Expert Systems With Applications 73 (2017) 43–56

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Detection of traffic congestion and incidents from GPS trace analysis

Eleonora D'Andrea, Francesco Marcelloni*

Dipartimento di Ingegneria dell'Informazione, University of Pisa, Largo Lucio Lazzarino 1, 56122 Pisa, Italy

ARTICLE INFO

Article history:
Received 6 June 2016
Revised 11 December 2016
Accepted 12 December 2016
Available online 16 December 2016

Keywords:
Expert systems
GPS
Incident detection
Traffic congestion detection
Urban mobility simulation

ABSTRACT

This paper presents an expert system for detecting traffic congestion and incidents from real-time GPS data collected from GPS trackers or drivers' smartphones. First, GPS traces are pre-processed and placed in the road map. Then, the system assigns to each road segment of the map a traffic state based on the speeds of the vehicles. Finally, it sends to the users traffic alerts based on a spatiotemporal analysis of the classified segments. Each traffic alert contains the affected area, a traffic state (e.g., *incident, slowed* traffic, *blocked* traffic), and the estimated velocity of vehicles in the area. The proposed system is intended to be a valuable support tool in traffic management for municipalities and citizens. The information produced by the system can be successfully employed to adopt actions for improving the city mobility, e.g., regulate vehicular traffic, or can be exploited by the users, who may spontaneously decide to modify their path in order to avoid the traffic jam. The elaboration performed by the expert system is independent of the context (urban o non-urban) and may be directly employed in several city road networks with almost no change of the system parameters, and without the need for a learning process or historical data. The experimental analysis was performed using a combination of simulated GPS data and real GPS data from the city of Pisa. The results on incidents show a detection rate of 91.6%, and an average detection time lower than 7 min. Regarding congestion, we show how the system is able to recognize different levels of congestion depending on different road use.

1. Introduction

Intelligent Transportation Systems (ITSs) are nowadays attracting growing attention due to the increasing number of vehicles (mainly private cars) causing frequent traffic congestions, bottlenecks and incidents. An ITS is an infrastructure which, by integrating ICTs (Information and Communication Technologies) with transport networks, vehicles and users, allows improving safety and management of transport networks. ITSs provide, e.g., real-time information about weather, traffic congestion and regulation, and/or plan efficient (e.g., shortest, fast driving, less polluting) routes (Anastasi et al., 2013; Boriboonsomsin, Barth, Zhu, & Vu, 2012; Perea & Dias, 2011; Sakaki, Okazaki, & Matsuo, 2013). A challenging issue in ITSs is to distinguish an incident from a traffic congestion situation. An *incident* is defined in Kamran and Haas (2007) and Srinivasan, Jin, and Cheu (2004) as "an unexpected event that temporarily disrupts the traffic flow on a segment of a roadway". An Incident Detection System (IDS) represents a part of the ITSs, which has received significant attention in the last years. IDSs are designed to detect, by means of several technologies, incidents, or unexpected situations causing traffic congestions and bottlenecks in order to plan appropriate alternatives (Basnayake, 2004; Kamran & Haas, 2007). An accurate and early detection of incidents becomes of the utmost importance in traffic management (Srinivasan et al., 2004).

Mainly, two types of ITS have been proposed. The first type is the most frequently used and exploits data collected from fixed sensors placed in the road, such as closed-circuit cameras, video recognition cameras, infrared sensors, inductive loop sensors, RFID sensors (Calderoni, Maio, & Rovis, 2014; Cheng, Gau, Huang, & Hwang, 2012;Wen, 2010). The second type is based on sensor technology within the vehicle (e.g., Global Positioning System (GPS), acceleration sensors, crash sensors, airbag activation detection) (Kamran & Haas, 2007) and allows obtaining a greater amount of information than the first type (Bacon et al., 2011). Indeed, fixed sensors placed in the road collect data related to the total volume of traffic on a certain road, while mobile sensors collect data belonging to the trajectory of the single car (Work & Bayen, 2008), which leaves a *trace*, i.e., a sequence of positions recorded in exact time instants. A GPS trace is a sequence of GPS points, each of which contains location information coordinates (latitude and longitude), recording timestamp, and instantaneous velocity. Nowadays, thanks to the growing number of GPS trackers installed in vehicles and to the widespread use of mobile devices (e.g., smart-

* Corresponding author.
E-mail addresses: eleonora.dandrea@for.unipi.it (E. D'Andrea), f.marcelloni@iet.unipi.it, francesco.marcelloni@unipi.it (F. Marcelloni).

http://dx.doi.org/10.1016/j.eswa.2016.12.018
0957-4174/

phones, tablets) equipped with GPS receivers, we are able to collect directly and easily traces from vehicles. The use of GPS trackers and mobile devices as sensor probes has several advantages. First, we can exploit the existing communication infrastructure and the rapid spread of mobile devices with integrated GPS sensor. Second, the use of mobile devices as sensor probes is independent of the device and the communication service provider (Work & Bayen, 2008). Third, we can overcome the main drawbacks of traditional monitoring systems (e.g., fixed sensors), namely, limited coverage of the road network, and high installation and maintenance costs (Yoon, Noble, & Liu, 2007).

In this paper we present an ITS able to detect in real-time traffic congestions and incidents from the analysis of GPS traces belonging to moving vehicles. The proposed system exploits data mining (on spatiotemporal data) and expert system technologies in order to detect the intensity of traffic congestion on segments of the road network, where an event (i.e., an incident) is causing traffic difficulties. The proposed system is intended to be a valuable support tool in traffic management for municipalities and citizens. Traces collected from vehicles moving in the city are analyzed in real-time by means of an expert system, without the need for a learning process or historical data. The elaboration performed by the expert system is independent of the context (urban o non-urban) and may be directly employed in several city road networks with almost no change of the system parameters. The information produced by the system can be successfully employed to adopt actions for improving the city mobility. In fact, the possibility of monitoring real-time events in the city road network may allow municipalities and police to, e.g., regulate vehicular traffic in order to divert vehicles from the incident location, act on traffic lights and speed limits, alert emergency services, adjust the demand and capacity of the road network, and dynamically provide route information. Further, the information provided by the system may be useful also for the user, who once registered to the system, receives alert notifications about traffic congestions or incidents and may spontaneously decide to modify her/his path in order to avoid the traffic jam. For all the above reasons, the system may arise in the context of ITSs as a simple (low-cost and low-complexity) yet effective model.

In the following, we highlight the possible challenges in building and using the proposed system, along with possible ways to face them:

i) The first challenge regards the difficulty of obtaining GPS data from users moving in the city due to privacy issues. However, recently the use of GPS trackers and GPS-equipped mobile devices (e.g., smartphones, tablets) has dramatically grown, and, in a near future, GPS data should be more easily available, taking appropriate actions aimed at respecting the users' privacy. In particular, GPS trackers in vehicles include also black-boxes installed by insurance companies. In U.S., about 96% of new cars are black-box equipped and, since September 1, 2014, every new vehicle must have one black-box installed (Globaltruth, 2016). In Italy, about 3 millions of black-boxes were installed in vehicles in 2014 (Ania, 2014), and this value is growing so much that the use of black-boxes could become mandatory by 2017. Thus, collecting GPS data from black-boxes will not be a problem in a near future. In addition, since the proposed ITS is supposed to be available to registered users (e.g., by means of a smartphone app), GPS traces can also be collected directly from users, in exchange for receiving traffic alerts. In fact, on registration, the user accepts the privacy policy associated with the service, according to which the user's GPS position is used only anonymously for the aim of the service;

ii) The second challenge is the capability to effectively distinguish an incident from a traffic congestion event. Indeed, if the incident resolves in a very short time, the system may not be able to correctly detect the incident event, since the incident itself can be considered not actually relevant for the impact on the city mobility. The system, in this case, will detect and indicate instead a traffic congestion event. This is an effective way to face this challenge. Indeed, the users will be notified in any case that a traffic bottleneck is occurring, allowing them, e.g., to modify their planned route to avoid it. Anyway, our system, by checking specific incident-related conditions (as described in Section 4) is able to detect 91.6% of the incidents simulated with different time durations, and takes into account the urban scenario, more challenging than the non-urban scenario (Dia & Thomas, 2011);

iii) The third challenge regards the scalability of the system: the bigger the size of the monitored city road network, the higher the number of road segments to take into account in the elaboration. To cope with this challenge, the system, designed and developed from the ground, has been implemented as a service of a wider service-oriented platform in the context of the SMARTY project (Smarty Project, 2015). The platform is built on a Service Oriented Architecture (SOA), which allows exploiting two important peculiarities, i.e., scalability of the service (e.g., by using a dedicated data elaboration server for each geographic area), and easy integration with other ITS services (e.g., smart parking, weather information, bike sharing). Further, the scalability and availability of our system are guaranteed by using a cloud computing architecture and different instances of the service for any specific geographic area. The subdivision in geographic areas is mainly based on the size of the area, and on the number of road segments in the area. Further it is dynamically adjusted based on the level of vehicles' activity in the area. In this way, the service is able to guarantee the fulfilment of minimum Quality of Service (QoS) requirements, e.g., response time, availability, capacity. The instances of the services, by synchronizing and communicating with each other, produce as output the traffic notifications, sent to the users. The users need only to register to the service for receiving notifications.

To simulate traffic conditions as close as possible to reality, the GPS traces employed in the experiments were generated exploiting the traffic simulator SUMO (Simulation of Urban Mobility) (Krajzewicz, Hertkorn, Rössel, & Wagner, 2002), based on a set of real-world GPS traces collected from smartphones (or other tracking devices) belonging to people moving by car.

The remainder of this paper is organized as follows. Sections 2 and 3 discuss, respectively, related work and road network representations. Section 4 presents the proposed system for incident and traffic congestion detection from GPS trace analysis. Section 5 describes the generation of the GPS trace dataset employed in the experiments. In Section 6 the experimental results are discussed. Lastly, Section 7 draws some concluding remark and future work.

2. Related work

In the literature several types of algorithms for detecting incidents and traffic conditions in ITS systems exist, e.g., prediction algorithms, statistical methods, model-based identification algorithms, traffic flow models, computational intelligence techniques, and rule-based expert systems (Kamran & Haas, 2007;Srinivasan et al., 2004). Moreover, some approaches use only real-time collected data, while other approaches also need a historical database

to build the model. In the following we provide a brief description of the most relevant works existing in the literature, which propose methods for incident detection and traffic monitoring based on GPS traces.

Liu, Liu, Ni, Li, and Fan (2013), present a mobility-based clustering algorithm that uses some vehicles as sensor probes for establishing the level of crowdedness of vehicles in nearby areas (spots) of the city. Li, Han, Lee, and Gonzalez (2007), propose a density-based algorithm (FlowScan) to study the traffic flow on the road network and identify densely congested routes, through a clustering of the road segments having similar traffic density. Bacon et al. (2011), exploit real-time traffic data coming from fixed and mobile sensors (buses are used as mobile sensor probes) to evaluate, using statistical techniques, the level of traffic congestion in the city, and to provide citizens with traffic information. Kamran and Haas (2007), use real-time GPS data obtained from vehicles to detect incidents. They divide roads into segments whose size is set based on the type of road and on weather conditions. The segments presenting an abnormal traffic condition, i.e., average traveling speed significantly lower than normal speed, are identified and further split into smaller segments with the aim of isolating the potential incident area. If necessary, vehicles in the segment are analyzed individually in order to identify abnormal behavior. Schäfer, Thiessenhusen, and Wagner (2002) detect blocked or congested road segments by using information from GPS-equipped vehicles, and by considering a road congested if the average speed of the vehicles is below 10 km/h. The IDS proposed by Basnayake (2004) collects real-time GPS data from probe vehicles moving on a road network subdivided into segments (with each segment being the part of road between two road intersections) in order to detect incidents by analyzing travel time of segments and acceleration noise of vehicles through statistical techniques. Yoon et al. (2007), detect traffic conditions from GPS data, by employing speed and temporal features referred to the segments identified on the road network; they can characterize unique traffic patterns on each road and identify traffic states on a segment-by-segment basis. Coifman (2003), a traffic monitoring system for detecting incidents on highways by means of dual-loop road sensors and GPS-equipped vehicles is presented. In Srinivasan et al. (2004), a neural network-based model is trained on GPS data collected from inductive loop sensors in a 5.8 km city-bound area. Ghosh and Smith (2014) adapt freeway incident detection algorithms to the urban scenario. They simulate incidents during rush hours, and use traffic volume and traffic occupancy, as inputs to a support vector machine model.

GPS traces can be employed also to model or predict traffic conditions. Castro, Zhang, and Li (2012), use the historical data of one month collected from GPS sensors onboard 5000 taxis to construct a model of traffic density in order to model and predict traffic conditions, and estimate the effects of emissions on air quality. The city is decomposed into disjoint areas (segments) and the traffic density is computed for each segment. Sirvio and Hollmén (2008), employ Markov chains and artificial neural networks for the prediction of road traffic conditions using data related to weather conditions and road characteristics. Yuan, Zheng, Xie, and Sun (2011), construct a Markov model to estimate future traffic conditions and travel times of the user's path, using both historical and real-time GPS data collected from taxi vehicles, along with weather and driver behavior information. Liu, Yue, and Krishnan (2013, 2015), the authors propose a Gaussian Process Dynamic Congestion Model to capture the dynamics and the uncertainty of traffic conditions, with the aim of simplifying the adaptive routing of a fleet of cooperative vehicles. They exploit GPS traces collected along one year from taxis.

With respect to the current research aimed at detecting traffic congestion and incidents, the characteristics of our system can be summarized as follows. We propose a system able to detect traffic congestion and incidents in real-time. The system basically: (1) places the GPS traces in the road map, (2) assigns to each road segment of the map a traffic state based on the speed of the vehicles travelling the segment and on the values of a few thresholds, and (3) sends traffic alerts based on a spatiotemporal analysis of the classified segments. We can identify the following strengths of our system:

i) Unlike the majority of works in the same field, it does not exploit statistical or clustering techniques. Thus it needs neither historical data (e.g., GPS traces, weather information, typical traffic condition), nor a learning time-consuming process. In fact, it exploits real-time GPS traces data consisting of position, velocity, and time, without the need for any other vehicular data such as acceleration noise and travel times;
ii) Unlike other ITSs, it is not based on the use of sensor probe vehicles (e.g., taxis, buses) or fixed sensors (e.g., inductive loop sensors, video recognition cameras) deployed in the road network. In fact, the GPS traces can be directly collected from smartphones or GPS trackers, resulting in a very low-cost framework in terms of installation and maintenance costs, and elaboration of data;
iii) Regarding the representation of the city road network, it employs a digital map in which the length of each road segment is not fixed (as in similar papers in the literature), but adapted to the particular conformation of the road, on the basis of the segment's speed limit;
iv) It is developed as an event-driven infrastructure, built on an SOA architecture, and it allows directly providing citizens with accurate and reliable information on traffic conditions. Using this architecture, it is able to directly notify traffic events to the drivers registered to the system, by means of a traffic alert service, without the need for them to access official news websites or radio traffic news channels. In addition, the SOA architecture allows building scalable services (e.g., by using a dedicated data elaboration server for each geographic area), and integrating new services (e.g., with other ITS services). In addition, the proposed system could be a valuable tool for traffic and city administrations to regulate traffic and vehicular mobility. Moreover, it could be used together with traffic sensors (e.g., loop detectors, cameras, infrared cameras) and ITS monitoring systems for the detection of traffic difficulties. In this way, our system provides a low-cost wide coverage of the road network, especially in those areas (e.g., urban and suburban) where traditional traffic sensors are missing, or less travelled by probe vehicles.

Conversely, possible weaknesses are:

i) The inaccuracy (or the loss) of the GPS signal may lead to wrong positioning of vehicles in the case of very close roads (e.g., an error of about 10 m may dislocate the vehicle on the wrong road segment or even on the wrong road). However, if the inaccuracy regards only one trace, the system is not seriously impacted, as we take into account the total volume of moving vehicles, not the single vehicle. On the other hand, this is a problem affecting all the ITSs that employ GPS traces for detecting incidents;
ii) A considerable delay in detecting an incident in case of a low number of vehicles, which transmit their GPS coordinates, in the segment. Indeed, with the aim of reducing the influence of personal drivers' behaviors (for instance, a driver stopping the vehicle for looking at the view) on the decision about the traffic state of a segment, we associate a level of traffic congestion with a segment only if the number of vehicles in the segment is higher than a pre-fixed

threshold. Obviously, if the number of vehicles moving in the segment is very low, a long time will be needed to detect the incident: the incident, however, will be detected by the system in any case as soon as a sufficient number of vehicles accumulate in the road segment. Systems, which employ other types of data sources in addition to GPS traces, could reduce this problem, but at a higher cost. We have to consider, however, that, as the number of involved vehicles is low, the impact will not be dramatic for the urban mobility;

iii) The choice of the thresholds used to determine the segment states can affect the performance of the system. In our simulations, however, we have experimented with different traffic conditions and the chosen thresholds have always proved to be effective. On the other hand, in general, the majority of the systems proposed in the literature for incident detection need to adequately tune some parameter, e.g., during the learning process, or the model calibration.

3. Road network representation

One of the most interesting problems when using GPS traces is the positioning of the vehicles on the map. The positioning can be carried out in the *continuous space* if the vehicle has a precise absolute position corresponding to its GPS coordinates, or in the *discrete space* if the vehicle belongs to a *spot* (e.g., areas, cell-grid areas, segments) identified on the map. The *continuous space* positioning has two drawbacks: (i) it is not suited for the aggregate analysis (of areas or road of a city), typical of traffic monitoring systems, and (ii) it may cause faults due to GPS accuracy errors.

In the *discrete space* positioning, the city is split into disjoint areas and a membership function of GPS traces is defined for each area. The most used approach for the decomposition of a city according to the *discrete space* positioning (Castro, Zhang, Chen, Li, & Pan, 2013) exploits the use of *digital maps*. A *digital map* is a graph (*V, E*) composed of a set of vertices *V*, defined as GPS positions, and a set of edges *E*, each one defined in terms of length, bearing and endpoint vertices. With this structure, a road is described as the conjunction of consecutive edges (also called *segments*) identified in correspondence with intersections, or changes in bearing of the road. The main disadvantages of this approach are the difficulty in finding a digital map and the lack of some information in the map, such as the number of lanes of the roads, the orientation of the edges and the width of the roads. For these reasons, several approaches have proposed to directly exploit GPS traces for the manual or automatic construction of digital maps (Biagioni & Eriksson, 2012;Fathi & Krumm, 2010), or for the enrichment of existing digital maps with useful information (Alvares et al., 2007;Chen & Krumm, 2010;Schroedl, Wagstaff, Rogers, Langley, & Wilson, 2004).

In this paper we refer to the discrete space positioning, as it represents the best compromise between accuracy and simplicity. Moreover, by making this choice we are able to exploit existing digital maps, such as the ones provided by Open Street Map (OSM) (Open Street Map, 2016), the well-known open-source framework for digital maps. Furthermore, the partition of roads into segments allows distinguishing different segments of road in the same area. In fact, each road segment has its typical traffic flow and thus should be considered individually. One approach for the identification of segments on the roads, frequently used in the literature and in existing digital maps, is to consider as endpoint vertices not only intersections and changes in bearing, but also traffic lights, pedestrian crossings, and other relevant points. In this way, urban areas are described by many segments of small size, while suburban areas are described by a small number of big segments (Yoon et al., 2007). In the analysis of the extracted data collected from GPS sensors, it is important to consider both the spatial and the temporal components of information. For example, detecting a low average speed of vehicles on a segment of road does not necessarily indicate traffic congestion on the road, e.g., in the case of an urban road, the low speed may be due to the presence of traffic lights and pedestrian crossings (Yoon et al., 2007). On the contrary, in the case of a highway road, the low average speed may indicate traffic congestion.

As said above, we exploited the digital map provided by OSM which combines data recorded by GPS devices, satellite images, and other manually entered information to create a rich-information digital map (Castro et al., 2013). OSM (Open Street Map, 2016) is an open source, free-license project aimed at collecting geographic data to create freely available maps of the world with free content. The map of an area (region, province, city, etc.) is implemented as an oriented graph characterized by two main elements: *nodes* and *ways*. Nodes represent important positions on the map identified by GPS coordinates (latitude and longitude), corresponding to Points Of Interest (POIs), intersections, points of change of direction on the same road (curves), etc. Thus, between two nodes a linear *segment* is defined. Each node is associated with an *id* and with a list of tags that describes the characteristics of the node. The number of nodes of a road depends on the type of road, and on the number of relevant points on the road (typically urban roads are described by a higher number of nodes with respect to highways). Ways are ordered sets of nodes constituting open or closed polygons, representing possible paths from the start node to the end node. The nodes of a way are listed consecutively and this allows identifying stretches of consecutive roads.

4. The proposed system

The proposed system for detecting traffic congestion and incidents is based on an expert system and is characterized by a service-oriented and event-driven architecture, composed of three main modules (Fig. 1).

The "*Pre-Processing*" module is devoted to match each GPS trace point (expressed in terms of latitude and longitude) with the corresponding OSM map segment. This operation is not straightforward, as it presents two problems. First, it may require a routing algorithm to perform the completion of incomplete routes in case of jumps/gaps or missing GPS points (Hofmann-Wellenhof, Lichtenegger, & Collins, 2012). Indeed, connectivity problems (e.g., a tunnel crossing) and accuracy errors caused by an anomalous driving behavior or resulting from the sampling may cause a physical jump between two known positions, i.e. segments of the map. The use of routing algorithms is needed to rebuild the vehicle route and remove the jumps. Second, an algorithm is needed to define the travel directions of the vehicles in each segment. In fact, a point of a GPS trace indicates only the position of the vehicle on a certain road, but not its travel direction. However, in the case of a unidirectional road, the travel direction is provided by the OSM map itself, but in case of a bidirectional road at least two consecutive GPS points of the same trace are required to identify the travel direction of the vehicle.

The second module "*Segment Traffic Classification*" aims to analyze the GPS traces belonging to the segments built on city streets, and to assign a *traffic state* to each segment on the basis of considerations made on the traveling velocity of the vehicles, the number of vehicles in the segment, and the traffic code speed limit on that segment. Obviously the mere *traffic state* classification of the segments is not sufficient to understand the real traffic condition on the whole road, which depends on spatial and temporal information. Thus, in the third module "*Traffic Alert Notification*", the traffic states previously found on the segments are analyzed in order

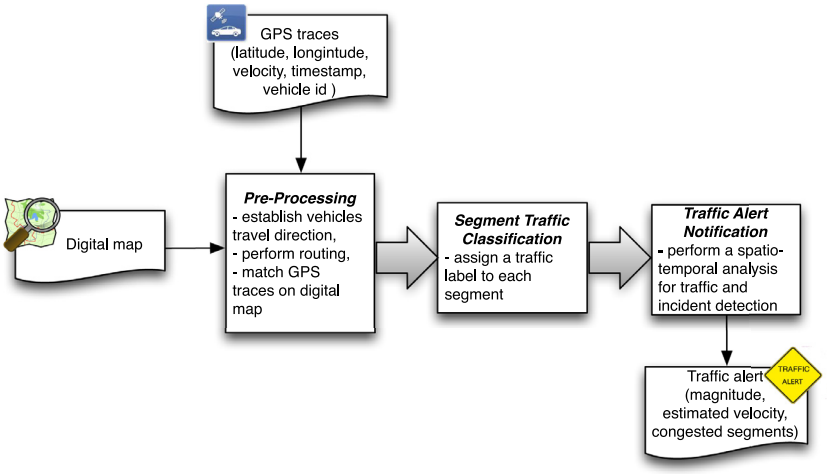

Fig. 1. System architecture for traffic congestion detection and incident detection from GPS trace analysis.

to decide whether to send *traffic alert* messages related to a given area based on a further spatial and temporal analysis.

More precisely, after the pre-processing phase, each segment of the map is observed for a time interval T sampled every f seconds. At each sampling time in T, the traces of the vehicles, which have travelled the segment, are collected, processed, and saved. The procedure is then repeated on each segment of the map in the following time interval, and so on. We denote with T_t the tth time interval analyzed. As in other papers (Hofmann-Wellenhof et al., 2012;Yoon et al., 2007), we set $f = 30$ s. This value represents a good trade-off between precision and complexity. Further, when the GPS traces are collected by using a smartphone, we cannot adopt too short sampling times because these would consume too fast the battery. The value of T is set to 2 min. We verified that this value allows us to have a reliable estimate of the average speed in the segment when the number of vehicles is low. On the other hand, when the traffic is high, this value permits to manage the corresponding complexity.

In the following, we explain in greater detail the steps performed in each module. First, we introduce some key concepts. A segment of the map is denoted with s_j, where $j = 1,...,S$, is the index of the segment, and S is the total number of segments in the map. A segment s_j is characterized by the maximum speed v_j^{lim} allowed on the portion of road corresponding to the segment and the distance d_j between the extremes of the segment. A vehicle is uniquely identified with a vehicle identifier. In a generic segment s_j, a vehicle is denoted as $m_{j,i,t}$, where $i = 1,...,M_{j,t}$, is the index of the vehicle within the segment s_j, and $M_{j,t}$ is the number of vehicles travelling s_j in T_t.

A GPS trace belonging to a vehicle is described by: i) the unique identifier of the vehicle, and ii) a set of GPS positions (latitude, longitude) of the vehicle along with the corresponding timestamps and the corresponding speeds $v_{j,i,t,h}^{ist}$, where $h = 1,..,Q_{j,i,t}$ is the index of the trace left from vehicle $m_{j,i,t}$, and $Q_{j,i,t}$ is the number of traces left by vehicle $m_{j,i,t}$ during the observed time interval T_t. Each vehicle in the segment can leave one or more traces depending on the speed of the vehicle and the level of traffic congestion of the segment (e.g., in the case of blocked traffic, a vehicle travelling the segment will be detected multiple times).

The analysis of the GPS traces is performed by using an OSM map appropriately modified. As mentioned above, segments in OSM are identified by changes of curvature of the road, pedestrian crossings, intersections, traffic lights, etc. However, in some cases the segments identified by pairs of nodes in OSM may be too long for the purposes of the proposed system; thus, the digital map should be modified by splitting longer segments. This additional segmentation allows locating more precisely a traffic event that happens on the road. Segments are split into sub-segments if the following condition holds:

$$\frac{v_j^{lim}}{\propto \cdot c} \cdot f < d_j, \tag{1}$$

where $\alpha = 3.6$ is the conversion factor from km/h to m/s, f is the sampling period, c is a parameter and d_j is the distance between the extremes of the segment. The choice of the value of c affects, on the one side, the precision and, on the other side, the complexity of the system. Indeed, the rules used in the expert system for determining the traffic state are based on the knowledge of the direction of each vehicle. The direction of a vehicle is determined by comparing two subsequent vehicle's GPS positions. The direction is easily determined if the two subsequent positions lie on adjacent segments of the map. On the contrary, determining the direction of the vehicle may be quite complex when the two subsequent positions lie on not adjacent segments. In this case, a routing algorithm has to be executed for computing the direction. Low values of c increase the precision by generating small segments. However, small segments may cause several jumps in the route, especially when a low sampling frequency is used, and thus may require a more frequent execution of the routing algorithm. Similarly, high values of c reduce the need for the routing algorithm, but result in an insufficient accuracy, due to longer segments.

By considering the Italian speed limits on highways, non-urban and urban roads, which are 130 km/h, 90 km/h and 50 km/h, respectively, we decided to set $c = 6$. With this value of c, we have that the maximum segment length is 180, 125 and 69.44 m for, respectively, highways, non-urban and urban roads. These values represent actually a good compromise between precision and computational cost. On the other hand, other works in the literature employ fixed road segments with lengths from 200 to 500 m (e.g., Kamran and Haas, 2007) by focusing on highways. In our work, however, the length of each segment is not fixed, but it adapts to the particular conformation of the road, since it depends on the speed limit on that segment.

If the condition in Eq. (1) is met, the segment is split into two sub-segments identified by a new node centered between the two original nodes. The condition in Eq. (1) is then checked on the two new sub-segments, and so on. At the end of this step, we obtained an optimized digital map with segments having an average size of about 50 m, as we focused primarily on urban roads.

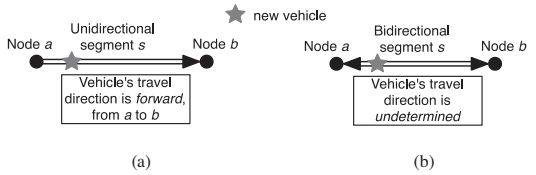

Fig. 2. Determining the travel direction on a segment for a new vehicle in the map: (a) unidirectional segment, (b) bidirectional segment.

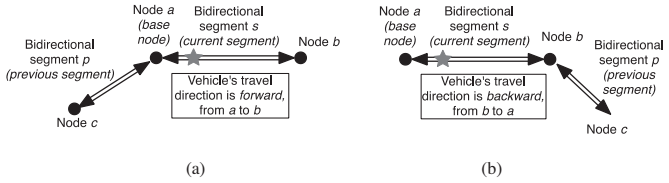

Fig. 3. Determining the travel direction on a bidirectional segment for a vehicle already moving in the map, when current and previous segments are adjacent: (a) *forward* travel direction, (b) *backward* travel direction.

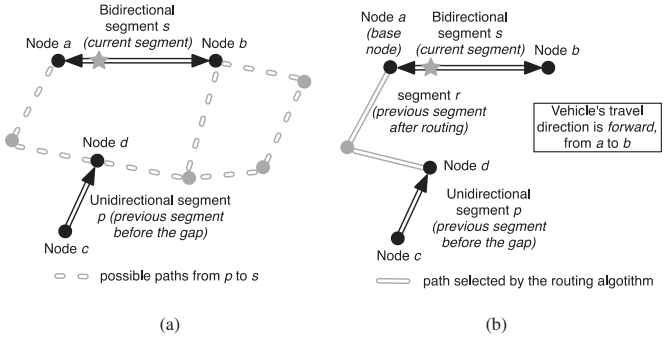

Fig. 4. Determining the travel direction on a bidirectional segment for a vehicle already moving in the map, when current and previous segments are not adjacent, by means of a routing algorithm. (a) before routing algorithm, (b) after routing algorithm.

4.1. Pre-Processing

The "*Pre-Processing*" module processes each GPS trace with the aim of obtaining: (i) the segments corresponding to the GPS positions that form the trace, (ii) the travel directions of the vehicles in each segment (i.e., *forward* and *backward*), and (iii) the segments needed to reconstruct the vehicle route in case of jumps or missing GPS points.

The mapping of GPS traces on the segments of the map consists of defining a membership function of GPS traces to segments or nodes of the map. A GPS trace is mapped, first, to the closest node of the map and then to the closest segment (whose exact position is identified in terms of the GPS coordinates of its midpoint) among those segments, which arise from the node. The mapping is performed by exploiting the GraphHopper API for Java (GraphHopper Route Planner, 2016). For each GPS point of a trace, after detecting the corresponding segment in the map, the travel direction of the vehicle in the segment is identified according to the following procedure:

1. If the GPS point is the first one coming from a given vehicle, and:
   1.1. if the point belongs to a unidirectional segment, the travel direction is directly provided by the road network map (see Fig. 2(a));
   1.2. if the point belongs to a bidirectional segment, we are not able to determine the travel direction. Thus, the vehicle is temporarily placed in both the lists corresponding to the travel directions *forward* and *backward* for that segment. The lists will be correctly updated afterwards by removing the vehicle from the incorrect travel direction list (see Fig. 2(b));
2. If the GPS point comes from a vehicle already moving in the map, we can determine the travel direction in the corresponding segment (current segment), since we know the previous position of the vehicle, corresponding to the previous segment in the trace. More precisely,
   2.1. If the current segment is unidirectional, the travel direction is directly provided by the road network map (see Fig. 2(a));
   2.2. If the current segment is bidirectional and the previous segment is unidirectional, the travel direction can be inferred from the road network map;
   2.3. If the current segment is bidirectional and the previous segment is bidirectional, it is necessary to check its adjacency with the previous segment of the trace:
      2.3.1. If current and previous segments are adjacent, the travel direction can be determined by exploiting a feature of the OSM graph: If the adjacency regards the *base node* (i.e., the first extreme node) of the current segment, the travel direction is *forward*, otherwise it is *backward* (see Fig. 3);
      2.3.2. If current and previous segments are not adjacent, a gap exists and an appropriate routing algorithm is needed to rebuild the missing path between the previous and the current segments. The routing algorithm implemented is a modified version of the one provided by the GraphHopper API (GraphHopper Route Planner, 2016), and it generates the best approximate path between two non-adjacent segments, by giving priority to main streets in the map, and by choosing the shortest path according to the classical Dijkstra's algorithm (Dijkstra, 1959) for route planning (Delling, Sanders, Schultes, & Wagner, 2009). Once that a possible path has been built, the operations described in item 2.3.1 are repeated first on the previous segment, and then backwards on the other segments of the approximate path (see Fig. 4).

Any time the travel direction of the current segment in a trace is correctly determined, the travel direction of the previous segment of the trace, if remained undetermined, can be updated by removing the vehicle from the incorrect travel direction list.

4.2. Segment traffic classification

The aim of this module consists in analyzing the GPS traces collected in the current time interval T_t and in assigning to each segment s_j of the map a *traffic state* TS_t in the set {*absent, flowing, slowed, very slowed*, and *blocked*}, and a *representative actual speed* $v_{j,t}^{actual}$ on the basis of the number and the speeds of the vehicles passing in the segment in T_t. The choice of the possible traffic states has been performed according to the classification employed in well-known traffic information systems, such as Google Traffic (Google Maps, 2016), and "Autostrade per l'Italia" (Autostrade per l'Italia, 2016), the main traffic news network in Italy. The two systems employ four traffic states (whose meaning matches with our states *flowing, slowed, very slowed*, and *blocked*), to which we added the additional state *absent*. The state *absent* means that there is no traffic event, and also indicates a total absence of vehicles. This state is used by our system during incident detection.

More in detail, Google Traffic (Google Maps, 2016) takes into account four traffic states: (i) normal speed of traffic (green color), (ii) slower traffic conditions (yellow color), (iii) congestion (red color), and (iv) blocked or stop-and-go traffic (dark red color). This classification matches with our classification, as follows: Google green color corresponds to our *flowing* state, Google yellow color

corresponds to our *slowed* state, Google red color corresponds to our *very slowed* state, and Google dark red color corresponds to our *blocked* state. Similarly, the web site "Autostrade per l'Italia" (Autostrade per l' Italia, 2016) provides an almost similar classification of four traffic states: (i) no relevant traffic event (green color), (ii) slowdowns (yellow color), (iii) criticalities (red color), and (iv) blocked traffic (black color). Also in this case, the traffic states (i), (ii), (iii), and (iv) correspond to our *flowing, slowed, very slowed*, and *blocked* traffic states, respectively.

The rules of the expert system are adapted to the number of vehicles that have crossed the considered segment in T_t in order to avoid false alarms and manage potential outliers. The following two sections explain the rules of the *Segment Traffic Classification* module, first in the case of an insufficient number of vehicles passing in the segment in T_t, and then in the case of a sufficient number of vehicles. We have to consider separately the case of an insufficient number of vehicles because we cannot be sure that the information extracted by analyzing these vehicles is dependent on the behaviors of the drivers or on the traffic conditions. More precisely, if the number $M_{j,t}$ of vehicles crossing the segment in T_t is below a certain threshold, i.e., $M_{j,t} < M_{min}$, special considerations have to be made in order to avoid an incorrect analysis.

We wish to point out that the system does not need a GPS trace from each vehicle moving in the road network in a given time interval. Obviously, the higher the number of traces, the more reliable the classification. However, a minimum number of vehicles is necessary for guaranteeing to distinguish a real traffic event from an anomalous driving behavior of an isolated driver. We are confident that this minimum number is easily reachable today thanks to the growing number of GPS-equipped vehicles (e.g., mounting insurance companies' black-boxes, GPS navigation devices, and overall GPS-equipped smartphones). Obviously, an incident or traffic event cannot be detected if there is not a sufficient number of vehicles involved, since it would not produce traffic congestion.

1) Insufficient Number of Vehicles in Segment s_j

If $M_{j,t} < M_{min}$, the system can assign only two traffic states to the segment s_j, namely *absent* and *flowing*, as follows:

1. if there are no traces, i.e., $M_{j,t} = 0$, the traffic state TS_t is set to *absent*;
2. if $0 < M_{j,t} < M_{min}$, the traffic state TS_t is set to *flowing*.

We consider just two traffic states because the number of vehicles is too low to reliably detect a traffic condition. Actually, the traffic might be *blocked, slowed* or *very slowed*, but we cannot reliably infer these states because of the low number of vehicles. On the other hand, a possible event causing traffic congestion will be detected whenever a sufficient amount of vehicles will accumulate in the segment.

2) Sufficient Number of Vehicles in Segment s_j

If $M_{j,t} \geq M_{min}$, the system is able to assign the traffic states *flowing, slowed, very slowed*, and *blocked*. First, the average speed of each vehicle $m_{j,i,t}$ in the considered segment s_j is calculated as:

$$\bar{v}_{j,i,t} = \sum_{h=1}^{Q_{j,i,t}} \frac{v_{j,i,t,h}^{ist}}{Q_{j,i,t}}, \tag{2}$$

Next, the speeds $\bar{v}_{j,i,t}$ of the vehicles are ordered, and the median value $\tilde{v}_{j,i,t}$ of these speeds is computed. By using the median value, we reduce the effect of outliers.

Then, the module assigns the representative speed $v_{j,t}^{actual} = \tilde{v}_{j,i,t}$ to the segment, and, by evaluating $\tilde{v}_{j,i,t}$, sets the traffic state TS_t as follows:

1. If $\tilde{v}_{j,i,t} \geq P_1 \cdot v_j^{lim}$, with P_1 a percentage of the maximum speed v_j^{lim} allowed on the segment s_j, then the traffic state is set to *flowing*;
2. If $P_2 \cdot v_j^{lim} \leq \tilde{v}_{j,i,t} \leq P_1 \cdot v_j^{lim}$, with $P_2 < P_1$ a percentage of the maximum speed v_j^{lim} allowed on the segment s_j, a further analysis is required because the traffic condition is not clearly defined. Thus, the behavior of the majority of vehicles is taken into account. The traffic state of the segment is set to:
   2.1. *flowing*, if the majority of vehicles have an average velocity $\bar{v}_{j,i,t} \geq P_1 \cdot v_j^{lim}$;
   2.2. *slowed*, otherwise;
3. If $v_{block} \leq \tilde{v}_{j,i,t} \leq P_2 \cdot v_j^{lim}$, the traveling velocity in the segment is quite low, thus the traffic state is set to *very slowed*;
4. If $\tilde{v}_{j,i,t} \leq v_{block}$, the traffic is still, thus the traffic state is set to *blocked*.

In the experiments, we set $M_{min} = 4$, $P_1 = 50\%$ and $P_2 = 40\%$. The value $M_{min} = 4$ has been chosen for reducing the influence of personal drivers' behaviors on the decision about the traffic state of the segment. Indeed, a driver could decide to stop the vehicle for drinking a coffee. If only this vehicle is travelling the segment in the time interval T, the traffic state could be erroneously set to blocked. In our experiments, we verified that $M_{min} = 4$ is a choice that guarantees, on the one side, to limit the effect of personal drivers' behaviors and, on the other side, to determine a state even if the traffic is not intense.

The values of P_1 and P_2 arise from considerations related to the classification of traffic events in traffic news channels (e.g. Google Maps, 2016). In fact, it is well known that there is no generally agreed definition of good or bad traffic states (Yoon et al., 2007). We set the values of parameters P_1 and P_2, on the one hand, by trying to follow the typical classification of traffic states provided by traffic news channels and, on the other hand, by trying to reduce the time needed to correctly detect the traffic states. In addition, the values represent only a choice aimed at producing a possible traffic state classification as done in other papers (de Fabritiis, Ragona, & Valenti, 2008). Further, we set $v_{block} = 3$ km/h by adapting the suggestions in Schäfer et al. (2002) to the small urban network of Pisa.

At the end of the elaboration made in this module, for the considered time interval, the system has associated with each segment of the map a representative speed and a traffic state (namely, *absent, flowing, slowed, very slowed, blocked*). The segments having traffic state *slowed, very slowed*, and *blocked* will be taken into account by the *Alert Notification* module with the aim of checking the presence of a traffic congestion or an incident.

Fig. 5 summarizes how the *Segment Traffic Classification* module works by using a flow chart.

4.3. Traffic alert notification

The *Traffic Alert Notification* module elaborates only segments having traffic state *slowed, very slowed*, and *blocked*, and appropriately generates a series of traffic alerts on the basis of: (i) a temporal analysis of the information associated with the segments in a certain number N of time intervals previous the current one, and (ii) a spatial analysis of the traffic condition in segments adjacent to the considered one.

The choice of N affects, on the one side, the reliability of the alert and, on the other side, the responsiveness of the system. Indeed, if, for instance, $N = 0$, we decide to launch an alert by considering only the current traffic state of the segment, but this state could be affected by drivers' behavior rather than by actual traffic difficulties. It might occur that a driver blocks the road segment for a while waiting for a parking slot: in this case, considering just the current time interval would cause a wrong alert. On the other hand, the larger the N value, the higher the delay for launching the alert. In the experiments, we realized that $N = 2$ allows achieving a good trade-off between reliability and responsiveness.

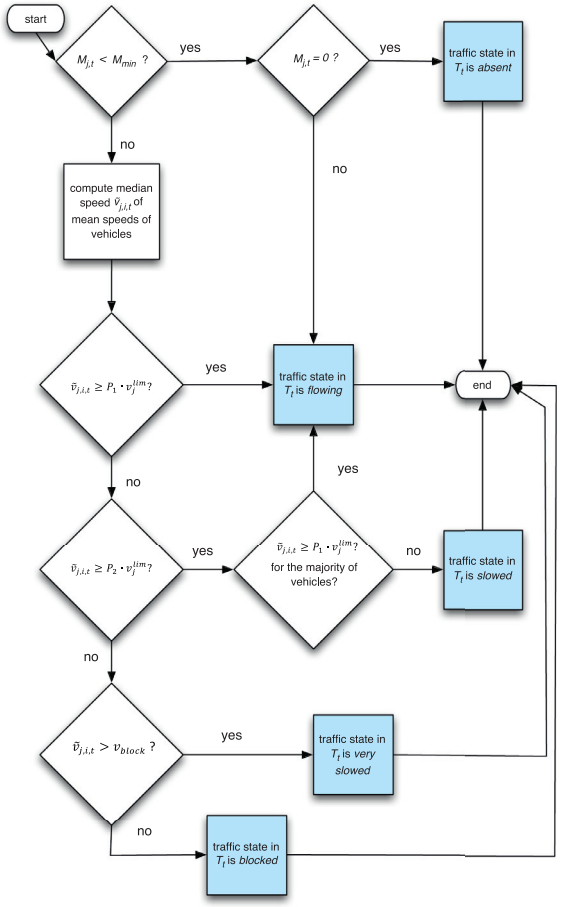

Fig. 5. The flowchart of the *Segment Traffic Classification* module.

A *traffic alert* contains the following information: i) the spatial extension of the event, i.e., the segment or the list of segments associated with the traffic event, ii) the traffic event (traffic congestion magnitude or incident) which depends on the classification of the segments previously made, iii) the average travelling speed associated with the segments involved in the traffic event. An *alert report* contains the list of traffic alerts related to the current time interval. More precisely, since the alert notification module uses the information related to $N+1$ time intervals (the N previous intervals and the current one), the first alert report will be sent at least after $N+1$ time intervals, while the following ones will be sent at the end of each time interval, i.e., every 2 min.

The traffic events of the "traffic alert" are:

- *alert for slowed traffic,*
- *alert for slowed or very slowed traffic,*
- *alert for very slowed traffic,*
- *alert for blocked traffic,*
- and *alert for incident*, if the specific circumstances of an incident are recognized.

The spatial extension of the event is initially expressed as a set of road segment ids, and is then converted to the city street name by using the OSM tool *Nominatim* (Nominatim, 2016) for reverse geocoding.

The processing carried out in this module consists of two steps: first we analyze the segments having traffic state *blocked*, and then we analyze the segments having traffic state *slowed* or *very slowed*.

1) Segments with traffic state blocked

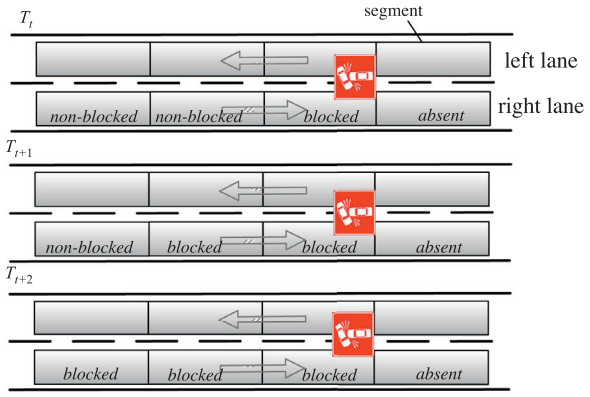

Fig. 6. The situation corresponding to a "queue with head", which may have been caused by an incident.

By exploiting spatial and temporal considerations, the system is able to determine situations of blocked traffic, or identify possible incidents, if specific conditions are met.

Typically, when an "incident" occurs, the traffic congestion caused by the incident lasts for a quite long time and the vehicles stuck in the queue are almost the same in subsequent time intervals. Two types of block may be identified: the incident can block the whole roadway, or it can block only part of the roadway, allowing the transit of vehicles in its free portion. In the former case, we will typically observe the "queue with head" (shown in Fig. 6), i.e., an almost total absence of GPS traces ahead of the incident, and a queue of vehicles behind the incident, which tends to grow. In the latter case, it is extremely hard to detect an incident (Basnayake, 2004) and distinguish it from a *blocked traffic* condition. Thus, in this paper, we will consider only incidents blocking the whole roadway.

Based on the above considerations we perform the following analysis. For each segment s_j having traffic state *blocked* in the current time interval, a spatiotemporal analysis is performed, taking into account segments adjacent to s_j, and the traffic condition in N previous time intervals.

First, with the aim of defining the spatial extension of the event, we iteratively check the state of all the segments adjacent to s_j (both in the *forward* and *backward* directions), and that of the segments adjacent to these, until we find a segment with traffic state different from *very slowed* or *blocked*.

1. If the event is restricted to a single segment s_j, i.e., the segments adjacent to s_j have all traffic state *slowed, absent*, or *flowing*, we perform a temporal analysis of the state of s_j in N previous time intervals, with the aim of checking if the event is relevant or not:
   1.1. If s_j had traffic state *blocked* or *very slowed* in all the N previous time intervals, the event is considered to be relevant. Thus, we need to distinguish between a possible *incident* and a *blocked traffic* condition. More in detail, we check whether the vehicles stuck in the queue are the same, or have changed, in consecutive time intervals, as follows:
      1.1.1. If a percentage P_3 of vehicles present in the segment in N previous time intervals are the same, we are dealing with an event that blocks the traffic in the entire roadway for a quite long period of time. Thus, an *alert for incident* in s_j is sent;
      1.1.2. Otherwise, an *alert for blocked traffic* on segment s_j is sent;
   1.2. Otherwise, the event is considered to be not relevant, and no alert is sent;

2. If the event involves a set of segments around s_j, i.e., the segments adjacent to s_j have traffic state *blocked* or *very slowed*, and,
   2.1. If the condition of the "queue with head" is met, i.e., i) a segment (or a series of segments) immediately behind s_j have traffic state *very slowed* or *blocked*, and ii) a segment ahead of s_j has traffic state *absent,* we check the condition for an incident, that is, if the vehicles stuck in the queue are the same, or have changed, in consecutive time intervals, as explained earlier (see conditions 1.1.1 and 1.1.2). Thus, the system may send an *alert for incident* or an *alert for blocked traffic,* for the set of segments around s_j;
   2.2. If the condition of the "queue with head" is not met, we set the magnitude of the event on the basis of the number and type of segments involved, as follows:
      2.2.1 If the majority of segments involved have traffic state *blocked*, we check the condition for an incident, as explained earlier (see conditions 1.1.1 and 1.1.2). Hence, the system sends an *alert for incident* or an *alert for blocked traffic* for the set of segments around s_j;
      2.2.2 If the majority of segments involved have traffic state *very slowed,* the system sends an *alert for very slowed traffic* for the set of segments around s_j.

At the end of processing, the alert notification module generates an alert report describing the alerts found in the performed analysis.

Theoretically, the value of P_3 should be 100% since, in case of traffic block, the same vehicles will remain in the same positions for a while. Actually, it could occur that one or more vehicles decide to make a U-turn or to opt for another route, taking a cross street. Thus, taking these situations into account, we did not set the value of P_3 to 100%: since the number of the vehicles, which decide to exit from the queue, is typically low and further the value of P_3 has to be chosen high in order to avoid false alarms, we set P_3 to 90%. In our simulations, we verified that this value allows achieving remarkable incident detection rates with a low number of false alarms.

2) Segments with traffic state: slowed, very slowed

For each segment s_j marked in the current time interval with the traffic states: *slowed*, or *very slowed*, a spatial and temporal analysis is performed as follows:

1. If all the segments adjacent to s_j in the current time interval have traffic state *absent* or *flowing*, the traffic event is restricted to s_j. To confirm it and decide the alert magnitude, the situation is analyzed in s_j in the N previous time intervals:
   1.1. If s_j had traffic state *slowed, very slowed,* or *blocked* in all the N time intervals, then an *alert for slowed traffic* in s_j or an *alert for very slowed traffic* in s_j is sent, based on the more recent information (the traffic state in s_j in the current time interval);
   1.2. Otherwise, no alert is sent;
2. If at least one of the segments adjacent to s_j has traffic state *slowed, very slowed*, or *blocked*, the traffic event affects multiple segments:
   2.1. For each segment adjacent to s_j having a traffic state *slowed, very slowed,* or *blocked*, the state of its adjacent segments is taken into account in order to define the spatial extension of the event, until we find a segment with different traffic state. Thus a traffic alert related to the set of segments adjacent to s_j is sent, with a magnitude based on the number and type of segments involved:
      2.1.1. If the majority of the segments have traffic state *slowed*, then an *alert for slowed traffic* is sent;
      2.1.2. If the majority of the segments have traffic state *very slowed*, then an *alert for very slowed traffic* is sent;
      2.1.3. If the segments have traffic state *slowed* and *very slowed* in equal number, then an *alert for slowed or very slowed traffic* is sent.

Fig. 7 summarizes how the *Traffic Alert Notification* module works by using a flow chart.

5. Simulation of GPS traces

Due to the difficulties (e.g., privacy or permission issues) to collect or obtain a reasonable number of real-world GPS traces, traffic simulators (Brinkhoff, 2002;Krajzewicz et al., 2002;Krajzewicz, Erdmann, Behrisch, & Bieker, 2012) are frequently used for validating systems of detection of traffic congestion and incidents. In this work the GPS traces used for the development of the traffic detection system were simulated using the well-known software SUMO (Krajzewicz et al., 2002;Krajzewicz et al., 2012).

SUMO allows creating multi-vehicular simulations of traffic flow by implementing the microscopic model of vehicular mobility developed by Stefan Krauß (Krauß, Wagner, & Gawron, 1997;Krauß, 1998), also known as *car-following* model, in which the dynamics of each vehicle is governed by the dynamics of the preceding vehicle (Krauß, 1998). The simulation is time-discrete, i.e., the time is marked by steps of 1 s, and space-continuous, i.e., each vehicle is modeled by a GPS position, and its instantaneous velocity. At each step of the simulation, the velocity of each vehicle is adapted according to that of the preceding vehicle, so as to avoid collisions with the preceding vehicle, and by respecting the speed limit of the road, and the safety distance. SUMO requires: i) an underlying *digital map* representing the road network on which the traffic simulation will take place, and ii) the *traffic demand*, i.e., the flows of vehicles that will travel on the road network. The traffic demand is modeled in terms of vehicles, routes and flows. Vehicles are described in terms of physical characteristics (acceleration, maximum speed, dimensions, etc.), and type (taxi, bus, private cars, motorbikes, etc.). Routes taken by vehicles contain the sequence of edges crossed (from departure edge to arrival edge), and are generated on the basis of the rules of the digital map (e.g., allowed path, speed limits, traffic lights, one ways). Flows are sets of vehicles joining the simulation at a given time, with a given penetration rate, and following a certain route. The penetration rate of vehicles is determined by the value of the parameter *period* of the flow in SUMO. This parameter determines the level of congestion of the route. It indicates how often (in number of seconds) a new vehicle (belonging to the flow) is added to the simulation. Low values of the parameter indicate a high flow rate (heavy traffic), while high values indicate a low flow rate (very light or flowing traffic). We have employed values of this parameter ranging from 5 to 120.

SUMO does not allow to directly simulating the traffic dynamics after an incident event. However, it allows simulating incidents by exploiting the *bus stop* feature. In fact, an incident can be represented by a vehicle that stops at a given location for a quite long time. More in detail, to simulate an incident, we define a bus stop, and a route covered by a special vehicle. The bus stop specifies the location and the duration of the incident. The special vehicle simulates the incident by stopping at the bus stop for the incident duration. All the vehicles following the special vehicle will stop behind it in queue until the end of the incident.

The dataset that we have simulated consists of the GPS traces of about 50,000 vehicles traveling in the city of Pisa, Italy, in two different scenarios, namely working day and holiday. The area monitored corresponds to about 70 km^2. To reproduce the appropriate traffic dynamics, as close as possible to the real traffic condition in the city, we have exploited the typical traffic conditions of Pisa in typical working day and holiday, extracted from Google Traffic (Google Maps, 2016), and the real-world data collected from

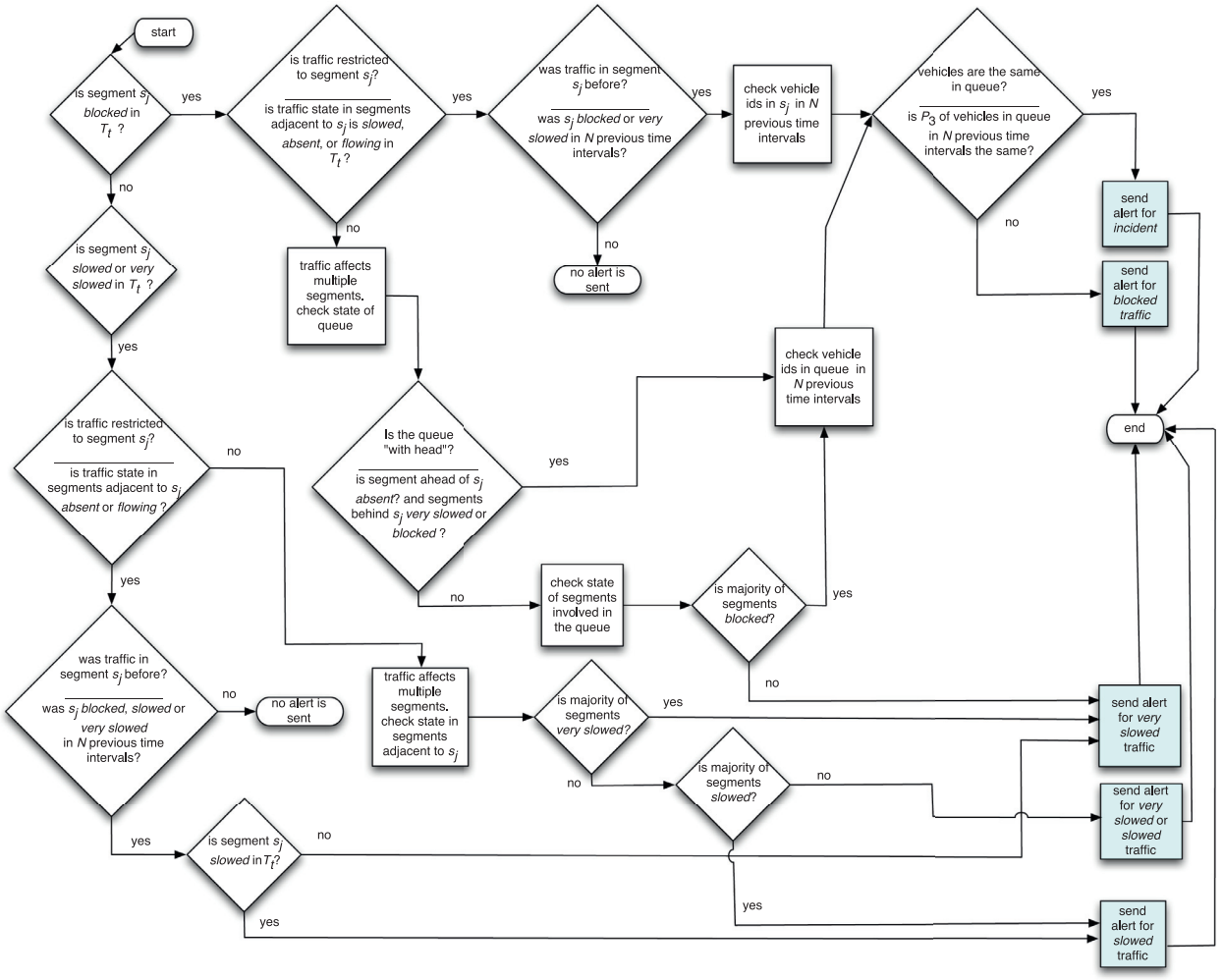

Fig. 7. The flowchart of the *Traffic Alert Notification* module.

smartphone and other devices of voluntary drivers within a specific data collection campaign. We wish to point out that the simulation of data was necessary, as real-world data collected by voluntary drivers are typically insufficient. The real GPS data were used as reference to build the simulated data. More in detail, we built the simulated GPS routes by combining the real GPS data collected from voluntary drivers with the information about traffic conditions provided by Google Traffic (Google Maps, 2016) and Here Map (Here Map, 2016) services. In particular, Here Map was used to check the correspondence between typical traffic data and real-time traffic data. Thus, we adjusted real routes in order to meet the corresponding typical traffic conditions of the time interval they belong to. We also added new routes, e.g., with same starting and/or ending points of real routes, but following a different path (typically congested in the considered time interval). The aim is to reproduce as best as possible the typical vehicle mobility and traffic at different time intervals in the city of Pisa, Italy. Thus, we took into account 20 different real routes of vehicles crossing the city at different times, as it typically happens in the two scenarios, and we obtained about 200 routes per scenario. The routes taken into account contain mainly urban roads, but they contain also a few freeway roads. Regarding incident events, we have simulated several incidents in the two scenarios at different times and places. The GPS traces were sampled every 30 s from 6 a.m. to 11 p.m. On average, about 1000 vehicles were involved in the simulations. To perform the experiments, we selected the most relevant (e.g., the most frequently travelled or the most popular) routes and time intervals for the volunteer drivers involved in collecting data in a typical working day and a typical holiday. In particular, we took into account rush hours and routes involving main roads, by leaving out side roads, where the occurrence of a traffic event is extremely rare. The places for simulating incidents and traffic congestions were chosen based on the following considerations. Regarding traffic congestions, we selected roads where often a traffic event occurs (e.g., during rush hours), according to the services Google Traffic (Google Maps, 2016) and Here Map (Here Map, 2016), which provide information about traffic conditions. Regarding incidents' locations, we employed information on incidents provided by the local administration (Comune di Pisa, 2016) and by local police reports for selecting places where incidents had occurred in the past. Thus, the simulation of traffic congestion and incidents reproduce as faithfully as possible the traffic condition in Pisa, Italy.

6. Experiments

We have performed two types of experiments for evaluating the capability of the system of detecting, respectively, incidents and traffic congestion. In the following, we present the results regarding incident and traffic events separately in order to i) better understand the capability of the system in recognizing incidents and ii) make the results themselves comparable with those from other

Table 1
Experimental results of incident detection.

Scenario	DR (%)	MR (%)	Prec. (%)	F_1-score (%)	NFA	MTD (min)
Workday	91.6	8.3	88	89.7	3	6.9
Holiday	91.6	8.3	88	89.7	3	6.63

incident detection systems. However, the experiments on incidents were performed by adding incidents to the typical workday and holiday scenarios, which already contain traffic events. Thus, the performance results are actually obtained by considering both incidents and traffic congestions.

6.1. Incident detection

We simulated 24 incidents in different times and places, for the two different scenarios. The incidents have durations of 15, 20 and 30 min. In fact, an incident blocking the whole roadway typically takes more than 15 min to be resolved. Anyhow, the average incident duration is estimated in 37, and 45 min, in Taiwan freeways, and U.S. cities, respectively (Viriyasitavat, 2015).

The computation time needed by the system to produce an Alert Report is in the range 2–15 s, depending on the considered time interval in the day. Indeed, in specific time intervals a very large number of vehicles are circulating in the city, and the state of several segments is classified as *slowed, very slowed*, and *blocked*, thus increasing the computational time of the system.

We evaluate the performance of the proposed system in terms of the following statistical measures:

i) *Detection Rate* (DR): the ratio between the number of correctly detected incidents and the number of simulated incidents (it corresponds to the recall of the system);
ii) *Miss Rate* (MR): the ratio between the number of not detected incidents and the number of simulated incidents,
iii) *Precision*: the ratio between the number of correctly detected incidents, and the sum of detected incidents and false alarms;
iv) F_1-*score*: the weighted harmonic mean of precision and recall;
v) *Number of false alarms* (NFA): the number of erroneously detected incidents;
vi) *Mean Time to Detect* (MTD): the mean time interval between the time of the incident and the time of its detection.

In both the scenarios, we were able to detect 22 incidents over 24. The two missed incidents may be due to the presence of several access points along the road interested by the incident. From the access points new vehicles enter the road and join the queue. We have verified, however, that in the case of the missed incidents, the system indicates that a congestion event is occurring on the road, by sending an alert for *blocked* traffic, instead of an alert for incident. Regarding false alarms, the system erroneously detected 3 incidents in both the scenarios. The overall simulation results are reported in Table 1, while Tables 2 and 3 report the information about each simulated incident for the holiday, and the workday scenarios, respectively. Higher values of the *Time to Detect* (TD) are due to light traffic conditions during the incident. This means that if the vehicles involved are many, the incident will be detected in a very short time. Otherwise, if the vehicles involved are few, the incident will be detected in any case, but in a longer time, depending on the number of vehicles travelling on the route of the incident location (on the basis of the value of the parameter *period* of the flow in SUMO). Thus, DR does not depend on the number of vehicles involved in the simulation: if the condition $M_{j,t} \geq M_{min}$ on the minimum number of vehicles required to detect a possible *blocked* traffic state on a segment s_i is verified, the incident will be detected. The only threshold that may affect DR is the percentage value P_3 of the same vehicles stuck in queue in N previous time intervals. However, by setting the value $P_3 = 90\%$, we are able to achieve a DR of 91.6%.

Table 2
Incident detection on holiday.

#	Incident information		Incident detection	
	Place and Time	Duration	Detection	TD
1	Lungarno Galilei, 7:30 a.m.	30 min	yes	2 min
2	Via V. Veneto, 7:30 a.m.	15 min	yes	14 min
3	Via Contessa Matilde, 7:30 a.m.	30 min	yes	8 min
4	Lungarno Galilei, 8:30 a.m.	15 min	yes	4 min
5	Via V. Veneto, 8:30 a.m.	15 min	no	–
6	Via Contessa Matilde, 8:30 a.m.	30 min	yes	16 min
7	Lungarno Gambacorti, 9:30 a.m.	15 min	yes	4 min
8	Via Fiorentina, 9:30 a.m.	30 min	yes	6 min
9	Via dell'Aeroporto, 9:30 a.m.	30 min	yes	0 min
10	Via C. Battisti, 9:30 a.m.	20 min	yes	6 min
11	FI-PI-LI Highway, 9:30 a.m.	20 min	yes	2 min
12	Via C. Matteucci, 9:30 a.m.	20 min	yes	4 min
13	Via Bonanno, 12:30 a.m.	30 min	no	–
14	Via Aurelia, 12:30 a.m.	30 min	yes	8 min
15	Via dell'Aeroporto, 12:30 a.m.	30 min	yes	2 min
16	Via Cisanello, 12:30 a.m.	30 min	yes	6 min
17	FI-PI-LI Highway, 12:30 a.m.	20 min	yes	10 min
18	Via Statale Abetone, 12:30 a.m.	30 min	yes	16 min
19	Via Aurelia, 4:30 p.m.	30 min	yes	8 min
20	Via delle Cascine, 4:30 p.m.	20 min	yes	10 min
21	Via dell'Aeroporto, 4:30 p.m.	30 min	yes	8 min
22	Via P. Nenni, 6:00 p.m.	30 min	yes	0 min
23	Via S. G. Bosco, 6:00 p.m.	20 min	yes	4 min
24	Via G. Moruzzi, 6:00 p.m.	30 min	yes	8 min

Table 3
Incident detection on workday.

#	Incident information		Incident detection	
	Place and Time	Duration	Detection	TD
1	Lungarno Galilei, 7:30 a.m.	30 min	yes	2 min
2	Via V. Veneto, 7:30 a.m.	15 min	yes	10 min
3	Via Contessa Matilde, 7:30 a.m.	30 min	yes	12 min
4	Lungarno Galilei, 8:30 a.m.	15 min	yes	2 min
5	Via V. Veneto, 8:30 a.m.	15 min	yes	10 min
6	Via Contessa Matilde, 8:30 a.m.	30 min	yes	14 min
7	Lungarno Gambacorti, 9:30 a.m.	15 min	yes	6 min
8	Via Fiorentina, 9:30 a.m.	20 min	no	–
9	Viale Bonaini, 9:30 a.m.	30 min	yes	4 min
10	Via C. Battisti, 9:30 a.m.	20 min	yes	6 min
11	FI-PI-LI Highway, 9:30 a.m.	20 min	yes	2 min
12	Via C. Matteucci, 9:30 a.m.	20 min	yes	12 min
13	Via C. Battisti, 12:30 a.m.	30 min	yes	6 min
14	FI-PI-LI Highway, 12:30 a.m.	30 min	yes	14 min
15	Lungarno Sonnino, 12:30 a.m.	30 min	yes	2 min
16	Via Bonanno, 12:30 a.m.	30 min	no	–
17	FI-PI-LI Highway, 12:30 a.m.	20 min	yes	10 min
18	Via dell'Aeroporto, 12:30 a.m.	30 min	yes	4 min
19	Via Aurelia, 4:30 p.m.	30 min	yes	12 min
20	Via delle Cascine, 4:30 p.m.	20 min	yes	10 min
21	Via dell'Aeroporto, 4:30 p.m.	30 min	yes	0 min
22	Via P. Nenni, 6:00 p.m.	30 min	yes	4 min
23	Via S. G. Bosco, 6:00 p.m.	20 min	yes	4 min
24	Via G. Moruzzi, 6:00 p.m.	30 min	yes	6 min

In the following, we report the results shown in some papers in the literature where incident detection systems were proposed. We highlight that these results are obtained on different datasets and in different scenarios. Thus, the aim of presenting these results is only to show how the performance of our system is comparable with the ones reported in similar works in the literature. In Basnayake (2004), the authors employ acceleration noise, travel times, and probe vehicles and use simulated GPS traces of traffic volumes in early morning of a typical working day to simulate 30-minutes incidents. In the experiment more similar to ours, they

obtain a DR of 100%, a False Alarm Rate (FAR) of 18%, and an MTD of 70 s. Kamran and Haas (2007), identify and isolate anomalous street segments, based on the speed of vehicles, and study the behavior of vehicles in the anomalous segments, by means of a spatial and temporal analysis. They achieve a DR of 78.7%, an FAR of 1.18%, and an MTD of 2 s, employing 10 segments on a route of about 10 miles, and 10–20 vehicles per segment. Srinivasan et al. (2004), a neural network-based model was trained on GPS data collected in a 5.8 km city-bound area from inductive loop sensors, obtaining a DR of 92%, an FAR of 0.69%, an MR of 8.27% and an MTD of 106 s on the test set. Ghosh and Smith (2014), the best performing model for the urban scenario, i.e., a support vector machine, achieves a DR of 87.31% and an MTD of 98.61 s.

Regarding the MTD performance, we achieved worse results than the above mentioned approaches. On the other hand, TDs strongly depend on the current traffic condition and on the average distance between the incident location and the starting point of the vehicles' routes. Regarding the number of false alarms generated, we would like to point out that only 3 false alarms were generated by the system per scenario, despite heavy simulations were performed.

Although the achieved results are similar to those of several works in the literature, our scenario is more complex than the scenarios employed in the cited papers. In fact, our system uses GPS data coming from a wider area (about 70 km^2), involves a larger number of road segments and vehicles, and uses less information (no need for, e.g., vehicle sensor probes, fixed sensors, historical data, weather data). In addition, we would like to highlight that incident detection in urban road networks is a more challenging task than incident detection on freeways (Dia & Thomas, 2011), as urban and freeway road networks have different traffic flows due, e.g., to different speed limits and access points.

6.2. Traffic congestion detection

We simulated several traffic conditions at 6 different times and places usually congested, by testing, for each traffic event, different levels of traffic magnitude, i.e., severity of the event in terms of congestion produced, in order to show how the response of the system changes depending on the magnitude of the traffic event. The magnitude of each traffic event was varied based on the *period* parameter associated with each flow. Each simulation involves 2

Table 4
Traffic state detection.

Traffic condition simulated	Flows' *period* value	Probe vehicle's travel time on first route	Probe vehicle's travel time on second route	Traffic detection on the common portion of the routes	Traffic detection on the exclusive portion of the single routes
1. Traffic simulated in Via Gronchi at 7:30 a.m.					
heavy	2	1227 s	1367 s	Yes – *slowed, v. slowed, blocked*	Yes – *slowed, v. slowed, blocked*
heavy	5	1144 s	1090 s	Yes – *v. slowed, blocked*	Yes – *very slowed, blocked*
light	10	876 s	806 s	Yes – *very slowed*	No
very light	20	853 s	809 s	Yes – *slowed, v. slowed*	No
very light	30	852 s	797 s	Yes – *v. slowed*	No
very light	60	857 s	790 s	Yes – *v. slowed*	No
2. Traffic simulated in Via Cisanello at 8:00 a.m.					
heavy	2	1782 s	183 s	Yes – *blocked*	Yes – *v. slowed, blocked*
heavy	5	1695 s	183 s	Yes – *blocked*	Yes – *v. slowed, blocked*
light	10	432 s	235 s	Yes – *v. slowed, blocked*	Yes – *v. slowed*
very light	20	361 s	178 s	yes – *v. slowed*	No
very light	30	368 s	179 s	No	No
very light	60	371 s	185 s	No	No
3. Traffic simulated in Lungarno Pacinotti at 7:30 a.m.					
heavy	2	905 s	395 s	Yes – *v. slowed, blocked*	Yes – *v. slowed, blocked*
heavy	5	908 s	349 s	Yes – *v. slowed, blocked*	Yes – *slowed, v. slowed, blocked*
light	10	517 s	218 s	Yes – *v. slowed, blocked*	No
very light	20	308 s	192 s	No	No
very light	30	304 s	189 s	No	No
very light	60	294 s	150 s	No	No
4. Traffic simulated in Via del Brennero at 8:00 a.m.					
heavy	2	1130 s	1852 s	Yes – *v. slowed, blocked*	Yes – *v. slowed, blocked*
heavy	5	1160 s	601 s	Yes – *v. slowed, blocked*	No
light	10	978 s	354 s	No	No
very light	20	876 s	369 s	No	No
very light	30	875 s	360 s	No	No
very light	60	874 s	363 s	No	No
5. Traffic simulated in Via C. Matilde at 4:30 p.m.					
heavy	2	1482s	1884 s	Yes – *blocked*	Yes – *v. slowed, blocked*
heavy	5	1579 s	1595 s	Yes – *blocked*	Yes – *v. slowed, blocked*
light	10	1584 s	1595 s	Yes – *blocked*	Yes – *v. slowed, blocked*
very light	20	973 s	1398 s	Yes – *blocked*	Yes – *blocked*
very light	30	809 s	948 s	No	No
very light	60	802 s	945 s	No	No
6. Traffic simulated in Via Aurelia at 6:00 p.m.					
heavy	2	868 s	1353 s	Yes – *v. slowed*	Yes – *slowed, v. slowed, blocked*
heavy	5	950 s	1195 s	Yes – *slowed*	Yes – *slowed, v. slowed, blocked*
light	10	824 s	1078 s	No	Yes – *v. slowed, blocked*
very light	20	810 s	960 s	No	No
very light	30	790 s	851 s	No	No
very light	60	785 s	858 s	No	No

flows and has a duration of 30 min. For the sake of simplicity, we chose the same value of the period for the two flows. To verify the actual traffic condition in the different simulations and to compare it with the *Alert Notification* module response, we took into account the travel times of a probe vehicle for each flow. The probe vehicle joins the simulation a few minutes after its beginning, with the aim of revealing the effective simulated traffic condition.

Table 4 shows the simulation of 6 traffic events with different traffic magnitudes. As the traffic magnitude increases (shown by the increasing travel time of probe vehicles), the *Alert Notification* module is able to correctly detect all the simulated traffic events both on the common portion of the routes, and also on the exclusive portion of the route.

In this case we do not compute exact performance measure, as objective criteria to define the actual traffic condition are missing (Yoon et al., 2007). Thus, the system only detects the presence of traffic congestion, by sending more serious alerts with increasing road use.

Concluding, our experimental results show that our approach can be very effective in detecting traffic congestions and incidents by using only GPS traces belonging to moving vehicles. Thus, the developed system can be integrated into classical ITSs, allowing them to manage also those roads where monitoring supports are not available. Although today the percentage of moving vehicles, which transmit their GPS coordinates, is quite low with respect to the total, it is rapidly increasing. We expect in the near future that this percentage becomes so relevant to allow a reliable and effective use of our approach, possibly improving the performance by tuning the parameters in different operating contexts.

7. Conclusions and future work

In this work we have presented a system for detecting traffic congestion and incidents from real-time GPS traces. The system, implemented on a Service Oriented Architecture, assigns to each road segment of the city map a traffic state based on the speeds of vehicles, and sends to the users *traffic alerts*, indicating the affected area, a traffic state, e.g., *incident, slowed traffic, blocked traffic*, and the estimated velocity of vehicles in the area. The system only exploits a real-time spatiotemporal analysis of the GPS traces and needs no learning phase. We presented and discussed some experiments performed using the GPS traces generated exploiting the SUMO traffic simulator on the basis of a combination of simulated GPS data and real GPS data collected in the city of Pisa. We achieved an incident detection rate of 91.6% and an average detection time shorter than 7 min. Regarding traffic congestion detection, we showed how the proposed system is able to recognize different levels of congestion depending on road use.

As future work, we would like to extend our system along the following directions:

i) The system could be integrated with a dynamic routing service in order to suggest to the user an alternative path to follow for avoiding the incident/traffic congestion location;
ii) The confidence of the system could be increased by predicting typical traffic state. This can be achieved by collecting the GPS traces and generating models of the traffic state, possibly taking into account other information sources (e.g., weather, events scheduled in the city that may affect the normal mobility). More in detail, for instance, by considering a main road of the city, we could predict the occurrence of traffic with higher magnitude, e.g., during rush hours, in case of a bad weather forecasting report, or concurrently with an event (e.g., football match, a concert, a flash-mob, a political demonstration) happening along the main road. We can therefore compare the traffic state predicted by the model with the state inferred by the approach described in this paper. The comparison will allow us to validate the outputs of our system, when these outputs coincide with the outputs of the model, and to identify more easily unexpected traffic events (such as incidents) by detecting differences between the output of the model and the output of the system;
iii) The system could be integrated with a service we have recently developed, which exploits the tweets to detect traffic congestions (D'Andrea, Ducange, Lazzerini, & Marcelloni, 2015). Once the system detects a traffic event by analyzing the GPS traces, we could focus the search for relevant tweets on the specific geographical area where the event occurred with the aim of confirming the event and possible extracting automatically information on its causes;
iv) The system could be integrated into a dashboard of services allowing to: (1) detect traffic and incidents, (2) predict traffic state, (3) suggest alternative routes both by car and using other transport modality, (4) employ social networks to detect traffic-related events, (5) show weather information.

Acknowledgements

This work is partially supported by the "SMARTY" project funded by "Programma Operativo Regionale (POR) 2007–2013″ - objective "Competitività regionale e occupazione" of the Tuscany Region", and the project "Metodologie e Tecnologie per lo Sviluppo di Servizi Informatici Innovativi per le Smart Cities" funded by "Progetti di Ricerca di Ateneo - PRA 2015″ of the University of Pisa.

References

Alvares, L. O., Bogorny, V., Kuijpers, B., de Macedo, J. A. F., Moelans, B., & Vaisman, A. (2007). A model for enriching trajectories with semantic geographical information. In *Proceedings of the 15th annual ACM international symposium on advances in geographic information systems*. 22:1–22:8.

Anastasi, G., Antonelli, M., Bechini, A., Brienza, S., D'Andrea, E., De Guglielmo, D., et al. (2013). Urban and social sensing for sustainable mobility in smart cities. In *Sustainable internet and ICT for sustainability* (pp. 1–4).

Ania. Dossier Ania: "Scatole nere, Italia leader nel mondo" (2014). Accessed June, 1, 2016. http://www.ania.it/export/sites/default/it/pubblicazioni/Dossier-e-position-paper/Dossier-Scatole-nere-Italia-leader-nel-mondo-Aggiornamento-novembre-2014-21.11.2014.pdf.

Autostrade per l'Italia. Real-time traffic. (2016). Accessed June, 1, 2016. http://www.autostrade.it/autostrade-gis/gis.do.

Bacon, J., Bejan, A. I., Beresford, A. R., Evans, D., Gibbens, R. J., & Moody, K. (2011). Using real-time road traffic data to evaluate congestion. In C. B. Jones, & J. L. Lloyd (Eds.), *Dependable and historic computing* (pp. 93–117). BerlinHeidelberg: Springer.

Basnayake, C. (2004). Automated traffic incident detection with GPS equipped probe vehicles. In *Proceedings of the 17th international technical meeting of the satellite division of the institute of navigation* (pp. 741–750). (ION GNSS 2004).

Biagioni, J., & Eriksson, J. (2012). Inferring road maps from global positioning system traces. *Transportation Research Record: Journal of the Transportation Research Board, 2291*, 61–71.

Boriboonsomsin, K., Barth, M. J., Zhu, W., & Vu, A. (2012). Eco-routing navigation system based on multisource historical and real-time traffic information. *IEEE Transactions on Intelligent Transportation Systems, 13*(4), 1694–1704.

Brinkhoff, T. (2002). A framework for generating network-based moving objects. *GeoInformatica, 6*(2), 153–180.

Calderoni, L., Maio, D., & Rovis, S. (2014). Deploying a network of smart cameras for traffic monitoring on a "city kernel". *Expert Systems with Applications, 41*(2), 502–507.

Castro, P. S., Zhang, D., & Li, S. (2012). Urban traffic modelling and prediction using large scale taxi GPS traces. In J. Kay, P. Lukowicz, H. Tokuda, P. Olivier, & A. Krüger (Eds.), *Pervasive computing* (pp. 57–72). BerlinHeidelberg: Springer.

Castro, P. S., Zhang, D., Chen, C., Li, S., & Pan, G. (2013). From taxi GPS traces to social and community dynamics: A survey. *ACM Computing Surveys, 46*(2). 17:1–17:34.

Chen, Y., & Krumm, J. (2010). Probabilistic modeling of traffic lanes from GPS Traces. In *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems* (pp. 81–88).

Cheng, H.-Y., Gau, V., Huang, C.-W., & Hwang, J.-N. (2012). Advanced formation and delivery of traffic information in intelligent transportation systems. *Expert Systems with Applications, 39*(9), 8356–8368.

Coifman, B. (2003). Identifying the onset of congestion rapidly with existing traffic detectors. *Transportation Research Part A: Policy and Practice, 37*(3), 277–291.

Comune di Pisa. Open Data. (2016). Accessed June, 1, 2016. http://opendata.comune.pisa.it/content/incidenti-stradali-2013-dato-geografico.

D'Andrea, E., Ducange, P., Lazzerini, B., & Marcelloni, F. (2015). Real-time detection of traffic from twitter stream analysis. *IEEE Transactions on Intelligent Transportation Systems, 16*(4), 2269–2283.

Delling, D., Sanders, P., Schultes, D., & Wagner, D. (2009). Engineering route planning algorithms. In J. Lerner, D. Wagner, & K. A. Zweig (Eds.), *Algorithmics of large and complex networks* (pp. 117–139). BerlinHeidelberg: Springer.

Dia, H., & Thomas, K. (2011). Development and evaluation of arterial incident detection models using fusion of simulated probe vehicle and loop detector data. *Information Fusion, 12*(1), 20–27.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik, 1*(1), 269–271.

de Fabritiis, C., Ragona, R., & Valenti, G. (2008). Traffic estimation and prediction based on real time floating car data. In *11th international IEEE conference on intelligent transportation systems* (pp. 197–203).

Fathi, A., & Krumm, J. (2010). Detecting road intersections from GPS traces. In S. I. Fabrikant, T. Reichenbacher, M. van Kreveld, & C. Schlieder (Eds.), *Geographic information science* (pp. 56–69). BerlinHeidelberg: Springer.

Ghosh, B., & Smith, D. P. (2014). Customization of automatic incident detection algorithms for signalized urban arterials. *Journal of Intelligent Transportation Systems, 18*(4), 426–441.

Globaltruth. Your car's hidden 'black box' and how to keep it private. (2016). Accessed June, 1, 2016. http://www.globaltruth.net/your-cars-hidden-black-box-and-how-to-keep-it-private.

Google Maps (2016). Accessed June, 1, 2016. https://www.google.it/maps.

GraphHopper Route Planner (2016). Accessed June, 1, 2016. www.graphhopper.com.

Here Map. Current traffic. (2016). Accessed June, 1, 2016. https://wego.here.com/traffic.

Hofmann-Wellenhof, B., Lichtenegger, H., & Collins, J. (2012). *Global positioning system: theory and practice*. Wien NewYork: Springer-Verlag.

Kamran, S., & Haas, O. (2007). A multilevel traffic incidents detection approach: Identifying traffic patterns and vehicle behaviours using real-time gps data. In *Proceedings of the IEEE intelligent vehicles symposium* (pp. 912–917).

Krajzewicz, D., Hertkorn, G., Rössel, C., & Wagner, P. (2002). SUMO (Simulation of Urban MObility) - an open-source traffic simulation. In *Proceedings of the 4th middle east symposium on simulation and modelling* (pp. 183–187).

Krajzewicz, D., Erdmann, J., Behrisch, M., & Bieker, L. (2012). Recent development and applications of SUMO - simulation of urban mobility. *International Journal On Advances in Systems and Measurements, 5*(3&4), 128–138.

Krauß, S., Wagner, P., & Gawron, C. (1997). Metastable states in a microscopic model of traffic flow. *Physical Review E, 55*(5), 5597–5602.

Krauß, S. (1998). *Microscopic modeling of traffic flow: investigation of collision free vehicle dynamics*. Germany: University of Cologne Doctoral dissertation.

Li, X., Han, J., Lee, J.-G., & Gonzalez, H. (2007). Traffic density-based discovery of hot routes in road networks. In D. Papadias, D. Zhang, & G. Kollios (Eds.), *Advances in spatial and temporal databases* (pp. 441–459). BerlinHeidelberg: Springer.

Liu, S., Liu, Y., Ni, L., Li, M., & Fan, J. (2013). Detecting crowdedness spot in city transportation. *IEEE Transactions on Vehicular Technology, 62*(4), 1527–1539.

Liu, S., Yue, Y., & Krishnan, R. (2013). Adaptive collective routing using gaussian process dynamic congestion models. In *Proceedings of the 19th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 704–712).

Liu, S., Yue, Y., & Krishnan, R. (2015). Non-myopic adaptive route planning in uncertain congestion environments. *IEEE Transactions on Knowledge and Data Engineering, 27*(9), 2438–2451.

Nominatim, Open Street Map (2016). Accessed June, 1, 2016. http://nominatim.openstreetmap.org.

Open Street Map (2016). Accessed June, 1, 2016. www.openstreetmap.org.

Perera, K., & Dias, D. (2011). An intelligent driver guidance tool using location based services. In *2011 IEEE International conference on spatial data mining and geographical knowledge services (ICSDM)* (pp. 246–251).

Sakaki, T., Okazaki, M., & Matsuo, Y. (2013). Tweet analysis for real-time event detection and earthquake reporting system development. *IEEE Transactions on Knowledge and Data Engineering, 25*(4), 919–931.

Schäfer, R.-P., Thiessenhusen, K.-U., & Wagner, P. (2002). A traffic information system by means of real-time floating-car data. In *Proceedings of the 9th ITS world congress* (pp. 1–8).

Schroedl, S., Wagstaff, K., Rogers, S., Langley, P., & Wilson, C. (2004). Mining GPS traces for map refinement. *Data Mining and Knowledge Discovery, 9*(1), 59–87.

Sirvio, K., & Hollmén, J. (2008). Spatio-temporal road condition forecasting with markov chains and artificial neural networks. In E. Corchado, A. Abraham, & W. Pedrycz (Eds.), *Hybrid artificial intelligence systems* (pp. 204–211). BerlinHeidelberg: Springer.

Srinivasan, D., Jin, X., & Cheu, R. L. (2004). Evaluation of adaptive neural network models for freeway incident detection. *IEEE Transactions on Intelligent Transportation Systems, 5*(1), 1–11.

The Smarty project (2015). Accessed June, 1, 2016. http://www.smarty.toscana.it.

Viriyasitavat, W. (2015). Quantifying the benefit of accident notification application in freeway VANETs. In *International conference and workshop on computing and communication* (pp. 1–3).

Wen, W. (2010). An intelligent traffic management expert system with RFID technology. *Expert Systems with Applications, 37*(4), 3024–3035.

Work, D. B., & Bayen, A. (2008). Impacts of the mobile internet on transportation cyberphysical systems: traffic monitoring using smartphones. In *National workshop for research on high-confidence transportation cyber-physical systems: automotive, aviation, & rail* (pp. 18–20).

Yoon, J., Noble, B., & Liu, M. (2007). Surface street traffic estimation. In *Proceedings of the 5th international conference on mobile systems, applications and services* (pp. 220–232).

Yuan, J., Zheng, Y., Xie, X., & Sun, G. (2011). Driving with knowledge from the physical world. In *Proceedings of the 17th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 316–324).