

Received December 6, 2021, accepted December 29, 2021, date of publication December 31, 2021, date of current version January 7, 2022.

Digital Object Identifier 10.1109/ACCESS.2021.3139867

A Bidirectional Trajectory Prediction Model for Users in Mobile Networks

NASRIN BAHRA[✉] AND SAMUEL PIERRE, (Senior Member, IEEE)

Mobile Computing and Networking Research Laboratory (LARIM), Department of Computer and Software Engineering, Polytechnique Montréal, Montreal, QC H3T 1J4, Canada

Corresponding author: Nasrin Bahra (nasrin.bahra@polymtl.ca)

ABSTRACT Future mobile networks are envisioned to have critical limitations in terms of latency, energy usage, capacity and network resources since these networks are expected to become extremely dense and complex. The rapid enormous advances in recent technologies such as Internet of Things (IoT) highlights the urgent need for network performance enhancement as well. To this end, self-organizing networks are a promising solution to push the network performance to the next level. These scalable networks can dynamically adapt to possible changes in the network. Smart mobility management, in particular mobility prediction, is a subsection of self-organizing functions which are mainly based on the machine learning techniques. In this paper, we propose to estimate user's future trajectory using machine learning approaches for a better network management. We propose a novel bidirectional trajectory prediction model called BTM to model the user mobility behavior. The proposed method exploits the potential benefits of bidirectional gated recurrent unit (GRU) for having an accurate prediction. Moreover, we introduce a data preprocessing phase to obtain better results with significantly lower execution time. The proposed approach takes full advantage of data analysis in both directions (backward and forward) in order to provide a long-term prediction and model user's mobility even with complex patterns. Experimental results show that the proposed bidirectional approach significantly improves the performance of the mobility predictor in terms of model accuracy, robustness and execution time. It achieves a model error of 0.014 and decreases the execution time up to 97%.

INDEX TERMS Recurrent neural networks, trajectory prediction, mobile networks.

I. INTRODUCTION

Future mobile networks have critically limited available resources. Although, mobility adds a wide range of great services for users in the network, it can cause several critical issues at the same time. Bandwidth restriction, high-speed packet transmission, frequent handovers, increase in call dropping probability and communication reliability pose serious challenges for future network management which all lead to quality of service (QoS) degradation for users [1]. A deep understanding of network traffic behavior is of crucial importance in today's rapidly growing mobile networks.

Mobility prediction is a promising key enabler for the intelligent mobility management in self-organizing networks (SONs). SON is an emerging paradigm that is based on adaptive and independent network management. It is able to learn from past experiences and to improve the network

performance based on the user prior knowledge. Mobility management is a part of self-optimization function in self-organizing networks that can address the aforementioned issues and meet the demands of future mobile networks [2]. The main idea is to estimate user's movement trajectory using machine learning approaches for a better network management.

From another point of view, according to Cisco, there will be approximately 5.5 billion mobile users by the year 2020 [3]. The exponential growth in the number of mobile devices results in a huge mobility data generation by the mobile users every day. This data can be effectively exploited to extract valuable insight from them which later can be helpful in many ways.

Some of the potential research areas that can significantly benefit from location awareness are: resource allocation techniques, optimized handover decision, call admission control optimization, bandwidth reservation, routing mechanisms and provision of high QoS. In particular, when a mobile node

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Maaz Rehan[✉].

moves between different access points with an active session, seamless connectivity without interruption is essential while user is moving. Location awareness can make handover process transparent to mobile users in dense networks and effectively eliminate the problem of QoS degradation [4]. Moreover, location information serves useful purposes for 5G networks in terms of energy consumption and latency reduction [5]. Massive increase in the number of mobile devices and connected objects stresses the needs for lower latency and high data rates.

Despite the efforts that have been made in the existing works, there are still several research aspects regarding user trajectory prediction that can be further improved such as obtaining a long-term reliable prediction, achieving high prediction accuracy dealing with complex and irregular patterns, low time complexity, preventing excessive assumptions for the network or model, balancing the trade-off between the model complexity and accuracy. Most of the related works in this field are based on Markov models and neural networks. In Markov-based approaches, first order Markov models are simple with low accuracy, whereas Markov models with higher orders are complex with acceptable accuracy that rises the problem of accuracy-complexity trade-off. In traditional neural networks, input samples are assumed to be independent of each other which is a questionable assumption. As a result, they cannot learn long-term dependencies and process sequential data. This can have disruptive impact on prediction accuracy. We believe that a mobility model is needed that can take advantage of numerous merits of neural networks as well as eliminating the memory issue. More precisely, the model should be able to learn from the available prior knowledge and should be able to adopt to probable changes without manual effort. The main challenge is to introduce a novel trajectory prediction technique that emphasizes the need for a generic approach for all users with any degree of predictability.

In this paper, to address the shortcomings of the existing methods, we propose a mobility model that can effectively maximize the impact of data dependency of prior knowledge about user's past experiences on the mobility model. Here, we propose a bidirectional trajectory prediction model called BTPM based on bidirectional gated recurrent unit (GRU) [6], [7]. In BTPM, first, we eliminate the unwanted raw data using Douglas-Peucker (D-P) algorithm [8]. Then, noisy data is discarded from user trajectory using Kalman Filter. Finally, we feed the data to the inference model which is based on bidirectional gated recurrent unit (GRU). The major contributions of our work are summarized in the following:

- Proposing a two-phase user mobility sequence preparation specifically introduced for mobility prediction purpose. To the best of our knowledge, this is the first time that this specific user trajectory preparation is introduced and deployed for trajectory prediction. The key feature of this procedure is huge time complexity reduction while improving the prediction accuracy and reducing the noise level. Kalman filter and line

simplification algorithms are deployed to reduce the noise and obtain significantly lower execution time. One major problem of using deep learning methods is high execution time dealing with large datasets. Therefore, BTPM benefits from numerous advantages of deep learning while eliminating the complexity issue. This phase results in reducing the execution time up to 97%.

- Proposing a bidirectional approach to extract mobility patterns by deeply considering the correlation in the user mobility trajectory. To the best of our knowledge, this is the first time that a bidirectional neural network is deployed to investigate the user trajectory prediction. The proposed bidirectional trajectory prediction model (BTPM) is based on bidirectional gated recurrent unit (BiGRU) that enables highly accurate predictions with analyzing both forward and backward correlations in the user past movement history. Another distinguishing feature of BTPM is achieving a robust performance dealing with different users with different degrees of predictability. It improves the model error up to 65%.
- Conducting a comprehensive series of experiments using three datasets and different users with different metrics. User mobility behavior and length of available data for each user are two factors that are inevitable for mobile users in reality. We considered the combination of these factors to evaluate different mobility models in terms of being practical in more close to real-world scenarios.

The rest of this paper is organized as follows. In Section II, we overview some of the related works in this field and provide a list of unsolved open problems. A preliminary to mobility prediction based on neural networks is provided in Section III. The proposed method for user trajectory prediction is presented in Section IV. Section V provides the experiment results. Section VI concludes this paper.

II. RELATED WORKS

Trajectory data mining and mobility prediction have been extensively studied and there are a lot of wide-ranging works about them in literature [1], [3], [9]–[12]. In general, almost all the methods in the mobility prediction concept can fall into two main categories [13]: (1) User future path/trajectory prediction: It is a regression task that predicts the geographical location of the user in the next time step. The prediction is the next geographical location point of the user (e.g., longitude and latitude). One popular example is user trajectory prediction (e.g., (lon1, lat1), (lon2, lat2), (lon3, lat3)) [14], [15]. (2) User next location prediction: It is a classification task that predicts the place that user will visit in future. In the dataset, each location has an specific location id. The prediction is the next location id of the user. One popular example of its application is point-of-interest recommendation methods (e.g., restaurant, park, gas station, . . .) [16]–[18]. Our work is a part of the first category. In the following, we review some of the recent works of both groups.

There are several works on user trajectory analysis. In [19], authors proposed a semantic model that converts raw movement history to semantic trajectories. In [20], authors proposed a user movement prediction method considering user geographic, temporal and semantic intentions. In [21], authors proposed an approach for trajectory representation learning using road networks and mobile user trajectories. A trajectory prediction model was proposed in [22]. Using similarity metrics, it predicts user's trajectory based on finding similar informative segments in available initial user trajectory.

Learning-based mobility prediction approaches have attracted a tremendous attention in the recent years due to the popularity of machine learning techniques. A mobility prediction method was proposed in [23], it predicts users trajectory based on an online learning process using mobility data history. The main objective of this work is to decrease handover-related signaling load and to reduce the latency using a predictive handover scheme. In [24], a long-term mobility estimation scheme was introduced to extract the mobility patterns based on previous trajectories. Trajectories were clustered based on the trajectory similarities for the offline learning process. The main goal of this work was to estimate a group of possible future paths for each individual in the network. In [25], a supervised learning method was used for location prediction since no data labeling is required when using check-in databases. One major problem of methods based on the standard neural networks and machine learning techniques is that all of them rest on a questionable assumption of independency between data input samples. They do not consider the data correlation in the sequential data.

In recent years, some mobility models were proposed based on recurrent neural networks (RNNs) which are a subsection of deep learning approaches. In [26], an RNN-based method is proposed: spatial temporal recurrent neural networks (ST-RNN). This model considers temporal and spatial dependencies in each layer using time-specific and distance-specific transition matrices. Furthermore, a linear interpolation method was used for training the transition matrices. Inspired by ST-RNN, authors in [27] and [28] proposed approaches for destination prediction based on long short-term memory (LSTM) models using spatial-temporal information in user's movement history. In [29], a learning-based model were proposed using RNN, LSTM and GRU models as mobility predictors using a preprocessed mobility data. In [30], given the history of call detail records of each user, authors proposed an LSTM-based model that predicts the next location of the user based on the cell tower IDs. Mobility models based on RNNs and LSTMs consider the information regarding to the past time steps and their correlations only up to the current step. However, if the mobility model could make the prediction based on the whole input sequence (past and future), it can significantly improve the performance. All of these works analyzed the sequence of locations in only forward path.

There are many works that were proposed different mobility models motivated by location-based social networks (LBSNs). A huge amount of check-in data is collected from LBSN (e.g., Foursquare) that can be exploited in mobility pattern learning and location-based services. In [31], authors proposed an RNN-based deep context-aware model that has three main parts: (1) The first part captures the long-term dependency in user mobility records, (2) The second part extracts user's location semantics and periodic pattern, and (3) The third part extracts social and temporal context. Using this deep model, they could predict the next point-of-interest (POI) of the user. The authors in [32] and [33] introduced two GRU-based models for the next POI prediction. Using LBSN datasets, a context-aware trajectory learning method was proposed in [34] to capture different characteristics of human mobility behavior.

A huge part of localization techniques and mobility predictors are based on different types of Markov models. The movement path of each user is modelled based on his/her record of visited cells or geographic information and Markov predictor is deployed in order to predict the future cells [35]. There is an important trade-off in Markov models that needs to be given a careful consideration. While higher orders of Markov models result in a better prediction accuracy, a huge number of model parameters are needed in these models. For an M th-order Markov chain with d states, $d^{M-1}(d - 1)$ parameters are needed. In [36], a Markov-based trajectory prediction technique called hidden Markov model-based Trajectory Prediction (HMTP) was proposed that can anticipate a complete path. Algorithm parameters are selected dynamically and a partitioning algorithm was introduced.

A location prediction system using GPS data was proposed in [37]. This approach contains three main parts: location extraction, location recognition and location prediction. Gaussian-means, k-nearest neighbor and HMM methods were deployed in these parts. Another Markov-based method was proposed in [38]. This approach consists of two main parts: destination prediction model (DPM) and path prediction model (PPM). DPM is responsible for estimating the final destination using second-order Markov chain. Then, the path toward the user's destination is anticipated by PPM.

In a traditional Markov model, states must be chosen from a large state space imposing restriction on the inference model. Hence, standard operations get less feasible in a hidden Markov model as hidden states increase. Therefore, each hidden state only depends on a few number of past states. It seems that Markov models tend to fail when the number of locations increases and can be more accurate when dealing with short-term dependencies with few number of locations. Past researches have shown that Markov models fall short of modelling human mobility [39]. Furthermore, Markov models are too sensitive and can easily be affected by user mobility pattern changes. They struggle to cope well with drastic changes in mobility behavior of users.

III. PRELIMINARY

In this section, we provide a preliminary to mobility prediction models based on neural networks. Traditional neural networks are not able to store information while processing new inputs. Therefore, input samples are assumed to be independent of each other which is a questionable assumption for applications in which inputs are not independent of each other and this dependency has an impact on the learning process.

Recurrent neural networks are a special kind of neural networks that can capture the dependency between input samples and therefore are suitable for sequential data. RNNs solve the mentioned problem of traditional neural networks since they include loops in the network and have a memory to store the information about past locations.

For an RNN at each time step t with $L(t)$ (i.e., user location at time step t) as input, $y(t)$ as prediction output and $h(t)$ as hidden states, the forward propagation equations are given in the following. Hidden layer activation function is:

$$h(t) = \tanh(b_1 + Wh(t-1) + UL(t)). \quad (1)$$

A softmax operation is used to obtain the output vector such that

$$y(t) = \text{softmax}(b_2 + Vh(t)), \quad (2)$$

where b_1 and b_2 are bias vectors. Also, U , V and W are weight matrices that decide to what extend each input should influence the location prediction [40].

However, RNN has the fundamental problem of vanishing or exploding gradients. The problem of exploding or vanishing gradients mostly happens when the network is large and has many layers. Simple RNNs are not able to remember very old information from the beginning of the sequence. In other words, RNN requires a very long time to learn long-term dependencies. Hence, this mobility model is not suitable for capturing users movement pattern from old states information.

The idea of gated RNN is a potential solution for the aforementioned problem that has been proved to be highly successful. The core idea is based on the fact that we do not necessarily need all the past information over a long period. In other words, for some states, it could be even more effective if network forget the previous states and eliminates the old information for making future decisions. When the network is able to learn which states to keep and which ones to forget, we have a gated RNN. Two popular forms of gated RNNs are long short-term memory and gated recurrent unit [7], [41].

For LSTM, the key difference is adding a *cell state* unit $s(t)$ to the RNN structure. Cell state keeps a record of current and previous user's locations information. This information can be regulated by a *forget gate* unit $f(t)$ in each LSTM cell. Therefore, LSTMs can effectively capture the long-term dependencies and are an appropriate model for processing time series sequences.

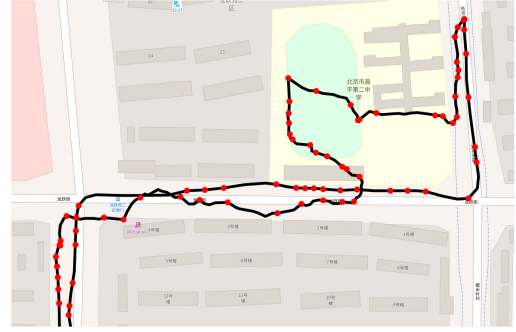


FIGURE 1. User raw GPS trajectory. Red points represent the necessary points that forms the path.

The next gated RNN is GRU, a simplified version of LSTM network, that has two main gates: *update gate* $u(t)$ and *reset gate* $r(t)$. The update equations for GRU is expressed as follows:

$$h(t) = u(t-1)h(t-1) + (1 - u(t-1))\sigma(b + UL(t-1) + Wr(t-1)h(t-1)), \quad (3)$$

where $u(t)$ and $r(t)$ are defined by

$$u(t) = \sigma(b_u + U_u L(t) + W_u h(t)), \quad (4)$$

$$r(t) = \sigma(b_r + U_r L(t) + W_r h(t)), \quad (5)$$

where σ denotes Sigmoid function that returns a value between 0 and 1. W_r and W_u are recurrent weight matrices. The update gate decides which part of the data to keep for future time steps. Sigmoid function of the reset gate determines what information to remove from the data. Mathematical notations for RNN and GRU are summarized in Table 1.

IV. BIDIRECTIONAL TRAJECTORY PREDICTION MODEL (BTPM)

In this section, we explain the proposed approach for predicting the user future trajectory.

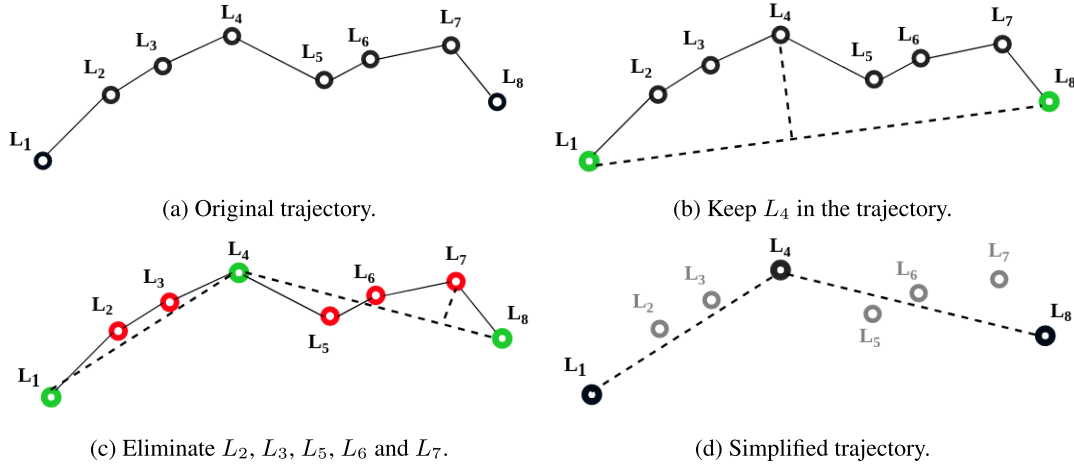
Problem(Trajectory Prediction): Given a trajectory of a user for the previous n time steps (T_{user}), we want to predict the next n time steps ($\tilde{T}_{user}$). Where $T_{user} = (L_{t-n}, \dots, L_{t-1}, L_t)$ and $\tilde{T}_{user} = (L_{t+1}, \dots, L_{t+n})$.

The primary step of the mobility learning is choosing an appropriate type of mobility data as the movement history of a user. One challenging part of deploying user movement history is the fact that this information needs preprocessing before it can be used for the mobility model. It is crucial to convert the raw data (i.e., user movement history) into a clean dataset for the analysis. In fact, with the right data, even simple methods are able to provide valuable insights from the data. We choose GPS trajectories as the user movement history.

Trajectory Definition: A trajectory for the user i is represented as $T_{useri} = (L_{t-n}, \dots, L_{t-2}, L_{t-1}, L_t)$ which is a sequence of time-stamped location points. We denote L_t as $L_t = (x_t, y_t)$ where x_t and y_t provide the geographical

TABLE 1. Mathematical notations for RNN and GRU.

$L(t)$	User location	$y(t)$	Prediction output
b_1, b_2, b_u, b_r, b	Bias vectors	$h(t)$	Hidden states
U, U_u, U_r	Input weight	$u(t)$	Update gate
V, W, W_u, W_r	Recurrent weight	$r(t)$	Reset gate

**FIGURE 2.** Trajectory simplification.

information of the user location at time t (i.e., longitude and latitude).

We carefully analyzed the input data of our mobility model and try to deal with the existing underlying issues related to the raw datasets that can have damaging impacts on the accuracy of the mobility learning. We realized there are two fundamental issues that need to be solved before training and inference phases: redundancy and noisiness. Raw GPS dataset is composed of user location information at almost each second with a high sample rate. All of this data is not necessarily needed for our purpose. We propose to deploy a line simplification method to reduce the irrelevant data from user raw trajectory and only keep the appropriate data for the mobility learning model from a big dataset. The core concept of line simplification approaches is to keep only the part of data that is essential to form the trajectory and delete the rest of the data points between them. This data reduction has the potential to substantially lower the processing time. Therefore, in BTPM, the first step is to remove the irrelevant data from the user trajectory. We deploy D-P algorithm to eliminate the unwanted data. Figure 1 depicts a mobile user GPS trajectory. The black line represents the actual path of the user and the raw information of user trajectory. Red points on the line are the points that we keep from the dataset and remove the rest. In this way, we have a similar trajectory with much fewer number of points.

Given the trajectory of a user with the length of m [$L_1 \rightarrow L_2 \rightarrow L_3 \rightarrow \dots \rightarrow L_m$], we want to convert

it to a simplified trajectory with the length of n [$L_1 \rightarrow L_2 \rightarrow L_3 \rightarrow \dots \rightarrow L_n$], where $n < m$.

Data reduction procedure has a global routine in which the whole user trajectory is considered during the data reduction procedure. Figure 2 represents the main steps for choosing the determinant location points to keep from the original user trajectory. Figure 2(a) shows an illustration of a user original trajectory, where $L_i = (lon_i, lat_i)$, $i \in \mathbb{N}$ denotes user locations in the trajectory in terms of longitude and latitude. This example considers 8 locations (i.e., $i = 8$). In Figure 2(b), we draw a line between the first and last points (i.e., L_1, L_8) and select the farthest location point from the line (i.e., L_4). We keep this location in the trace since its distance from the line is more than a predefined threshold. Similarly, in Figure 2(c), we eliminate L_2, L_3, L_5, L_6 and L_7 from the trajectory since their distances are lower than the predefined threshold. Figure 2(d) shows the final simplified trajectory. This procedure is repeated for all the locations in the trajectory. It will result in almost the same performance accuracy but with a remarkably lower execution time.

In this work, we deal with data produced by real users, so this data reflects the noisy reality as an inevitable consequence. It is vital to deal with this noise and remove it from data in the preprocessing stage. Therefore, as a subsequent step, we exploit Kalman filter to reduce the noise in the user data that commonly happens during GPS measurement process [42], [43]. This technique is composed of a prediction process followed by an update

(or correction). Firstly, Kalman filter estimates the next state of the process. Then, it updates the estimation based on the noisy observations (i.e., measurements). These updates are performed in two distinct categories: time update and measurement update. Mathematical equations for updating time (also known as prediction stage) are used in order to update the process time step and calculating a prior estimation of the next step. Measurement updates (also called as correction stage) are in charge of providing feedback based on the observations that we already have and improve the estimation. Given some noisy information, Kalman filter tries to decrease the level of uncertainty or error in the data. The state equation of such process is

$$x_k = Bx_{k-1} + Cu_{k-1} + \alpha_{k-1}, \quad (6)$$

and the measurement equation is

$$M_k = Hx_k + \beta_k, \quad (7)$$

where x_k and M_k are state and measurements vectors at time step k . u_k is the control input vector. Matrix B shows the relationship between two consecutive time steps and Matrix C indicates the relationship between the control input and the states. Measurement and process noises are respectively shown by β and α . Measurement matrix is represented by H .

Therefore, we smooth the GPS trajectory using Kalman filter. This effectively leads to a noticeable decrease in the noise level. Kalman filter has the potentials to predict the state of a process with the minimum error. Here, we provided a high-level explanation for Kalman filter algorithm. The details and corresponding equations for prediction-correction process in Kalman filter algorithm are given in Appendix. It is worth mentioning that we directly apply these techniques to the raw GPS dataset. Then, the obtained data after preprocessing phase is used as the input for the mobility model.

Concurrently, we generated a tailored data sequence of user past locations suited to our specific framework. This can have a profound impact on the final results, since the model is learning from the initial data. We feed the data as an input for the mobility model.

In the previous works, for each state $L(t)$ at time step t for a user, only the forward dependencies of information from past time steps until present $\{L(t-n), \dots, L(t-2), L(t-1)\}$ is considered for predicting the next step. However, BTM considers both the forward and backward dependencies of the user past movement history. It has both forward and backward paths starting from beginning and end of the sequence using the concept of bidirectional neural networks. When network can make a decision based on the whole input sequence (past and future), it would tremendously enhance the performance of the mobility model.

In the forward path, we investigate user's movement history in chronological order and basically deal with the past steps' information regarding user's previous location points (i.e., $\{L(t-n) \rightarrow L(t-n+1) \rightarrow L(t-n+2) \rightarrow \dots \rightarrow$

Algorithm 1 BTM Procedure

Require: User movement history (i.e., GPS data)

Ensure: Learning user mobility behavior pattern

- 1: Draw a line between the first and last user locations in the trajectory.
- 2: Select the farthest location point from the line.
- 3: If the distance of the selected location to the line is greater than a threshold, keep the point; otherwise, remove it from the trajectory.
- 4: Continue steps 1 to 3 recursively for all the location points in the user trajectory.
- 5: Kalman filter initialization

Time update - Prediction

- 6: Predict the next state based on the previous state (GPS position and velocity).
- 7: Predict error covariance.

Measurement update - Correction

- 8: Calculate the Kalman gain.
- 9: Update state estimation based on actual measurements.
- 10: Update error covariance.
- 11: Produce the smooth user trajectory.
- 12: Neural network initialization (weights and bias)

Forward direction

- 13: Complete forward pass for forward and backward hidden layers at each step.
- 14: Complete forward pass for output layer.

Backward direction

- 15: Complete backward pass for output layer.
- 16: Complete backward pass for forward and backward hidden layers.

$L(t-2) \rightarrow L(t-1)\}$). Hidden layer activation function for forward path is given by

$$\vec{h}(t) = \tanh(\vec{b}_1 + \vec{W} \vec{h}(t-1) + \vec{U} \vec{L}(t)). \quad (8)$$

We can add a backward path to the network and convert the unidirectional network into a bidirectional network. Therefore, we can take advantage of useful information from the future time steps in order to make an even more accurate decision. In the backward path, user's movement history is analyzed in reverse chronological order and user locations correlation is investigated in backward path (i.e., $\{L(t) \rightarrow L(t-1) \rightarrow L(t-2) \rightarrow \dots \rightarrow L(t-n)\}$). Hidden layer activation function for backward path is given by

$$\overleftarrow{h}(t) = \tanh(\overleftarrow{b}_1 + \overleftarrow{W} \overleftarrow{h}(t-1) + \overleftarrow{U} \overleftarrow{L}(t)), \quad (9)$$

and output vector is expressed as

$$y(t) = f(b_2 + V[\vec{h}(t), \overleftarrow{h}(t)]). \quad (10)$$

This model is able to deeply investigate user past locations correlations and controls which locations to keep and which ones to forget using update and reset gates. Figure 3 and Algorithm 1 show the proposed architecture in detail. First, we have a sequence of spatial time series data as input for

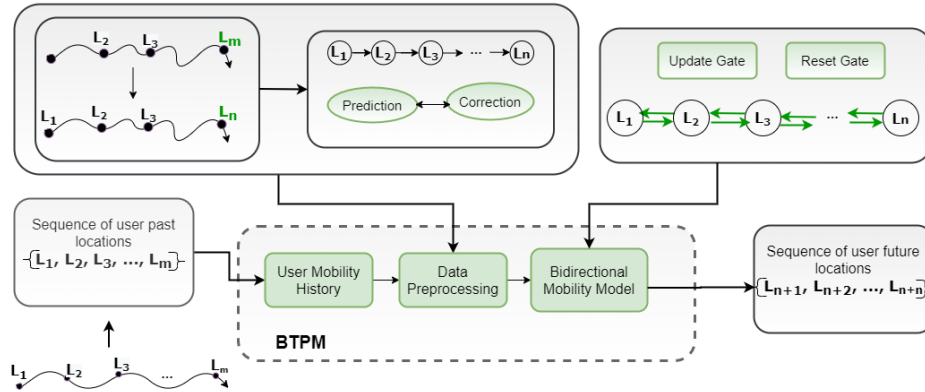


FIGURE 3. The proposed approach.

TABLE 2. User past locations dependencies analysis using different approaches.

Approach	Procedure
Mobility models based on Linear regression	$L_1, L_2, L_3, \dots, L_n \Rightarrow$ Future locations prediction
Mobility models based on RNN/LSTM/GRU	$L_1 \rightarrow L_2 \rightarrow L_3 \rightarrow \dots \rightarrow L_n \Rightarrow$ Future locations prediction
BTPM (proposed)	$L_1 \rightleftharpoons L_2 \rightleftharpoons L_3 \rightleftharpoons \dots \rightleftharpoons L_n \Rightarrow$ Future locations prediction

the model. Then, in the preprocessing stage, we apply some techniques to clean the data. The input data is desirably manipulated in order to reduce the number of unwanted observations ($n < m$). Next, the reduced version of the data passes through a Kalman filter to generate a clean data with the minimum level of noise. After data preparation, using bidirectional neural network, we train the model with BPM and extract user mobility behavior.

Table 2 presents the substantial difference between the proposed approach and the popular methods towards trajectory prediction. BPM gives us a distinct advantage of analyzing user mobility trajectory in both forward direction and backward direction. To be more specific, user's current location is highly dependent on the previous locations (forward direction) and also the next locations (backward direction) as well. The proposed model thoroughly explore the correlation between locations in both chronological and reverse orders. This can have a profound impact on the analysis and gives us a deeper understanding of the user mobility behavior with a significantly lower execution time.

Here, we want to provide an appropriate clarification on how exactly BPM works to make a prediction. For example, imagine we have a sequence containing the previous 50 geographical location points of a user (i.e., $n = 50$) and we want to predict the trajectory of the next 50 future locations of the user. After data preparation procedure, the following steps should be completed for training and inference phases: $L_1, L_2, L_3, \dots, L_{50} \rightarrow L_{51}, L_{52}, L_{53}, \dots, L_{100}$

Step 1: We take the sequence of location points as input samples and eliminate the redundant and noisy samples.

$$L_1, L_2, L_3, \dots, L_m \Rightarrow [] L_1, L_2, L_3, \dots, L_n \quad (n < m)$$

Step 2: We feed the information of the first to fiftieth locations ($n = 50$) to the network and obtain the information for the second to fifty-first locations in output layer.

$$[L_1 \rightleftharpoons L_2 \rightleftharpoons L_3 \rightleftharpoons \dots \rightleftharpoons L_{50}] \Rightarrow L_2, L_3, \dots, L_{50}, L_{51}$$

Step 3: Network takes the output of the last step as input (i.e., $L_2, L_3, \dots, L_{50}, L_{51}$) and we have the third to fifty-second locations as results.

$$[L_2 \rightleftharpoons L_3 \rightleftharpoons L_4 \rightleftharpoons \dots \rightleftharpoons L_{51}] \Rightarrow L_3, L_4, \dots, L_{51}, L_{52}$$

Step 4: Again, given the output of the last step, network predicts the forth to fifty-third locations.

$$[L_3 \rightleftharpoons L_4 \rightleftharpoons \dots \rightleftharpoons L_{51} \rightleftharpoons L_{52}] \Rightarrow L_4, L_5, \dots, L_{52}, L_{53}$$

Step 5: We continue the prediction re-iteratively as previous steps.

Step 6: The final step is to predict the trajectory of the 51th to 100th locations.

$$[L_{50} \rightleftharpoons L_{51} \rightleftharpoons \dots \rightleftharpoons L_{98} \rightleftharpoons L_{99}] \Rightarrow L_{51}, \dots, L_{99}, L_{100}$$

Therefore, for making a prediction, we investigate not only the forward locations correlations in the user movement

history but also the backward consecutive locations dependencies as well.

Next, we want to perform a complexity analysis of the proposed approach. For data reduction phase, worst case scenario time complexity is $O(n^2)$. For noise removal phase, computational complexity is $O(n^{2.376})$ [44]. Finally, for the learning phase, time complexity is $O(2)$. Therefore, the overall complexity of BTPM is $O(n^{2.376})$. We have conducted simulations to evaluate the time complexity of the proposed approach even with large input data in Section 4.4. (see Figure 11).

V. EXPERIMENTAL RESULTS

In this section, we present results of a series of experiments that have been carried out to evaluate the performance of the proposed approach and to highlight the effectiveness of the proposed model. To this end, first, we describe the data that has been used for the simulations in Section A. Then, we conduct the experiments in several phases including assessing the impact of data preprocessing, performance evaluation of trajectory prediction models, comparing the proposed approach with the existing techniques, robustness of the proposed method, and finally extending the results to the cellular networks.

For comparison, we implemented some of the most related previous methods including:

- Linear regression: A model based on linear regression is a basic method that take each user location independently [40].
- Support vector regression (SVR): A SVR-based model is based on machine learning techniques and considers each user location independently. It makes the prediction using kernels [45].
- Recurrent neural networks: RNN-based model considers user past locations to make a decision [46].
- Long short term memory: LSTM-based methods analyze past locations using state unit and forget gate [41].
- Gated recurrent unit: GRU-based methods predict future trajectory using update and reset gate to control user movement history [7].

We considered the same framework and configuration for implementing all the methods in order to have a fair comparison. Keras library was deployed for the simulations that is a neural network library in Python (<https://keras.io/>) and all the simulations were executed using Intel core i7-6700k CPU with 4.00 GHz and 32 GB RAM. A neural network with 3 layers (i.e., one hidden layer) and 100 neurons is used in each layer for all the mentioned methods. We set the drop-out rate to 0.2 (i.e., 20 percent of the connections are randomly dropped in the training phase). Also, the learning rate is set to 0.001 and Adam optimizer is used in the experiments.

Next, we choose the appropriate metrics for this work. Basically, our work is predicting user mobility trajectory. The proposed model for trajectory prediction is based on regression-based machine learning methods. Common

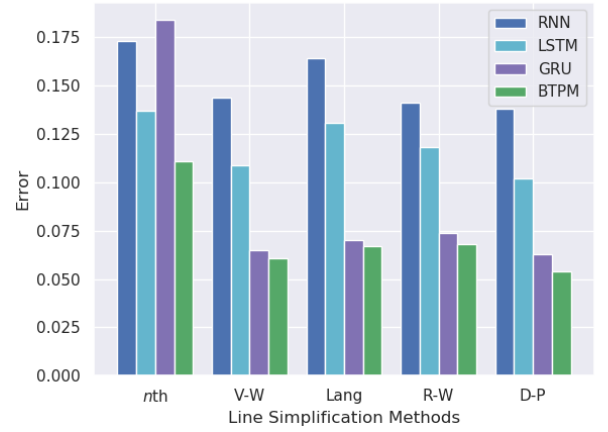


FIGURE 4. Trajectory simplification.

metrics to evaluate regression-based machine learning algorithms are mean square error (MSE), mean absolute error (MAE) and root mean square error (RMSE) [17], [47]. We used these three metrics to evaluate and compare the performances of all the methods. These metrics help us describe how effective each technique is. Given k samples from the total of n samples, MAE is expressed as:

$$MAE_k(\bar{y}) = 1/n \sum_{i=1}^k |y(x) - \bar{y}(x)|, \quad (11)$$

where x , $y(x)$ and $\bar{y}(x)$ as respectively model input, actual output and predicted output. MSE is defined by

$$MSE_k(\bar{y}) = 1/n \sum_{i=1}^k (y(x) - \bar{y}(x))^2, \quad (12)$$

and RMSE is given by

$$RMSE_k(\bar{y}) = \sqrt{1/n \sum_{i=1}^k (y(x) - \bar{y}(x))^2}. \quad (13)$$

Lower values for MAE, MSE and RMSE are desirable and ideally the best value for a perfect model is zero for these errors. Root Mean squared error is very sensitive to high errors. In other words, RMSE value noticeably increases when error variance is high (i.e., there is a major difference between the lowest and highest error value) in comparison with MSE and MAE values.

A. DATA DESCRIPTION

For conducting the experiments, three different GPS datasets were considered as user mobility data. Using different mobility data sources can guarantee the robustness and generality of a mobility model. An efficient approach should be able to appropriately predict user trajectory based on different mobility data with an acceptable accuracy. Information regarding the datasets are provided in the following:

- Dataset 1: Geolife [48]

This GPS dataset was collected by 182 persons during five years (from 2007 to 2012). This dataset contains

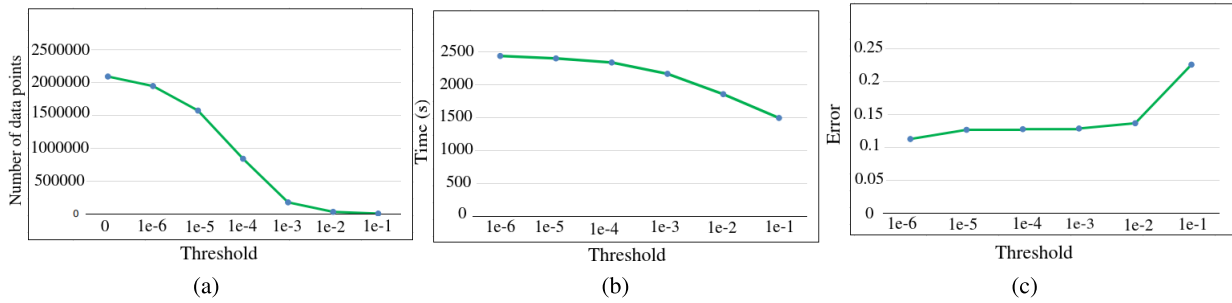


FIGURE 5. Impact of different thresholds on (a) data reduction, (b) execution time, and (c) model performance.

TABLE 3. Impact of changing RDP threshold.

RDP threshold	Reduction	Time(s)	MSE loss
$\varepsilon = 1e - 6$	2087853 $\rightarrow$ 1942782	2432	0.112
$\varepsilon = 1e - 5$	2087853 $\rightarrow$ 1570032	2395	0.126
$\varepsilon = 1e - 4$	2087853 $\rightarrow$ 837544	2333	0.127
$\varepsilon = 1e - 3$	2087853 $\rightarrow$ 174978	2161	0.128
$\varepsilon = 1e - 2$	2087853 $\rightarrow$ 30668	1852	0.136
$\varepsilon = 1e - 1$	2087853 $\rightarrow$ 4346	1491	0.225

a set of points representing several values such as latitude, longitude and altitude. The total distance of this trajectory is 1,292,951 kilometres. A large amount of this data is generated in Beijing, China.

- Dataset 2: Open street map (OSM)
OSM provide public access to all GPS tracks ever uploaded by different users (<https://www.openstreetmap.org/traces>). It is one of the largest GPS traces dataset that is publicly available.
- Dataset 3: T-Drive trajectory [49], [50]
This dataset provides GPS trajectories of 10,357 taxis over a period of one week for a total distance of almost 9 million kilometres.

For the experiments, 80% of the dataset was used for model training and the rest for the inference phase.

B. DATA PREPROCESSING

This section mainly analyzes the impact of applying pre-processing techniques to the raw GPS data. User trajectory data is loaded and then the preprocessing techniques are applied to the data frames. We deployed python panda library (<http://pandas.pydata.org/>) to store the data into several data frames.

Figure 4 represents the impact of different line simplification approaches on the mobility models' performances in terms of mean square error. We applied 5 line simplification methods on the dataset to evaluate the impact of data reduction on the prediction performance. These methods are: nth point, Reumann-Witkam (R-W), Lang, Visualingam-Whyatt (V-W) and D-P [29]. As shown, D-P technique outperforms

TABLE 4. Model performance (loss value) of BTM based on different number of layers and neurons.

	BTM			
	$n = 10$	$n = 50$	$n = 100$	$n = 150$
$L = 3$	0.029	0.028	0.025	0.035
$L = 4$	0.056	0.081	0.093	0.082
$L = 5$	0.123	0.093	0.084	0.091

other data reduction methods in terms of lower mobility model error. Therefore, this technique can be deployed to eliminate the unnecessary data without any disruptive impact on prediction performance. Also, in comparison with the other mobility models, BTM obtains the lowest error of almost 0.054.

Figure 3 depicts the impact of different thresholds of data reduction algorithm on three parameters. Figure 3(a) shows how data reduction procedure works as we increase the threshold value. It is shown that data reduces noticeably with higher thresholds. In Figure 3(b), we can see the effect of different thresholds on execution time (i.e., data reduction and mobility learning steps). Logically, as we increase the threshold and obtain smaller dataset, the execution time reduces significantly as well. Lastly, Figure 3(c) demonstrates that how data reduction affects model performance (i.e., loss value of the learning algorithm). It is evident that reducing data for some thresholds does not result in model performance degradation. However, for higher thresholds (e.g., 1e-2, 1e-1), data reduction has disruptive effect on the performance. When we increase the threshold, the model is learning from the fewer data points and it is reasonable to have a slight decrease in the model performance. Furthermore, the impact of different RDP thresholds on execution time and accuracy performance is summarized in Table 3. It is evident that a higher value of threshold results in fewer data points and slightly higher loss error in some cases. The threshold value (i.e., epsilon) is set to 1e-4. It is the optimum value that gives us the appropriate amount of data points to form an accurate trajectory without redundancy.

Figure 6(a) indicates a sample of a plain trajectory of a user with all the data points and Figure 6(b) shows the same

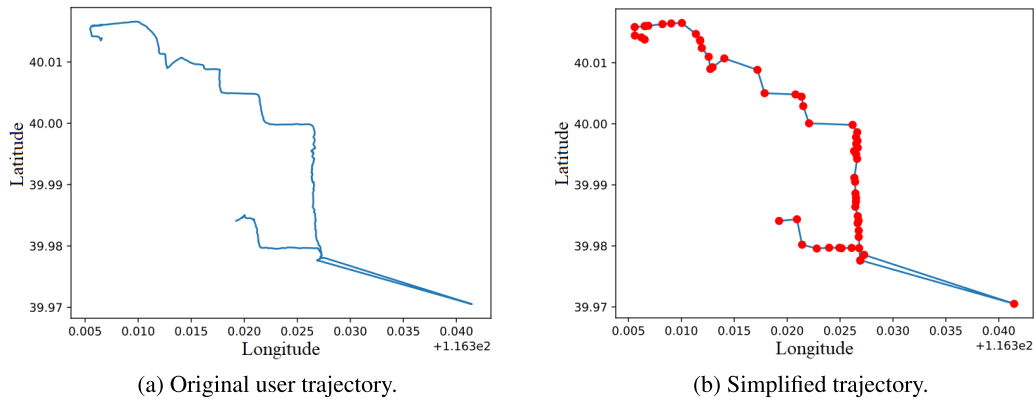


FIGURE 6. Removing unnecessary observations in data cleaning phase.

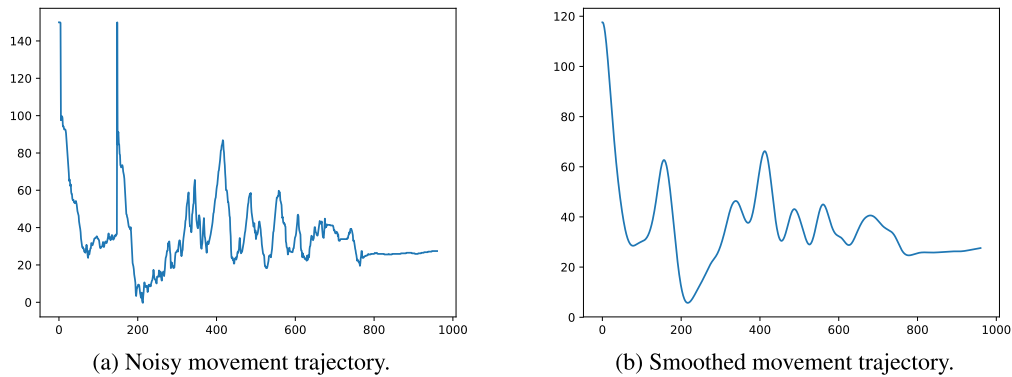


FIGURE 7. Noise reduction in data cleaning phase.

trajectory after choosing necessary points from the trajectory. As shown, the simplified version of the user movement trajectory is very similar to the original track and keeps only the effective samples. This huge data reduction leads to a noticeable simplification for the next steps in terms of complexity and run time.

Next, in order to reduce the noise or uncertainty in the data, the trajectory data is fed into a Kalman filter. Given the noisy measurements and initial assumptions, it takes the imperfect information and provides only the useful parts at the end. In Figure 7(a), we observe that the trajectory is quite noisy and spiky. However, it is much more cleaner and smother after passing it through Kalman filter which is shown in Figure 7(b).

Finally, before applying mobility learning predictor to the user sequence, the dataset is standardized. This is a common step for many learning predictors which set the mean value of the data to zero and the standard deviation to 1.

C. TRAJECTORY PREDICTION PERFORMANCE EVALUATION

In this section, the main purpose is to assess the impact of the proposed model on the prediction accuracy. First, the influence of different number of layers and neurons in the deep neural network is investigated for the model.

Table 4 summarizes the results of different configurations for different numbers of layers and neurons in a multi layer bidirectional GRU-based network. Basically, a lower loss value is obtained as the number of neurons and layers increases. However, if the number of layers and neurons is increased too much, it may lead to model overfitting. For instance, in the proposed approach, a network with 5 layers and 100 neurons in each layer results in a higher error. It implies that the model is unnecessarily complex and it can effectively train the data with a simpler network; therefore, there is no need to add unnecessary layers to the model. Overfitting happens in the case that the network is not able to generalize the pattern from the training input samples to test data and performs poorly with unseen data. Therefore, a neural network with 3 layers and 100 neurons is chosen for the model.

BTPM performance in terms of MSE value is shown in Figure 8. Figure 8(a) and Figure 8(b) respectively show mobility model performance for GRU-based model and BTPM. Comparison of the proposed method with GRU is provided since it has the best performance in comparison with other methods. The main purpose is to show the overall trend of the model in training and test phases. For GRU-based model, test error is high at first and gradually decreases from almost 0.5 to 0.2 in 60 epochs. However, for BTPM, the loss value for the test data starts from 0.028 and reduces to 0.01 in 10 epochs which shows the excellent performance

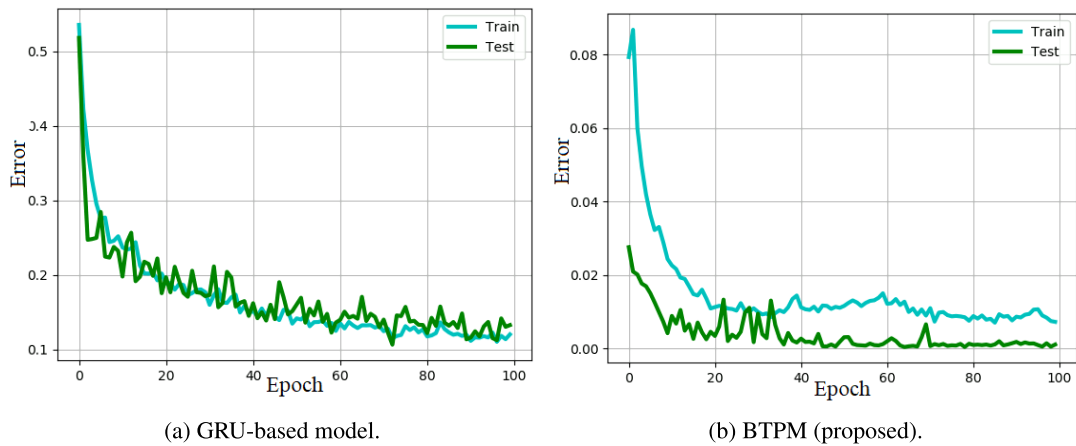


FIGURE 8. Mobility model learning performance.

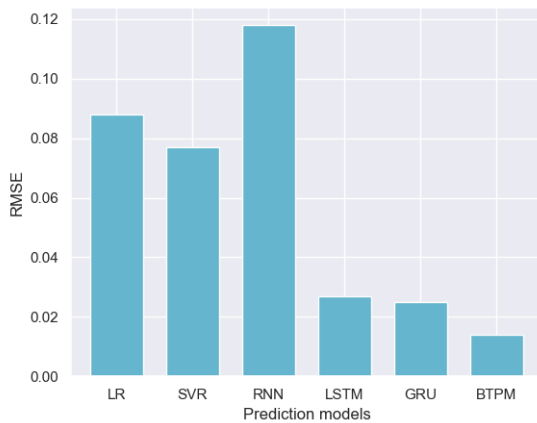


FIGURE 9. Mobility model prediction performances.

of the BTM in terms of training fast and with high accuracy.

Figure 9 shows the result of prediction errors for all the techniques. As shown, there is a spike in error with RNN-based model due to the vanishing/exploding gradient issue of RNNs. The prediction error noticeably decreases for LSTM and GRU since their gated structure can learn user mobility behavior more effectively. We can observe that BTM outperforms other approaches because of the fact that it extracts the correlations bidirectionally. Figure 10 shows the prediction errors for all the techniques in terms of MSE, MAE and RMSE applied on the three datasets. This comparison provides an intuitive idea of how the same mobility predictor can perform differently dealing with different user mobility data. For example, for dataset 3, BTM slightly improves the prediction performance in comparison with LSTM/GRU-based models. However, for datasets 1 and 2, there is a noticeable improvement. As is shown, BTM has the lowest error for all the three datasets compared to other methods. This can guarantee the effectiveness and generality of the proposed model.

Moreover, we have conducted simulations to evaluate the time complexity of the proposed mobility learning model

with a large input (i.e., when we have a big dataset as user prior information). We considered user 153, from dataset 1, with almost 5 years movement trajectory from July 2007 to June 2012. Figure 11 presents processing times for all the methods. It is shown that GRU-based model is the most time-consuming method with 110,250 seconds. BTM has the lowest time complexity with only 2,618 seconds. Using preprocessing techniques result in a huge reduction in execution time for the proposed method since a huge part of the unnecessary location points were removed before mobility learning. The proposed model takes advantage of numerous merits of bidirectional neural networks and addresses the inevitable problem of time complexity by applying appropriate preprocessing techniques.

D. ROBUSTNESS OF THE PROPOSED METHOD

From another point of view, we conduct another experiment in order to evaluate the proposed model based on the robustness of the technique. We want to investigate the impact of two important factors on different mobility models:

- 1) *User mobility behavior*: Mobility behavior varies from one person to another and people have different degrees of predictability. Users that regularly travel hundreds of kilometres are most likely to have a low mobility predictability [51]. Therefore, a mobility predictor that can perform well for different users is a robust and reliable predictor.
- 2) *Length of available data for each user*: A determining factor in mobility learning is the amount of available data for each user. Undoubtedly, mobility learning accuracy for a user with longer available prior data is higher than having less available data for the same person.

These two factors are inevitable for mobile users in reality. We considered the combination of these factors to evaluate different mobility models in terms of being practical in more close to real-world scenarios. To this end, for the input data, we consider different users with different levels of predictability and also with different amounts of data

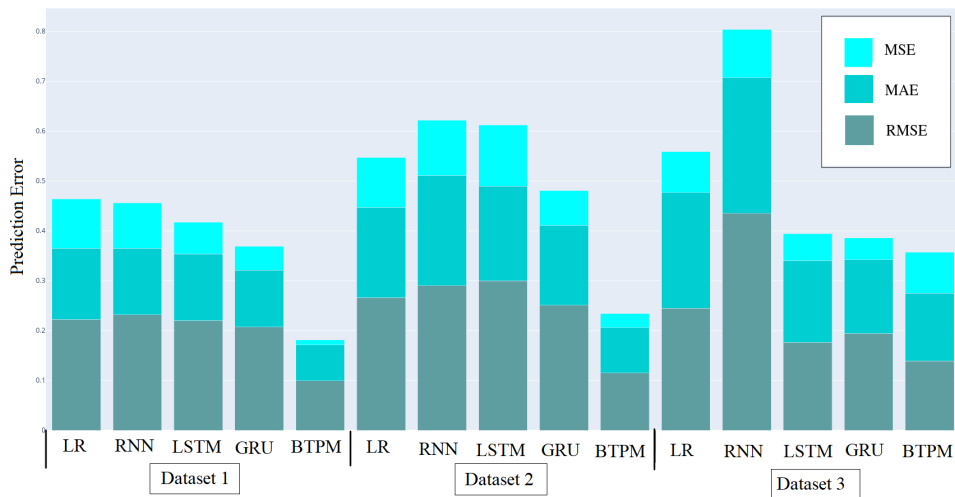


FIGURE 10. Mobility model performances using three different datasets.

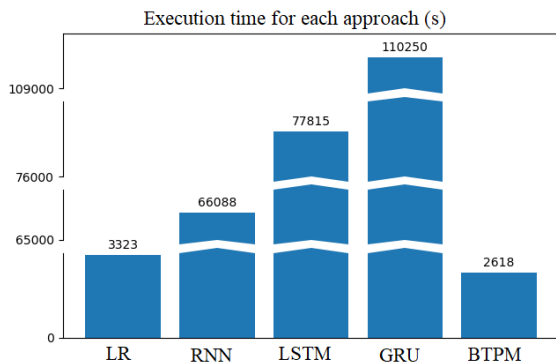


FIGURE 11. Execution time for each approach.

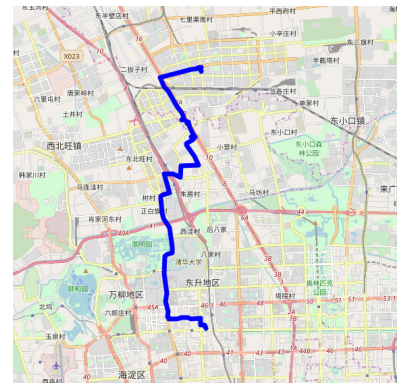


FIGURE 12. A sample of predicted trajectory for the user.

available as movement history for each one them. In other words, we choose different users with their own different mobility behaviors to find out how well different techniques work dealing with different users.

Here, we selected three users from each of datasets. Table 5 presents the loss value of each method for each user. In this table, the term “range” refers to the subtraction of the highest value and the lowest value of error for each method. When a method has a high value of range, it means that there is a noticeable difference between the highest and the lowest prediction errors. In other words, this mobility model can work well for a specific user with a specific mobility behavior (i.e., can learn the mobility pattern of the user) and can have a poor performance for another user; therefore, it is not a robust mobility model. The lowest range belongs to BTM with 0.251 which is smaller than other methods and shows that it can perform very well for different mobility behaviors and it is not dependent on a specific mobility behavior. Table 6 represents a qualitative comparison between different approaches for user trajectory prediction. BTM extracts mobility patterns by deeply considering both the prior and posterior correlations in the user mobility trajectory. Consequently, it achieves

a relatively better accuracy performance compared to the other methods. Moreover, the key feature of the two-phase user sequence preparation is huge time complexity reduction while improving the prediction accuracy

E. EXTENSION TO CELLULAR NETWORKS

Cellular networks mobility datasets (i.e., call detail records (CDR) datasets) are mostly used to predict the next cell of the user in mobile networks. These methods to some extent suffer from a low location accuracy depending on the cell range. On the other hand, GPS data have the best location accuracy performance and mostly are used for the next location prediction based on the geographical coordinates of the user. An ideal situation is when we can benefit from both methods (i.e., to predict the next cell of the user with the exact geographical coordinates in the cell). In this work, we fulfill this objective. A GPS dataset was deployed for the proposed approach to take advantage of having a precise and accurate record of user movement history which will result in a precise prediction as well. Then we convert it to cellular networks information.

TABLE 5. Robustness of different approaches.

	Users	Duration	RNN	LSTM	GRU	BTPM (proposed)
Dataset 1	150	10 days	0.245	0.235	0.234	0.094
	153	3 months	0.118	0.027	0.025	0.014
	165	1 month	0.086	0.047	0.053	0.032
Dataset 2	1	hours	0.331	0.271	0.256	0.144
	2	hours	0.695	0.195	0.171	0.170
	3	hours	0.498	0.417	0.407	0.265
Dataset 3	t1	1 week	0.614	0.585	0.469	0.178
	t2	1 week	0.331	0.329	0.262	0.091
	t3	1 week	0.095	0.068	0.057	0.029
Range			0.609	0.558	0.444	0.251

TABLE 6. Comparison between different approaches (more stars show a relatively better performance).

	RNN based model	LSTM based model	GRU based model	BTPM
Consider prior data correlation	Yes	Yes	Yes	Yes
Consider posterior data correlation	No	No	No	Yes
Fix the exploding/vanishing problem	No	Yes	Yes	Yes
Prediction accuracy	*	**	***	****
Time complexity	***	**	*	****
Example in the literature	[29]	[16]	[15]	Proposed

TABLE 7. Cell tower ID and location area code (LAC) for the predicted trajectory.

Order	Cell ID	LAC	Order	Cell ID	LAC	Order	Cell ID	LAC	Order	Cell ID	LAC
1	2896615	41192	6	2907505	41192	11	58594	4526	16	58592	4526
2	2896616	41192	7	31137	4526	12	36422	4523	17	4331139	4147
3	19995649	4523	8	2918015	41192	13	2895107	41192	18	23535874	4523
4	59618	4309	9	2895117	41192	14	127883653	4523	19	20162561	4523
5	31085	4523	10	18181132	4423	15	2898306	41192	20	4926831	4147

In this section, we want to take the predicted information of user's future trajectory in geographical coordinates and transfer it to cellular networks information. Figure 12 shows a sample of predicted future trajectory of a user. Next, using a real-world database of cellular towers with GPS positions (OpenCellID [52]), we transferred the predicted information based on geographical coordinates (i.e., longitude and latitude) to the cellular network information (i.e., Cell ID and corresponding location area code). We map the predicted trajectory to the corresponding cell towers on that coordinates (based on closest distance). This database contains the information of cell towers, including their locations based on latitude and longitude, cell range, mobile country codes (MCC), mobile network code (MNC), location area code (LAC), cell tower ID (Cell ID) and radio type (i.e., LTE,GSM). Table 7 summarizes an example of the

corresponding cell towers information (Cell tower ID and location area code) for the predicted trajectory sample for the 20 time steps ahead.

VI. CONCLUSION

In this paper, we highlighted the importance of mobility prediction as a promising key enabler for intelligent mobility management in self-organizing cellular networks. The proposed method mainly focuses on the prediction accuracy performance improvement to meet the demands of future mobile networks. A novel bidirectional trajectory prediction model is proposed that is mainly based on the bidirectional recurrent neural networks. Moreover, a preprocessing phase is introduced to prepare the mobility dataset specifically suitable for the framework. Line simplification techniques and Kalman filter were deployed to respectively reduce the

unnecessary data and remove the noise from the data before training the model. The proposed approach (BTPM) has two main distinguishing features in comparison with other methods: (i) having a two-phase user mobility sequence preparation specifically introduced for mobility prediction purpose which results in decreasing the noise level and significant time complexity reduction, (ii) having a bidirectional approach to extract mobility patterns by deeply considering the correlation in the user mobility trajectory which results in significant prediction accuracy improvement, obtaining a robust performance dealing with different users with different degrees of predictability and being practical in more close to real-world scenarios. This can have a profound impact on our analysis and gives us a deeper understanding of user mobility behavior with a significantly lower execution time. Simulation results show that BTPM has a high accuracy performance and outperforms other alternative approaches. It obtains a model error of 0.014 and effectively decreases the execution time up to 97%. The bidirectional network takes full advantage of data analysis in both directions (backward and forward) in order to provide a long-term prediction and to model user's mobility even with a complex pattern.

For future work, the proposed bidirectional mobility model can be applied to the other types of user mobility data (e.g., CDR datasets, check-in datasets). Moreover, it can be used as a potential tool for providing mobility-aware services in cellular networks such as mobility-aware call admission control techniques and resource allocation.

APPENDIX

The standard procedure of Kalman filter algorithm has two main steps including time update and measurement update [42]. Equations for time update (prediction) are

$$\hat{x}_k^- = B\hat{x}_{k-1} + Cu_{k-1}, \quad (14)$$

$$P_k^- = Bp_{k-1}B^T + Q, \quad (15)$$

and the equations for measurement update (correction) are

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}, \quad (16)$$

$$\hat{x}_k = \hat{x}_k^- + K_k(M_k - H\hat{x}_k^-), \quad (17)$$

$$P_k = (I - K_k H)P_k^-. \quad (18)$$

The description for the variables are as follows:

$\hat{x}_k^-$: Posterior estimation of process states at time step k

$\hat{x}_k^-$: Prior estimation of process states at time step k

B : Matrix B Shows the relationship between two consecutive timesteps

C : Matrix C indicates the relationship between the control input and the states

u_k : Control input vector

P_k : Posterior estimation of error covariance

P_k^- : Prior estimation of error covariance

Q : Process noise covariance

R : Measurement noise covariance

K_k : Kalman filter gain

H : Measurement matrix

M_k : Measurement vector at time step k

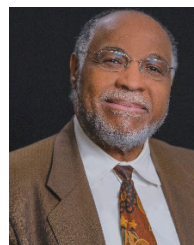
REFERENCES

- [1] P. Fazio, F. De Rango, and M. Tropea, "Prediction and QoS enhancement in new generation cellular networks with mobile hosts: A survey on different protocols and conventional/unconventional approaches," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1822–1841, 3rd Quart., 2017.
- [2] P. V. Klaine, M. A. Imran, O. Onireti, and R. D. Souza, "A survey of machine learning techniques applied to self-organizing cellular networks," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2392–2431, 4th Quart., 2017.
- [3] *Cisco Annual Internet Report (2018–2023) White Paper*, Cisco, San Jose, CA, USA, 2020.
- [4] R. Di Taranto, S. Muppirisetty, R. Raulefs, D. Slock, T. Svensson, and H. Wymeersch, "Location-aware communications for 5G networks: How location information can improve scalability, latency, and robustness of 5G," *IEEE Signal Process. Mag.*, vol. 31, no. 6, pp. 102–112, Nov. 2014.
- [5] P. Fan, J. Zhao, and C.-L. I., "5G high mobility wireless communications: Challenges and solutions," *China Commun.*, vol. 13, no. 2, pp. 1–13, 2016.
- [6] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.
- [7] J. Chung, C. Gülcehre, K. H. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, *arXiv:1412.3555*.
- [8] D. H. Douglas and T. K. Peucker, *Algorithms for Reduction Number Points Required to Represent a Digitized Line or its Caricature*. Hoboken, NJ, USA: Wiley, 2011, pp. 15–28, ch. 2.
- [9] G. Xu, S. Gao, M. Daneshmand, C. Wang, and Y. Liu, "A survey for mobility big data analytics for geolocation prediction," *IEEE Wireless Commun.*, vol. 24, no. 1, pp. 111–119, Feb. 2017.
- [10] N. Bui, M. Cesana, S. A. Hosseini, Q. Liao, I. Malanchini, and J. Widmer, "A survey of anticipatory mobile networking: Context-based classification, prediction methodologies, and optimization techniques," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1790–1821, 2017.
- [11] Y. Zheng, "Trajectory data mining: An overview," *ACM Trans. Intell. Syst. Technol.*, vol. 6, no. 3, pp. 29:1–29:41, May 2015.
- [12] R. Wu, G. Luo, J. Shao, L. Tian, and C. Peng, "Location prediction on trajectory data: A review," *Big Data Mining Analytics*, vol. 1, no. 2, pp. 108–127, Jun. 2018.
- [13] Z. Zhao, H. N. Koutsopoulos, and J. Zhao, "Individual mobility prediction using transit smart card data," *Transp. Res. C, Emerg. Technol.*, vol. 89, pp. 19–34, Apr. 2018.
- [14] D. Yao, C. Zhang, J. Huang, and J. Bi, "SERM: A recurrent model for next location prediction in semantic trajectories," in *Proc. ACM Conf. Inf. Knowl. Manage.*, New York, NY, USA, Nov. 2017, pp. 2411–2414.
- [15] J. Feng, Y. Li, C. Zhang, F. Sun, F. Meng, A. Guo, and D. Jin, "DeepMove: Predicting human mobility with attentional recurrent networks," in *Proc. World Wide Web Conf. World Wide Web (WWW)*, Geneva, Switzerland, 2018, pp. 1459–1468.
- [16] C. Wang, L. Ma, R. Li, T. S. Durrani, and H. Zhang, "Exploring trajectory prediction through machine learning methods," *IEEE Access*, vol. 7, pp. 101441–101452, 2019.
- [17] A. B. Adege, H.-P. Lin, and L.-C. Wang, "Mobility predictions for IoT devices using gated recurrent unit network," *IEEE Internet Things J.*, vol. 7, no. 1, pp. 505–517, Jan. 2020.
- [18] 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. Surveys Tuts.*, vol. 20, no. 4, pp. 2784–2803, 4th quart., 2018.
- [19] Z. Yan, D. Chakraborty, C. Parent, S. Spaccapietra, and K. Aberer, "Semantic trajectories: Mobility data computation and annotation," *ACM Trans. Intell. Syst. Technol.*, vol. 4, no. 3, pp. 1–38, Jun. 2013.
- [20] J. J.-C. Ying, W.-C. Lee, and V. S. Tseng, "Mining geographic-temporal-semantic patterns in trajectories for location prediction," *ACM Trans. Intell. Syst. Technol.*, vol. 5, no. 1, pp. 1–33, Dec. 2013.
- [21] T.-Y. Fu and W.-C. Lee, "Trembr: Exploring road networks for trajectory representation learning," *ACM Trans. Intell. Syst. Technol.*, vol. 11, no. 1, pp. 1–25, Feb. 2020.
- [22] A. Sadri, F. D. Salim, Y. Ren, W. Shao, J. C. Krumm, and C. Mascolo, "What will you do for the rest of the day?: An approach to continuous trajectory prediction," *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 2, no. 4, pp. 1–26, Dec. 2018.
- [23] A. Mohamed, O. Onireti, M. A. Imran, A. Imran, and R. Tafazolli, "Predictive and core-network efficient RRC signalling for active state handover in RANs with control/data separation," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1423–1436, Mar. 2017.

- [24] H. Lee, M. Wicke, B. Kusy, O. Gnawali, and L. Guibas, "Predictive data delivery to mobile users through mobility learning in wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 64, no. 12, pp. 5831–5849, Dec. 2015.
- [25] D. Lian, X. Xie, V. W. Zheng, N. J. Yuan, F. Zhang, and E. Chen, "CEPR: A collaborative exploration and periodically returning model for location prediction," *ACM Trans. Intell. Syst. Technol.*, vol. 6, no. 1, pp. 1–27, Apr. 2015.
- [26] Q. Liu, S. Wu, L. Wang, and T. Tan, "Predicting the next location: A recurrent model with spatial and temporal contexts," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 194–200.
- [27] Z. Lu, F. Zhao, H. Luo, Q. Wang, and W. Shao, "A destination prediction algorithm using spatial temporal bidirectional lstm networks," in *Proc. IPIN*, 2019, pp. 1–8.
- [28] J. Xu, J. Zhao, R. Zhou, C. Liu, P. Zhao, and L. Zhao, "Destination prediction a deep learning based approach," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 2, pp. 651–666, Feb. 2019.
- [29] N. Bahra and S. Pierre, "RNN-based user trajectory prediction using a preprocessed dataset," in *Proc. 16th Int. Conf. Wireless Mobile Comput., Netw. Commun. (WiMob)*, Oct. 2020, pp. 1–6.
- [30] Y. Leng, "Urban computing using call detail records: Mobility pattern mining, next-location prediction and location recommendation," M.S. thesis, Dept. Civil Environ. Eng., Massachusetts Inst. Technol., Cambridge, MA, USA, Jun. 2016. [Online]. Available: <https://dspace.mit.edu/handle/1721.1/104156>
- [31] Y. Chen, C. Long, G. Cong, and C. Li, "Context-aware deep model for joint mobility and time prediction," in *Proc. 13th Int. Conf. Web Search Data Mining*, New York, NY, USA, Jan. 2020, pp. 106–114.
- [32] B. Altaf, L. Yu, and X. Zhang, "Spatio-temporal attention based recurrent neural network for next location prediction," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2018, pp. 937–942.
- [33] Q. Gao, F. Zhou, G. Trajcevski, K. Zhang, T. Zhong, and F. Zhang, "Predicting human mobility via variational attention," in *Proc. World Wide Web Conf. (WWW)*, New York, NY, USA, 2019, pp. 2750–2756.
- [34] F. Zhou, X. Yue, G. Trajcevski, T. Zhong, and K. Zhang, "Context-aware variational trajectory encoding and human mobility inference," in *Proc. World Wide Web Conf. (WWW)*, New York, NY, USA, 2019, pp. 3469–3475.
- [35] M. C. Bishop, *Pattern Recognition and Machine Learning* (Information Science and Statistics). Berlin, Germany: Springer-Verlag, 2006.
- [36] 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.
- [37] S.-B. Cho, "Exploiting machine learning techniques for location recognition and prediction with smartphone logs," *Neurocomputing*, vol. 176, pp. 98–106, Feb. 2016.
- [38] A. Nadembega, A. Hafid, and T. Taleb, "A destination and mobility path prediction scheme for mobile networks," *IEEE Trans. Veh. Technol.*, vol. 64, no. 6, pp. 2577–2590, Jun. 2015.
- [39] V. Kulkarni, A. Mahalunkar, B. Garbinato, and D. J. Kelleher, "On the inability of Markov models to capture criticality in human mobility," 2018, *arXiv:1807.11386*.
- [40] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
- [41] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.
- [42] G. Welch and G. Bishop, "An introduction to the Kalman filter," Univ. North Carolina Chapel Hill, Chapel Hill, NC, USA, Tech. Rep. 95-041, 1995. [Online]. Available: <http://www.cs.unc.edu/~welch/kalman/kalmanIntro.html>
- [43] F. Wilhelm, "Handling GPS data with Python," in *Proc. EuroPython*, 2016, doi: [10.5446/21141](https://doi.org/10.5446/21141).
- [44] C. Montella, "The Kalman filter and related algorithms: A literature review," May 2011. [Online]. Available: https://www.researchgate.net/publication/236897001_The_Kalman_Filter_and_Related_Algorithms_A_Literature_Review
- [45] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statist. Comput.*, vol. 14, no. 3, pp. 199–222, Aug. 2004.
- [46] D. E. Rumelhart, P. Smolensky, J. L. McClelland, and G. E. Hinton, *Schemata and Sequential Thought Processes in PDP Models*. Cambridge, MA, USA: MIT Press, 1986, pp. 7–57.
- [47] A. Gron, *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 1st ed. Newton, MA, USA: O'Reilly Media, 2017.
- [48] Y. Zheng, Y. Chen, Q. Li, X. Xie, and W.-Y. Ma, "Understanding transportation modes based on GPS data for web applications," *ACM Trans. Web*, vol. 4, no. 1, pp. 1–36, Jan. 2010.
- [49] J. Yuan, Y. Zheng, X. Xie, and G. Sun, "Driving with knowledge from the physical world," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, New York, NY, USA, 2011, pp. 316–324.
- [50] J. Yuan, Y. Zheng, C. Zhang, W. Xie, X. Xie, G. Sun, and Y. Huang, "T-drive: Driving directions based on taxi trajectories," in *Proc. 18th SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst. (GIS)*, New York, NY, USA, 2010, pp. 99–108.
- [51] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási, "Limits of predictability in human mobility," *Science*, vol. 327, no. 5968, pp. 1018–1021, 2010.
- [52] OpenCellID. *The World's Largest Open Database of Cell Towers*. Accessed: Oct. 12, 2019. [Online]. Available: <https://opencellid.org/>



NASRIN BAHRA received the B.Sc. and M.Sc. degrees from the Electrical Engineering Department, Sadjad University of Technology. She is currently pursuing the Ph.D. degree with the Department of Computer and Software Engineering, Polytechnique Montréal. She is a member of the Mobile Computing and Networking Research Laboratory (LARIM). Her current research interests include self-organized networking, intelligent mobility management, and mobility prediction in cellular networks.



SAMUEL PIERRE (Senior Member, IEEE) received the first Honorary Doctorate degree from the University of Quebec at Trois-Rivières (UQTR), in May 2014, and the second Honorary Doctorate degree from the University of Quebec in Outaouais (UQO), in November 2016. He is currently a Professor with the Department of Computer and Software Engineering, Polytechnique Montréal, and the Director of the Mobile Computing and Networking Research Laboratory (LARIM). He has authored or coauthored more than 550 technical publications, including papers in refereed archival journals, textbooks, patents, and book chapters. His research interests include wired and wireless communications, mobile computing and networking, cloud computing, and e-learning. He has received several awards, including the Prix Poly 1873 for excellence in teaching and training, in 2001 and 2005, and the Knight of the National Order of Quebec, in 2009. He is a fellow of the Engineering Institute of Canada, in 2003, and the Canadian Academy of Engineering, in 2008. In December 2011, he was appointed as a member of the Order of Canada. In 2017, he obtained the El Fassi Prize from the Agence Universitaire de la Francophonie (AUF) to highlight the action of a person who has exerted a significant influence through the quality of his expertise and the innovative nature of his achievements at the international level in the fields of research, training, development and international cooperation, governance, and/or transfer of knowledge or skills. In 2020, he received the Grand Prize for Professional Excellence from the Order of Engineers of Quebec (OIQ). In 2021, he received the Gold Medal from Engineers Canada.

...