrastructure available in a particular highway stretch. Each available RSU along the structure has a certain cover radius, and once under this radius, beacons emitted by vehicles must be recovered by the RSU. On the other hand, beacons outside the RSU covering area must be discarded. The high frequency in which beacons are spread out in the vehicular network scenario is required because vehicles, in general, do not know if they are in a coverage area, so they must be recognized as soon as possible. However, this frequency can represent a risk of network overload. The data reduction approach proposed in this article tries to mitigate this effect.

There are several RSUs typically attached along roads or intersections on the communication aspects, using beacon packets to exchange data with vehicles. We assume a message exchange frequency of 1 Hz, the minimum to detect a traffic jam according to [31]. RSU was structured under protocol 802.11p, and we use a standard header as a form to standardize the exchange of information among layers. IEEE 802.11p is based on dedicated short-range communications (DSRC) radio technology to exchange data between vehicles and RSUs.

### 3.3.2. Fog computing layer

The fog layer deploys two different approaches for sampling (Baseline) and clustering (Adapted-DBSCAN and Adapted-OPTICS). The Baseline was built in the fog to create similar conditions when comparing the performance with clustering approaches. Actually, Baseline is a simple and straightforward algorithm (Algorithm 1). Merely dealing with traffic data by road segment without concern about data reduction issues, the baseline algorithm forwards all traffic data collected from the cloud's vehicular environment.

The Baseline Algorithm 1 goes through all segments of the road (lines 1 to 11) and performs the steps to follow. In line 1, all information in the $e_i$ segment is initialized to 0 (to ensure that the path information is empty). Lines 3 to 10 run through all the new $P$ points and check if this $P$ point has already been visited (line 5). If true, a $P_t$ counter is incremented to signal that point has been visited $P_t$ times. If false, the point $P$ is marked as visited, and its information is assigned to the segment $e_i$.

As the Baseline Algorithm operates in the fog, sending all data received from the urban mobility layer to the cloud allows us to perform comparisons with data reduction methods concerning traffic classification accuracy. However, the main goal of fog is to reduce traffic data, acting as the receiver of all data streams from the mobility environment, and providing data reduction strategies based on the density and distribution of the vehicles along the roads. As aforementioned, the data reduction process implemented

**Algorithm 1:** Baseline Algorithm.

**Require:** Set of stream points ($D$) containing all positional vehicle information in the time interval $\Delta t$
**Ensure:** Set of stream points ($D$) organized by road segment $e_i$

```
1: Initialize visited = 1
2: for all road segment e_i ∈ E do
3:     Initialize road segment information e_i = 0
4:     for all new point arrived P ∈ D do
5:         if P is visited then
6:             P = P_{i+1}
7:         else
8:             P = visited
9:             e_i = (τ, P^{x,y}, id)
10:        end if
11:    end for
12: end for
13: return (E)
```

in this layer aims, among other things, to reduce the communication cost in the LTE/5G link, besides mitigating latency issues and network overloading. The definition of traffic reduction is presented as follows:

**Definition 2.** Let a vehicle $N$ be modeled as $N = (n_1, n_2, ..., n_i)$, periodically sending Traffic Data $(B = \tau, id, P^{x,y})$ to the nearest RSU/fog, where $\tau$ represents the data timestamp, $id$ is the vehicle identification number, and $P^{x,y}$ gives the location of the vehicle $n_i$ for the latitude $x$ and the longitude $y$. Besides, the speed $s(i)$ of each vehicle $n_i$ is automatically obtained by the difference between two consecutive position points $(P_{x_i,y_i} - P_{x_{i-1},y_{i-1}})$. Hence, the total amount of traffic data stream $\sum_{i=0}^{|N|}(\sum_{j=0}^{|TD|} TD_j^i)$ is clustered to $R = (\sum_{i=0}^{|N|}(\sum_{j=0}^{|TD|} TD_j^i))/\Omega \mid (\forall\, n \in N : \Omega \geq 1)$, where $\Omega$ is the data reduction factor and $R$ is the traffic reduction achieved.

As mentioned earlier, the data reduction process proposed in this article takes advantage of the density-based algorithm Density-Based Spatial Clustering of Applications with Noise (DBSCAN), developed by [10]. Our contribution lies in the technical and conceptual adaptation of the DBSCAN algorithm (Algorithm 2) to use it in a continuous data flow environment. This adaptation enables the DBSCAN to efficiently extract *"on-the-fly"* the traffic condition insights from the traffic data stream clustering.

In order to provide a comparative analysis of Adapted-DBSCAN with another clustering algorithm, we also implemented the Adapted-OPTICS algorithm. The OPTICS implementation follows the same adaptations made in DBSCAN, creating a fair environment to measure and analyze the results. This comparative analysis can be useful to validate our proposed framework.

The Adapted-DBSCAN Algorithm 2 initializes the $k$ cluster set with 0 (line 1) and runs through all the $P$ points present in the $D$ set (lines 2 to 27). As the Baseline, it checks if the point has already been visited (line 3) and increments its counter (line 4). If it has not been visited yet, it is marked as visited (line 6) and continues processing. In line 8, all neighbors $Nbrs$ of $P$ are retrieved, and then it is checked whether the number of neighbors exceeds the minimum value $MinPts$ passed as a parameter. If larger, $P$ is marked as a node. Otherwise, an incrementing variable $k_i$ (line 12) responsible for counting the number of clusters $k$ is incremented $(k_{i+1})$ and the value in $P$ is assigned to this cluster $k$ (line 13). Then, all $P'$ points present in $Nbrs$ are processed (lines 14 to 25). It is verified if that point has not been visited (line 15), and if true, $P'$ is marked as visited on line 16. Soon afterward, all neighbors $Nbrs'$ of $P'$ are recovered (line 17), and it is checked if the total number of neighbors $Nbrs'$ is equal to or greater than $MinPts$, that is, there is a minimum number of neighboring nodes in $Nbrs'$ to be formed a cluster (line 18). If this is true, the set of neighbors $Nbrs$ merges itself with the set $Nbrs'$ (line 19). In line 22, it is verified that $P'$ is not present in the set of clusters $k$ and adds it to this set true case. In the end, line 29, all clusters are processed and added to the set of clusters $k'$, adding their centroid and some neighbors (defined by a percentage $\lambda$) in the representation of the object. This $k'$ is the return of the Adapted-DBSCAN algorithm.

**Algorithm 2:** Adapted-DBSCAN.

**Require:** Set of stream points ($D$) containing all positional vehicle information in the time interval $\Delta t$, as well as $MinPts$ and NeighborRadius $\varepsilon$
**Ensure:** Set of Clusters ($k$ and $k'$) assignment for each $\Delta t$

```
1: Initialize clusters set k = 0 and visited = 1
2: for all new point P ∈ D do
3:     if P is visited then
4:         P = P_{i+1}
5:     else
6:         P = visited
7:     end if
8:     Nbrs = all P ∈ ϵ.neighborhood
9:     if |Nbrs| < MinPts then
10:        P = noise
11:    else
12:        k = k_{i+1}
13:        k = P
14:        for all P' ∈ Nbrs do
15:            if P' is not visited then
16:                P' = visited
17:                Nbrs' = all P' ∈ ϵ.neighborhood
18:                if |Nbrs'| ≥ MinPts then
19:                    Nbrs = Nbrs ⊕ Nbrs'
20:                end if
21:            end if
22:            if P' ∉ k ← (k = 0, ..., k = n) then
23:                k = P'
24:            end if
25:        end for
26:    end if
27: end for
28: for all k ← (k = 0, ..., k = n) do
29:    k'.append(k.centroid ⊕ k.Nbrs(λ))
30: end for
31: return (k')
```

It is worth highlighting that a suitable $\epsilon$ can impact the quality and accuracy of the Adapted-DBSCAN algorithm. In [26], the authors make an exploratory analysis in order to measure this impact. The results showed that with low $\epsilon$ values, the data reduction rate increases but with low accuracy. On the other hand, when the $\epsilon$ value increases, the accuracy rate also increases up to a threshold identified by the knee method [30].

DBSCAN requires a representative dataset to discover new clusters. However, in a continuous data flow environment, this representativeness may be lost since, with each interaction, a new data flow is arriving at processing. Thus, to handle this behavior, we use a sliding window which works as a mechanism to adjust flexible limits in the unlimited flow, to seek a finite, but always a variable set of tuples, which is considered as a temporary relationship, as explained at [37] and [25]. In this way, the timestamp values of the streaming tuples are checked for inclusion in a pre-specified time interval, producing an approximate response to a query in the data stream, which allows the Adapted-DBSCAN to analyze parts of the recent data instead of looking at all the history of the data stream. Therefore, the sliding window gathers the coming data in a pre-specified time interval $(\Delta t)$, and when the $elapsed\_time > window\_time\_limit$, the accumulated data are submitted to the processing by the Adapted-DBSCAN algorithm.

After providing traffic data stream clustering, the first reduction process is offered automatically by discarding everyday noises resulting from the Adapted-DBSCAN algorithm. The next step is to

reduce the amount of traffic data sent to the cloud. Thus, based on the vehicles' density distribution on the road for each sliding window, each created cluster is transformed into a new smaller cluster but maintaining the same similarity representation as the original one. However, before reducing the number of elements within each cluster, we choose the most representative point within each cluster to act as a centroid element. In the ordinary DBSCAN, there is no notion of a centroid, and we built this option to provide a central element inside the cluster $k$ with the minimum distance from one another. The goal is to choose a spread set of points ($\lambda$) in each cluster with the same similarity (position and speed) from the centroid. As a result, we sent to the cloud a subset representing more relevant elements for each cluster. As aforementioned, this data selection strategy was also implanted in Adapted-OPTICS algorithms in order to create a fair comparative analysis environment.

**Algorithm 3:** Adapted-OPTICS.

**Require:** Set of stream points ($D$) containing all positional vehicle information in the time interval $\Delta t$, as well as $MinPts$ and NeighborRadius $\varepsilon$

```
1: Initialize clusters k = Unprocessed
2: for all points P ∈ D do
3:     pt.reachable_dist = undefined
4: end for
5: for all unprocessed point P ∈ D do
6:     Nbrs = getNbrs(P, ε)
7:     mark P as processed
8:     k'.append(P)
9:     if core_dist(P, ε, MinPts) != undefined then
10:        Seeds = empty priority queue
11:        update(Nbrs, P, Seeds, ε, MinPts)
12:        for all q ∈ Seeds do
13:            Nbrs' = getNbrs(q, ε)
14:            mark q as processed
15:            k'.append(q)
16:            if core_dist(q, ε, MinPts) != undefined then
17:                update(Nbrs', q, Seeds, ε, MinPts)
18:            end if
19:        end for
20:    end if
21: end for
22: return (k')
```

The first step of the Adapted-OPTICS Algorithm 3 is to initialize the distance of all points in $D$ with *undefined* (lines 1 to 3). Then, for each point not yet processed, the following tasks will be performed: In line 5, all neighbors of $P$ are captured given a distance defined by $\varepsilon$, in line 6, the point is marked as processed, and in line 7 it is added to an ordered list $k'$ (which is the output of the algorithm). Soon after (line 8), it is verified that the point $pt$ is not a noise, that is, if the core distance of $P$ is not *undefined*. If true, a priority queue is initialized and assigned to *Seeds* (line 9). In line 10, the update function (detailed in the next paragraph), responsible for updating the reachable distance of each neighbor from $P$, is invoked, passing as parameters the neighbors of $P$ ($Nbrs$), $P$ itself, the queue *Seeds* previously initialized, $\varepsilon$ and $MinPts$. After updating neighboring points and filling out the *Seeds* queue, each item within this queue is processed (lines 11 to 18). The neighbors for each $q$ item in *Seeds* are retrieved on line 12, marked as processed (line 13), and added to the ordered list $k'$ (line 14). In line 15, it is checked whether the core distance of $q$ is not also noise. If so, the update function is again invoked to process $q$ and its neighbors $Nbrs'$. In the end, the Adapted-OPTICS algorithm returns an ordered list $k'$ with all $P$ and $q$ processed points.

The Adapted-OPTICS update method (lines 10 and 16) works as follows:

1. First, it calculates the central distance to a given point $P$;
2. Second, a loop is made for all $P$ neighbors so that their reach distance is updated;
3. Then, if the object in the loop has not been processed, proceeds to the next step;
4. It calculates the new reach distance of this object in the loop;
5. If the object does not have the defined distance yet, then this object receives the value of the previous step, and it is added to the *Seeds* queue;
6. Otherwise, it is checked whether the newly calculated distance is less than the object's current reach distance;
7. If true, the current distance is replaced by the previous one, and the object is moved up in the *Seeds* queue.

**Table 4**
HCM Classification.

| ($w_i$) | LOS | Traffic Congestion Classification |
|---|---|---|
| [0, 0.15] | A | Free-flow |
| [0.15, 0.33] | B | Reasonably free-flow |
| [0.33, 0.50] | C | Stable-flow |
| [0.50, 0.60] | D | Approaching unstable-flow |
| [0.60, 0.70] | E | Unstable-flow |
| [0.70, 1.00] | F | Breakdown-flow |

### 3.3.3. Cloud computing layer

After the clustering algorithm reduces the traffic data in the fog layer, it is forwarded to the cloud via LTE/5G. In this article, we are concerned only with minimizing communication costs for this network link (LTE/5G) due to the high communication costs applied by cellular operators. In the cloud computing layer, cloud act as a global vehicular traffic manager that discovers the congestion roads in a VANET environment. Once the cloud receives the data, it calculates the traffic condition $w_i$ for each road $e_i$, providing the traffic condition for all road segments on road map. The definition of the traffic congestion is as follows:

**Definition 3.** Let a road segment $e_i$ be modeled as a set of weights $W = \{(w_1, t_1), (w_2, t_2), ..., (w_{|W|}, t_{|W|})\}$ representing the traffic condition $w_j$ over the time $t_i$, in which $w : E \longrightarrow \mathbb{R}^*_+$, that is, each $w_i$ is addressing the traffic condition at the time $t_i$ for the road segment $e_i$.

Based on data received from the fog computing layer, the framework at the cloud layer is responsible to perform the classification of the traffic flow. The traffic classification process used in this article uses the **Level of Service (LoS)**, as metric defined by Highway Capacity Manual (HCM) [4]. This approach is structured in six levels that provide a reference to classify the traffic condition based on Equation (1):

$$w_i = 1 - \frac{s_i^{avg}}{s_i^{max} \times d_i} \mid d_i > 0 \tag{1}$$

where $s_i^{avg}$, $s_i^{max}$, and $d_i$ represent the average speed, maximum allowed road speed, and density, respectively, of $e_i$. Each $w_i$ is used to detect high traffic density areas combined with low speeds, providing information about the congestion, location, and severity. Thus, the cloud is used to classify the traffic condition in each road $e_i$ based on each weight $w_i$ obtained. As shown in Table 4, LOS defines six different levels of service, providing a reference of measurement used to describe the conditions of traffic flow. Each of these levels represents the minimum and the maximum speed based on the maximum road speed allowed.

TCDS (Algorithm 4) in the framework describes the traffic classification condition at the cloud layer.

**Algorithm 4:** TCDS: Traffic Congestion Detection System.

**Require:** Set of clusters ($k' \in R$) with traffic data reduced
**Ensure:** Set of traffic classification ($LoS$) given by traffic condition $w_i$ for each road segment $e_i$
Initialize clusters set $LoS = 0$ and $visited = 1$
**for all** new cluster $k \in k'$ **do**
  **if** k is visited **then**
    $k_i = k_{i+1}$
  **else**
    $k = visited$
    $v^{avg} = k.avg()$
    $v^{max} = k.max()$
    $d_i = k.density()$
    $w_i = 1 - \frac{v_i^{avg}}{v_i^{max} \times d_i} \mid d_i > 0;$
    $LoS = w_i.value$
  **end if**
**end for**
**for all** $k \leftarrow (k = 0, ..., k = n)$ **do**
  $w_i.classification = w_i.value$ (Table 4)
  $LoS = w_i.classification$
**end for**
**return** ($LoS$)

**Table 5**
Simulation Parameters.

| Parameters | Values |
|---|---|
| Simulation Area | 1 km$^2$ |
| Number of road segments | 12 |
| Scenario | Grid |
| Vehicles Speed | 13 - 80 $km/h$ |
| MAC layer | IEEE 802.11p PHY |
| Mobility Simulator | SUMO 0.32.0 |
| Vehicular Network Simulation | Veins 4.7 |
| Discrete Event Simulator | OMNeT++ 5.3 |
| Transmission Power | 20 mW |
| Bit Rate | 6 Mbps |
| Beacon Transmission Rate | 1 Hz |

## 4. Performance analysis

In order to perform an initial validation of the proposed approach, a series of experiments was defined to be executed on a simulation environment for vehicular networks. For that, we considered an environment based on three main elements: 1) Veins (4.7v) [39], an open-source tool to model the connectivity among elements of the urban context (offering an implementation of the IEEE 802.11p protocol for the simulated connectivity); 2) SUMO (0.32v) [36], responsible for modeling the urban context (Roads, vehicles, and maps); and 3) OMNET++ (5.3v) [24], which deals with the network and connectivity aspects. The performance parameters used in our experiments are described in Table 5.

We performed the simulations using one Grid scenario but with two distinct behaviors, i.e., sparse and congested vehicular density. While the 1 to 6 road segments have low traffic flow (sparse), the 7 to 12 road segments are highly congested. All experiments were performed for all methods implemented in the fog (Baseline, Adapted-DBSCAN, and Adapted-OPTICS) for later transmission of data to the cloud (TCDS). We assume a scenario with only one RSU with a coverage of 1 $km^2$.

Data forwarding methods were employed in the fog Layer to deal with the continuous data stream environment. **(i) Baseline** (Algorithm 1) represents a well-spread algorithm that works with a packet frequency of 1 Hz to identify traffic congestion, sending all traffic data collected in the VANET environment to the cloud. **(ii) Adapted-DBSCAN** (Algorithm 2), which uses a sliding window to reduce traffic data and efficiently extract online traffic congestion levels, and **(iii) Adapted-OPTICS** (Algorithm 3), the third employed method.

An ordinary DBSCAN requires two parameters: the maximum distance between two points given by $\varepsilon$ and the minimum number of points required in a neighborhood (MinPts). In addition to these parameters, our Adapted-DBSCAN requires the $\lambda$, representing a set of data stream points to each cluster. Thus, as we deal with fewer data in each time window in a continuous data flow environment, the parameter $MinPts$ was defined $\geq 3$ to identify the formations of the traffic congestion, but it is important to highlight that the greater the amount of data, the more significant the chosen value for $MinPts$ should be, which avoids the increase of noise. The neighborhood radius $\varepsilon$ is related to the radius coverage and was defined according to coverage of the RSU (1 km$^2$) and $\lambda = 10\%$. Both algorithms were explained in detail in Section 3.3.2.

Adapted-OPTICS is closely related to Adapted-DBSCAN; however, unlike it, this algorithm implements an automatic variable neighborhood radius. This feature tries to deal with the problem of detecting meaningful clusters in data of varying density. Adapted-OPTICS receives two parameters $\varepsilon$ and $MinPts$, which are used to define the coverage of the data that should be processed. In this work, $\varepsilon$ was defined based on a knee method [30], which is reprocessed for each data set received from the temporal window. The $MinPts$ was configured using a percentage of this resultant data set.

Following our proposed goals, we analyzed the algorithms' impact on network cost and accuracy in the next subsections.

### 4.1. Network cost evaluation

The traffic data generated over time by vehicles is usually large in volume, leading to network congestion and higher communication expenses. We assume a wireless network link between fog and cloud with a max rate of 340 (KB/sec). Network Bandwidth $NB$ of this wireless link is given by $SB/T$, where $SB$ represents the traffic data, and $T$ is the channel bandwidth. $SB$ is also given by $LB + Bx$, where $LB$ is the $\sum_{i=0}^{i=n} B$, representing the sum of all packets sizes and $Bx$ is the average arrival rate. The greater the number of vehicles, the greater is $NB$. We approach this problem minimizing $LB$, which is subjected to the constraint $Ao \geq A_min$, where $Ao$ is the accuracy obtained, and $A_min$ is the minimum accuracy acceptable (LOS from HCM) to detect traffic congestion.

$NB$ was obtained from an average of 30 replication of an experiment (cf., Table 5) regarding the network usage for Baseline, Adapted-DBSCAN, and Adapted-OPTICS. These replications of a single experiment were conducted to quantify 95% Confidence Interval (CI) of the network usage variable. Hence, the performance analysis project is composed of factors and levels to shows the impact on the response variables. The factors analyzed were the (i) algorithms, which we alternate among Adapted-DBSCAN, Adapted-OPTICS, and Baseline; (ii) flow of vehicles, based on "SUMO" simulator to generate a random flow; and (iii) beacons rate, which is based on the frequency generation. These factors were evaluated to show the impact on the response variables, which allowed us to get measures such as network used rate, average speed, and accuracy, which can be obtained by the level of service implemented in Algorithm 4.

As shown in Fig. 3, the Adapted-DBSCAN generates less use of the network than other methods over time. This occurs due to the data reduction rate achieved with this adaptation and the way how this algorithm tackles the clustering forming. Adapted-DBSCAN minimizes the network usage by 60% compared to the Baseline and by 40% compared to the Adapted-OPTICS. Analyzing the Baseline algorithm, it is possible to note that the network usage increases over time. This can be explained by the increase in the number of vehicles during the experiment. To a lesser scale, a trend of growth can also be observed in the Adapted-OPTICS algorithm. On the other hand, network usage decreases for Adapted-

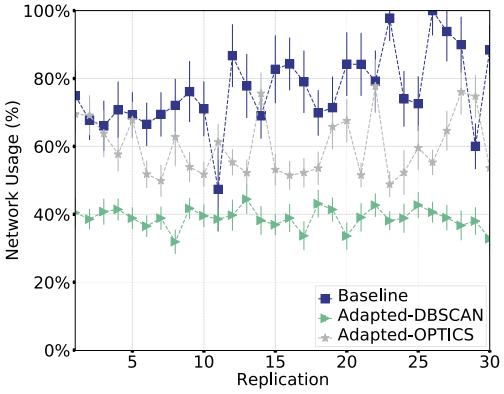



**Fig. 3.** Single Road Segment Experiment: Analysis of Network Usage.

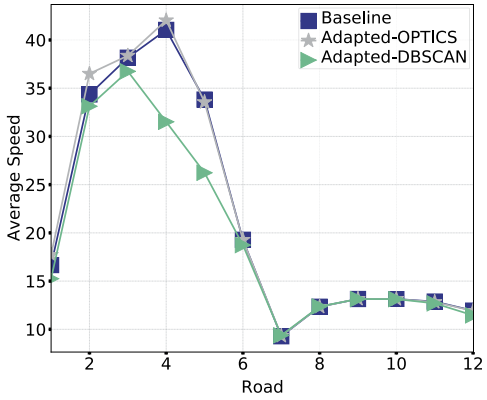



**Fig. 4.** Analysis of Speed Accuracy.

DBSCAN as more groups are formed, and therefore less data is sent over the network link.

Although the proposed framework provides an efficient model for data reduction and minimizing network cost, we need to ensure that the data reduction process maintains its original main characteristics without compromising the necessary accuracy to identify the variation in traffic congestion.

### 4.2. Accuracy evaluation

In order to check the LOS metric (accuracy) results in-depth, we assume in this subsection that the Baseline algorithm ensures high accuracy when detecting traffic congestion due to the absence of data reduction. Therefore, we use Baseline as a reference for the Adapted-DBSCAN and Adapted-OPTICS algorithms. Therefore, Fig. 4 shows each traffic data containing the vehicle speed over time for each road. The first six roads have a higher average speed than the last six. These two different behaviors offer a view of the environment with and without traffic congestion.

As shown in Fig. 4, both Adapted-DBSCAN and Adapted-OPTICS show a very close trend for speed monitoring; Adapted-OPTICS demonstrates a slight advantage in terms of hits compared to the Adapted-DBSCAN. Moreover, we can observe that while the level of traffic increases, the clustering algorithms behave in a similar way. Fig. 5 shows the similarity when using LOS precision.

Fig. 6 shows a regression model for all strategies of our framework. As we can observe, the correlation between algorithms increases as the simulation time also increases. This scenario can be explained by an increase in the traffic level over the simulation time and the increase in the amount of data analyzed. Within this context, we can observe that the cluster-based algo-

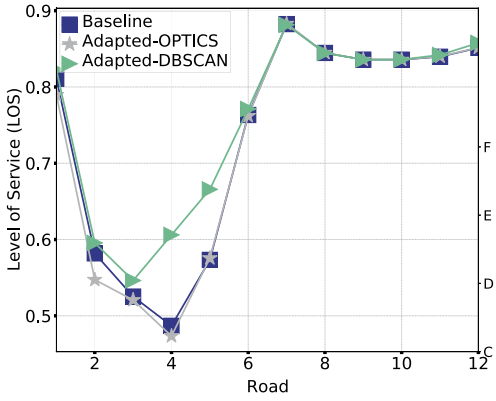



**Fig. 5.** Analysis of LoS Accuracy.

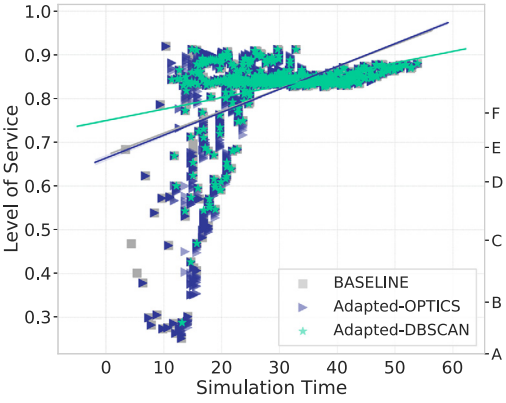



**Fig. 6.** Analysis of Traffic Congestion Classification (LOS).

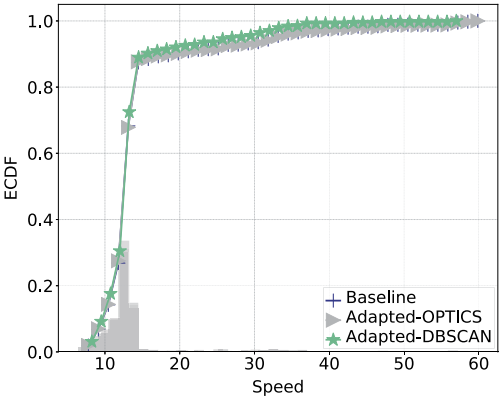



**Fig. 7.** Cumulative Distribution Function for Speed.

rithms Adapted-DBSCAN and Adapted-OPTICS reach a better accuracy when applied to high traffic flow scenarios.

Fig. 7 shows an Empirical Cumulative Distribution Function (ECDF), which presents an analyze data behavior during the simulation. We analyze the increase in the speed frequency during the experiments. Fig. 7 presents that 70% of speed data are upper to $20km/h$. This trend was observed for all algorithms. Thus, we can note that even using less data, Adapted-DBSCAN and Adapted-OPTICS were able to follow the data distribution of the Baseline during the experiments.

The average accuracy (LOS) of the Adapted-DBSCAN per road is shown in Fig. 8. It is possible to notice a high accuracy achieved by the framework using Adapted-DBSCAN despite the data reduction. There are fewer vehicles in scenarios of low traffic congestion, and therefore the number of clusters is reduced. For example, we have

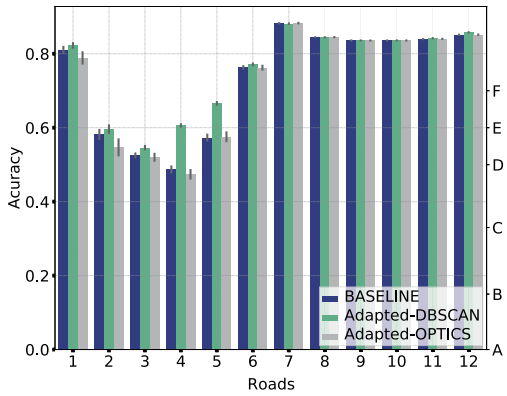



**Fig. 8.** Average of LOS Accuracy.

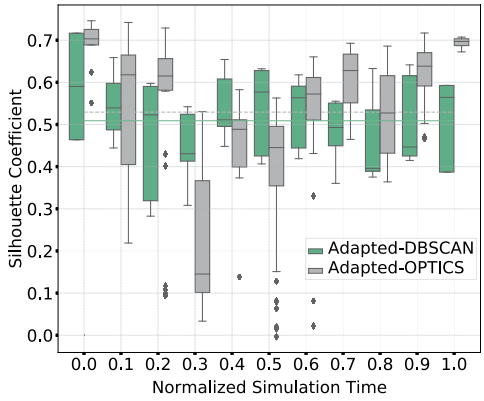



**Fig. 9.** Silhouette Coefficient.

a few data clusters produced in road number 2, leading to a 5% loss of accuracy. This minimum decrease in accuracy occurs for most roads number 1 to 6 when the scenario has a light traffic flow. On the other hand, when the scenario is overloaded, our algorithm reaches almost 100% accuracy, following the Baseline method and Adapted-OPTICS.

The Silhouette Coefficient is a measure of quality for a structured cluster. In this work, we use this measure to evaluate the quality level of clusters generated by the Adapted-DBSCAN and Adapted-OPTICS in the whole simulation. This method quantifies both the cohesion and separation between groups of instances $n$. The cohesion $a_{xi}$ of an instance $xi$ is calculated as the average distance between $xi$ and all other instances in the same group as $xi$. The separation $b_{xi}$ is the minimum distance between $xi$ and instances in all other groups. Rousseeuw [29] denotes the Silhouette Coefficient by Equation (2):

$$\frac{1}{n}\sum_{xi\in\mathcal{X}}\frac{(b_{xi}-a_{xi})}{max(a_{xi},b_{xi})} \tag{2}$$

The Silhouette ranges in $[-1;1]$, with larger values suggesting better cohesion and separation among clusters. Fig. 9 shows the result of the Silhouette Coefficient for the clustering algorithms used in our framework. To compare both clustering methods, the x-axis is given by a normalized simulation time, which represents the aggregated results at different points in time. Moreover, Fig. 9 shows all the groups formed during the simulation time. There is a certain variation in the values found for the two algorithms over each simulation time unit. Adapted-OPTICS presents great variability due to the natural characteristic of cluster a more number of disjunct groups in relation to Adapted-DBSCAN. There is a greater difference between the results at the beginning of the simulation. This occurs due to the reduced number of vehicles, which generates fewer groups for analysis. As the number of vehicles increases, the density grouping algorithms tend to maintain regularity and similarities in the results, with coefficients between 0.5 and 0.7 with low variability.

Silhouette is one of the most important metrics to represent quality for general clustering algorithms. However, the Silhouette is not appropriate to fully explain the dynamic of flow vehicles on the roads, as it disregards issues related to traffic, such as the direction of the vehicle and the position on the road for the creation of the groups.

## 5. Conclusion

Traffic congestion is one of the major problems for citizens living in large cities worldwide, leading to economic, environmental, and social issues in urban centers. In this way, traffic congestion detection systems use a large amount of traffic data stream to measure the source and severity of traffic flow online, increase network costs, and overload the existing network link infrastructure, especially from the edge of the network to the cloud. To overcome this continuous traffic data stream issue, we presented in this article a data clustering framework to reduce the traffic data stream, avoiding flooding the network link between RSUs/fog and the cloud.

The traffic data clustering framework employs two adapted clustering methods to reduce traffic data used by TCDS. Besides, a non-clustering method called Baseline was used to cover congestion detection without data reduction. As expected, Baseline has high accuracy but affects the network usage due to the absence of a data reduction method. On the other hand, our Adapted-DBSCAN and Adapted-OPTICS were able to reduce network usage and delay by clustering the traffic data based on vehicle density. As a comparative basis to the Adapted-DBSCAN, we used the clustering density algorithm Adapted-OPTICS, which showed a good level of accuracy compared to Baseline, but with a lower data reduction than Adapted-DBSCAN. For clustering methods, the more density, the less proportional information is transmitted from fog to the cloud due to the increasing number of vehicles per cluster. Therefore, the Adapted-DBSCAN proves worth reducing the traffic data while maintaining accuracy, mainly when the vehicular traffic environment is highly congested.

As future works, we intend to make improvements in the Silhouette Coefficient to cover traffic congestion aspects, allowing us to incorporate traffic flow characteristics in the equation that measures the quality of the clusters.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgements

The authors would like to thank the São Paulo Research Foundation (FAPESP), grants #2018/23126-3 and #2015/24494-8.

## References

[1] F. Aadil, K.B. Bajwa, S. Khan, N.M. Chaudary, A. Akram, Caconet: Ant colony optimization (aco) based clustering algorithm for vanet, PLoS ONE 11 (2016) e0154080, https://doi.org/10.1371/journal.pone.0154080.

[2] R. Bauza, J. Gozálvez, Traffic congestion detection in large-scale scenarios using vehicle-to-vehicle communications, J. Netw. Comput. Appl. 36 (2013) 1295–1307, https://doi.org/10.1016/j.jnca.2012.02.007.

[3] R. Bauza, J. Gozalvez, J. Sanchez-Soriano, Road traffic congestion detection through cooperative vehicle-to-vehicle communications, in: IEEE Local Computer Network Conference, IEEE, 2010, pp. 606–612.
[4] T.R. Board, Highway Capacity Manual, National Research Council, 2010.
[5] V. Cerqueira, L. Moreira-Matias, J. Khiari, H. van Lint, On evaluating floating car data quality for knowledge discovery, IEEE Trans. Intell. Transp. Syst. 19 (2018) 3749–3760, https://doi.org/10.1109/TITS.2018.2867834.
[6] C. Chatrapathi, M.N. Rajkumar, V. Venkatesakumar, Vanet based integrated framework for smart accident management system, in: 2015 International Conference on Soft-Computing and Networks Security (ICSNS), IEEE, 2015, pp. 1–7.
[7] K. Chen, Z. Li, Prediction of traffic state based on fuzzy logic in vanet, Inf. Technol. J. 12 (2013) 4642–4646, https://doi.org/10.3923/itj.2013.4642.4646.
[8] B. Cherkaoui, A. Beni-Hssane, M.E. Fissaoui, M. Erritali, Road traffic congestion detection in vanet networks, Proc. Comput. Sci. 151 (2019) 1158–1163, https://doi.org/10.1016/j.procs.2019.04.165.
[9] D.G. Costa, C. Duran-Faundez, D.C. Andrade, J.B. Rocha-Junior, J.P. Just Peixoto, Twittersensing: an event-based approach for wireless sensor networks optimization exploiting social media in smart city applications, Sensors 18 (2018), https://doi.org/10.3390/s18041080.
[10] M. Ester, H.P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, AAAI Press, 1996, pp. 226–231.
[11] A. Gharaibeh, M.A. Salahuddin, S.J. Hussini, A. Khreishah, I. Khalil, M. Guizani, A. Al-Fuqaha, Smart cities: a survey on data management, security, and enabling technologies, IEEE Commun. Surv. Tutor. 19 (2017) 2456–2501, https://doi.org/10.1109/COMST.2017.2736886.
[12] B.F. Guedes, C.A. Campos, A data aggregation scheme for traffic information systems in urban vanets, in: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), IEEE, 2016, pp. 564–569.
[13] G. Guido, D. Rogano, A. Vitale, V. Astarita, D. Festa, Big data for public transportation: a dss framework, in: 2017 5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS), 2017, pp. 872–877.
[14] B. Havers, R. Duvignau, H. Najdataei, V. Gulisano, A.C. Koppisetty, M. Papatriantafilou, Driven: a framework for efficient data retrieval and clustering in vehicular networks, in: 2019 IEEE 35th International Conference on Data Engineering (ICDE), IEEE, 2019, pp. 1850–1861.
[15] Z. He, D. Zhang, Cost-efficient traffic-aware data collection protocol in vanet, Ad Hoc Netw. 55 (2017) 28–39, https://doi.org/10.1016/j.adhoc.2016.09.021.
[16] IDC2016, Worldwide Internet of Things Forecast Update 2015–2019, document #US40983216, Framingham, MA, USA, 2016.
[17] Saptawati G.A.P. Irrevaldy, Spatio-temporal mining to identify potential traffic congestion based on transportation mode, in: 2017 International Conference on Data and Software Engineering (ICoDSE), 2017, pp. 1–6.
[18] B. Jarupan, E. Ekici, A survey of cross-layer design for vanets, Ad Hoc Netw. 9 (2011) 966–983, https://doi.org/10.1016/j.adhoc.2010.11.007.
[19] K. Kaur, S. Kad, Enhanced clustering based aodv-r protocol using ant colony optimization in vanets, in: 2016 IEEE 1st International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES), IEEE, 2016, pp. 1–5.
[20] A. Keramatian, V. Gulisano, M. Papatriantafilou, P. Tsigas, Mad-c: multi-stage approximate distributed cluster-combining for obstacle detection and localization, J. Parallel Distrib. Comput. 147 (2021) 248–267, https://doi.org/10.1016/j.jpdc.2020.08.013.
[21] H. Li, Y. Liu, Z. Qin, H. Rong, Q. Liu, A large-scale urban vehicular network framework for iot in smart cities, IEEE Access 7 (2019) 74437–74449, https://doi.org/10.1109/ACCESS.2019.2919544.
[22] W.S. Manjoro, M. Dhakar, B.K. Chaurasia, Traffic congestion detection using data mining in vanet, in: 2016 IEEE Students' Conference on Electrical, Electronics and Computer Science (SCEECS), 2016, pp. 1–6.
[23] H. Najdataei, Y. Nikolakopoulos, V. Gulisano, M. Papatriantafilou, Continuous and parallel lidar point-cloud clustering, in: 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS), 2018, pp. 671–684.
[24] OMNET++, Omnet++ - Network Simulation Framework, available https://www.omnetpp.org/, 2020. (Accessed 11 January 2020).
[25] K. Patroumpas, T. Sellis, Window specification over data streams, in: International Conference on Extending Database Technology, Springer, 2006, pp. 445–464.
[26] M.L.M. Peixoto, E.M. Cruz, A.H.O. Maia, M.C.A. Santos, W.V. Lobato, L.A. Villas, Exploiting fog computing with an adapted dbscan for traffic congestion detection system, in: 2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall), 2020, pp. 1–5.
[27] T. Reed, J. Kidd, Global Traffic Scorecard, INRIX Research, Altrincham, 2019.
[28] M.E. Renda, G. Resta, P. Santi, F. Martelli, A. Franchini, Ieee 802.11p vanets: experimental evaluation of packet inter-reception time, Comput. Commun. 75 (2016) 26–38, https://doi.org/10.1016/j.comcom.2015.06.003.
[29] P.J. Rousseeuw, Silhouettes: a graphical aid to the interpretation and validation of cluster analysis, J. Comput. Appl. Math. 20 (1987) 53–65.
[30] V. Satopaa, J. Albrecht, D. Irwin, B. Raghavan, Finding a "kneedle" in a haystack: detecting knee points in system behavior, in: 2011 31st International Conference on Distributed Computing Systems Workshops, 2011, pp. 166–171.
[31] R.K. Schmidt, T. Leinmuller, E. Schoch, F. Kargl, G. Schafer, Exploration of adaptive beaconing for efficient intervehicle safety communication, IEEE Netw. 24 (2010) 14–19, https://doi.org/10.1109/MNET.2010.5395778.
[32] J. Shen, D. Liu, X. Chen, J. Li, N. Kumar, P. Vijayakumar, Secure real-time traffic data aggregation with batch verification for vehicular cloud in vanets, IEEE Trans. Veh. Technol. 69 (2019) 807–817, https://doi.org/10.1109/TVT.2019.2946935.
[33] M. Silva, G. Signoretti, J. Oliveira, I. Silva, D.G. Costa, A crowdsensing platform for monitoring of vehicular emissions: a smart city perspective, Future Internet 11 (2019), https://doi.org/10.3390/fi11010013.
[34] A.M. de Souza, L.L.C. Pedrosa, L.C. Botega, L. Villas, Itssafe: an intelligent transportation system for improving safety and traffic efficiency, in: 2018 IEEE 87th Vehicular Technology Conference (VTC Spring), 2018, pp. 1–7.
[35] A.M. de Souza, R.S. Yokoyama, A. Boukerche, G. Maia, E. Cerqueira, A.A. Loureiro, L.A. Villas, Icarus: improvement of traffic condition through an alerting and re-routing system, Comput. Netw. 110 (2016) 118–132, https://doi.org/10.1016/j.comnet.2016.09.011.
[36] SUMO, SUMO - Simulation of Urban Mobility, available http://sumo.sourceforge.net/, 2020. (Accessed 11 January 2020).
[37] D. Tolpin, Progressive temporal window widening, preprint, arXiv:1604.00997, 2016.
[38] A. Ullah, X. Yao, S. Shaheen, H. Ning, Advances in position based routing towards its enabled fog-oriented vaneta survey, IEEE Trans. Intell. Transp. Syst. 21 (2020) 828–840, https://doi.org/10.1109/TITS.2019.2893067.
[39] VEINS, Vehicles in network simulation, available http://veins.car2x.org, 2020. (Accessed 11 January 2020).
[40] A. Wahid, A. Rao, D. Goel, Server communication reduction for gps-based floating car data traffic congestion detection method, in: Integrated Intelligent Computing, Communication and Security, Springer, 2019, pp. 415–425.