Ad Hoc Networks 140 (2023) 103046

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

FogJam: A Fog Service for Detecting Traffic Congestion in a Continuous Data Stream VANET

M.L.M. Peixoto [a,b,*], E. Mota [a], A.H.O. Maia [a], W. Lobato [b], M.A. Salahuddin [c], R. Boutaba [c], 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] David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, Canada

ARTICLE INFO

Keywords:
Traffic congestion
Data stream
Data clustering
VANET
Fog computing

ABSTRACT

In a continuous data stream vehicular network environment, a Traffic Congestion Detection Service (TCDS) receives many periodic information to update and discover road segments with low speeds and high vehicular density. The use of Cloud computing to support the massive amounts of traffic data received from multiple vehicles significantly increases network traffic. Therefore, we propose FogJam, a Fog service to detect traffic congestion directly at the edge of the vehicular network. In the network edge, FogJam leverages sampling and clustering-based methods to reduce the traffic data stream transmitted by all vehicles on the network links to the Cloud. In the Cloud, FogJam is used as a macro-control of all vehicles geographic positions, acquiring traffic flow data to detect traffic congestion. We evaluate FogJam using *OMNeT++, Veins and SUMO* simulators. The results suggest that FogJam is highly accurate in detecting traffic congestion at a lower cost, even in a high vehicular density scenario. Furthermore, using clustering-based methods, FogJam is able to, on average, reduce the impact on network usage by approximately 70% compared to the sampling methods, while maintaining an acceptable level of congestion detection accuracy.

1. Introduction

Traffic congestion is a major concern in metropolitan areas, which inevitably leads to substantial social and economic consequences. In 2019, Texas A&M University's Transportation Institute reported that drivers in the United States were stuck in traffic for up to 54 h in 2017 [1]. According to the 2019 Urban Mobility Report from INRIX [2], this increased to 97 h in 2018, costing Americans nearly $87 billion annually. Indeed, traffic congestion increases the carbon emissions and the probability of accidents.

Despite the increase in traffic congestion over the years, congestion and time spent in traffic considerably reduced in 2020 [3]. This is primarily attributed to the COVID-19 pandemic that was and still is plaguing the world. During the pandemic in Hong Kong, local travel volume decreased by 52.3% [4]. An annual travel time decrease for all car passengers of $5.58 billion was also observed, at 54.02% reduction in the Pre-COVID-19 total time costs [5]. However, there has also been a reduction in transport sharing during the pandemic due to health safety concern, which increases the total number of vehicles with underutilized capacity on the roads [6]. Notwithstanding the pandemic's unusual period, it is necessary to identify traffic congestion as efficiently as possible. As transportation analysts expect more intense traffic congestion in the coming months, this analysis is essential to address post-pandemic traffic congestion, particularly in large urban centers [7–9].

Urban Computing is the acquisition, integration, and analysis of a large volume of data that is continuously generated by various sources, such as sensors, devices, vehicles, buildings, and people [10]. Along with the development of Urban Computing, sensing technologies are becoming ubiquitous within the Intelligent Transportation System (ITS). This has contributed to the enormous growth of continuous data stream in the urban mobility scenario. The continuous data stream is produced at high volume, variety, and velocity from a Vehicular Ad-Hoc Network (VANET), which must be stored and processed to mitigate traffic congestion in ITS [11]. According to the International Data Corporation (IDC), the number of connected devices worldwide is expected to reach 75 billion by 2025, generating 73.1 ZB of data. This includes 110 million cars and 1.2 million homes connected with 5.5 billion and 200 million sensors, respectively [12,13]. Arcaro predicts that by 2023, almost 70% of new vehicles worldwide (i.e., 90% in the United States) will have an integrated connectivity interface to a

* 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.adhoc.2022.103046
Received 11 November 2021; Received in revised form 9 August 2022; Accepted 13 November 2022
Available online 21 November 2022

cellular network, which will provide vehicular data from gateways, software, or sensors [14]. This leads to an exponential growth in data, increases the communication cost of the network, and the computational complexity for data orchestration (e.g., storage, processing, and analysis) and extraction of knowledge [15,16].

Although vehicular traffic data is growing, ITS uses this valuable mobility information from multiple vehicles to facilitate travel-related applications, such as carpooling, public transportation, and taxis [17]. These applications require information regarding the level of road congestion to plan travel routes and minimize travel time, collisions, fuel consumption, and carbon emissions. Hence, a Traffic Congestion Detection Service (TCDS) is widely used in ITS to discover areas with high vehicle density, primarily facilitated by the low speed of vehicles. TCDS uses a large amount of situational vehicular traffic data (e.g., vehicle position, vehicle speed, and vehicle acceleration) to characterize traffic flow congestion [11,18,19].

One way to provide a global view of traffic congestion, including the condition for each road segment, is by using a Cloud-like service, i.e., a macro-control service based on the geographic position of all vehicles [11,20–24]. However, some urban areas can have millions of vehicles transmitting continuous traffic information data to the Cloud, which can significantly increase the network traffic. Both communication cost and the management of continuous data streams are barriers for TCDS. Although authors in [20,21,24] have proposed approaches to reduce data transmissions and minimize communication overhead, it is unclear whether these works can guarantee accuracy on traffic congestion detection with only data reduction. In addition, to the best of our knowledge, none of the existing traffic congestion detection services explore reduction in a continuous data stream, but rather perform traditional batch processing in a static, offline manner.

In this work, we propose FogJam, a Fog service to detect traffic congestion in VANETs. FogJam overcomes the limitations of the state-of-the-art approaches by leveraging Fog and Cloud computing to offer a TCDS over continuous data stream. Unlike some works in the literature that consider data reduction in communication between vehicles and infrastructure to address the limitations of the WAVE IEEE 802.11p communication protocol, FogJam focuses on the communication between Fog's IoT devices and the Cloud.

FogJam moves the distribution of computing capacity closer to vehicles, reducing network traffic. The high-level view of the architecture is illustrated in Fig. 1. The VANET layer is composed of vehicles, RSUs (Roadside Units), road maps, and it is further related to the IEEE 802.11p vehicular network connectivity elements. FogJam does not modify this layer, receiving traffic data transparently from RSUs. In the Fog layer, FogJam has the Internet of Things (IoT) devices to deal with all traffic data stream collected from the VANET layer, deploying data reduction methods based on sampling or clustering traffic data. In this layer, FogJam aims to reduce the amount of traffic data that is used to detect road congestion in the Cloud. Then, FogJam uses the highest capacity computing resources in the Cloud layer to process TCDS. To extract value from large data volumes, processing power offered by cloud computing is essential.

FogJam aims at detecting traffic congestion over the continuous data stream, while reducing the data directly at the network edge. To achieve this, FogJam deploys two different methods: (i) sampling-based, and (ii) clustering-based. We consider the state-of-the-art methods proposed to reduce traffic data in a TCDS, and compare them with each other. Note that these methods are adapted to support continuous data stream. While methods based on sampling use statistical metrics to provide traffic information for each road segment, the machine learning (ML)-based clustering methods group vehicular data based on the density of vehicles in a region. FogJam works incrementally and adaptively with the traffic data stream to detect road congestion, while reducing communication cost, i.e., the traffic data on the network links.

We outline the following strengths of our work:

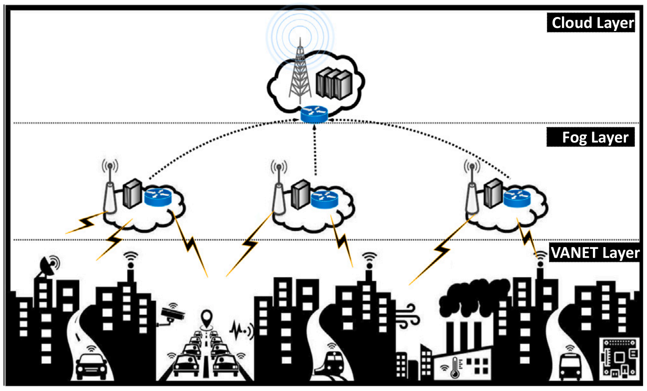


Fig. 1. High-level view of the VANET, Fog and Cloud Architecture.

1. We propose FogJam, a Traffic Congestion Detection Service for ITS that has a low communication cost.
2. FogJam effectively supports continuous data stream analysis in an online mode for real-time ITS applications, such as emergency vehicles, carpooling, public transportation, and taxis.
3. Through extensive experiments, we show that FogJam can manage and reduce continuous traffic data stream, while accurately detecting traffic congestion areas and demonstrating its effectiveness in a well-established simulators, i.e., OMNeT++, Veins, and SUMO.

The remainder of the article is structured as follows: Section 2 introduces the related works. Section 3 describes the system model and our problem formulation. The experiments and their results are discussed in Section 4, while Section 5 provides a summary of our work and instigates future research direction.

2. Related work

The main idea behind TCDS is to offer essential road congestion information for other ITS services, such as planning new routes, traffic monitoring, speed alerts, smart traffic light, and others. Therefore, several studies attempted to improve TCDS in ITS. Regardless of whether the Fog computing is employed or not, most works in literature focus on traffic congestion detection without considering its impact on the network infrastructure. The approaches mentioned in this section address the basic TCDS issues.

Images and videos have also been employed to model or estimate traffic conditions. Kumar and Kushwaha [25] and, [26] discussed traffic congestion detection based on images captured from cameras installed at several places in the city. Their approach uses the number of vehicles in the image and road width to estimate traffic density. Authors in [27] use an unsupervised ML to count the number of vehicles in an image. However, all works, including [28–32] that are based on images and/or videos to detect traffic congestion, inherently suffer in poor environmental conditions, such as rain and mist. Besides, these approaches require the installation of numerous cameras, which increases the infrastructure cost.

Several studies have addressed the infrastructure cost issue [18,33–41], by focusing on low-cost sensors to collect traffic data for of congestion detection. In the case of Rath et al. [18], a mobile agent embedded in a micro-controller is proposed to detect traffic congestion and reduce the time for detecting congestion. On the other hand, Wang et al. [40] combined the extraction of traffic event signals from social media to GPS data to estimate traffic congestion. Seid et al. [41] used low-cost devices combined with LoRaWAN networks to process traffic management services. They extended the conventional coupled hidden Markov model to integrate these two types of data. Although these studies are important within the sensor data collection service, they do not show the impact of data acquisition on network communication. Besides, these works are oblivious to the communication cost associated with data exchange between vehicles and infrastructure.

Similarly, Ramachandra et al. [42] proposed an IoT prototype to measure the traffic flow density using the On-Board Diagnostic (OBD) sensor. All vehicles were equipped with an OBD device to acquire speed data transmitted via Bluetooth to an on-board central unit (OBU). After data acquisition, the OBUs communicate with nearby light poles (as a RSU) using Zigbee, which sends the aggregated data to a central server (as a Cloud-like service). The authors used a small time interval for data transfer to minimize network usage. However, this time interval nor the amount of data transferred over the network is specified.

Rani and Shaw [43] used fuzzy logic as a service to estimate traffic congestion, employing technical attributes (e.g., traffic volume, green light time, cycle time) and non-technical attributes (e.g., weather condition, road condition, visibility). Both types of attributes are input to the rule aggregation in the fuzzy logic process to estimate the traffic congestion. Kangli and Zhipeng [44] proposed a traffic flow estimation service using Fuzzy logic, which is based on the data obtained between the vehicle's position. They classify the traffic congestion in three states: (i) free flow, (ii) synchronized flow, and (iii) wide movement congestion flow. The authors in [45] proposed the CoTEC (COperative Traffic congestion detECtion) technique, a service to optimize the detection of vehicle traffic using Fuzzy logic. In Bauza et al. [45], the authors proposed a mechanism based on signal messages received from neighboring vehicles and observing the increase of re-transmitter vehicles to perform congestion detection. The higher the number of messages, the higher the quality of information, and the lower the RMSE (Root Mean Square Error). Although these Fuzzy Logic works estimate of traffic congestion at intersections of a road network, they do not analyze the communication cost for data acquisition.

As discussed by Wang et al. [40], social media has a vast amount of data that has been explored for traffic detection. This data describes "what happens now and where", which can be used to understand the user's mobility pattern. For example, Twitter, Foursquare, and Facebook have geographic posting tools, which show the location of users. Similarly, Wang et al. [46] proposed a Twitter-based railway delay detection method that is based on topic propagation analysis of geotagged tweets between railway stations. Their main goal is to predict congestion in traffic accident events by analyzing topic propagation using real railway network topology. This approach is efficient as it does not use other sensors to detect traffic. However, using social networks require a large volume of public data and processing.

On the other hand, clustering techniques have been used to partition a large number of heterogeneous vehicular data into different homogeneous groups. Lai et al. [47] proposed a cluster-based method to process density queries in a road network. Their approach computes the density using parameters, such as road segment length and the number of objects on the road segment. They define a cluster as a group of continuously moving objects close to each other. In another clustering-based approach [48], the authors leveraged clustering algorithms Expectation–Maximization (EM) and Density-Based Spatial Clustering of Applications with Noise (DBSCAN) to analyze traffic data characteristics (e.g., traffic flow, density, speed). Their approach applied spatio-temporal data mining to detect potential traffic congestion on the road. However, the authors neither analyze the communication cost nor identify the traffic congestion source (e.g., road segment). In contrast, FogJam records the trajectory data from vehicles represented by the centroid of a group, providing information on the origin of traffic congestion. Zhang et al. [49] proposed a clustering algorithm to analyze the degree of traffic congestion based on gray relational analysis theory, which considers multidimensional data and different clustering levels. They used three variables for traffic flows (i.e., speed, density, and volume) to evaluate the degree of traffic congestion. Unlike FogJam, which analyzes continuous data stream from moving vehicles, cluster-based approaches, including [50,51], assume a static dataset, working only in batch mode. In addition, these works do not evaluate the impact of their approach on network communication costs.

To minimize network usage and cost related to the traffic congestion detection process, the authors in [20,21,24] proposed sampling-methods that reduce the number of transmissions from vehicles to a central server (e.g., Cloud-like service). They assumed that there are a lot of redundant message transmissions in a TCDS. In their approaches, each vehicle must decide when to send data (i.e., travel time on a road segment) to the TCDS server. Kerner et al. [20] presented a deterministic approach where the vehicles send their current traffic data (i.e., speed) to the server when it exceeds a predefined speed-threshold stored in the vehicles. As a more advanced traffic congestion detection approach with data reduction, Tanizaki et al. [21] proposed an approach, which we refer to as Rand. Rand is a TCDS used to reduce the communication cost through a randomized update function. In their approach, the vehicles transmit their speed to the server with a certain probability p. As an extension of [20], Wahid et al. [24] proposed a server update policy, which we refer to as Lim. In Lim, a vehicle sends data to the server if the difference between the current speed and an already stored speed in a interval time, is greater than or equal to a threshold. Due to the similarity of the objective of these approaches in reducing traffic data of a TCDS, we use their approaches for comparison.

TCDS can generate a large data stream, leading to a significant increase in communication cost. Almost all the related works presented in this section are inconsiderate to the amount of data used for estimating congestion and the impact of spreading this knowledge on communication cost. These works do not evaluate data reduction, not even methods that perform clustering of vehicle density information over time. Besides, to the best of our knowledge, no work was found that provides data reduction in a continuous data stream to TCDS. Therefore, to cope up with the limitations of traditional batch solutions, which involve the processing of static datasets, we propose FogJam, which offers low-cost, online processing of continuous data to detect traffic congestion. It incrementally and adaptively processes the traffic data stream, and reduces communication between Fog and Cloud. FogJam leverages clustering techniques for data reduction and identification of the source and severity of traffic congestion.

3. FogJam: Service model

In conjunction with Cloud computing, Fog computing improves traffic congestion detection, which is useful for other real-time ITS services, such as emergency vehicles, carpooling, public transportation, and taxis. Therefore, FogJam leverages Fog computing to provide an accurate traffic congestion detection service for VANET, which focuses on minimizing communication cost. Fog computing reduces vehicular network traffic by distributing computing capacity closer to the vehicles. As illustrated in Fig. 2, FogJam is divided into three layers: (i) Cloud Layer, (ii) Fog Layer, and (iii) VANET Layer.

Analyzing Fig. 2, we can see that there are three layers that work together to perform the monitoring of traffic jams. First, in the VANET Layer, we have a RSU that collect data from vehicular traffic. Next, the Fog Layer is responsible for processing data received from the VANET Layer and implementing the data reduction methods proposed in this work. Fog Layer is also responsible for dealing with data from different roads forming a subset that represents the traffic situation in a certain radius. Then, the data reduced are sent to the Cloud Layer where the decisions can be made based on the subset received from Fog Layer.

Thus, while the Cloud Layer analyzes traffic data to extract road congestion information, the Fog Layer focuses on traffic data reduction, and the VANET Layer generates traffic information. All service model parameters are shown in Table 1.

Table 1
Table of the service model parameters.

Cloud parameters	Description
$R = \{r_1^1, \dots, r_i^j, \dots, r_c^f\}$	Set of computing resources in the Fog-Cloud hierarchy
c	Resource id
$r_i^j.B$	Associated Bandwidth Vector
s_i^{avg}	average speed
s_i^{max}	maximum allowed road speed
d_i	density on the road
$LOS\|W = \{w_1, w_2, \dots, w_i\}$	Set of level of service
D	Set of traffic data
Fog parameters	**Description**
$A = \{a_1, a_2, \dots, a_f\}$	Set of services deployed by the fog devices
$F = \{f_1, f_2, \dots, f_{\|F\|}\}$	Set of fog device id
q	Traffic flow given by speed and density
$S = \{(S_1, \tau_1), (S_2, \tau_2), \dots\}$	Continuous data streams
τ	Timestamp
Δ	Interval time
γ	Sliding-window size
VANET parameters	**Description**
$G = (V, E)$	Directed and weighted graph
$V = \{v_1, \dots, v_{\|V\|}\}$	Set of intersections
$E = \{e_1, \dots, e_{\|E\|}\}$	Set of road segments connecting the intersections
$U_{o,p} = \{e_1, \dots, e_{\|E\|}\}$	Ordered set of road segments in the route with origin o and destination p
$N = \{n_1, n_2, \dots, n_{\|N\|}\}$	Set of vehicles
$M_i = \{P_{(x,y),n_{id}}\}$	Beacon message
$P_{(x,y)}$	Vehicle's position (x ← latitude and y ← longitude)
n_{id}	Vehicle's identification
$RSU = \{rsu_1, \dots, rsu_{\|RSU\|}\}$	Set of RSUs

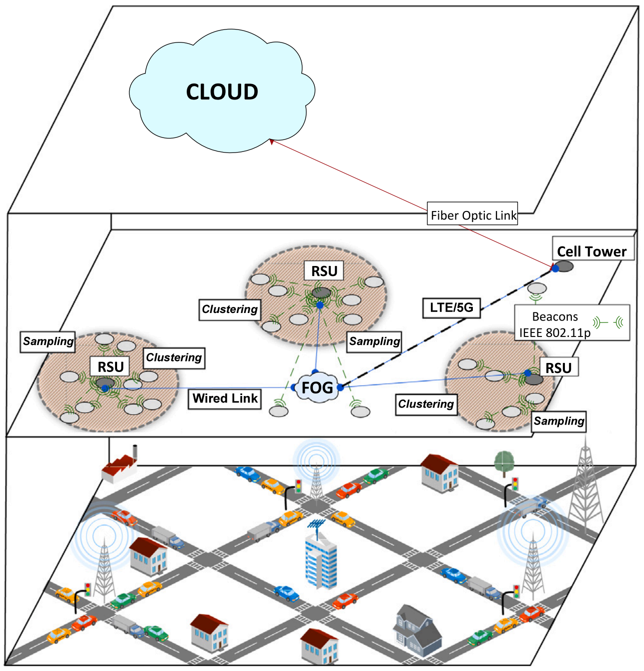


Fig. 2. FogJam: A Low-cost Communication Service for Detecting Traffic Congestion in Continuous Data Stream.

3.1. Cloud layer

The uppermost layer is the Cloud Layer, which is composed of computing resources that are used to process the traffic congestion detection service. The Cloud-Fog computing resources are represented by the overall set of computing devices $R = \{r_1^1, r_2^1, \dots, r_i^j, \dots, r_{c-1}^{f-1}, r_c^f\}$, with $1 \leq i \leq c$ and $1 \leq j \leq f$, where c and f are the indices representing the unique identification number for Cloud and Fog resources, respectively. Each cloud resource $r_i^j \in R$ has a connectivity vector B representing the available bandwidth with every resource in the Fog, i.e., $r_a^j.B[r_b^f] \geq 0$ and $r_a^j.B.[r_a^j] = \infty, \forall r_a^j, r_b^j \in R$. If r_a^j and r_b^f are unreachable, then $r_a^j.B[r_b^f] = r_b^f.B[r_a^j] = 0$.

TCDS is executed in the Cloud Layer, and is responsible for detecting congestion level on the roads. In this way, acting as an extension of Fog, Cloud has a centralized view of the road infrastructure, and provides road congestion information for other ITS services to enable decision-making in an orchestrated manner across the vehicular environment. TCDS running in the Cloud receives traffic data from the Fog with information about the position, density, and speed of vehicles. TCDS uses this information to calculate the traffic conditions w_i, where $W = \{(w_1, \tau_1), (w_2, \tau_2), \dots, (w_{|W|}, \tau_{|W|})\}$ represents a set of traffic condition w_i over the time τ_i. And, $W : E \longrightarrow R_+^*$, that is, each w_i is addressing the traffic condition at time τ_i for each road segment e_i.

For TCDS, we adopted the traffic congestion classification defined by the Highway Capacity Manual (HCM) [52]. HCM was created by the United States Transportation Research Board in the form of mathematical equations to estimate traffic Level-Of-Service (LOS). This provides a reference to classify the traffic condition based on the resultant factor of the following equation:

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

where s_i^{avg}, s_i^{max}, and d_i represent the average speed, maximum allowed road speed, and traffic density, respectively, for e_i. Each w_i is used to detect high traffic density areas combined with low speeds, providing information about congestion, its location and severity. As shown in Table 2, there are six different LOS, providing a reference that is used to describe the conditions of traffic flow. Each of these levels represent the minimum and the maximum speed based on the maximum road speed limit. We included color references to facilitate the understanding of a heat map.

All congestion information (i.e., w_i) for each road segment e_i is stored in the LOS matrix, as illustrated in Fig. 3. Each row represents a road segment e_i, while each column indicates the corresponding timestamp τ_i.

TCDS in the Cloud Layer is shown in Algorithm 1. The algorithm aims to obtain the traffic classification condition for each road. TCDS algorithm first joins the data streams from all Fog nodes into a single stream, offering a global view of the road congestion at the Cloud. After

Table 2
Traffic condition w_i classification.

Road weight (w_i)	LOS	Traffic flow 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

$$\text{LOS} = \begin{array}{c|ccccc} & \tau_1 & \tau_2 & \tau_3 & \cdots & \tau_j \\ e_1 & w_{(e_1,\tau_1)} & w_{(e_1,\tau_2)} & w_{(e_1,\tau_3)} & \cdots & w_{(e_1,\tau_j)} \\ e_2 & w_{(e_2,\tau_1)} & w_{(e_2,\tau_2)} & w_{(e_2,\tau_3)} & \cdots & w_{(e_2,\tau_j)} \\ e_3 & w_{(e_3,\tau_1)} & w_{(e_3,\tau_2)} & w_{(e_3,\tau_3)} & \cdots & w_{(e_3,\tau_j)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ e_i & w_{(e_i,\tau_1)} & w_{(e_i,\tau_2)} & w_{(e_i,\tau_3)} & \cdots & w_{(e_i,\tau_j)} \end{array}$$

Fig. 3. LOS Congestion information Matrix.

all road segments are visited, the traffic condition is calculated based on Eq. (1). Then, the entire set of traffic conditions W is stored in the LOS congestion information matrix (cf. Fig. 3). It is worth mentioning that the set D containing several traffic data streams can be obtained using data reduction methods deployed by Fog nodes in the Fog Layer.

Algorithm 1: TCDS: Traffic Congestion Detection Service

```
Require: Sets of (reduced) traffic data $D$ from all Fog nodes
    $\in F$ in the time interval $\Delta t$
Ensure: Set of traffic congestion information ($LOS$)
 1: Initialize $LOS = 0$
 2: $D \bowtie D_{f_1}, D_{f_2}, \dots$
 3: for all new road segment $e_i \in D$ do
 4:   if ($e_i$ is visited) then
 5:     $e_i = e_{i+1}$
 6:   else if ($D_{e_i} \subseteq \Delta_{(\tau_\alpha,\tau_\beta)}$) then
 7:     $e_i = visited$
 8:     $s^{avg} = D.e_i.s^{avg}()$
 9:     $s^{max} = D.e_i.s^{max}()$
10:     $d_i = D.e_i.density()$
11:     $w_i = 1 - \frac{s_i^{avg}}{s_i^{max} \times d_i} \mid d_i > 0;$
12:     $w_i.classification = w_i.value$ (cf. Table 2)
13:     $LOS = (w_i.classification, \tau_i)$
14:   end if
15: end for
16: return ($LOS$)
```

3.2. Fog layer

In the Fog Layer, FogJam receives all traffic data stream from the VANET Layer and implements data reduction based on sampling or clustering of the vehicular data. To accomplish this, there are Fog devices connected to the RSUs, which are used to execute traffic data reduction service. Let $F = \{f_1, f_2, \dots, f_{|F|}\}$ be a set of Fog devices connected to RSUs in the Cloud-Fog infrastructure. Also, let $A = \{a_1, a_2, \dots, a_f\}$ be a set of data reduction services running in the Fog devices.

The primary objective of FogJam is to minimize the cost of communication over the LTE/5G network that connects the Fog devices to the Cloud. FogJam operates at the network edge with two main functions: (i) handle traffic flow information made up of continuous data stream, and (ii) employ sampling and clustering methods to reduce the amount of traffic flow that is used to detect road congestion.

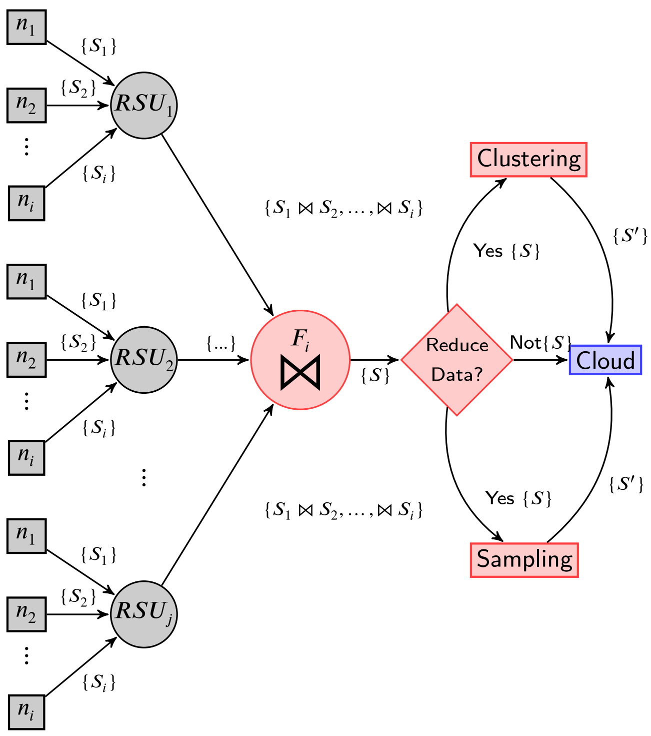


Fig. 4. Data stream processing in FogJam.

3.2.1. Continuous data stream

Each Fog device is continuously receiving data streams from several RSUs. All of these data streams are joined into a single Data stream. Data streams (S) are sequences of unlimited tuples generated continuously over time. $S = \{(S_1, \tau_1), (S_2, \tau_2), \dots\}$, where S is an infinite sequence of elements (i.e., vehicular traffic information), and the time is given by $\tau \in R_+^* = \{x \in R | x \geq 0\}$. FogJam processes S, clustering and reducing the traffic data as new input traffic data becomes available.

To handle the temporal characteristics of S, we partitioned the sequence of elements into a series of segments $S' = \{p_1, p_2, \dots, p_{|S'|}\}$ according to a sliding-window size γ. Then, the τ-th window is reformulated to a *micro batch* facilitating FogJam processing. Only the most recent S with a fixed *gamma* size is maintained at D in the sliding-window. The elements in the sliding-window follow a *first in, first out* (FIFO) scheduling method, which considers elements from the current period (τ_i) up to a period in the past (τ_{i-j}), where i is the begin and j the end of the time interval Δ. The abstraction of stream processing performed by FogJam in Δ is shown in Fig. 4, where n_i represents the identification of each vehicle in a RSU coverage.

The gray elements in Fig. 4, i.e., the set of vehicles $\{n_1, n_2, \dots, n_i\}$ and the set of RSUs $\{rsu_1, rsu_2, \dots, rsu_j\}$, are part of the VANET Layer. Each vehicle n_i sends a traffic data stream $\{S_i\}$ to RSU_j in its coverage area. All traffic data streams are joined through a $\bowtie$ operation, and the resultant stream $\{S_j\}$ is forwarded to the closest Fog device. The red elements in Fig. 4 indicate the methods employed by FogJam in the Fog Layer. The first step of processing in this layer is to join all traffic data from all RSUs through a $\bowtie$ operation. Although it is possible to forward this joint data stream $\{S\}$ directly to the Cloud, two types of traffic data reduction are implemented: (i) based on sampling, and (ii) based on clustering.

3.2.2. Sampling methods

We leverage an adaptation of the sampling method from Tanizaki et al. [21], which we denote as *Rand*. Rand was adapted to accommodate a continuous data stream, and partitioning the data into a sliding-window. In the original work, only a static dataset was used in batch mode. In brief, *Rand* uses a randomized update function in which the vehicles transmit their speed data only when a probability p is less than 1. Thus, for each time interval (Δt), the randomization condition is checked, and when this condition is true, the data is transmitted. A function of p gives the randomization condition. It is the time measured between the beginning of the vehicle exceeding the speed limit for the first time and the transmission of the revised speed to the Cloud. Therefore, it is the sum of this time, considering two consecutive vehicles, multiplied by the probability $p * (1-p)^{i-1}$.

Following adaptation made for *Rand*, we also use the sampling method from Wahid et al. [24], which we denote as *Lim*. *Lim* only sends the current traffic data to the Cloud when the speed of the vehicle exceeds a predetermined threshold value (T) that is already stored in the vehicles. This decision is given by $\bar{s}$, which is the difference between the current speed ($s^{ct}_{n_i}$) and the previous speed ($s^{p}_{n_i}$) of a vehicle n_i. If $\bar{s}$ is greater than or equal to T in Δ, the data is sent to the Cloud, otherwise, it is discarded. Hence, the vehicle forwards the traffic data to the Cloud when $(s^{ct}_{n_i} - s^{p}_{n_i}) \geq T$, i.e., $\bar{s} \geq T$.

We also compare against two simple methods to cater to a continuous data stream environment: (i) a naive sampling method called $1to2$, which uses a rate of $[1:2]$, i.e., one out of two data is always sent to the Cloud, and (ii) a straightforward algorithm Baseline (cf. Algorithm 2). Baseline deals with traffic data without data reduction, i.e., it forwards all the traffic data collected from the VANET Layer to the Cloud Layer.

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

As stated before, the process of sending all the data received from the VANET Layer to the Cloud, allows us to carry out a factual comparative analysis of Baseline with other data reduction methods, with respect to traffic flow classification accuracy (i.e., LOS).

3.2.3. Clustering methods

In addition to sampling techniques, we also proposed two adapted-clustering techniques to accommodate for a continuous data stream. We leverage the density-based algorithm Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [53], and the X-Means algorithm based on K-Means [54]. Our contribution to these clustering methods lies in the technical and conceptual adaptation to use them in a continuous data stream environment. This enables the clustering methods to efficiently extract the traffic condition insights *"on-the-fly"* from the traffic data stream.

As shown in Algorithm 3, our Continuous Data Stream DBSCAN (CDS-DBSCAN) starts by visiting an arbitrary point ($P \in D$) that has not been visited before. Each point in P comprises of two information, the vehicle's position and the vehicle's identification. Then, all points for each neighborhood are retrieved, and as long as the minimum number of neighbors is satisfied, a group is formed. Otherwise, P is deemed as a noise. Note that this noise P can be found as part of other clusters for a different size of $\epsilon.neighborhood$, where $epsilon$, also known as radius, defines the maximum distance between two points to be considered as neighbors. This process continues until the density for all clusters is determined.

Algorithm 3: CDS-DBSCAN Algorithm

```
Require: Set of stream points ($D$) containing all positional
    vehicle information, $minPts$ and NeighborRadius $\epsilon$ in the time interval $\Delta t$
Ensure: Set of Clusters $k'$ assignment for each $\Delta t$
 1: Initialize clusters set $k = 0$
 2: for all new point $P$ arriving in the stream $\in D$ do
 3:   if P is visited then
 4:     $P_i = P_{i+1}$
 5:   else
 6:     $P = visited$
 7:   end if
 8:   $NhoodSet$ = all $P \in \epsilon$.neighborhood
 9:   if $|NhoodSet| < minPts$ then
10:     $P = noise$
11:   else
12:     $k_i = k_{i+1}$
13:     $k = P$
14:     for all $P' \in NhoodSet$ do
15:       if $P'$ is not visited then
16:         $P' = visited$
17:         $NhoodSet'$ = all $P' \in \epsilon$.neighborhood
18:         if $|NhoodSet'| \geq minPts$ then
19:           $NhoodSet = NhoodSet \oplus NhoodSet'$
20:         end if
21:       end if
22:       if $P' \notin k \leftarrow (k = 0, ..., k = n)$ then
23:         $k = P'$
24:       end if
25:     end for
26:   end if
27: end for
28: for all $k \leftarrow (k = 0, ..., k = n)$ do
29:   $k' = k.centroid \oplus k.NhoodSet(\lambda)$
30: end for
31: return ($k'$)
```

To deal with the continuous data stream environment, we adapted DBSCAN to use a sliding-window (cf. Section 3.2.1) as a mechanism to adjust flexible limits in the unlimited data stream, to seek a finite, but always a variable set of tuples. In this way, the timestamp values of the streaming tuples are checked for inclusion in a time interval (Δ), producing an approximate response to a query in the data stream. This approach enables DBSCAN to analyze parts of the recent data in the stream, instead of looking at all the history of the data stream. Therefore, the sliding-window gathers the incoming data in a pre-specified Δt, and when the elapsed time is greater than the sliding-window size γ, the accumulated data is processed for clustering.

After clustering, the reduction process automatically discards noise. The next step in this CDS-DBSCAN is to reduce the amount of traffic data sent to the Cloud. Therefore, based on the vehicles' density distribution on the road for each sliding-window γ, each cluster is transformed into a new smaller one, but maintains the similarity of original distribution. Nevertheless, before reducing the number of elements within each cluster, we choose the most representative data point within each cluster to act as a centroid. In ordinary DBSCAN, there is no notion of a centroid, hence, we inculcate this to provide a central element inside the cluster k with a reasonable distance from one another. The goal is to choose a spread set of points (λ) in each cluster with similarity (i.e., position and speed) from the centroid. As a result, we send a subset to the Cloud that represents more relevant elements for each cluster.

Although K-Means is widely used to identify clusters in data, it requires the number of clusters (i.e., k-number) be to know a priori.

Therefore, we adapted the X-Means algorithm [54], called CDS-X-Means (cf. Algorithm 4) to work in a continuous data stream environment. As X-Means, CDS-X-Means automatically sets k-number based on Bayesian Information Criterion (BIC) score.

Algorithm 4: CDS-X-Means Algorithm

```
Require: Set of stream points ($D$) containing all positional
   vehicle information in the time interval $\Delta t$, as well as $k_{max}$
Ensure: Set of clusters centroids $C = \{c_1, \dots, c_k\}$ assignment
   for each $\Delta t$
1: while $|C_k| > k_{max}$ do
2:    for all new point $p_i$ arriving in the stream $\in D$ do
3:       for all new centroid $c_k$ in the stream $\in C$ do
4:          ed = Euclidean_distance($p_i, c_k$)
5:          if ed($\tau_i$) < ed($\tau_{i-1}$) then
6:             $c_k = p_i$
7:             $update(c_k)$
8:          end if
9:       end for
10:   end for
11:   $Old_{BIC} = 0$
12:   for all new cluster formed $k$ in the stream $\in D$ do
13:      $old_{BIC} = log\ M_j(D) - \frac{1}{2}kp_j\ log\ n$
14:   end for
15:   $Sub_k = 2$
16:   for all new cluster formed $k$ in the stream $\in D$ do
17:      KMeans(D, $Sub_k$)
18:      for all new cluster $Sub_k$ in the stream $\in D$ do
19:         $New_{BIC} = log\ M_j(D) - \frac{1}{2}kp_j\ log\ n$
20:      end for
21:      if $New_{BIC} > Old_{BIC}$ then
22:         $k+ = 1$
23:         $C = update(c_k)$
24:      end if
25:   end for
26: end while
27: return ($C$)
```

CDS-X-Means uses 2-Means (i.e., K-Means with k=2) as a subroutine that repeatedly attempts to subdivide, while maintaining the best resulting clusters until BIC is reached. BIC is based on the log-likelihood of the dataset and it is computed using the following equation:

$$BIC(j) = log\ M_j(D) - \frac{1}{2}kp_j\ log\ n. \tag{2}$$

Where $M_j(D)$ is the maximal likelihood of D using the model j, kp_j is the number of parameters of the model, and $n = |D|$. Note that in Algorithm 4, we first cluster the traffic data within an interval time (Δt) using 2-Means, thus creating two clusters. Then, 2-Means is executed for each cluster of the initial clustering created. There are two centroids for new clusters, which are based on the centroids of the original cluster. According to the BIC metric, if any of the clusters in the initial clustering does not have a good representation of data, we replace them with their "sons". In case of no cluster being worse than their son, we choose a member of the cluster that best fits the data based on the adopted metrics. This process is repeated until the number of clusters exceed the maximum number K_{max} that is provided as input.

Therefore, FogJam reduces the volume of data communicated over the network, relieving the LTE/5G links between Fog and Cloud by using sampling and clustering methods. It helps alleviate data growth at its source, due to its proximity to the vehicles in the VANET environment. The storage and computational complexity of the clustering approaches are as follows. DBSCAN needs $O(n)$, while X-Means requires $O((n+K)m)$ of storage [55,56]. They both have a computational complexity of $O(n^2)$.

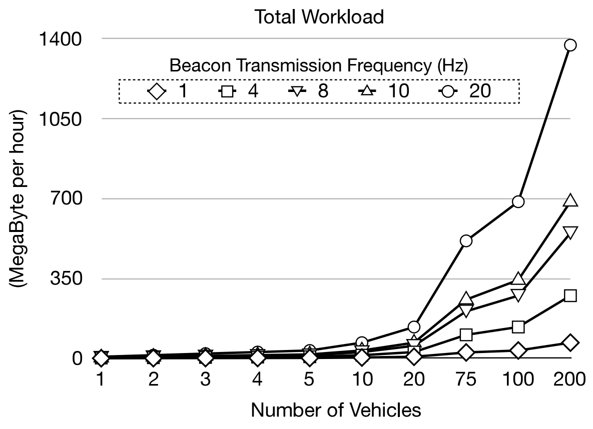


Fig. 5. Network Usage vs. Number of Vehicles, IEEE 802.11p-based vehicular network.

3.3. VANET layer

We model VANET as a directed and weighted graph $G = (V, E)$, where $V = \{v_1, v_2, \dots, v_{|V|}\}$ is the set of intersections and $E = \{e_1, e_2, \dots, e_{|E|}\}$ is the set of road segments connecting the intersections. Also, let $N = \{n_1, n_2, \dots, n_{|N|}\}$ represent the vehicles on the road. When a vehicle n_i, where i represents the ith vehicle $n_i \in N$, 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 $U_{o,p} = \{e_1, e_2, \dots, e_{|e|}\}$, where $|e|$ is the number of road segments in the route.

Each vehicle $n_i \in N$ periodically sends a beacon message (M_j) to the nearest RSU, where i represents the ith beacon sent by vehicle n_i, and M_j consists of a tuple including vehicle's position ($P_{x,y}$) and vehicle's identification (n_i). A single continuous data stream is given by $S_i = \sum_{i=0}^{|M|} M_i$ at time τ_i for $(M \in R_+^*) = \{i \in R | i \geq 0\}$. Besides, the probability distribution is non-stationary due to the changes in the traffic data generation process.

The VANET Layer is related to the IEEE 802.11p vehicular network's connectivity elements, offering a set of weighted graphs from road information, which is based on the spatial and temporal analysis. We consider a V2I (*Vehicle-to-Infrastructure*) communication model in this paper. However, this can be extended to other communication models that use an infrastructure to provide a global and centralized view of traffic congestion. V2I uses IEEE 802.11p to enable wireless access in vehicular environments (WAVE). IEEE 802.11p is based on dedicated short-range communications (DSRC) radio technology to exchange data between vehicles and infrastructure, i.e., RSUs. A RSU is typically attached along roads, at intersections, or near parking spaces, and uses beacon packets to exchange data with vehicles. A beacon is an IEEE 802.11p periodic message for exchanging situational information. RSUs and vehicles use it in the Control Channel (CCH), typically 27 bytes in length, but can reach up to 100 bytes [51,57,58].

To demonstrate the impact on network usage, Fig. 5 shows a toy example of network link utilization with 100 bytes beacon, depicting the amount of data that is transmitted from a single RSU/Fog to the nearest LTE/5G base station (BS). Although 5G networks are provisioned to support exponential growth in traffic, the regular LTE/5G network usage in an urban environment combined with VANET data growth, can burden the network. Note that the VANET data is impacted by the beacon transmission frequency and the number of vehicles.

FogJam assumes a beacon frequency of 1 Hz [57] to detect traffic congestion. The beacon frequency pertains to the number of beacons sent per second by a single vehicle to the RSU. We also assume that there are two modes of V2I communication to deal with traffic data. First, the vehicle sends the traffic data directly to the LTE/5G BS, which is forwarded to the Cloud. In this case, the BS acts as a Macro-Cell in the Cellular architecture. Second, the RSU receives the traffic

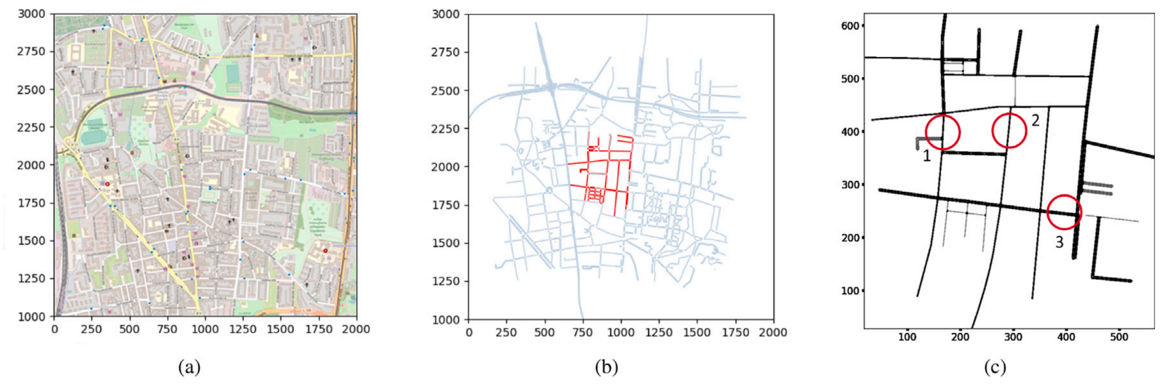


Fig. 6. (a) TAPAS Cologne map (b) Subset Cologne used in the simulation (c) Road Segments Chosen for Traffic Analysis.

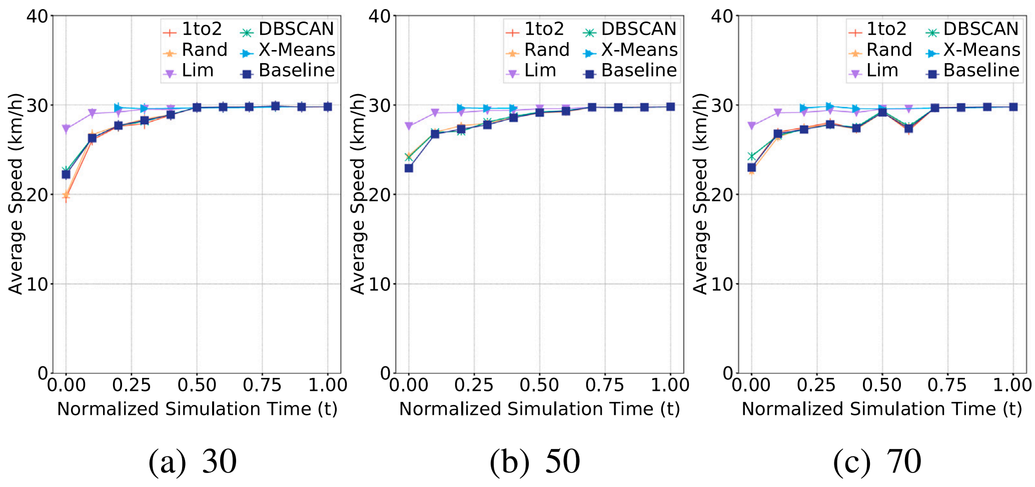


Fig. 7. Analyzing Speed (km/h) for Road Segment (1).

data via IEEE 802.11p and acts as a Small-Cell in the 5G Cellular architecture. The RSUs communicate with vehicles via WAVE to collect traffic information, such as time, speed, and vehicle location. The acquired traffic data is forwarded to the Fog Layer.

4. Results and discussion

In this section, we evaluated the performance of the sampling algorithms, mentioned in Section 3 (i.e., 1to2, Rand [21] and Lim [24]), along with the clustering methods (i.e., CDS-DBSCAN and CDS-X-Means) employed in FogJam. In addition, the Baseline algorithm [59, 60], a non-data-reducing method, is used as a baseline due to its standardization and accuracy in detecting traffic congestion levels. This section also presents the scenarios, simulation environment and setup, and the results obtained for vehicle speed, LOS, and network analysis, respectively.

4.1. Simulation setup

We perform evaluations using Veins [61], an open-source simulator that implements the standard IEEE 802.11p protocol stack for inter-vehicle communication. Veins is integrated into OMNet++ [62] to manage the network and connectivity. We also use the Simulation of Urban MObility (SUMO) [63], which is an open-source traffic simulator to model and manage objects in the road scenario. This allows us to reproduce the desired vehicle movements with random cruise speed and interactions according to empirical data.

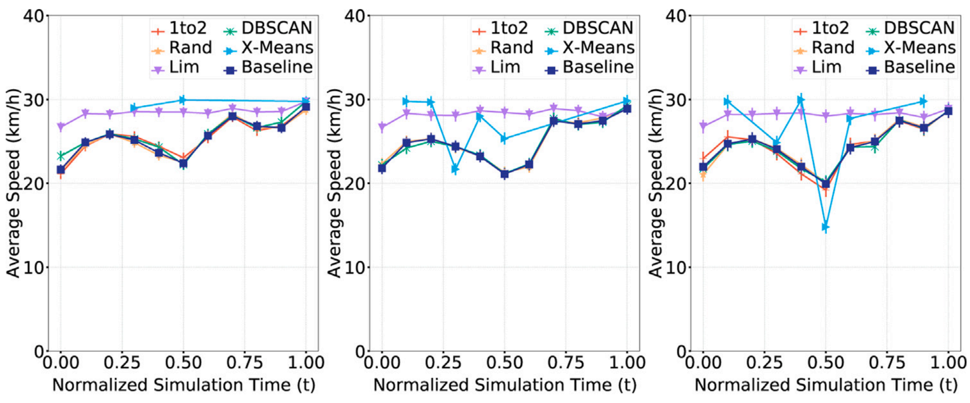
All experiments were performed for methods implemented in FogJam. We assume a network link between Fog and Cloud with a max rate of 340 (KB/s). Our scenario has a single RSU with a coverage of 1 km^2. The main simulation parameters are summarized in Table 3.

Table 3
Simulation parameters.

Parameters	Values
Simulation Area	4.6 km^2
Scenario	TAPASCologne [64]
Boundary Box	(12200,13000) (14500,15000)
Number of Vehicles	{30, 50 and 70}
Vehicle Speed	Mean: 30.05 km/h
Beacon Size	100 bytes
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

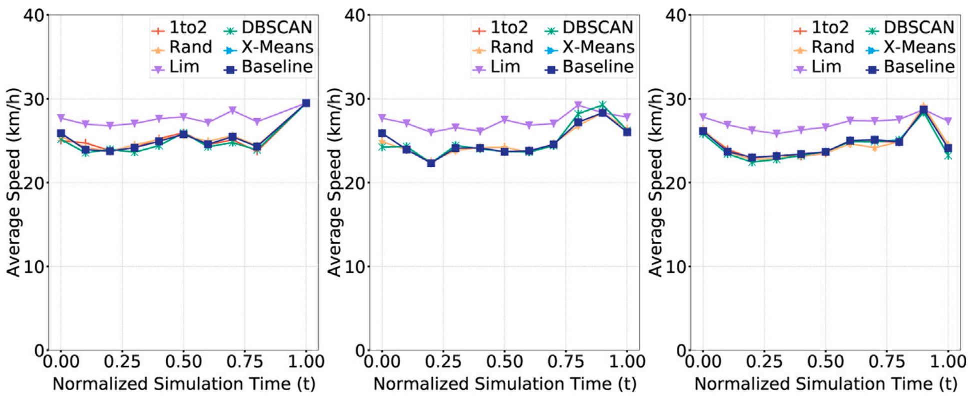
Our simulations use a subset of the TAPASCologne [64], which is an open-source project that provides a large-scale dataset with high realism for urban vehicular simulation that is based on SUMO. TAPAS-Cologne uses a realistic map of Cologne, Germany, obtained from OpenStreetMap,[1] as shown in Fig. 6(a). Vehicles used in the simulation share the same characteristics, such as the same size, mean, and standard deviation of speed. To perform the simulations, we used a central

[1] https://www.openstreetmap.org



(a) 30 (b) 50 (c) 70

Fig. 8. Analyzing Speed (km/h) for Road Segment (2).



(a) 30 (b) 50 (c) 70

Fig. 9. Analyzing Speed (km/h) for Road Segment (3).

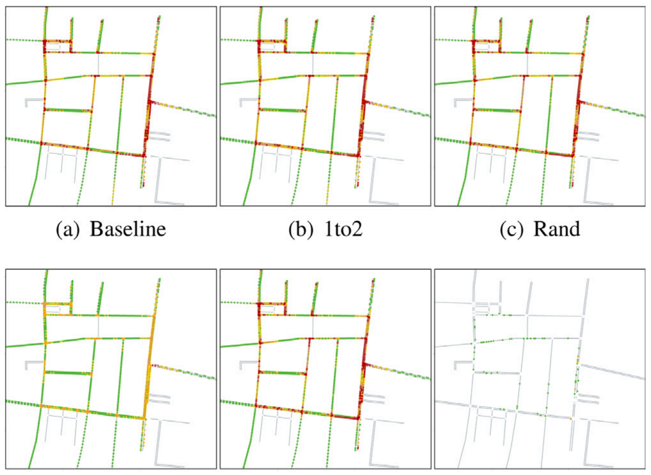
region of the data set, shown in red in Fig. 6(b), which exhibits a high density of vehicles.

Experiments were performed to observe the behavior of six different methods, which were discussed in Section 3. We use random routes for each vehicle, different vehicle densities, representing the active vehicles on the map during the entire simulation. These vehicle densities were chosen to characterize traffic congestion on the clipped road topology of Cologne. The experiments aim to analyze the accuracy in detecting congestion, given as LOS, and the impact on network cost.

4.2. Speed analysis

In this subsection, we analyze the behavior of the average speed *(km/h)* over the normalized simulation time for the Baseline, 1to2, Rand, Lim, CDS-DBSCAN, and CDS-X-Means methods, with different vehicular densities. Each zone of interest is marked in Fig. 6(c) with red circles (i.e., 1, 2, and 3), which highlights the chosen segments for the speed analysis.

We chose road segment (1), as it is the longest, with the highest number of intersections, and is one-way. It serves as an alternative to highways. Road segment (2) represents a main urban artery, an alternative route for those who travel between points 1 and 3, causing traffic retention by derivation. Also, it is a way between regions and has a single direction. Road segment (3) was chosen due to its intersection with a fast transit route. It is bidirectional and has a bottleneck that can generate traffic retention. Although we show only these three zones of interest for speed analysis in this paper, we cover the LOS and network analysis for the entire map.



(a) Baseline (b) 1to2 (c) Rand
(d) Lim (e) CDS-DBSCAN (f) CDS-X-Means

Fig. 10. Traffic Congestion Detection with density of 30 vehicles/km^2.

Figs. 7–9 show the average speed for road segment 1, 2 and 3, respectively, and the impact of the sampling and clustering methods for different number of vehicles. In general, it can be observed that 1to2 and Rand algorithms have a behavior that is similar to the Baseline, i.e., even with a reduction of data, these approaches represent the

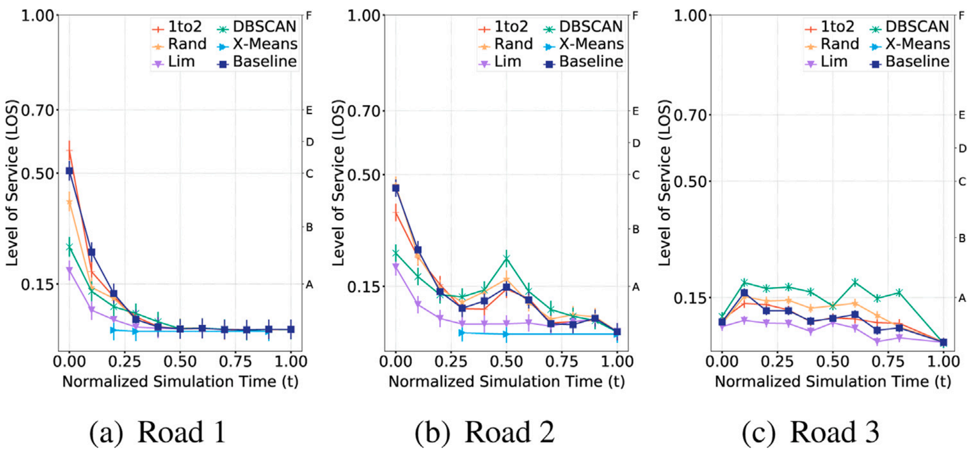


Fig. 11. Analyzing LOS for a density of 30 vehicles.

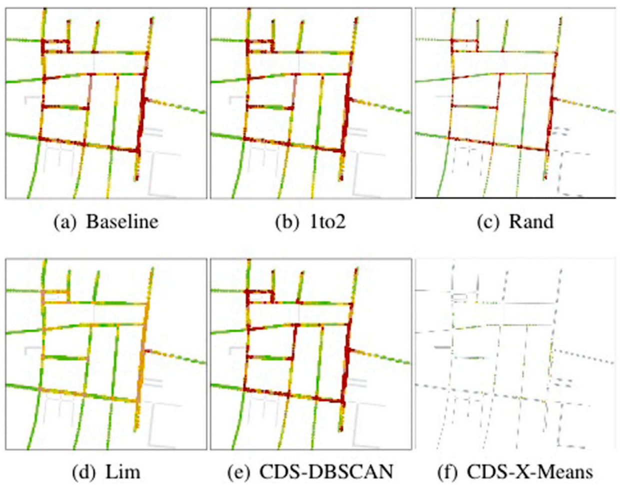


Fig. 12. Traffic Congestion Detection with density of 50 vehicles/km^2.

approximate average speed of Baseline. Although the minimum beacon transmission rate indicated for TCDS is 1 Hz, the number of beacons is higher than necessary to characterize the speed metric for this scenario. Therefore, the small data reduction provided by these methods does not impact the speed representation.

As aforementioned, Fig. 7 highlights a similar behavior for 1to2, Rand, and Baseline, showing that the speeds are practically overlapping. As for the clustering methods, there is a sensitive variation in speed, shown by Fig. 7A with the Normalized Simulation Time (NST) between 0.0 and 0.3 and by Fig. 7C with NST between 0.3 and 0.6, which indicates a short-time traffic congestion peak. Although clustering methods move further away from the original data for Fig. 7A and Fig. 7C due to the short time the road was congested, a slight similarity is observed in the trend between clustering techniques and Baseline. Short-time traffic congestion peaks may not be detected with high precision by the clustering algorithms, as the short-time interval may not be enough to receive all the traffic data that characterize a vehicle's spatial density. However, due to this rapid transience, this issue is not significant for ITS applications.

For all scenarios experimented, the Lim algorithm is furthest from the Baseline result due to the established threshold for traffic data transmission. Environments with high variations in speed cause a reduction in accuracy for the Lim algorithm when compared to Baseline. One way to improve Lim results would be to use a threshold representing the average speed acquired from historical traffic data. Except for Lim and CDS-X-Means, all other algorithms in Fig. 8 closely follow the behavior of the Baseline algorithm.

According to Fig. 9 (mainly A and C), it is observed that despite the high variation in speed, CDS-DBSCAN is capable of following the Baseline, even with speed ranging from 0 to 30 km/h. In this scenario, CDS-X-Means does not show good results when the density increases to 50 and 70, due to the low number of clusters created on the road map.

The speed analysis is essential to demonstrate the traffic data that each Fog Layer algorithm provides to the Cloud. With this type of traffic information, FogJam detects the source and classifies the level of congestion according to its severity, given by LOS. To provide a suitable TCDS, the algorithms must provide the correct speed tracking, which is used in the HCM Eq. (1). If speed observations are compromised, the reported accuracy of the traffic congestion level may also be compromised, as we will see in the next subsection.

4.3. LOS analysis

We adopted LOS as a metric to define the level of congestion on the roads. As explained in Section 3.1, it categorizes the traffic flow level based on parameters, such as speed, density, and delay. We applied the LOS metric for each algorithm with the aim to observe the level of accuracy when we use the different data reduction processes proposed in this work. For each data reduction approach, we need to identify the ability of a certain algorithm to keep the representativeness face an original dataset. Moreover, to observe the LOS behavior when applied within the context of traffic data flow, we analyze the LOS for each road segment chosen, and correlate it with a series of maps for different traffic densities (i.e., 30, 50, and 70 vehicles). It is worth noting that for each road segment, we performed a set of experiments with 10 replications for each using a 95% confidence interval.

Fig. 10 shows the LOS behavior for a density of 30 vehicles. Baseline, 1to2, and Rand algorithms have similar LOS behavior, presenting similar levels of congestion (cf. Table 2) in most cases. On the other hand, although CDS-DBSCAN does not cover all congestion points due to the low density of vehicles in this scenario, it detects most of the congestion on the road map.

When analyzing Fig. 11, we can see that in Road 1, all algorithms had similar behavior, showing high accuracy. In this scenario, the LOS goes down as a function of the low density of the road. On the other hand, Road 2 shows a loss of accuracy for the algorithms X-Means and Lim, which occurs also in Road 3. These roads have a larger traffic flow, which was not identified by the algorithms.

Fig. 12 shows the behavior of the algorithms when we increase the number of vehicles on the roads to 50 vehicles per km^2. Within this scenario, while the Lim algorithm maintains a lower congestion detection capability, we observed an improvement in the accuracy

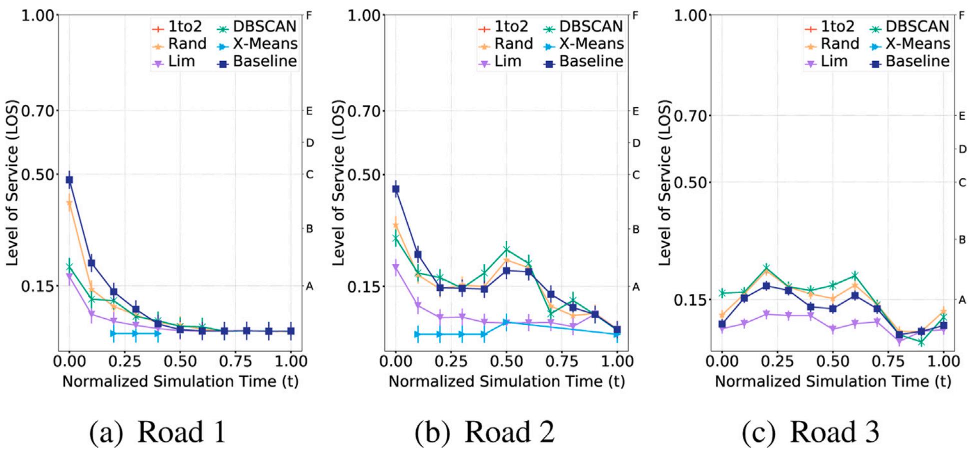


Fig. 13. Analyzing LOS for a density of 50 vehicles.

of CDS-X-Means, in comparison to Fig. 10. This is primarily due to the increase in the number of data points, which results in more clusters across the map. This increase in accuracy is also observed for CDS-DBSCAN.

In the same manner, analyzing Fig. 13 we observe that on Road 1 the behavior is similar between the algorithms. When we analyze Road 2, we see an increase in the traffic flow. As illustrated in Fig. 10, the algorithms X-Means and Lim have a lower accuracy than other algorithms. This behavior was also observed in Road 3.

When local traffic has 70 vehicles per km^2 (cf. Fig. 14), we can observe an increase in the congestion and, as a consequence, more presence of the E and F weights (w_i) of the LOS Table. In this scenario, the Lim algorithm has shown a higher efficiency in congestion detection than previous scenarios. This suggests that the application of this technique is more efficient in situations with a higher concentration of vehicles. Although CDS-X-Means has shown a slight improvement for this scenario, it is the CDS-DBSCAN algorithm that maintains the consistent high precision provided by LOS, accurately showing the points and congestion levels.

Analyzing Fig. 15, we can observe that on Road 1 the behavior is similar between the algorithms. When we analyze Road 2, we can see a significant increase in the traffic flow, the algorithms X-Means and Lim have a lower accuracy than other algorithms. In Road 3, we can observe a similarity among the algorithms, with a small variation in the accuracy of Lim.

Analyzing all three Figures (cf. 10, 12 and 14), we see Lim and CDS-X-Means algorithms with less LOS representativeness. Lim does not have all the traffic data necessary to accurately calculate the LOS, with many congestion points that are not identified on the road map. In the CDS-X-Means algorithm, only traffic data from each centroid is sent from the Fog to the Cloud that represent the entire cluster, leading to lack of information for traffic flow classification. It is worth mentioning that even when CDS-X-Means has a lower LOS compared to Baseline, it can still identify some points of congestion.

Although LOS is essential to ensure traffic flow classification, it is necessary to understand its cost for a TCDS. Therefore, the next subsection discusses the cost associated with traffic data communication between Fog and Cloud.

4.4. Network analysis

The continuous traffic data flow generated over time by vehicles is usually large in volume, leading to network congestion, packet losses, higher communication cost, wastage of bandwidth, and increase in delay. Network Bandwidth (NB) is given by S_B/T, where S_B represents the traffic data, and T is the channel bandwidth. S_B is also given by $(L_B + \lambda_B)$, where L_B is the sum of all packet sizes and λ_B is the average arrival rate. The higher the number of vehicles, the higher is the consumed NB. One way to approach this problem is to minimize L_B subject to the constraint $a_o \geq a_{min}$, where a_o is the accuracy obtained, and a_{min} is the minimum accuracy acceptable to detect traffic congestion.

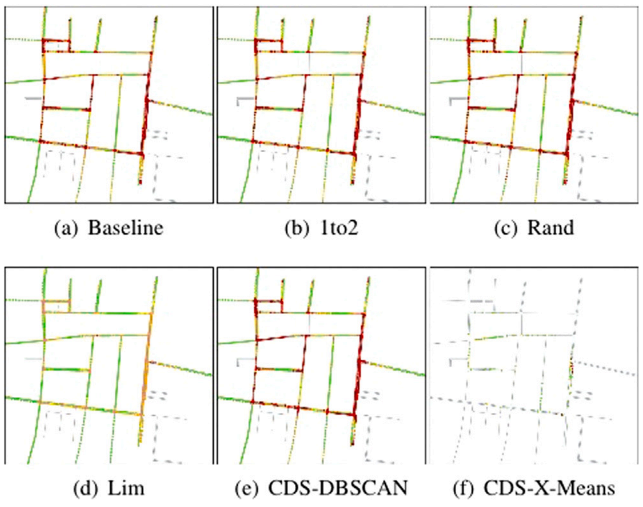


Fig. 14. Traffic Congestion Detection with density of 70 vehicles/km^2.

Fig. 16 shows the consumed NB for each method (i.e., Baseline, 1to2, Lim, CDS-X-Means, Rand, and CDS-DBSCAN), considering the vehicle density for the entire simulation time. In this way, we can see that the clustering algorithms are less costly with respect to network communication. The Baseline provides LOS with higher network usage, for the same reasons described above. On the other hand, the CDS-DBSCAN, CDS-X-Means, and Rand algorithms consumed less than 50% of network usage for all scenarios, and still maintained good LOS detection accuracy.

When there are a higher number of vehicles in the scenario, there are more possibilities of congestion and, consequently, more reduction in network consumption. Lim, 1to2, and Rand algorithms are able to reduce the network usage compared to the Baseline, but with a higher cost than the clustering algorithms. Both CDS-X-Means and CDS-DBSCAN reduce the traffic data by 60.85% and 75.61% compared to the Baseline, respectively. Besides, CDS-DBSCAN reduces 28.90% more traffic data than Rand, which is the best approach for traffic data reduction in the literature.

On analyzing the data reduction strategies proposed in this work facing the costs related to OPEX (Operational Expenditure), we observe

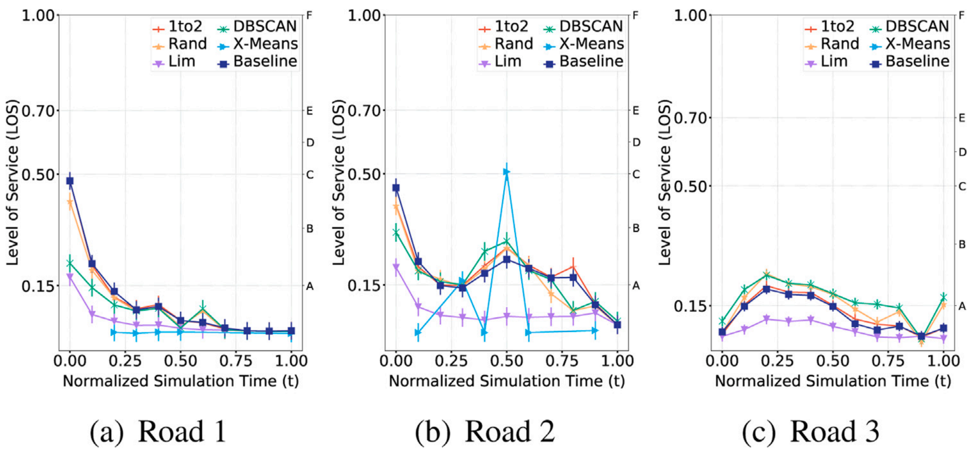


Fig. 15. Analyzing LOS for a density of 70 vehicles.

Table 4
Network usage.

Density km^2	30		50		70	
Algorithm	**NB (KB/s)**	**%**	**NB (KB/s)**	**%**	**NB (KB/s)**	**%**
Baseline	928.0	41.87	7.606	31.93	607.0	28.37
1to2	247.0	39.2	1.06	31.42	544.0	31.28
Rand	350.39	49.0	7.580	43.7	300.3	32.7
Lim	505.66	71.51	2.318	32.4	610.33	37.1
CDS-X-Means	183.34	0.94	8.934	2.20	816.30	1.88
CDS-DBSCAN	183.59	5.39	8.931	3.39	815.75	3.09

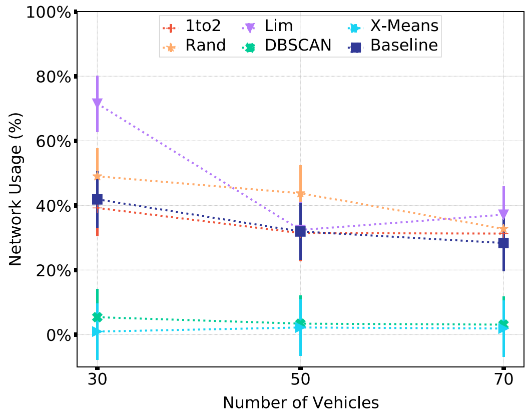


Fig. 16. Network Usage for Different Vehicular Density.

their practical implications from two concurrent perspectives. First, data storage, which is related to the amount of data required by an intelligent transportation system to analyze and identify eventual changes in traffic behavior. Second, communication cost, which is related to the data traffic in the network during monitoring and management of urban spaces. Within this context, approaches such as DBSCAN and X-Means can contribute to reducing the communication cost once they are able to make the same work using just a little part of the original dataset. DBSCAN was able to reduce about 70% of the network resource use, keeping a high level of accuracy. This strategy can represent a relevant reduction of the traffic data costs through the network. This action can contribute to the increase in the general quality of the service and generate better use of this infrastructure

Fig. 16 is summarized in Table 4. CDS-DBSCAN uses the lowest relative amount of network resources, and as shown in Figs. 10, 12 and 14, it is able to identify and classify the main congestion points. We can observe the growing behavior of network usage in Baseline, due to the increase in the number of vehicles, approaching 80% of the network usage. Similar behavior can be observed with 1to2 and Rand in higher vehicle density scenarios, and these methods tend to amplify network consumption.

According to simulation results, cluster-based methods provide a similar traffic flow (i.e., LOS) classification with less usage of the network than other approaches, such as 1to2, Rand, and Lim. This behavior is corroborated by a previous work [65]. In this paper, we studied the relationship between the network usage and the number of groups performed by DBSCAN. It was found that the increase in the number of vehicles and, consequently, in the number of clusters, indicates a lower number of noise points. Besides, with the increase in the number of vehicles, the number of vehicles per cluster also increases, proportionally reducing the amount of data sent per group. Therefore, we show that the number of clusters, vehicular density, and LOS is highly dependent on each other. In addition, despite the lower network usage, the aggressiveness of the cluster-based CDS-X-Means in reducing traffic data causes a significant loss in the LOS classification accuracy. However, CDS-DBSCAN shows that it is possible to address a multi-objective problem (accuracy and data reduction) for low or high-intensity traffic congestion.

5. Conclusion

Traffic congestion or Traffic Jam is one of the biggest issues for citizens living in large cities all over the world, leading to economic and social issues. The traffic congestion detection systems use a large amount of traffic data to measure the source and severity of traffic flow, increase network costs, and overload the existing network infrastructure, especially from the edge of the network to the Cloud. To overcome this issue, we have presented FogJam, a low communication cost Fog service to detect Traffic Congestion in a continuous data stream environment.

We showed that the literature's traditional sampling traffic data algorithms can reduce data, but cluster-based methods increase the amount of reduced traffic data while maintaining accuracy. Clustering methods were employed to reduce total network usage based on vehicle density and speed on the road. For Clustering methods, the more density, the less proportional information is transmitted from the Fog to the Cloud, due to the increasing number of vehicles per cluster. Although the CDS-X-Means cluster-based algorithm has presented a high data reduction rate, it achieved a poor result with respect to LOS accuracy. On the other hand, CDS-DBSCAN could offer high levels of reduction in traffic data, and efficiently recognize the source and severity of the main congestion points on the road map, even when the traffic is congested. FogJam using clustering-based methods was able to, on average, reduce the network usage by approximately 70% compared to the sampling methods, while maintaining an acceptable level of accuracy.

In our future work, we intend to minimize the effect of cluster-based algorithms on LOS accuracy for low traffic congestion. In this case, there are fewer cars on the road and therefore data reduction must be dynamic, adapting to the vehicle environment.

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.

Data availability

Data will be made available on request.

Acknowledgment

The authors would like to thank the São Paulo Research Foundation (FAPESP), Brazil, grants #2018/23126-3, #2019/19105-3, and #2015/24494-8.

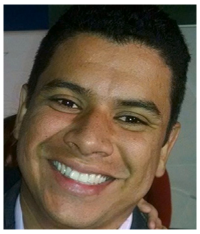
References

[1] D. Schrank, T. Lomax, B. Eisele, Urban Mobility Report 2019, Published By the Texas A&M Transportation Institute with Cooperation from INRIX Research, 2019.
[2] T. Reed, J. Kidd, Global traffic scorecard, 2019, Altrincham: INRIX Research.
[3] D. Schrank, L. Albert, B. Eisele, T. Lomax, Urban Mobility Report 2021, Published By the Texas A&M Transportation Institute with Cooperation from INRIX Research, 2021.
[4] N. Zhang, W. Jia, P. Wang, C.-H. Dung, P. Zhao, K. Leung, B. Su, R. Cheng, Y. Li, Changes in local travel behaviour before and during the COVID-19 pandemic in Hong Kong, Cities (2021) 103139.
[5] D.A. Hensher, E. Wei, M. Beck, C. Balbontin, The impact of COVID-19 on cost outlays for car and public transport commuting-the case of the greater Sydney Metropolitan Area after three months of restrictions, Transp. Policy 101 (2021) 71–80.
[6] R. Tomás, P. Fernandes, J. Macedo, M.C. Coelho, Carpooling as an immediate strategy to post-lockdown mobility: A case study in university campuses, Sustainability 13 (10) (2021).
[7] H. Smith, L.A. traffic behavior is changing. Is post-pandemic gridlock inevitable?, 2021, https://www.latimes.com/california/story/2021-06-18/post-covid-l-a-traffic-analysis-has-rush-hour-changed. (Accessed 14 September 2021).
[8] ITDP, The next pandemic surge: Traffic, 2021, Institute for Transportation and Development Policy: https://www.itdp.org/2021/03/22/the-next-pandemic-surge-traffic/. (Accessed 14 September 2021).
[9] LGC, Curbing post-pandemic traffic, 2021, Local Government Commission: https://www.lgc.org/newsletter/curbing-post-pandemic-traffic/. (Accessed 14 September 2021).
[10] Y. Zheng, L. Capra, O. Wolfson, H. Yang, Urban computing: Concepts, methodologies, and applications, ACM Trans. Intell. Syst. Technol. 5 (3) (2014) 38:1–38:55.
[11] M. Peixoto, A. Maia, E. Mota, E. Rangel, D. Costa, D. Turgut, L. Villas, A traffic data clustering framework based on fog computing for VANETs, Veh. Commun. 31 (2021) 100370.
[12] IDC2018, Worldwide Internet of Things Forecast Update, 2017–2021, Document #US43304017, IDC, Framingham, MA, USA, 2018.
[13] International Renewable Energy Agency(IRENA), Innovation landscape brief: Internet of Things, 2019.
[14] M. Arcaro, Worldwide Connected Vehicle Forecast, 2019–2023, Document #US45050319, IDC, Framingham, MA, USA, 2019.
[15] P. Desai, A survey on big data applications and challenges, in: 2018 Second International Conference on Inventive Communication and Computational Technologies, ICICCT, 2018, pp. 737–740.
[16] M. Peixoto, T. Genez, L.F. Bittencourt, Hierarchical scheduling mechanisms in multi-level fog computing, IEEE Trans. Serv. Comput. 15 (5) (2022) 2824–2837.
[17] V. Cerqueira, L. Moreira-Matias, J. Khiari, H. van Lint, On evaluating floating car data quality for knowledge discovery, IEEE Trans. Intell. Transp. Syst. (2018) 1–12.
[18] M. Rath, B. Pati, B.K. Pattanayak, Mobile agent-based improved traffic control system in VANET, in: A. Krishna, K. Srikantaiah, C. Naveena (Eds.), Integrated Intelligent Computing, Communication and Security, Springer Singapore, Singapore, 2019, pp. 261–269.
[19] X. Lei, S. Rhee, Performance analysis and enhancement of IEEE 802.11p beaconing, EURASIP J. Wireless Commun. Networking 2019 (2019).
[20] B.S. Kerner, C. Demir, R.G. Herrtwich, S.L. Klenov, H. Rehborn, M. Aleksic, A. Haug, Traffic state detection with floating car data in road networks, in: Proceedings. 2005 IEEE Intelligent Transportation Systems, 2005, 2005, pp. 44–49.
[21] M. Tanizaki, O. Wolfson, Randomization in traffic information sharing systems, in: Proceedings of the 15th Annual ACM International Symposium on Advances in Geographic Information Systems, GIS '07, Association for Computing Machinery, New York, NY, USA, 2007.
[22] S. Basagni, M. Conti, S. Giordano, I. Stojmenovic, The next paradigm shift: From vehicular networks to vehicular clouds, in: Mobile Ad Hoc Networking: The Cutting Edge Directions, 2013, pp. 645–700.
[23] K. Bilal, O. Khalid, A. Erbad, S.U. Khan, Potentials, trends, and prospects in edge technologies: Fog, cloudlet, mobile edge, and micro data centers, Comput. Netw. 130 (2018) 94–120.
[24] A. Wahid, A.C.S. Rao, D. Goel, Server communication reduction for GPS-based floating car data traffic congestion detection method, in: A. Krishna, K. Srikantaiah, C. Naveena (Eds.), Integrated Intelligent Computing, Communication and Security, Springer Singapore, Singapore, 2019, pp. 415–425.
[25] T. Kumar, D. Kushwaha, An efficient approach for detection and speed estimation of moving vehicles, Proc. Comput. Sci. 89 (2016) 726–731.
[26] T. Kumar, D.S. Kushwaha, An approach for traffic congestion detection and traffic control system, in: S. Fong, S. Akashe, P.N. Mahalle (Eds.), Information and Communication Technology for Competitive Strategies, Springer Singapore, Singapore, 2019, pp. 99–108.
[27] T. Idé, T. Katsuki, T. Morimura, R. Morris, City-wide traffic flow estimation from a limited number of low-quality cameras, IEEE Trans. Intell. Transp. Syst. 18 (4) (2017) 950–959.
[28] J. Cheng, B. Liu, X. Tang, A traffic-congestion detection method for bad weather based on traffic video, in: K. Li, J. Li, Y. Liu, A. Castiglione (Eds.), Computational Intelligence and Intelligent Systems, Springer Singapore, Singapore, 2016, pp. 506–518.
[29] C. Lam, H. Gao, B. Ng, A real-time traffic congestion detection system using on-line images, in: 2017 IEEE 17th International Conference on Communication Technology, ICCT, 2017, pp. 1548–1552.
[30] A. Rao, A. Phadnis, A. Patil, T. Rajput, P. Futane, Dynamic traffic system based on real time detection of traffic congestion, in: 2018 Fourth International Conference on Computing Communication Control and Automation, ICCUBEA, 2018, pp. 1–5.
[31] Y. Li, H. Wang, B. Buckles, Traffic Congestion Assessment Based on Street Level Data for On-Edge Deployment, Association for Computing Machinery, New York, NY, USA, 2019.
[32] G.S.R. Satyanarayana, S. Majhi, S.K. Das, A vehicle detection technique using binary images for heterogeneous and lane-less traffic, IEEE Trans. Instrum. Meas. 70 (2021) 1–14.
[33] N. Anjum, N. Badruddin, M. Drieberg, Simulation of traffic congestion detection using VANETs, in: 2014 5th International Conference on Intelligent and Advanced Systems, ICIAS, 2014, pp. 1–4.
[34] M. Lv, L. Chen, G. Chen, D. Zhang, Detecting traffic congestions using cell phone accelerometers, in: Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication, in: UbiComp '14 Adjunct, Association for Computing Machinery, New York, NY, USA, 2014, pp. 107–110.
[35] R. Wang, D.B. Work, R. Sowers, Multiple model particle filter for traffic estimation and incident detection, IEEE Trans. Intell. Transp. Syst. 17 (12) (2016) 3461–3470.

[36] S. Abraham, N. Weller, Next generation traffic event detection: A framework based on internet of vehicles (IoV), in: Proceedings of the Second International Conference on Internet of Things, Data and Cloud Computing, ICC '17, Association for Computing Machinery, New York, NY, USA, 2017.
[37] E. D'Andrea, F. Marcelloni, Detection of traffic congestion and incidents from GPS trace analysis, Expert Syst. Appl. 73 (2017) 43–56.
[38] M.H. Ahmadzadegan, H.A. Deilami, M. Izadyar, H. Ghorbani, Implementation and evaluation of the impact of traffic congestion on the detection of the missing packets in VANET, in: 2019 Third International Conference on I-SMAC, IoT in Social, Mobile, Analytics and Cloud I-SMAC, 2019, pp. 169–172.
[39] A. Dimri, H. Singh, N. Aggarwal, B. Raman, K.K. Ramakrishnan, D. Bansal, BaroSense: Using Barometer for Road Traffic Congestion Detection and Path Estimation with Crowdsourcing, Vol. 16, no.1, Association for Computing Machinery, New York, NY, USA, 2019.
[40] S. Wang, X. Zhang, F. Li, P.S. Yu, Z. Huang, Efficient traffic estimation with multi-sourced data by parallel coupled hidden Markov model, IEEE Trans. Intell. Transp. Syst. 20 (8) (2019) 3010–3023.
[41] S. Seid, M. Zennaro, M. Libsie, E. Pietrosemoli, P. Manzoni, A low cost edge computing and LoRaWAN real time video analytics for road traffic monitoring, in: 2020 16th International Conference on Mobility, Sensing and Networking, MSN, 2020, pp. 762–767.
[42] S.H. Ramachandra, K.N. Reddy, V.R. Vellore, S. Karanth, T. Kamath, A novel dynamic traffic management system using on board diagnostics and zigbee protocol, in: 2016 International Conference on Communication and Electronics Systems, ICCES, 2016, pp. 1–6.
[43] P. Rani, D.K. Shaw, A hybrid approach for traffic delay estimation, in: A. Krishna, K. Srikantaiah, C. Naveena (Eds.), Integrated Intelligent Computing, Communication and Security, Springer Singapore, Singapore, 2019, pp. 243–250.
[44] K. Chen, Z. Li, Prediction of traffic state based on fuzzy logic in vanet, Inform. Technol. J. 12 (18) (2013) 4642–4646.
[45] 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.
[46] Y. Wang, P. Siriaraya, Y. Kawai, T. Akiyama, Twitter-based traffic delay detection based on topic propagation analysis using railway network topology, Pers. Ubiquitous Comput. 23 (2) (2019) 233–247.
[47] C. Lai, L. Wang, J. Chen, X. Meng, K. Zeitouni, Effective density queries for moving objects in road networks, in: APWeb/WAIM'09, Springer-Verlag, Berlin, Heidelberg, 2007, pp. 200–211.
[48] A. Amelia, G.A.P. Saptawati, Detection of potential traffic jam based on traffic characteristic data analysis, in: 2014 International Conference on Data and Software Engineering, ICODSE, 2014, pp. 1–5.
[49] Y. Zhang, N. Ye, R. Wang, R. Malekian, A method for traffic congestion clustering judgment based on grey relational analysis, ISPRS Int. J. Geo-Inf. 5 (5) (2016) 71.
[50] 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.
[51] Doreswamy, O.A. Ghoneim, Traffic jams detection and congestion avoidance in smart city using parallel K-means clustering algorithm, in: Proceedings of International Conference on Cognition and Recognition, Springer Singapore, Singapore, 2018, pp. 21–30.
[52] T. Board, Highway Capacity Manual, National Research Council, 2010.
[53] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: ACM SIGKDD'2, KDD '96, AAAI Press, 1996, pp. 226–231.
[54] D. Pelleg, A.W. Moore, X-Means: Extending K-means with efficient estimation of the number of clusters, in: Proceedings of the Seventeenth International Conference on Machine Learning, ICML '00, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2000, pp. 727–734.
[55] M.K. Pakhira, A linear time-complexity k-means algorithm using cluster shifting, in: 2014 International Conference on Computational Intelligence and Communication Networks, IEEE, 2014, pp. 1047–1051.
[56] K. Khan, S.U. Rehman, K. Aziz, S. Fong, S. Sarasvady, DBSCAN: Past, present and future, in: The Fifth International Conference on the Applications of Digital Information and Web Technologies, ICADIWT 2014, IEEE, 2014, pp. 232–238.
[57] R.K. Schmidt, T. Leinmuller, E. Schoch, F. Kargl, G. Schafer, Exploration of adaptive beaconing for efficient intervehicle safety communication, IEEE Netw. 24 (1) (2010) 14–19.
[58] M.E. Renda, G. Resta, P. Santi, F. Martelli, A. Franchini, IEEE 802.11 p VANets: Experimental evaluation of packet inter-reception time, Comput. Commun. 75 (2016) 26–38.
[59] A.M. de Souza, L.A. Villas, A fully-distributed traffic management system to improve the overall traffic efficiency, in: Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, MSWiM '16, ACM, New York, NY, USA, 2016, pp. 19–26.
[60] 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.
[61] VEINS, Vehicles in Network Simulation, 2021, Available: < http://veins.car2x.org >. (Accessed: 15 April 2021).
[62] OMNET++, Omnet++ - Network Simulation Framework. Available: . (Accessed: 15 April 2021), 2021, https://Www.Omnetpp.Org/.
[63] SUMO, SUMO - Simulation of Urban Mobility, 2021, Available: < http://sumo.sourceforge.net/ >. (Accessed: 15 April 2021).
[64] S. Uppoor, O. Trullols-Cruces, M. Fiore, J.M. Barcelo-Ordinas, Generation and analysis of a large-scale urban vehicular mobility dataset, IEEE Trans. Mob. Comput. 13 (5) (2013) 1061–1075.
[65] 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.

Maycon Leone Maciel Peixoto received the master's and the Ph.D. degrees in computer science from the University of Sao Paulo, Brazil, in 2008 and 2012, respectively. He performed a postdoctoral researcher with the University of Campinas in 2020, Brazil. He is also a professor with the Institute of Computing of the Federal University of Bahia. His research interests mainly include urban computing, smart grids, vehicular ad hoc networks, performance evaluation, cloud, edge, and fog computing.

Edson Mota is a Ph.D. Candidate at the Department of Computer Science of the Federal University of Bahia (UFBA). He holds a Master's degree in Systems and Computation at the Salvador University (UNIFACS), Brazil, 2015, where he acted as a software engineering researcher. Currently, he is acting at the Urban Computing Group, working with distributed systems. His research interests include Fog Computing, Visualization, Machine Learning, Performance Evaluation, and vehicular ad hoc networks.



Adriano Maia is a Master Degree student in Computer Science at the Department of Computer Science of the Federal University of Bahia (UFBA). He holds a University Graduate in Information Systems at the Estacio de Sa University, Brazil, 2017. Currently, he acting at the Urban Computing Group working with vehicular networks and distributed systems. His research interests include Cloud Computing, Fog Computing, Performance Evaluation and vehicular ad hoc networks.

Wellington Lobato Junior is a Ph.D. student at the Institute of Computer Science at State University of Campinas (UNICAMP). He holds a Master's degree in Electronical Engineering at the Federal University of Pará (UFPa), Brazil, 2019. Currently, he is acting at the Computer Networks Laboratory, working with connected autonomous vehicles and vehicle networks. His research interests include connected autonomous vehicles, routing algorithms, wireless networks, and vehicular ad hoc networks.

Mohammad A. Salahuddin is a research assistant professor of computerscience at the University of Waterloo. He received his Ph.D. in computer science from Western Michigan University in 2014. His current research interests include VANETs, Internet of Things, content delivery networks, network softwarization, cloud computing, cognitive network management, and network security. He serves as a TPC member for international conferences and is a reviewer for various peer-reviewed journals and magazines.

Raouf Boutaba received the M.Sc. and Ph.D. degrees in computer science from Sorbonne University in 1990 and 1994, respectively. He is currently a University Chair Professor and the Director of the David R. Cheriton School of Computer science at the University of Waterloo (Canada). He also holds an INRIA International Chair in France. He is the founding Editorin-Chief of the IEEE Transactions on Network and Service Management (2007–2010) and the current Editor-in-Chief of the IEEE Journal on Selected Areas in Communications. He is a fellow of the IEEE, the Engineering Institute of Canada, the Canadian Academy of Engineering, and the Royal Society of Canada.

Leandro Aparecido Villas holds a Ph.D. in Computer Science from the Federal University of Minas Gerais, Brazil, 2012. Currently, he is a Professor of Computer Science at the State University of Campinas (UNICAMP), Brazil. He received his Master's Degree in Computer Science from the Federal University of São Carlos, Brazil, in 2007. He has been a Visiting Ph.D. student in the PARADISE Research Laboratory, University of Ottawa, Canada, in 2011. His research interests include distributed algorithms, routing algorithms, wireless sensor networks, vehicular ad hoc networks. In the last five years, he has published over 70 papers in international conferences and journals and presented several papers in international conferences.