Travel Demand Prediction using Deep Multi-Scale Convolutional LSTM Network

Conference Paper · November 2018

DOI: 10.1109/ITSC.2018.8569427

3 authors:

Kai Fung Chu
The University of Hong Kong
6 PUBLICATIONS **16** CITATIONS
SEE PROFILE

Albert Lam
The University of Hong Kong
76 PUBLICATIONS **1,959** CITATIONS
SEE PROFILE

Victor O. K. Li
The University of Hong Kong
634 PUBLICATIONS **11,714** CITATIONS
SEE PROFILE

Travel Demand Prediction using Deep Multi-Scale Convolutional LSTM Network

Kai Fung Chu*†, Albert Y.S. Lam*† and Victor O.K. Li*†
*Department of Electrical and Electronic Engineering
The University of Hong Kong
Pokfulam Road, Hong Kong
Email: {kfchu, ayslam, vli}@eee.hku.hk
†The University of Hong Kong Shenzhen Institute of Research and Innovation

***Abstract*—Mobility on Demand transforms the way people travel in the city and facilitates real-time vehicle hiring services. Given the predicted future travel demand, service providers can coordinate their available vehicles such that they are pre-allocated to the customers' origins of service in advance to reduce waiting time. Traditional approaches on future travel demand prediction rely on statistical or machine learning methods. Advancement in sensor technology generates huge amount of data, which enables the data-driven intelligent transportation system. In this paper, inspired by deep learning techniques for image and video processing, we propose a new deep learning model, called Multi-Scale Convolutional Long Short-Term Memory (MultiConvLSTM), by considering travel demand as image pixel values. MultiConvLSTM considers both temporal and spatial correlations to predict the future travel demand. Experiments on real-world New York taxi data with around 400 million records are performed. We show that MultiConvLSTM outperforms the existing prediction methods for travel demand prediction and achieves the highest accuracy among all in both one-step and multiple-step predictions.**

I. Introduction

Advancement in sensing and Internet of Things (IoT) technologies enable analysis of our physical world with huge amount of data. These technologies have been widely deployed in transportation system, such as global positioning system (GPS) for vehicle localization and navigation [1], traffic flow estimation by inductive loop or camera to count and classify vehicles for traffic management [2], etc. With big data, the traditional technology-driven transportation system is transformed into data-driven intelligent transportation system [3]. Many vehicles are also equipped with various sensors and communication units to collect information. This facilitates many new applications, such as autonomous intersection management [4], dynamic lane reversal [5], and vehicular energy network [6].

Nowadays, we inevitably need to travel to different places for various activities. Traditionally, taxi drivers need to roam the streets to look for passengers by chance. In recent years, a more traveler-centric transportation system – Mobility on Demand (MoD), has emerged, such as Uber [7] and Lyft [8], which allows passengers to actively submit travel requests to specify their pickup locations. By incorporating autonomous driving into MoD, a new form of transportation – Autonomous MoD (AMoD), which utilizes autonomous vehicles as the transportation service carriers with more service options in smart city, has been proposed [9], [10]. However, The common shortcoming of all these transportation systems is that, after submitting their travel requests, passengers normally have to wait until the assigned vehicles arrive at the dedicated pickup locations. The waiting time varies with the distance between the original location of the assigned vehicle and the pickup point. If the vehicle supply and service demand are imbalanced, there may be no vehicle available nearby and a faraway vehicle may need to be assigned, resulting in long waiting time for the passenger. The survey done for San Francisco [11] shows that most MoD and taxi calling services take less than 10 minutes and 10–20 minutes waiting time, respectively. The situation becomes much worse during rush hours or in a congested city. If vehicles can be pre-allocated to those areas with high travel demand, the waiting time will be much reduced and the service quality will be improved. Hence, the location and time information of travel demand in the city is valuable.

In order to improve the traffic conditions, such as reducing the number of unoccupied vehicles on the road and minimizing the energy consumption and air pollution from vehicle emission, developing a re-balancing strategy is promising for the traffic environment. However, future information of travel demand is generally unknown. Predicting travel demand is challenging since there are limited rules guiding the complicated human movement. Thanks to the available big data and abundant computation resources, the data-driven deep learning approach becomes feasible to predict travel demand. When the future travel demand can be predicted, a re-balancing strategy can be performed for AMoD in the smart city.

Several studies tried to analyze the taxi GPS data to obtain the origin-destination pattern [12]. Some researchers investigated a recommender system to maximize the profit of taxi [13]. However, these approaches work for a single driver only while the positions of other drivers are ignored. Several existing works do focus on demand prediction. Autoregressive integrated moving average (ARIMA) [14] is well-known for addressing the short-term time-series prediction problem and ARIMA is traditionally used to predict the travel demand

[15]. In recent years, due to the success of deep learning [16] in tackling various challenging tasks [17], [18], [19], it has been applied to address traffic problems. For example, [20] used a deep neural network to predict traffic flow with high accuracy. [21] applied the Restricted Boltzmann Machine (RBM) and Recurrent Neural Network (RNN) to predict traffic congestion evolution. In particular, [22] used RNN to predict taxi demand. However, solely using RNN to handle the sequential nature of taxi demand data may overlook its spatial correlation. We need a model that considers both the temporal and spatial correlations of the historical data to predict the future travel demand.

In this paper, we investigate both the temporal and spatial correlations to predict the travel demand. To do this, we "visualize" the travel demands in the city at a particular moment as an image, where each small area is considered as a pixel and its travel demand as the corresponding pixel value. Inspired by deep learning technique for image and video processing, a deep learning model called Multi-Scale Convolutional Long Short-Term Memory network (MultiConvLSTM) is proposed to address the problem. MultiConvLSTM takes historical demand data as input and outputs future demand. It performs convolution between the input and weights, and stores extracted feature as in LSTM. Historical data of different spatial resolutions and metadata, such as time and weather information, can be input to MultiConvLSTM to further enhance the prediction accuracy. Experiments are conducted to evaluate the proposed deep learning model with six and a half years (Jan. 2009 – Jun. 2015) of real taxi data, which contain around 400 million records for a selected region of Manhattan, New York. Experimental results show that the proposed method outperforms the existing methods.

The rest of this paper is organized as follows. Section II defines the travel demand prediction problem and reviews existing deep learning models. We present our proposed deep learning model in Section III and experiments on real-world trip data is given in Section IV. Finally, Section V concludes this paper.

II. Background

In this section, we first discuss the travel demand prediction problem and then introduce some related deep learning models.

A. Travel demand prediction problem

We aim to predict the travel demand for a specific region A, which is equally divided into $n \times m$ grid cells. The travel demand, $d_{i,t}$, is defined as the total number of travel requests with pickup location in the ith grid cell in the tth time interval. The time interval will typically be in minutes. While historical travel demands are the major source of information for the prediction, some other metadata, such as the day, time, and weather, may also have correlations with travel demand. For example, higher demand is expected during the rush hours in working days. Thus they may optionally be considered to further improve the prediction accuracy.

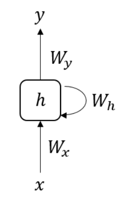

Fig. 1. RNN architecture.

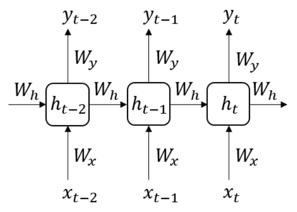

Fig. 2. Unfolded RNN architecture.

The objective of the problem is to predict the demand of each grid cell in A for the future $k \geq 1$ intervals. Suppose we are in the Tth interval. We predict the travel demands, $d_{i,T+1}, \ldots, d_{i,T+k}$, based the historical travel demand data $D = \{d_{j,t} \forall j \in A, t \in 1, 2, ..., T\}$ and other optional metadata.

B. Deep learning Models

Before we discuss our proposed deep learning model in Section III, two related popular neural network architectures are presented for background information.

1) Recurrent Neural Network (RNN): Predicting travel demand is similar to sequence prediction and thus the RNN may be a good model to handle time-series data in travel demand prediction. It makes use of a series of historical data as input rather than a single input, and it is capable of storing some extracted information of previous inputs in its memory for later prediction. This is achieved by recurrently computing all elements of the input sequence to output the results. A loop is used to pass the information from one time step to the next. Fig. 1 shows a typical structure of RNN with input x, hidden state h, output y, and the corresponding weights W_x, W_h, W_y. They are usually scalars or vectors in an LSTM. Matrix can be used after being flattened to vectors. In practice, the infinite loop can be unfolded to a finite number of input time steps as shown in Fig. 2. The hidden state h_t at time t can be considered as the memory of the network, computed as follows:

$$h_t = f(W_h h_{t-1} + W_x x_t), \tag{1}$$

where f is an activation function such as $tanh$ or $ReLU$.

A popular kind of RNN is Long Short-Term Memory (LSTM) [23], which is capable of learning long-term dependencies. The main difference is the inner structure of the repeating cell. Recall that inside the cell of standard RNN, there is a single $tanh$ or $ReLU$ layer only. However, In LSTM, a memory cell state, c_t, is added and there are four layers interacting together as shown in Fig. 3. A "Forget gate layer", f_t, is used to decide the information from the previous cell that should be forgotten. An "Input gate layer", i_t, is for deciding which part of the information from the current input and hidden state should be stored. We use a $tanh$ layer to extract the information to the new cell state. An "Output gate layer", o_t, acts as a filter to extract the information from the cell state to produce the output. More specifically, f_t, i_t, c_t, o_t, and h_t are updated as follows:

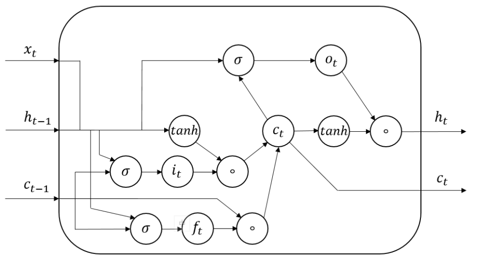

Fig. 3. LSTM architecture.

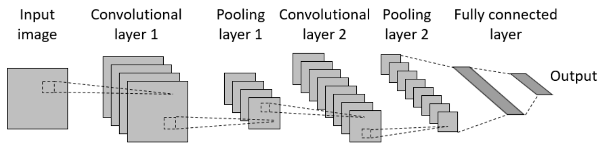

Fig. 4. CNN architecture.

$$f_t = \sigma(W_{fc} \circ c_{t-1} + W_{fh}h_{t-1} + W_{fx}x_t + b_f), \tag{2}$$
$$i_t = \sigma(W_{ic} \circ c_{t-1} + W_{ih}h_{t-1} + W_{ix}x_t + b_i), \tag{3}$$
$$c_t = f_t \circ c_{t-1} + i_t \circ tanh(W_{ch}h_{t-1} + W_{cx}x_t + b_c), \tag{4}$$
$$o_t = \sigma(W_{oc} \circ c_t + W_{oh}h_{t-1} + W_{ox}x_t + b_o), \tag{5}$$
$$h_t = o_t \circ tanh(c_t), \tag{6}$$

where $\circ$ denotes the Hadamard product.

2) Convolution Neural Network: Recall that the travel demand data, D, contain a series of historical data for different time intervals. In each interval t, $d_{i,t}$ represents the total number of travel requests where pickup location is in ith grid cell. Adjacent grid cells may have certain spatial correlations because neighboring locations may share similar travel demand patterns. Handling the travel demand with RNN-LSTM alone may lead to overlooking of this spatial information hidden in the data. One popular architecture to capture this high dimensional spatial information is the convolutional neural network (CNN), which has been applied to many challenging computer vision problems, such as object detection and recognition, with great success. CNN consists of a number of convolutional and pooling layers followed by a fully connected layer at the end of the network. A convolutional layer is used to extract high-dimension features. It has k kernels of size n by n as filters, each of which computes the dot product between the weights of the kernel and a small part of the input until it slices through the whole image. A pooling layer performs subsampling of the image, where the max operation is a typical pooling method that selects the maxium value for each m-by-m pixel. The fully connected layer computes the output based on the extracted features from the convolutional and pooling layers. Fig. 4 shows a typical CNN for image classification.

III. Travel Demand Prediction Model

In this paper, we consider both the temporal and spatial correlations to predict travel demand. The basic unit used in our model is ConvLSTM [24]. ConvLSTM can take advantage of both temporal and spatial correlations of the data by performing convolution in the network and storing extracted feature as in an LSTM. However, ConvLSTM can only account for a short range of spatial correlation depending on the kernel size. Incorporating long range data is of benefit to the prediction since the network can be trained to identify useful information and ignore irrelevant information from long range data. To overcome this, we convert this model into a multi-scale network and propose the Multi-Scale Convolutional LSTM Network (MultiConvLSTM) for the travel demand prediction problem.

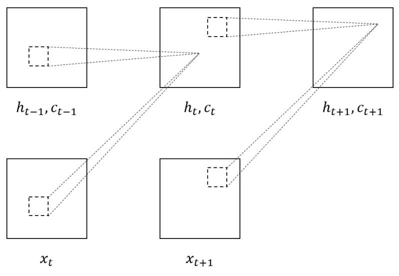

Fig. 5. ConvLSTM architecture

A. ConvLSTM

ConvLSTM preserves the ability of LSTM to store the extracted information and it contains hidden state and cell state units. In LSTM, the input and hidden states are multiplied by the weights. However, in ConvLSTM, instead of multiplying input and hidden states by the weights directly, we perform convolution with the weights as well. Fig. 5 shows the architecture of ConvLSTM and the key equations are as follows:

$$f_t = \sigma(W_{fc} \circ c_{t-1} + W_{fh} * h_{t-1} + W_{fx} * x_t + b_f), \tag{7}$$
$$i_t = \sigma(W_{ic} \circ c_{t-1} + W_{ih} * h_{t-1} + W_{ix} * x_t + b_i), \tag{8}$$
$$c_t = f_t \circ c_{t-1} + i_t \circ tanh(W_{ch} * h_{t-1} + W_{cx} * x_t + b_c), \tag{9}$$
$$o_t = \sigma(W_{oc} \circ c_t + W_{oh} * h_{t-1} + W_{ox} * x_t + b_o), \tag{10}$$
$$h_t = o_t \circ tanh(c_t), \tag{11}$$

where $\circ$ and $*$ denote the Hadamard product and the convolution operator, respectively. Using ConvLSTM, the spatial correlation of data can also be considered in the prediction. Note that ConvLSTM can be reduced to RNN-LSTM by setting the kernel size to 1×1.

B. Multi-Scale Network

The pooling layer is important for CNN to extract the features from the input. However, the pooling function is missing in ConvLSTM since the output of ConvLSTM has the same resolution as the input and the data flow in ConvLSTM is similar to that in LSTM. Therefore, without the pooling layer, convolution in ConvLSTM can only account for short-range spatial correlation depending on the kernel size. To address this issue, we convert the model into a multi-scale network by inputting lower-resolution versions of the same set of data [25]. A pixel in lower resolution data is combined with several nearby pixels of higher resolution data. Hence, with the same kernel size, more pixels can be considered, which is similar to a pooling layer. Let θ_r be the resolutions of the input data for $r = 1, 2, ..., R$, where r and R are the resolution index and the highest resolution index, respectively. Let $X^{\theta_{r-1}}$ and $Y^{\theta_{r-1}}$ be the lower resolution data sets of X^{θ_r} and Y^{θ_r}, respectively, where $X^{\theta_r} = \{x^{\theta_r}_{T-K}, ..., x^{\theta_r}_{T-1}, x^{\theta_r}_{T}\}$ is the historical data and $Y^{\theta_r} = \{y^{\theta_r}_{T+1}\}$ is the future value. We aim to predict future demand, Y^{θ_R}, with input data in different resolutions, $\{X^{\theta_1}, X^{\theta_2}, ..., X^{\theta_R}\}$. The prediction of Y^{θ_R} is computed by:

$$\widehat{Y}^{\theta_R} = ConvLSTM(X^{\theta_R}, up(\widehat{Y}^{\theta_{R-1}})), \tag{12}$$

where $ConvLSTM(\cdot)$ and $up(\cdot)$ are the ConvLSTM network and upsampling operation, respectively.

C. *MultiConvLSTM*

We develop our deep learning model MultiConvLSTM for travel demand prediction by combining ConvLSTM with the multi-scale network and Fig. 6 shows its architecture. The basic unit in the model is ConvLSTM, in which there are multiple ConvLSTMs for different resolutions. They form a hierarchical structure such that the output of lower-resolution ConvLSTM is fed to the input of the next level for higher resolution. Metadata such as time and weather information can also be input and a fully connected neural network (FCNN) is used to convert the metadata into τ^{θ_R} of the same dimension as the predicted demand, concatenated with the input data. In order to overcome the degradation of such a deep network, we employ the technique from deep residual learning [26], in which τ^{θ_R} and $up(\widehat{Y}^{\theta_{R-1}})$ are added to the output with 0.5 weighting. This allows ConvLSTM to model the residual, which is easier to be optimized in the training process, instead of the original predicted future demand. Although MultiConvLSTM can only predict one future time-step in each execution, subsequent time-steps can be predicted by using historical data and the recently predicted demand as the input.

IV. EXPERIMENTS

We evaluate the proposed MultiConvLSTM deep learning model using the New York taxi dataset [27], which contains the pickup and dropoff times, and pickup and dropoff locations of taxi services recorded in the New York from January 2009 to June 2015. A region in Manhattan selected for the study is shown in Fig. 7. The region is divided into 28×20 grid cells, each of which has an area of 220×170 m^2. There are around 400 million taxi traveling records and the data are converted into travel demands. The duration of one time-step is set to 10 minutes. We extract 80% of the data as training data while the rest are for testing.

Two commonly used performance metrics are used in our study: root mean square error (RMSE) and symmetric mean absolute percentage error (SMAPE), which are computed as follows:

$$RMSE(y, \hat{y}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y - \hat{y})^2} \tag{13}$$

$$SMAPE(y, \hat{y}) = \frac{1}{N}\sum_{i=1}^{N}\frac{|y_i - \hat{y}_i|}{y_i + \hat{y}_i + c}, \tag{14}$$

where c is a positive constant to handle the division by zero when both y_i and $\hat{y}_i$ are zero. To simplify it, we follow the commonly used practice and consider it as $c = 1$ [28].

The MultiConvLSTM used in the experiment required 12 time-steps input of historical data. The kernel size is 3×3. Three levels of resolutions are used, including 28×20, 14×10 and 7×5. The following models are also used to compare with MultiConvLSTM to evaluate the performance:

1. Autoregressive Integrated Moving Average (ARIMA) [14]: a well known statistical analysis method for time series data.
2. Fully Connected Neural Network (FCNN) [29]: two hidden layers with 256 hidden units in each layer.
3. CNN [17]: Four convolutional layers with filters size of 16, 32, 16, 1. The kernel size is 3×3.
4. RNN-LSTM [23]: 12 time-step input and the input data of each time-step composed of demand data and metadata.
5. ConvLSTM [24]: 12 time-steps input and 3×3 kernel size.

The activation function used in the models is *RELU* except the one in between fully connected layers which is *sigmoid*. Mean square error (MSE) is used as the loss function for all the models except ARIMA for training:

$$MSE(y, \hat{y}) = \frac{1}{N}\sum_{i=1}^{N}(y - \hat{y})^2, \tag{15}$$

where N is the total number of grid cells. The training algorithm is Adam optimization [30] with 0.001 learning rate. The batch size is set to 50 and the number of training steps is set to 100,000. The metadata, represented as a 5-tuple vector, contains time information only: ⟨month, day, hour, minute, days of week⟩.

A. *Comparison of prediction methods*

Table I shows the RMSE and SAMPE results of different methods for predicting one future time-step, i.e., $k = 1$. ARIMA provides the baseline performance in time-series analysis. Although FCNN and CNN are powerful for classification in general, their performances are not satisfactory for travel demand prediction. We can see that MultiConvLSTM achieves the lowest RMSE (1.5405) and SAMPE (0.1663) among all the methods. ConvLSTMand RNN-LSTM have similar performance. As discussed in Section III-A, ConvLSTM can be reduced to RNN-LSTM by setting the kernel size to 1×1. The kernel size is set as 3×3 of ConvLSTM in this experiment which only accounts for short range dependence and thus, the performances of RNN-LSTM and

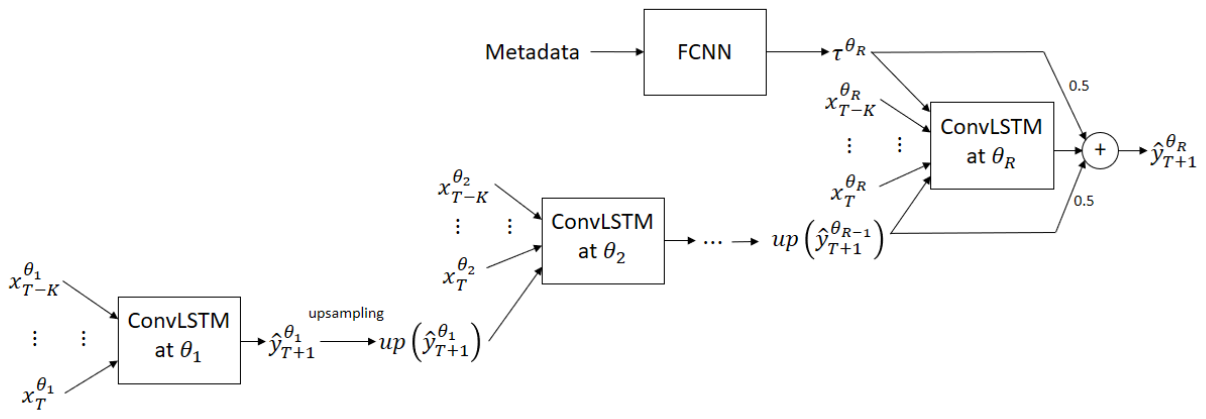

Fig. 6. MultiConvLSTM architecture

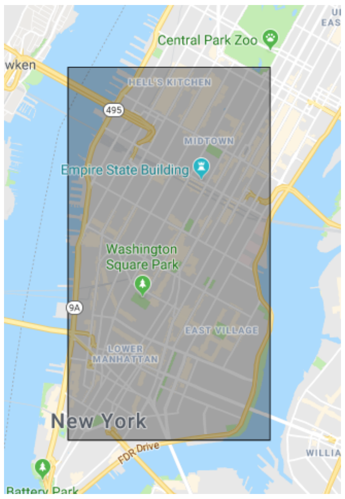

Fig. 7. Selected region in Manhattan.

TABLE I
ERRORS OF DIFFERENT PREDICTION MODELS FOR $k = 1$.

	RMSE	SMAPE
FCNN	1.9609	0.2140
CNN	1.8561	0.1949
ARIMA	1.7671	0.1861
RNN-LSTM	1.6243	0.1766
ConvLSTM	1.6485	0.1776
MultiConvLSTM	**1.5405**	**0.1663**

ConvLSTM are similar. The performance is further improved with MultiConvLSTM since it can account for long-range dependence even with the same kernel size of 3 $\times$ 3.

B. *Multi-step prediction*

The temporal resolution in our experiment is 10 minutes which may not be enough for certain applications. Demand in further future may also be more useful for certain applications. Subsequent time-steps can be predicted by using historical data and the recently predicted demand as the input. Figs. 8 and 9 show the multi-step prediction errors,

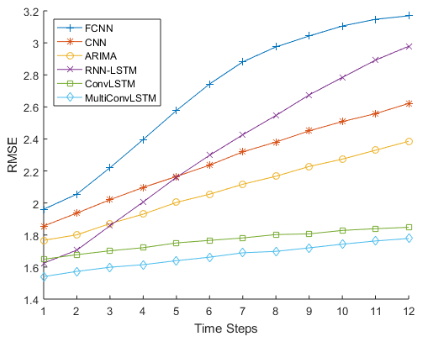

Fig. 8. RMSE of multi-step prediction.

RMSE and SMAPE of different models, respectively, in which 12 future time steps (a total of 2 hours) are considered for all methods. Table II shows the average RMSE and SMAPE for 12 time steps of different models. In general, the errors of all models increase with more time-steps due to error accumulation. Our proposed model, MultiConvLSTM, achieves the lowest RMSE and SMAPE in prediction for all time-steps. While RNN-LSTM is the second best in one-step prediction, the error increases and accumulates dramatically with more time-steps. It is because RNN-LSTM focuses on temporal prediction only. Even metadata are concatenated with demand data as the input, the prediction still depends mainly on the historical data. When noise and error are present in the historical data, RNN-LSTM cannot identify and filter out the noise. As a result, the error accumulates in multi-step prediction. On the other hand, the errors of MultiConvLSTM and ConvLSTM increase sightly in multi-step prediction. They can extract both temporal and spatial features from historical data and metadata for prediction and account for the demand in time and space. Even though noise and error appear in historical data, their effect can be suppressed.

V. CONCLUSION

In most existing on-demand transportation services, passengers have to wait until assigned vehicles arrive at the

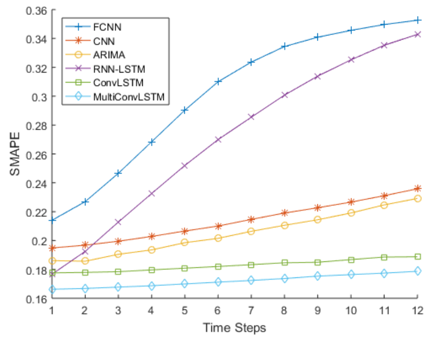

Fig. 9. SMAPE of multi-step prediction.

TABLE II
AVERAGE ERROR FOR 12 STEPS PREDICTION OF DIFFERENT PREDICTION MODELS

	RMSE	SMAPE
FCNN	2.6893	0.3002
CNN	2.2625	0.2134
ARIMA	2.0774	0.2050
RNN-LSTM	2.3295	0.2700
ConvLSTM	1.7645	0.1828
MulitConvLSTM	**1.6683**	**0.1720**

passengers' pick up points. In order to reduce the unnecessary waiting time, we propose a new deep learning model, MultiConvLSTM, to accurately predict the future travel demand. MultiConvLSTM is based on the multi-scale network and Convolutional LSTM to extract both temporal and spatial features from historical data and metadata for predicting the future demand. Experiments on real-world New York transportation dataset with around 400 million records show that MultiConvLSTM outperforms the existing prediction methods. MultiConvLSTM also performs the best when compared to other existing methods for one-step and multi-step prediction for predicting the demand in further future.

In the future, weather data will be used as additional metadata for further improving the prediction accuracy.

ACKNOWLEDGMENT

This work was partially supported by The National Natural Science Foundation of China (NSFC) 51707170.

REFERENCES

[1] S. Sukkarieh, E. M. Nebot, and H. F. Durrant-Whyte, "A high integrity imu/gps navigation loop for autonomous land vehicle applications," *IEEE Trans. Robot. Autom.*, vol. 15, no. 3, pp. 572–578, June 1999.
[2] S. S. M. Ali, B. George, L. Vanajakshi, and J. Venkatraman, "A multiple inductive loop vehicle detection system for heterogeneous and lane-less traffic," *IEEE Trans. Instrum. Meas.*, vol. 61, no. 5, pp. 1353–1360, May 2012.
[3] J. Zhang, F. Y. Wang, K. Wang, W. H. Lin, X. Xu, and C. Chen, "Data-driven intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1624–1639, Dec 2011.
[4] K. Dresner and P. Stone, "A multiagent approach to autonomous intersection management," *J. of artificial intell. res.*, vol. 31, pp. 591–656, 2008.
[5] K. F. Chu, A. Y. S. Lam, and V. O. K. Li, "Dynamic lane reversal routing and scheduling for connected autonomous vehicles," in *Proc. of The 3rd IEEE Annu. Int. Smart Cities Conf.*, Sept. 2017, pp. 1–6.
[6] A. Y. S. Lam, K. C. Leung, and V. O. K. Li, "Vehicular energy network," *IEEE Trans. Transport. Electrific.*, vol. 3, no. 2, pp. 392–404, June 2017.
[7] Uber. (2018) Uber. [Online]. Available: https://www.uber.com/
[8] Lyft. (2018) Lyft. [Online]. Available: https://www.lyft.com/
[9] A. Y. S. Lam, "Combinatorial auction-based pricing for multi-tenant autonomous vehicle public transportation system," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 3, pp. 859–869, Mar. 2016.
[10] A. Y. S. Lam, Y. W. Leung, and X. Chu, "Autonomous-vehicle public transportation system: Scheduling and admission control," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 5, pp. 1210–1226, May 2016.
[11] L. Rayle, D. Dai, N. Chan, R. Cervero, and S. Shaheen, "Just a better taxi? a survey-based comparison of taxis, transit, and ridesourcing services in san francisco," *Transport Policy*, vol. 45, pp. 168 – 178, 2016.
[12] Z. Deng and M. Ji, "Spatiotemporal structure of taxi services in shanghai: Using exploratory spatial data analysis," in *Proc. of 2011 19th Int. Conf. on Geoinformatics*, June 2011, pp. 1–5.
[13] N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie, "T-finder: A recommender system for finding passengers and vacant taxis," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 10, pp. 2390–2403, Oct. 2013.
[14] G. E. P. Box and D. A. Pierce, "Distribution of residual autocorrelations in autoregressive-integrated moving average time series models," *J. of the Amer. Statistical Assoc.*, vol. 65, no. 332, pp. 1509–1526, 1970.
[15] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "Predicting taxi-passenger demand using streaming data," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1393–1402, Sept. 2013.
[16] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.
[18] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[19] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.
[20] Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 865–873, Apr. 2015.
[21] X. Ma, H. Yu, Y. Wang, and Y. Wang, "Large-scale transportation network congestion evolution prediction using deep learning theory," *PLOS ONE*, vol. 10, no. 3, pp. 1–17, 03 2015.
[22] J. Xu, R. Rahmatizadeh, L. Bölöni, and D. Turgut, "Real-time prediction of taxi demand using recurrent neural networks," *IEEE Trans. Intell. Transp. Syst.*, vol. PP, no. 99, pp. 1–10, 2017.
[23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[24] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Advances in Neural Information Processing Systems 28*, 2015, pp. 802–810.
[25] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," *arXiv preprint arXiv:1511.05440*, 2015.
[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. 2016 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 770–778.
[27] New York City. (2017, Oct.) New york city open data. [Online]. Available: https://opendata.cityofnewyork.us/
[28] E. T. Jaynes, *Probability theory: the logic of science*. Cambridge university press, 2003.
[29] R. J. Schalkoff, *Artificial neural networks*. McGraw-Hill New York, 1997, vol. 1.
[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.