Received 28 January 2021; accepted 26 February 2021. Date of publication 2 March 2021;
date of current version 1 April 2021. The review of this article was coordinated by Associate Editor Tay Wee Peng.

*Digital Object Identifier 10.1109/OJVT.2021.3063125*

# A Comparative Evaluation of Probabilistic and Deep Learning Approaches for Vehicular Trajectory Prediction

**LUIS IRIO** [1] **AND RODOLFO OLIVEIRA** [2,3] **(Senior Member, IEEE)**

[1] Instituto de Telecomunicações, 1049-001 Lisbon, Portugal
[2] Departamento de Engenharia Electrotécnica, Faculdade de Ciências e Tecnologia, FCT, Universidade Nova de Lisboa, 2829-516 Caparica, Portugal
[3] Instituto de Telecomunicações, 1049-001 Lisbon, Portugal

CORRESPONDING AUTHOR: RODOLFO OLIVEIRA (e-mail: rado@fct.unl.pt)

This work was funded by Fundação para a Ciência e Tecnologia, through the projects InfoCent-IoT (PTDC/EEI-TEL/30433/2017), CoSHARE (PTDC/EEI-TEL/30709/2017), and UIDB/50008/2020.

**ABSTRACT** This work compares two innovative methodologies to predict the future locations of moving vehicles when their current and previous locations are known. The two methodologies are based on: (a) a Bayesian network model used to infer the statistics of prior vehicles, trajectory data that is further adopted in the estimation process; (b) a deep learning approach based on recurrent neural networks (RNNs). We present experimental results obtained with both prediction methodologies. The results indicate that the prediction accuracy is improved in both methods as more information about prior vehicle mobility is available. The Bayesian network-based method is advantageous because the statistical inference can be updated in real-time as more trajectory data is known. On the contrary, the RNN-based method requires a time-consuming learning task every time new data is added to the inference dataset. However, the RNN achieves a higher prediction accuracy performance (3% to 5% higher). Additionally, we show that the computational cost to predict the next position a vehicle will move to can be substantially reduced when the Bayesian network is adopted, a scenario where the RNN method requires more computational time. But when the quantity of prior data used in the prediction increases, the computational time required by the RNN-based method can be two orders of magnitude lower, showing that the RNN method is advantageous in both accuracy and computational time. Both methods achieve a next position successful prediction rate higher than 90%, confirming the applicability and validity of the proposed methods.

**INDEX TERMS** Bayesian networks, deep learning, machine learning, performance evaluation, trajectory prediction.

## I. INTRODUCTION

The massive adoption of mobile devices are supporting a plethora of location-based services, which have allowed the generation of a very high volume of spatio-temporal data and motivated the design of many location-based services, ranging from data traffic offloading [1] and data delivery [2], urban planning [3], transportation optimization [4], the recommendation of points of interest [5], and analytical facts about the distribution of the population in a specific area [6]. Several works have studied human mobility [7] and mobility of vehicular networks [8], [9]. Mobility is useful in several applications, such as the prediction of public transportation passengers' flow [10], the increase of vehicular safety protection [11], and to estimate the spread of contagious diseases such as COVID-19 [12], [13]. Most of geolocated datasets available in the community are based on GPS traces sampled over time [14].

The prediction of vehicular trajectories is usually based on prior vehicular data, particularly the latest visited positions, as considered in [1], [15]–[19]. The prediction is useful for several purposes, ranging from the computation of the estimated travel time of a given route, the support for route planning or the computation of vehicular staying time at particular locations [20], [21]. The use of prior vehicular data was also considered in [16] to forecast the most likely potential passenger for taxi drivers.

In the literature we find several probabilistic-based methods to represent and predict vehicular mobility [4], [5], [22]–[25]. Regarding the forecast of mobility aspects, the models can be categorized into two main categories: short-term and long-term trajectory estimation models. The short-term forecasting models are usually based on prior information [26], consisting of one or two previous locations visited by the vehicle plus the current location [27]. Regarding the long-term trajectory models, they predict the trajectory positions at a longer time horizon [28]. The majority of the works published so far are focused on short-term forecasting of vehicular trajectories in urban areas, where the vehicular motion is constrained by traffic lights, complex roads with segments, pedestrians, intersections, and traffic density. Due to the random nature of the vehicles' location over time, long-term prediction models are more difficult to achieve higher prediction accuracy for urban vehicles' mobility [29]. More regular motion patterns lead to more accurate predictions [30], as is the case of deterministic journeys such as the ones traveled by buses. However, in some mobility scenarios regular motion can not be assumed, as is the case for taxis' trajectories, because the passengers' pick up, drop off, and journeys exhibit a higher level of randomness. In these cases, the forecast of future locations is more challenging due to the higher level of uncertainty.

## A. MOTIVATION

This work is motivated by the lack of performance comparative results of different vehicular trajectory prediction techniques. More specifically, we aim at answering the following research questions:

- How Bayesian-based and neural network-based prediction schemes behave in terms of prediction performance and computational cost?
- What is the influence of the quantity of prior data used in the trajectory prediction?
- How different is the prediction accuracy and the computational performance for short-term and long-term estimates?

We start with a detailed description of two different prediction methodologies compared throughout the paper, based on a Bayesian network model and a LSTM neural network. In a first step, we compare the performance of the two proposed methods for short-term predictions, considering the forecast of the next location. Then, we compare both methods for long-term predictions by analyzing the prediction performance of the next five future locations assuming periodic sampling. An additional goal is to decrease the computation time by decreasing the observation state space's size through a preprocessing algorithm that divides the travel region into multiple cells. The cells are used to define the vehicle's trajectory represented by a set of sub-trajectories (sequences) composed of the consecutive traveled cells. Regarding the Bayesian-based method, we adopt a Hidden Markov inference model to capture the taxis' mobility's statistical properties. The prediction approach is based on an improved version of the Viterbi algorithm that computes the most likely sequence of future locations given a sequence of prior locations. Regarding the neural network-based approach, we propose a LSTM neural network where the initial locations of the sub-trajectories are used in the network learning process to estimate the latest positions of the sequence.

## B. NOVELTY AND CONTRIBUTIONS

The main novelty of this work is the comparison of two techniques for vehicle mobility prediction, so we can answer the research questions that have motivated the paper. The comparative analysis is carried out for short-term and long-term predictions and characterizes the influence of the number of prior locations in the prediction performance (longer/shorter length sequences). The contributions of this work are listed as follows:

- The trajectories represented by sampled GPS coordinates are converted into geographic cells so that the spatial data can be downsampled for increased performance. For a fairer comparison, the same downsampled data is adopted in the performance evaluation of both techniques;
- The first technique relies on an innovative Bayesian network represented by a Hidden Markov model (HMM), where the hidden states represent a single sequence of locations, thus embodying the Markov relation between prior visited locations to capture the sequential relation of each trajectory. The prediction relies on an improved version of the Viterbi algorithm, OPTVIT, that achieves a lower computational time while maintaining the optimal prediction performance;
- The second technique compared in this work is based on recurrent neural networks, more precisely a LSTM network, to attenuate the gradient problem that can have a significant impact on long-term prediction of sequential data;
- The prediction techniques are assessed using a dataset of real traces, comparing the OPTVIT algorithm and the LSTM approach for a variable quantity of prior data used in the prediction, as well as for short and long-term predictions. The computation times and prediction performance are reported based on the experimental results;
- A final contribution has to do with the experimental results. They show that: (i) a higher quantity of prior data always improves the prediction accuracy of both techniques; (ii) the Bayesian approach can achieve a lower computational cost for short-term prediction and a similar prediction performance; (iii) For the long-term horizon the prediction accuracy decreases linearly with time; (iv) for long-term predictions, the proposed neural-network is effectively a better solution, as it jointly achieves higher prediction accuracy and lower computational time.

### C. PAPER STRUCTURE

Regarding the paper's organization, Section II presents an overview of selected works in the field. Section III introduces the definitions adopted in the prediction techniques, states the problem to solve, and describes the adopted spatio-temporal model and its analytics. Section IV presents the Bayesian-based technique to model and predict the locations of the trajectory. Section V presents the deep learning approach and the details of the LSTM neural network. Section VI describes the comparison of the performance achieved by the different techniques, and Section VI concludes the paper.

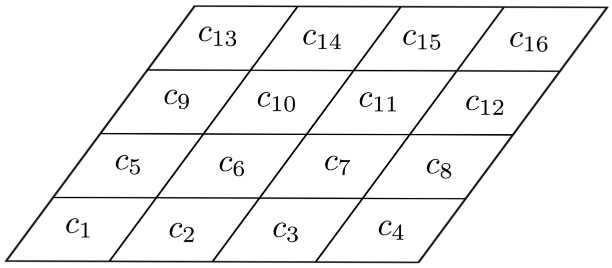

**FIGURE 1.** **Grid map representation with $N = 16$ cells.**

## II. RELATED WORK

Several works have been proposed to address the problem of mobility prediction, and particularly vehicular mobility estimation [31]. Multiple methodologies were proposed so far, including but not limited to fundamental concepts of Information Theory [20], [32] and traditional Markovian-based predictors [15]. The categorization of the existing mobility prediction approaches was addressed in [33]. The majority of the works proposing mobility prediction schemes can be divided into Bayesian Network-Based Methods, Neural Network-Based Methods, and Markov-Based Methods.

The Markov-Based methods are based on the Markov property, which states that the probability of traveling to a future position depends only on the current one. The majority of vehicular prediction schemes are based on Markov-based methods [2], [4], [5], [15], [22]–[24]. The work in [4] adopts the Markov property in a hidden Markov model to characterize the prediction performance of a fixed number of trajectory segments. More recently, the problem of sparse trajectory data was addressed in [24], which has proposed a prediction scheme that makes use of group mobility statistics to increase the prediction accuracy. The sequences of the different trajectory locations are used in [24] to run spatial clustering algorithms capable of classifying them in different groups, which are employed in a variable-order Markov model that estimates the trajectory. However, the use of group trajectory data can lead to bad results because it effectively relies on crowd behavior only. The crowd mobility was also used in [23] to improve individual trajectory estimation by dividing the spatial region into a set of points of interest. Other works have considered Markov-based models with data enrichment, as in [5] where the travelers living habits are taken into account in the prediction process.

Contrarily to the Markov-based methods, the Bayesian Network-Based methods are not only based on the current position but also on the sequence of positions that have preceded it. Bayesian inference is employed to characterize the likelihood of a given vehicle trajectory given prior observed location or locations, thus using the statistics of the historical trajectory data. The adoption of a Bayesian inference model is described in [34] for location prediction, which takes into account multiple predictive factors to enhance the prediction performance, such as road topology information and motion information. The work in [35] identifies the patterns containing regions frequently visited to build a Bayesian network to estimate future locations. Differently, [36] proposed a dynamic Bayesian scheme that represents the consecutive occurrence of observable random variables to estimate the future user's location. Although Bayesian network-based methods are easily implemented, the sparsity of trajectories represents an additional challenge that usually results in high computational costs. Consequently, the methods based on Bayesian inference are most of the time improved with the adoption of other techniques for enhanced performance [27].

Deep learning has also been used to predict vehicular trajectories. Although neural networks need a long time to learn from data, the inference model to predict the mobility benefits from its low computational complexity and is run in deterministic time. Neural networks are gaining popularity in mobility prediction when mobility data is available in a centralized way [17]. Neural networks were adopted in [37] to predict vehicular trajectories using a Multi-Layer Perceptron. The work in [38] tackled the estimation of taxi trajectory destinations based on convolutional neural networks. Innovative neural network models have been developed in the last years and have already been used for mobility prediction, including long short-term memory (LSTM) recurrent networks and generative adversarial networks (GAN). LSTMs were proposed in [39] to predict the vehicular trajectories in highways. GANs were used in [19] to predict vehicular trajectory position and speed in urban scenarios. The adoption of different neural network models combined in a single architecture has also been used to address mobility prediction, as in [16], where a LSTM model is combined with a convolutional one to forecast the most likely potential passenger for taxi drivers.

## III. PROBLEM DEFINITION AND DATA MODEL

### A. PROBLEM DEFINITION

The work considers multiple vehicles moving on a spatial region, delimited by a grid map. Fig. 1 illustrates a grid map with 16 cells. The grid map is divided into two-dimensional geographical sub-regions designated as cells, denoted by $c_\eta$. The location of a vehicle is sampled periodically and linked to a cell. Spatio-temporal trajectories are generated when the vehicles move from the starting point to the endpoint of a journey. Considering that each journey has a variable duration, the trajectories are represented by fixed-length sequences of cells

**TABLE I.** Table of Notations

| Symbol | Description |
|---|---|
| N | Number of different cells |
| $c_\eta$ | Cell $\eta$, $\eta \in \{1, ..., \mathrm{N}\}$ |
| $\Xi_j$ | Number trajectory cells |
| $S_\kappa$ | Sequence $\kappa$, $\kappa \in \{1, ..., \Omega\}$ |
| $\Omega$ | Number of trajectory sequences |
| $\Lambda$ | Number of sequence cells |
| $\beta$ | Length of the prediction (in number of cells) |
| $\Phi$ | Set of all sequences |
| $\Psi$ | Number of unique sequences in $\Phi$ |
| $\lambda = \{\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{B}\}$ | Hidden Markov model |
| $\boldsymbol{\pi}$ | Hidden Markov initial state distribution |
| $\boldsymbol{A}$ | Hidden Markov Transition matrix |
| $\boldsymbol{B}$ | Hidden Markov Emission matrix |

to guarantee a coherent granular temporal basis. The notation and a few definitions adopted in this work are next introduced to provide practical insights into the proposed approach.

*Definition 1:* A **cell** represented by $c_\eta$, with $\eta \in \{1, \ldots, N\}$, indicates each two-dimensional region of the grid map representing the area where a vehicle travels. The maximum number of cells is denoted by $N$.

*Definition 2:* The set of multiple cells, $T_j = \{c_\eta^1, c_\eta^2, \ldots, c_\eta^{\Xi_j}\}$, represents a **trajectory**. The trajectory is an ordered set of $\Xi$ cells where the vehicle travels through. In the trajectory, $c_\eta^k$ denotes its $k$-th cell. Finally, we highlight that a trajectory is formed by a non-constant number of $\Xi_j$ cells, $\Xi_j > 1$.

While a trajectory represents the total number of locations traveled by a vehicle, in this work we introduce a subset of trajectory locations denoted as a sequence. The sequences are the subsets adopted in the prediction algorithms by partitioning the trajectory in multiple sequences.

*Definition 3:* The ordered set $S_\kappa = \{c_\eta^1, c_\eta^2, \ldots, c_\eta^\Lambda\}$ represents a **sequence** formed by a set of $\Lambda$ visited cells, $\Lambda \leq \Xi_j$. The number of cells ($\Lambda$) integrating the sequence $\kappa \in \{1, \ldots, \Omega\}$ remains constant for all $\Omega$ sequences that constitute a trajectory.

*Definition 4:* The **set of sequences** $\Phi = \{S_1, S_2, \ldots, S_\Omega\}$ is formed by all $\Omega$ sequences that result from the preprocessing of vehicles' trajectories. The set $\Phi$ will have a significant number of identical sequences. The symbol $\Psi$ represents the number of unique sequences found in $\Phi$.

*Definition 5:* The **prediction problem** uses the knowledge of the $\Lambda - \beta$ cells observed so far (cells of a given sequence $S_\kappa$) to predict the next $\beta$ cells of the sequence.

The symbols previously defined and adopted in this work are represented in Table I.

## B. MOBILITY DATASET PREPROCESSING

This subsection describes the method to convert the GPS raw data representing a trajectory into sequences. The sequences are characterized in Section III-C.

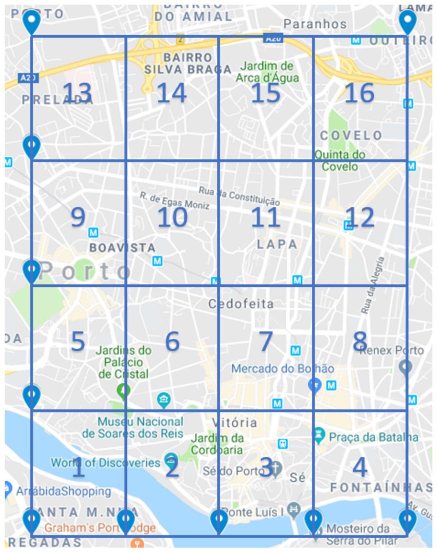

**FIGURE 2.** Region of Porto city, Portugal, considered in this work.

We use a dataset containing the real traces of trajectories completed by 442 taxis servicing in Porto, Portugal, from the beginning of July 2013 to the end of June 2014. The dataset is available at the UCI Machine Learning Repository and reported in [40]. Each entry of the dataset corresponds to a taxi's completed trip, describing are the trip's identifier, the timestamp, and a polyline with the GPS coordinates gathered during the taxi's travel. The polyline is formed by a sequence of GPS coordinates that are sampled every 15 seconds.

Fig. 2 illustrates the Porto city region considered in the paper. The lower-left corner of the map is represented by the GPS coordinates [41.1 391 696, −8.6 341 313]. The grip map represented by the blue lines positioned over the region has a height of 2953 m and a width of 3921 m. Additionally, each of the $N$=16 cells has area 0.724 km$^2$.

The offline preprocessing of raw trajectory data plays an essential role in prediction performance. The data preprocessing algorithm that transforms the data is next introduced. Its primary purpose is to define the set of sequences ($\Phi$).

We consider that the trajectories are no longer defined by a set of GPS coordinates but as a set of cells, simplifying the process of describing trajectories. Since each cell contains several GPS locations this assumption can be seen as a down-sampling of the spatial positions. Each pair of GPS coordinates is mapped into a cell of the map, $c_\eta$, i.e., $1 \leq \eta \leq 16$ cells depicted in Fig. 2. Each trajectory $T_j = \{c_\eta^1, c_\eta^2, \ldots, c_\eta^{\Xi_j}\}$ has a variable number of locations represented by $\Xi_j$ cells. Each trajectory is divided in one or more sequences. A sequence $S_\kappa = \{c_\eta^1, c_\eta^2, \ldots, c_\eta^\Lambda\}$ represents a set of consecutive $\Lambda$ cells. The number of cells ($\Lambda$) of each sequence $\kappa \in \{1, \ldots, \Omega\}$ is maintained fixed for all $\Omega$ sequences in the dataset.

After representing each trajectory into a set of cells, the Algorithm 1 is run to define the set of sequences from raw data. In line 2 it is evaluated if the number of cells forming

**TABLE II.** Number of All Sequences ($\Omega$) and Number of Unique Sequences ($\Psi$) for Different Sequence Lengths ($\Lambda$)

| $\Lambda$ | $\Omega$ | $\Psi$ |
|---|---|---|
| 4 | 2752580 | 1349 |
| 8 | 2354791 | 10473 |
| 12 | 1967109 | 36403 |
| 16 | 1598190 | 82082 |
| 20 | 1263477 | 136450 |

**Algorithm 1:** Conversion of the Dataset in $\Omega$ Sequences.

```
Data: Set of all trajectories $T_j$
Input: $\Lambda$
Output: $\Phi = \{S_1, S_2, ..., S_\Omega\}$
1  $\Phi = \{\}$
2  if $\Lambda > 1$ then
3      $S_\kappa = \{\}$
4      $\kappa = 1$
5      forall $T_j$ do
6          $i = 1$
7          foreach $c_\eta^1, ..., c_\eta^{\Xi_j} \in T_j$ do
8              $S_\kappa.append(c_\eta^i)$
9              $i = i + 1$
10             if $i > \Lambda$ then
11                 $\Phi.append(S_\kappa)$
12                 $\kappa = \kappa + 1$
13                 $i = i - (\Lambda - 1)$
14                 $S_\kappa = \{\}$
15     return $\Phi$
```

the sequence $S_\kappa$ is greater than 1. In line 7, the algorithm identifies the $\Xi_j$ cells that form each trajectory, which are added in line 8 to each sequence $S_\kappa$ until reaching the number of sequence cells ($\Lambda$). The sequence $S_\kappa$ is then copied to the set of sequences (line 11). The procedure is then repeated for all $T_j$ trajectories.

### C. CHARACTERIZATION OF THE MOBILITY DATA

This subsection assesses the set of sequences, $\Phi$, computed in Algorithm 1 for the mobility dataset. Table II indicates the amount of unique sequences ($\Psi$) and non-unique sequences ($\Omega$) considering various sequence lengths, i.e., $\Lambda = \{4, 8, 12, 16, 20\}$. As indicated in Table II, the number of sequences ($\Omega$) increases for shorter sequences (when $\Lambda$ decreases). Additionally, it is observed that the number of unique sequences ($\Psi$) increases for longer sequences.

To characterize the occurrence of each unique sequence $S_\kappa$ in the dataset, we plot the cumulative distribution function (CDF) of the unique sequences in Fig. 3. In the figure, the most likely sequences are ordered in descending order in the $x$-axis. The $y$-axis represents the cumulative probability of each sequence ($S_\kappa$). As illustrated in Fig. 3, different CDFs are achieved for the different lengths of the sequences ($\Lambda$), confirming that lower sequence occurrence probabilities are

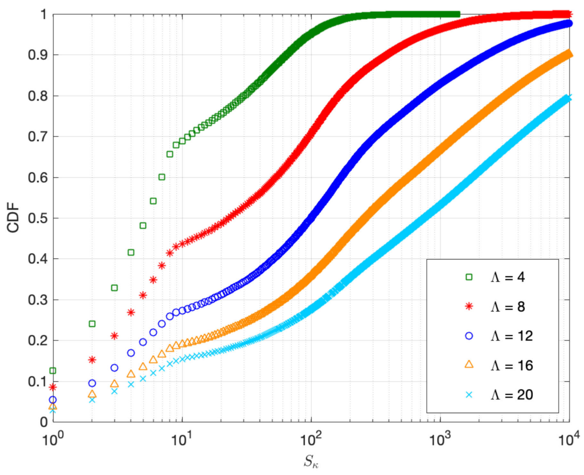

**FIGURE 3.** Cumulative distribution function of the unique sequences for different sequence lengths ($\Lambda$).

observed for longer sequences. The increase of $\Lambda$ is beneficial for the prediction because the vehicles' trajectories are described by sequences with lower occurrence probabilities, increasing the diversity of unique sequences.

## IV. BAYESIAN NETWORK MODEL

This section describes the Bayesian network model. The model contains two different stages: 1) the statistical inference; and 2) the mobility prediction. The inference stage is supported by an HMM model, and it is detailed in Section IV-A. Taking into account the inferred information, Section IV-B presents the mobility prediction algorithm OPTVIT.

### A. INFERENCE STAGE

Each trajectory is modeled through a Markov chain that describes the transition between states. The set of states is represented by $\Psi$, and each hidden state is assigned to a unique sequence $S_\kappa$. The transition probability between two adjacent hidden states represents the probability of traveling from $S_\kappa$ to $S_{\kappa+1}$ and is defined as follows

$$a_{\kappa,\kappa+1} = \frac{\#(S_\kappa, S_{\kappa+1})}{\#(S_\kappa)}, \tag{1}$$

where $\#(S_\kappa, S_{\kappa+1})$ represents the total number of transitions between two adjacent sequences $\{S_\kappa, S_{\kappa+1}\}$ that occur in all trajectories and $\#(S_\kappa)$ is the number of times that $S_\kappa$ occurs in all trajectories. The result of the transition probability $a_{\kappa,\kappa+1}$ is stored in matrix $\boldsymbol{A}$. The transition probability matrix $\boldsymbol{A}$ is an $\Psi \times \Psi$ matrix.

The HMM describes the relation between hidden events represented by different random variables and the conditional relation of all observable variables with each hidden event. In real-time, the sequences are not fully observed, but only the current cell where a vehicle is located. Thus, the visited cells represent the observed events, while the possible sequences that characterize the vehicles' mobility represent the hidden events. In this section, the cells already visited by a vehicle are considered observable variables (prior information) to predict a sequence of locations represented by a hidden event.

The hidden states, $q(t)$, and the observable states, $o(t)$, describe the HMM model at discrete time $t$. Additionally, the unique sequences, summing up $\Psi$, represent the set of hidden states of the HMM, represented by $\boldsymbol{Q} = \{q_1, q_2, \ldots, q_\Psi\}$. The $N$ cells of the grid map are the information sampled over time, thus they represent the HMM's observation set, denoted by $\boldsymbol{O} = \{o_1, o_2, \ldots, o_N\}$. The HMM's transition probability matrix is denoted by $\boldsymbol{A} = \{a_{1,1}, a_{1,2}, \ldots, a_{\Psi,\Psi}\}$, where each matrix element is represented by $a_{i,j} = P(q(t+1) = S_i | q(t) = S_j)$, $i, j \in \{1, 2, \ldots, \Psi\}$. The HMM's emission matrix is represented by $\boldsymbol{B} = \{b_1(o_1), \ldots, b_1(o_N), \ldots, b_\Psi(o_1), \ldots, b_\Psi(o_N)\}$, where $b_\kappa(o_n) = P(o_n|q_\kappa)$. Finally, $\boldsymbol{\Pi} = \{\pi_1, \pi_2, \ldots, \pi_\Psi\}$ represents the HMM's initial distribution, with $\pi_i = P(q(0) = S_i)$.

Defining $\#(q_\kappa, o_n)$ as the number occurrences of the state $o_n$ at the hidden state $q_\kappa$, the computation of each element of $\boldsymbol{B}$ is as follows

$$b_\kappa(o_n) = \frac{\#(q_\kappa, o_n)}{\Lambda}. \tag{2}$$

The HMM's initial distribution is denoted by $\boldsymbol{\Pi}$, an $1 \times \Psi$ vector where each element is computed through

$$\pi_i = \frac{\#(q_i)}{\Omega}. \tag{3}$$

After the inference stage, we estimate the most likely hidden state through the proposed prediction algorithm described in the next section.

### B. PREDICTION STAGE

In the prediction stage, we identify the most probable sequence of hidden states given a set of consecutive observations. The prediction can be seen as a decoding problem as follows,

$$P(\boldsymbol{O}|\lambda) = \max_{q_1 \le q_k \le q_\Psi} \{P(\boldsymbol{O}|q_k, \lambda) P(q_k|\lambda)\}, \tag{4}$$

and the solution relies on the identification of the hidden states that maximize the probability $P(\boldsymbol{O}|\lambda)$. An optimal solution for the decoding problem is the traditional Viterbi algorithm (TDVIT). The algorithm can be divided into three main phases described as follows:

1) First, the initial probability $\pi_i$ is multiplied by the emission probability elements. Thus, the Viterbi variable $\delta_1(i)$ is initialized with respect to the hidden state $q_i$, i.e., $\delta_1(i) = \pi_i b_i(o_1)$, $1 \le i \le \Psi$;

2) The forward variable $\delta_{2:\Lambda}(j)$ is obtained recursively with respect to each hidden state $q_j$ through $\delta_{2:\Lambda}(j) = \max_{q_1 \le q_i \le q_\Psi} \{\delta_{1:\Lambda-1}(i) a_{i,j} b_j(o_{2:\Lambda})\}$, which considers the transition probability from $q_i$ to state $q_j$ and the emission probabilities of state $q_j$.

3) Lastly, for all $\delta_T(i)$ Viterbi variables, the algorithm finds the Viterbi path with the maximum transition probability, i.e., $P^* = \max_{q_1 \le q_i \le q_\Psi} \{\delta_T(i)\}$.

**Algorithm 2:** OPTVIT Algorithm.

```
Input: $\mathcal{T}^{test}$; $\lambda = \{\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{B}\}$; $S_{test}$; $\boldsymbol{O}$; R
Output: $S_{pred}$
1  $\chi = \{c_\eta^1, c_\eta^2, \ldots, c_\eta^{\Lambda-\beta}\}$= Transform($S_{test}$)
2  $\zeta$ = Search_sequences($\chi$, $\mathcal{T}^{test}$)
3  foreach $S_j \in \zeta$ do
4      if $S_j \notin \boldsymbol{R}$ then
5          $\boldsymbol{O} = \{o_1, o_2, \ldots, o_\Lambda\}$= Transform($S_j$)
6          if $j = 1$ then
7              $P^*$ = Viterbi($\boldsymbol{O}$, $\lambda$)
8              $V_{\text{vit}}$ = $\delta_{1:(\Lambda-\beta)}(j)$
9          else
10             $P^*$ = Viterbi($\boldsymbol{O}_{(\Lambda-\beta+1):\Lambda}$, $\lambda$, $V_{\text{vit}}$)
11         $\Upsilon.append([P^*, S_j])$
12         $\mathbf{R}.append([P^*, S_j])$
13     else
14         $P^*$ = $\mathbf{R}[S_j]$
15         $\Upsilon.append([P^*, S_j])$
16 $S_{pred} = \arg\max_{P^*}\{\Upsilon\}$
17 return $S_{pred}$
```

The algorithm TDVIT, denoted as Viterbi($\boldsymbol{O}$, $\lambda$), predicts the last $\beta$ cells $(c_\eta^{\Lambda-\beta+1}, \ldots, c_\eta^\Lambda\})$ of a sequence $S_{test} = \{c_\eta^1, c_\eta^2, \ldots, c_\eta^\Lambda\}$, given that $\Lambda - \beta$ visited cells are already known. Its computational complexity is O($\Psi^2$ $\Lambda$). The computational performance of the TDVIT is enhanced in Algorithm 2, referred as OPTVIT.

The rationale behind Algorithm 2 is the computation of only the first $\Lambda - \beta$ cells, i.e., $(\delta_{1:(\Lambda-\beta)}(j))$, for the first sequence $S_j$ stored in $\zeta$. First, the algorithm copies to $S_{test}$ the $\{\Lambda - \beta\}$ cells (line 1). Then, it is copied to $\zeta$ the unique sequences in $\mathcal{T}^{test}$ beginning with the sequence in $\chi$ (line 2). Thus, $\zeta$ contains all sequences in which the latest $\beta$ cells are hypothetical candidates for the prediction. Each $S_j \in \zeta$ (line 3) is checked (line 6) to verify if the variables $\delta_{1:(\Lambda-\beta)}(j)$ were already computed.

Because all $S_j \in \zeta$ share the first $\Lambda - \beta$ observable states, i.e., $\{o_1, o_2, \ldots, o_{\Lambda-\beta}\}$, $\delta_{1:(\Lambda-\beta)}(j)$ are only computed for $S_1$ (line 7). For $S_{j>1}$ only the Viterbi variable $(\delta_\Lambda(j))$ is computed (line 10). In lines 11 and 12 the pair $[P^*, S_j]$ is stored in $\Upsilon$ and $\mathbf{R}$. $\mathbf{R}$ is used to store all probabilities $P^*$ associated with the different sequences, so we can avoid computing them again. The probabilities already computed in $P^*$ are used in line 14. Finally, in line 16 the algorithm identifies the pair $[P^*, S_j]$ exhibiting the highest $P^*$ value, which corresponds to the most likely sequence $S_{pred}$.

## V. DEEP LEARNING APPROACH

This section describes the structure of the LSTM neural network and specifies the enhancements involved in the training phase to obtain accurate vehicle trajectory predictions.

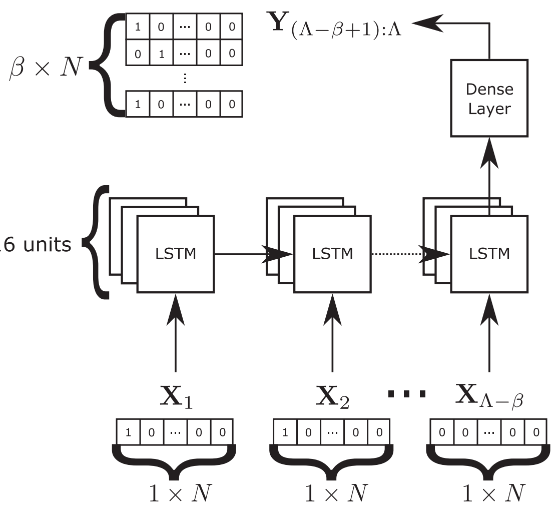

**FIGURE 4.** LSTM structure.

### A. LSTM STRUCTURE

We use an LSTM recurrent neural network to solve the vanishing gradient problem that can have a significant impact on dealing with long-term sequential data [41]. The LSTM is well-known for sharing the cell states across each forward step, making the architecture ideal to deal with trajectories, where it is possible to reinforce important patterns or discard the redundant ones.

In Fig. 4, we show the structure of the adopted LSTM neural network. As can be seen, the LSTM layer is composed by $\Lambda - \beta$ discrete time steps and admit an input vector $\mathbf{X}_i$ ($i \in \{1, \ldots, \Lambda - \beta\}$) for each step, containing the information of the visited cell. We use one-hot encoding to generate $\mathbf{X}_i$. Thus, $\mathbf{X}_i$ is a $1 \times N$ one-hot vector with the value 1 assigned to the index $\eta$, used to identify the cell of the grid map, and zeros in the remaining vector positions. Given an input sequence represented as $\{\mathbf{X}_1, ..., \mathbf{X}_{\Lambda-\beta}\}$, the proposed LSTM network computes the output $\mathbf{Y}_{(\Lambda-\beta+1):\Lambda}$. The output is an one-hot encoding matrix ($\beta \times N$) where each row $j \in \{1, \ldots, \beta\}$ contains the information of each of the $\beta$ predicted cells. The index $\eta$ of each row that contains the single 1 will correspond to the predicted cell $c^{(\Lambda-\beta+1):\Lambda}$, or the labeled output during the learning stage. In the LSTM layer, we adopt 16 LSTM units for each step. Since the structure of the LSTM unit may adopt different models, we follow the one proposed by Hochreiter & Schmidhuber [41]. The structure of the unit is composed of the operations involving the input, output, and forget gates.

To finalize the LSTM structure, we adopt a Sigmoid function in the Dense Layer. The Sigmoid activation function is a logistic function that is useful in the prediction of one-hot vectors [41], as is the case of the desired output $\mathbf{Y}_{(\Lambda-\beta+1):\Lambda}$.

### B. LSTM TRAINING

To learn the vehicles' mobility patterns, during the training process we use the dataset $\Phi$ containing the sequences described in Section III-A. We selected 70% of the dataset for the training phase, 20% for the validation, and 10% for testing. To optimize the training phase, we adopt the *Adam* optimizer and the categorical cross-entropy as the loss function. We start the training phase with a learning rate of 0.00 001, and a fixed number of 100 epochs. Furthermore, we added an early stoppage algorithm where we selected two levels of patience, i.e., the training phase stops when two negative oscillations in the loss function occur. We decided to add the stoppage algorithm to avoid over-fitting in the model.

In Table III we present the LSTM structure and model configurations.

**TABLE III.** LSTM Model Configuration

| | |
|---|---|
| **LSTM Layer** | Units = 16 |
| **Dense Layer** | Sigmoid with 16 outputs |
| **Loss Function** | Categorical cross entropy |
| **Optimizer** | *Adam* (learning rate = 0.00001) |

## VI. PERFORMANCE COMPARISON

This section evaluates the performance of the prediction algorithms proposed in Sections IV and V. The evaluation methodology is presented in Section VI-A and the accuracy of the estimation process is discussed in Section VI-B and VI-C.

### A. EVALUATION METHODOLOGY

The evaluation is based on the dataset described in Section III-B. We choose a grid map representation with $N = 16$ cells (Fig. 2), where each cell has a lateral size of 738 m and a longitudinal size of 980 m. Then, the raw data is filtered to take into account the trajectories starting and ending within the defined area of the grid map. Each trajectory is then characterized as a list of cells, and the set of sequences $\Phi$ is computed to be used as input of the Bayesian network model and the LSTM network.

The method used to evaluate the prediction process is based on the outputs of Algorithm 2 and the LSTM recurrent neural network. The prediction performance is evaluated by comparing each predicted cell $c^i_{pred}$ ($i \in \{\Lambda - \beta + 1, \ldots, \Lambda\}$), with the cell $c^i_{test}$ of the trajectory sequence in test, $S_{test}$.

The prediction performance is defined taking into account the cells correctly predicted, $c^i_{pred}$, as follows

$$\mathrm{PP} = \frac{1}{|\mathcal{T}^{test}|} \sum_{j=1}^{|\mathcal{T}^{test}|} \mathcal{F}(c^i_{pred_j}, c^i_{corr_j}), \tag{5}$$

where $\mathcal{F}(c^i_{pred_j}, c^i_{corr_j})$ holds 1 when the cell $c^i_{pred_j}$ is correctly predicted, i.e., is equal to $c^i_{corr_j}$, and holds 0 otherwise. $|\mathcal{T}^{test}|$ denotes the number of sequences in the set $\mathcal{T}^{test}$.

The estimation assessment is performed for five values of $\Lambda$ ($\Lambda = \{4, 8, 12, 16, 20\}$) and two values of $\beta$ ($\beta = \{1, 5\}$). The prediction performance was characterized for a dataset ($\mathcal{T}^{test}$) formed by $10^5$ sequences randomly selected from the dataset of unique sequences ($\Phi$). The cumulative computation time was obtained for a smaller dataset also formed by $10^3$

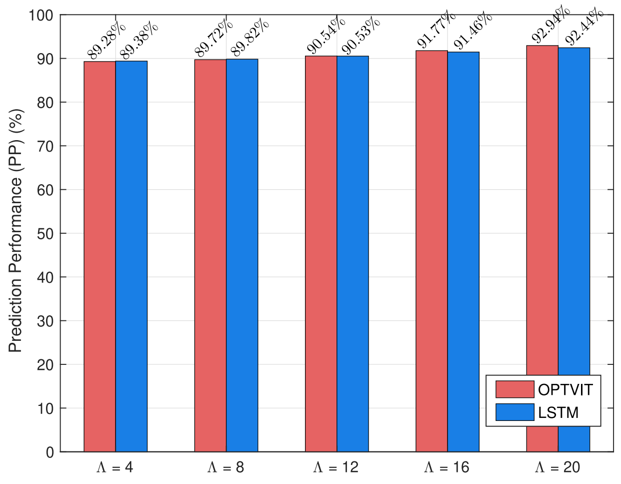

**FIGURE 5.** **Prediction performance for a single predicted cell ($\beta = 1$).**

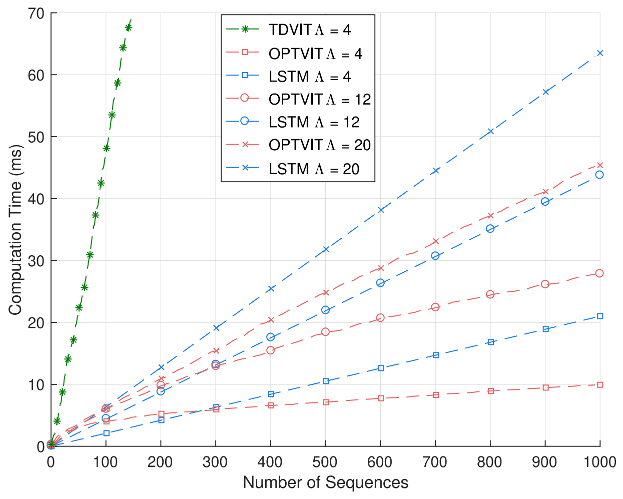

**FIGURE 6.** **Cumulative computation time for a single predicted cell ($\beta = 1$).**

sequences randomly selected from the dataset of unique sequences ($\Phi$).

The experiments were deployed using the NumPy package in Python. Regarding the setup, we have run the prediction approaches in an Intel 8-core i7-9800X @ 3.8 GHz computer with 128 GB of memory.

### B. SHORT-TERM PERFORMANCE

First, we evaluate the prediction performance for a single predicted cell ($\beta = 1$), meaning a prediction for a 15-second time horizon. Fig. 5 represents the prediction performance obtained with OPTVIT, and LSTM approaches for different $\Lambda$ values, confirming that the increase of the sequence length ($\Lambda$) improves the prediction results. The prediction process achieves higher short-term prediction performance as the amount of prior mobility information (cells) increases, and this is observed for both OPTVIT and LSTM approaches.

From Fig. 3, considering $\Lambda = 4$, we know that the 8 most probable sequences in the dataset of unique sequences represent more than 60% of the occurrence probability. This fact partially justifies the high prediction performance values in Fig. 5 even for shorter sequences. The increase of $\Lambda$ from 4 to 20 increases the prediction performance even more. This is because the number of unique sequences increases with $\Lambda$ (see Table II) and less dominant sequences (in terms of probability) are obtained for higher $\Lambda$ values (Fig. 5). Consequently, the diversity of unique sequences increases with $\Lambda$, and the mobility is more accurately described because the probability of occurrence of the sequences is not so dissimilar. Finally, the proposed methods (OPTVIT and LSTM) achieve approximately the same prediction performance of the next cell. Although the prediction performance of the TDVIT method is not represented in Fig. 5, its performance is equal to the OPTVIT approach.

As mentioned before, TDVIT, OPTVIT, and LSTM approaches achieve almost the same prediction performance for $\beta = 1$. However, we observe a significant difference in terms of the computation time performance of three methods, as shown in Fig. 6, where the prediction time ($\tau$) is evaluated for each sequence, considering a test set with 1000 sequences randomly observed in $\Phi$. The computation time plotted in the figure is the cumulative time to predict the number of sequences indicated in the x-axis. The average of the prediction time for the 1000 sequences is represented in Table IV.

**TABLE IV.** **Average Prediction Time Per Sequence**

| | $\Lambda$ | $\mathrm{E}[\tau]$ ($\mu s$) |
|---|---|---|
| TDVIT | 4 | $215.5197 \pm 1.3423$ |
| | 20 | $488.0032 \pm 1.4942$ |
| LSTM | 4 | $21.0000 \pm 0.2832 \times 10^{-13}$ |
| | 20 | $63.5052 \pm 0.1370 \times 10^{-12}$ |
| OPTVIT | 4 | $4.1472 \pm 0.0020$ |
| | 20 | $10.6402 \pm 0.0153$ |

The results in Table IV and Fig. 6 show that the computation time achieves the lowest values for the OPTVIT algorithm. Moreover, the computation time and prediction time increase with $\Lambda$, which is explained by the increasing number of cells that compose the observable state set. i.e., a higher amount of prior data is used in the prediction. In Fig. 6 we also plot the computation time for the Viterbi algorithm (TDVIT) without the enhancements proposed in OPTVIT. The results for TDVIT are only plotted for $\Lambda = 4$ because the other computation times for $\Lambda = \{8, 12, 16, 20\}$ are significantly higher and are not comparable with the other two prediction methods.

Regarding the LSTM approach, since the duration of each computation of the neural network outputs is constant, the total computation time increases linearly with the number of sequences as shown in Fig. 6. However, the OPTVIT approach achieves lower computational times because it grows sub-linearly with the number of sequences. This is mainly due to the reuse of prior computations in the OPTVIT algorithm, which effectively leads to higher computational times for the first sequences but can be used afterward to avoid unnecessary computations associated with similar sequences.

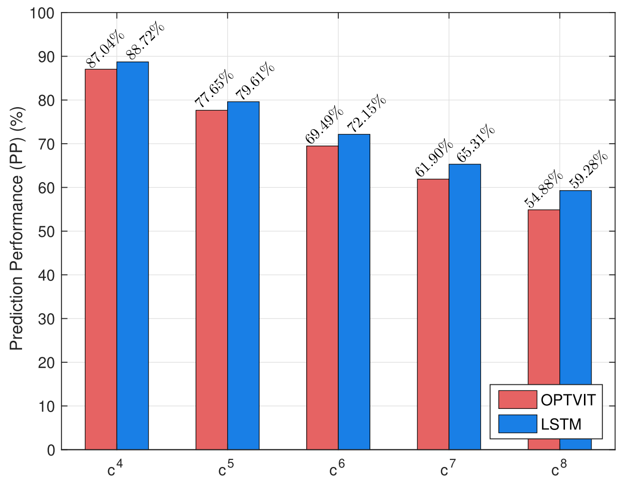

**FIGURE 7.** **Prediction performance for the next five predicted cells ($\beta = 5$), when $\Lambda = 8$ is considered.**

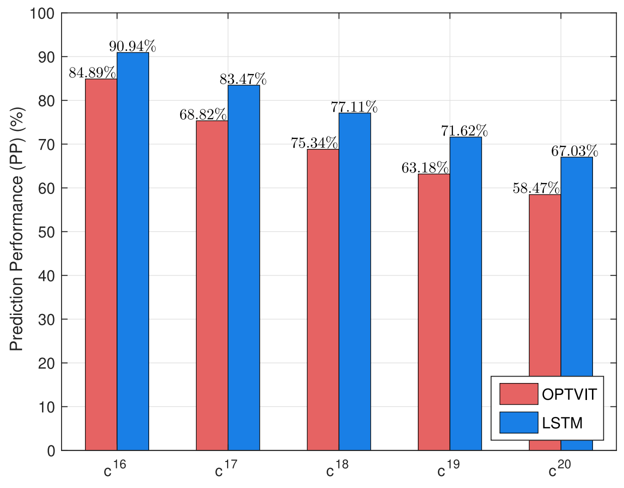

**FIGURE 8.** **Prediction performance for the next five predicted cells ($\beta = 5$), when $\Lambda = 20$ is considered.**

## C. LONG-TERM PERFORMANCE

Instead of predicting only the next cell, we analyze the prediction performance of OPTVIT and LSTM methods for the next five predicted cells. In the time domain, the prediction of a single cell occurs every 15 seconds. Consequently, the prediction of the next five cells is a prediction for the next 75 seconds. The results in Figs. 7 and 8 indicate the prediction performance for long-term predictions, considering the next five predicted cells ($\beta = 5$). Because different sequence lengths ($\Lambda$) were adopted, for $\Lambda = 8$ in Fig. 7 the predicted sequence cells are $c^4$, $c^5$, $c^6$, $c^7$, $c^8$. For $\Lambda = 20$ in Fig. 8 the predicted sequence cells are $c^{16}$, $c^{17}$, $c^{18}$, $c^{19}$, $c^{20}$. We recall that more prior information (15 cells) is used in the prediction of the scenario considered in Fig. 8, which compares with only 3 cells in the scenario considered in Fig. 7.

The results in Figs. 7 and 8 show that the prediction performance decreases for a longer time horizon. In Fig. 7 we observe that 87.04% of successful prediction rate is achieved for

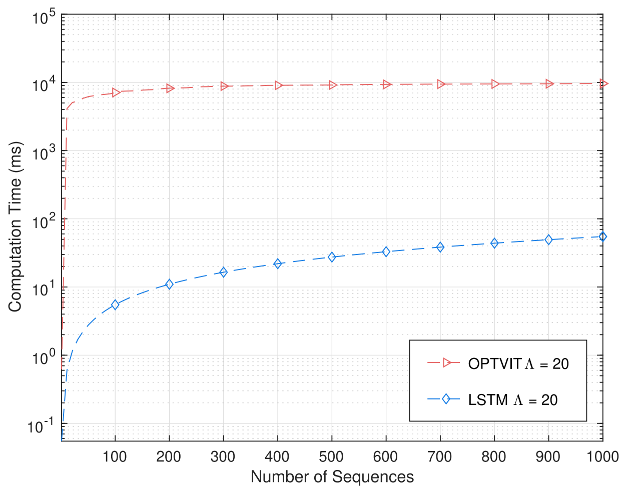

**FIGURE 9.** **Cumulative computation time for the next five predicted cells ($\beta = 5$).**

**TABLE V.** **Average Prediction Time Per Sequence**

| | $\Lambda$ | $E[\tau]$ ($\mu s$) |
|---|---|---|
| OPTVIT | 20 | $578.4637 \pm 1.8063$ |
| LSTM | 20 | $54.9000 \pm 0.1604 \times 10^{-12}$ |

the next cell (estimated for a 15-second time horizon) adopting the OPTVIT algorithm. However, the prediction probability decreases to 54.88% for the fifth cell, i.e., for a 75-second time horizon. The decrease of the prediction performance with the increase of the time horizon is due to the increase of uncertainty associated with the long-term time duration.

By comparing the results in Figs. 7 and 8, we conclude that the LSTM method's performance is improved with the increase of $\Lambda$, whereas for the OPTVIT method this trand only occurs from the 3rd predicted cell ($c^6$ and $c^{18}$ in Figs. 7 and 8, respectively). Contrarily to the results achieved for $\beta = 1$, in which both methods present similar performances, for long-term predictions ($\beta = 5$) the LSTM method presents higher performance performance.

Additionally, we analyze the time performance of the proposed prediction methods for long-term predictions ($\beta = 5$). The cumulative computation times for the 1000 unique sequences are plotted in Fig. 9 and the average of the sequences' computation times are given in Table V.

The results in Fig. 9 and Table V show that the cumulative computation time and the average prediction time favors the LSTM approach. Regarding the OPTVIT approach, the curves in Fig. 9 indicate a higher computation time for the first predicted cells (estimated for the first 100 sequences) and a decrease of the computation time for the remaining ones. This is due to the high computation efforts related to the computation of the Viterbi variables, which are computed once and its posterior computation is avoided by the enhancements proposed in OPTVIT. The cumulative computation time of the predictions obtained with the LSTM approach has a linear trend since no advantage is taken from sequences computed so far. For long-term prediction, the average computation times presented in Table V indicate that LSTM achieves an order of magnitude speedup for the adopted $\Lambda$ and $\beta$ values. Consequently, the results show that both prediction performance and computation times benefit from the adoption of the LSTM approach for long-term vehicular trajectory prediction.

## VII. CONCLUSION

This paper has proposed two efficient methodologies to predict vehicles' future locations: (a) a Bayesian network model that models the interaction between sequences (sub-trajectory) and between the cells and the sequences; (b) a LSTM RNN that is adequate to deal with sequential data. We have compared the performance of both prediction methodologies for short-term and long-term predictions. The experimental results indicate that the prediction accuracy is improved in both methods as more observations as more prior information is used in the prediction process. The two proposed methods have achieved almost the same performance for short-term predictions. However, we have shown that the LSTM RNN is more suitable for long-term predictions. Additionally, we have compared the two proposed prediction methods' time performance, showing that the computation time of (a) is shorter for short-term predictions while (b) is shorter for long-term predictions.

Although the methodology (a) can predict the next location in a shorter time (as shown in Fig. 6) and does not require a time-consuming learning task every time new data is added to the inference dataset, the results reported in this work show that it is not recommended to predict more than a single location because the computation time can be several orders of magnitude higher than the methodology b). Regarding b), the main limitation is the learning task, which hampers the adoption of fresh prior data into the prediction stage.

As future work, it would be interesting to evaluate the computation performance and accuracy for different spatial sampling strategies and using other deep learning techniques with the main purpose of achieving a lower computation time without compromising the mobility prediction accuracy.

## ACKNOWLEDGMENT

We thank the support of André Ip in the results reported in Figures 7 and 8.

## REFERENCES

[1] Y. Sun, L. Xu, Y. Tang, and W. Zhuang, "Traffic offloading for online video service in vehicular networks: A cooperative approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 8, pp. 7630–7642, Aug. 2018.

[2] Y. Zhu, Y. Wu, and B. Li, "Trajectory improves data delivery in urban vehicular networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 4, pp. 1089–1100, Apr. 2014.

[3] S. Chen, Y. Li, W. Ren, D. Jin, and P. Hui, "Location prediction for large scale urban vehicular mobility," in *Proc. 9th Int. Wireless Commun. Mobile Comput. Conf.*, 2013, pp. 1733–1737.

[4] S. Qiao, D. Shen, X. Wang, N. Han, and W. Zhu, "A self-adaptive parameter selection trajectory prediction approach via hidden Markov models," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 1, pp. 284–296, Feb. 2015.

[5] Q. Lv, Y. Qiao, N. Ansari, J. Liu, and J. Yang, "Big data driven hidden Markov model based individual mobility prediction at points of interest," *IEEE Trans. Veh. Technol.*, vol. 66, no. 6, pp. 5204–5216, Jun. 2017.

[6] D. Brockmann, L. Hufnagel, and T. Geisel, "The scaling laws of human travel," *Nature*, vol. 439, no. 7075, pp. 462–465, Jan. 2006.

[7] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding individual human mobility patterns," *Nature*, vol. 453, no. 7196, pp. 779–782, Jun. 2008.

[8] L. N. Balico, A. A. F. Loureiro, E. F. Nakamura, R. S. Barreto, R. W. Pazzi, and H. A. B. F. Oliveira, "Localization prediction in vehicular ad hoc networks," *IEEE Commun. Surv. Tuts.*, vol. 20, no. 4, pp. 2784–2803, Oct./Dec. 2018.

[9] V. Namboodiri and L. Gao, "Prediction-based routing for vehicular ad hoc networks," *IEEE Trans. Veh. Technol.*, vol. 56, no. 4, pp. 2332–2345, Jul. 2007.

[10] F. Sun, X. Wang, Y. Zhang, W. Liu, and R. Zhang, "Analysis of bus trip characteristic analysis and demand forecasting based on GA-NARX neural network model," *IEEE Access*, vol. 8, pp. 8812–8820, Jan. 2020.

[11] P. Lytrivis, G. Thomaidis, M. Tsogas, and A. Amditis, "An advanced cooperative path prediction algorithm for safety applications in vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 3, pp. 669–679, Sep. 2011.

[12] A. Ramchandani, C. Fan, and A. Mostafavi, "Deepcovidnet: An interpretable deep learning model for predictive surveillance of covid-19 using heterogeneous features and their interactions," *IEEE Access*, vol. 8, pp. 159 915–159 930, Aug. 2020.

[13] J. S. Jia, X. Lu, Y. Yuan, G. Xu, J. Jia, and N. A. Christakis, "Population flow drives spatio-temporal distribution of covid-19 in China," *Nature*, vol. 582, no. 7812, pp. 389–394, Jun. 2020.

[14] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma, "Understanding mobility based on GPS data," in *Proc. 10th Int. Conf. Ubiquitous Comput. (UbiComp)*, Sep. 2008, pp. 312–321.

[15] L. Yao, A. Chen, J. Deng, J. Wang, and G. Wu, "A cooperative caching scheme based on mobility prediction in vehicular content centric networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 6, pp. 5435–5444, Jun. 2018.

[16] K. Niu, C. Cheng, J. Chang, H. Zhang, and T. Zhou, "Real-time taxi-passenger prediction with L-CNN," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 4122–4129, May 2019.

[17] Y. Tang, N. Cheng, W. Wu, M. Wang, Y. Dai, and X. Shen, "Delay-minimization routing for heterogeneous VANETs with machine learning based mobility prediction," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3967–3979, Apr. 2019.

[18] Y. Xing, C. Lv, and D. Cao, "Personalized vehicle trajectory prediction based on joint time-series modeling for connected vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 1341–1352, Feb. 2020.

[19] L. Zhao, Y. Liu, A. Al-Dubai, A. Y. Zomaya, G. Min, and A. Hawbani, "A novel generation adversarial network-based vehicle trajectory prediction method for intelligent vehicular networks," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 2066–2077, Feb. 2021.

[20] Y. Li, W. Ren, D. Jin, P. Hui, L. Zeng, and D. Wu, "Potential predictability of vehicular staying time for large-scale urban environment," *IEEE Trans. Veh. Technol.*, vol. 63, no. 1, pp. 322–333, Jan. 2014.

[21] H. Zhang, Y. Wu, H. Tan, H. Dong, F. Ding, and B. Ran, "Understanding and modeling urban mobility dynamics via disentangled representation learning," *IEEE Trans. Intell. Transp. Syst*, doi: 10.1109/TITS.2020.3030259.

[22] J. Ding, H. Liu, L. T. Yang, T. Yao, and W. Zuo, "Multiuser multivariate multiorder Markov-based multimodal user mobility pattern prediction," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4519–4531, May 2020.

[23] S. Fang, L. Lin, Y. Yang, X. Yu, and Z. Xu, "CityTracker: Citywide individual and crowd trajectory analysis using hidden Markov model," *IEEE Sensors J.*, vol. 19, no. 17, pp. 7693–7701, Sep. 2019.

[24] F. Li, Q. Li, Z. Li, Z. Huang, X. Chang, and J. Xia, "A personal location prediction method based on individual trajectory and group trajectory," *IEEE Access*, vol. 7, pp. 92 850–92 860, Jul. 2019.

[25] K. Ansari, "Cooperative position prediction: Beyond vehicle-to-vehicle relative positioning," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 1121–1130, Mar. 2020.

[26] S. Cheng, F. Lu, and P. Peng, "Short-term traffic forecasting by mining the non-stationarity of spatiotemporal patterns," *IEEE Trans. Intell. Transp. Syst.*, pp. 1–19, May 2020.

[27] C. Wang, L. Ma, R. Li, T. S. Durrani, and H. Zhang, “Exploring trajectory prediction through machine learning methods,” *IEEE Access*, vol. 7, pp. 101 441–101 452, Aug. 2019.
[28] S. Qiao, N. Han, J. Wang, R. Li, L. A. Gutierrez, and X. Wu, “Predicting long-term trajectories of connected vehicles via the prefix-projection technique,” *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 7, pp. 2305–2315, Jul. 2018.
[29] P. Rathore, D. Kumar, S. Rajasegarar, M. Palaniswami, and J. C. Bezdek, “A scalable framework for trajectory prediction,” *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 10, pp. 3860–3874, Oct. 2019.
[30] Y. Qiao, Y. Cheng, J. Yang, J. Liu, and N. Kato, “A mobility analytical framework for big mobile data in densely populated area,” *IEEE Trans. Veh. Technol.*, vol. 66, no. 2, pp. 1443–1455, Feb. 2017.
[31] G. Marfia and M. Roccetti, “Vehicular congestion detection and short-term forecasting: A new model with results,” *IEEE Trans. Veh. Technol.*, vol. 60, no. 7, pp. 2936–2948, Sep. 2011.
[32] Y. Daraghmi, C. Yi, and T. Chiang, “Negative binomial additive models for short-term traffic flow forecasting in urban areas,” *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 2, pp. 784–793, Apr. 2014.
[33] H. Zhang and L. Dai, “Mobility prediction: A survey on state-of-the-art schemes and future applications,” *IEEE Access*, vol. 7, pp. 802–822, Dec. 2019.
[34] Y. Zhang, J. Hu, J. Dong, Y. Yuan, J. Zhou, and J. Shi, “Location prediction model based on Bayesian network theory,” in *Proc. IEEE Glob. Telecommun. Conf.*, Nov. 2009, pp. 1–6.
[35] C. Liu, E. Jou, and C. Lee, “Analysis and prediction of trajectories using bayesian network,” in *Proc. 6th Int. Conf. Natural Comput.*, Aug. 2010, pp. 3808–3812.
[36] M. Dash, K. K. Koo, J. B. Gomes, S. P. Krishnaswamy, D. Rugeles, and A. Shi-Nash, “Next place prediction by understanding mobility patterns,” in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops*, Mar. 2015, pp. 469–474.
[37] A. De Brébisson, É. Simon, A. Auvolat, P. Vincent, and Y. Bengio, “Artificial neural networks applied to taxi destination prediction,” in *Proc. 2015th Int. Conf. ECML PKDD Discov. Challenge*, Porto, Portugal, 2015,vol. 1526, pp. 40–51.
[38] J. Lv, Q. Sun, Q. Li, and L. Moreira-Matias, “Multi-scale and multi-scope convolutional neural networks for destination prediction of trajectories,” *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 8, pp. 3184–3195, Aug. 2020.
[39] F. Altché and A. de La Fortelle, “An LSTM network for highway trajectory prediction,” in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2017, pp. 353–359.
[40] “Taxi Service Trajectory (TST) Prediction Challenge, @ ECML/PKDD 2015,” [Online]. Available: http://www.geolink.pt/ecmlpkdd2015-challenge/dataset.html
[41] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

**LUIS IRIO** received the M.Sc. and Ph.D. degrees in electrical and computer engineering from the Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Lisbon, Portugal, in 2013 and 2019, respectively. From 2014 to 2015, he was a Researcher with CTS-UNINOVA, Portugal. During 2015–2019, the Ph.D. research work was developed with Instituto de Telecomunicações, Lisbon, Portugal. He is currently a Postdoctoral Researcher with Instituto de Telecomunicações. During the Ph.D. period, he has authored or coauthored seven articles in prominent international journals (six of which in the first quarter of the Scimago Journal Rank) and 16 papers in international peer-reviewed conferences. His current research interests include mobility prediction, machine learning, and big data analytics.

**RODOLFO OLIVEIRA** (Senior Member, IEEE) received the Licenciatura degree in electrical engineering from the Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa (UNL), Lisbon, Portugal, in 2000, the M.Sc. degree in electrical and computer engineering from the Instituto Superior Técnico, Technical University of Lisbon, Portugal, in 2003, and the Ph.D. degree in electrical engineering from UNL, in 2009. From 2007 to 2008, he was a Visiting Researcher with the University of Thessaly, Volos, Greece. From 2011 to 2012, he was a Visiting Scholar with Carnegie Mellon University, Pittsburgh, PA, USA. He is currently with the Department of Electrical and Computer Engineering, UNL, and is also affiliated as a Senior Researcher with the Instituto de Telecomunicações, where he researches in the areas of Wireless Communications, Computer Networks, and Computer Science. He is in the Editorial Board of the Ad Hoc Networks (Elsevier), the IEEE Open Journal of the Communications Society, and the IEEE Communications Letters.