applied sciences

MDPI

Article

Deep Learning System for Vehicular Re-Routing and Congestion Avoidance

Pedro Perez-Murueta [1,*], Alfonso Gómez-Espinosa [1,*], Cesar Cardenas [2] and Miguel Gonzalez-Mendoza Jr. [3]

[1] Tecnologico de Monterrey, Escuela de Ingeniería y Ciencias, Av. Epigmenio Gonzalez 500, Querétaro 76130, Mexico
[2] Tecnologico de Monterrey, Escuela de Ingeniería y Ciencias, Av. General Ramon Corona 2514, Jalisco 45138, Mexico
[3] Tecnologico de Monterrey, Escuela de Ingeniería y Ciencias, Av. Lago de Guadalupe KM 3.5, Estado de México 52926, Mexico
* Correspondence: pperezm@tec.mx (P.P.-M.); agomeze@tec.mx (A.G.-E.); Tel.: +52-442-332-4056 (P.P.-M.); +52-442-238-3302 (A.G.-E.)

Received: 29 May 2019; Accepted: 3 July 2019; Published: 5 July 2019

Abstract: Delays in transportation due to congestion generated by public and private transportation are common in many urban areas of the world. To make transportation systems more efficient, intelligent transportation systems (ITS) are currently being developed. One of the objectives of ITS is to detect congested areas and redirect vehicles away from them. However, most existing approaches only react once the traffic jam has occurred and, therefore, the delay has already spread to more areas of the traffic network. We propose a vehicle redirection system to avoid congestion that uses a model based on deep learning to predict the future state of the traffic network. The model uses the information obtained from the previous step to determine the zones with possible congestion, and redirects the vehicles that are about to cross them. Alternative routes are generated using the entropy-balanced k Shortest Path algorithm (EBkSP). The proposal uses information obtained in real time by a set of probe cars to detect non-recurrent congestion. The results obtained from simulations in various scenarios have shown that the proposal is capable of reducing the average travel time (ATT) by up to 19%, benefiting a maximum of 38% of the vehicles.

Keywords: traffic congestion detection; minimizing traffic congestion; traffic prediction; deep learning; urban mobility; ITS; Vehicle-to-Infrastructure

1. Introduction

Excessive population growth in urban areas is one of the biggest challenges for every government around the world. The congestion generated by public and private transport is the most important cause of air pollution, noise levels, and economic losses caused by the time used in transfers, among others. For example, the inhabitants of Mexico City lose an estimated 23 h per month in transfers, which translates into losses of more than 1.5 billion dollars per year, which means 1% of the Mexico City's contribution to the country's GDP, a very significant figure for a developing country [1].

The implementation of vehicular network standards and advances in wireless communication technologies has enabled the implementation of intelligent transport systems (ITS). One of the objectives of ITS is the real time traffic management using vehicle data collected from the road infrastructure. With this data, it is sought to characterize the traffic and thus be able to detect, control, and minimize traffic congestion. The main challenge of this approach is to forecast the congestion and re-route the vehicles without causing new congestion in other places [2].

Commercial solutions such as Waze [3] and Google Maps [4] are capable of providing alternative routes from origin to destination based on the traffic information published by users. Although these systems are capable of predicting long-term traffic congestion as well as its duration, they are often used with reactive solutions that still cannot prevent congestion. In addition, the routes suggested to users are based exclusively on the algorithm of the shortest route to their destination, without taking into account the impact that the re-routing has on future traffic conditions [5].

In this work, we present a predictive congestion avoidance by re-routing system that uses a mechanism based on deep learning with the goal of characterizing the future traffic conditions to make an early detection of congestion. Based on these predictions and a short route generation algorithm, alternatives for the vehicles that are about to cross the possible congested areas are provided. In addition, the system uses vehicle to infrastructure communications (V2I) for the early detection of non-recurrent congestion. The results obtained from simulations in synthetic scenarios have shown that the proposal is capable of reducing the average travel time (ATT) by up to 19%, benefiting a maximum of 38% of the vehicles. The rest of this document is organized as follows. In Section 2, previous research in this field is summarized. Section 3 presents the details of the proposal. Section 4 presents the tools used in the development of the proposal and the scenarios used for the evaluation. Section 5 shows the obtained results. Finally, Section 6 presents conclusions and future work.

2. Related Work

CoTEC [6] (Cooperative Traffic congestion detECtion) is a cooperative vehicle system based on vehicle-to-vehicle (V2V) communication. This system uses fuzzy logic to detect local traffic conditions, using beacon messages received from surrounding vehicles. When detecting congestion in the area, CoTEC activates a cooperative process that shares and correlates the individual estimates made by the different vehicles, thus achieving a characterization of the degree of traffic congestion. Tested in large scenarios, COTEC obtained good results in the detection of congestion; however, it does not integrate any vehicle re-routing strategy.

Araujo et al. [7] present a system, based on CoTEC, called CARTIM (identification and minimization of traffic congestion of cooperative vehicles). CARTIM, like CoTEC, uses fuzzy logic to characterize road traffic, adding a heuristic that seeks to reduce drivers' travel time by allowing them to modify their travel routes, thus improving vehicle flow in the event of congestion. The authors report that CARTIM achieves a reduction of 6% to 10% in ATT.

Urban CONgestion DEtection System (UCONDES) [8] is another proposal based on V2V communications that use an artificial neural network (ANN) designed to detect and classify existing levels of congestion in urban roads. The proposal also adds a mechanism that suggests new routes for drivers to avoid congested areas. The ANN uses the speed and traffic density of a street as input to determine the existing level of congestion: light, moderate, and congested. The vehicles receive the classification obtained, as well as a new route, through beacon messages so that the vehicle can alter its current route and thus avoid congested roads. The simulations carried out in a realistic Manhattan scenario showed a reduction of 9% to 26% of ATT.

SCORPION [9] (A Solution using Cooperative Rerouting to Prevent Congestion and Improve Traffic Condition) uses vehicle-to-infrastructure (V2I) communications. All vehicles send their information (id, current position, route, and destination) using 4G and LTE technologies to the nearest road side unit (RSU), which uses the k-nearest neighbor algorithm to classify street congestion. After determining the traffic conditions, a collective re-routing scheme plans new routes to those vehicles that will pass through congested areas. Like the UCONDES system, SCORPION performed simulations using a realistic Manhattan scenario, achieving a 17% reduction in the ATT. The aforementioned proposals present some limitations, such as a limited re-routing distance, no real-time mechanism for the detection of congestion or no mechanism for re-routing with a global perspective.

Pan et al. [5] propose a centralized system that uses the data obtained in real-time of the vehicles (position, speed, and direction) to detect traffic congestion. Once the congestion is detected, the vehicles are redirected according to two different algorithms. First, the shortest dynamic path (DSP), which re-routes vehicles using the shortest routes that also, has the lowest travel time. The authors also mention that a defect of this algorithm is the possibility of moving the congestion to another point. To solve this, a second algorithm, random k-shortest path (RkSP), randomly chooses a route between k shorter routes. The objective of this algorithm is to avoid switching the congestion from one place to another and, in this way, to balance the redirected traffic between several routes. This scheme does not implement a mechanism in real time to detect congestion as it occurs; it is only able to detect it during the next phase of re-routing.

One way to improve the above mentioned solutions, it is to forecast the state of vehicular traffic and, in this way, anticipate congestion. Kumar [10] proposes a traffic prediction system using the Kalman filtering technique. The predictive model uses historical and real-time information when making the prediction. Despite showing very good results in terms of prediction, the proposal does not include redirection strategies in case of congestion. Another point to consider is that this model works in a single street segment, which does not allow to determine its efficiency on a global scale (the entire vehicular network).

From the perspective of machine learning, traffic prediction is a spatio-temporal sequence prediction [11]. Recently, long short-term memory networks (LSTM) have gained popularity as a tool for the prediction of spatio-temporal sequences [12]. LSTM are a special type of recurrent neural network (RNN) that have been shown to be able to learn long-term dependencies [13–15]. Hochreiter and Schmidhuber [14] introduced them. The RNNs work very well with a wide variety of problems and, because of this, are widely used today. LSTMS are specifically designed to avoid the problem of long-term dependency. Each cell acts as an accumulator of state information and controlled by a set of cells that determines the flow of information. There is a gateway that allows you to control whether the incoming information is going to be accumulated or not. There is also an oblivion door, which allows us to determine if the state of the previous cell will be taken into account. In addition, an output cell determines whether the output of the cell is to be propagated to the final state. The cell type and control gates are the most important advantage of the LSTMs since they get the gradient trapped in the cell and prevent it from disappearing too fast, which is a critical problem of the basic RNN model [15]. Xingjian et al. [12] developed a multivariable version of LSTM that they called ConvLSTM, which has convolutional structures in both state-to-state and state-to-state transitions. By stacking multiple ConvLSTM layers, they were able to build a model that proved to have excellent spatio-temporal sequence prediction capability, suitable for making predictions in complex dynamic systems.

DIVERT [16] is a distributed hybrid vehicle redirection system to avoid congestion. This system uses a server and Internet communication to determine a precise global view of traffic. Although, DIVERT presents improvements in ATT, the comparison is made with respect to a centralized re-routing system and does not make any comparison against a scenario without any strategy.

Next route routing (NRR) [17] extends the functionality of the SCATS system (Sidney Coordinate Adaptive Traffic Systems) by adding vehicle congestion and re-routing detection mechanisms. The re-routing decision is made taking into account the travel destination and the current local traffic conditions. The proposed architecture allows the positive impacts of re-routing to spread over a larger area. The results obtained show a reduction of 19% in ATT. It is necessary to emphasize two points on this proposal. First, it is an extension of a previous system (SCATS), it is not an autonomous system. Second, it requires that all vehicles provide information to know the real state of the traffic.

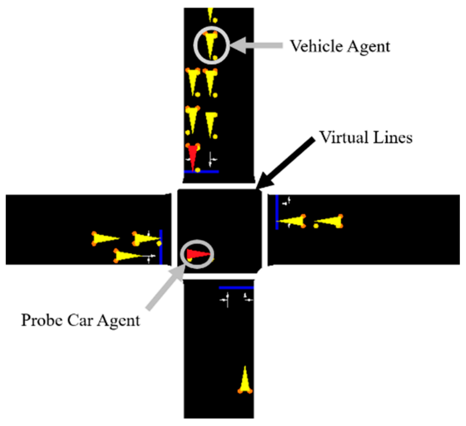
3. Proposed Solution

The novelty of the proposal lies in the combination of various elements. In the first place, the system does not require a continuous flow of information both for the detection of possible congested areas and for the generation of routes due to the use of a deep learning prediction model. The model uses a deep ConvLSTM layer architecture that has been shown to be able to handle prediction problems in spatio-temporal sequences similar to the prediction of vehicular traffic [12]. This architecture and the use of the EBkSP algorithm to generate alternative routes, allows us to create a global strategy for congestion management. Finally, information obtained from the probe cars allows us to detect early non-recurrent congestion.

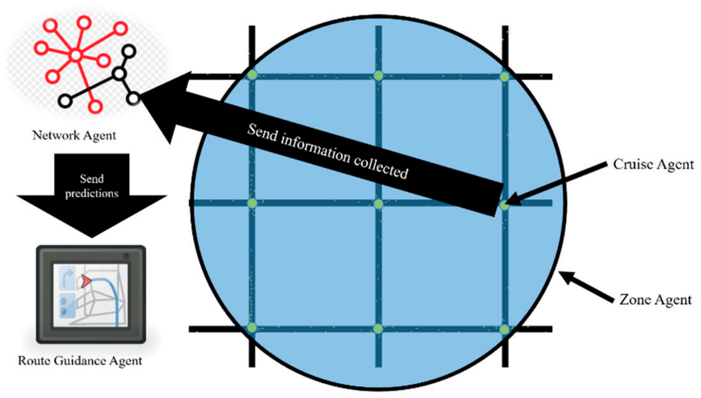
3.1. Architecture Proposed

The architecture of the proposal takes into consideration that traffic is a dynamic problem and of geographical distribution in which several autonomous entities interact. Based on the above, we opted for a hybrid architecture based on agents. The use of agents significantly improves the analysis and design of problems similar to the problem of traffic detection [18]. In addition, the need for dynamic traffic management involving a large number of vehicles requires more reactive agents than cognitive agents. Unlike cognitive agents, reactive agents have no representation of their environment. Reactive agents can cooperate and communicate through interactions (direct communication) or through knowledge of the environment.

The Vehicle Agent (VA) is responsible for requesting the best possible route to the Route Guidance Agent (RGA), both at the beginning of the trip, and whenever an incident exists in the assigned route. One out of 10 vehicle is designated as Probe Car Agent (PCA). These agents have, as an additional task, the responsibility of reporting at each intersection the delay experienced in traveling that street segment to the corresponding Cruise Agent (Figure 1a). The Cruise Agent (CA) gathers data from those PCAs that cross the streets that it controls. Every 300 s, it uses the collected data to generate a report, which is sent to the Zone and Network Agents.



(**a**)



(**b**)

Figure 1. (**a**) As soon as a Probe Car Agent crosses an intersection, it sends to the Cruise Agent the status of that street; (**b**) the Cruise Agent transmits to the Network Agent, who makes the prediction.

The Zone Agent (ZA) detects congestion in an area of the city, and send alert to vehicles (Figure 1b). The Network Agent (NA) uses the deep learning model to predict the future state of the network for the next 300 s. Finally, the Route Guidance Agent generates alternative routes.

3.1.1. Starting a Trip

At the beginning of the trip, the Vehicle Agent sends a route suggestion message to the Zone Agent. This message contains its id, the origin and destination of its trip. The Zone Agent forwards this message to the Route Guidance Agent. The RGA verifies if there are alternatives generated. If there are none, it generates a set of k shorter path (kSP) using a multithreaded algorithm [19]. With the set of kSP, the RGA sends the shortest route less suggested until that moment to the Zone Agent. The Zone Agent forwards the response to the VA, but first registers the id of the vehicle and the route that will follow. This information will be used to send alert messages when congestion is detected in the area (Figure 2).

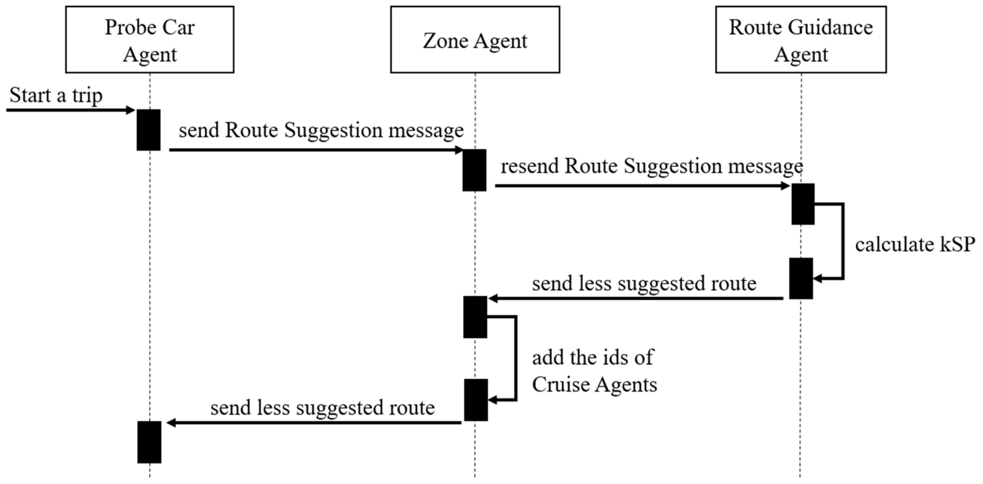

Probe Car Agent
Zone Agent
Route Guidance Agent
Start a trip
send Route Suggestion message
resend Route Suggestion message
calculate kSP
send less suggested route
add the ids of Cruise Agents
send less suggested route

Figure 2. Sequence diagram of a typical initial routing process.

A similar exchange of messages is made between the Probe Car Agent, Zone Agent, and Route Guidance agent. The difference is that: when the Zone Agent forwards the route, it adds a list of Cruise Agents to which Probe Car Agent must report on the situation of the streets through which it crosses.

3.1.2. Predicting the State of the Vehicular Network

One of the responsibilities of Probe Car agents is to report to the cruise agents the delay experienced when traveling one of the streets that the cruise agent controls. The algorithm used to calculate the delay is based on the algorithm defined by Chuang [20] and is modified to detect and report non-recurrent bottlenecks. The idea behind this is that a trip is an alternate sequence of periods in which the car is stationary and periods in which it is moving. A STOP event indicates when a vehicle has moved from a complete stop and can be described with the GPS position and the time the event occurred. A GO event indicates when a stopped vehicle moves (Figure 3).

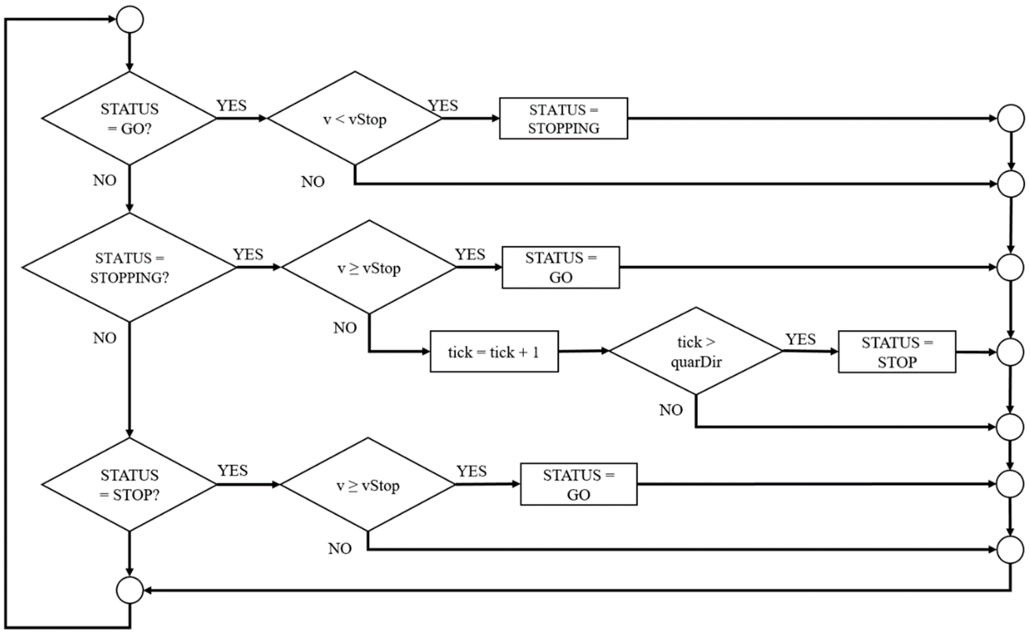


Figure 3. Flowchart showing the state machine used to detect the GO-STOP events [19].

The Probe Car Agent is continuously recording the time that each GO-STOP event happens until it reaches an intersection. As soon as the PCA passes an intersection, it calculates the total delay it got when traveling on that street. The total delay is the sum of the time differences between a GO event and the previous STOP event as show in (1). The result, as well as the identifier of the street segment, are sent to the Cruise Agent of the intersection that the PCA has just crossed.

$$Delay_{vehicle} = \sum_{j=2} \left(TIME\_GO_j - TIME\_STOP_{j-1} \right) \tag{1}$$

Every 300 s, the Cruise Agent processes all the reports received and determines the average delay that exists in that segment of the street. The calculation of the average delay is based on (2).

$$Delay_{avg} = \frac{\sum_{i=1}^{N} Delay_{vehicle_i}}{N} \tag{2}$$

The CA sends an Update Message to the Network Agent. This message contains a time stamp, IDs of the streets it controls, as well as the average delay of each of the streets.

The Network Agent is the responsible for making predictions using a deep five-layer architecture. As mentioned, from the perspective of Machine Learning, traffic prediction is a spatio-temporal sequence prediction [11]. If we assume that the traffic prediction problem in a vehicular network is a dynamic system on a spatial region represented by MxN adjacency matrix of M rows and N columns, and within each cell of this matrix, there exist P measurements, which vary with time. Based on the above, it can be observed that the problem of traffic prediction requires a prediction model that is capable of handling a sequence that contains spatial and temporal structures. When we see the results obtained by the ConvLSTM networks to predict the intensity of rain in a local region during a relatively short period, we decided to build a deep architecture with this type of network, since it has demonstrated to have excellent spatio-temporal sequence prediction capability, suitable for making predictions in complex dynamic systems. The process of training and defining the architecture is explained in detail in Section 4.

The Network Agent uses the information received from the Cruise Agents to rectify the previous forecast and generate a new forecast using the ConvLSTM five-layer deep architecture mentioned above. This forecast will be valid for the next 300 s.

If a non-recurrent congestion message is received during this time, the message will be stored for later use if a more recent update message does not arrive. Finally, the Network Agent sends the new forecast to both the Zone Agent and the Route Guidance Agent. The ZA analyzes the forecast to determine if there is a possibility of future congestion. While the RGA uses the forecast to generate new alternative routes.

3.1.3. Congestion Detection

The system contemplates two possible types of congestion: recurrent and non-recurrent. The proposal has a defined strategy for each of them.

We will start by talking about the strategy for recurrent congestion. Every 300 s, the Zone Agent receives a new forecast from the Network Agent. The ZA analyzes the forecast in search of streets that may have an average delay greater than the limit determined by the system. Once these streets are detected, the Zone Agent sends a message of congestion to vehicles that have one or more of these streets on their route (Figure 4). The Zone Agent sends a route update message to the Route Guidance Agent and all Vehicle Agents who have one or more of the congested streets on their routes. Any VA that receives a route update message should verify how far it is from the congested highway. If the distance is less than a limit defined by the system, it will send a route suggestion message to the Zone Agent. In this case, the message will indicate the current position of the vehicle as point of origin. Meanwhile, as soon as RGA receives the route update message, it invalidates all previously generated kSPs.

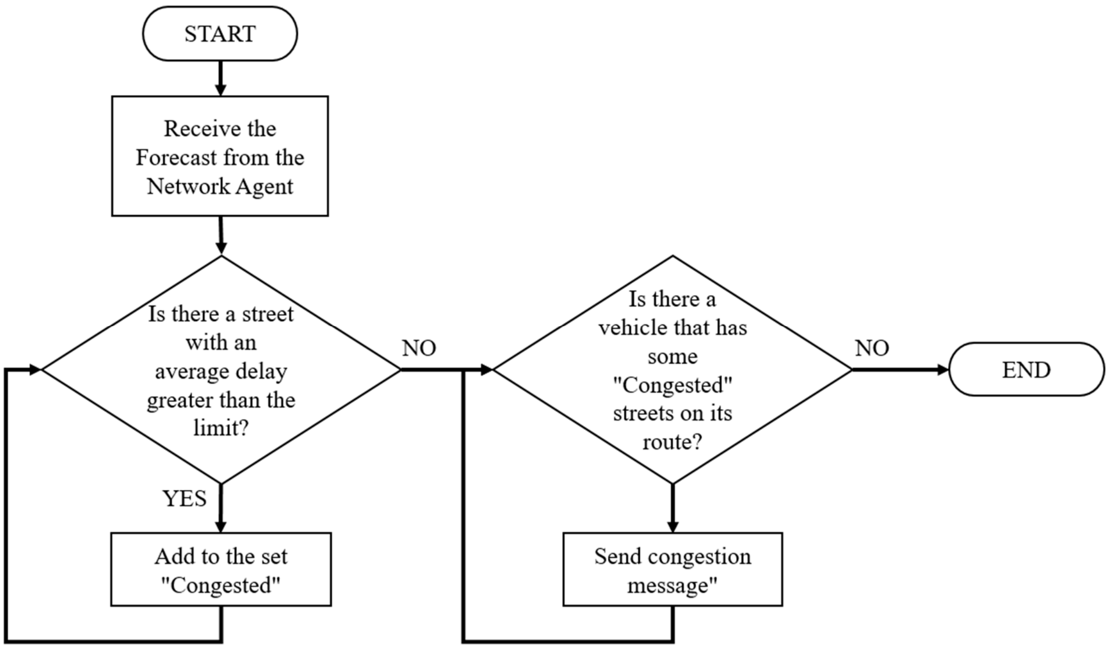

START
Receive the Forecast from the Network Agent
Is there a street with an average delay greater than the limit?
NO
YES
Add to the set "Congested"
Is there a vehicle that has some "Congested" streets on its route?
NO
END
Send congestion message"

Figure 4. Flowchart illustrating the selection of congested streets and sending notices from the Zone Agent.

With regard to non-recurrent congestion, the Probe Car Agent carries out detection. If the PCA detects an excessive number of STOP events while crossing a street, it will proceed to send a non-recurrent congestion message to both the Zone Agent and the Network Agent. This message contains a timestamp, street ID, and delay experienced so far. The Zone Agent forwards a non-recurrent congestion message to the Route Guidance Agent and to all Vehicle Agents that that street has on its route. In the case of the RGA, it will verify if the message has updated information and, if so, it will invalidate only the kSP containing the ID of the congested street and update the information of that street in its forecast. Any VA that receives a non-recurrent congestion message should verify how far away it is from the congested road. If the distance is less than a limit defined by the system, it will send a route suggestion message to the Zone Agent. In this case, the message will indicate the current position of the vehicle as point of origin. For its part, the NA will keep this report for later use, during the process of generating a new forecast.

4. Performance Evaluation

4.1. Tools Used

The set of tools used for traffic simulation is composed of two main components: Simulation of Urban MObility (SUMO) and TraCI4J. SUMO [21] is an open source program for urban traffic simulation. It is a microscopic simulator; that is, each vehicle is modeled explicitly, has its own route and moves independently through the network. It is mainly developed by the Institute of Transportation Systems. The version used in this work was version 0.25.0, published in December 2015. TraCI4J [22] is an Application Program Interface (API) developed for SUMO that allows communication in time of execution between the proposed system and the simulator. TraCI4J was developed by members of AppeAL (Applied Pervasive Architectures Lab) of Politecnico di Torino. This API offers a higher level of abstraction and better interoperability than other APIs developed for the same task.

Keras was used for the implementation and training of deep architecture. Keras [23] is a high-level neural network API, developed in Python and capable of running over TensorFlow, CNTK, or Theano. In our case, and intending to take advantage of a server with two NVidia GTX 1080 cards, we chose to use TensorFlow [24] as a backend.

4.2. Test Scenario

After reviewing previous works, we observed that one of the most used scenarios to evaluate proposals is the scenario known as Manhattan (or grid type) [25–27]. In our case, two synthetic networks were designed based on the Manhattan scheme: one of the 4×4 (M4) and another 5×5 (M5). For example, 5×5 means that this map has five intersections on the vertical and horizontal axes. All segments of the street have a length equal to 200 m and each of them consists of two lanes with opposite directions.

4.3. Simulation Setup

SUMO includes a set of applications that help prepare and perform the simulation of a traffic scenario. A simulation scenario can contain a large number of vehicles and their routes. Even for small areas, such as the test traffic networks, it is almost impossible to define traffic demand manually. For this reason, we use two of the tools included in SUMO, "randomTrips.py" [28] and duarouter [29], to generate the traffic demand of a full day for the synthetic networks M4 and M5. With the intention of simulating the traffic of one day as realistic as possible, we define three traffic flows: LIGHT (5–20% of maximum capacity), MEDIUM (60–70% of maximum capacity), and HEAVY (80–95% of maximum capacity). The distribution of these traffic flows are shown in Table 1.

Table 1. Traffic flows simulated by time of day

Hours	Traffic Flow
0–5	LOW
6–7	MEDIUM
8	HIGH
9–14	MEDIUM
15	HIGH
16–19	MEDIUM
20	HIGH
21–22	MEDIUM
23	LOW

Depending on the flow defined for a certain period of the day, a random amount of trips are made and distributed evenly throughout that period. A trip describes the movement of a vehicle from a street of origin to a street destination, time of departure, and route followed. Although the trips are generated randomly, each one of them meets a series of requirements:

1. They have a minimum distance. In the case of the M4 network, the minimum is 600 m. While for the M5 network, the minimum is 800 m.
2. The start and end streets are selected at random. Nonetheless, the tools allow favors to the streets that are in the periphery of the network.
3. The algorithm used to generate the route is the Dijkstra [30] algorithm.

The trips generated are stored in a configuration file that defines the traffic demand that will be used in the simulation.

To generate the dataset necessary for deep architecture training, we perform 2400 simulations using the previously generated traffic demand file, but using a different pseudo-random number as seed for each simulation. During these simulations, all the vehicles reported the delay when traveling on a street the segment (1). With these reports, every 300 s, an adjacency matrix is generated with the average delays of each of the streets of the synthetic network (2). The simulation of a full day generates 288 matrices of this type containing the traffic behavior of the 24 h. The values are normalized using the Min-Max technique (3) and stored in a file for later use.

$$Z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)} \tag{3}$$

The files obtained are divided at random into two groups: training (80%) and validation (20%). Regardless of the group, each file contains a spatio-temporal sequence of P measurements of a region of size M × N. For example, for the M5 network we are talking about 288 measurements of a 5 × 5 region.

Initially, a two-layer architecture was implemented and evaluated. The training files are segmented using a 20-measurements-wide sliding windows (5 for input and 15 for prediction). The initial network consisted of two ConvLSTM layers with each layer containing 64 hidden states and 3 × 3 kernel. We used a per-GPU batch size of 16 and trained with Adam optimizer, setting learning rate to 10^{-4}. This architecture worked correctly, achieving 75% accuracy. To increase the accuracy of the model, a new ConvLSTM intermediate layer was added. This process is repeated four times more, using the same hyperparameters each time. After several test with different configurations, it was determined that the best configuration was a deep five-layer ConvLSTM network (Figure 5), with an accuracy of 88%. This architecture proved to have a more accurate spatiotemporal sequence prediction capability, which makes it suitable to provide predictions in complex dynamic systems such as the problem of traffic flow prediction.

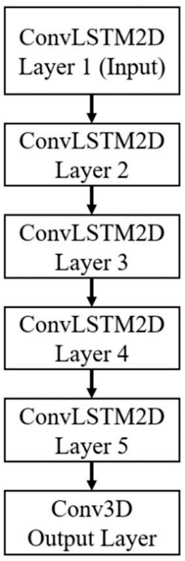


Figure 5. Final deep architecture.

With the intention of to obtain baselines to determine the efficiency of the proposal, we decided to run two groups of simulation, each with 100 simulations were carried out. The first group has no components of the proposal were used. However, this time, SUMO was configured to generate an output file that contained all the information related to each vehicle's trip: the departure time, the time the vehicle wanted to start, the time of arrival, the total duration of the trip, the distance traveled, the number of times the route changed, among other registered values. With these values as a basis, it was possible to determine both the general and the individual performance.

4.4. Performance Metrics

As we mentioned at the beginning of this document, the goal of the proposed sample is to mitigate the congestion of the streets and, at the same time, minimize the average travel time (ATT). The ATT is calculated using (4). We use this metric because its reduction leads to lower fuel consumption, greater economic growth, and better life experience for citizens [31].

$$Time_{avg} = \frac{Time_{sum}}{K} = \frac{\sum_{i=1}^{K} Time_{vehivle_i}}{K} \tag{4}$$

5. Results

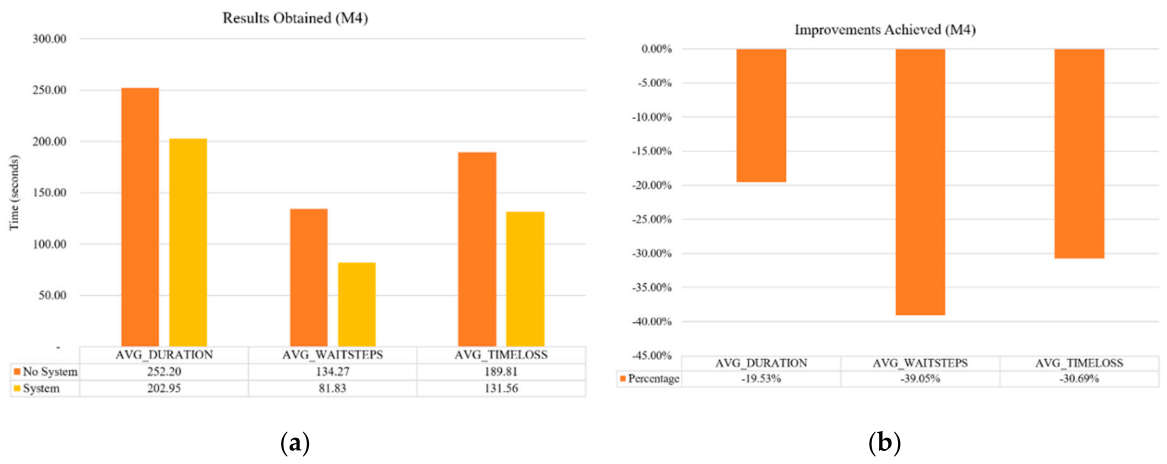
The second group were executed again, using, this time, the components of the proposal with the parameters presented in Table 2. It is also necessary to highlight that:

1. Only 10% of the population reports the delay they experience when crossing a street (probe cars) and only they receive notifications of congestion on their travel route.
2. If the system does not have updated information of a certain area or street, a forecast is generated on the status of that area for the next 300 s.
3. All the simulations continue until all vehicles complete their trips.

Table 2. Parameters used in the evaluation

Parameter	**Value**
Period	Frequency of triggering the re-routing; by default period = 300 s.
Threshold	Maximum delay allowed per street; by default delay = 25 s.
Level L	Distance that should be the vehicles of the congested segment to request re-routing; by default L = 5.
k Paths	Max number of alternative paths for each vehicle; by default k = 5.
Probe cars	10% of the population.

In both test traffic networks, M4 and M5, we observed important reductions in the average travel time, the waiting time (the time when the vehicle speed was below 0.1 m/s) and the loss of time (the time lost due to driving below the ideal speed). In the case of the M4 network, we observed a reduction of 19.53% in the average time of travel, as well as a reduction of 39.05% in the waiting time and a 30.69% in time loss (Figure 6).

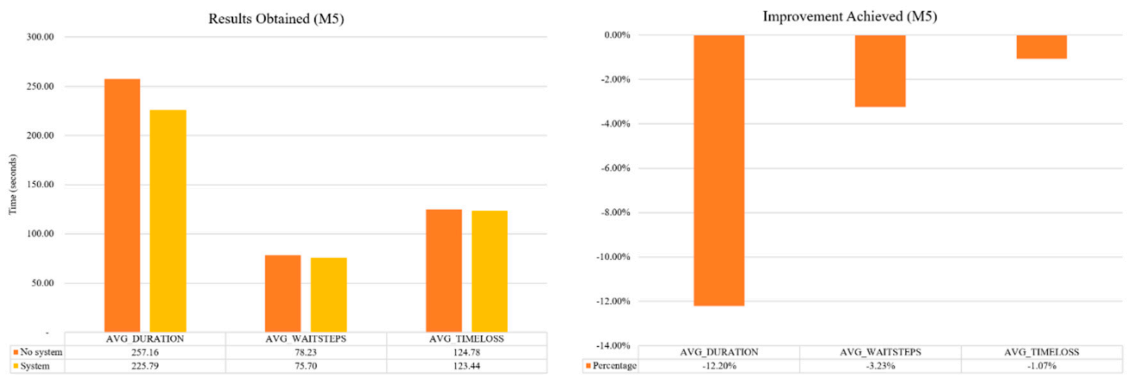


	AVG_DURATION	AVG_WAITSTEPS	AVG_TIMELOSS
🟧 No System	252.20	134.27	189.81
🟨 System	202.95	81.83	131.56

	AVG_DURATION	AVG_WAITSTEPS	AVG_TIMELOSS
🟧 Percentage	-19.53%	-39.05%	-30.69%

Figure 6. (**a**) Results obtained for M4 network, both without using the proposal (no system) and using it (system); (**b**) reductions obtained for different time measures.

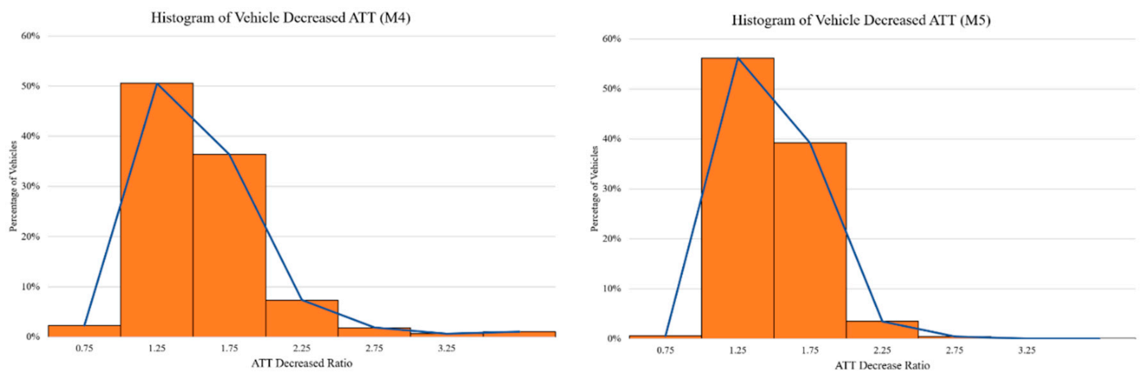
With regard to the M5 network, the results obtained were also very good. A reduction of 12.20% in ATT was obtained, as well as a reduction of 3.23% in the waiting time and of 1.07% in time loss (Figure 7).



(**a**) (**b**)

Figure 7. (**a**) Results obtained for M5 network, both without using the proposal (no system) and using it (system); (**b**) reductions obtained for different time measures.

Figure 8 shows the histograms of the improvements of ATT under normal traffic conditions when we use the proposed system. In this case, the improvement ratio is defined as *ratio* = *oldtime/newtime*. For example, a ratio of two means that the vehicle was able to complete its trip in half of the time, while a ratio of three means that it finished its trip in one-third of the previous time. As we observed, a large number of vehicles, 51% (M4) and 56% (M5), saved 20% of the travel time (ratio 1.25). Although we can also observe that, there are vehicles that could avoid the worst part of the traffic by completing their trip two or three times faster (relations 2 and 3).



(**a**) (**b**)

Figure 8. Histogram of gain in travel times: (**a**) improvements obtained for the M4 network; (**b**) improvements obtained for the M5 network.

6. Conclusions and Future Work

The congestion caused by public and private transport is the main cause of many of the problems that cities face today. The best approach to solve this congestion is a more efficient use of the installed road infrastructure. The use of intelligent transportation systems (ITS) technologies can optimize the route, reducing distances, increasing safety in transport, reducing the time spent in traffic jams, reducing fuel consumption, and improving air quality. In this document, we present a platform that continuously monitors the traffic flow status in a specific geographic area and provides a congestion detection and warning service. This allows vehicles to adjust their route from their current point to their destination when the system detects congestion on this route, even when the vehicle is already moving. The proposed system also considers those situations for which it is not possible to obtain updated information, using a real-time prediction model based on a deep neural network. The results obtained from a series of simulations carried out on two synthetic scenarios show that the proposal is capable of generating improvements in ATT for each of the scenarios analyzed, achieving better results than other existing proposals.

It is important to mention that the proposal is not expensive, nor difficult to implement, since it does not require specialized infrastructure that requires installation and maintenance. The proposal requires a distributed computational architecture to accommodate the different agents that make up the proposal, and a small number of cars equipped with smartphones operating as probe cars. A good choice of probe cars would be public transport vehicles, since when traveling through the main streets of the city; they could provide relevant information, thus benefiting a large part of the population. In addition, for those streets where there is no updated information, the platform can generate good approximations of the real traffic conditions using the predictive model developed in the proposal.

As part of future work, we will further evaluate the proposal with real data. For this, we have already selected a sub-scenario of TAPASCologne 0.17.0 [32]. TAPAS Cologne is an open project that provides a large-scale data set that allows you to make, using SUMO, a very realistic urban vehicle simulation. This scenario uses a real Cologne map drawn from OpenStreetMap [33] is used and generates a traffic demand of 6:00 a.m. at 8:00 a.m., using the methodology of Simulation of Travel and Activity Patterns (TAPAS) and the dynamic assignment algorithm called Gawron.

Author Contributions: Conceptualization, P.P-M.; Methodology, P.P.-M.; Software, P.P.-M.; Data Curation, P.P.-M.; Validation, P.P.-M.; Formal Analysis, P.P.-M.; Investigation, P.P.-M.; Resources, P.P.-M.; Visualization, P.P.-M.; Writing—original draft, P.P.-M.; Writing—review & editing, P.P.-M. and A.G.-E.; Supervision, A.G.-E., C.C., and M.G.-M. The manuscript was finalized through contributions from all authors, and all authors approved the final manuscript.

Funding: The authors would like to acknowledge the financial support of Tecnologico de Monterrey, in the production of this work.

Acknowledgments: The authors would like to thank Benjamin Valdes and Magdalena Navarro for the review of the manuscript and their valuable suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Cárdenas-Benítez, N.; Aquino-Santos, R.; Magaña-Espinoza, P.; Aguilar-Velazco, J.; Edwards-Block, A.; Medina Cass, A. Traffic Congestion Detection System through Connected Vehicles and Big Data. *Sensors* **2016**, *16*, 599. [CrossRef] [PubMed]
2. De Souza, A.M.; Yokoyama, R.S.; Maia, G.; Loureiro, A.; Villas, L. Real-Time Path Planning to Prevent Traffic Jam through an Intelligent Transportation System. In Proceedings of the 2016 IEEE Symposium on Computers and Communication (ISCC), Messina, Italy, 27–30 June 2016; pp. 726–731.
3. Waze. Available online: https://www.waze.com/es-419 (accessed on 1 April 2019).
4. Google Maps. Available online: https://www.google.com.mx/maps (accessed on 5 April 2019).
5. Pan, J.; Khan, M.A.; Popa, I.S.; Zeitouni, K.; Borcea, C. Proactive Vehicle Re-Routing Strategies for Congestion Avoidance. In Proceedings of the 2012 IEEE 8th International Conference on Distributed Computing in Sensor Systems, Hangzhou, China, 16–18 May 2012; pp. 265–272.
6. Bauza, R.; Gozálvez, J. Traffic Congestion Detection in Large-Scale Scenarios Using Vehicle-to-Vehicle Communications. *J. Netw. Comput. Appl.* **2013**, *36*, 1295–1307. [CrossRef]
7. Araújo, G.B.; Queiroz, M.M.; de LP Duarte-Figueiredo, F.; Tostes, A.I.; Loureiro, A.A. Cartim: A Proposal toward Identification and Minimization of Vehicular Traffic Congestion for Vanet. In Proceedings of the 2014 IEEE symposium on computers and communications (ISCC), Madeira, Portugal, 23–26 June 2014; pp. 1–6.
8. Meneguette, R.I.; Ueyama, J.; Krishnamachari, B.; Bittencourt, L. Enhancing Intelligence in Inter-Vehicle Communications to Detect and Reduce Congestion in Urban Centers. In Proceedings of the 20th IEEE symposium on computers and communication (ISCC), Larnaca, Cyprus, 6–9 July 2015; pp. 662–667.

9. De Souza, A.M.; Yokoyama, R.S.; Botega, L.C.; Meneguette, R.I.; Villas, L.A. Scorpion: A Solution Using Cooperative Rerouting to Prevent Congestion and Improve Traffic Condition. In Proceedings of the 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing, London, UK, 26–28 October 2015; pp. 497–503.
10. Kumar, S.V. Traffic Flow Prediction Using Kalman Filtering Technique. *Procedia Eng.* **2017**, *187*, 582–587. [CrossRef]
11. Ma, X.; Dai, Z.; He, Z.; Ma, J.; Wang, Y.; Wang, Y. Learning Traffic as Images: A Deep Convolutional Neural Network for Large-Scale Transportation Network Speed Prediction. *Sensors* **2017**, *17*, 818. [CrossRef] [PubMed]
12. Xingjian, S.; Chen, Z.; Wang, H.; Yeung, D.-Y.; Wong, W.-K.; Woo, W. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2015; pp. 802–810.
13. Graves, A. Generating Sequences with Recurrent Neural Networks. *arXiv* **2013**, arXiv:1308.0850.
14. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
15. Pascanu, R.; Mikolov, T.; Bengio, Y. On the Difficulty of Training Recurrent Neural Networks. In Proceedings of the 2013 International Conference on Machine Learning, Atlanta, GA, USA, 16–21 June 2013; pp. 1310–1318.
16. Pan, J.S.; Popa, I.S.; Borcea, C. Divert: A Distributed Vehicular Traffic Re-Routing System for Congestion Avoidance. *IEEE Trans. Mob. Comput.* **2017**, *16*, 58–72. [CrossRef]
17. Wang, S.; Djahel, S.; Zhang, Z.; McManis, J. Next Road Rerouting: A Multiagent System for Mitigating Unexpected Urban Traffic Congestion. *IEEE Trans. Intell. Transp. Syst.* **2016**, *17*, 2888–2899. [CrossRef]
18. Adler, J.L.; Blue, V.J. A Cooperative Multi-Agent Transportation Management and Route Guidance System. *Transp. Res. Part C Emerg. Technol.* **2002**, *10*, 433–454. [CrossRef]
19. Murueta, P.O.P.; Pérez, C.R.C.; Uresti, J.A.R. A Parallelized Algorithm for a Real-Time Traffic Recommendations Architecture Based in Multi-Agents. In Proceedings of the 2014 13th Mexican International Conference on Artificial Intelligence, Tuxtla Gutierrez, Mexico, 16–22 November 2014; pp. 211–214.
20. Chuang, Y.-T.; Yi, C.-W.; Lu, Y.-C.; Tsai, P.-C. ITraffic: A Smartphone-Based Traffic Information System. In Proceedings of the 2013 42nd International Conference on Parallel Processing, Lyon, France, 1–4 October 2013; pp. 917–922.
21. SUMO–Simulation of Urban Mobility. Available online: http://www.dlr.de/ts/en/desktopdefault.aspx/tabid-9883/16931_read-41000/ (accessed on 3 September 2017).
22. Gueli, E. TraCI4J. Available online: https://github.com/egueli/TraCI4J (accessed on 3 September 2017).
23. Keras: The Python Deep Learning Library. Available online: https://keras.io/ (accessed on 3 September 017).
24. Tensorflow: An Open Source Machine Learning Framework for Everyone. Available online: https://www.tensorflow.org/ (accessed on 3 September 2017).
25. Wang, S.; Djahel, S.; McManis, J. A Multi-Agent Based Vehicles Re-Routing System for Unexpected Traffic Congestion Avoidance. In Proceedings of the 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), Qingdao, China, 21 July 2014; pp. 2541–2548.
26. de Souza, A.M.; Yokoyama, R.S.; Boukerche, A.; Maia, G.; Cerqueira, E.; Loureiro, A.A.; Villas, L.A. Icarus: Improvement of Traffic Condition through an Alerting and Re-Routing System. *Comput. Netw.* **2016**, *110*, 118–132. [CrossRef]
27. Codeca, L.; Frank, R.; Engel, T. Traffic Routing in Urban Environments: The Impact of Partial Information. In Proceedings of the 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), Qingdao, China, 21 July 2014; pp. 2511–2516.
28. randomTrips. Available online: http://sumo.dlr.de/wiki/Tools/Trip (accessed on 3 September 2017).
29. DUAROUTER. Available online: http://sumo.dlr.de/wiki/DUAROUTER/ (accessed on 3 September 2017).
30. Dijkstra, E.W. A Note on Two Problems in Connexion with Graphs. *Numer. Math.* **1959**, *1*, 269–271. [CrossRef]
31. Rao, A.M.; Rao, K.R. Measuring Urban Traffic Congestion-a Review. *Int. J. Traffic Transp. Eng.* **2012**, *2*, 286–305.

32. Data/Scenarios/TAPASCologne. Available online: http://sumo.dlr.de/wiki/Data/Scenarios/TAPASCologne/ (accessed on 3 September 2017).
33. Haklay, M.; Weber, P. Openstreetmap: User-Generated Street Maps. *IEEE Pervasive Comput.* **2008**, *7*, 12–18. [CrossRef]