Taylor & Francis
Taylor & Francis Group

Applied Artificial Intelligence
An International Journal

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/uaai20

Short-term Demand Forecasting for Online Car-hailing Services Using Recurrent Neural Networks

Alireza Nejadettehad, Hamid Mahini & Behnam Bahrak

To cite this article: Alireza Nejadettehad, Hamid Mahini & Behnam Bahrak (2020) Short-term Demand Forecasting for Online Car-hailing Services Using Recurrent Neural Networks, Applied Artificial Intelligence, 34:9, 674-689, DOI: 10.1080/08839514.2020.1771522

To link to this article: https://doi.org/10.1080/08839514.2020.1771522

Published online: 04 Jun 2020.

APPLIED ARTIFICIAL INTELLIGENCE
2020, VOL. 34, NO. 9, 674–689
https://doi.org/10.1080/08839514.2020.1771522

Taylor & Francis
Taylor & Francis Group

Short-term Demand Forecasting for Online Car-hailing Services Using Recurrent Neural Networks

Alireza Nejadettehad, Hamid Mahini, and Behnam Bahrak

School of Electrical and Computer Engineering, College of Engineering, University of Tehran, Tehran, Iran

ABSTRACT

Short-term traffic flow prediction is one of the crucial issues in intelligent transportation system, which is an important part of smart cities. Accurate predictions can enable both the drivers and the passengers to make better decisions about their travel route, departure time, and travel origin selection, which can be helpful in traffic management. Multiple models and algorithms based on time-series prediction and machine learning were applied to this issue and achieved acceptable results. Recently, the availability of sufficient data and computational power motivates us to improve the prediction accuracy via deep-learning approaches. Recurrent neural networks have become one of the most popular methods for time-series forecasting; however, due to the variety of these networks, the question that which type is the most appropriate one for this task remains unsolved. In this paper, we use three kinds of recurrent neural networks including simple RNN units, GRU, and LSTM neural network to predict short-term traffic flow. The dataset from TAP30 Corporation is used for building the models and comparing RNNs with several well-known models, such as DEMA, LASSO, and XGBoost. The results show that all three types of RNNs outperform the others; however, more simple RNNs such as simple recurrent units and GRU perform work better than LSTM in terms of accuracy and training time.

Introduction

Online car-hailing apps have evolved as novel and popular services to provide on-demand transportation service via mobile apps. Comparing with the traditional transportation means such as the subways and buses, the online car-hailing service is much more convenient and flexible for the passengers. Furthermore, by incentivizing private car owners to provide car-hailing services, it promotes the sharing economy and enlarges the transportation capacities of the cities. Several car-hailing mobile apps have gained great popularities all over the world, such as Uber, Didi, and Lyft. A large number of passengers are served and a significant volume of car-hailing orders are generated routinely every day. For example, TAP30, one of the largest online

CONTACT Alireza Nejadettehad alireza_ettehad@ut.ac.ir School of Electrical and Computer Engineering, College of Engineering, University of Tehran, Tehran, Iran

car-hailing service providers in Iran, handles hundreds of thousands of orders per day all over Iran.

These platforms serve as a coordinator who matches requesting orders from passengers (demand) and vacant registered cars (supply). There exists an abundance of leverages to influence drivers' and passengers' preference and behavior, and thus affect both the demand and supply, to maximize profits of the platform or achieve maximum social welfare. Having a better understanding of the short-term passenger demand over different spatial zones is of great importance to the platform or the operator, who can incentivize drivers to the zones with more potential passenger demands, and improve the utilization rate of the registered cars. However, in metropolises like Tehran, it is common to see passengers seeking for taxicabs roadside while some taxi drivers are cruising idly on the street. This contradiction reveals the supply-demand disequilibrium with the following two scenarios: Scenario 1, demand exceeds supply, where passengers' needs would not be met in a timely response. Scenario 2, supply exceeds demand, where drivers would spend an overly long time in seeking for passengers. To solve the problem of disequilibrium, an overall prediction for passenger demand in different zones provides a global distribution of passengers, upon which providers of car-hailing services can adjust prices and dispatch policies of supply dynamically in advance. We define the taxi-demand prediction problem as follows: Given historical taxi-demand data in a region $\mathcal{R}$, we want to predict the number of ride requests that will emerge within $\mathcal{R}$ during the next time interval.

Over the past few decades, many data analysis models have been proposed to solve the short-term traffic forecasting problem, including probabilistic models (Yuan et al. 2013), time-series forecasting methods (Li et al. 2011; Moreira-Matias et al. 2013) and decision tree-based methods (Zhang et al. 2017). Recently approaches based on neural networks gained noticeable attention in studies related to traffic flow prediction (Mukai and Yoden 2012; Wang et al. 2017; Zhao et al. 2016). One of the most popular kinds of NNs in this context is Recurrent Neural Networks (RNNs) (Tian and Pan 2015; Zhao et al. 2017; Tian and Pan 2015). Since 2015, when (Tian and Pan 2015) proposed long-short term memory (LSTM) NNs for traffic flow prediction and showed that LSTMs (due to their excellent ability to memorize long-term dependencies) outperform other methods in this particular context, almost every study that attempted to use RNNs for demand prediction has utilized LSTMs (Zhao et al. 2017; Xu et al. 2018; Ke et al. 2017). In this paper, the performance of different types of RNNs is evaluated and compared with some other powerful methods such as eXtreme Gradient Boosting (XGBoost) (Chen et al. 2015) and least absolute shrinkage and selection operator (LASSO) (Tibshirani 1996) and also with each other. Experimental results demonstrate that RNNs outperform the other methods according to the metrics chosen for comparison; However, when it comes to the comparison between RNNs, Simple RNN units, and

Gated recurrent unit (GRU) defeat LSTM in terms of performance and computational (training) time.

The results obtained from experiments show that the best non-RNN method (XGBoost) reached error rates 3.78% and 40.8% according to RMSE and MAPE, respectively. However, these errors were reduced to 3.22% and 37.42% by simple RNN units. In addition to the fact that simple RNN units outperformed other non-RNN methods and LSTM, computation time required for simple RNN units is approximately 0.13 and 0.1 the time needed to train XGBoost and LSTM, respectively. Although the experimental results denote that simple RNN units and GRU perform nearly the same, there is a significant difference between their training time and simple RNN units train nearly 13 times faster than GRU.

Related Work

Although there have been many efforts to predict traffic flow using spatio-temporal data; the most related studies to the demand prediction problem show that the most implemented methods consist of probabilistic models such as Poisson (Yuan et al. 2013), time-series forecasting methods such as auto-regression integrated moving average (ARIMA) (Li et al. 2011; Moreira-Matias et al. 2013) and neural networks (Mukai and Yoden 2012; Zhao et al. 2016; Wang et al. 2017). Between the time-series forecasting methods, ARIMA is more prevalent because of its performance in short-term forecasting. (Li et al. 2011) presented an improved ARIMA-based method to forecast the spatial-temporal distribution of passengers in an urban environment. First, urban regions with high demand are detected; then, demand in next hour is predicted in those regions using ARIMA and finally, demand is forecasted using an improved ARIMA-based method that uses both time and type of the day. Moreira-Matias et al. (2013) propose the challenge that ARIMA is not necessarily the best method to forecast demand. They propose an end-to-end framework to predict the number of services that will happen at taxi stands by applying the time-varying Poisson model and ARIMA. Moreover, they used a sliding-window ensemble framework to originate a prediction by combining the prediction of each model's accuracy. The dataset was generated from 441 vehicles with 63 taxi stands in the city of Porto. Yuan et al. (2013) presented an algorithm based on Poisson model to recommend the most probable points to find passengers for taxi drivers in shortest time. Davis, Raina, and Jagannathan (2016) proposed a multi-level clustering technique to improve the accuracy of linear time-series model fitting, by exploring the correlation between adjacent Geo-hashes.

Recently, the success of deep learning in the fields of computer vision and natural language processing (LeCun, Bengio, and Hinton 2015; Krizhevsky, Sutskever, and Hinton 2012), motivated researchers to apply deep-learning

techniques on traffic prediction problems. Mukai and Yoden (2012) is one of the first studies that implemented NNs in order to forecast taxi demand. They have used a multi-layer perceptron to achieve this target. Zhao et al. (2016) introduced a new parameter named "Maximum predictability" showed that different predictors (Markov predictor (a probability-based predictive algorithm), the Lempel-Ziv-Welch predictor (a sequence-based predictive algorithm) and the Neural Network predictor (a predictive algorithm that uses machine learning)), perform differently according to the maximum predictability of a region. They showed that considering maximum predictability, in the regions with more random demand pattern, NNs perform better and in the regions with lower randomness in their demand pattern, Markov predictor beats the others. Wang et al. (2017) proposed an end-to-end framework named DeepSD, based on a novel deep neural network structure that automatically discovers the complicated supply-demand patterns in historical order, weather and traffic data, with minimal amount of hand-crafted features.

In 2015 (Tian and Pan 2015) proposed long-short term memory NNs (LSTMs) for traffic flow prediction and showed that LSTMs (due to their excellent ability to memorize long-term dependencies) perform better in comparison to the other methods in this particular context. Since then, almost every study that used Recurrent neural networks to predict demand used LSTMs (Zhao et al. 2017; Xu et al. 2018; Ke et al. 2017). In this paper, we are going to compare the performance of different types of RNNs and also evaluate their performance in comparison to some other powerful methods such as XGBoost and LASSO.

Material and Methods

In this section, first, we explain how we cleaned the dataset and prepared it for modeling. Second, the features used in the models are introduced and finally, three different types of recurrent neural networks that we have used as models are explained in detail.

Data Processing

The dataset used in this study is real-world data from TAP30 corporation ride requests from September 1 to December 20, 2017. The details of raw data taken from database are shown in Table 1.

The urban area is partitioned into 16×16 grids uniformly where each grid refers to a region. On the other hand, we consider variables aggregated in a 15-minute time interval in this paper. We have removed the ride requests canceled in 5 seconds, because there are not considered to be real demand and potentially are noisy data. Also, the ride requests that a passenger with his/her unique passenger id has made in a time interval of 15 minutes length are aggregated to become a single request. The number of unique ride requests made represents the demand. We

Table 1. Details of raw data.

Data type	Description
Ride Request ID	The unique ID of the ride request
Passenger ID	The unique ID of the passenger that made the ride request
Timestamp	Timestamp of the ride request
Latitude/Longitude	GPS location of origin of the ride request

aggregated the number of unique ride requests for all 256 regions, every 15 minutes. In order to obtain robust and interpretable results, we decided to consider only the regions that at least 300 ride requests per day on average (nearly 3 ride requests in each time interval on average) had been made in them. After eliminating the regions that do not satisfy our limit, 64 regions were left.

Features

There are 68 main features for the predictive model. Each data point in our final cleaned data has 4 temporal features and 64 spatial features.

Temporal Features

We have extracted four main temporal features from the time-stamps of the cleaned raw data. In order to use the continuous nature of the time-slot feature, first, we converted the timeslot number to triangular format and used its sine and cosine as features. Table 2 includes the temporal features and their description.

Spatial Features

Since there are correlations between the amount of demand in a region and the other regions, we used the amount of demand in all regions in the previous timeslots as features. For example to predict the demand in timeslot $t + 1$ in region number i, not only we used the demand in previous time-slots in that region, but also we used the demand in all other regions as features in our models.

Methods

In this section, we briefly describe our selected recurrent neural networks for the aforementioned task, which are Simple RNN, GRU (Gated recurrent unit), and LSTM (Long short-term memory).

Table 2. Temporal features.

Feature	Description
Day of week	The ID of the day of week
National holiday	Whether the day is a national holiday or not
Time slot Sineunus	$\sin(2\pi \times$ time-slot number$/96)$
Time slot Cosineus	$\cos(2\pi \times$ time-slot number$/96)$

Simple RNN

A recurrent neuron is a special kind of artificial neuron which has a backward connection to the neurons in previous layers. RNNs have internal memory which allows them to operate over sequential data effectively. This feature made the RNNs one of the most popular models for dealing with sequential tasks such as handwriting recognition (Graves et al. 2009), NLP (Graves, Mohamed, and Hinton 2013), and time-series forecasting (Connor, Martin, and Atlas 1994).

Figure 1 shows the structure of an RNN and Figure 2 illustrates an unrolled RNN and how it deals with sequential data. Given a sequence X = $\{x_1, x_2, x_3, \ldots, x_t\}$ as input, RNN computes the hidden state sequence H = $\{h_1, h_2, h_3, \ldots, h_t\}$ and output sequence Y = $\{y_1, y_2, y_3, \ldots, y_t\}$ using Equations (1) and (2).

$$h_t = f(W_{hx}x_t + W_{hh}h_{t-1} + b_h) \tag{1}$$

$$y_t = g(W_{yt}h_t + b_y) \tag{2}$$

In Equations (1) and (2) W_{hx}, W_{hh} and W_{yt} denote the input-to-hidden, hidden-to-hidden and hidden-to-output weight matrices, respectively. b_h and b_y are hidden layer bias and output layer bias vectors. $f(.)$ and $g(.)$ are the activation functions of the hidden layer and output layer, respectively. The hidden state of each time step is passed to the next time step's hidden state.

Long-Short Term Memory

Long-Short Term Memory networks are a special kind of RNN, capable of learning long-term dependencies. They were introduced by Hochreiter and Schmidhuber (1997), Hochreiter and Schmidhuber (1997), and were refined and popularized by many researchers in different contexts. LSTMs are explicitly designed to avoid the long-term dependency problem. In comparison to simple RNN, LSTM has a more complicated structure and contains three kinds of gates: input gate, forget gate, and cell state gate. Figure 3 illustrates an LSTM cell.

Forget gate: After getting the output of previous state, $h(t-1)$, forget gate helps to take decisions about what must be removed from $h(t-1)$ state and thus keeping only relevant stuff. It is surrounded by a sigmoid function which helps to crush the input between 0 and 1 (Equation (3)):

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \tag{3}$$

Input Gate: In the input gate, we decide to add new stuff from the present input to our present cell state scaled by how much we wish to add them. Sigmoid layer decides which values to be updated and *tanh* layer creates a vector for new candidates to added to present cell state (Equations (4) and (5)):

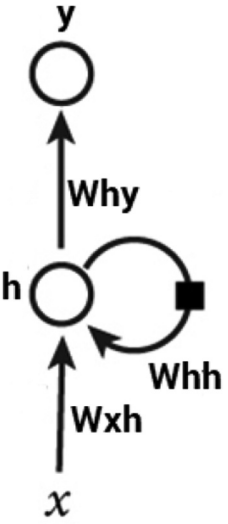

Figure 1. A recurrent neural network.

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i) \tag{4}$$

$$\hat{C}_t = tanh(W_C.[h_{t-1}, x_t] + b_C) \tag{5}$$

Then, the cell state is calculated by Equation (6):

$$C_t = f_t * C_{t-1} + i_t * \hat{C}_t \tag{6}$$

Output Gate: Finally, the sigmoid function decides what to output from the cell state as shown in Equation (7). We multiply the input with "tanh" to crush the values between (−1) and 1, and then multiply it with the output of sigmoid function so that we only output what we want to (Equations (7) and (8))

$$o_t = \sigma(W_o.[h_{t-1}, x_t] + b_o) \tag{7}$$

$$h_t = o_t * tanh(C_t) \tag{8}$$

Gated Recurrent Unit

GRU was proposed by Cho et al. in 2014 (Cho et al. 2014). It is similar to LSTM in structure but simpler to compute and implement. The difference between a GRU cell and an LSTM cell is in the gating mechanism. It combines the forget and input gates into a single update gate. It also merges the cell state and the hidden state. The function of reset gate is similar to the forget gate of LSTM. Since the structure of GRU is very similar to LSTM, we will not get into the detailed formula. The structure of a GRU cell is shown in Figure 4.

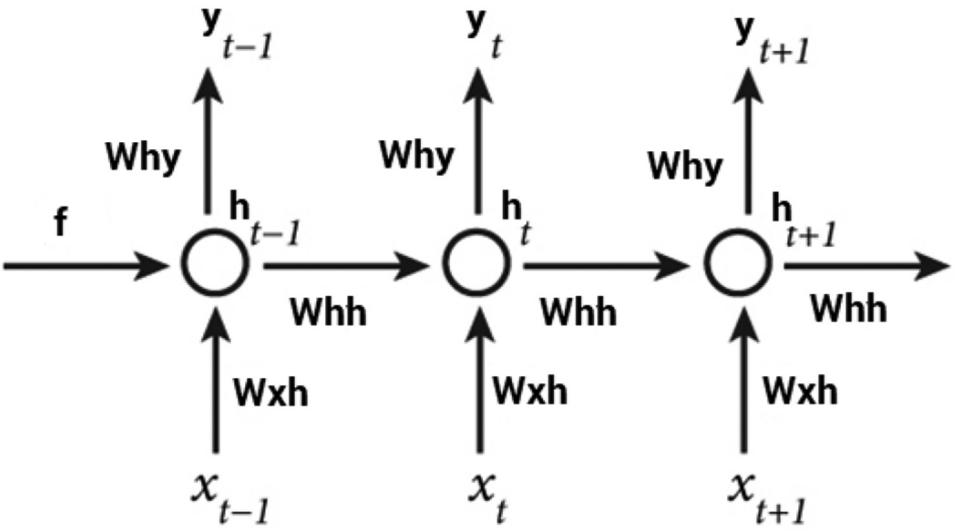

Figure 2. An unrolled recurrent neural network.

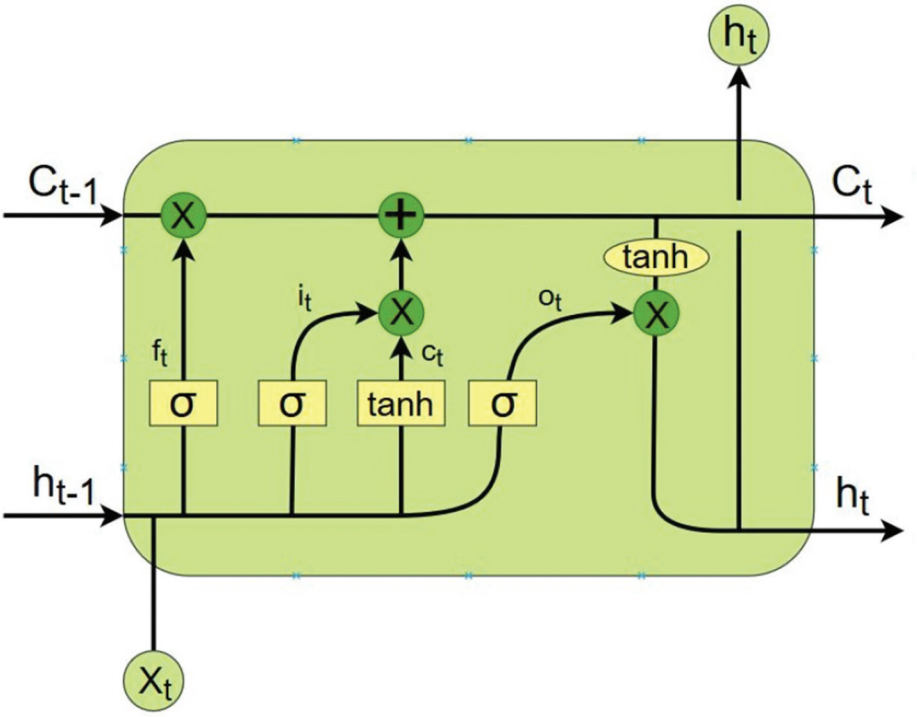

Figure 3. An LSTM cell.

Methods for Comparison

We compared the results obtained from recurrent neural networks with a tree-based regression method (XGBoost), one linear regression method (LASSO), and one moving average time-series forecasting method (DEMA). We have tuned the parameters for all these methods, then reported the results. Since these methods are not able to process sequentially formed data, demand intensity for four previous timeslots (the sequence length chosen for RNNs) were fed to them as features.

DEMA

Double exponential moving average is a well-known method for time-series forecasting problems. It attempts to remove the inherent lag associated with Moving Averages by placing more weight on recent values. The name suggests this is achieved by applying a double exponential smoothing which is not the case. The name double comes from the fact that the value of an EMA (Exponential Moving Average) is doubled. To keep it in line with the actual data and to remove the lag value "EMA of EMA" is subtracted from the previously doubled EMA.

$$DEMA = 2 \times EMA - EMA(EMA) \tag{9}$$

LASSO

Least absolute shrinkage and selection operator (LASSO) is a linear model that estimates sparse coefficients. It usually produces a better prediction result than simple linear regression. We use the LASSO implementation from the scikit-learn library. (Tibshirani 1996)

XGBoost

eXtreme Gradient Boosting (XGBoost) is a powerful ensemble boosting tree-based method and is widely used in data mining applications both for classification and regression problems. We use the XGBoost implementation from XGBoost python package. (Chen et al. 2015)

Results

In this section, we declare our RNNs' specifications and introduce the metrics that evaluations are performed based on them. Then, we evaluate different RNN models on our dataset and see how well they can predict the requests in the future. In addition, we compare our model with three other baselines and show that RNNs outperform all.

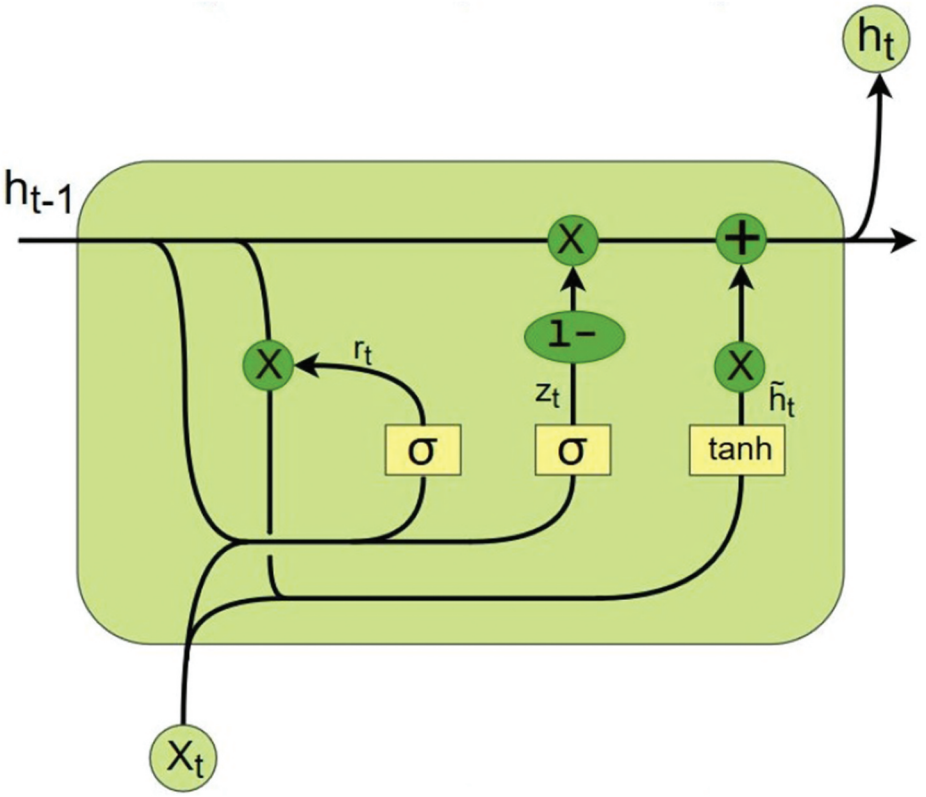

Figure 4. A GRU cell.

Experimental Setup

Our dataset is obtained from TAP30 Co. ride requests in Tehran from September 1 to December 20, 2017. We used the first prior 80 days to train the models and last 30 days for validation. All three kinds of recurrent neural networks (Simple RNN, GRU, LSTM) were implemented in Keras API built on top of Tensorflow. Although recurrent neural networks can accept sequences with any length as input, because of the nature of our problem we had to choose a constant sequence length. Due to the constrained computational power we had, we used every hour data as a sequence. Because the time interval for each data point is 15 minutes, each sequence consists of four data points. Since the data contains records for 110 days, the shape of data would be (110*24, 4, 68). Table 3 includes the list of parameters used in the experiment for all three types of RNNs.

Evaluation Metrics

We use root mean absolute error (RMSE) and mean absolute percentage error (MAPE) to evaluate the models. These metrics are defined as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_{t+1}^{i} - \hat{y}_{t+1}^{i}\right)^2} \tag{10}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{\left|y_{t+1}^{i} - \hat{y}_{t+1}^{i}\right|}{y_{t+1}^{i}} \tag{11}$$

where y_{t+1}^{i} and $\hat{y}_{t+1}^{i}$ mean the real and prediction value for demand in region i for time interval $t+1$ and n denotes total number of samples.

Experimental Results

First, we report the performance of RNNs (RMSE and MAPE) over the entire city (all selected regions) and then we report the errors on each category of regions.

Performance over the Entire City

To evaluate the prediction performance over the entire city which includes 64 regions, we compare the performance of RNNs with other methods described in 3.4 in terms of RMSE and MAPE from Equations (10) and (11). We report the RMSE and MAPE over the entire city during daily hours in Figures 5 and 6.

As it can be seen in Figures 5 and 6, all methods share common patterns through both metrics. For instance, they reach their minimum values at about 3 am and maximum values at about 7 pm. All three kinds of RNNs show better performance than the other methods, but between them, RNN and GRU have nearly the same error values during the day and are better than LSTM with a considerable difference. There is a haphazard pattern between hours 12:00 am and 6:00 am in Figure 6. According to Equation (11), MAPE is a very sensitive metric and depends on the real value's range. Since the amount of

Table 3. Experimental parameters.

Data of each sequence	1 hour data
Time-step length	15 mins
Sequence length	4
Number of regions	64
Number of features	68
Number of hidden layers	2
Number of neurons in each hidden layer	1500–2000
Activation function of hidden recurrent layers	tanh
Loss function	Mean squared error

ride requests through these hours is extremely low, this metric fails to have a specific pattern during these hours. Predicting demand intensity during rush hours (about 8 am and 5 pm) is considered more crucial than the other times. According to error rates, both RMSE and MAPE, it can be observed that RNNs demonstrate considerably better performance in comparison to the others.

Table 4 shows the detailed values of errors over the entire city for each method. Training was performed on a core-i7-7700HQ CPU with 16 GBs of RAM.

Performance over Categorized Regions

We have categorized 64 regions in Tehran into five distinct categories. The regions with average ride requests per day greater than 1600 are categorized as very crowded regions and the regions with average ride requests per day less than 400, are categorized as very uncrowded regions and the other 3 categories are placed between these 2 categories. Figures 7 and 8 illustrate the performance in terms of RMSE and MAPE, respectively, over these five categories. As we move from the very uncrowded regions to very crowded ones, since the real value of demand gets greater, the range for RMSE gets greater and the range for MAPE becomes less. But over all five categories, RNNs show a better performance. Especially simple RNN and GRU are the best models.

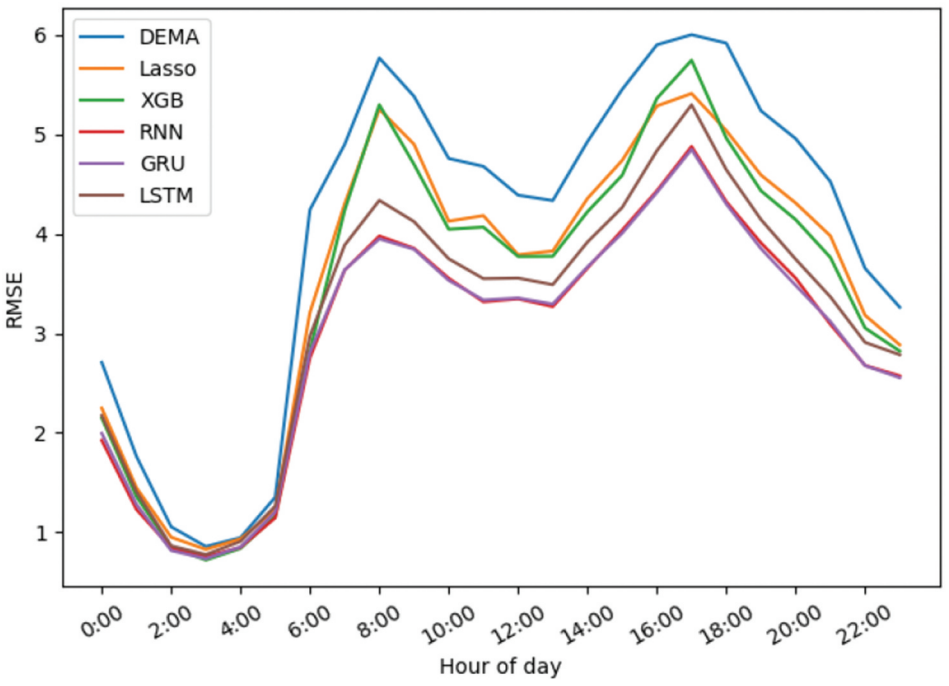

Figure 5. Prediction performance at different hours according to RMSE.

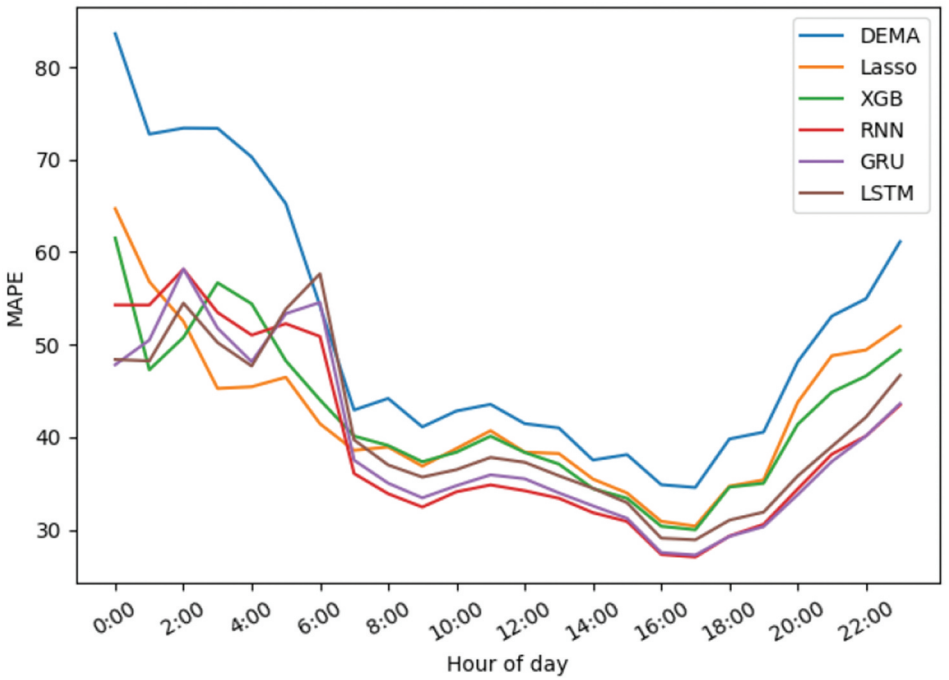

Figure 6. Prediction performance at different hours according to MAPE.

Conclusion

In this paper, different types of recurrent neural networks were implemented and used in order to forecast short-term demand in different regions on an online car-hailing company's data. We compared the performance of prediction between three types of RNNs including simple RNN, GRU, and LSTM with tree-based models (XGBoost and Random forest), a very powerful linear regression model (LASSO) and time-series forecasting models based on moving averages (SMA, DEMA). The results indicated that all three types of RNNs outperformed the other methods but the simple RNN and GRU showed the best results between RNNs. Compared to the best non-RNN method (XGBoost), GRU and Simple RNN reduced RMSE about 15% and reduced MAPE nearly 8%. Since the nature of the demand prediction problem for traffic flow is a short-term history-dependent kind, more simple types of RNN performed better than long-short term memory

Table 4. Errors over the entire city.

Method	RMSE	MAPE (%)	Training time
DEMA	4.37	48.54	-
LASSO	3.87	41.42	4 mins/37 secs
XGBoost	3.78	40.80	120 mins/53 secs
LSTM	3.46	39.04	146 mins/43 secs
Simple RNN	3.22	37.42	16 mins/40 secs
GRU	3.21	37.50	119 mins/19 secs

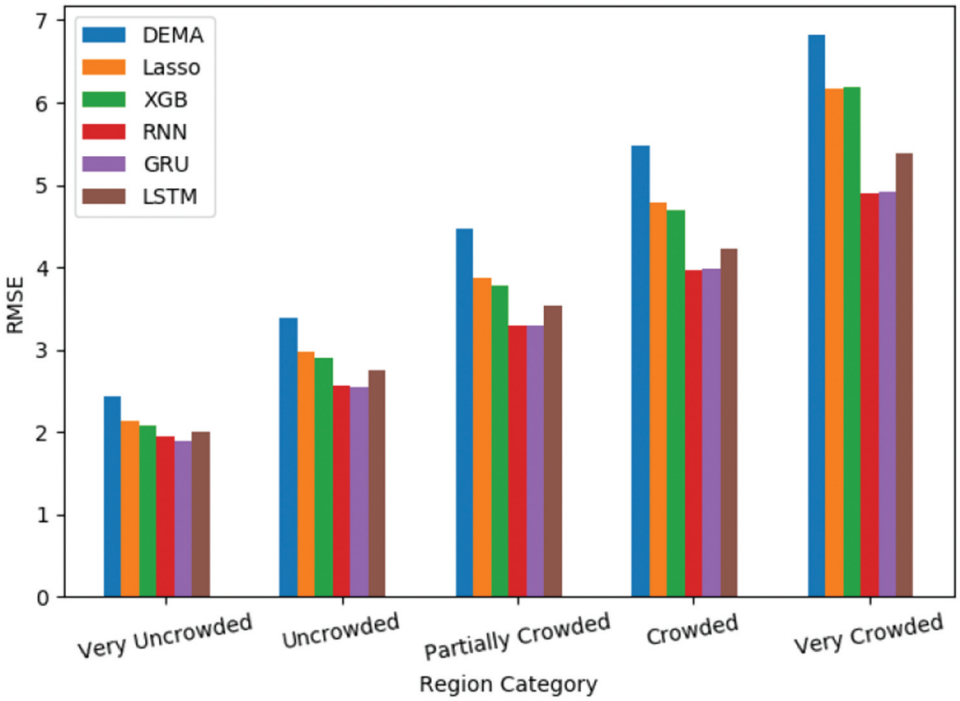

Figure 7. Prediction performance in different region categories according to RMSE.

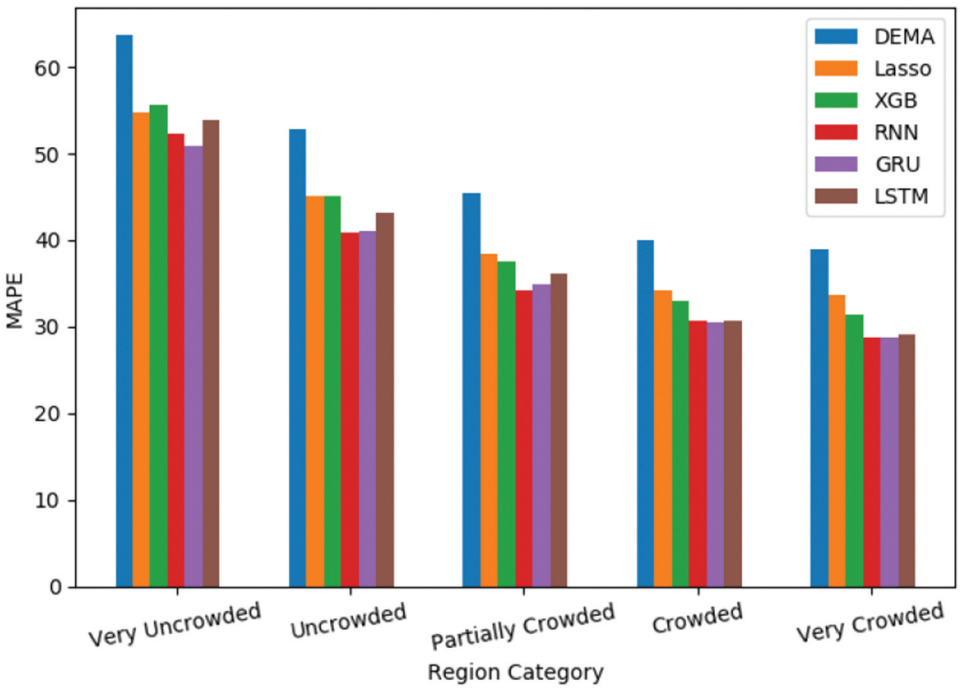

Figure 8. Prediction performance in different region categories according to MAPE.

networks (LSTM). Not only LSTM networks' performance is worse than other RNNs, but also it takes more time for training due to the complexity of these networks.

Acknowledgments

This research is financially supported by TAP30 Co. and also the authors are grateful to TAP30 Co. for providing sample data.

References

Chen, T., T. He, M. Benesty, V. Khotilovich, and Y. Tang. 2015. Xgboost: Extreme gradient boosting. *R Package Version 0.4-2* 1–4.

Cho, K., B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv Preprint arXiv:1406.1078*.

Connor, J. T., R. D. Martin, and L. E. Atlas. 1994. Recurrent neural networks and robust time series prediction. *IEEE Transactions on Neural Networks* 5 (2):240–54. doi:10.1109/72.279188.

Davis, N., G. Raina, and K. Jagannathan. 2016. A multi-level clustering approach for forecasting taxi travel demand. 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), 223–28, Rio de Janeiro, Brazil.

Graves, A., M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber. 2009. A novel connectionist system for unconstrained handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31 (5):855–68. doi:10.1109/TPAMI.2008.137.

Graves, A., A.-R. Mohamed, and G. Hinton. 2013. Speech recognition with deep recurrent neural networks. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 6645–49, Vancouver Convention & Exhibition Centre, Vancouver, Canada.

Hochreiter, S., and J. Schmidhuber. 1997. Long short-term memory. *Neural computation* 9 (8):1735–80. doi:10.1162/neco.1997.9.8.1735.

Ke, J., H. Zheng, H. Yang, and X. M. Chen. 2017. Short-term forecasting of pas- senger demand under on-demand ride services: A spatio-temporal deep learning approach. *Transportation Research Part C: Emerging Technologies* 85:591–608. doi:10.1016/j.trc.2017.10.016.

Krizhevsky, A., I. Sutskever, and G. E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, edited by P. Bartlett, 1097–105. Neural Information Processing Systems Foundation, Inc.

LeCun, Y., Y. Bengio, and G. Hinton. 2015. Deep learning. *Nature* 521 (7553):436. doi:10.1038/nature14539.

Li, X., G. Pan, G. Qi, and S. Li. 2011. Predicting urban human mobility using large-scale taxi traces. Proceedings of the First Workshop on Pervasive Urban Applications, San Francisco.

Moreira-Matias, L., J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas. 2013. Predicting taxi–passenger demand using streaming data. *IEEE Transactions on Intelligent Transportation Systems* 14 (3):1393–402. doi:10.1109/TITS.2013.2262376.

Mukai, N., and N. Yoden. 2012. Taxi demand forecasting based on taxi probe data by neural network. In *Intelligent interactive multimedia: Systems and services*, edited by T. Watanabe, J. Watada, N. Takahashi, R. J. Howlett, L. C. Jain, 589–97. Berlin, Heidelberg: Springer-Verlag Berlin Heidelberg.

Tian, Y., and L. Pan. 2015. Predicting short-term traffic flow by long short-term mem- ory recurrent neural network. 2015 IEEE International Conference on Smart City/Socialcom/ Sustaincom (Smartcity), 153–58, Chengdu, China.

Tibshirani, R. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* 58 (1):267–88. doi:10.1111/j.2517-6161.1996.tb02080.x.

Wang, D., W. Cao, J. Li, and J. Ye. 2017. DeepSD: Supply-demand prediction for online car-hailing services using deep neural networks. 2017 IEEE 33rd International Conference On Data Engineering (Icde), 243–54, San Diego, California.

Xu, J., R. Rahmatizadeh, L. Bölöni, and D. Turgut. 2018. Real-time prediction of taxi demand using recurrent neural networks. *IEEE Transactions on Intelligent Transportation Systems* 19 (8):2572–81. doi:10.1109/TITS.2017.2755684.

Yuan, N. J., Y. Zheng, L. Zhang, and X. Xie. 2013. T-finder: A recommender system for finding passengers and vacant taxis. *IEEE Transactions on Knowledge and Data Engineering* 25 (10):2390–403. doi:10.1109/TKDE.2012.153.

Zhang, X., X. Wang, W. Chen, J. Tao, W. Huang, and T. Wang. 2017. A taxi gap prediction method via double ensemble gradient boosting decision tree. Big Data Security on Cloud (bigdatasecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and 2017 IEEE 3rd International Conference on Data and Security (IDS), 255–60, Beijing, China.

Zhao, K., D. Khryashchev, J. Freire, C. Silva, and H. Vo. 2016. Predicting taxi demand at high spatial resolution: Approaching the limit of predictability. 2016 ieee international conference on Big data (big data), 833–42, Washington, DC.

Zhao, Z., W. Chen, X. Wu, P. C. Chen, and J. Liu. 2017. Lstm network: A deep learn- ing approach for short-term traffic forecast. *IET Intelligent Transport Systems* 11 (2):68–75. doi:10.1049/iet-its.2016.0208.