Edge Learning of Vehicular Trajectories at Regulated Intersections

Edge Learning of Vehicular Trajectories at Regulated Intersections

Dinesh Cyril Selvaraj*, Christian Vitale†, Tania Panayiotou†, Panayiotis Kolios†,
Carla Fabiana Chiasserini* and Georgios Ellinas†
*CARS@Polito, Politecnico di Torino, Torino, Italy
†KIOS CoE and Dept. Electrical and Computer Eng., University of Cyprus, Nicosia, Cyprus

***Abstract*—Trajectory prediction is crucial in assisting both human-driven and autonomous vehicles. Most of the existing approaches, however, focus on straight stretches of road and do not address trajectory prediction at intersections. This work aims to fill this gap by proposing a solution that copes with the higher complexity exhibited for the intersection scenario, leveraging the 5G-MEC capabilities. In particular, the reduced latency and edge computational power are exploited to centrally collect and process measurements from both vehicles (e.g., odometry) and road infrastructure (e.g., traffic light phases). Based on such a holistic system view, we develop a Long Short Term Memory (LSTM) recurrent neural network which, as shown through simulations using a real-world dataset, provides high-accuracy trajectory predictions. The encountered challenges and advantages of the presented approach are analyzed in detail, paving the way for a new vehicle trajectory prediction methodology.**

***Index Terms*—LSTM Recurrent Networks, Trajectory Predictions, Intelligent Transportation Systems**

I. Introduction

As accidents on the worldwide road infrastructure cause more than 1.35 million casualties annually [1], safety is of the utmost concern for the automotive sector. Furthermore, a new generation of (connected) autonomous vehicles is about to enter the market and new, intelligent functionalities need to be developed [2]. For both human-driven vehicles and autonomous vehicles, trajectory prediction plays a crucial role; in the former case, it represents an important tool for collision avoidance and driver assistance, while in the latter it helps to estimate the actions of surrounding vehicles and better plan the mobility of autonomous vehicles.

Several solutions for vehicle trajectory prediction have been proposed, based on: (i) map-based maneuver classification and probabilistic trajectory estimation [3], (ii) exploitation of contextual information, e.g., the movement of surrounding vehicles [4], and (iii) non-linear regression techniques for predicting the target vehicle's future location, e.g., LSTM Recurrent Neural Networks (RNNs) [5], [6]. In particular, given their high accuracy, non-linear regression techniques considering also contextual information have recently attracted the interest of the research community [7].

This work was supported in part by the European Union's Horizon 2020 Research and Innovation Programme under Grant 739551 (KIOS CoE) and Grant 101003439 (C-AVOID), and in part by the Government of the Republic of Cyprus through the Directorate General for European Programmes, Coordination and Development. This work was also partially funded by the EU Commission under the RAINBOW project (Grant Agreement no. 871403)
Correspondence: vitale.christian@ucy.ac.cy

Most of these efforts focus on highway scenarios or straight stretches of urban roads, without tackling the more challenging scenario of road intersections. Unlike highways, where vehicles tend to maintain a constant speed, at regulated intersections vehicle speed changes significantly depending on factors such as distance from intersection, queue at traffic lights, and traffic light status. Also, map-based maneuvers' classification and contextual information-based predictions suffer from the large number of possible movements and interactions and from intrinsic quantization errors.

This work aims to advance trajectory prediction in the presence of intersections, leveraging both Vehicle-to-Infrastructure (V2I) and Infrastructure-to-Infrastructure (I2I) communications to collect relevant data at a centralized Intersection Manager (IM). Such data, generated by both vehicles (e.g., location and speed) and smart city sensors (e.g., traffic light phases), are processed at the 5G Multi-Access Edge Computing (5G-MEC) platform to estimate the vehicles' future locations with low computational cost. The reduced latency ensured by 5G-MEC and the holistic view available at the IM ensure that the proposed framework promptly provides the vehicles with accurate trajectory predictions. These predictions are computed through our proposed Encoder-Decoder model, which leverages past observations to learn habitual behaviors, in a simple, yet effective manner. This approach is validated through Simulation of Urban MObility (SUMO), using the Luxembourg traffic dataset [8], and a real-world dataset obtained via video captured by drones. The cumulative density function (CDF) of the prediction error is used to showcase the benefits of the proposed solution and the challenges faced in the addressed scenario. To summarize, our main contributions are as follows:

- unlike most of the previous work, we focus on road intersections and design a novel framework for vehicle trajectory prediction based on an enhanced LSTM Encoder-Decoder model. The model takes as input a selected set of vehicle and road infrastructure observations that lead to a high accuracy trajectory prediction;
- we provide a detailed explanation of the advantages and challenges encountered, analyzing the CDF of the difference between predictions and ground truth;
- we validate our approach using both a SUMO and a real-world mobility dataset representing a mid-sized city.

II. Related Work

As mentioned, several approaches have been presented for vehicle trajectory prediction, and a useful survey thereof can be found in [9]. Nevertheless, very few works address road scenarios that include intersections. Specifically, early studies have mainly addressed the problem of classifying the vehicles' movement intentions at an intersection. Based on the measurements obtained from a test vehicle at a T-intersection in Germany, the work in [3] is the first that introduces both a categorization technique identifying vehicles turning, or proceeding straight, at an intersection, and a 1-second look-ahead prediction through Monte Carlo simulation. Similarly, [10] uses LSTM RNNs to detect whether vehicles turn or proceed straight at an intersection, achieving a classification accuracy of 85%. Although representing an important contribution, we remark that [10] has a different scope than ours, and that predicting driver intentions is much simpler than providing full trajectory predictions, as we do in our work.

Full trajectory prediction, i.e., predictions not involving only human driver future maneuvers, are reported in [6], [11], [12]. In these works, a multi-modal probability distribution for the vehicle's future locations is obtained through a set of concatenated neural networks, simplifying the environment ahead of the ego vehicle (i.e., the vehicle under test): (i) with a regular grid-map; (ii) with a finite set of possible target locations; or (iii) with a schematic (rasterized) representation of the road infrastructure. Although very relevant, the presented solutions face inevitable discretization issues, which become even more prominent in complex scenarios as in those including intersections. To solve this issue, [13] derives a multi-modal distribution of the vehicle's future location based on a finite set of possible maneuvers. This approach can be easily applied when the possible choices of the ego vehicle and of the surrounding vehicles are limited, i.e., at a highway section, while its complexity becomes overwhelming in urban environments. A further extension [14] introduces a latent variable to learn from the data obtained by the different possible actions that a vehicle may perform. In this case, given the fact that possible maneuvers are obtained through sampling, the final set of the driver's intentions are difficult to obtain and to interpret.

Better suited to intersections and dense environments are the approaches that, instead of modeling driver intentions, they evaluate vehicles' interactions to improve trajectory prediction. In [7], [15], interactions are modeled as: (i) a graph, where edges represent a function of the distance between vehicles, or (ii) a generative adversarial network, which outputs a possible trajectory prediction depending on the forecasted coordination between vehicles. On one hand, such approaches can exploit the large number of vehicles present at intersections to improve the accuracy of the predicted future locations. On the other hand, complexity can grow exponentially, especially when full mesh relationships among vehicles are considered.

Unlike previous approaches, this work presents a framework exploiting a holistic view of the system, including information obtained from the vehicles and the infrastructure, as well as the extended capabilities of 5G-MEC. Furthermore, the encountered challenges for the proposed computationally inexpensive trajectory prediction obtained with LSTM are presented and analyzed, so as to help future research directions.

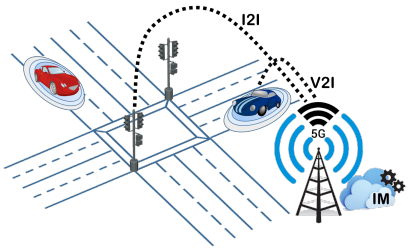

Fig. 1: A 5G-MEC based scenario, including an IM collecting data generated by the vehicles and the road infrastructure.

III. System Model

This work focuses on an urban intersection (Fig. 1), where an IM is hosted in a 5G-MEC platform [16] in the proximity of a gNodeB (gNB) covering the geographical area around the intersection. Through integrated V2I and V2V communications, the IM can gather data from multiple sources, namely, (i) onboard sensors measurements sent by the vehicles crossing the intersection through Cooperative Awareness Messages (CAMs), and (ii) contextual and environmental measurements collected by the road infrastructure. Note that CAMs can include several information provided by the vehicle's Controller Area Network (CAN) bus, including speed, direction, steering angle, acceleration, braking, yaw rate, and relative distances with surrounding vehicles obtained through lidar. As for the data collected by the road infrastructure, this can include traffic light phases, vehicle lane information, and number of vehicles in each lane. Among those, in this work, we identify which information should be fed to our predictive model, ultimately leading to high performance accuracy.

Further, the benefits of using a central entity at the 5G-MEC for trajectory estimation are multifold: (i) at any time, the IM has a significantly richer view of the intersection than the individual vehicles; (ii) the IM can collect the CAMs from the vehicles crossing the intersection in a local database, and use such extended historical knowledge to train a model better than a vehicle processing only its own data; (iii) unlike cloud implementations, the IM can collect the required data so as to infer the trajectory prediction of multiple vehicles in real time, hence providing such additional knowledge to any safety or path planning application that may be requiring it.

Given the aforementioned scenario, next we present our solution for trajectory prediction, intended to run at the IM.

IV. Vehicle Trajectory Estimation at Intersections: An LSTM Approach

Vehicle trajectory prediction is essentially a time-series forecasting problem in which past observations are exploited to predict one or more future observations. Capturing the intrinsic non-linear nature of the trajectory estimation problem and, because of their ability to scale over high-dimensionality

datasets, deep learning models have been revolutionary for ML in general, and for control and management of transportation systems in particular. Among the various deep learning models (e.g., feed forward NNs, recurrent NNs, convolutional NNs), LSTM RNNs have gained considerable attention in the research community as a possible candidate to solve the gradient vanishing problem, i.e., the problem of storing long time-steps in the learning memory [17]. Furthermore, LSTM RNNs have proved to learn much faster than conventional RNNs.

To this end, in this work, an LSTM is applied to model vehicle trajectories at intersections. Specifically, an encoder-decoder LSTM model is adopted [18], as it is a model designed for problems where an input sequence is mapped into an output sequence of arbitrary length. This is precisely what happens in trajectory prediction problems, where the historical data of the vehicle's movement and applied controls are used to project, over a target time window, the vehicle's future location. Such information can be used by the IM, for example, to proactively and timely act upon a possible (i.e., predicted) collision.

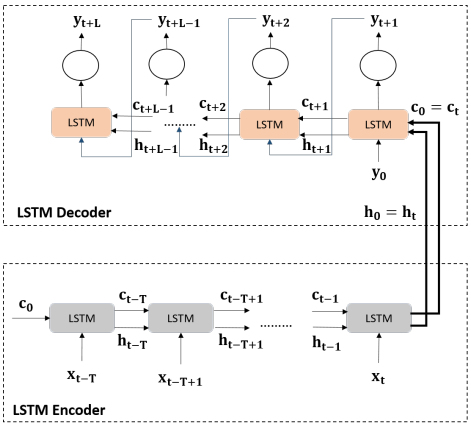

Fig. 2: The encoder-decoder LSTM architecture used for vehicle trajectory estimation.

A. Problem Statement

In our use case, the objective of the encoder-decoder LSTM model is to learn a non-linear function $F(\cdot)$ that, given the past and present vehicle's information,

$$\mathbf{x} = [\mathbf{x}_{t-T}, \mathbf{x}_{t-T+1}, \cdots, \mathbf{x}_{t-1}, \mathbf{x}_t], \tag{1}$$

accurately predicts the vehicle's future location,

$$\mathbf{y} = [\mathbf{y}_{t+1}, \mathbf{y}_{t+2}, \cdots, \mathbf{y}_{t+L}], \tag{2}$$

where T are the past and present observations, L is the prediction window, and t is the present time instant. Each $\mathbf{x_{t-k}} \in \mathbb{R}^d$ is a vector containing d measurements, on-board and infrastructure-based, sampled at time instant $t-k\tau$, $k = 0, .., T$. Each $\mathbf{y_{t+k'}} \in \mathbb{R}^{d'}$ is a vector describing the trajectory values, longitude and latitude, at a future time instant $t+k'\tau$, $k' = 1, ..., L$. Therefore, past and future time instants of interest are τ seconds apart, with $\mathbf{x}$ and $\mathbf{y}$ forming a time series (i.e., being sequential in time). Vectors $\mathbf{x}$ and $\mathbf{y}$ are formally defined in Sec. IV-C.

B. The Encoder-decoder LSTM Model

The encoder-decoder LSTM architecture we use is illustrated in Fig. 2. Note that, for simplicity, encoders and decoders of one layer only are considered, but the model can be extended to more hidden layers (i.e., stack of LSTM layers). Both encoder and decoder components are formed by LSTM cells and in Fig. 2 appear unfolded through time to depict their operation over the input and output sequences. In reality, as described later, inputs are processed sequentially, and outputs are obtained one after the other (and reintroduced at the input of the decoder for predicting the next time step).

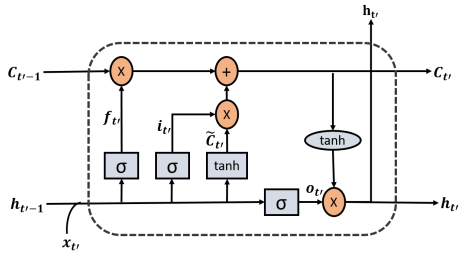

Fig. 3: General LSTM cell structure.

As shown in Fig. 2, the basic components of the model are the LSTM cells. Figure 3 shows a representation of the LSTM cell when the information at a generic time instant $t' \in [t-T, ..., t, .., t+L]$ is processed. The key of LSTM is the cell state $\mathbf{c}_{t'}$, that acts as the cell memory, storing a summary of the past input sequence. At the output, the LSTM cell combines such accumulated memory with the input received, so as to obtain the hidden state $\mathbf{h}_{t'}$, also representing the output vector of the LSTM cell. Note that $\mathbf{h}_{t'}$ is also passed to any upper layer, if present. Furthermore, at the decoder cell, the output $\mathbf{h}_{t'}$ is transformed to generate $\mathbf{y}_{t'-1}$.

An LSTM cell is composed of an input gate vector $\mathbf{i}_{t'}$, an output gate vector $\mathbf{o}_{t'}$, and a forget gate vector $\mathbf{f}_{t'}$. The three gate vectors control the flow of information in and out of the LSTM cell. Specifically, with the input gate vector the LSTM cell decides whether to update the cell state or not, with the forget gate vector the LSTM cell can erase cell state memory, and with the output gate vector it decides whether to make the output information available. The following recursive equations describe in detail how the LSTM cell works:

$$\mathbf{i}_{t'} = \sigma(\mathbf{x}_{t'}\mathbf{U}_i + \mathbf{h}_{t'-1}\mathbf{W}_i), \tag{3}$$
$$\mathbf{f}_{t'} = \sigma(\mathbf{x}_{t'}\mathbf{U}_f + \mathbf{h}_{t'-1}\mathbf{W}_f), \tag{4}$$
$$\mathbf{o}_{t'} = \sigma(\mathbf{x}_{t'}\mathbf{U}_o + \mathbf{h}_{t'-1}\mathbf{W}_o), \tag{5}$$
$$\tilde{\mathbf{c}}_{t'} = \tanh(\mathbf{x}_{t'}\mathbf{U}_g + \mathbf{h}_{t'-1}\mathbf{W}_g), \tag{6}$$
$$\mathbf{c}_{t'} = \mathbf{f}_{t'} \circ \mathbf{c}_{t'-1} + \mathbf{i}_{t'} \circ \tilde{\mathbf{c}}_{t'}, \tag{7}$$
$$\mathbf{h}_{t'} = \tanh(\mathbf{c}_{t'}) \circ \mathbf{o}_{t'}, \tag{8}$$

where $\mathbf{x}_{t'}$ is a generic input vector, $\sigma(\cdot)$ is the sigmoid function, $\circ$ denotes the element-wise product, $\mathbf{W}_i \in \mathbb{R}^{u\times u}$, $\mathbf{W}_f \in \mathbb{R}^{u\times u}$, $\mathbf{W}_o \in \mathbb{R}^{u\times u}$, $\mathbf{U}_i \in \mathbb{R}^{d\times u}$, $\mathbf{U}_f \in \mathbb{R}^{d\times u}$, $\mathbf{U}_g \in \mathbb{R}^{d\times u}$ are linear transformation matrices (i.e., unknown parameters obtained during learning), and u is the number of hidden units.

In the encoder-decoder LSTM, the encoder reads each vector $\{\mathbf{x}_{t-k'}\}_{k'=0}^{T}$ of an input trajectory $\mathbf{x}$ sequentially. As

it reads each vector, the cell state of the LSTM changes according to Eq. (7) and the hidden state changes according to Eq. (8). After reading the end of the trajectory, the encoder summarizes the entire input sequence into the final vectors $\mathbf{c}_t$ and $\mathbf{h}_t$. The decoder is trained to generate an output trajectory $\{\mathbf{y}_{t+k}\}_{k=1}^{L}$ sequentially, given $\mathbf{c}_t$ and $\mathbf{h}_t$ as the initial cell and hidden states (Fig. 2). The decoder is also initialized according to a dummy input $\mathbf{y}_0$. The decoder recursively generates the outputs $\mathbf{y}_{t+1}, .., \mathbf{y}_{t+L}$. At every update k, the decoder feeds each output $\mathbf{y}_{t+k-1}$ obtained in the previous update, to the input of the current update. Nevertheless, the generic output $\mathbf{y}_{t+k}$ is not the output of the decoder cell, but it is obtained as $\mathbf{y}_{t+k} = g(\mathbf{h}_{t+k})$, where $g(\cdot)$ can be the activation function of the output layer and $\mathbf{h}_{t+k}$ is given by Eq. (8). In our trajectory prediction problem, the output of the decoder is followed by a fully connected dense layer (in essence, a linear transformation of the decoder output) to fit our regression task.

To summarize, the encoder-decoder LSTM model, parameterized by a set of parameters θ (i.e., all the unknown parameters of the model), learns a nonlinear function that predicts the current output value $\mathbf{y_{t+k}}$ as:

$$\mathbf{y}_{t+k} = F_\theta(\mathbf{y}_{t+1}, \mathbf{y}_{t+2}, \cdots, \mathbf{y}_{t+k-1}, \mathbf{x}_{t-T}, \cdots, \mathbf{x}_{t-1}, \mathbf{x}_t), \tag{9}$$

with the encoder providing the summary of the input trajectory $\{\mathbf{x}_{t-k'}\}_{k'=0}^{T}$ through the cell and hidden states $\mathbf{c}_t$ and $\mathbf{h}_t$.

The two components of the LSTM encoder-decoder are trained on a dataset $D = \{\mathbf{x}^j, \mathbf{y}^j\}_{j=1}^{n}$ where n is the number of labeled vehicle sequences, to minimize the mean squared error (MSE) loss function. For training, the Adam optimization algorithm [19] is used. The actual implementation of the presented LSTM encoder-decoder model is available online and can be found in https://github.com/krish-din/C-AVOID.

C. Trajectory Prediction Problem Formulation

In our work, each $\mathbf{x}_{t-k'} \in \mathbb{R}^d$ in sequence $\mathbf{x}$ consists of the following set of observations:

$$\mathbf{x}_{t-k'} = \{v_{t-k'}, \chi_{t-k'}, \psi_{t-k'}, r_{t-k'}, s^1_{t-k'}, ..., s^l_{t-k'}\}, \tag{10}$$

where:

- $v_{t-k'}$ is the velocity of the vehicle;
- $\chi_{t-k'}$ is the longitudinal position of the vehicle;
- $\psi_{t-k'}$ is the lateral position of the vehicle;
- $r_{t-k'}$ is the movement angle of the vehicle;
- $s^i_{t-k'}$ is the status of the i-th traffic light at the intersection, where $i = 1, .., l$. Note that number of traffic lights l, depends on the structure of the intersection.

For the output trajectories, each $\mathbf{y}_{t+k} \in \mathbb{R}^2$ in predicted sequence $\mathbf{y}$ consists of the following observations:

$$\mathbf{y}_{t+k} = \{\chi_{t+k}, \psi_{t+k}\}, \tag{11}$$

where χ_{t+k} and ψ_{t+k} are the longitudinal and lateral position coordinates of the vehicle, respectively.

It is worth mentioning that additional input features (i.e., acceleration, braking, and vehicle lane) were also tested during model training and inference, without however contributing to further improvements on model accuracy. The consideration of additional features, e.g., the steering angle, is also possible. However, such features were not available in the datasets used for performance evaluation. Hence, in the rest of the paper, only the aforementioned input features are considered. Even though our model achieves an excellent accuracy, as shown in the following section, a further analysis of input features naturally present in CAM messages or at the infrastructure constitutes an interesting future research direction.

V. Performance Evaluation

A. Data Collection and Pre-processing

We detail the characteristics of both the SUMO and the real-world dataset below, and we describe how they have been used to train and test the LSTM network.

SUMO dataset: We first consider the mobility traces of the Luxembourg SUMO Traffic (LuST) Scenario [8] and simulate them through SUMO. The TraCI SUMO library is used to extract the vehicles' movement at an intersection at peak hour, i.e., 6-11 am, to create the SUMO dataset. The size of the area monitored around the intersection is 500x250 m^2 and all vehicles traveling across such area are included in the SUMO dataset. For each vehicle, TraCI allows obtaining all features mentioned in Sec. IV-C, including traffic light information. Each traffic light is described by a ternary variable (green, orange, red), and we considered all 16 traffic lights that are present at the intersection. The final SUMO dataset contains the information of $6,236$ vehicles, out of which $2,042$ turn at the intersection, while $4,194$ proceed straight.

Real-world (RW) dataset: This dataset, includes trajectories of multiple vehicles extracted from the videos captured by a drone flying over an intersection of a mid-sized city, i.e., Nicosia, Cyprus, at peak hour, where the area containing the intersection in the video footage is 250x150 m^2. Over a span of 80 minutes, the trajectories of $5,105$ vehicles are captured, with $2,699$ vehicles traveling straight and $2,406$ vehicles turning. Unlike other state-of-the-art datasets, apart from the vehicles' longitudinal and lateral locations, this dataset is enriched with additional features through post-processing. Specifically, using a Kalman filtering technique, vehicles are tracked in the video footage and their velocity is computed by calculating the approximated displacement over the last 25 frames (see [20] for additional details). Furthermore, to account for the data provided by the road infrastructure, traffic light phases are also obtained from the video footage. Specifically, there are 6 traffic lights in operation at the intersection under consideration, having three possible states, namely red, green, and orange, with the latter stored as a "TimeToGreen" feature. The final enhanced real-word dataset is available online at [21].

In both datasets, as in current V2I and I2I communications, vehicles and road infrastructure information are sampled every $\tau = 100$ ms. Furthermore, features have values across different magnitudes, range, and units. Therefore, we have used a feature scaling technique, namely, standardization, to scale the input features (except the traffic light phase), with zero mean and unit standard deviation. For prediction, input data of the last 3 s are considered, i.e., $T = 30$, with predictions

performed at each time instant with a slide window approach, i.e., consecutive predictions share 29 input data samples. The presented LSTM network is designed to predict the future vehicles' location in the next 3 s, i.e., $L = 30$. With these parameters, 3.7M and 2M sequences are obtained for the SUMO and real-world (drone-based) datasets, respectively.

For training and testing the encoder-decoder LSTM model, the dataset is partitioned into training and test datasets, with the training dataset used to optimize the unknown parameters θ, and the test dataset used to evaluate the performance accuracy of the model. Specifically, the dataset is split in such a way that 65% of the vehicle trajectories are randomly selected for training, 15% for validation, and 20% for testing.

B. Performance Results - SUMO Dataset

To assess the additional value attained by infrastructure-based information, we first tested the proposed approach using the SUMO dataset, with and without traffic light signal (TLS) features being considered. The system model architecture uses a single encoder and a single decoder layer, in both cases with $u = 128$ hidden units. The output of the decoder layer is connected to a Time-Distributed (TD) dense layer with a hidden layer of 128 units when TLS information is present, and with 64 units otherwise. The output of the TD dense layer has 2 units, to output the two features of interest, i.e., χ_{t+k} and ψ_{t+k}, for each time instant in the prediction window. Both model variations, with and without TLS information, are trained according to a batch size equal to 64 and with a learning rate of 0.0001. Training is stopped when the MSE is less than 0.0001 on the validation test, lasting 3 epochs, with and without TLS. The aforementioned configurations have been selected as they reached the lowest MSE after several hyperparameter tuning trials (e.g., on learning rate, units, etc.), that resulted in over 30 different configurations trained.

To evaluate the model accuracy, we compare the χ_{t+k} and ψ_{t+k} predictions against the true vehicle position, and we measure the Euclidean Distance (ED) between the predicted and the actual vehicle position. Table I compares the performance of both models, i.e., with and without TLS information considered. Results show that infrastructure-based features play a crucial role, as they reduce the prediction error on average by 8.3%, leading to a performance improvement, especially when the prediction horizon increases. This improvement is consistent both in case of vehicles traveling straight or vehicles turning at the intersection. Table I also presents the MSE of the prediction error, i.e., the actual function minimized during training with our Encoder-Decoder LSTM approach, with and without TLS. Even in terms of MSE, including TLS information improves consistently the achieved performance (of up to 14.85%). Finally, Table I also compares the results achieved with several state-of-the-art approaches. First, several variations of LSTM have been tested on the SUMO dataset without TLS information. Specifically, Vanilla, Stacked Encoder-Decoder, and Bi-Directional LSTMs were tested (see [22] for more details on these approaches). Results show that in all prediction windows and both for vehicles turning or proceeding straight at the intersection, our selected LSTM model performs the best. Only the Stacked Encoder-Decoder LSTM approach performs similarly to our approach, albeit requiring additional complexity. Second, as suggested in [7], the performance of Encoder-Decoder LSTM is compared with the constant speed Kalman Filter technique, which performs quite well in highways. The proposed comparison shows that, assuming little to no modifications of the vehicles' behavior at intersections, leads to very large errors both in short and long prediction windows, proving that intersections are a completely different use-case scenario.

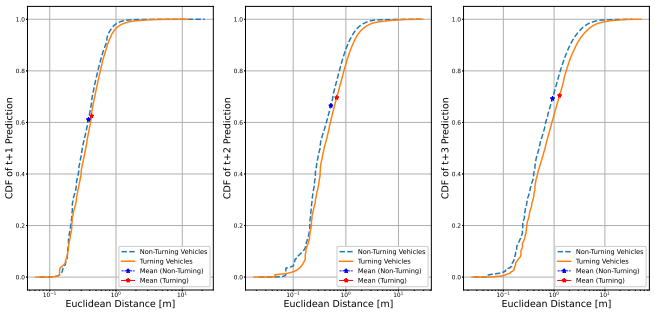

Fig. 4: Mean error CDF for predicted trajectories (SUMO dataset).

Figure 4 showcases a more comprehensive view of the achieved performance, presenting the distribution of the prediction error (in terms of the ED between the predicted and actual vehicle position) through its CDF. One can observe that, in the challenging intersection scenario, not only the proposed approach presents very small prediction errors on average, but also in distribution. Specifically, 97.3% of the 1-s look-ahead predictions exhibit an error of less than 1 m, 96.91% of the 2-s look-ahead predictions an error of less than 2 m, while 97.54% of the 3-s look-ahead predictions an error of less than 5 m.

The CDF of the error also gives us the possibility to identify critical cases where the prediction of a vehicle's future location is not as accurate. Indeed, we go one step further and examine the source of these cases – a fundamental step to further improve the performance of our approach. Analyzing the tail of the error's CDF, we can identify two main scenarios were prediction errors arise. The first one is accentuated by SUMO, since SUMO is an opportunistic simulator, that allows vehicles to perform lane changes any time the destination can be reached faster. Unfortunately, this simulator behavior causes rapid (unrealistic) vehicle position changes between lanes at the traffic light, thus increasing LSTM prediction error, especially for large prediction windows. Nevertheless, this is a SUMO implementation issue that does not impact the methodology presented in this work.

The second scenario, where vehicles traveling in the rightmost lane of the road can either travel straight or turn right at the intersection, represents the main challenge faced by the proposed approach (Fig. 5). In this case, our approach relies on other features besides location, e.g., velocity, to identify the vehicle's movement intention. These features, however, are not useful when the vehicle starts from a standstill position, as

TABLE I: Prediction error and MSE with and without traffic light info (SUMO dataset) and comparison with several other approaches.

	Prediction Time [s]	Euclidean Error [m]						MSE [m^2]	
		Enc. LSTM w/o TLS	Enc. LSTM with TLS	Vanilla LSTM	Encoder Stacked LSTM	Bidirect. LSTM	Constant Speed Kalman Filter	Enc. LSTM w/o TLS	Enc. LSTM with TLS
All vehicles	$t+1$	**0.381**	0.398	0.401	0.414	0.467	0.989	**0.138**	0.148
	$t+2$	0.617	**0.567**	0.714	0.641	0.778	2.921	0.549	**0.480**
	$t+3$	1.135	**1.041**	1.472	1.133	1.586	5.187	2.233	**1.901**
Non-turning Vehicles	$t+1$	**0.370**	0.384	0.379	0.403	0.428	0.927	**0.122**	0.130
	$t+2$	0.580	**0.519**	0.637	0.613	0.709	2.710	0.420	**0.339**
	$t+3$	1.031	**0.929**	1.315	1.044	1.449	4.773	1.713	**1.336**
Turning Vehicles	$t+1$	**0.405**	0.429	0.447	0.436	0.549	1.103	**0.173**	0.185
	$t+2$	0.695	**0.670**	0.882	0.703	0.928	3.312	0.829	**0.781**
	$t+3$	1.357	**1.281**	1.811	1.323	1.880	5.953	3.352	**3.116**

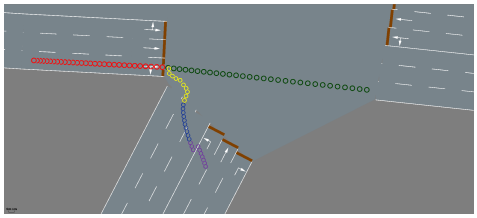

Fig. 5: SUMO dataset: Vehicle traveling in the right lane is wrongly predicted to turn. Past observations (red), actual trajectory (green), predicted trajectory at $t+1$ (yellow), $t+2$ (blue), and $t+3$ (purple).

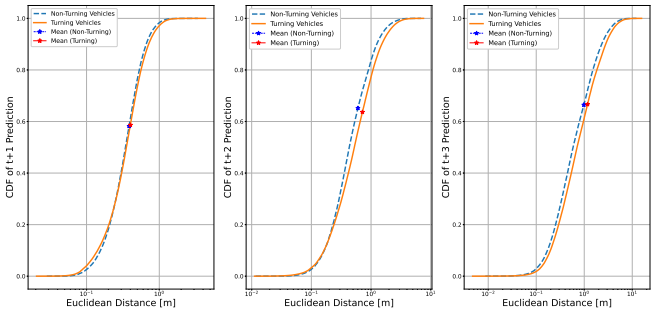

Fig. 6: Mean error CDF for predicted trajectories (RW dataset).

Fig. 7: Vehicle accelerates from a standstill position (RW dataset).

it often happens at a regulated intersection when the vehicle stops at the traffic light. Furthermore, in our SUMO dataset, vehicles having a green traffic light approach the intersection with very similar velocity values regardless of whether they plan to turn or proceed straight, i.e., 11.49 m/s and 12.46 m/s, respectively, up to 0.5 seconds before the intersection. Only when a vehicle is just about to enter the intersection, our approach is able to predict the vehicle's intention. Before that, the 3-s prediction of the proposed approach always matches the most common behavior of the vehicles traveling on that

TABLE II: Euclidean prediction error with and without traffic light information (real-world dataset)

	Prediction Time [s]	Euclidean Error [m]	
		w/o TLS	with TLS
All vehicles	$t+1$	0.496	0.394
	$t+2$	0.851	0.669
	$t+3$	1.299	1.092
Non-turning Vehicles	$t+1$	0.444	0.384
	$t+2$	0.749	0.603
	$t+3$	1.197	0.982
Turning Vehicles	$t+1$	0.534	0.401
	$t+2$	0.925	0.717
	$t+3$	1.372	1.171

specific lane, thus generating large errors for the small subset of vehicles that follow a different trajectory on that lane. Note that, even if such error is not entirely dependent on the simulator, SUMO's simulated mobility patterns (with vehicles barely braking before turning) play a crucial role in accentuating the obtained prediction errors. Thus, a second, real traffic dataset, was also used to better evaluate the performance of the proposed technique under real traffic conditions and to overcome the limitations the incur due to the simulation tool.

C. Performance Results - Drone-based Real-world Dataset

For the drone-based real-world dataset, the best model configuration (i.e., best hyperparameters) utilizes a single layer for the encoder and a single layer for the decoder, with 128 hidden units in the LSTM cell for both cases. This time, with and without traffic light signal information, the hidden layer of the TD dense layer is set to 64 units. As with the SUMO dataset, the output of the dense layer includes 2 units, one for each output feature. The model is trained according to a batch size equal to 64 and a learning rate equal to 0.0001, with training lasting 3 epochs, as for the SUMO dataset.

Table II presents the average ED between predicted and real vehicle locations, at $t+1$, $t+2$, and $t+3$, with and without TLS information. As with the SUMO dataset, when the real-world dataset exploits traffic light information, the trajectory prediction accuracy improves. Specifically, for longer prediction windows, i.e., 3 s, the prediction accuracy improves by 15.93%, Interestingly, even though the real-world dataset presents additional variability, due to human unpredictability, the average errors are very similar to those obtained with the SUMO dataset. This is due to the fact that, in the real-world dataset, the error amplifications observed in

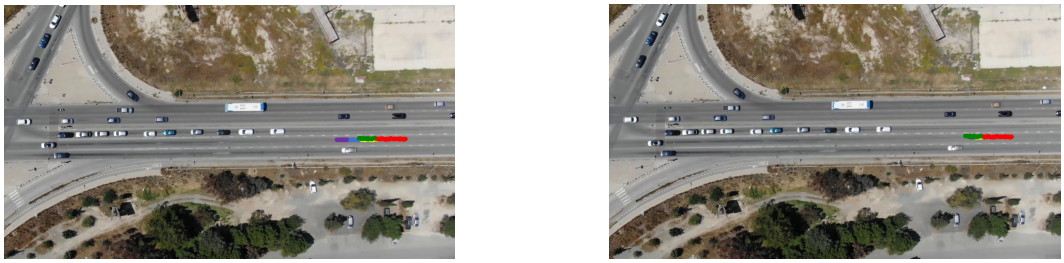

Fig. 8: Real-world dataset: Trajectory prediction when the vehicle approaches a standstill state without (left) and with (right) TLS information. Past observations (red), actual trajectory (green), predicted trajectory at $t+1$ (yellow), $t+2$ (blue), and $t+3$ (purple).

the SUMO dataset are not present now, compensating for the human unpredictability factor. Remarkably, the model is able to predict the future vehicle location with very good accuracy. In fact, the obtained prediction errors are better, by a large margin, compared to existing LSTM-based approaches used for trajectory predictions on real-world data in less challenging scenarios such as highways (i.e., [5], [23] present errors larger than 3 m, on average, for a 3-s look-ahead prediction).

Figure 6 presents the CDF of the mean error obtained through our approach for the 1-s, 2-s, 3-s look-ahead predictions. From the results obtained, it is clear that the proposed approach performs similarly in distribution as well for both datasets. Specifically, 97.72% of the 1-s look-ahead predictions exhibit an error of less than 1 m, 96.13% of the 2-s look-ahead predictions an error of less than 2 m, while 98.45% of the 3-s look-ahead predictions an error of less than 5 m.

Exploiting the CDF, the characterization of the larger prediction errors is again possible. The biggest challenge for the proposed approach, as depicted in Fig. 7, is to correctly predict the moment of the vehicle's first movement when queued at the traffic light. Evidently, the larger the distance between the vehicle and the traffic light the larger the error, as in this case the correlation between the vehicle's initial acceleration from the standstill position and the traffic light state is minimal.

Nevertheless, there is a common scenario where the traffic light information improves the trajectory prediction. By adding traffic light information, the errors pertaining to vehicles approaching a standstill state are reduced substantially, as shown in Fig. 8. Indeed, when a vehicle approaches a red traffic light, the proposed solution can accurately predict the time instant at which the vehicle starts braking.

VI. Conclusions and Future Work

We addressed trajectory prediction in scenarios including road intersections, and presented a novel framework that effectively leverages an IM hosted at a 5G-MEC platform, as well as V2I and I2I communications to collect and process the relevant data. Due to its advantageous location, the IM can obtain a holistic view of the system and, exploiting LSTM RNNs, it can provide accurate trajectory predictions in such complex scenarios as urban intersections. Unlike the existing approaches, our solution jointly leverages data collected at the vehicles and data collected through the smart city infrastructure, namely, traffic light phases. Our results, obtained using both a simulated SUMO dataset and a real-world dataset, demonstrate that the novel proposed methodology can be effectively used for accurate vehicle trajectory prediction.

Future research will exploit additional data that can be collected at the IM, e.g., information on the vehicles surrounding the ego vehicle, to further improve accuracy.

References

[1] W. H. Organization *et al.*, "Global status report on road safety 2018: Summary," World Health Organization, Tech. Rep., 2018.

[2] P. Bansal and K. M. Kockelman, "Forecasting Americans' long-term adoption of connected and autonomous vehicle technologies," *Transportation Research Part A*, vol. 95, pp. 49–63, 2017.

[3] Q. Tran and J. Firl, "Online maneuver recognition and multimodal trajectory prediction for intersection assistance using non-parametric regression," in *Proc. IEEE IV*, 2014.

[4] J. F. Kooij *et al.*, "Context-based path prediction for targets with switching dynamics," *Int.l J. of Computer Vision*, vol. 127, no. 3, 2019.

[5] F. Altché and A. de La Fortelle, "An LSTM network for highway trajectory prediction," in *Proc. IEEE ITSC*, 2017, pp. 353–359.

[6] S. H. Park *et al.*, "Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder-decoder architecture," in *Proc. IEEE IV*, 2018.

[7] N. Deo and M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," in *Proc. IEEE CVPR Workshops*, 2018.

[8] L. Codecá *et al.*, "Luxembourg SUMO traffic (LuST) scenario: Traffic demand evaluation," *IEEE Intell. Transp. Syst. Mag.*, vol. 9, no. 2, 2017.

[9] A. Rudenko *et al.*, "Human motion trajectory prediction: A survey," *The International Journal of Robotics Research*, vol. 39, pp. 895–935, 2020.

[10] D. J. Phillips *et al.*, "Generalizable intention prediction of human drivers at intersections," in *Proc. IEEE IV*, 2017.

[11] B. Kim *et al.*, "Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network," in *Proc. IEEE ITSC*, 2017.

[12] N. Djuric *et al.*, "Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving," in *Proc. IEEE/CVF Winter Conf. on Applications of Computer Vision*, 2020, pp. 2095–2104.

[13] N. Deo and M. Trivedi, "Multi-modal trajectory prediction of surrounding vehicles with maneuver based LSTMs," in *Proc. IEEE IV*, 2018.

[14] N. Lee *et al.*, "DESIRE: Distant future prediction in dynamic scenes with interacting agents," in *Proc. IEEE CVPR*, 2017.

[15] J. Li *et al.*, "Coordination and trajectory prediction for vehicle interactions via Bayesian generative modeling," in *Proc. IEEE IV*, 2019.

[16] G. Avino *et al.*, "A MEC-based extended virtual sensing for automotive services," *IEEE Trans. Netw. Service Manag.*, vol. 16, 2019.

[17] Y. Bengio *et al.*, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Net.*, vol. 5, no. 2, 1994.

[18] K. Cho *et al.*, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *arXiv:1406.1078*, 2014.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv:1412.6980*, 2014.

[20] R. Makrigiorgis *et al.*, "Extracting the fundamental diagram from aerial footage," in *Proc. IEEE 91st Vehicular Techn. Conf.*, 2020, pp. 1–5.

[21] "Traffic light status (TLS) dataset," https://www2.kios.ucy.ac.cy/harpydata/tlsdataset/, 2021.

[22] R. Chandra *et al.*, "Evaluation of deep learning models for multi-step ahead time series prediction," *IEEE Access*, vol. 9, 2021.

[23] L. Xin *et al.*, "Intention-aware long horizon trajectory prediction of surrounding vehicles using dual LSTM networks," in *Proc. IEEE ITSC*, 2018.