Reinforcement Learning-designed LSTM for Trajectory and Traffic Flow Prediction

Bachelorarbeit
der Philosophisch-naturwissenschaftlichen
Fakultät der Universität Bern

Vorgelegt von

Ryan David Aebi
FS 19

Leiter der Arbeit:
Professor Dr. Torsten Braun
Institut für Informatik und angewandte Mathematik

Abstract

With the increasing number of available data of vehicles' trajectories, a wide range of trajectory and traffic flow prediction tasks arise. Both trajectory and traffic density play an essential role in Intelligent Transportation Systems (ITS). ITS resolves around new applications such as intelligent traffic management, intelligent traffic forecasting and also infotainment applications. In order to accurately predict trajectories and traffic densities, deep learning models such as Recurrent Neural Networks (RNN) and especially Long Short Term Memory (LSTM) are excellent alternatives due to their ability to learn spatiotemporal (long-term) dependencies in time-series data. Hand-crafting these neural networks however turns out to be a very time consuming trial-and-error task, due to the broad set of hyper-parameters. In this thesis an automated framework was proposed to predict future trajectories and traffic flows in urban areas without human interventions. Reinforcement Learning (RL) is employed to generate high-performance LSTM predictors; this is referred to as RL-LSTM. In addition to speed up the training process, Transfer Learning (TL) was applied, which reuses pre-existing knowledge instead of training every LSTM predictor anew. Further, a novel deep learning algorithm for traffic flow prediction is proposed, namely HERITOR (*H*igh ord*E*r t*R*aff*I*c convolu*T*i*O*n *R*l-lstm). HERITOR attempts to capture pure spatiotemporal features of urban traffic. The extracted features are then used as input for the RL-LSTM to find a high performance LSTM for traffic flow prediction. For this work, two large-scale datasets were used and consistent improvements of **15% - 25%** over the state-of-the-art could be observed. By using transferrable knowledge, an acceleration of up to **70%** could be observed when searching for an optimal architecture for an LSTM.

Reinforcement Learning-designed LSTM for Trajectory and Traffic Flow Prediction

Mostafa Karimzadeh, Ryan Aebi, Allan M. de Souza, Zhongliang Zhao, Torsten Braun,
Susana Sargento, Leandro Villas
Institute of Computer Science, University of Bern, Switzerland
Institute of Computing, University of Campinas, Brazil
Institute de Telecomunicações - Aveiro, Portugal
Email : {mostafa.karimzadeh, zhongliang.zhao, torsten.braun}@inf.unibe.ch, ryan.aebi@students.unibe.ch,
{allanms, leandro}@lrc.ic.unicamp.br, susana@ua.pt

***Abstract*—Trajectory and traffic flow prediction will play an essential role in Intelligent Transportation Systems (ITS) to enable a whole new set of applications ranging from traffic management to infotainment applications. In this scenario, deep learning approaches such as Recurrent Neural Networks (RNN) and its variant Long Short Term Memory (LSTM) are excellent alternatives due to their ability to learn spatiotemporal dependencies. However, these neural networks tend to be over-complex and hard to design due to the broad set of hyper-parameters. We propose an automated framework to predict future trajectories and traffic flows in urban areas without human interventions. We employ Reinforcement Learning (RL) and Transfer Learning (TL) to generate high-performance LSTM predictors, which is referred as RL-LSTM. In addition, we introduce HERITOR (*H*igh ord*E*r t*R*aff*I*c convolu*T*i*O*n *R*l-lstm), a novel deep learning algorithm for traffic flow prediction. Specifically, HERITOR attempts to capture pure spatiotemporal features of urban traffic. The extracted features are fed into the RL-LSTM to realize a high performance LSTM for traffic flow prediction. We examine the proposed trajectory and traffic flow predictors on two real-world, large-scale datasets and observe consistent improvements of 15% - 25% over the state-of-the-art. By using transferred knowledge, we can accelerate the process of searching an optimal architecture of an LSTM by up to 70%.**

***Index Terms*—Trajectory prediction, Traffic flow prediction, Reinforcement Learning, Transfer Learning, Graph convolution, LSTM.**

I. Introduction

Intelligent Transportation Systems (ITS) will enable more efficient, safer, and greener traffic mobility, which will pave the way to a whole new set of services that will change the way that we live, work, and play [1]. Trajectory and traffic flow prediction will play an important role to improve traffic management decisions, communication protocols, and infotainment applications [1]. A trajectory predictor attempts to estimate the path that a moving object is going to take to travel from one location to another one. The goal of the traffic flow predictor is to estimate the number of moving objects in urban areas given historic mobility trace and the underlying trajectories in a city. Vehicular networking is one of the ITS foundations and also will take advantage of trajectory and traffic flow predictions to improve information retrieval, data dissemination, and resource allocations. For instance, knowing in advance the number of vehicles that will be in a region in the next minutes can not only reduce latency in infotainment applications (e.g., multimedia applications, video streaming, etc), but also provide better resource allocation for multi-access edge computing (MEC) services such as virtual machines, bandwidth and etc.

However, due to the spatiotemporal dependencies of the urban environment and the time-varying traffic patterns, predicting the traffic hotspots (e.g., areas with high traffic, congested areas, etc.), the future trajectory of moving objects, and also the traffic flow estimation between two locations are challenging tasks.

Deep learning-based approaches such as Recurrent Neural Networks (RNN) and its variant Long Short Term Memory (LSTM) have excellent performance in traffic prediction due to their ability to learn temporal dependencies [2]. On the other hand, defining a high-performance architecture for neural networks is still a hard task due to the broad set of hyper-parameters [2], [3]. Typically, hyper-parameters are the variables which determine the structure of neural networks (e.g., number of hidden layers).

In this work, we propose an adaptive framework to predict future trajectories and the urban traffic flow between two locations based on historical data. To optimally capture the spatiotemporal dependencies of an urban environment, we used a graph-structured convolution approach, which can learn the interactions between locations and also support better traffic flow forecasting [2]. On the other hand, to remove the human interventions in the neural network architecture design we used recently proposed model [3], which employs a Reinforcement Learning (RL) based method to find the most suitable architecture for a given dataset. Basically, the RL provides a self-learning approach by rewarding the *high-performance architectures* and punishing the *low-performance ones*. In this way, the goal is to change the architecture of a neural network (e.g., the number of hidden layers, the number of neurons in each hidden layer) based on its previous performance (e.g., the reward) towards a most suitable architecture for a given dataset. Nevertheless, due to the high number of possible neural network architectures, the time to find an architecture with the desired performance may be a concern. Thus, Transfer Learning (TL) is used to reduce this time, the idea is transferring knowledge from a trained predictor to a newly suggested predictor. This will help the new algorithm to pass through training phase faster. Moreover, to identify the spatiotemporal traffic hotspots named as Zone of Staying (ZoS), a tuning-free method is also presented. The main contributions of this work can be stated as follows:

- We study a tuning-free approach for eliciting ZoSs of moving objects from spatiotemporal trajectories without any *a-priori* assumptions.
- We design a trajectory predictor that benefits from Reinforcement Learning and transferable knowledge for searching the best architectural description of an LSTM predictor.
- We present HERITOR, a deep learning technique to analyze graph-structured data, to extract and predict complex spatiotemporal dependencies of traffic flows in road networks of urban areas.
- Quantitative experiments on two real-world, large-scale datasets demonstrate that the proposed trajectory and traffic flow predictors deliver consistently satisfactory results.

The rest of this paper is organized as follows. Section II reviews related work. Section III discusses the problem statement. Section IV introduces the proposed method to discover ZoSs. Section V describes the trajectory predictor using Reinforcement Learning and LSTM, while Section VI describes the urban traffic flow predictor. The evaluation methodology and the performance analysis are presented in Sections VII and VIII, respectively. Finally, Section IX concludes the paper.

II. Related Work

A. Zone of Staying Discovery

Discovering Zones of Staying (ZoS) is an essential component for making cities smart, particularly with regards to traffic management, early congestion warning, mobile network resource allocation, etc. A ZoS refers to a city area, where a moving object (e.g., vehicles, pedestrian) has frequent visits and stays for a considerable amount of time. The initial study [4] uses an iterative clustering approach to extract ZoSs. Clustering is a task in data mining for grouping similar objects into a set known as a cluster. Other clustering-based models, including temporal clustering [5], hybrid clustering [6] [7], k-shape and k-multi-shape clustering [8] have been attempted to discover ZoSs in urban environments. All of the introduced techniques critically rely on some spatial (e.g., acceleration alteration) and temporal (e.g., distance measure) parameters to detect hot-spots. Moving objects are specified by distinct mobility patterns, which leads to having different optimal values of these parameters. Therefore, in this work, we define a novel technique by benefiting from signal processing approaches, which omit the affiliation on the spatial and temporal parameters to detect ZoSs in urban areas.

B. Trajectory Prediction

Future trajectory prediction helps drivers and pedestrians to have safe and efficient navigation through complex traffic scenarios. The pioneering survey on trajectory prediction [9] classified trajectory predictors into two categories: physics-based and maneuver-based models. Physics-based prediction in crowded scenes takes into account the mutual interaction (e.g., traffic rules, road geometry) between surrounding moving objects to estimate short term trajectories [10], [11]. [12] employed Kalman Filter (KF) to estimate future trajectories. The maneuver-based approach could estimate longer trajectories for the moving objects [13]. Heuristic-based classifiers [14], Markov models [15] [16] and random forest classifiers [17] are maneuver- based approaches, which could predict long trajectories in urban areas. Since future trajectories could be estimated through time-series mobility data, a number of Recurrent Neural Networks (RNNs) and their variants including Gated Recurrent Units (GRU) [18] and Long Short Term Memory (LSTM) [19] have been proven to be very effective for trajectory prediction task. [20], [21] use LSTMs to predict motion of human drivers in a grid map. Authors in [22] propose a *social* LSTM, which predicts the trajectory of pedestrians in crowded spaces through the use of a social pooling layer. Despite the success of the introduced trajectory predictors, describing the architecture of neural networks is an effortful task. We automate the process of developing a high-performance LSTM based trajectory predictor without human intervention.

C. Traffic Flow Prediction

Traffic flow prediction is a crucial task in ITS. Investigations on traffic flow prediction most fall into two main categories [2]: statistical methods and machine learning methods. Statistical methods [23], [24] were developed years ago when traffic systems were less complex in the terms of the number of moving objects and the rate of mobility. The ability of such traditional models to accurately estimate future traffic states is quite limited. In recent years, researchers shifted their attention to the deep learning models, which are more robust and accurate in dynamic and complex urban areas. Deep Belief Networks (DBN) [25], Deep Recurrent Neural Networks (DRNN) [26] and Convolutional Neural Networks (CNN) [27] can effectively learn features of time series data and achieve good prediction performance. [28] introduces the first Graph Convolutional Neural network (GCN), which integrates spectral graph theory with deep neural networks. ChebNet was proposed in [29] to improve GCN using fast localized convolutional filters. Authors in [30] developed the idea of the graph Laplacian matrix, which operates on the graph spectrum. [31], [2], [32] propose Diffusion-Convolutional Neural Networks (DCNN) to define convolution as a diffusion process in each vertex of a graph-structured input. The main shortcoming of the mentioned research works is that they do not investigate the effect of neighboring base stations (e.g., RSUs and mobile antennas) and length of trajectories among base stations on the estimated traffic flow. In this work, we introduced an algorithm that employs a high-order convolution operator and adaptive distance adjacency matrix to capture spatiotemporal dependencies of traffic flow in city areas. Besides, we benefit from RL and TL to generate the best possible LSTM architecture to make the traffic flow prediction.

III. Problem Statement

Our work addresses problems of future trajectory prediction and traffic flow prediction in urban areas, which are the core components of Intelligent Transportation Systems (ITS). Hereafter, to address the main problem we split it to the three sub-problems as described below.

Problem 1: Zone of Staying (ZoS) Discovery

We propose a tuning-free technique for detecting hot-spots from spatiotemporal trajectories without any *a-priori* assumption (e.g., constraints on distance, speed of movement, duration of staying, number of collected Global Positioning System (GPS) point, etc.). We eliminate parameter dependence by treating spatiotemporal trajectories as space-time signals and apply signal processing algorithms to discover ZoSs for moving objects (e.g., vehicles and Pedestrians).

• *Requirements and Challenges.*

(i) extracting a moving object's trajectory as an ordered set of visited locations $Tr(lat_n, lon_n, t_n) = [(lat_1, lon_1, t_1), \ldots, (lat_n, lon_n, t_n)]$, where lat_n and lon_n are GPS coordinates and t_n denotes timestamp of the visited location point. *(ii)* transforming extracted trajectory T into a 2D signal $S(t)$. *(iii)* interpreting the space-time signal $S(t)$ in time and frequency domain to detect ZoSs.

Problem 2: Trajectory Prediction with LSTM

Recurrent Neural Networks (e.g., LSTM) are powerful and flexible models that work well for trajectory prediction in city areas. Despite their success, designing an architecture for the LSTM based predictors requires both human expertise and effort. In this research, we study a method to generate a high-performance LSTM for the given learning task automatically.

• *Requirements and Challenges.*

(i) defining a learning agent to suggest the architecture description for the LSTM to have a satisfying accuracy on a validation dataset. *(ii)* transferring the knowledge from a previous model to the new suggested architecture to speed up the experimentation process.

Problem 3: Traffic Flow Prediction

In this research a traffic flow predictor attempts to estimate future traffic states, in terms of the number of moving objects in the trajectories. The trajectories, base stations (e.g., RSUs and mobile antennas), and the collected traffic flow of a city can be represented by a directed graph $G = (V, E, A, W)$, where V is a set of nodes $|V| = N$ (*base stations*), E is a set of ordered pairs of edges (*trajectories*). Directional adjacency matrix presented by $A \in \mathbb{R}^{N \times N}$, in which each element $A_{i,j} = 1$ if there is a path connecting base station i and base station j, otherwise $A_{i,j} = 0$. $W \in \mathbb{R}^{N \times N}$ is a distance adjacency matrix representing base stations mutual influence as a function of their real road distance. The traffic flow collected in a city is shown as $F = \left[f^t \ldots f^{t+T}\right]$, where $f^t \in \mathbb{R}^{N \times P}$ is a set representing number of connected users (P) for each base station (N) at time interval t. The traffic predictor $L(.)$ attempts to learn patterns of traffic flow at time T and make an estimation for the future time T', given a traffic graph G:

$$L\left[\left(f^{(t)}, \cdots, f^{(t+T)}; G = (V, E, A, W)\right)\right] \equiv \left(f^{(t+1+T)}, \cdots, f^{T'}\right) \tag{1}$$

• *Requirements and Challenges.*

(i) modeling spatiotemporal dependencies of traffic flow, indicating explicitly where and when the traffic happens. *(ii)* learning the impact of traffic flows among adjacent trajectories and neighboring base stations.

IV. From Spatiotemporal Trajectories to ZoSs

In this section we address *Problem 1*. The Solution includes translating spatiotemporal GPS trajectories into two-dimensional signals and interpreting generated signal in time and frequency time to detect ZoSs for every single moving object.

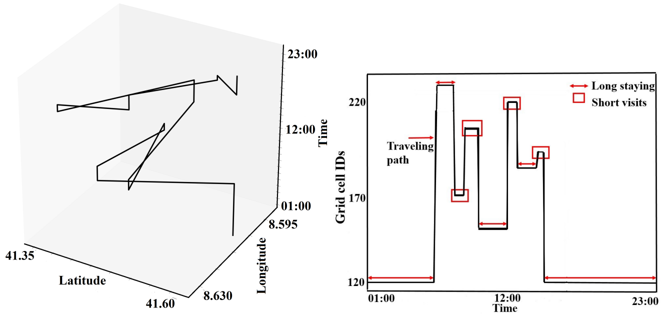
A. Trajectory Extraction

A trajectory is an observed path of a moving object when it travels from one ZoS to another one. Both pedestrians and vehicles can take multiple routes to move among different locations. To discover trajectories, we explore two rich real-world datasets (see Section VII). To extract trajectories for each moving object, we need the list of sequentially connected base stations (e.g., RSUs, antennas) and GPS coordinates of each connected station. We use the Google Maps API Direction Service[1] to discover all possible routes between two consecutive base stations. The discovered trajectory per each individual moving object between ZoS_i and ZoS_j is stored as a 3D signal $T(lat_n, lon_n, t_n) = [(lat_1, lon_1, t_1), \ldots, (lat_n, lon_n, t_n)]$. After discovering the paths the next step is to partition the trajectories into grid cells, for which we use the Python Google S2 Geometry Library [2]. Each grid cell is a four-corner cell, which covers a specific region. Each observed path t_k is partitioned into a sub-list of grid cells. In this work, the coverage area of each grid cell is set to be 300 m^2. The resulting partitioned path can be shown as a 2D signal $r(c_l, t_l) = \{(c_1, t_1), (c_2, t_2), \ldots, (c_l, t_l)\}$, where c_l is the grid cell ID and t_l is the time stamp of visiting the grid cell. Figure 1(a) visualizes a moving object's trajectory as a 3D trajectory and the transformation to a 2D signal is presented in Figure 1(b). In the next subsection, we will analyze the 2D signal in time and frequency domain to extract ZoSs for each single moving object.

1) Time Domain Analysis: Interprets the signal concerning time. As shown in Figure 1(b), a 2D signal in time domain reveals two main features of a moving object's mobility patern: *(i)* staying at locations, *(ii)* traveling along paths. The local maxima/minima of the signal are interpreted as staying locations of a moving object. A set of distinct staying locations can be discovered by selecting the maxima/minima with distinct cell IDs. The staying location areas, where the

[1] https://cloud.google.com/maps-platform/

[2] http://s2geometry.io/



(a) Visualizing one day trajectory as a 3D space-time signal (b) Visualizing one day trajectory as a 2D signal

Fig. 1. Visualizing the OBU's movements as space-time signal.

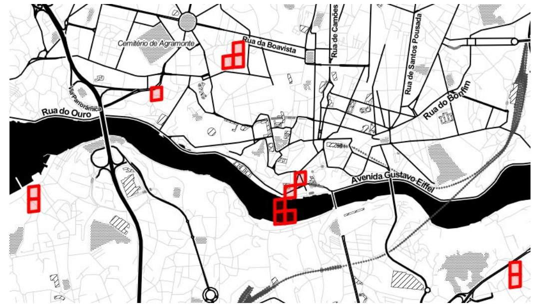


Fig. 2. Discovered ZoSs for OBU ID = 2599

moving objects spend a long duration of time, can directly assume to be their ZoSs (e.g., home, workplace and congested city center, etc.). Moreover, travel paths depict the routes that moving objects take to travel from one staying location to another one. The rest of the detected locations have a shorter staying time duration. Therefore, to be able to transform them as a ZoS, it is necessary to know how often a user visits these staying locations. This information can be obtained by analyzing the signal in the frequency domain, which is the second step of ZoS detection.

2) Frequency Domain Analysis: A periodic signal $S(t)$ is typically represented as $S(t) = S(t + T)$ for all time stamps t, where T is the period of the signal [33]. It represents the smallest duration of time that the signal needs to repeat itself. Analyzing a signal in frequency domain reveals visiting periodicity of each location by a moving object. High periodicities mean that the user visits the location frequently so that the location can be assumed as a ZoS. On the other hand, low periodicity shows that users visit the location infrequently. As explained in Equation 2, applying *Discrete Fourier Transform* (DFT) converts a signal from the space-time domain to a representation in the frequency domain.

$$P[l] = \sum_{l=0}^{L-1} r(c_l, t_l) e^{-jl2\pi/L} \tag{2}$$

$r(c_l, t_l)$ is the 2D trajectory of length L composed of grid cells (c_l). $P[l]$ is the computed visiting periodicity for grid cell c_l. From calculated periodicities for the grid cells, we select the first four dominant periodicities. To interpret the selected visiting periodicities, we applied the *Inverse Discrete Fourier Transform* (IDFT) to convert the signal back from the frequency domain to the time domain (see Equation 3).

$$ZoS(c_l) = (1/N) \sum_{n=0}^{N-1} P\prime[k] e^{jn2\pi/N} \tag{3}$$

$ZoS(c_l)$ denotes detected ZoSs including grid cell c_l and $P\prime[l]$ is the selected dominant periodicity. Figure 2 illustrates an example of discovered ZoSs for a vehicle in the city of Porto, which are represented by a set of rectangular grid cells.

V. Trajectory Predictor

In this section, we address *Problem 2*. We introduce RL-LSTM, a trajectory predictor based on Reinforcement Learning (RL) to automatically realize a high-performing LSTM predictor for a given learning task. Besides, to accelerate the architecture search process, we benefit from Transfer Learning (TL). Using TL the knowledge of the pre-trained architecture (*teacher LSTM*) to estimate the trajectory of a moving object is used as the starting point of the newly suggested architecture (*student LSTM*) for the same task. The leading search method that we use in this work is the Neural Architecture Search (NAS) framework [3]. In NAS, the RL-based controller generates architectures for the predictor. Basically, architecture of a neural network refers to the number of hidden layers, the number of neurons per each layer, and how they are connected. Then, the predictor is trained to make predictions on a validation dataset. The outputs of the algorithm are used to update the controller so that it will generate better architectures over time. Our proposed RL-LSTM has three main units: *(i)* Long Short Term Memory (LSTM) as a student predictor to grow up to get a satisfying accuracy for the prediction task *(ii)* Q-learning as the controller to propose better architectures for the student LSTM to maximize the expected prediction results and *(iii)* the Transfer Learning (TL) unit to accelerate the architecture search process. Details of each unit and how they are integrated to predict future trajectories are explained in the following subsections.

A. Long Short Term Memory (LSTM)

A special kind of Recurrent Neural Network (RNN) that can be applied to time series forecasting is the popular Long Short Term Memory (LSTM) [2]. This architecture represents only the most common implementation of the LSTM as a predictor. To have a highly accurate LSTM predictor the learning agent in Section V-B explores a search space, which includes: (i) *Action space* refers to a set of constraints that restrict the learning agent from taking certain actions. First, we allow the agent to terminate the iterations if the student LSTM can deliver a satisfying prediction accuracy (e.g., 90%). Otherwise, the process will terminate when the learning agent has explored the whole search space. Besides, we force the learning agent to have a dropout layer [34] after each hidden layer. (ii) *Parameter space* is defined as a set of all relevant layer parameters that the learning agent can take. The number of hidden layers is an integer value selected from $(0, 150]$. For each hidden layer, the number of neurons is chosen from $\{1, 5, 10, 20, 40, 60, 80, 100, 150, 200\}$ and the dropout ratio is chosen from $\{0.1, 0.3, 0.5, 0.7, 0.9\}$. Additionally, we use Rectified Linear Unit (ReLU) as non-linearities [35] for each dense layer. Note that defining the *Action* and *Parameter* spaces must have a faster convergence because of limited hardware resources, and it is not a limitation of the method itself.

B. LSTM Architecture Design With Reinforcement Learning

In this subsection, we seek to automate the process of LSTM architecture selection through a searching procedure based on RL. We create a controller using Q-learning, that attempts to define high-performance architectural description of an LSTM that performs well to predict the future trajectory of moving objects without human intervention. As explained in Section V-A, by limiting the layer parameters (e.g., number of hidden layers, number of neurons in each hidden layer and dropout ratio) and actions to choose from, the controller has a finite but large space of possible architectures to search from. The controller as a learning agent trains through random exploration and slowly begins to exploit its finding to select higher-performance architectures employing the ϵ-greedy strategy [36]. The learning agent receives the computed accuracy for the estimated future trajectory as the reward. Based on the reward signal, the learning agent suggests better architectures over time. The whole procedure is shown on the right side of Figure 3. As explained, we benefit from the Q-learning approach as a learning agent to propose architectures for the *student LSTM*. We now summarize the theoretical formulation of Q-learning, as adapted to our problem. We construct an environment where an agent interacts with a discrete and finite *Parameter space* S, which includes a set of all relevant parameters (see Section V-A) that the learning agent is allowed to take. Moreover, we define *Action space* A, which refers to a set of all possible actions that the agent should consider (see Section V-A). At each iteration $t \in \{0, 1, 2, \ldots\}$, the agent in state $s \in S$ will take an action $a \subseteq A(s)$ to pass into next state s'. At each iteration t, computed accuracy is given to the agent as a reward signal ($r_t \in \mathbb{R}$), which depends on the transition from state s to the next state s'. The ultimate objective of the agent is to maximize the total cumulative reward over all possible iterations. Although we limit the agent to a finite search space, there is still a large number of possible architectures, which motivates the use of RL. We define the reward maximization problem recursively in terms of sub-problems as follows; for any state $s \in S$ and action $a \in A\left(s\right)$, we define the maximum total expected reward over all possible iterations to be $Q'\left(s, a\right)$. $Q'(\cdot)$ is named as the action-value function and individual $Q'\left(s, a\right)$ is known as Q-value. The recursive maximization equation, which is known as Bellman's Equation, can be written as:

$$Q'\left(s, a\right) = \mathbb{E}_{s'|s,a}\left(\mathbb{E}_{r|s,a,s'(r|s,a,s') + \gamma \max_{a' \in \mathcal{A}(s')} Q'(s',a')}\right) \quad (4)$$

In our problem, the learning agent does not know *a-priori* what are the effects of each suggested architecture. The agent only knows what the set of possible parameters and actions are. In this case, the agent has to learn through the output of the suggested architectures. Therefore, we can write the Bellman's Equation as an iterative formula (see Equation 5):

$$Q_{t+1}\left(s, a\right) = (1 - \alpha) Q_t\left(s, a\right) + \alpha\left(r_t + \gamma \max_{a' \in \mathcal{R}(s')} Q_t\left(s', a'\right)\right) \quad (5)$$

$\alpha \in (0, 1]$ is the Q-learning rate, which determines the weight given to new information over old information and $\gamma \in (0, 1]$ is the discount factor, which determines the weight given to immediate rewards over future rewards. Q-learning is *off policy* RL, i,e. the learning agent could explore the environment randomly, and despite of this, it can find the optimal architecture for the *student LSTM*. In this research, we use the ϵ-greedy exploration/exploitation strategy. In the exploration phase, the learning agent begins to suggest a new architecture to the *student LSTM* randomly. However, as soon as the agent gets better and suggests high-performance architectures to the LSTM, the agent starts to converge, which refers to the exploitation phase. With ϵ-greedy [36], the learning agent at each iteration suggests a random architecture including a set of possible parameters with probability ϵ, $0 \leq \epsilon \leq 1$. At the beginning of the architecture search, we start with $\epsilon = 1.0$ to ensure that the learning agent has enough time for the exploration phase and we slowly decay ϵ to 0.01 (and not to $\epsilon = 0$) to move toward the exploitation phase.

C. Accelerated Training with Transfer Learning

When training an LSTM, it is not very efficient to train every suggested architecture from scratch. Transfer Learning (TL) offers a solution to this, as it offers possibilities on how to transfer knowledge from a trained predictor (*teacher*), to a new predictor (*student*). Typically, this will help the *student LSTM* to pass through the learning phase faster.

If two LSTMs have a similar architecture (in terms of layers and connectivity) partly, we can call them *semi-homogeneous*. Now given that *teacher* and *student* LSTMs are *semi-homogeneous*, then we can transfer the knowledge from the *teacher* to the *student*. This is achieved by extracting the learned knowledge, which is saved as weights from the *teacher* predictor and initializing the new *student* predictor with those weights. Specifically, we use an adaptation of the Net2Net research [37], where the authors attempt to transfer knowledge from the pre-trained predictor at iteration $t-1$ to the new one at iteration t. In [37] , for a newly added hidden layer, it must have more hidden units than the previous hidden layer, otherwise initializing weights (transferring knowledge) from the previous hidden layer to the new hidden layer will not work.

As a solution for this deficiency, we introduce a new technique to transfer knowledge from the *teacher LSTM* to the *student LSTM*. Let the layers of the pre-trained LSTM be $L = \{l_1, l_2, \ldots, l_n\}$, where the layers l_1 and l_n represent input and output layers, respectively. Suppose that a new layer l'_i is proposed by the learning agent, and then implanted into the *student LSTM* between index $n-1$ and n before the output layer; thus its layers would be $\tilde{L} = \{\tilde{l}_1, \tilde{l}_2, \ldots, \tilde{l}_{n-1}, l'_i, \tilde{l}_n\}$. Then, we define the function $\upsilon(l_j) \in \mathbb{N}{>}\, 0$ to represent the number of hidden neurons for each layer l_j, where $1 \leq j \leq n$.

Further, we define a weight function $\omega(l_j) \in \mathbb{R}^{n\times m}$, where $n, m \in \mathbb{N}> 0$, to form the weight matrix. Now, we can find hidden layers with the same number of units in each layer between L and $\tilde{L}$ by defining $L \bigcap \tilde{L} := \{l_i \mid v(l_i) = v(\tilde{l}_i)$ for $i = 1, \ldots, n-1$. So the first $n-1$ layers of both networks are found to be similar; thus we can transfer knowledge from l_i to $\tilde{l}_i$ for $i = 1, \ldots, n-1$. Further, the transfer learning can then be defined as applying $\omega(\tilde{l}_i) := \omega(l_i), \forall i = 1, \ldots, n-1$. This describes copying the first $n-1$ weights from the teacher to the student. With our proposed approach we can choose the number of hidden units n in l'_i freely, so $v(l'_i) = n$ for any $n \in \mathbb{N}$. Therefore, we could effectively add the layer l'_i to the *student* and then perform transfer knowledge on the layers l_j for $j = 1, \ldots, n-1$.

VI. Urban Traffic Flow Prediction

In this Section, we address *Problem 3*. We introduce HERITOR (***H**igh ord**E**r t**R**aff**I**c convolu**T**i**O**n **R**l-lstm*), a novel deep learning algorithm to estimate future states of urban traffic flow in terms of the number of moving objects in the trajectories. HERITOR employs a high order convolution operator and an adaptive distance adjacency matrix to extract rich spatiotemporal features of urban traffics. Then, the RL-LSTM is fed by the extracted features to generate the best possible LSTM to predict traffic flows. Details of the proposed urban traffic flow predictor are described in the following subsections.

A. High Order Traffic Graph Convolution Operator

The convolution operator at a specific node v_j in graph G can be generally expressed as:

$$Conv(j) = \sum_{i\in N_j} w_{ij} f^i \tag{6}$$

$f^t \in F$ is the extracted feature for node v_j, $w_{i,j}$ is the weight and N_j is the set of nodes that are adjacent to v_j. In this section we introduce the high order convolution operator in the graph. The high order (k-th order) neighborhood can be defined as $N_j = \{v_i \in V | d(v_i, v_j) \le k\}$ for node v_j [38]. The directional adjacency matrix of a graph G represents the 1-hop neighborhood. Then, the k-hop adjacency matrix of n stations can be obtained by calculating the k-th product of $A \in \mathbb{R}^{N\times N}$. Therefore, we can define the k-th order convolution operator for a specific time interval t of the traffic graph as follows:

$$G^{(k,t)}_{Conv} = \left[W_k \odot A'^k\right] f^t \tag{7}$$

Here, $\odot$ refers to element-wise matrix product. A'^k is obtained by adding identity matrix I to the k-hop directional adjacency matrix A^k that creates a self-loop for each node to make them self-accessible in the graph. This means that the moving object can stay connected to the base stations. Otherwise, they are forced to make a transition in each time step. $f^t \in F$ is the feature matrix, to show the number of connected users to each station at a specific time interval t. $W \in \mathbb{R}^{N\times N}$ is a directional distance adjacency matrix computed based on the real distance among stations and threshold Gaussian kernel weighting function [39] in the traffic network. The k-order convolution operator for the time interval t ($G^{(k,t)}_{Conv}$) takes the k-hop directional adjacency matrix, k-hop directional distance adjacency matrix, and the feature matrix as the input. Then, it computes the weighted average of the feature matrix, which has the same size as f^t.

The introduced k-order convolutional operator in Equation 7 takes into account only the number of hops around each base station that fails to capture pure spatiotemporal features of traffic flows. In urban areas, nearby base stations are more related than distant base stations. Following this idea, we aim to give large weights to the short trajectories between nodes. Therefore, we propose the adaptive directional distance adjacency matrix in Equation 8. More specifically, we use the real length of trajectories to assign the weight between two base stations. So, closer nodes will be linked with higher weights.

$$W'_k = \sigma\left[W_k f^t\right] \tag{8}$$

W_k and f^t are k-hop directional distance adjacency matrix and feature matrix, respectively. Sigmoid non-linearity is applied to map elements of the W'_k into a range between $[-1, 1]$. By adding the adaptive directional distance adjacency matrix (Equation 8) into the high order convolution operator, we introduce our convolution operator in Equation 9.

$$G'^{(k,t)}_{Conv} = \left[W'_k \odot A'^k\right] f^t \tag{9}$$

Extracted information about traffic flow in urban road network by the graph convolution within k-hops adjacent nodes with respect to time t are accumulated as follows:

$$G'^{(k,t)}_{Conv-Total} = \left[G'^{(1,t)}_{Conv}, \ldots, G'^{(k,t)}_{Conv}\right] \tag{10}$$

$G'^{(k,t)}_{Conv-Total}$ is a set of k-order traffic graph convolutional feature, that can be fed to the predictor described in the following subsection.

B. High Order Traffic Convolution RL-LSTM (HERITOR)

The proposed traffic flow predictor on a directional graph is a holistic approach that aims to capture patterns of traffic dynamics in urban areas by taking both node features and graph connections into account. The extracted features by incorporating both k-hop convolution operator and adaptive directional distance adjacency matrix are fed into the predictor. We leverage the LSTM predictor to estimate the future spatiotemporal state of urban traffic. In particular, to design the most efficient architecture for the LSTM, similar to Section V-B, we applied Q-learning as a controller to suggest architectures to the *student LSTM*. The model architecture of HERITOR is illustrated in Figure 3. The RL-LSTM as a component of the model is shown on the right side and the high order traffic convolution operator is the left side, where k-hop convolution orders for each time interval t are represented with respect to a red node. HERITOR can learn and predict spatiotemporal dependencies in directional graph-structured data for various forecasting problems (e.g., pedestrians and vehicles)

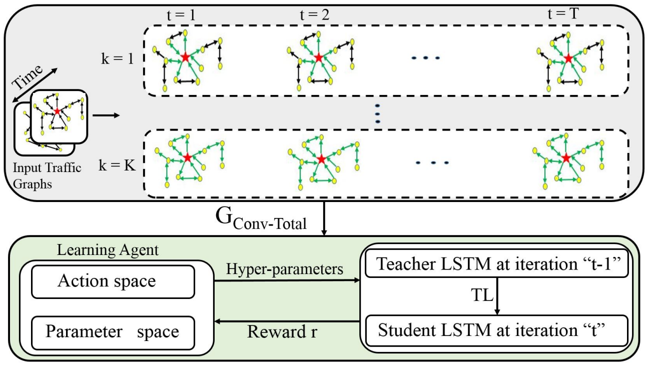


Fig. 3. The system architecture of HERITOR designed for spatiotemporal traffic flow forecasting in urban areas.

VII. Evaluation

In this section, we present an evaluation methodology to validate the proposed moving object trajectory predictor and urban traffic flow estimator.

A. Dataset

We conduct experiments on two rich real-world datasets: (i) *MDC dataset:* This dataset includes reach context information from the smartphones of around $100+$ users connected to 500 mobile antennas around the Lake Geneva region in Switzerland from October 2009 to March 2011 [40]. (ii) *Porto dataset:* This dataset includes real vehicle traces collected from the VANET testbed deployed in the city of Porto in Portugal from October 2016 until August 2017. This urban-scale testbed consists of $100+$ networked vehicles connected to the infrastructure through 120 RSUs [41]. In both of these datasets, we select mobility traces of 100 users, which include connected base station IDs, GPS coordinates of each base station, and the time stamps of the connections. Using this information, the introduced estimators can discover movement patterns of the users and predict the future behaviors of them.

B. Evaluation Metrics

To interpret the prediction success of the proposed RL-LSTM algorithm as a trajectory predictor, we use F_1-Score (11), which is the harmonic mean of precision and recall:

$$\mathrm{F_1\text{-}Score} = \left(\frac{\mathrm{recall}^{-1} + \ \mathrm{precision}\ ^{-1}}{2}\right)^{-1} \tag{11}$$

Here, the precision is the part of the predicted trajectory that genuinely belongs to the observed trajectory. The recall refers to the part of the observed trajectory that is correctly estimated. The performance of the introduced traffic flow predictor is examined by the Mean Absolute Error (MAE) (12):

$$\mathrm{MAE}(f, \hat{f}) = \frac{1}{|N|} \sum_{n \in N} \left| f_n - \hat{f}_n \right| \tag{12}$$

$F = f_1, \cdots, f_n$ represents the number of users connected to each base station, which is extracted directly from the dataset, $\hat{F} = \hat{f}_1, \cdots, \hat{f}_n$ represents the estimated values, and N denotes the number of base stations in each dataset.

C. Experimental Details

During the exploration phase, which refers to searching the highly accurate architecture for the *student LSTM*, we train each algorithm with 70% of the trace data and 30% of the data is used for testing each suggested predictor. After convergence and starting the exploitation phase, the discovered LSTM predictor was trained for 100 epochs. An epoch represents one iteration over the entire dataset. To speed up the training over defined epochs, we use a method called *Early Stopping*. Early Stopping monitors the training progress within epochs by checking the computed accuracy of each training epoch. If the fluctuation of accuracy over the patience epochs (e.g., $P_{epochs} = 10$) is less than $\Delta_{min} = 0.1$, we stop the training. Besides, the batch sizes are set to 200, and the initial learning rate of the LSTM is set to 0.002. We set the Q-learning rate (α) and discount factor (γ) to 0.01 and 1, respectively to prioritize rewards in the distant future. The predictors are trained and evaluated on a High Performance Computing Cluster at the University of Bern in Switzerland (HPC Cluster - UBELIX [3]) with Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz.

VIII. Performance Analysis

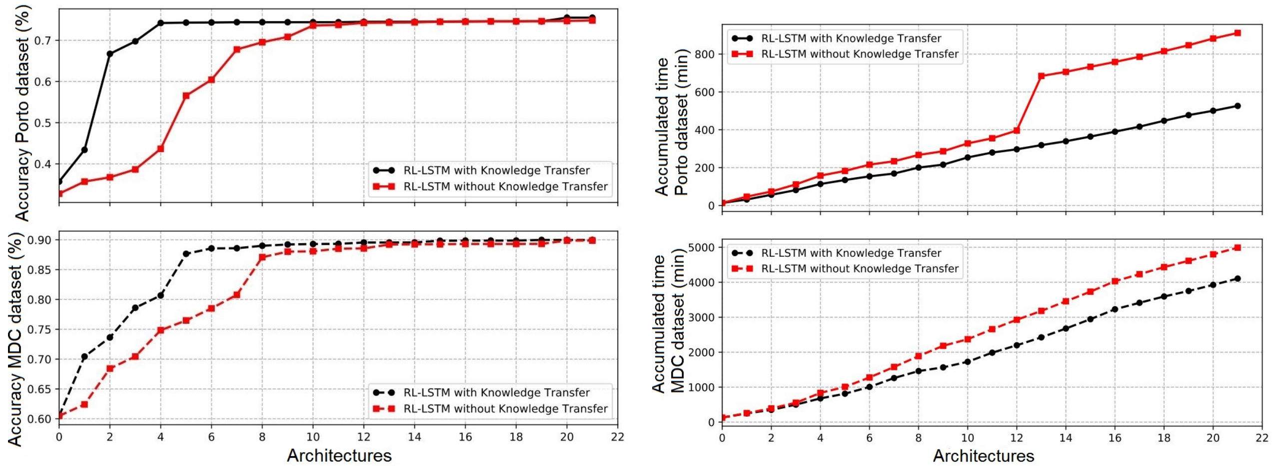
We evaluate the performance of the proposed RL-LSTM and the advantages provided by the TL to predict the trajectory of moving objects and also to predict the traffic flow in urban environments. To evaluate adaptability of the RL-LSTM, we analyze the prediction in the MDC and Porto datasets. The results of the advantages provided by TL is presented in Subsection VIII-A. Subsection VIII-B shows the trajectory prediction results while Subsection VIII-C shows the results for the traffic flow prediction.

A. Transfer Learning Results

Due to the large number of different architectures suggested by the RL-LSTM, the convergence time (e.g., reaching to a highly accurate architecture for the *student LSTM*) can be an issue. Therefore, to have a faster convergence, we employed knowledge transfer between pre-trained LSTM at iteration $t-1$ and newly suggested LSTM at iteration t by the controller. The results of such a reduction for trajectory prediction for both datasets (e.g., MDC and Porto) are shown in Figure 4(a). In particular, Figure 4(a) shows the trajectory accuracy results in function of the architectures suggested by the RL-LSTM, while Figure 4(b) shows the accumulated training time of each architecture comparing the RL-LSTM with knowledge transfer against RL-LSTM without it.

When using the TL the RL-LSTM can reach the desired performance (trajectory prediction accuracy of 75% in the Porto dataset and 90% in MDC) earlier than RL-LSTM without knowledge transfer. Therefore, RL-LSTM with knowledge transfer converges at the 5-th and at the 7-th suggested architecture considering the Porto and MDC datasets, respectively (see Figure 4(a)). On the other hand, the RL-LSTM without

[3] https://docs.id.unibe.ch/ubelix



(a) Convergence time

(b) Exploration time

Fig. 4. Reinforcement Learning-designed LSTM results

knowledge transfer converges at the 11-th and 13-th suggested architecture considering the Porto and MDC datasets, respectively (see Figure 4(a)). In terms of exploration time (e.g., time to find the most suitable architecture), the knowledge transfer provides a reduction of 70% in the accumulated time for both datasets (see Figures 4(b)).

B. Trajectory Prediction Results

In this analysis, we evaluate the performance of the most suitable LSTM architecture found in the previous section to predict the trajectory of moving objects in an urban environment based on the Porto and MDC datasets. We have compared the RL-LSTM with the following literature solutions: Hybrid Markov Chain (HMC) [42], Markov Chain (MC), Random Forest (RF), J48, and the LSTM proposed in [43]. Figure 5 shows the F_1-Score results of each solution. Figure 5(a) shows the average F_1-Score considering business days and weekends, while Figure 5(b) shows the results for all predictions as a Cumulative Distribution Function (CDF).

The results show that J48 and RF present the worst results in both datasets, reaching an average F_1-Score of approximately 0.4 during business days and weekend for both datasets (see Figure 5(a)). Also, for 80% of the predictions, both J48 and RF present an F_1-Score lower than 0.6 in both datasets (see Figure 5(b)). In turn, the performance of MC depends on the quality of the input data. Therefore, MC presents better results in the MDC dataset achieving an average F_1-Score of about 0.6 during business days and weekends (see Figure 5(a)), while in the Porto dataset the MC reaches an average F_1-Score of around 0.4 for both business days and weekends. To improve prediction performance of the MC predictor, the authors in [42] proposed the HMC, which can switch dynamically between the first and second order Markov Chain based on the quality of the input data, consequently improving the performance of the predictor. The HMC was specifically designed for performing prediction tasks (e.g., mobility and trajectory estimation) for the Porto and MDC datasets. The results show that HMC provides an average F_1-Score of approximately 0.6 in both datasets considering business days and weekends (see Figure 5(a)). In addition, for 40% of the predictions, HMC provides a F_1-Score higher than 0.7 in both datasets (see Figure 5(b)).

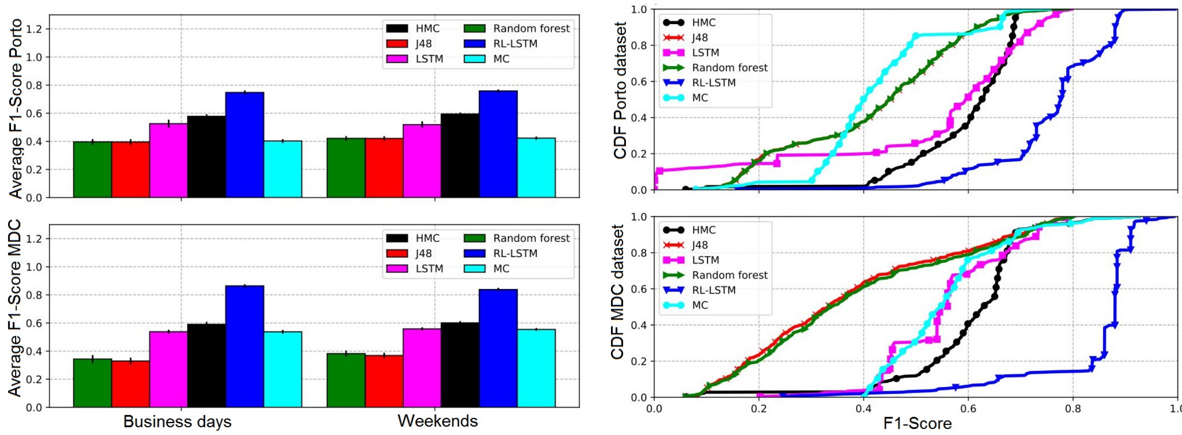
The LSTM predictor presented in [43] does not achieve a satisfying prediction performance in both datasets. This is the result of the hyper-parameters adjustments that need to be tuned specifically for each dataset, but the predictor has not such a capability. In this way, by using the LSTM present in [43], the average F_1-Score decreases by 15% when compared to the HMC in both datasets. Moreover, only 20% of the predictions present a F_1-Score greater than 0.6 in both datasets (see Figure 5(b)). Finally, the efficiency of the architecture suggested by the RL-LSTM can be seen by analyzing the substantial improvements over the HMC results. Therefore, by using the most suitable architecture for each dataset, the RL-LSTM increases the average F_1-Score by 33% in the Porto dataset and by 50% in the MDC dataset compared to HMC. Also, for 80% of the predictions, the F_1-Score is higher than 0.7 in both datasets. Therefore, by employing the RL to realize high-performance architectures for the LSTM, we overcome the performance of all predictors, even the HMC, which is a predictor designed explicitly for the MDC and Porto datasets.

C. Traffic Flow Prediction Results

TABLE I
HIGH ORDER CONVOLUTION RESULTS

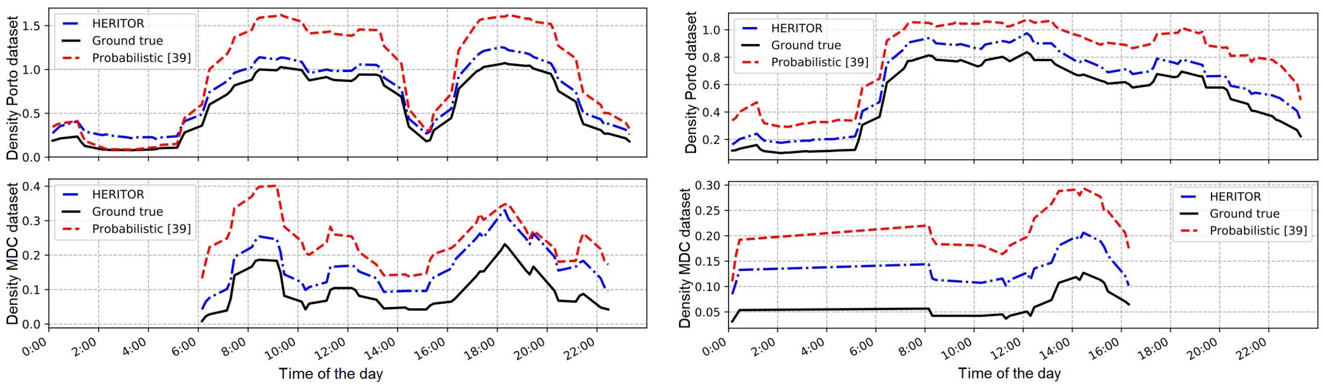
	Convolution orders				
k-order	1-st	2-nd	3-rd	4-th	5-th
MAE-Porto dataset	0.3782	0.3280	*0.2567*	0.3378	0.3976
MAE-MDC dataset	1.7348	*1.1203*	1.4351	1.5644	1.8421

In this subsection, we evaluate the performance of HERITOR to predict traffic flow dynamics in comparison with the novel probabilistic model proposed in [42]. In this way, first, we need to find the most efficient convolution order (e.g., the k-th order) for HERITOR. Table I shows the MAE results for 5 different orders in which $k \in \{1, 2, 3, 4, 5\}$. The convolution order represents the area (e.g., number of hops) around each node that will be taken into account during the



(a) Average F_1-Score (b) F_1-Score distribution

Fig. 5. Spatiotemporal trajectory prediction results.



(a) Business days (b) Weekends

Fig. 6. Traffic flow prediction results.

convolution task. For the given datasets HERITOR reaches its best prediction performance with $k = 3$ and $k = 2$, which represents the optimal spatiotemporal structure based on the road network used. These results are presented in Table I. Therefore, the HERITOR will be fed using the 3-rd convolution order in the Porto dataset and 2-nd in the MDC dataset.

Figure 6 shows the urban traffic flow prediction results as an average density over the city throughout the day comparing HERITOR and the solution proposed in [42]. Figure 6(a) represents the predictions and the ground truth for business days while Figure 6(b) shows the results during weekends. The results show the efficiency of HERITOR, which provides a traffic flow prediction very similar to the real traffic flow during business days and weekends (see Figures 6(a) and 6(b)). This is due to the high order graph convolution operator and adaptive distance adjacency matrix that capture the pure spatiotemporal dependencies and also due to the efficient LSTM designed using RL. Specifically, in the worst case scenario HERITOR introduces an average error of 10% when compared to the real traffic, while the solution proposed in [42] can introduce an error of approximately 50% in the traffic flow prediction when compared to the real traffic flow.

With these results, we can conclude that *(i)* the TL methods employed by the framework proposed in this work can speed up the time to find the most efficient LSTM architecture; *(ii)* the LSTM designed using Reinforcement Learning is highly adaptive and outperforms literature solutions for trajectory and traffic flow prediction; and *(iii)* the graph convolution methods extract optimal spatiotemporal dependencies of a road network and provide better support for traffic flow prediction.

IX. Conclusions

In this work, we employed LSTM to estimate future trajectories and traffic flows of moving objects in urban areas. The main challenges are (i) realizing a high-performance architecture for the LSTM to have satisfying prediction performance for the given task (ii) capturing the rich spatiotemporal dependencies of traffic flows over trajectory networks in urban areas. To address the first challenge, we propose an RL-based method for training a learning agent as a controller within a large search space to automatically generate high-performance LSTMs for the given prediction task. To accelerate this process, we introduced a TL approach to transfer knowledge from a pre-trained LSTM to the new suggested LSTM. Besides, we represent network traffic as graph-structured data, and introduced a high-order convolution operator and directional distance adjacency matrix to learn spatiotemporal dependencies in the traffic network. Experimental evaluations on two real-world, large-scale datasets show that the proposed predictors provides better prediction performance over state-of-the-art works. Using transferred knowledge, we speed up the process of searching an optimal architecture of an LSTM by up to 70%.

References

[1] H. Ye, L. Liang, G. Y. Li, J. Kim, L. Lu, and M. Wu, "Machine learning for vehicular networks: Recent advances and application examples," *IEEE Vehicular Technology Magazine*, vol. 13, no. 2, pp. 94–101, June 2018.

[2] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Graph convolutional recurrent neural network: Data-driven traffic forecasting," *CoRR*, vol. abs/1707.01926, 2017. [Online]. Available: http://arxiv.org/abs/1707.01926

[3] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," *CoRR*, vol. abs/1611.01578, 2016. [Online]. Available: http://arxiv.org/abs/1611.01578

[4] D. Ashbrook and T. Starner, "Using gps to learn significant locations and predict movement across multiple users," *Personal Ubiquitous Comput.*, vol. 7, no. 5, pp. 275–286, Oct. 2003.

[5] B. Chapuis, A. Moro, V. Kulkarni, and B. Garbinato, "Capturing complex behaviour for predicting distant future trajectories," in *Proceedings of the 5th ACM SIGSPATIAL International Workshop on Mobile Geographic Information Systems*, ser. MobiGIS '16. New York, NY, USA: ACM, 2016, pp. 64–73.

[6] M. Karimzadeh, Z. Zhao, F. Gerber, and T. Braun, "Pedestrians complex behavior understanding and prediction with hybrid markov chain," in *2018 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, Oct 2018, pp. 200–207.

[7] ——, "Mobile users location prediction with complex behavior understanding," in *2018 IEEE 43rd Conference on Local Computer Networks (LCN)*, Oct 2018, pp. 323–326.

[8] J. Paparrizos and L. Gravano, "Fast and accurate time-series clustering," *ACM Trans. Database Syst.*, vol. 42, no. 2, pp. 8:1–8:49, Jun. 2017.

[9] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *ROBOMECH Journal*, vol. 1, no. 1, p. 1, Jul 2014.

[10] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1549–15 498, 2018.

[11] J. Schulz, C. Hubmann, J. Löchner, and D. Burschka, "Interaction-aware probabilistic behavior prediction in urban environments," *CoRR*, vol. abs/1804.10467, 2018. [Online]. Available: http://arxiv.org/abs/1804.10467

[12] C. Barrios, Y. Motai, and D. Huston, "Trajectory estimations using smartphones," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 12, pp. 7901–7910, Dec 2015.

[13] G. Xie, H. Gao, L. Qian, B. Huang, K. Li, and J. Wang, "Vehicle trajectory prediction by integrating physics- and maneuver-based approaches using interactive multiple models," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 7, pp. 5999–6008, July 2018.

[14] A. Houenou, P. Bonnifait, V. Cherfaoui, and W. Yao, "Vehicle trajectory prediction based on motion model and maneuver recognition," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov 2013, pp. 4363–4369.

[15] N. Deo, A. Rangesh, and M. M. Trivedi, "How would surround vehicles move? A unified framework for maneuver classification and motion prediction," *CoRR*, vol. abs/1801.06523, 2018. [Online]. Available: http://arxiv.org/abs/1801.06523

[16] M. Karimzadeh, F. Gerber, Z. Zhao, and T. I. Braun, "Pedestrians trajectory prediction in urban environments," in *2019 International Conference on Networked Systems (NetSys) (NetSys'19)*, Garching b. München, Germany, Mar. 2019.

[17] J. Schlechtriemen, F. Wirthmueller, A. Wedel, G. Breuel, and K. Kuhnert, "When will it change the lane? a probabilistic regression approach for rarely occurring events," in *2015 IEEE Intelligent Vehicles Symposium (IV)*, June 2015, pp. 1373–1379.

[18] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *CoRR*, vol. abs/1412.3555, 2014. [Online]. Available: http://arxiv.org/abs/1412.3555

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: http://dx.doi.org/10.1162/neco.1997.9.8.1735

[20] D. J. Phillips, T. A. Wheeler, and M. J. Kochenderfer, "Generalizable intention prediction of human drivers at intersections," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, June 2017, pp. 1665–1670.

[21] B. Kim, C. M. Kang, S. Lee, H. Chae, J. Kim, C. C. Chung, and J. W. Choi, "Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network," *CoRR*, vol. abs/1704.07049, 2017. [Online]. Available: http://arxiv.org/abs/1704.07049

[22] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[23] D. R Drew, "Traffic flow theory and control / donald r. drew," *SERBIULA (sistema Librum 2.0)*, 06 2019.

[24] G. E. P. Box and G. Jenkins, *Time Series Analysis, Forecasting and Control*. San Francisco, CA, USA: Holden-Day, Inc., 1990.

[25] W. Huang, G. Song, H. Hong, and K. Xie, "Deep architecture for traffic flow prediction: Deep belief networks with multitask learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 5, pp. 2191–2201, Oct 2014.

[26] N. Laptev, J. Yosinski, E. L. Li, and S. Smyl, "Time-series extreme event forecasting with neural networks at uber," 2017.

[27] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," *CoRR*, vol. abs/1610.00081, 2016. [Online]. Available: http://arxiv.org/abs/1610.00081

[28] J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, "Spectral networks and locally connected networks on graphs," in *International Conference on Learning Representations (ICLR2014), CBLS, April 2014*, 2014.

[29] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *CoRR*, vol. abs/1609.02907, 2017.

[30] J. Atwood and D. Towsley, "Diffusion-convolutional neural networks," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, Inc., 2016, pp. 1993–2001. [Online]. Available: http://papers.nips.cc/paper/6212-diffusion-convolutional-neural-networks.pdf

[31] ——, "Diffusion-convolutional neural networks," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, ser. NIPS'16. USA: Curran Associates Inc., 2016, pp. 2001–2009. [Online]. Available: http://dl.acm.org/citation.cfm?id=3157096.3157320

[32] D. Andreoletti, S. Troia, F. Musumeci, S. Giordano, G. Maier, and M. Tornatore, "Network traffic prediction based on diffusion convolutional recurrent neural networks," 02 2019.

[33] V. Kulkarni, A. Moro, B. Chapuis, and B. Garbinato, "Capstone: Mobility modeling on smartphones to achieve privacy by design," *CoRR*, vol. abs/1802.07132, 2018. [Online]. Available: http://arxiv.org/abs/1802.07132

[34] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014. [Online]. Available: http://dl.acm.org/citation.cfm?id=2627435.2670313

[35] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10. USA: Omnipress, 2010, pp. 807–814. [Online]. Available: http://dl.acm.org/citation.cfm?id=3104322.3104425

[36] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015. [Online]. Available: http://dx.doi.org/10.1038/nature14236

[37] T. Chen, I. J. Goodfellow, and J. Shlens, "Net2net: Accelerating learning via knowledge transfer," *CoRR*, vol. abs/1511.05641, 2016.

[38] Z. Zhou, "Graph convolutional networks for molecules," *CoRR*, vol. abs/1706.09916, 2017. [Online]. Available: http://arxiv.org/abs/1706.09916

[39] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "Signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular data domains," *CoRR*, vol. abs/1211.0053, 2012. [Online]. Available: http://arxiv.org/abs/1211.0053

[40] J. K. Laurila, D. Gatica-Perez, I. Aad, B. J., O. Bornet, T.-M.-T. Do, O. Dousse, J. Eberle, and M. Miettinen, "The mobile data challenge: Big data for mobile computing research," 2012. [Online]. Available: http://infoscience.epfl.ch/record/192489

[41] C. Ameixieira, A. Cardote, F. Neves, R. Meireles, S. Sargento, L. Coelho, J. Afonso, B. Areias, E. Mota, R. Costa, R. Matos, and

J. Barros, "Harbornet: a real-world testbed for vehicular networks," *IEEE Communications Magazine*, vol. 52, no. 9, pp. 108–114, Sep. 2014.

[42] Z. Zhao, L. Guardalben, M. Karimzadeh, J. Silva, T. Braun, and S. Sargento, "Mobility prediction-assisted over-the-top edge prefetching for hierarchical vanets," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 8, pp. 1786–1801, Aug 2018.

[43] F. Altché and A. de La Fortelle, "An LSTM network for highway trajectory prediction," *CoRR*, vol. abs/1801.07962, 2018. [Online]. Available: http://arxiv.org/abs/1801.07962

Intelligent Safety Message Dissemination with Vehicle Trajectory Density Predictions in VANETs

Mostafa Karimzadeh[a], Allan M. de Souza[a], Ryan Aebi[a], Zhongliang Zhao[a,*], Torsten Braun[a], Leandro Villas[b], Susana Sargento[c], Antonio A. F. Loureiro[d]

[a]*Institute of Computer Science, University of Bern, Switzerland*
[b]*Institute of Computing, University of Campinas, Brazil*
[c]*Instituto de Telecomunicações - Aveiro, Portugal*
[d]*Computer Science Department, Federal University of Minas Gerais, Brazil*

Abstract

Integration of wireless communication systems and machine learning techniques are generating new applications and services in vehicle ad-hoc networks (VANETs). By analyzing data transmission in vehicle-to-vehicle (V2V) communications and vehicle-to-infrastructure (V2I) communications, an intelligent transportation system (ITS) can provide better safety applications. This work explores machine learning approaches to estimate vehicle density on predicted trajectories, which is further utilized to provide intelligent safety message dissemination. With our approach, the traffic safety message, such as accident notifications, will only be disseminated to relevant vehicles that are predicted to pass by the accident areas. Depending on the network connectivity, our system adaptively chooses vehicle-to-vehicle (V2V), vehicle-to-infrastructure (V2I) or hybrid communications to disseminate a message to relevant vehicles. We evaluate the system by using real-world VANET mobility datasets, and experiment results show that our system outperforms other mechanisms without considering predicted vehicle trajectory density information.

Keywords: Vehicle Trajectory Density Prediction, Congestion Prediction, Intelligent Transport System, Safety Data Dissemination.

*Corresponding author
Email address: `zhao@inf.unibe.ch` (Zhongliang Zhao)

1. Introduction

Intelligent transportation systems (ITS) refer to systems that fuse information processing, wireless communication and sensor technologies to vehicles and transportation infrastructure to provide real-time information for road users and transportation system operators to make better decisions. The main goals of ITS are to improve traffic safety and efficiency via intelligent vehicle-to-infrastructure (V2I) or vehicle-to-vehicle (V2V) communications.

The initial motivation behind the development of V2X communications was road safety application, which aims to provide efficient early warning information and assistance to road users in order to prevent road accidents and disseminate accident information to relevant road users. In this perspective, knowing the future trajectories of road users and the vehicle's trajectory density play an important role. The trajectory density is defined as the number of vehicles that may take the same trajectory to travel in the city at a future moment.

Many solutions have been proposed to estimate the vehicle's density [1][2][3], which can be generally divided into two categories: infrastructure-based and infrastructure-less. The infrastructure-based approach requires some additional devices on the road networks, such as roadside magnetic loop detectors, surveillance cameras, wireless vehicle sensors, pressure pads, roadside radar and infrared counters. Instead, the infrastructure-less approach requires only information obtained from the vehicle networks, such as network connectivity and traffic flow information. Compared to infrastructure-based solutions, infrastructure-less solutions are more robust and reliable.

Predicting the density of vehicles is important to support various types of ITS applications (e.g., improve communication capabilities and reduce vehicle congestion), but only a few efforts have been made to study how the estimated vehicles density can be used to enhance wireless communications or improve traffic flow. In addition, few studies have focused on the combination of V2V and V2I communications to enhance system performance or to overcome infrastructure failures without compromising system reliability. In this work, we tackle the two problems of vehicle density prediction and integrating V2V and V2I communications simultaneously, and present a system that is able to predict the vehicle trajectory density, and explore the density prediction information to improve the safety message dissemination mechanism in an efficient way. We design a vehicle trajectory density prediction algorithm using a hybrid Markov chain approach. With this prediction information, road accident information will be only disseminated to vehicles that are predicted to pass by the accident areas. The dissemination mechanism will select direct V2V communication to transmit safety messages, if the network is connected. In case the network is sparse and interested vehicles can not be reached via V2V communications, the system will automatically switch to V2I mode to inform the relevant vehicles to guarantee the timely delivery of the safety messages. The main contributions of this work can be summarized as follows:

- We design a trajectory density prediction algorithm that considers vehicle mobility and trajectory prediction. we propose an inverted index model to estimate the density of vehicles in the routes.
- We present a hybrid early warning system together with a multi-hop data dissemination protocol for VANETs based on vehicle mobility and density prediction to deliver alert messages to relevant vehicles (e.g., vehicles that need to receive such warning) in both sparse and dense scenarios. Our system can adaptively select either the V2V or V2I mechanism to disseminate a warning message based on the network connectivity information to guarantee the timely delivery of the warning message.
- We evaluate our trajectory density predictor, hybrid early warning system, and multi-hop data dissemination protocol using a rich vehicle dataset collected from a real-world VANET testbed. Through comprehensive experiments, we obtain consistently satisfactory results.

This paper is organized as follows. Section 2 discusses existing work on vehicle density prediction in VANETs and its application in VANETs. Section

4 presents the proposed vehicle trajectory density prediction algorithm. Section 5 explains the density prediction-based traffic accident message dissemination mechanism. Section 6 describes the experiments and evaluation methodology applied to validate our solutions. Section 7 analyzes the experimental results. Finally, Section 8 presents the concluding remarks.

2. Related Work

This section presents the related work of vehicles' trajectory and density prediction, and discusses their applications in intelligent transportation systems with their advantages and drawbacks.

2.1. Trajectory and Density Prediction

Accurate density estimation of On-Board Units (OBUs) in urban environments is important for various applications in VANETs: *safety-related* applications are expected to reduce the risk and severity of accidents, and warn a driver whenever a collision at an intersection is probable; *efficiency-related* applications aim at managing the traffic flow on roads and monitoring vehicles' movements; and *comfort-related* applications are aimed to provide entertainments facilities and up-to-date contextual information for passengers by means of Internet access while traveling in urban areas. These applications could be more efficient if they become aware of the density of OBUs at any given time and place [4]. Knowing the accurate density of OBUs in a vehicular communication environment is essential for these applications because a vehicle should be able to adapt its behavior according to vehicle's density anytime and anywhere due to the high mobility and dynamics of VANETs [1]. Therefore, several research efforts have been made to study predictability of vehicle density in trajectories as discussed in the following. The first studies to predict the density of vehicles used Kalman filter to extract traffic density by monitoring camera images [2], or estimating the traffic density by utilizing cumulative acoustic signal collected from a roadside-installed microphone in the road segments [5]. However, applying these approaches are very costly, and in real-world urban areas, only a small fraction of the road segments are covered by sensors. For those road segments without sensors, those methods, may be no longer applicable. In addition, the damage of installed sensor is another shortcoming that should be considered. To deliver better prediction results, some studies explore models that rely on the vehicles' movement patterns. A novel model to forecast traffic flow that depends on acceleration and velocity of vehicles is presented in [6]. Online learning weighted support-vector regression (OLWSVR) [3] integrates a regression model with a weighted learning method. Bastani et al. [7] studied urban traffic estimation by analyzing the number of vehicles located in the transmission range of road-side units (RSUs). These density estimation approaches are not robust enough for accurate vehicle density estimation in urban areas, since they rely on vehicles' movement information (e.g., acceleration variation, velocity or direction of movement), which includes many complicated interactions over the roads and involved crowds. Due to the non-linear and stochastic nature of traffic, some proposals have used deep learning methods [8], such as deep belief network [9] and stacked autoencoder (SAE) [10]. Most of them perform traffic prediction for highways, where traffic flow is relatively stable. In addition to the aforementioned methods, type-2 FL model [11], infinite-mixture model [12] and dynamic traffic assignment [13] were also used to estimate traffic flow in city areas. However, the main drawback of these models is their inability to estimate the specific time of predicted density in trajectories, since these models only work on spatial granularity (*location of congestion*), which means that the algorithm outputs only the number of vehicles at each specific segment of a trajectory without any timing information. To overcome this shortcoming, we propose a model that improves prior methods with the ability to predict the density of vehicles in trajectories by estimating the time of congestion. Moreover, the proposed technique does not demand costly actions as opposed to existing deep learning models.

2.2. Intelligent Traffic Management

Several solutions have been proposed to deal with mobility issues including traffic congestion [14, 15, 16], unexpected traffic incidents [17, 18] and vehicular traffic re-routing [19, 20, 21]. Services such as INRIX [22] provide real-time traffic information, which might support drivers to choose their routes. In turn, Google Maps and Waze are Vehicular Navigation Systems (VNSs) that can recommend faster routes based on a global traffic view whenever a route planning is requested [23, 24]. On the other hand, vehicular re-routing based solutions focus on recommending periodically faster routes to vehicles to improve the overall traffic efficiency.

Those approaches try to improve traffic efficiency by recommending faster routes based on current traffic conditions on the roads, but their performance potentially decrease during unexpected traffic incidents, since they do not implement any pro-active mechanism to deal with such specific issue. Unexpected traffic incidents, such as vehicle crashes, can dramatically decrease the mobility efficiency if not properly handled [25, 24]. In this scenario, vehicles affected by such unexpected incident, e.g., vehicles that will pass by the accident location, need to be notified as soon as possible to take some actions such as change the route or delay their departure to minimize the effects introduced by this unexpected event.

To tackle this issue, most proposals rely either on multi-hop data dissemination approaches [17, 14, 20] or on infrastructure-based approaches [18]. Multi-hop data dissemination approaches consider only vehicle-to-vehicle (V2V) communications, thus assuming a fully connected network to send notification warnings to target vehicles. In infrastructure-based approaches, RSUs are responsible for delivering the notifications to the set of target vehicles, which need to be covered by RSUs. However, for most cities, it is far from reality to consider having a fully connected vehicular network all day long in such a dynamic environment as well as full RSU coverage. Moreover, the delivery of early warnings could be a very difficult task [23]. For instance, in sparse scenarios, data dissemination-based approaches need to replicate warning notifications periodically during the whole duration of an unexpected incident to ensure that they will deliver the warning to all target vehicles. This approach potentially overloads the network with unnecessary transmissions and decreases the overall system effectiveness. In some cases, vehicles might receive the warning too late, not allowing them to take alternative routes to improve their mobility. In principle, infrastructure-based approaches can not work if the unexpected incident occurs in some area not covered by RSUs.

Wang et al. [18] proposed NRR, an adaptive next road rerouting system for unexpected urban traffic congestion avoidance. NRR saves the cost of obtaining a global traffic view by relying on local information available at RSUs (which are assumed to be deployed at each intersection) to select the best next road for each vehicle. The idea is to avoid the road that contains an unexpected incident rather than computing a whole new route. The local information is built based on the vehicles' report. In order to detect unexpected congestion, NRR relies on a central server responsible for sending a notification to the RSU closest to the congestion. Thus, the RSU broadcasts such notification to all vehicles within its coverage, consequently enabling them to verify whether their routes go toward some road that will potentially become congested. The RSU uses the latest obtained traffic information to compute the next road, so that vehicles can avoid the unexpected incident based on their local traffic view. However, having a RSU deployed at each intersection of the city for delivering the service efficiently is not a realistic assumption due to geographical and economic issues.

In our previous work, we proposed ICARUS [17], an intelligent system to improve the traffic condition based on an alerting and rerouting system. ICARUS is aware of both the current traffic condition and unexpected traffic incidents. Therefore, when it detects any traffic congestion or an unexpected traffic incident, it creates a warning message and spreads it through the network (based on a predefined area of interest) to warn vehicles about the incident. ICARUS employs a delay-based data dissemination protocol, which addresses the broadcast storm problem by relying on a broadcast suppression mechanism based on the *sweet spot* concept. In addition to ensure that all vehicles receive the warning, ICARUS periodically rebroadcasts it. Finally, to avoid congestion, whenever a vehicle receives a warning message, it verifies if it will pass through the congested area or the area with the unexpected traffic incident and requests a new route to a central server that possesses global traffic view. Yet, as ICARUS uses only V2V communications to deliver the warning for the vehicles that will pass by the accident, thus delivering an early warning in partitioned scenarios (e.g., scenarios with network partitions) might not be possible, and in some cases it can also decrease the system efficiency due to late delivery.

Considering the aforementioned issues found in the literature such as limited performance when dealing with unexpected traffic incidents [14, 19, 20, 21, 22], the deployment of RSUs at each intersection of the city [18], limited coverage and latency in partitioned scenarios [17], and not considering future trajectories and density to improve their services [14, 17, 18, 19, 20, 21, 22, 23]. We propose an intelligent safety message dissemination mechanism that offloads network traffic whenever it is possible (e.g., multi-hop data dissemination), and delivers early warning messages in partitioned scenarios, through an infrastructure-based data forwarding considering the RSUs deployed at the scenario and the vehicles' trajectory. To do so, we employ an efficient vehicle's trajectory and road density predictor to enable the system to know in advance the network scenario (e.g., connected or partitioned), and the set of vehicles that need to receive the early warning about some unexpected traffic incident to greatly improve network utilization and system performance. Therefore, our system addresses the following challenges: (i) how to predict the vehicles' trajectory and traffic density; (ii) how to decide when to use a multi-hop data dissemination approach or infrastructure-based solution to deliver early warning messages; (iii) how to identify the set of vehicles that need to receive the early warning notification; and (iv) how to perform efficient multi-hop data dissemination.

3. System Overview

In this section we provide an overview of the proposed system, which is designed to deliver reliable traffic warning messages to relevant vehicles using V2V multi-hop data dissemination or infrastructure-based V2I forwarding whenever V2V connections are not available. The system includes a vehicle trajectory and density prediction mechanism, which estimates future locations and densities of vehicles such that only vehicles that are predicted to pass the accident areas will be informed about the en-route unexpected traffic incident.

As shown in Figure 1, the system is based on a hybrid architecture composed of vehicles as OBUs, edge servers as RSUs, and centralized cloud servers. OBUs are responsible for detecting unexpected incidents, notifying near-by edge servers, and disseminating warning messages. RSUs, which are working as edge servers, are responsible for running vehicle trajectory and density predictions using pre-built models, and deciding whether to deliver the warning messages to relevant vehicles using V2V or V2I approaches. The cloud servers are responsible for managing global network information, and making complex data analysis, such as building machine learning models that will be used by edge servers to make real-time trajectory and density predictions.

Whenever an OBU detects an unexpected traffic incident (e.g., such as engine damages, air bags activation, hard breaking), it sends an unexpected traffic incident notification to the nearest RSU either using the cellular network or the IEEE 802.11p protocol. In this way, the trajectory and density prediction functions running in edge or cloud servers will be triggered such that the RSUs can identify vehicles that will pass the detected traffic incident area and should receive the early warning. Additionally, from the trajectory and density prediction the system can know vehicles' future locations and consequently the connectivity among vehicles. This enables the system to transmit the early warning messages to relevant vehicles using either a V2V multi-hop data dissemination approach or a V2I solution when the V2V connectivity is not available.

On the one hand, for the vehicles that can receive the early warning through

V2V communication, the RSU notifies the source vehicle (the one that sent the massage to the RSU) to start the multi-hop data dissemination using an efficient algorithm (described later) that selects the best relay vehicles towards the destination based on density predictions. On the other hand, for those vehicles that should receive the message (they are predicted to pass the accident areas) through the infrastructure, two cases can occur: (i) the target vehicle is within the coverage area of another RSU; and (ii) the target vehicle is not within the coverage area of any RSU. In the first case, the early warning is forwarded to that RSU and delivered to the vehicle. In the latter case, we rely on the mobility prediction to know which are the next RSUs that the target vehicle will pass. These are the RSUs that will receive the early notification to increase the delivery rates and as soon as the target vehicles are connected to them, they will be notified immediately.

Finally, when the early warning is delivered to the target vehicle, it can change its route using a re-routing algorithm to avoid that particular area (e.g., the unexpected incident location).

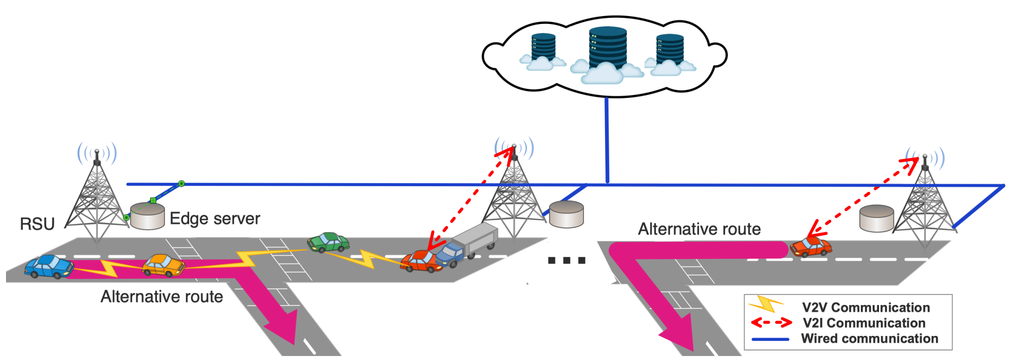


Figure 1: System overview

Therefore, the proposed system can be organized into two main components: (i) vehicle trajectory density prediction; and (ii) intelligent accident message dissemination, as described in the next sections. It is worth noticing that the proposed system is built on top of the dataset collected from real-world VANET testbed deployed in Porto, Portugal [26] [27].

4. Vehicle Trajectory Density Prediction

4.1. Trajectory Extraction

A vehicle's trajectory is defined as the route from one location to another one in an urban area [28]. Vehicles can take multiple routes to move among different locations. To discover the trajectories, we explore a rich vehicle dataset collected from a real-world VANET testbed deployed in the city of Porto, Portugal [26], [29]. The testbed consists of 600 OBUs and 120 RSUs. This dataset includes the actual OBU-RSU connectivity from October 2016 until August 2017, which enables us to discover the vehicles' trajectories. The set of discovered paths of each single vehicle among two $RSU\ ID_i$ and $RSU\ ID_j$ are recorded in $T_{i,j}$ $= [t_1,\ t_2, ...,\ t_n]$. The raw data collected from the VANET testbed includes a large amount of information. However, for OBU's trajectory extraction, we only need the list of sequentially connected RSUs, global positioning system (GPS) coordinates of each connected RSU and the time stamps of the connections. Therefore, we have to generate a file with all the connected RSU IDs as well as timestamp information for every OBU. The data points are received at a sampling rate of 20 to 40 seconds, and the maximum distance between two sequential location points is 15 meters. The first five fields indicate the timestamps of the connections, the sixth field indicates the connected RSU ID, and the last two fields indicate *latitude* and *longitude* of the connected RSU. After generating the trace files for each OBU, the next step is to discover the trajectories. We use the Google Maps API Direction Service[1] to discover all possible routes between two consecutive RSUs. By specifying a starting RSU (*connected RSU at time* t), a destination RSU (*connected RSU at time* $t+2$) and a way-point RSU (*connected RSU at time* $t+1$), the Direction Service can return the route using the way-point. By determining a way-point, the Direction Service can calculate a more accurate path between origin and destination. After discovering the routes, the next step is to partition the geographical space

[1] https://cloud.google.com/maps-platform/

into grid cells, for which we use the Python Google S2 Geometry Library[2]. Each grid cell is a four-corner cell, which covers a specific region. Generated grid cells are accumulated into a set of Cell = $\{c_1,\ c_2, ...,\ c_m\}$. Each observed path t_k is partitioned into a sub-list of l visited grid cells. The resulting partitioned path can be shown as $t_k : \{c_1, c_2, ..., c_l\}$. Given the *GPS* coordinates, the location of any moving object at a specific time can be mapped to a grid cell. In this work, we define a grid cell with of a size of $300\,\mathrm{m}^2$. Each grid cell is identified by a 64-bit Cell ID. As shown in Figure 2, the discovered trajectory for the OBU ID 2007 among two locations (RSU ID 811 and 981) using way-points in the city center of Porto is partitioned by successive grid cells.

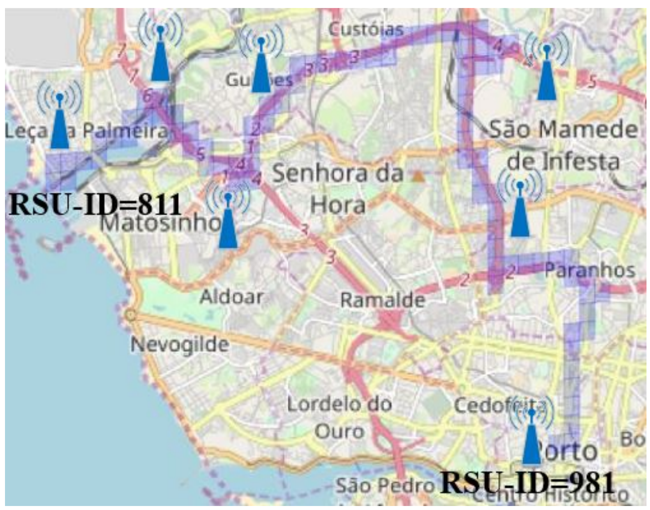


Figure 2: Partitioned trajectory between two locations for OBU-ID = 2007.

4.2. Trajectory Density Prediction Algorithm

Collected mobility traces of vehicles during their movement could be used to explore some regularities in their travels in the city. This knowledge is used by a density predictor to forecast the number of vehicles that may take the same trajectories in the future. The two key aspects for density prediction in trajectories are: (i) predicting the vehicles future locations over time (*connected*

[2] http://s2geometry.io/

RSU); and (ii) estimating the future trajectories between predicted RSUs for multiple vehicles during the same time interval (*e.g., next 1 minute*). The collected mobility trace, which includes connected RSU IDs and timestamps for each vehicle, is used to train our mobility predictor to predict the next connected RSU. The implemented mobility predictor constantly chooses the first order or the second order Markov chain, based on the quality of mobility trace to forecast future locations of the vehicle [30], [31]. The mobility predictor has a tunable time threshold, which enables the algorithm to determine the next connected RSU ID of each vehicle in scales of *minutes* or *hours*. After predicting the next movement of each vehicle, our trajectory predictor attempts to estimate the future trajectory that each vehicle is going to take.

As shown in Section 4.1, each trajectory is partitioned as a series of grid cells between two sequential RSUs. The trajectory predictor is introduced in our previous work [28], in which the sequence of visited grid cells by each moving object is the input of the model. The first order and the second order Markov based algorithm are available predictors. The proposed model can adapt its behavior continuously based on the availability of grid cells and the behavior in the city to maximize the prediction performance. By storing and aggregating the predicted trajectories for multiple vehicles, we can estimate the OBUs that will meet each other in a certain grid cell. To do so, we use an inverted index model to store the predicted trajectories of multiple vehicles during the same time interval. The model includes n rows, where n refers to the number of unique grid cells in all predicted trajectories for a specific time slot. Figure 3 depicts the proposed vehicles' trajectory density predictor, which consists of three main units: hybrid Markov chain as the mobility predictor, adaptive Markov chain as the trajectory predictor, and the model which is based on an inverted index model. The next four steps enable us to estimate the density of vehicles along the trajectories.

(1) Input for our model: mobility trace of each single vehicle with the connected RSU IDs and connection time.

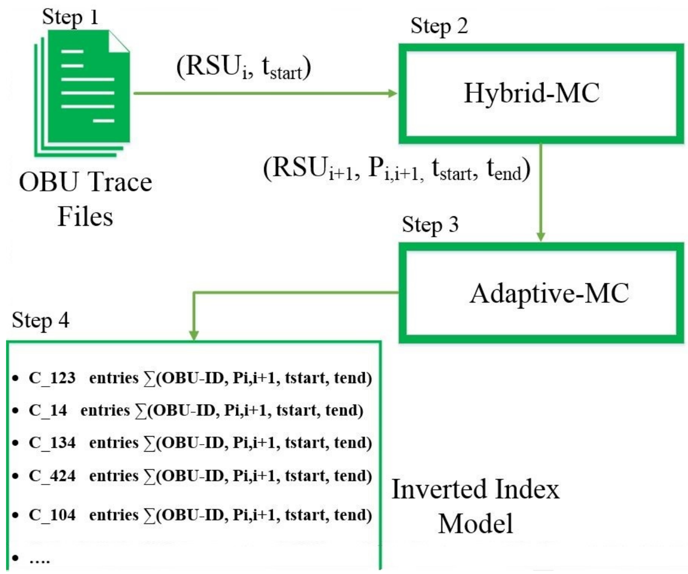


Figure 3: Vehicle trajectory density predictor

(2) Output of our model: after finishing the training step, the hybrid Markov chain generates a 4-item tuple ($RSU\ ID_{i+1}$, $P_{i,i+1}$, t_{start}, t_{end}), where $RSU\ ID_{i+1}$ is the predicted RSU that the OBU is going to connect, $P_{i,i+1}$, represents the transition probability to move from $RSU\ ID_i$ to $RSU\ ID_{i+1}$, t_{start} and t_{end} are the timestamps to indicate when the vehicle was connected to current RSU ($RSU\ ID_i$) and when the vehicle will arrive at the next RSU ($RSU\ ID_{i+1}$), respectively. The connection time to a future RSU is estimated by the hybrid Markov chain.

(3) Using the current $RSU\ ID_i$, which can be easily obtained from the trace for each vehicle and the estimated $RSU\ ID_{i+1}$ from Step 2, the trajectory between these two RSUs can be predicted using adaptive Markov chain. The output of the algorithm is an ordered list of grid cells, which the OBU will pass through when moving from $RSU\ ID_i$ to $RSU\ ID_{i+1}$.

(4) As soon as the predicted trajectory among two RSUs is identified, the inverted index should be updated. To do so, first, each row in the inverted index model is indexed by each grid cell ID of the predicted trajectory. Then, the tuple ($OBU\ ID$, $P_{i,i+1}$, t_{start}, t_{end}) is stored in the correspond-

ing row of the inverted index model. Expired tuples for each vehicle are deleted from the rows. Finally, by summing up the number of tuples stored at each row, we can estimate the number of vehicles that may pass through the respective grid cell from t_{start} to t_{end}.

The output of the proposed trajectory density predictor is stored in $TDp_{i,j}$ = $\{(c_1, P\text{-}OBUs), (c_2, P\text{-}OBUs),...,(c_l, P\text{-}OBUs)\}$. The set includes two item tuples, where the $c_l \in \{$c$_1$, c_2,..., $c_m\}$ demonstrate grid cells on the predicted trajectory that is passing from $RSU\ ID_i$ to $RSU\ ID_{i+1}$, and $P\text{-}OBUs$ store the OBU IDs that we estimated to meet each other on the specific grid cell of the predicted trajectory.

5. Intelligent Safety Message Dissemination

With the trajectory density prediction, the system knows in advance the future position of each OBU and OBU densities. This enables it to identify: (i) communication scenario (e.g., connected or partitioned); (ii) set of vehicles that will pass a traffic incident location whenever this occurs; (iii) most suitable way to send the warning to each vehicle that needs to receive it, considering multi-hop data forwarding and infrastructure-based forwarding; and (iv) set of most suitable relay OBUs to perform multi-hop data forwarding.

Thus, whenever an OBU detects an accident, it notifies it to the closest RSU, such that the RSU can start the procedures to send the early warning to vehicles that need to be informed. Algorithm 1 describes the main procedures of the early warning mechanism. In this scenario, based on the current position of the OBU that has detected the incident, the system can extract its location (e.g., its current cell c_i) (Line 2). Therefore, to identify the communication scenario (e.g., connected or partitioned) the system needs to create an undirected graph $G = (V, E)$ representing the communication network based on the trajectory density prediction TD_p (Line 3), in which the set V represents the OBUs while the set E represents the communication links between them. In other words, considering the position of each OBU, the system can identify the set of OBUs that are connected together and can communicate to each other by means of multi-hop data forwarding. This identification is based on the communication range of each OBU and the distance among them (e.g., the distance between two OBUs is smaller than the their communication range). Therefore, to identify the set of vehicles that need to receive the early warning $OBU_{Targets}$, the system predicts the trajectory $T_{i,j}$ of each OBU and checks if they will pass by the accident location (e.g., $c_{incident} \in T_{i,j}$) (Lines 5-8). With the set of target vehicles ($OBU_{Targets}$) available, the system can determine how to deliver the early warning to each target OBU. In this case, if there is a path in the communication graph from the source OBU to any target OBU, the system will forward the early warning using a V2V multi-hop data dissemination (Lines 11-13). Otherwise, the early warning is sent using the RSUs. Thus, the system needs to predict the set of RSUs that each target vehicle will pass by to forward the early warning to them to increase the delivery probability (Lines 15-17).

Algorithm 2 describes the efficient multi-hop data dissemination based on mobility and density predictions. Considering the origin OBU_{source}, the destination $OBU_{destination}$ and the communication graph G, the system is able to select the best path (e.g., the best set of relay OBUs) to deliver the early warning. To do so, it builds a sub-graph *Links* based on the k-Shortest Paths [32] from OBU_{source} and $OBU_{destination}$ to identify the set of possible paths that the warning message can be sent using multi-hop forwarding (Line 3). In this way, the system selects the next OBU based on the set of neighbors in the *Links* sub-graph and the distance to the destination of each neighbor to determine the set of relay OBUs. Therefore, the next OBU will be the neighboring OBU closest to the destination. The set of relay OBUs is computed iteratively from OBU_{source} towards $OBU_{destination}$ (Lines 6-9). The distance-based approach maximizes the progress of every forwarding operation, and reduces the number of transmissions to deliver the early warning.

Finally, whenever a target vehicle receives an early warning, it can employ a re-routing algorithm to change its route and avoid the incident area to improve its mobility. A new route changing algorithm is out of the scope of this paper.

Algorithm 1: Hybrid early warning system

Input : OBU_{source} ; // OBU-ID that has detected the unexpected incident

TD_p ; // set of grid-cells (c) with predicted density (|P-OBUs|) $\geq$ 1

Output: Warning target vehicles either using $V2V$ or Infrastructure-based forwarding

$c_{incident} \leftarrow getc(OBU_{source})$; // Identify the grid-cell (c) of the OBU_{source}

$G \leftarrow buildNetworkGraph(TD_p)$; // Build V2V communication network graph

$OBU_{Targets} \leftarrow []$; // List that defines the OBUs that need to receive the warning

// Identify which OBUs need to receive the warning

foreach $OBU \in P\text{-}OBUs$ **do**

    $T_{i,j} \leftarrow predictTrajectory(OBU)$

    // Checks if the OBU will pass by the incident location

    if $c_{incident} \in T_{i,j}$ **then**

        $OBU_{Targets}.add(OBU)$;

    end

end

// Decide how to deliver the warning to $OBU_{Targets}$

foreach $OBU_{destination} \in P\text{-}OBUs$ **do**

    if $G.hasPath(OBU_{source}, OBU_{destination})$ **then**

        // Deliver the warning using $V2V$

        $dissemination(OBU_{source}, OBU_{destination}, c_{incident})$

    end

    else

        // Forward the warning to the set of RSUs that cover the $OBU_{destination}$ trajectory

        $RSUs \leftarrow identifyRSUsCoveringTrajectory(T_{i,j})$;

        $forward(c_{incident}, RSUs, OBU_{destination})$;

    end

end

Therefore, we have implemented the efficient route changing mechanism employed by our previous work [17]. This route changing approach considers the ongoing traffic conditions (e.g., which is based on the density of each road) and the location of the unexpected incident to compute an alternative route. In addition, to avoid the problem of creating another congestion spot by computing the same alternative route for many vehicles with the same origin and destination (limitation presented in deterministic re-routing approaches), it uses a probabilistic approach for selecting a route in a set of possible alternative routes to each vehicle.

Algorithm 2: Data dissemination algorithm based on the trajectory and density prediction

Input : OBU_{source} ; `// OBU that will start the dissemination task`
$OBU_{destination}$; `// OBU that needs to receive the warning`
G ; `// Graph representing V2V communication`

Output: The set of relay OBUs selected to perform the multi-hop data forwarding

`// Build K-Path subgraph connecting` OBU_{source} `and` $OBU_{destination}$
$Links \leftarrow G.KShortestPathsSubGraph(OBU_{source}, OBU_{destination})$
$Relay_{OBUs} \leftarrow []$;
; `// List that defines the relay OBUs during the data dissemination task`
$next_{OBU} \leftarrow OBU_{source}$;
`// Define the set of relay OBUs to deliver the warning`
while $next_{OBU} \neq OBU_{destination}$ **do**
    `// Add the next relay OBU`
    $Relay_{OBUs}.add(next_{OBU})$;
    `// Find neighbors of` $next_{OBU}$ `in the K-Path subgraph`
    $neighbors \leftarrow Links.getNeighbors(next_{OBU})$;
    `// Selects the neighboring OBU closest to the` $OBU_{destination}$
    $next_{OBU} \leftarrow getOBUClosest2Destination(neighbors)$;
end

6. Experiment Setup

In this section, we describe the experiment details and evaluation methodology to examine the performance of the proposed trajectory density predictor and its impacts on the safety message dissemination mechanism.

6.1. Dataset

As described in Section 4, the proposed trajectory density predictor for this work integrates both mobility and trajectory predictors to estimate the number of vehicles that may meet each other in grid-cells. In order to train also test our hybrid and adaptive Markov chains, we need mobility trace datasets. In this research, we are using a real-world VANET testbed deployed in the city of Porto, Portugal [26] [27]. This large-scale dataset consists of 600+ networked OBUs using IEEE 802.11p, WiFi or 4G as well as 120+ RSUs scattered along the city. The dataset includes different pieces of information, such as RSUs information (*RSU location, RSU ID, volume of the download/upload traffic via 4G,* *WiFi, DSRC, etc*), and OBU information (*connected RSUs, OBU ID, timestamps, transmitted 4G/DSRC traffic, etc.*). However, for trajectory density prediction, only RSU IDs, GPS coordinates of connected RSUs and timestamps are required. To evaluate the prediction performance of our mobility predictor, we split the trace data as 70% for learning and 30% for testing. To train our adaptive Markov chain to estimate future trajectories for every single OBU, the 10-fold cross validation was performed.

6.2. Evaluation of Vehicle Trajectory Density Prediction

To evaluate the effectiveness of the proposed vehicle density predictor, we use the *accuracy*. This metric is commonly used as an evaluation measure in *information retrieval* [33], which indicates the relationship between the numbers of True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN), which are defined in Equations 1, 2, 3 and 4, respectively.

$$TP = c_i \in TDpi,j \wedge c_i \in TDri,j, \tag{1}$$

$$TN = c_i \notin TDpi,j \wedge c_i \notin TDri,j, \tag{2}$$

$$FP = c_i \in TDpi,j \wedge c_i \notin TDri,j, \tag{3}$$

$$FN = c_i \notin TDpi,j \wedge c_i \in TDri,j, \tag{4}$$

where $TDp_{i,j}$, $TDr_{i,j}$ are the predicted density of vehicles in a trajectory by the density predictor and the real density, which are extracted directly from the dataset, according to the vehicles that take the partitioned trajectory to move from the location of $RSU\text{-}ID_i$ to the location of $RSU\text{-}ID_j$, respectively. We introduce the function *Accuracy(*$TDp_{i,j}$, $TDr_{i,j}$*)* to quantify prediction performance of the proposed algorithm, in 5.

$$Accyracy\left(TDp_{i,j}, TDr_{i,j}\right) = \frac{TP\left(TDp_{i,j}, TDr_{i,j}\right) + TN\left(TDp_{i,j}, TDr_{i,j}\right)}{\left(TP\left(TDp_{i,j}, TDr_{i,j}\right) + FN\left(TDp_{i,j}, TDr_{i,j}\right) + FP\left(TDp_{i,j}, TDr_{i,j}\right) + TN\left(TDp_{i,j}, TDr_{i,j}\right)\right)} \tag{5}$$

6.3. Evaluation of Traffic Accident Message Dissemination

To evaluate the proposed hybrid early warning system, first we need to produce a simulation platform based on the real-world VANET dataset. The simulation platform is composed of: (i) the mobility simulator SUMO [34], which provides the urban mobility and vehicular traffic management using a Traffic Command Interface (TraCI) framework; (ii) the network simulator OMNeT++ [35], which is a discrete event-based simulator that provides a set of tools for developing network applications; and (iii) the vehicular network framework Veins [36], which implements the IEEE 802.11p standard and extends SUMO and OMNeT++ simulators to offer a comprehensive suite of models for developing ITS and VANET application and protocols. To produce the same characteristics of the real-world VANET dataset, we extracted the Porto city map using the OpenStreetMap (OSM) tool and converted it into a SUMO scenario. Moreover, to produce the same traffic presented in the dataset we used the predicted trajectories, i.e., based on the centroid coordinates of each c_i that composes the OBU trajectory we mapped it to the closest road present in the scenario. In this case, we were able to employ the proposed hybrid early warning system and evaluate its performance considering traffic and network based metrics. For the sake of clarity, Figures **??** and 5 show the extracted SUMO scenario using OSM and its traffic density for one week all days long considering the trajectories exported from the dataset.

Table 1: Simulation parameters

Parameters	**Values**
OSM bounding box	41.1790,-8.6912; 41.1390,-8.5765
Channel frequency	5.89e0 Hz mW
Propagation model	Two ray
Transmission power	2.2 mW
Communication range	300 m
Bit rate	18 Mbit/s
PHY model	IEEE 801.11p
MAC model	EDCA

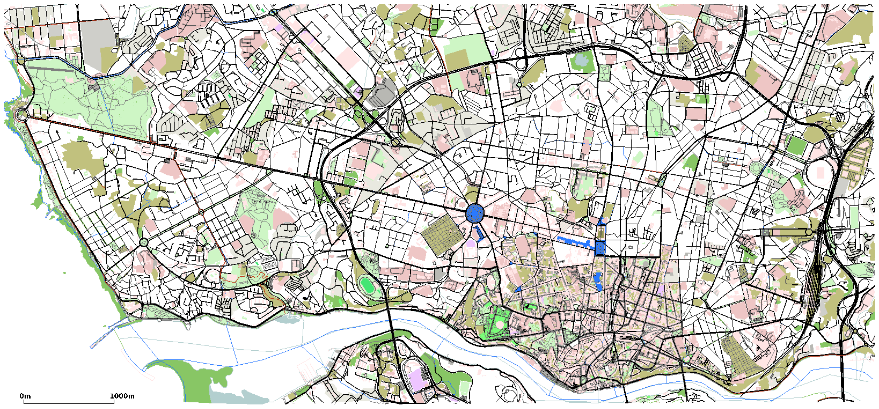
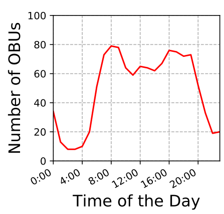
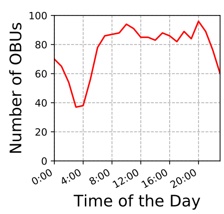


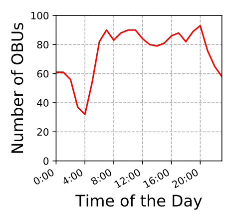
Figure 4: SUMO scenario of Porto map extracted using OSM.



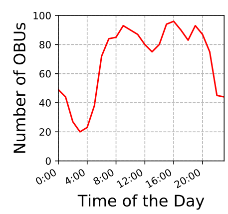
(a) Monday



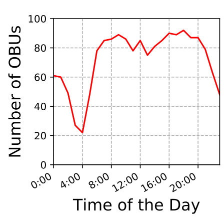
(b) Tuesday



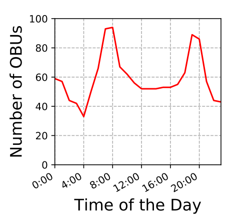
(c) Wednesday



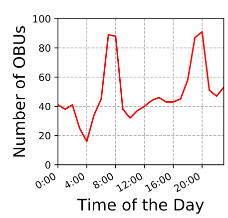
(d) Thursday



(e) Friday



(f) Saturday



(g) Sunday

Figure 5: Scenario density along each day of trajectories extracted from the dataset.

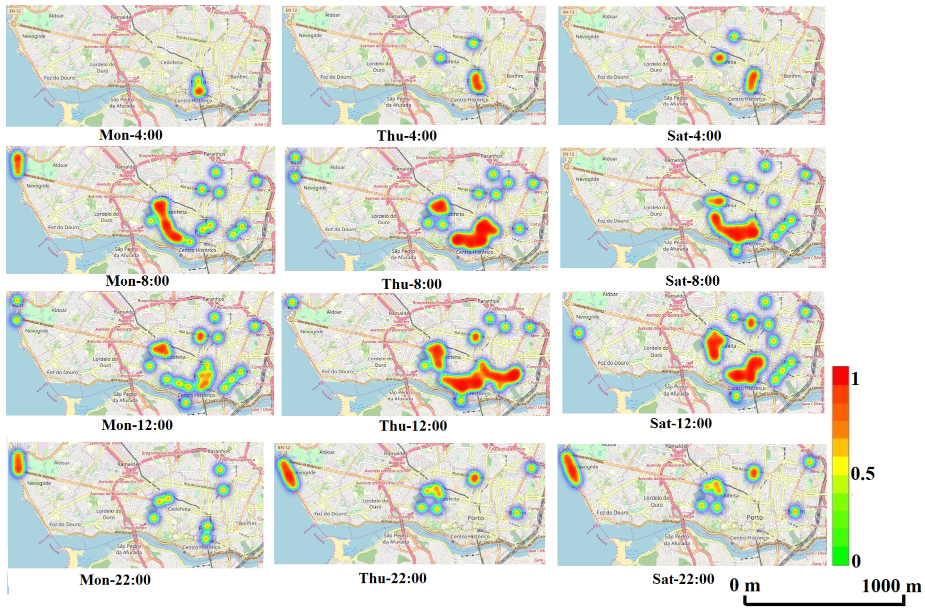


Figure 6: Density prediction in trajectories at different times of days.

With the simulation platform, we analyzed the system performance by randomly introducing a set of 100 unexpected traffic incidents into the scenario considering different days and periods. To produce an unexpected traffic incident, for each day and period choose we randomly selected an OBU from the mobility trace. Thus, we randomly selected and a *grid-cell* (e.g., c_i) from its trajectory in which the OBU will make an unexpected stop, thus the vehicle in the simulation stays stopped in that c_i closing the entire road for one hour, and, after that, the road is released [37]. Therefore, as soon as the unexpected traffic incident occurs, we trigger the proposed hybrid early warning. As simulation parameters, we set the bitrate to 18 Mbit/s at the MAC layer and the transmission power to 2.2 mW, resulting in a communication range of approximately 300 m under a two-ray ground propagation model. Table 1 summarizes the main simulation parameters used in our assessment.

In this dataset, we observed that there are potentially two communication scenarios for all weekdays, which can be broadly organized as *connected* and *partitioned*. Figure 6 shows a *heatmap* representing the OBUs density in the city for different periods of different days. As can be seen, both communication scenarios can occur. For instance, on Saturday at 8:00 we potentially have a connected scenario in the city center, while at 22:00 the OBUs are more spread out characterizing a partitioned scenario. More detailed analysis of the OBU heat map can be found in the experiment evaluation section. Therefore, to analyze the multi-hop data dissemination algorithm and the infrastructure-based forwarding mechanism, we consider the following scenarios:

- **Connected**: the scenario in which it is possible to deliver the early warning messages using only V2V communications,

- **Partitioned**: the scenario in which it is not possible to deliver the early warning message promptly using the V2V communication.

To evaluate the performance considering the network perspective, we measure the following metrics:

- **Delivery ratio**: the percentage of warning messages successfully delivered to the target vehicles. It is desired that an efficient system delivers about 100% of its generated messages for managing the traffic efficiently.

- **Transmitted messages**: the total number of messages transmitted by the system to guarantee its service. A high number of transmitted messages is an indication of redundant and unnecessary transmissions.

- **Latency**: the time spent to deliver the warning message to a target vehicle. High latency potentially degrades the system efficiency when dealing with traffic management.

For the end users who receive the early warning messages, they will adapt their routing decisions. The following metrics were used to analyze the efficiency of the system for improving end users' travel experiences:

- **Travel time**: the total time that each OBU takes to travel a trajectory.

- **Time loss**: the total time spent in some traffic congestion and/or traffic bottleneck.
- **Travelled distance**: the total distance travelled by each OBU considering their trajectory.
- **Fuel consumption**: the total fuel consumed by each OBU to travel its trajectory.
- **CO_2 emissions**: the total CO_2 emitted by each OBU to travel its trajectory.

7. Evaluation Results

In this section, we present and discuss the results of both the trajectory density predictor as well as the hybrid early warning system. To examine our density predictor, we mainly focus on prediction accuracy, whereas to evaluate the performance of the hybrid early warning system, we evaluate its network cost and traffic efficiency.

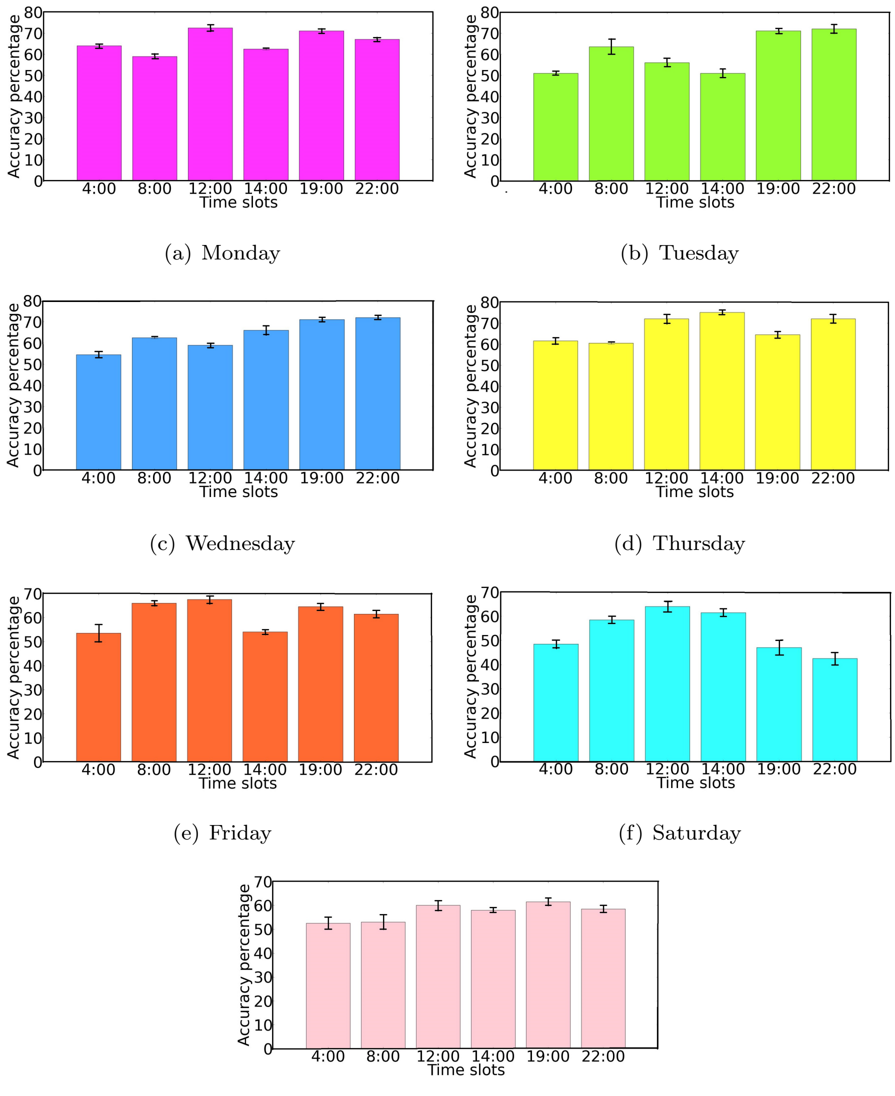
7.1. Trajectory Density Prediction Results

To verify the trajectory density predictor introduced in Section 4.1, we conduct extensive experiments using a real-world VANET dataset. We use time granularity of 1 minute ($\lambda_{density} = 1$) to estimate the density of vehicles in trajectories for different time slots of each day. Figure 7 presents the accuracy of the proposed density predictor, for 100 OBUs in different days. As we can see, for most of the time slots in different days, our trajectory density predictor can deliver a satisfactory prediction accuracy of around 60% to 68% and the highest obtained prediction accuracy is around 75%. To clearly show the prediction performance of the proposed model, we calculate the average density prediction accuracy over different time slots for each single day. As we can see in Figure 8, for weekdays (*Monday to Thursday*), we reach an average prediction accuracy of 60% to 71%. However, for weekends (*Sat and Sun*), the prediction performance of the algorithm is slightly decreased (*52% to 56%*). This situation arises typically because during weekends road users and vehicles do not follow their daily movement patterns. Usually, they are taking different routes to travel among various locations, which shows the difficulty in exploring uncertain mobility patterns [30]. Therefore, our trajectory density predictor cannot make more accurate predictions on weekends.

In addition to obtaining the accuracy of predicted trajectory density for vehicles, we are also interested in identifying the number of OBUs that may take the same route to travel at a specific time slot of a day in the city of Porto. As shown in Figure 5, through experiments that we have conducted in the VANET dataset, we found that during weekdays vehicles are following almost the same traffic pattern. However, during weekends, traffic flows are different. Therefore, to predict the density of all OBUs, we selected randomly *Monday* and *Thursday* as weekdays, and *Saturday* as a weekend, and, then, we estimated the traffic pattern of all vehicles for different time slots per each selected day. As described before, Figure 6 presents the predicted density of vehicles for different time intervals of days. We discover that, in the early morning (4:00 am), most vehicles are traveling in the *south-west* part of the city center. We observed that in the morning (8:00 am) vehicles tend to take most of the trajectories that are passing through the city center. Besides, the predicted density area is more extensive than the area that is detected at 4:00 am. According to our predictions, during mid-day (12:00 pm) most vehicles are still traveling in the routes that are passing through the city center. However, in the evening (10:00 pm), most vehicles' flows are scattered among the touristic center (*west side of the city*) and city center of the Porto.

7.2. Network Performance Results

To evaluate the network performance of the proposed hybrid early warning system we compared its results with Flooding and ICARUS [17]. Figure 9 shows the average results considering all assessed metrics in both scenarios partitioned and connected. As it can be seen, all solutions present a delivery ratio higher



(a) Monday (b) Tuesday

(c) Wednesday (d) Thursday

(e) Friday (f) Saturday

(g) Sunday

Figure 7: Trajectory density prediction accuracy for different time slots.

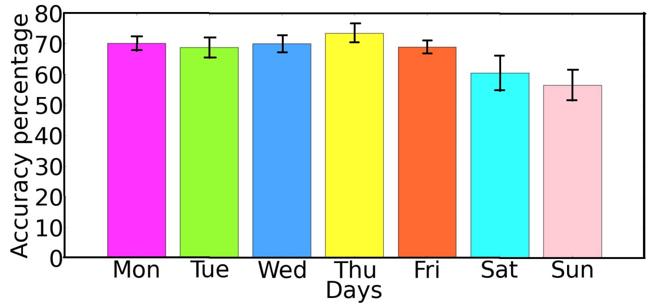
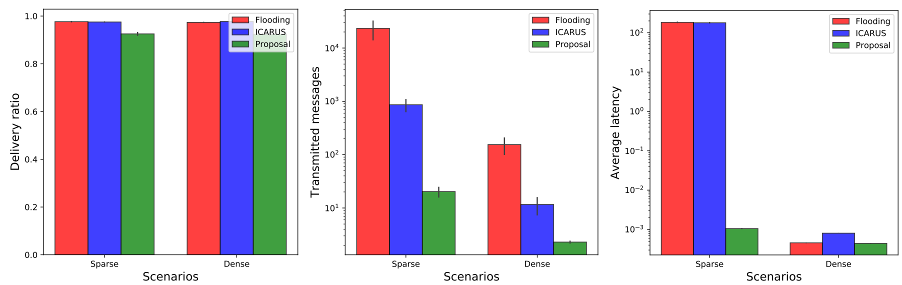


Figure 8: Average trajectory density prediction accuracy per each day.



(a) Delivery ratio (b) Transmitted messages (c) Latency

Figure 9: Network performance results for both sparse and dense scenarios.

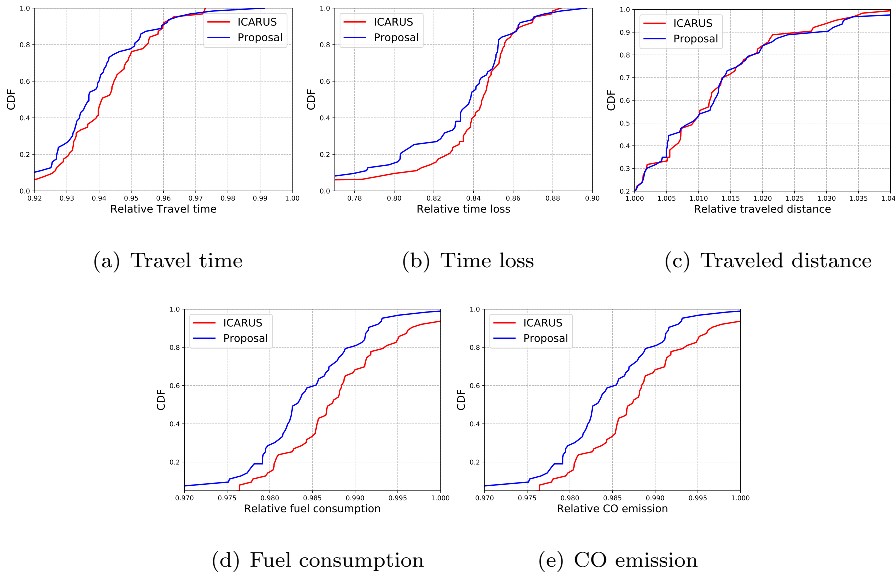
than 90%. However, as our proposal considers the density predictions to perform the data dissemination, in case of mispredictions it potentially cannot deliver the warning accurately, consequently decreasing the delivery ratio in approximately 5% when compared to Flooding and ICARUS in both scenarios (see Figure 9(a)).

Yet, we can see the efficiency of the proposed system when analyzing the transmitted messages (Figure 9(b)) and the latency (Figure 9(c)) for delivering the traffic incident messages. As expected, Flooding has the highest number of transmissions because all OBUs that receive the warning message rebroadcast it, consequently producing a high number of redundant and unnecessary transmissions. Therefore, in partitioned scenarios, in which Flooding needs to replicate the transmission of the incident message periodically to ensure the delivery to all relevant vehicles (e.g., the set of OBUs predicted to pass the incident location), the number of transmissions dramatically increases.

On the other hand, considering latency results, Flooding presents low latency in connected scenarios because it does not introduce any additional delay in the multi-hop data forwarding, thus the vehicles rebroadcast the incident message as soon as they receive it. However, in partitioned scenarios, an undesired latency is introduced, since the multi-hop dissemination is not possible the source OBU (e.g., the OBU involved in the unexpected traffic incident) needs to wait for the relevant vehicles to be within its coverage to forward the warning message. In particular, in connected scenarios, Flooding presents an average latency of 300 milliseconds, while in partitioned scenarios Flooding presents an average latency of approximately 200 seconds.

Unlike Flooding, ICARUS employs an efficient broadcast suppression mechanism to avoid that all OBUs that receive the warning rebroadcast it. Hence, ICARUS reduces the number of transmissions in 70% when compared to Flooding. However, as ICARUS only relies on V2V communications to deliver the warning messages, in partitioned scenarios the source OBU also needs to rebroadcast the warning periodically, consequently increasing the number of transmissions (see Figure 9(b)). Therefore, the latency results for partitioned scenarios ICARUS potentially has the same behavior as Flooding (e.g., it increases the latency because the multi-hop dissemination is not possible), but for connected scenarios, ICARUS presents a higher latency than Flooding because it introduces a small delay in each hop to enable the broadcast suppression (see Figure 9(c)). This delay is based on the distance between the transmitting and receiving OBU, allowing that the OBUs that received the warning message wait for few milliseconds to know whether or not the rebroadcast of the incident message is necessary.

Finally, different from ICARUS and Flooding, our proposal can use the multi-hop and infrastructure-based forwarding. Hence it can decrease the number of transmissions and the latency even in partitioned scenarios. Besides, due to the efficient data dissemination algorithm based on vehicles' trajectory and traffic density predictions, our proposal selects the most efficient set of relay OBUs that will promptly rebroadcast it as soon as they receive the incident mes-



(a) Travel time (b) Time loss (c) Traveled distance

(d) Fuel consumption (e) CO emission

Figure 10: Traffic efficiency results for the assessed metrics comparing the proposed system and ICARUS.

sage, consequently providing a low latency in connected scenarios. On the other hand, in partitioned scenarios, the system forwards the incident message to the next RSU that the relevant vehicle will pass, decreasing the latency. Moreover, the proposed data dissemination algorithm only sends the warning message towards the relevant vehicles direction (since the system knows where each relevant vehicles is located based on predictions), thus reducing the number necessary transmission to deliver the warning accurately. However, Flooding and ICARUS need to disseminate the message in all directions to find the relevant vehicles, increasing the number of transmissions. In particular, our proposal can reduce the number of transmissions and the latency in 95% and 98% when compared to Flooding and ICARUS, respectively in partitioned scenarios.

7.3. Traffic Efficiency Results

To evaluate the traffic efficiency of the proposed system, we have compared its results with ICARUS [17]. Figure 10 shows the results of the assessed metrics as relative values (e.g., the ratio between the mobility with a solution to avoid the unexpected traffic incident and the mobility of real mobility extracted from the data set) in a Cumulative Distribution Function (CDF).

As expected, both ICARUS and the proposed system present better results for all assessed metrics when compared to the real mobility extracted from the data set. This is result of the knowledge about the unexpected traffic incident deliver to the vehicles and also to the re-routing mechanism employed by them to avoid the incident area. Therefore, the vehicles will not get stuck in the congestion produced by the traffic incident, consequently decreasing their travel time as well as time loss (see values smaller than 1 in Figure 10(a) and Figure 10(b)). To avoid the incident location, the vehicles increase their traveled distance by approximately 5% (see Figure 10(c) values greater than 1). However, such increasing in the traveled distance is not an issue, since it not only improves the overall traffic efficiency, but also decreases fuel consumption and CO_2 emissions (see Figures 10(d) and 10(e)).

The slight improvement in the assessed metrics achieved by the proposed system in comparison to ICARUS is because our system can deliver the warning messages earlier than ICARUS even in partitioned scenarios, consequently enabling the relevant vehicles to change their route and improve their mobility with appropriated time. However, due to the limited performance of ICARUS in partitioned scenarios, in some cases, the vehicles can be informed too late about the traffic incident, not allowing them to perform a detour to avoid the traffic incident location.

In this context, due to the lower overhead produced by the proposed system and to its traffic efficiency, we can conclude that knowing in advance the vehicles' trajectory and road density can provide substantial improvements for ITS applications, not only for improving traffic efficiency, but also for improving network performance.

8. Conclusions

This work proposes an intelligent safety message dissemination system by using machine learning-empowered vehicle trajectory density prediction information. With the proposed system, the traffic safety alarming message will only be disseminated to relevant vehicles that are predicted to pass through the accident areas. Depending on the network connectivity, the system adaptively chooses either the V2V or V2I approach to disseminate the message to relevant vehicles to ensure timely message delivery. We evaluate the system by using a real-world VANETs mobility dataset. Simulation results showed that our system outperforms other dissemination mechanisms without considering the predicted vehicle trajectory density information. For future work, we will investigate the following two aspects: (1) model the trajectories of urban as a graph and define a traffic graph convolution operation to capture spatial and temporal features from traffic network; (2) further improvements of the message dissemination mechanism together with the predicted trajectories traffic flow.

References

[1] J. A. Sanguesa, M. Fogue, P. Garrido, F. J. Martinez, J. Cano, C. T. Calafate, P. Manzoni, Real-time density estimation in urban environments by using vehicular communications, in: 2012 IFIP Wireless Days, 2012, pp. 1–6.

[2] M. Balcilar, A. C. Sonmez, Extracting vehicle density from background estimation using kalman filter, in: 2008 23rd International Symposium on Computer and Information Sciences, 2008, pp. 1–5.

[3] Y. Jeong, Y. Byon, M. M. Castro-Neto, S. M. Easa, Supervised weighting-online learning algorithm for short-term traffic flow prediction, IEEE Transactions on Intelligent Transportation Systems 14 (4) (2013) 1700–1707.

[4] K. Tokuda, Applications of wireless communication technologies for intelligent transport systems, Wirel. Pers. Commun. 17 (2-3) (2001) 343–353.

[5] V. Tyagi, S. Kalyanaraman, R. Krishnapuram, Vehicular traffic density state estimation based on cumulative road acoustics, IEEE Transactions on Intelligent Transportation Systems 13 (3) (2012) 1156–1166.

[6] R. Shirani, F. Hendessi, T. A. Gulliver, A store-carry-forward based message dissemination approach with local density estimation in vehicular adhoc networks, in: VTC Fall, IEEE, 2009, p. 0.

[7] S. Bastani, B. Landfeldt, L. Libman, A traffic density model for radio overlapping in urban vehicular ad hoc networks, in: 2011 IEEE 36th Conference on Local Computer Networks, 2011, pp. 85–92.

[8] Y. Bengio, Learning deep architectures for ai, Found. Trends Mach. Learn. 2 (1) (2009) 1–127.

[9] W. Huang, G. Song, H. Hong, K. Xie, Deep architecture for traffic flow prediction: Deep belief networks with multitask learning, IEEE Transactions on Intelligent Transportation Systems 15 (5) (2014) 2191–2201.

[10] Y. Lv, Y. Duan, W. Kang, Z. Li, F. Wang, Traffic flow prediction with big data: A deep learning approach, IEEE Transactions on Intelligent Transportation Systems 16 (2) (2015) 865–873.

[11] L. Li, W. . Lin, H. Liu, Type-2 fuzzy logic approach for short-term traffic forecasting, IEE Proceedings - Intelligent Transport Systems 153 (1) (2006) 33–40.

[12] S. Sun, X. Xu, Variational inference for infinite mixtures of gaussian processes with applications to traffic flow prediction, IEEE Transactions on Intelligent Transportation Systems 12 (2) (2011) 466–475.

[13] B. Ran, Using traffic prediction models for providing predictive traveler information, International Journal of Technology Management - INT J TECHNOL MANAGE 20.

[14] A. M. de Souza, L. A. Villas, A fully-distributed traffic management system to improve the overall traffic efficiency, in: ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM '16), 2016, pp. 19–26.

[15] M. Wang, H. Shan, R. Lu, R. Zhang, X. Shen, F. Bai, Real-Time Path Planning Based on Hybrid-VANET-Enhanced Transportation System, IEEE Transactions on Vehicular Technology 64 (5) (2015) 1664–1678.

[16] Y. E. Hawas, H. El-Sayed, Autonomous real time route guidance in inter-vehicular communication urban networks, Vehicular Communications 2 (1) (2015) 36 – 46.

[17] A. M. de Souza, R. Yokoyama, A. Boukerche, G. Maia, E. Cerqueira, A. A. Loureiro, L. A. Villas, Icarus: Improvement of traffic condition through an alerting and re-routing system, Computer Networks 110 (2016) 118–132.

[18] S. Wang, S. Djahel, Z. Zhang, J. McManis, Next Road Rerouting: A Multiagent System for Mitigating Unexpected Urban Traffic Congestion, IEEE Transactions on Intelligent Transportation Systems 17 (10) (2016) 2888–2899.

[19] A. M. de Souza, R. S. Yokoyama, G. Maia, A. Loureiro, L. Villas, Real-time path planning to prevent traffic jam through an intelligent transportation system, in: IEEE Symposium on Computers and Communication (ISCC '16), 2016, pp. 726–731.

[20] J. . Pan, I. S. Popa, C. Borcea, DIVERT: A Distributed Vehicular Traffic Re-Routing System for Congestion Avoidance, IEEE Transactions on Mobile Computing 16 (1) (2017) 58–72.

[21] R. Doolan, G. M. Muntean, Ecotrec: A novel vanet-based approach to reducing vehicle emissions, IEEE Transactions on Intelligent Transportation Systems PP (99) (2016) 1–13.

[22] http://www.inrix.com.

[23] M. T. Garip, M. E. Gursoy, P. Reiher, M. Gerla, Scalable reactive vehicle-to-vehicle congestion avoidance mechanism, in: IEEE Consumer Communications and Networking Conference (CCNC '15), 2015, pp. 943–948.

[24] S. Djahel, R. Doolan, G.-M. Muntean, J. Murphy, A communications-oriented perspective on traffic management systems for smart cities: Challenges and innovative approaches, IEEE Communications Surveys Tutorials 17 (1) (2015) 125–151.

[25] A. M. de Souza, C. A. R. L. Brennand, R. S. Yokoyama, E. A. Donato, E. R. M. Madeira, L. A. Villas, Traffic management systems: A classification, review, challenges, and future perspectives, International Journal of Distributed Sensor Networks 13 (4) (2017) 1550147716683612.

[26] C. Ameixieira, A. Cardote, F. Neves, R. Meireles, S. Sargento, L. Coelho, J. Afonso, B. Areias, E. Mota, R. Costa, R. Matos, J. Barros, Harbornet: a real-world testbed for vehicular networks, IEEE Communications Magazine 52 (9) (2014) 108–114.

[27] FutureCities Project. Living lab: Vehicular adhoc networking, http://futurecities.up.pt/site/vehicular-ad-hoc-networking-testbed/.

[28] M. Karimzadeh, F. Gerber, Z. Zhao, T. I. Braun, Pedestrians trajectory prediction in urban environments, in: 2019 International Conference on Networked Systems (NetSys) (NetSys'19), Garching b. München, Germany, 2019.

[29] R. Monteiro, L. Guedes, T. Condeixa, F. Neves, S. Sargento, L. Guardalben, P. Steenkiste, Lessons learned from a real vehicular network deployment of delay-tolerant networking, in: 2015 IEEE International Conference on Communication Workshop (ICCW), 2015, pp. 2489–2494.

[30] Z. Zhao, L. Guardalben, M. Karimzadeh, J. Silva, T. Braun, S. Sargento, Mobility prediction-assisted over-the-top edge prefetching for hierarchical

vanets, IEEE Journal on Selected Areas in Communications 36 (8) (2018) 1786–1801.

[31] M. Karimzadeh, Z. Zhao, F. Gerber, T. Braun, Pedestrians complex behavior understanding and prediction with hybrid markov chain, in: 2018 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), 2018, pp. 200–207.

[32] H. C. Joksch, The shortest route problem with constraints, Journal of Mathematical Analysis and Applications (2) 191–197. doi:http://dx.doi.org/10.1016/0022-247X(66)90020-5.

[33] A. Moffat, J. Zobel, Rank-biased precision for measurement of retrieval effectiveness, ACM Trans. Inf. Syst. 27 (1) (2008) 2:1–2:27.

[34] M. Behrisch, L. Bieker, J. Erdmann, D. Krajzewicz, SUMO - Simulation of Urban MObility: An Overview, in: International Conference on Advances in System Simulation (SIMUL '11), 2011, pp. 63–68.

[35] A. Varga, R. Hornig, An overview of the OMNeT++ simulation environment, in: International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops (Simutools '08), 2008, pp. 1–10.

[36] C. Sommer, R. German, F. Dressler, Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis, IEEE Transactions on Mobile Computing 10 (1) (2011) 3–15.

[37] A. M. de Souza, R. S. Yokoyama, G. Maia, A. A. F. Loureiro, L. A. Villas, Minimizing traffic jams in urban centers using vehicular ad hoc networks, in: 2015 7th International Conference on New Technologies, Mobility and Security (NTMS), 2015, pp. 1–5. doi:10.1109/NTMS.2015.7266505.

Paper overview

Submitted:

1. Karimzadeh Motallebiazar, Mostafa; Aebi, Ryan; Mariano de Souza, Allan; Zhao, Zhongliang; Braun, Torsten; Sargento, Susana; Villas, Leandro (2020). *Reinforcement Learning-designed LSTM for Trajectory and Traffic Flow Prediction (Submitted).* In: IEEE International Conference on Computer Communications (INFOCOM 2020)

2. Karimzadeh Motallebiazar, Mostafa; Mariano de Souza, Allan; Aebi, Ryan; Zhao, Zhongliang; Braun, Torsten; Villas, Leandro; Sargento, Susana; Loureiro, Antonio A. F. (28 February 2019). *Intelligent Safety Message Dissemination with Vehicle Trajectory Density Predictions in VANETs (Submitted)* Future generation computer systems: Elsevier