This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2936507, IEEE Transactions on Industrial Informatics

DeepVM: RNN-based Vehicle Mobility Prediction to Support Intelligent Vehicle Applications

WEI LIU, *Member, IEEE,* YOZO SHOJI, *Member, IEEE*

***Abstract*—The recent advances in vehicle industry and vehicle-to-everything communications are creating a huge potential market of intelligent vehicle applications, and exploiting vehicle mobility is of great importance in this field. Hence, this paper proposes a novel vehicle mobility prediction algorithm to support intelligent vehicle applications. First, a theoretical analysis is given to quantitatively reveal the predictability of vehicle mobility. Based on the knowledge earned from theoretical analysis, a deep recurrent neural network (RNN)-based algorithm called DeepVM is proposed to predict vehicle mobility in a future period of several or tens of minutes. Comprehensive evaluations have been carried out based on the real taxi mobility data in Tokyo, Japan. The results have not only proved the correctness of our theoretical analysis, but also validated that DeepVM can significantly improve the quality of vehicle mobility prediction compared with other state-of-art algorithms.**

***Index Terms*—vehicle mobility, vehicle-to-everything (V2X), recurrent neural network, deep learning.**

I. Introduction

With the recent advance of automobile technologies, the motor vehicle has evolved from a simple mechanical device to a smart platform incorporating various communication, computation and sensing functions. It is expected that future vehicles can provide not only pleasant and safe driving experiences, but also various kinds of services such as multimedia infotainment and social interactions. One of the most promising technologies to meet such expectations is vehicular networks that enable vehicles to efficiently exchange information through vehicle-to-vehicle, vehicle-to-infrastructure, and vehicle-to-pedestrian communications. The Third Generation Partnership Project (3GPP) group collectively defines these technologies as vehicle-to-everything (V2X) communications [1]. Gartner estimates that the annual production of network-connected vehicles will reach 61 million by 2020 [2] and this leads to a great potential market of many intelligent vehicle applications like self-driving assistance, vehicle-based sensing data collection, traffic safety, geo-advertising, in-vehicle Internet access, and pothole detection [3–7].

The mobility of vehicles makes the topology of V2X networks highly dynamic, and this is one of the main challenges faced by V2X communications. Thus, exploiting vehicle mobility is of great importance in implementing intelligent vehicle applications. As an example, a number of smart city applications only require sensing data periodically and are delay-tolerant to data transmission, e.g., smart grid applications like advanced metering are tolerant to a data delay from tens of minutes to several hours [8], and an application that monitors the running statuses of street lights is tolerant to a delay of several hours or even longer [3]. Since it is quite expensive to deploy so many femtocells of cellular networks to transmit these sensing data generated by a large amount of geo-distributed IoT devices, many researchers suggest utilizing short-range V2X communication to offload these delay tolerant data from cellular networks [3, 8–10]. Two representative short-range V2X communication standards are IEEE 802.11p [11] and LTE-V2X Mode 4 [12] that cover a communication range from several hundred meters to very few kilometers. Figure 1 illustrates an example of applying vehicle mobility prediction to support this application. Assume that taxis *A* and *B* are moving around a city and they opportunistically collect sensing data via short-range communication when they encounter the sensors deployed in city (i.e., when vehicles enter the communication range of sensors). Both taxis want to deliver these sensing data to a road side unit (RSU) in the left of figure by short-range communication, and this RSU utilizes wired broadband networks to transmit sensing data back to a data center for further processing. When two taxis encounter each other, taxi *A* can intelligently forward its stored data to taxi *B* by short-range communication if taxi *B* is predicted to be moving towards the RSU. Consequently, when taxi *B* encounters the RSU later, the data from taxi *A* are successfully delivered even if taxi *A* never encounters that RSU directly. This kind of multi-hop data forwarding strategy powered by mobility prediction can significantly improve the quality of vehicle-based sensing data collection [9, 10]. Other intelligent vehicle applications that may benefit from mobility prediction include geo-advertising that utilizes vehicle mobility to broadcast advertisement in a specific city region [7] and mobile edge computing that utilizes vehicle mobility to stimulate the computational resource sharing in V2X networks [13].

Fig. 1: Applying vehicle mobility prediction to assist the delay-tolerant sensing data collection in smart city.

Wei Liu and Yozo Shoji are with the Open Innovation Promotion Headquarters, National Institute of Information and Communications Technology, Tokyo, 184-8795, JP.

Wei Liu is the corresponding author of this paper (e-mail: wei_liu@nict.go.jp)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2936507, IEEE Transactions on Industrial Informatics

However, since most vehicles move at their own wills, it would be difficult to obtain a perfect knowledge about their future mobility. To avoid this great uncertainty, existing works either make use of some metrics such as the encounter and inter-encounter time distributions to implement coarse-grained vehicle mobility predictions [10, 13, 14], or simplify the problem to a Markov model that is not sufficient to make accurate prediction [9]. Consequently, this paper proposes a deep recurrent neural network (RNN)-based algorithm called DeepVM to predict vehicle mobility accurately, and its main contributions are:

(1) A solid theoretical analysis is presented to reveal the predictability of vehicle mobility quantitatively;

(2) Based on the knowledge earned from theoretical analysis, a deep RNN-based algorithm called DeepVM is proposed to predict vehicle mobility. To the best of our knowledge, DeepVM is the first trial of deep learning technology in this field worldwide;

(3) Extensive evaluation results based on real taxi movements have not only validated the correctness of our theoretical analysis, but also shown that DeepVM significantly improves the quality of vehicle mobility prediction compared with the state-of-art algorithms.

A preliminary study of this work was presented in a conference paper [15]. Compared with its conference version, this paper supplements an entropy-based theoretical analysis to quantitatively evaluate the predictability of vehicle mobility and its correlation with vehicular trajectory knowledge. This analysis not only reveals the benefits of using deep learning for vehicle mobility prediction, but also explains the motivation of the proposed DeepVM algorithm. Furthermore, this paper supplements extensive evaluations to validate DeepVM from different aspects. Instead of simply comparing the performances of DeepVM and other state-of-art algorithms, the evaluation results presented in this paper emphasize on clarifying the theoretical factors that contribute to the superior performance of DeepVM. Over 75% analysis and evaluation results of this paper are first presented. Finally, the introduction and related work parts of this paper are also improved to better illustrate the application scenarios and novel points of DeepVM.

The organization of this paper is as follows: Section II gives a review of related literature. Section III introduces the system model and problem formulation. Section IV conducts a theoretical analysis on the predictability of vehicle mobility. Section V describes the detailed process of our proposed DeepVM algorithm. Section VI presents the comparison results of DeepVM and other state-of-art algorithms. Section VII concludes this paper with a final discussion of future work.

II. Related Work

There are many existing works aim at predicting vehicle mobility in the background of selecting a stable wireless link for routing data in vehicle ad-hoc networks [16–18]. Agarwal et al. proposed a Dead Reckoning mechanism that uses the linear sum of a vehicle's instant position and velocity to predict its near future positions [16]. Balico et al. adopted a shallow feedforward neural network with one hidden layer to predict vehicle mobility [17]. Their neural network accepts the instant position and velocity of a vehicle as input, and outputs its predicted next position with a time interval from 0.5 to 2 seconds. Evaluation results based on real vehicular trajectories have illustrated that this algorithm reduces mobility prediction error when compared with the Dead Reckoning and Kalman filter-based algorithms. Aljeri et al. described a prediction algorithm based on a Particle filter [18]. They modeled the mobility prediction problem as an iterative Particle filtering process on three parameters, i.e., the position, velocity, and acceleration of a vehicle. Their results have validated that the Particle filter-based algorithm outperforms those based on Kalman and extended Kalman filters. However, since these works only aim at predicting vehicle mobility to improve the quality of ad-hoc data routing, they assume a vehicle's kinetic parameters like velocity and acceleration are relatively constant during the concerned data transmission period of a few seconds. Obviously, this assumption does not hold in the scenario with a longer prediction period like several or tens of minutes. Zhu et al. have proved that the uncertainty of future vehicle mobility can be reduced by giving the knowledge of previous vehicular trajectory, and used a 2-order Markov model-based algorithm to predict vehicle mobility accordingly [9]. However, there are two limitations in their work: (1) The theoretical analysis only concerns predicting a vehicle's position in the next one time slot; and (2) Based on their incomplete theoretical analysis, a 2-order Markov model is claimed to be sufficient for predicting vehicle mobility. Our work in this paper not only extends the theoretical analysis in [9] to predict vehicle mobility in multiple future time slots, but also proposes a novel deep learning algorithm that outperforms Markov model-based algorithms significantly.

Several existing works also introduce the intelligent vehicle applications that may benefit from an accurate prediction of vehicle mobility. Bonola et al. evaluated the performance of using 120 taxis to collect and disseminate sensing data in Rome, Italy [3]. They have shown that even a small fleet of 120 taxis can disseminate sensing data to 80% areas of Rome in one day. However, their work does not exploit the possibility of predicting vehicle mobility to accelerate this process. Lin et al. introduced a sensing data collection framework by using the short-range V2X communication in smart city [10]. Their algorithm extracts the regular routes of vehicles from their daily mobility trajectories, and derives the encounter opportunities between different pairs of vehicles and RSUs accordingly. As a result, they let a vehicle with less opportunities to encounter RSUs forward its sensing data to other vehicles that have more opportunities to encounter RSUs. This kind of multi-hop data forwarding strategy can not only improve the success ratio of data collection, but also reduce the delay of data collection significantly. Compared with our work that focuses on predicting vehicle mobility, Lin et al. hypothesized that vehicle mobility is almost regular and their algorithm does not try to use any vehicle mobility prediction approach to estimate the encounter opportunities between vehicles and RSUs. Liu et al. proposed a mobile edge computing architecture that can be applied to V2X networks [13]. In this architecture, a vehicle partitions its

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2936507, IEEE Transactions on Industrial Informatics

computational task into several subtasks and delegates them to the service providers like RSUs and other vehicles that are opportunistically encountered during movement. Service providers start to execute the received subtasks by using their own computational resources. When the execution of a subtask is finished, the requesting vehicle downloads task results from the corresponding service provider when they encounter again. This architecture adopts some coarse-grained mobility statistics such as the encounter interval and duration between vehicles and service providers to accelerate task completion while ignoring the potential of trajectory-based mobility prediction

Finally, an increasing number of researchers are applying deep learning technology to explore the crowd and traffic flows in city [19–22]. Song et al. proposed a deep RNN architecture to jointly learn human mobility and transportation transition model from a heterogeneous data source of human movements and city transportation networks [19]. Their algorithm receives sequential input data of five time steps, and successfully explores the correlation between human mobility and their transportation modes to give accurate prediction. Compared with their work, our proposal presented in this paper aims at predicting vehicle mobility from raw GPS data only, and adopts a different RNN architecture that receives a much longer sequence of input data to against the high uncertainty of vehicle mobility. Zhang et al. designed a novel architecture called DeepST to predict the crowd flow in city [20]. They modeled the in-flow and out-flow of the crowd in different city regions and used a sequence of convolutional neural networks to learn the spatial-temporal patterns of these flows. A software tool was also developed for users to view the historical, real-time and forecasting crowd flows in city. Lv et al. studied predicting the macro-level traffic flow in city with a deep stacked autoencoder, and trained the network layer by layer greedily [21]. They have proved that the deep learning-based model is more accurate compared with other baseline models. Li et al. proposed a deep belief network to mine the hidden features of the traffic data in Macao, and combined the deep belief network with a support vector regression classifier to predict traffic congestion accordingly [22]. Different from the previous three works [20–22] that mainly focus on optimizing urban transportation system by using city-wide traffic statistics to predict the macro-level flows of vehicles and crowd, our work in this paper aims at applying deep learning technology to predict the micro-level mobility of a vehicle by using its mobility trajectory directly.

It can be concluded from the above discussion that none of the existing works considers the possibility of using deep learning technology to predict vehicle mobility. Thus, our work presented in this paper validates the potential and superiority of this strategy by providing both theoretical and empirical evidences. Compared with the existing works on predicting the macro-level statistics of vehicular traffic flow, our proposal is more helpful to the intelligent vehicle applications that are driven by the separate mobility of each vehicle and aim at utilizing the opportunistic communication window between nearby vehicles and other internet of things to provide novel services.

Fig. 2: The snapshot of taxi mobility in one day.

III. System Model and Problem Formulation

This paper focuses on proposing an algorithm to predict the probability for a vehicle to enter any city region in a concerned future period, when its mobility trajectory is given. It is assumed that vehicles move in a city space (S) and this space is divided into n disjoint square grids,

$$S = \{g_1, g_2, g_3 \cdots, g_n \mid g_i \cap g_j = \emptyset\}, \tag{1}$$

where g_i denotes grid identity. The typical size of a grid ranges from several hundred meters to very few kilometers, so that a vehicle can communicate with other internet of things locating in the same grid through short-range V2X communications to support intelligent vehicle applications. The time is also discretized into a sequence of time slots. In every time slot, vehicles sample their positions through a positioning system such as GPS.

When a vehicle v moves around the city, its position at a time slot t is a random variable s_t that takes a value from the space defined by Eq. (1). Thus, its mobility trajectory in the previous k time slots (including t) is represented by a sequence of these random variables,

$$T_t^k = \langle s_{t-k+1}, s_{t-k+2} \cdots s_{t-1}, s_t \rangle, \tag{2}$$

where k is also called the order of this trajectory. Given its k-order mobility trajectory at time t, the probability for the vehicle v to locate in the grid g_i at a future time slot $t^{'}$ is denoted by

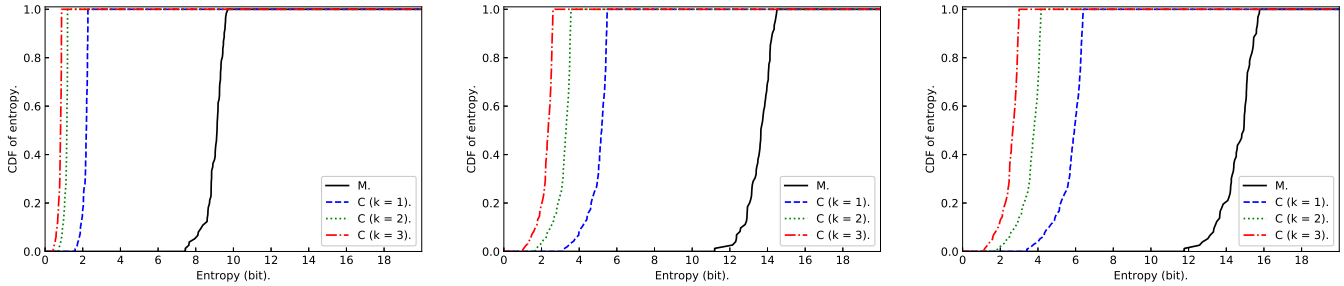
$$P_{t'}(g_i) = \{P(X_{t'} = g_i \mid T_t^k), t^{'} > t\}. \tag{3}$$

Similarly, given its k-order mobility trajectory at time t, the probability for the vehicle v to enter the grid g_i in a concerned prediction period $(t, t+c]$ is represented by

$$P_{t\sim t+c}(g_i) = P(\cup_{j=1}^{c} X_{t+j} = g_i \mid T_t^k), \tag{4}$$

where c indicates the length of period. As a result, the aim of this paper is transformed to propose an algorithm that can solve Eq. (4) in an accurate and effective way.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2936507, IEEE Transactions on Industrial Informatics



(a) A prediction period of 1 min. (b) A prediction period of 5 min. (c) A prediction period of 10 min.

Fig. 3: The cumulative distribution functions (CDFs) of vehicles' marginal (M) and conditional (C) entropies.

IV. The Predictability of Vehicle Mobility

This section presents a quantitative analysis on the predictability of vehicle mobility. The following analysis is conducted based on the real taxi mobility data that are collected from a wireless IoT testbed located in Tokyo, Japan [23]. This data set contains the mobility data of 65 taxis in a period of 4 months, i.e., from January 2018 to April 2018. Every taxi periodically sends its mobility data back to a data center with an interval of 1 min. The city space for vehicle mobility prediction is the urban area of Tokyo that occupies about 700 km^2. This space is discretized into 2791 square grids of 500×500 m^2. Figure 2 visualizes a snapshot of taxi mobility in one day.

For any vehicle v, suppose that we have observed its trajectory G^l for l time slots, i.e., $G^l = \langle g^{(1)}, g^{(2)}, \cdots, g^{(l)} \rangle$, where $g^{(i)}$ indicates the vehicle's position at the i-th time slot. For a prediction period of c time slots, let W_c denote a sliding window of size c. By applying W_c to G^l, a series of trajectory snippets in c time slots are generated, i.e., $\langle g^{(1)}, g^{(2)}, \cdots, g^{(c)} \rangle, \langle g^{(2)}, g^{(3)}, \cdots, g^{(c+1)} \rangle, \cdots, \langle g^{(l-c+1)}, g^{(l-c+2)}, \cdots, g^{(l)} \rangle$. Without the loss of generality, assume that $\langle g^{(j)}, g^{(j+1)}, \cdots, g^{(j+c-1)} \rangle$ is one snippet and this mobility pattern appears q times in all the snippets of G^l. Here, a mobility pattern represents a group of snippets which trajectory sequences in c time slots are the same. Hence, the probability for this mobility pattern to appear in the trajectory of vehicle v is given by

$$P_c(m_i) = \frac{q}{l - c + 1}, \tag{5}$$

where m_i is the identity of mobility pattern and $l - c + 1$ is the total amount of snippets in the trajectory G^l. The marginal entropy of v's mobility pattern in a prediction period of c time slots can be computed by

$$H(c) = -\sum_{i=1}^{z} P_c(m_i) * log_2(P_c(m_i)), \tag{6}$$

where z is the total amount of different mobility patterns in the trajectory G^l. Based on Eq. (6), the conditional entropy of v's mobility pattern in a prediction period of c time slots given its k-order previous trajectory can be calculated by

$$\begin{aligned} H(c|k) &= H(c,k) - H(k) \\ &= H(c+k) - H(k). \end{aligned} \tag{7}$$

The second step of Eq. (7) comes from the fact that a vehicle's k-order previous trajectory and positions in the future c time slots comprise a trajectory snippet of $k + c$ time slots.

In essence, the marginal entropy of a vehicle, i.e., $H(c)$, reveals its spatial regularity to move around the city in a prediction period of c time slots. Differently, the conditional entropy of a vehicle, i.e., $H(c|k)$, reveals its spatial-temporal regularity to move around the city in c time slots after giving its previous k-order trajectory, and it demonstrates the correlation of the vehicle's positional transitions over a series of consecutive time slots. In both cases, a larger entropy value indicates that there is a higher uncertainty in vehicle mobility, and makes it theoretically less predictable. Figure 3 plots the cumulative distribution functions (CDFs) of 65 vehicles' marginal and conditional entropies in the data set, with a prediction period of 1, 5, and 10 min. Several insights of vehicle mobility prediction can be observed from this figure:

(1) The vehicles' marginal entropies increase with the length of prediction period. This result well matches our common sense that the task of predicting vehicle mobility in a long period is more difficult than that in a short period;

(2) The vehicles' conditional entropies are significantly smaller than their marginal entropies. This implies that the uncertainty of vehicle mobility decreases when vehicular trajectory is known;

(3) As the order k of the known trajectory increases, the vehicles' conditional entropies continuously decrease. This implies that a longer previous trajectory helps to reduce the uncertainty of vehicle mobility, and makes it more predictable. This point becomes clearer with the increase of prediction period, since the benefit of increasing k becomes more significant.

As revealed by this theoretical analysis, a long vehicular trajectory can help to predict vehicle mobility accurately. The state-of-art solution proposed by Zhu et al. [9] only adopts a 2-order Markov model to predict vehicle mobility based on a 2-order vehicular trajectory. A straightforward approach to

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2936507, IEEE Transactions on Industrial Informatics

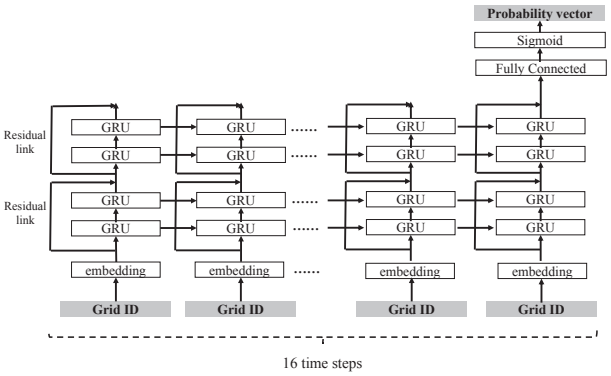


Fig. 4: The neural network architecture of DeepVM algorithm unfolded in time.

improve the quality of prediction might be increasing the order of Markov model to cope with a longer vehicular trajectory. However, as shown by Eq. (2), the amount of possible mobility patterns in a trajectory increases exponentially with its order k. This indicates an $O(n^k)$ computational complexity of training a Markov model where n denotes the number of grids in city space. Clearly, this complexity prevents Markov model-based algorithms from processing a long vehicular trajectory to improve their predictions, especially when the city space for mobility prediction is large.

V. The Proposed DeepVM Algorithm

This section describes the mechanism of our proposed DeepVM algorithm in detail. Briefly speaking, DeepVM adopts a deep RNN architecture and processes a 16-order vehicular trajectory to predict vehicle mobility. The choice of a 16-order trajectory is somewhat arbitrary, and it is determined to balance the trade-off between algorithm performance and our computational resources. As illustrated by the previous theoretical analysis, a longer trajectory may further improve the performance of DeepVM. Figure 4 shows the neural network architecture of DeepVM unfolded in time.

DeepVM first encodes every grid identity to an n-dimensional one-hot vector where n is the number of grids in city space, e.g., a grid identity of 2 is encoded to $(0, 0, 1, 0 \cdots 0)$. Every input data item of DeepVM contains a sequence of 16 grid identities that represent where a vehicle located in the past 16 time slots. The advantage of this one-hot vector representation is that it takes each grid identity equally regardless of the geographical location of grid. However, it also leads to a very sparse data item that delays the convergence of deep learning, e.g., there may exist thousands of '0's but only a '1' in a vector. Thus, DeepVM uses an embedding layer to transform a sparse grid identity vector into a smaller-and-denser feature vector. As will be shown in Section VI, this embedding step helps to accelerate the convergence of DeepVM without much performance degradation, and it is a well-accepted feature extraction method in deep learning [24].

The embedded vector of vehicular trajectory is fed into the RNN cells of DeepVM. Hochreiter et al. have proved that vanilla RNN cells cannot extract the long temporal dependency of input data due to the gradient vanishing and exploding problems, and Long Short-Term Memory (LSTM) [25] and Gated Recurrent Unit (GRU) [26] cells have been widely used to address these drawbacks. DeepVM adopts GRU cells based on the two observations in our preliminary experiments: (1) The performance discrepancies between two kinds of cells are usually less than 1%; and (2) GRU cells converge faster in training since they employ less parameters than LSTM cells. DeepVM integrates two GRU blocks, and each block is composed by two layers of GRU cells. When trained by the embedded vectors of vehicular trajectory, these GRU blocks are able to learn and store the spatial-temporal correlations among the embedded vectors and use them to make predictions. A residual link is added to connect the input and output layers of each GRU block, and it helps to alleviate the issue of gradient vanishing in GRU blocks [27]. A fully connected layer with sigmoid activation transforms the output vector of the last GRU cell to a real-valued probability vector,

$$P^j_{pred} = (p^j_{g_1}, p^j_{g_2}, p^j_{g_3}, \cdots p^j_{g_{n-1}}, p^j_{g_n}). \tag{8}$$

Here, j is the index number of data item, and $p^j_{g_i} \in (0, 1)$ denotes the probability for the corresponding vehicle of data item j to enter the grid g_i when its 16-order mobility trajectory is fed into DeepVM.

DeepVM is trained by minimizing a binary cross-entropy loss function. Let a binary vector L^j_{truth} denote the ground truth vector of the data item j,

$$L^j_{truth} = (l^j_{g_1}, l^j_{g_2}, l^j_{g_3} \cdots, l^j_{g_{n-1}}, l^j_{g_n}), \tag{9}$$

where $l^j_{g_i} = 1$ indicates that the corresponding vehicle enters the grid g_i during the concerned prediction period in reality, and vice versa. By training DeepVM with a mobility data set including m data items, its average binary cross-entropy loss is defined by

$$C = -\frac{1}{m}\sum_{j=1}^{m}\sum_{i=1}^{n} l^j_{g_i} * log(p^j_{g_i}), \tag{10}$$

where n indicates the number of grids in city space. As illustrated in Figure 4, the forward and backward operations of DeepVM at every time step are the same. Thus, its computational complexity is linear to the order k of mobility trajectory, i.e., $O(n * k)$. This superiority enables DeepVM to process a much longer vehicular trajectory than Markov model-based algorithms can do.

VI. Performance Evaluations

This section presents the evaluation results of our proposed DeepVM algorithm. The same data set described in Sect. IV is used in the following evaluations and it contains the mobility data of 65 taxis in Tokyo, Japan, from January 2018 to April 2018. Vehicle mobility prediction is conducted in the urban area of Tokyo, and it comprises 2791 square grids of 500×500 m^2 by default. The length of time slot for sampling vehicle mobility data is 1 minute. This mobility data set is segmented into training, validation, and test sets. The training set comprises the mobility data of taxis in the beginning 14 weeks (from 01/01/2018 to 08/04/2018), the validation set

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2936507, IEEE Transactions on Industrial Informatics

TABLE I: Basic evaluation parameters.

Embedding layer size	200
GRU cell size	200
GRU cell activation function	exponential linear unit (ELU)
Fully connected layer size	2791
Drop out ratio	0.5
learning rate	1e-3
Prediction period	10 min
Grid size	500 m

comprises the mobility data of taxis in the next 1 week (from 09/04/2018 to 15/04/2018), and the test set comprises the mobility data of taxis in the final 2 weeks (from 16/04/2018 to 30/04/2018). DeepVM is implemented by using Tensorflow 1.8.0 [28] and the following experiments are executed on a server with one Intel Core i7-4790 CPU, one Nvidia Tesla K-80 GPU, and 32GB RAM. Without specifically pointing out, the basic parameters summarized in Table I are used in the following evaluations.

A. Performance comparisons with the state-of-art algorithms

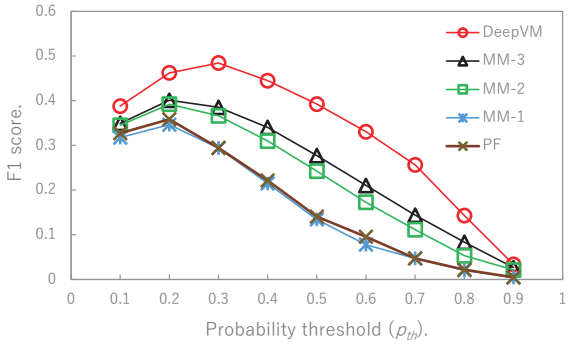
This part compares DeepVM with the state-of-art Markov model-based [9] and Particle filter-based (PF) [18] prediction algorithms. Besides the 2-order Markov model-based algorithm (MM-2) that is recommended in [9], the evaluations of 1-order and 3-order Markov model-based algorithms (MM-1 and MM-3) are also conducted to better validate the theoretical analysis presented in Sect. IV. These five algorithms are compared on the basis of four metrics in this subsection, i.e., precision, recall, F1 score, and hamming loss. All the four metrics require a binary prediction vector to calculate their values, while these algorithms only output a real-valued prediction vector P^j_{pred} as the one shown by Eq. (8). Hence, a probability threshold $p_{th} \in [0, 1]$ is employed to transform P^j_{pred} into a binary format,

$$P^j_{b-pred} = (b^j_{g_1}, b^j_{g_2}, b^j_{g_3}, \cdots b^j_{g_{n-1}}, b^j_{g_n}), \tag{11}$$

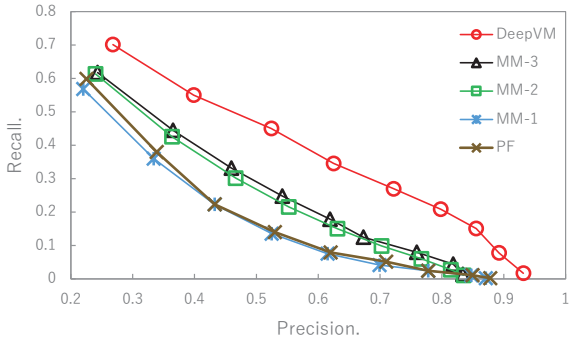
where $b^j_{g_i} = 0$ when $p^j_{g_i} \in P^j_{pred}$ is less than p_{th} and $b^j_{g_i} = 1$ otherwise. Intuitively, $b^j_{g_i} = 1$ predicts that the corresponding vehicle of data item j will enter the grid g_i because its estimated probability surpasses p_{th}, and vice versa. According to the ground truth vector L^j_{truth} defined by Eq. (9) and the binary prediction vector P^j_{b-pred} defined by Eq. (11), a brief introduction of the four comparison metrics is given below for the completeness of this paper, and their formal definitions can be found in [29]:

Precision (p): The precision for a data item j is defined as the size of its intersection set between the positive labels in L^j_{truth} and P^j_{b-pred} divided by the number of its positive labels in P^j_{b-pred}. A higher precision is preferred to reduce false positive prediction;

Recall (r): The recall for a data item j is defined as the size of its intersection set between the positive labels in L^j_{truth} and P^j_{b-pred} divided by the number its positive labels in L^j_{truth}. A higher recall is preferred to reduce false negative prediction;



(a) The F1 scores of algorithms.



(b) The precisions and recalls of algorithms.

Fig. 5: The F1 scores, precisions, and recalls of five algorithms with a prediction period of 10 min.

F1 score ($F1$)**:** F1 score is defined as the harmonic average of precision and recall, and it is an integrated metric to measure prediction quality, i.e., $F1 = (2 \times p \times r)/(p + r)$. A higher F1 score indicates better prediction quality;

Hamming loss: The hamming loss for a data item j is defined as the fraction of its labels that are incorrectly predicted, i.e., the fraction of inconsistent labels between L^j_{truth} and P^j_{b-pred}. A lower hamming loss denotes a better quality of vehicle mobility prediction.

Figure 5(a) plots the prediction F1 scores of five algorithms with a prediction period of 10 min. Markov model-based algorithms perform better when the length of their input vehicular trajectory increases, e.g., MM-2 improves the highest F1 score of MM-1 by 13.2% and MM-3 further improves that of MM-2 by 2.4%. It is clear that the improvement of F1 score between MM-2 and MM-3 is smaller than that between MM-1 and MM-2. This result well matches our theoretical analysis shown in Figure 3 that the reduction of conditional entropy between 2-order and 3-order vehicular trajectories is less than that between 1-order and 2-order trajectories. Since the Particle filter-based algorithm (PF) accepts a vehicle's instant position, velocity, and acceleration to predict vehicle mobility, its input features can be interpreted as the input feature of MM-1

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2936507, IEEE Transactions on Industrial Informatics

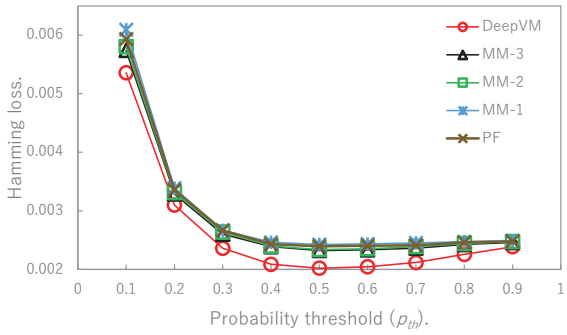


Fig. 6: The hamming losses of five algorithms with a prediction period of 10 min.

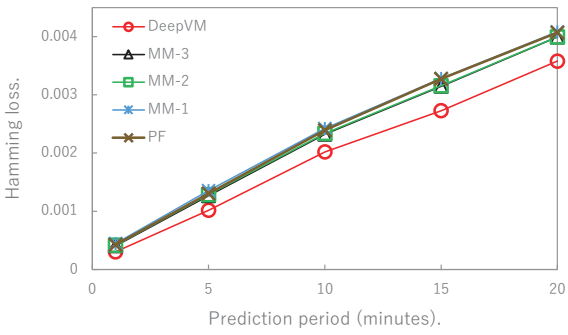


Fig. 8: The lowest hamming losses of five algorithms with a prediction period of 1, 5, 10, 15, and 20 min.

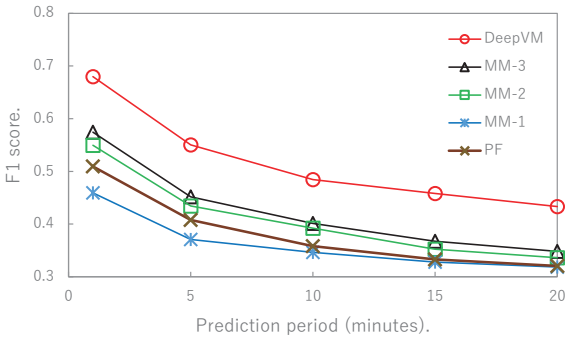


Fig. 7: The highest F1 scores of five algorithms with a prediction period of 1, 5, 10, 15, and 20 min.

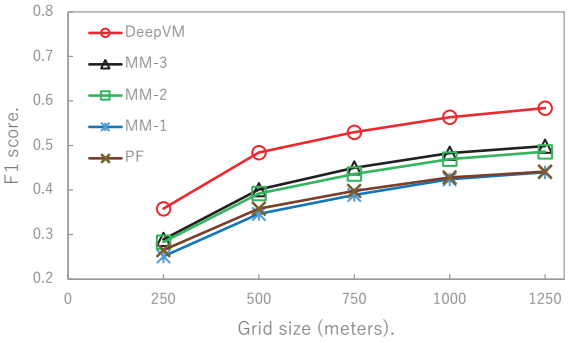


Fig. 9: The highest F1 scores of five algorithms with a grid size of 250, 500, 750, 1000, and 1250 m.

(i.e., one-order trajectory) plus velocity and acceleration. As discussed in Sect. II, the instant velocity and acceleration of a vehicle have few impact on its future mobility in a long prediction period like 10 min. Thus, PF only performs a little better than MM-1, while significant worse than MM-2 and MM-3. Benefitted from its ability to process a much longer (i.e., 16-order) vehicular trajectory than other algorithms, the proposed DeepVM algorithm improves the highest F1 score of MM-3 by 20.8%. This validates our theoretical findings in Sect. IV that a long previous vehicular trajectory is essential to improve the quality of vehicle mobility prediction.

The probability threshold p_{th} that is used by Eq. (11) to generate a binary prediction vector also has an impact on the F1 scores of these algorithms. Figure 5(b) clarifies the reason behind this fact. When the threshold is high, these algorithms prefer to predict that a vehicle will not enter a grid even though its related probability is not low. This choice decreases false positive predictions to produce a high precision, while increasing false negative predictions to produce a low recall. Conversely, a low threshold tends to produce a high recall accompanied by a low precision. Since F1 score is defined as the harmonic average of precision and recall, it is maximized only when a threshold well balances both factors. Thus, an algorithm's highest F1 score is widely used to measure its prediction quality, and this metric also avoids the risk of comparing the F1 scores of different algorithms with unfair thresholds.

Figure 6 plots the prediction hamming losses of five algorithms with a prediction period of 10 min. It can be observed that the hamming losses of all algorithms are very small. This is due to the highly-skewed label distribution of vehicle mobility prediction problem. More specifically, since every vehicle usually only passes through a few of 2791 grids in Tokyo during a period of 10 min, most labels in the ground-truth vector of a data item are zeros and they are easy to be predicted, e.g., all algorithms can easily predict that a vehicle is unlikely to enter any grid 20 km away in 10 min. Consequently, only a very small fraction of labels in the prediction vector are incorrect, and this leads to a small hamming loss for all prediction algorithms. Nevertheless, compared with the best performed Markov model-based algorithm MM-3, DeepVM decreases its lowest hamming loss by 13.1%. This reduction is very significant when compared with other algorithms because MM-3 only decreases the lowest hamming loss of the worst performed MM-1 by 7.9%. Similar to the previous discussions on F1 score results, the hamming losses of different algorithms also vary with the probability threshold for generating binary prediction vector. As a result, an algorithm's lowest hamming loss is often used to evaluate its prediction quality. As shown in Figures 5(a) and 6, the optimal thresholds for F1 score and hamming loss are usually not the same. Thus, it is common for different applications to choose different metrics and thresholds according to their own needs.

Figure 7 plots the highest prediction F1 scores of five algorithms with different prediction periods. Similar to the pre-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2936507, IEEE Transactions on Industrial Informatics

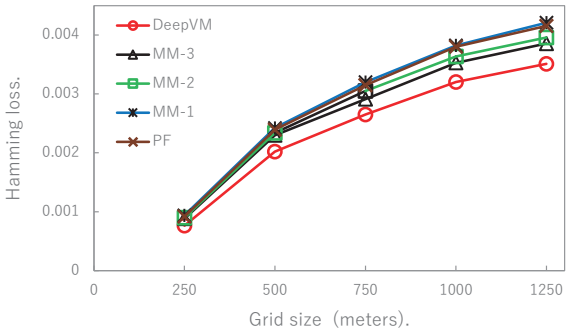


Fig. 10: The lowest hamming losses of five algorithms with a grid size of 250, 500, 750, 1000, and 1250 m.

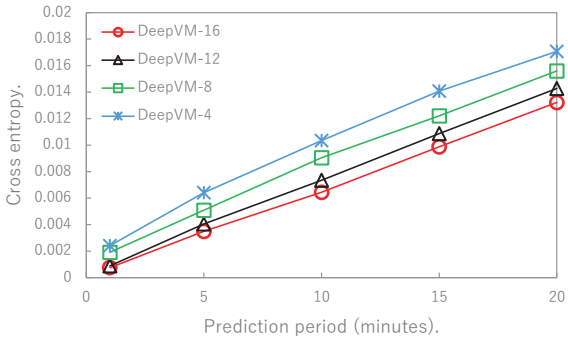


Fig. 11: The performance of DeepVM with different orders of vehicular trajectory.

vious results, Markov model-based algorithms perform better when the length of their input vehicular trajectory increases, e.g., the best-performed Markov model-based algorithm MM-3 outperforms MM-1 by a range of 9.3% $\sim$ 25.0%. It is interesting to note that the performance gap between PF and MM-1 shrinks with the increase of prediction period. This indicates the additional kinetic features of PF, i.e., the instant velocity and acceleration of a vehicle, better determine short-term vehicle mobility rather than long-term one. Because DeepVM can process a much longer vehicular trajectory than other algorithms, it outperforms MM-3 by a range of 18.3% $\sim$ 24.6%. It is worth to note that the superiority of DeepVM becomes more significant with the increase of prediction period, e.g., it outperforms MM-3 by 18.3% with a prediction period of 1 min, while that performance gap increases to 24.6% and 24.3% with a prediction period of 15 and 20 min. This trend is consistent to our theoretical findings presented in Sect. IV that a long vehicular trajectory can reduce the conditional entropy of vehicle mobility, especially when the concerned prediction period becomes long. Similar results are observed in Figure 8 that plots the lowest hamming losses of algorithms with different prediction periods. Compared with MM-3, DeepVM decreases the lowest hamming loss by a range of 10.4% $\sim$ 20.0% and the amount of reduction increases with prediction period.

Figure 9 plots the highest prediction F1 scores of five algorithms with different grid sizes. A larger grid size decreases the number of grids in the city space and makes vehicle mobility prediction easier. Thus, the F1 scores of all algorithms increase with grid size. It is worth to note that the change of grid size has few impact on the fact that a long vehicular trajectory helps to reduce the uncertainty of vehicle mobility prediction. Consequently, DeepVM still performs better than other algorithms, e.g., it improves the highest F1 score of the second best algorithm MM-3 by a range of 16.7% $\sim$ 23.8%. Figure 10 plots the lowest hamming losses of algorithms with different grid sizes. It is a little counter-intuitive to observe that the hamming losses of algorithms increase with grid size because it seems that the quality of vehicle mobility prediction decreases with the increase of grid size. Recall that the previous discussion related to Figure 6 reveals that there are many easily predictable grids of label

TABLE II: Contributions from RNN design factors.

	Cross-entropy	Training time
DeepVM	0.00644	3.3 hours
DeepVM with LSTM	0.00640	3.9 hours
DeepVM without embedding	0.00646	5.3 hours
DeepVM without residual link	0.00668	4.3 hours
DeepVM without dropout	0.00695	1.6 hours

zero because each vehicle can only passes through a few grids of a large city space during a prediction period of 10 min. With the increase of grid size, the fraction of these easily predictable grids decreases because the number of grids in the city space becomes less. Since hamming loss measures the fraction of incorrectly predicted labels and the reduction of easily predictable grids increases this fraction, the hamming losses of algorithms numerically increase with grid size and this is little related to the quality of vehicle mobility prediction. Nevertheless, with any fixed grid size, DeepVM performs best among five algorithms and decreases the lowest hamming loss of MM-3 by a range of 9.1% $\sim$ 13.6%.

B. Factor contributions of DeepVM

This part presents the performance contributions from different design factors of DeepVM. Since all the models discussed below are the variations of DeepVM and they are trained by minimizing the same binary cross-entropy loss function denoted by Eq. (10), these models are evaluated by their cross-entropy values on the test data set directly. A lower cross-entropy value indicates a better quality of vehicle mobility prediction.

Figure 11 explores the relationship between the length of vehicular trajectory used by DeepVM and its performance. It is clear that the performance of DeepVM improves with the length of its adopting vehicular trajectory, e.g., 8-order DeepVM (DeepVM-8) reduces the cross-entropies of 4-order DeepVM (DeepVM-4) by a range of 10.8% $\sim$ 20.8%, and the DeepVM with a 16-order vehicular trajectory (DeepVM-16) further reduces the cross-entropies of DeepVM-12 by 7.5% $\sim$ 15.1%. These results again illustrate that a long previous vehicular trajectory is essential to improve the quality of vehicle mobility prediction. Since the processing capability of DeepVM increases with its scale, a better performance may

be achieved by using a larger DeepVM that also requires more computational resources.

The neural network architecture of DeepVM also adopts some optimizing technologies of deep learning, i.e., vector embedding [24], GRU cells [26], residual links [27], and dropout [30]. Table II presents the contributions of these technologies from both views of algorithm performance and the length of training time. Benefitted from compacting a large-but-sparse input vector of grid identity to a small-and-dense one, the embedding method largely reduces the training time of DeepVM by 37.8%. By using GRU cells instead of LSTM cells, DeepVM reduces its training time by 15.4% while sacrificing its performance by 0.6%. As explained in Sect. V, the compact design of GRU cells makes DeepVM converge faster in training and has few impact on the final performance. The residual links of DeepVM reduce its training time by 23.3% and improve its performance by 3.6%. This is because residual links can not only accelerate the back-propagation training process of a neural network but also stabilize its training process to achieve a better convergence point [27]. Dropout improves the performance of DeepVM by 7.4%. However, since dropout only activates a fraction of neurons during training DeepVM, it also significantly increases the training time of DeepVM and this side-effect is also mentioned in its original paper [30].

VII. Conclusions and Discussions

This paper proposes a deep learning-based vehicle mobility prediction algorithm called DeepVM to support intelligent vehicle applications. A theoretical analysis is first given to show that a long vehicular trajectory helps to reduce the uncertainty of future vehicle mobility. Based on the knowledge earned from theoretical analysis, DeepVM uses a deep recurrent neural network to predict vehicle mobility. Comprehensive evaluations have proved that DeepVM can largely improve the quality of vehicle mobility prediction, and this superiority mainly comes from its ability to process a much longer vehicular trajectory than other state-of-art algorithms.

Our work is only the first step toward utilizing deep learning technology to predict vehicle mobility, and DeepVM may be further discussed in the following directions:

(1) Improving the learning strategy of DeepVM: DeepVM trains a vehicle mobility prediction model in a centralized manner, i.e., it collects many vehicles' mobility data in a data center and uses a deep neural network to process these big data. The evaluation results presented in this paper have shown that a DeepVM model that is trained in a few hours retains its prediction quality in a future period of 2 weeks, and its parameter size is less than 10 megabytes. Thus, it is feasible for an intelligent vehicle to locally cache a pre-trained DeepVM model and update its parameters from data center periodically. Different from the centralized strategy, a distributed learning strategy trains a mobility prediction model by only using the local processing system and mobility data of every vehicle separately. Both strategies have their advantages and disadvantages, i.e., the centralized learning strategy emphasizes on mining the cumulative mobility patterns of many different vehicles by using the plentiful data and computational resources in data center, while the distributed learning strategy may be more effective on extracting each vehicle's unique mobility preference even with restricted data and computational resources. It is interesting to clarify and utilize this trade-off to improve the quality of vehicle mobility prediction.

(2) Customizing DeepVM to fit new applications: Vehicle mobility prediction should facilitate various intelligent vehicle applications like sensing data collection and mobile edge computing. These applications may have different requirements for vehicle mobility prediction such as the scale of vehicles for mobility prediction and the range of vehicle movement. On one hand, it may be necessary to tune the hyper-parameters of DeepVM like the length of its input vehicular trajectory and its dropout ratio to fit new applications. On the other hand, when the scale of vehicles for mobility prediction becomes large such as tens of thousands of vehicles, it is interesting to work on the parallelization of DeepVM to retain its computational time at a reasonable level. Nevertheless, the key theoretical and empirical findings presented in this paper remain the same in these applications, i.e., a long vehicular trajectory helps to reduce the uncertainty of future vehicle mobility, and DeepVM significantly improves prediction quality by processing a much longer vehicular trajectory than any solution before. Hence, it is interesting to apply DeepVM and its variations to different intelligent vehicle applications in the near future.

References

[1] "Study on lte-based v2x services (release 14)." Tech. Specification Group Serv. Syst. Aspects (TSG SA), 3GPP TR 36.885, 2015.

[2] Gartner, "Forecast: Connected car production, worldwide," 2016.

[3] M. Bonola, L. Bracciale, P. Loreti, R. Amici, A. Rabuffi, and G. Bianchi, "Opportunistic communication in smart city: Experimental insight with small-scale taxi fleets as data carriers," *Ad Hoc Networks*, vol. 43, pp. 43–55, 2016.

[4] J. E. Siegel, D. C. Erb, and S. E. Sarma, "A survey of the connected vehicle landscapearchitectures, enabling technologies, applications, and development areas," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 8, pp. 2391–2406, 2018.

[5] A. Fox, B. V. Kumar, J. Chen, and F. Bai, "Multi-lane pothole detection from crowdsourced undersampled vehicle sensor data," *IEEE Transactions on Mobile Computing*, vol. 16, no. 12, pp. 3417–3430, 2017.

[6] B. Lonc and P. Cincilla, "Cooperative its security framework: Standards and implementations progress in europe," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2016 IEEE 17th International Symposium on A*. IEEE, 2016, pp. 1–6.

[7] J. He, L. Cai, P. Cheng, and J. Pan, "Delay minimization for data dissemination in large-scale vanets with buses and taxis," *IEEE Transactions on Mobile Computing*, vol. 15, no. 8, pp. 1939–1950, 2016.

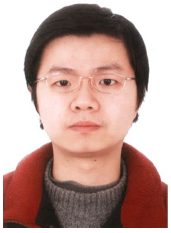
[8] N. Cheng, N. Lu, N. Zhang, T. Yang, X. S. Shen, and J. W. Mark, "Vehicle-assisted device-to-device data delivery for smart grid," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 4, pp. 2325–2340, 2016.

[9] Y. Zhu, Y. Wu, and B. Li, "Trajectory improves data delivery in urban vehicular networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 4, pp. 1089–1100, 2014.

[10] Z. Lin, Y. Lai, X. Gao, G. Li, T. Wang, and G. Huang, "Data gathering in urban vehicular network based on daily movement

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2936507, IEEE Transactions on Industrial Informatics

patterns," in *Computer Science & Education (ICCSE), 2016 11th International Conference on*. IEEE, 2016, pp. 641–646.
[11] S. Ucar, S. C. Ergen, and O. Ozkasap, "Multihop-cluster-based ieee 802.11 p and lte hybrid architecture for vanet safety message dissemination," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 4, pp. 2621–2636, 2016.
[12] R. Molina-Masegosa and J. Gozalvez, "Lte-v for sidelink 5g v2x vehicular communications: A new 5g technology for short-range vehicle-to-everything communications," *IEEE Vehicular Technology Magazine*, vol. 12, no. 4, pp. 30–39, 2017.
[13] W. Liu, R. Shinkuma, and T. Takahashi, "Opportunistic resource sharing in mobile cloud computing: The single-copy case," in *The 16th Asia-Pacific Network Operations and Management Symposium*. IEEE, 2014, pp. 1–6.
[14] K. Lee, Y. Yi, J. Jeong, H. Won, I. Rhee, and S. Chong, "Max-contribution: On optimal resource allocation in delay tolerant networks," in *INFOCOM, 2010 Proceedings IEEE*. IEEE, 2010, pp. 1–9.
[15] W. Liu and Y. Shoji, "Applying deep recurrent neural network to predict vehicle mobility," in *2018 IEEE Vehicular Networking Conference (VNC)*. IEEE, 2018, pp. 1–6.
[16] A. Agarwal and S. R. Das, "Dead reckoning in mobile ad hoc networks," in *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, vol. 3. IEEE, 2003, pp. 1838–1843.
[17] L. N. Balico, H. A. Oliveira, E. L. Souza, R. W. Pazzi, and E. F. Nakamura, "On the performance of localization prediction methods for vehicular ad hoc networks," in *Computers and Communication (ISCC), 2015 IEEE Symposium on*. IEEE, 2015, pp. 359–364.
[18] N. Aljeri and A. Boukerche, "Performance evaluation of movement prediction techniques for vehicular networks," in *Communications (ICC), 2017 IEEE International Conference on*. IEEE, 2017, pp. 1–6.
[19] X. Song, H. Kanasugi, and R. Shibasaki, "Deeptransport: Prediction and simulation of human mobility and transportation mode at a citywide level." in *IJCAI*, vol. 16, 2016, pp. 2618–2624.
[20] J. Zhang, Y. Zheng, D. Qi, R. Li, and X. Yi, "Dnn-based prediction model for spatio-temporal data," in *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 2016, p. 92.
[21] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: a deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2015.
[22] D. Li, L. Deng, Z. Cai, B. Franks, and X. Yao, "Intelligent transportation system in macao based on deep self coding learning," *IEEE Transactions on Industrial Informatics*, 2018.
[23] W. Liu, K. Nakauchi, and Y. Shoji, "A neighbor-based probabilistic broadcast protocol for data dissemination in mobile iot networks," *IEEE Access*, 2018.
[24] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.
[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[26] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.
[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
[28] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: a system for large-scale machine learning." in *OSDI*, vol. 16, 2016, pp. 265–283.
[29] G. Madjarov, D. Kocev, D. Gjorgjevikj, and S. Džeroski, "An extensive experimental comparison of methods for multi-label learning," *Pattern recognition*, vol. 45, no. 9, pp. 3084–3104, 2012.
[30] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.



Wei Liu (M'19) received the B.E. and M.E degrees in software engineering from Chongqing University, China, in 2006 and 2009, respectively, and the Ph.D. degree in communications and computer engineering from Kyoto University, Japan, in 2015. He was with PICC Corporation as an ICT Engineer. Since 2015, he has been currently a Researcher with the National Institute of Information and Communication Technology, Japan. His research interests include edge computing, IoT, machine learning, and smart city applications. He received the Best Paper Award for his paper at IEEE CCWC 2017 and Young Author Recognition for his paper at ITU Kaleidoscope 2013.

Yozo Shoji (S'98-M'99) joined the Communications Research Laboratory (CRL), Ministry of Posts and Telecommunications, Japan, in 1999. Since then, he has researched millimeter-wave and optical communications systems and made a lot of contributions to the standardization for the 60-GHz band in IEEE802.15.3c. In 2000, he invented the millimeter wave self-heterodyne system and succeeded in a 60-GHz band wireless transmission of OFDM-based digital TV broadcast signals for the first time in the world. In 2010, he was awarded the Excellent Young Researchers Overseas Visit Program Fellowship by the Japan Society for the Promotion of Science (JSPS) and spent one year as a Visiting Researcher of University College London (UCL), U.K. Dr. Shoji is currently the director of the Social-ICT System Laboratory, NICT (formerly CRL) and engaging the research for the community-based wireless IoT network system. He is a Senior Member of the Institute of Electrical, Information and Communication Engineers (IEICE), Japan. He was the recipient of the IEICE Young Researchers Award (2000), IEICE Electronics Society: Electronics Society Award (2007), Young Scientists Prize in the Commendation for Science and Technology by the Minister of Education, Culture, Sports, Science and Technology (2008), and the Meritorious Award on Radio by the Association of Radio Industries and Businesses (ARIB) (2010).