IET Intelligent Transport Systems

Research Article

Predicting station level demand in a bike-sharing system using recurrent neural networks

ISSN 1751-956X
Received on 1st May 2019
Revised 20th December 2019
Accepted on 18th February 2020
E-First on 27th March 2020
doi: 10.1049/iet-its.2019.0007
www.ietdl.org

Po-Chuan Chen[1], *He-Yen Hsieh*[1,2], *Kuan-Wu Su*[1], *Xanno Kharis Sigalingging*[1], *Yan-Ru Chen*[1], *Jenq-Shiou Leu*[1] ✉

[1]*Department of Electronic and Computer Engineering, National Taiwan University of Science and Technology, Taipei City 10607, Taiwan*
[2]*Institute of Information Science, Academia Sinica, 128 Nankang, Taipei, Taiwan*
✉ *E-mail: jsleu@mail.ntust.edu.tw*

Abstract: The modern multi-modal transportation system has revolutionised the landscape of public mobility in cities around the world, with bike-sharing as one of its vital components. One of the critical problems in persuading citizens to commute using the bike-sharing service is the uneven bikes distribution which leads to bike shortage in certain locations, especially during rush hours. This study offers a system, which provides predictions of both rental and return demand in real-time for each bike station by using only one model, which can be used to formulate load balancing strategies between stations. Five different architectures based on recurrent neural network are described and compared with four evaluation metrics: mean absolute percentage error, root mean squared logarithmic error, mean absolute error and root mean squared error. This system has been tested with New York Citi Bike dataset. The evaluation shows the authors' proposed methods demonstrate satisfying results at both the global and station levels.

1 Introduction

In recent years, bike-sharing has become a vital form of transportation in many cities worldwide. Research in 2013 showed more than 500 cities around the world have implemented bike-sharing services, providing convenient transportation for their citizens [1]. Midgley [2] notes that bike-sharing experienced very rapid growth compared to other forms of transportation. In most cases, a bike-sharing system consists of numerous bike stations in various locations around the city, with each station providing bikes to pick up from and return to. Typically, the cost of a ride is relatively cheaper compared to other means of transportation. This encourages users to rent bikes as an option for short-term transportation instead of using their own bikes or taking a bus. Moreover, it is more friendly to our environment. However, some measures can be performed to improve the bike-sharing system. At the global level, the system generally does not consider the local or station level fluctuation or local pattern which occurs briefly because of people travelling in different stations periodically. In light of this, we illustrate this situation with several scenarios, for example, a case where there is an unbalanced distribution of available slots between various stations caused by people travelling in clusters. This leaves several stations with few or no bikes and others with limited parking space available, as shown in Fig. 1. There are even more patterns generated in the dynamic system. For example, when people travelling in a certain path between work and home during lunch hour, they will create a loop of travelling pattern, so if we wait until the unbalanced bike distribution occurs and then use trucks to move bicycles between each station to fill out the station with no bicycle, it will generate a situation in which the filling of empty bicycle station will always trailing behind the user travelling pattern. This situation causes the inefficiency of the truck dispatching. On the other hand, we will have a systematic problem generated by some stations without bicycles. When the users are parking the bicycle at stations with buffered or reserved bicycles, there are constant unpacking and packing bicycles happening at those stations. It will overfill the station when the amount of traffic flow through the system is too high. Hence we need to find a solution or prediction system that not only takes into account the global traffic condition or global pattern in the past, but also the local pattern over time, constantly involving and adapting to the habit of the users. Recent research studies about the prediction of bike distributions have produced good results, but most of them only deal with the global or cluster level distribution, not on individual station level which can contribute more in improving bike rebalancing strategies. To address this issue, we extend our previous work [3] and present five recurrent neural network (RNN) based architectures which excel at finding high-dimensional time-series patterns, to calculate a prediction of station-level bike distribution. These schemes employ only one model for both bike pick-up and return demands. The proposed methods take not only the accuracy of station level but also the accuracy of the global level into consideration. Evaluation of our models using the dataset provided by the New York Citi Bike [4]. The results show that both at station level and global level, these models outperform the baselines models, especially at the global level. Moreover, our proposed architectures process the predictions for all stations simultaneously, showing very efficient data processing.

2 Related works

Recently, bike-sharing system has drawn great attention and the research about it has been widely studied. The survey and analysis of bike-sharing systems in Paris [5], New York [6], Washington D.C. [7] and Montreal [8] have been organised. In this section, we review relevant recent work in different aspects such as system design, system pattern, system prediction, and system operation.

2.1 System design

With better design, the bike-sharing system can not only meet the expectation from citizens, but also reduce the operating costs. DellOlio *et al.* [9] estimate the potential demand for the bike, the price that users will accept, and optimal location distribution. Prem Kumar and Bierlaire [10] and Martinez *et al.* [11] propose new models in an attempt to resolve the optimal location issue. Lin and Yang [12] propose a model that takes the interests of both users and investors into account. Chen *et al.* [13] formulate the location problem of bike stations as a bike demand prediction issue and use semi-supervised learning algorithm for the prediction. Garcia-

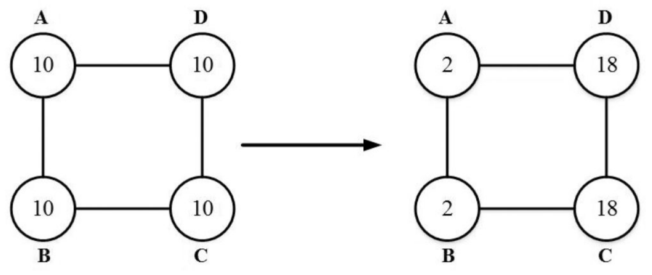

Fig. 1 *Unbalanced bike distribution*

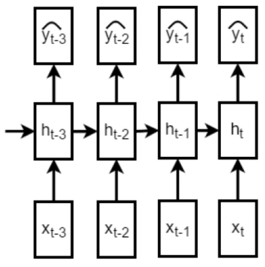

Fig. 2 *Expand serial data into parallel input to FNN*

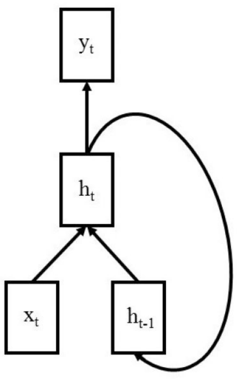

Fig. 3 *Recurrent neural network*

Palomares *et al.* [14] propose a GIS-based method to estimate the demand and solve the station location problem.

2.2 System pattern

Analysing a bike-sharing system can help us to find out the mobility pattern in a city. Bargar *et al.* [15] explore using user demographics and trip characteristics between different bike-sharing programs to reveal the pattern differences among cities. Some research studies focus on how to partition bike stations into groups rested on their usage locations or patterns. Vogel *et al.* [16] compare different clustering algorithms based on trip volumes to analyse the bike usage pattern in Vienna. Borgnat *et al.* [17] view a bike-sharing system as a dynamical network and apply community detection algorithm for clustering to obtain the distribution pattern.

2.3 System prediction

Froehlich *et al.* [18] use a Bayesian network to forecast the number of available docks and bikes for each station and compare different classifiers to predict the status of each station. Kaltenbrunner *et al.* [19] use ARIMA method to forecast the number of available docks and bikes for each station. Yoon *et al.* [20] proposed an spatio–temporal ARIMA method to forecast the available docks and bikes at each station. Other research studies focus on forecasting bike rental demand. Vogel *et al.* [16, 21] adopt time-series analysis to forecast hourly global bike demand in Vienna. Borgnat *et al.* [22] consider both non-stationary trends with hourly fluctuation to predict next hour global rental demand. Li *et al.* [23] use two models to predict the next hour bike rental and return demand, respectively, at the cluster level. Yang *et al.* [24] also use two models to meet station level prediction. The check-in model considers spatio–temporal information to predict bike return demand, and the check-out model uses random forest (RF) to predict bike rental demand. Unlike the method proposed by Yang *et al.*, our proposal only needs one model for both rental and return demand prediction, and can predict for all the stations simultaneously. Besides bike demand prediction, Zhang *et al.* [25] use two models to predict each trip. One of the models predicts the trip duration and the other predicts its destination.

2.4 System operation

Due to the unbalanced bike distribution in a bike-sharing system, many research studies focus on solving bike rebalancing problem which is the reallocation of bikes efficiently to satisfy the demand from the user. To solve this problem, Tal *et al.* [26] aim to consider all rebalancing aspects, such as the capacity of a truck, the time required to load a bike onto a truck, travel time and cost, and model by minimising the expected cost of events to determine the truck routes. Contardo *et al.* [27], Benchimol *et al.* [28] and Chemla *et al.* [29] consider the external features for determining truck routes. Schuijbroek *et al.* [30] propose a cluster-first route-second heuristic to deal with this problem.

3 Recurrent neural networks

One traditional way to learn time-dependent data using neural network is by transforming them into parallel order-independent data and feeding them into feedforward neural network (FNN), as shown in Fig. 2. However, this framework may not fit for every case. The pattern may vary in length, or the repeated pattern may lie very deep and scattered far between. On the other hand, RNNs suit time-series signals very well. Inputs, like sound waves, seismic activities, or stock market trends, can be fed sequentially into RNN, as shown in Fig. 3 and subsequently (1a) and (1b). Equation (1a) represents the hidden layer at time t receives data both from hidden layer output h_{t-1} at time $t-1$, and input x_t at current time t. The hidden layer recurrently receives input of itself from the previous time step. And in (1b), the output layer y_t receives input from hidden layer output h_t at time t

$$h_t = f_{\text{h}}(W_{\text{h}}h_{t-1} + W_{\text{i}}x_t + b_n) \tag{1a}$$

$$\hat{y}_t = f_{\text{o}}(W_{\text{o}}h_t + b_{\text{o}}) \tag{1b}$$

where W_{h} is the weight of hidden unit, h_{t-1} is the hidden unit at time $t-1$, W_{i} is the weight of input unit, x_t is the input unit at time t, b_{h} is the bias of hidden unit, f_{h} is the activation function of the hidden unit, h_t is the hidden unit at time t, W_{o} is the weight of output unit, b_{o} is the bias of output unit and $\hat{y}_t$ is the prediction at time t.

3.1 Vanishing gradient problem

One of the major problems of the RNNs and FNNs is the vanishing gradient problem [31], in which the magnitude of the gradients goes exponentially smaller in the lower layers when the networks become deeper. The vanishing gradient problem can be solved in the FNNs by using particular initialisation methods [32]. In addition, the variants of the RNNs such as long short-term memory network (LSTM) and gated recurrent unit (GRU) are designed to deal with the vanishing gradient problem.

3.2 Long short-term memory

To solve the vanishing gradient problem and the inability of RNN to learn longer time-series, the LSTM [33] was proposed. LSTM can store information within its memory cells and control information flows via the gate mechanism. The vanishing gradient can be avoided by using the flow of the gradient through units controlled by the gates. LSTMs are composed of multiple LSTM units shown in Fig. 4 and the corresponding equations are shown in (2a)–(2e)

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \tag{2a}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \tag{2b}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{2c}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \tag{2d}$$

$$h_t = o_t \odot \tanh(c_t) \tag{2e}$$

where σ is the sigmoid function, $\odot$ is the element-wise multiplication, $\oplus$ is the element-wise addition.

The cell state c_t is controlled by the input gate i_t and the forget gate f_t. Both of the output values of the input gate i_t and the forget gate f_t are mapped into the range between 0 and 1 with the sigmoid function σ. The input gate i_t determines the ratio of input information to be passed in, and the forget gate f_t decides whether the previous cell state c_{t-1} can be passed through partially or fully. Furthermore, the output gate o_t behaves similarly while controlling the output of the hidden state h_t. In the backpropagation stage, the gradient flow of c_{t-1} and h_{t-1} are controlled by the forget gate f_t. Therefore, the vanishing gradient problem is efficiently solved by introducing the gate mechanism in the LSTMs.

3.3 Gated recurrent units

GRUs [34] shown in Fig. 5 and (3a)–(3d) are simplified version of the LSTM unit which shares many identical properties as LSTM albeit with less parameters

$$z_t = \sigma(W_{xz}x_t + W_{hz}h_{t-1}) \tag{3a}$$

$$r_t = \sigma(W_{xr}x_t + W_{hr}h_{t-1}) \tag{3b}$$

$$\hat{h}_t = \tanh(W_{xh}x_t + W_{rh}(r_t \odot h_{t-1})) \tag{3c}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t \tag{3d}$$

This design is based on empirical evaluation. The update gate z_t decides the linear interpolation between the previous hidden state h_{t-1} and the candidate hidden state $\hat{h}_t$. Furthermore, the previous hidden state h_{t-1} can also be mixed into the candidate hidden state $\hat{h}_t$ or be removed by the rest gate r_t. During backpropagation, the update gate z_t works similarly to the forget gate f_t in LSTM.

4 Data analysis

Bike-sharing has become an increasingly important form of transportation among the cities in the world recently. The dataset such as Taipei YouBike [35] and New York Citi Bike [4] in the bike-sharing field have been released for research purpose. Taipei YouBike [35] dataset only contains the current status of bike stations without providing historical data. On the other hand, the historical data of New York Citi Bike [4] dataset have been released since its first launch in 2013. The dataset contains rich trip level information which can be used as bike usage patterns.

We use the New York Citi Bike [4] dataset of the year 2014, which contains 8,081,216 individual trips. In the following subsections, we discuss the data attributes.

4.1 Data attributes and basic analysis

The data attributes of Citi Bike dataset shown in Fig. 6 provide enough information implicitly revealing the trip patterns [36]. We first analyse the monthly available bike stations shown in Fig. 7 which provide information about the expanding speed of the bike-sharing system. The number of individual stations is 332, and the number of available stations stays around 330 but it goes up and down, which means some stations are shut down during the year.

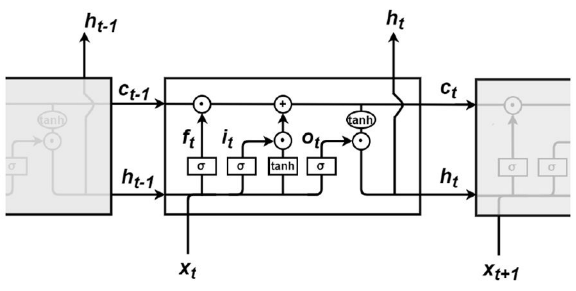

Fig. 4 *Long short-term memory*

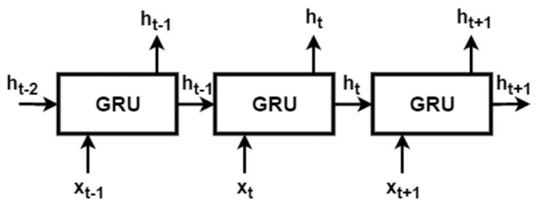

Fig. 5 *Gated recurrent units*

Trip Duration (seconds)
Start Time and Date
Stop Time and Date
Start Station Name
End Station Name
Station ID
Station Lat/Long
Bike ID
User Type (Customer = 24-hour pass or 7-day pass user; Subscriber = Annual Member)
Gender (Zero=unknown; 1=male; 2=female)
Year of Birth

Fig. 6 *Attributes in NY Citi Bike dataset*

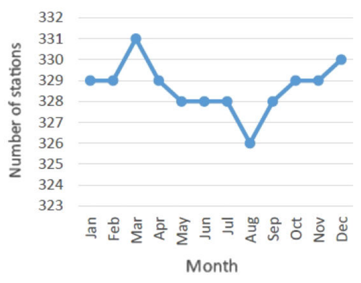

Fig. 7 *Number of available bike stations*

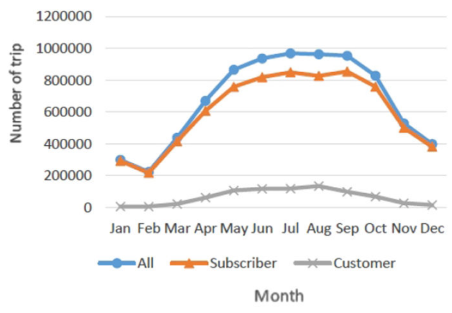

Fig. 8 *Number of trips (monthly)*

The bike rental demand is shown in Fig. 8 grow after February, reach the peak during summer, and then decline. Fig. 9 shows subscribers contribute 90% of trips, while customers contribute the other 10%. Interestingly, the percentage of trips by customers follows the same pattern as rental demand. We observe either the total number or the percentage of trips contributed by customers reveals a highly positive relationship with temperatures shown in Fig. 10. The percentage of males and females in subscribers are 77.35 and 22.65%, respectively, which also follow the same pattern as rental demand with a small standard deviation of 1.84%.

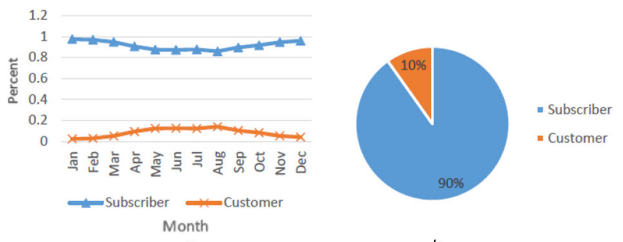

Fig. 9 *Percentage of trips by users*
(a) Monthly, ***(b)*** Monthly average

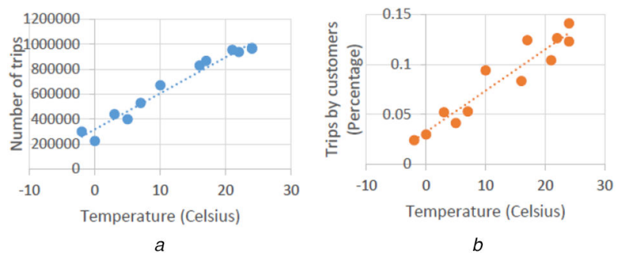

Fig. 10 *Number/percentage of trips versus temperature*
(a) Monthly, ***(b)*** By customers (monthly)

From the dataset there are several important information that affect our approach in shaping the model:

- The distribution of trip duration shows that 95% of trips take <50 min and 90% take <30 min.
- The rental demand also shows noticeable periodic pattern during the day.
- The pattern on workdays is very similar to each other except that the demand is decreasing sharply after 6:00 p.m. on Friday. During weekdays, there are two peaks of bike rental demand. One is around 8 a.m. and the other is from 5 p.m. to 7. p.m.
- During the day, the weather greatly affects the rental demand, but not so during the night.
- Wind speed and humidity can also affect the rental demand. Compared to the effect of temperature, the rental demand has a negative correlation with wind speed and humidity.
- According to the hourly weather condition, the weather data set can be divided into seven categories: Clear, Cloudy, Fog, Partly-cloudy, Rain, Snow, and Wind. The rental demands in the cloudy hour are about half in clear hour.
- Interestingly, some rental demands in raining hour are quite high, although the correlation between rental demands and precipitation is negative. The large rental demands during raining hour can be denoted as outliers. Not surprisingly, the results match our daily experiences.

5 Proposed methods

In this paper, we propose five architectures which are based on RNNs to forecast the bike rental demands as well as the check-in and check-out for each station within a certain time period. The bike rental demands are varied through time and highly influenced by weathers. In addition, there are hundreds of stations and each station has at least two variables – check-in and check-out bikes. We use RNNs due to two reasons. First, RNNs have a convincing performance on dealing with high-dimensional inputs such as time-series data. Secondly, RNNs work well on finding out the hidden temporal patterns such as the inter-correlation among stations.

5.1 Data preprocessing

The input data attributes can be split into three categories: Time, Weather and Station, which are shown in Table 1. The predicted total of check-in and check-out bikes at each station are shown in Table 2. In the data analysis section, the patterns are almost similar during weekdays and changed into other patterns on weekends. Therefore, we input the time information to help models learn historical patterns. First, the trip data for each station is transformed into the number of check-in and check-out bikes with the time interval Δt_{in}, where Δt_{in} is set to be 5 min in our experiments. And, we predict the number of check-in and check-out bikes at the next time interval Δt_{out}, where Δt_{out} is set to be 60 min. We separate the check-in and check-out bikes from subscribers and customers so that the models have more detailed information about each station.

5.2 Loss function

We propose a loss function which considers the station level as well as global level. Our loss function considers the status of each station and overall condition of all stations and can be expressed as a multi-objective optimisation shown in (4). Therefore, the result of our models can predict the rental and returned demands both for all stations as a whole and for each station

Table 1 Input data format

Category	Attribute	Value type
time	Year	one-hot encode
	Month	one-hot encoding
	Weekday	one-hot encoding
	Hour	one-hot encoding
	quarter-hour	one-hot encoding
weather	hourly weather condition	one-hot encoding
	hourly temperature	real value
	hourly apparent temperature	real value
	hourly humidity	real value
	hourly intensity of precipitation	real value
	hourly probability of precipitation	real value
	hourly dew point	real value
	hourly wind speed	real value
	hourly visibility	real value
station *i* (*i* is from 1 to *N*)	The total of check-in bikes from subscribers during Δt_{in} at station *i*	real value
	The total of check-out bikes from subscribers during Δt_{in} at station *i*	real value
	The total of check-in bikes from customers during Δt_{in} at station *i*	real value
	The total of check-out bikes from customers during Δt_{in} at station *i*	real value

Table 2 Output data format

Category	Attribute	Value type
Station *i* (*i* is from 1 to *N*)	The total of check-in bikes from subscribers during Δt_{out} at station *i*	real value
	The total of check-out bikes from subscribers during Δt_{out} at station *i*	real value
	The total of check-in bikes from customers during Δt_{out} at station *i*	real value
	The total of check-out bikes from customers during Δt_{out} at station *i*	real value

$$\text{loss} = \sum_{i=1}^{N} \left(\left|\hat{y}_{\text{in},i} - y_{\text{in},i}\right| + \left|\hat{y}_{\text{out},i} - y_{\text{out},i}\right|\right) + \alpha\left[\log\frac{\sum_{i=1}^{N}\hat{y}_{\text{in},i} + 1}{\sum_{i=1}^{N} y_{\text{in},i} + 1} + \log\frac{\sum_{i=1}^{N}\hat{y}_{\text{out},i} + 1}{\sum_{i=1}^{N} y_{\text{out},i} + 1}\right] \tag{4}$$

where α is the parameter of global level MSLE, N is the total number of all stations, $\hat{y}_{\text{in},i}$ is the predicted number of check-in bikes for station i, $\hat{y}_{\text{out},i}$ is the predicted number of check-out bikes for station i, $y_{\text{in},i}$ is the ground truth number of check-in bikes for station i and $y_{\text{out},i}$ is the ground truth number of check-out bikes for station i.

Our loss function shown in (4) is the combination of station-level loss and global-level loss which are balanced by a parameter α. The station-level loss is measured with mean absolute error (MAE), while the global-level loss is measured with MSLE. We can view this equation as a loss function based on both the station-level loss and global-level loss. Therefore, the models can learn bike demand distribution among all stations with the global-level loss which is served as a penalty. The α term is an adjustable parameter. When α becomes smaller, the influences on the loss function from the station-level part become bigger. On the other hand, when α becomes bigger, the influences become smaller. Due to the instability to outliers with MSE measurement, we use MAE instead. The case between MSLE and mean absolute percentage error (MAPE) measurement behaves similarly.

5.3 Proposed architectures

In our previous work, we presented an architecture of RNN to predict the demand for bike-sharing system. For this work, we add four more architectures to obtain a better understanding of how the improvements are made by using those architectures, and what improvements can be achieved by using different architectures. Each addition of these methods to our architecture adds a different point of view to the system, which can reduce the error of the system by enhancing our prediction in a specific approach. Our presentation of the error from each architecture in the Section 7 helps to explain the improvement of each additional mechanism.

5.3.1 Recurrent neural network: The first architecture we proposed is shown in Fig. 11. The architecture has two parts – an RNN encoder and a normal feedforward network as output. First, the RNN is utilised to encode the time-series data into a feature vector which represents the temporal patterns among the input data. We then input the encoded feature vector into a one-layer feedforward network followed by rectified linear unit (ReLU) activation function [37]. Note that we use ReLU [37] as the activation function because the predicted values need to be greater than or equal to 0. The ReLU activation function satisfies this condition.

5.3.2 Shared feedforward neural network + recurrent neural network (SFNN + RNN): In the second architecture, we use the shared-weight feedforward networks as the input layer for every timestamp, which is shown in Fig. 12. Each input network is identical and can be treated as a weak classifier capturing the same patterns among the data for every timestamp. In short, the shared-weight feedforward networks capture local patterns which are used as input to RNN. Since all input networks are identical, we only need to store one set of weights which will reduce the memory usage greatly compared to using the non-shared-weights networks.

5.3.3 Shared feedforward neural network + recurrent neural network + soft attention mechanism (SFNN + RNN + SAM): The third architecture is based on the previous architecture with an addition of SAM [38], and it is similar to a soft selection to choose important encoded features output from RNN by giving each feature a weight. The diagram of this architecture is shown in Fig. 13. Instead of integrating all input information into one fixed-length vector, the SAM allows the network to refer back to the input sequence and outputs the weighted average of all RNN units from feedforward networks. Therefore, the output relies on the weighted combination of all input states.

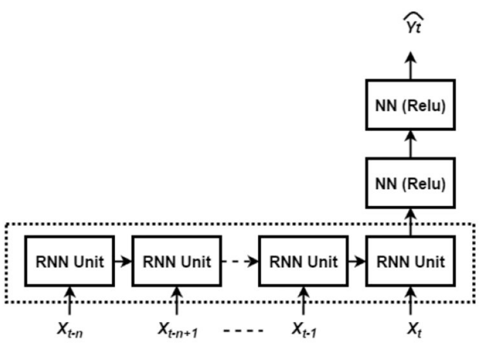

Fig. 11 *RNN*
X_t: the information collected at present; $\hat{Y}_t$: the predicted information for the next 1 h

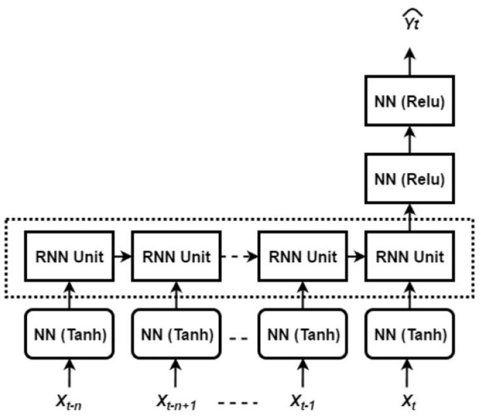

Fig. 12 *SFNN + RNN*

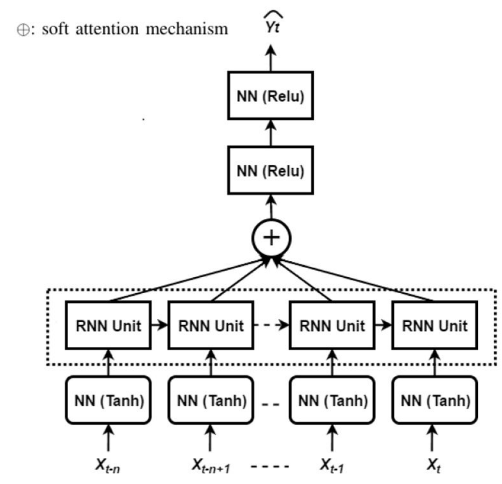

Fig. 13 *SFNN + RNN + SAM*

5.3.4 Shared feedforward neural network + bi-directional recurrent neural network + soft attention mechanism (SFNN + Bi-RNN + SAM): In the fourth architecture, we utilise a BRNN [39], which is shown in Fig. 14. The BRNN consists of forward direction and backward direction and is used to increase the amount of input information. The two hidden layers, forward hidden layer and backward hidden layer, are connected to the same output in BRNN and the output layer can get rich information from the past and future states. Therefore, the network can scan the timestamps similar to the grammatical analysis of sentences, and it helps improve the results considerably.

5.3.5 Shared feedforward neural network + bi-directional recurrent neural network (SFNN + Bi-RNN): The fifth

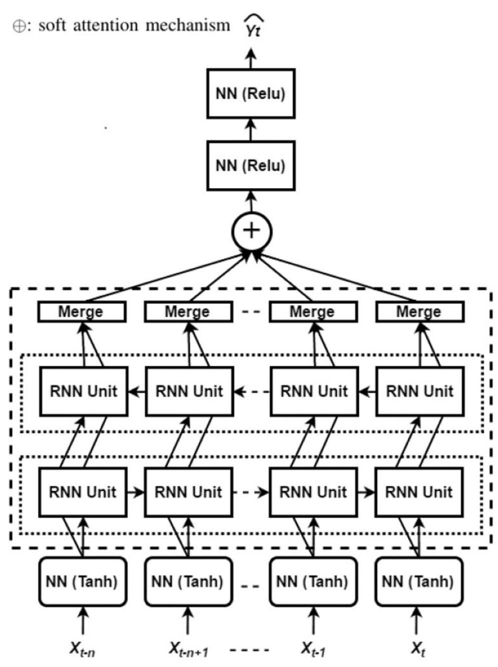

Fig. 14 *SFNN + Bi-RNN + SAM*

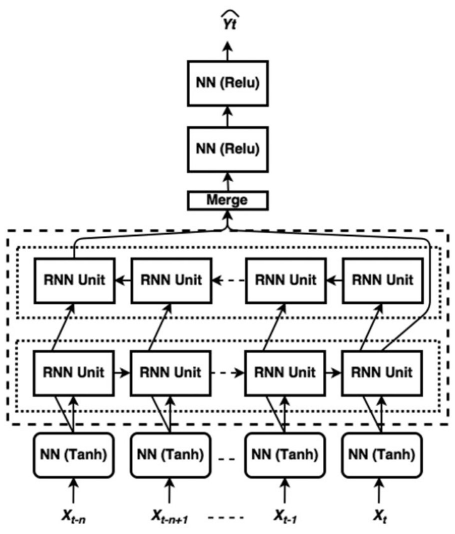

Fig. 15 *SFNN + Bi-RNN*

architecture is based on SFNN + Bi-RNN + SAM without the SAM to compare the difference between the architecture with and without SAM. The diagram is shown in Fig. 15. The architecture with soft attention can help us understand the influence on output from each RNN timestamp weight.

6 Evaluation

The data for both our training and testing set are obtained from New York Citi Bike 2014 with a total of 8,081,216 trip records. Data from 1 January to 30 September are used as the training set and data from 1 October to 31 December are used as the testing set. Before we process the data, they are arranged into input format as can be seen in Table 1 with $\Delta t_{\text{in}} = 5$ min, and for ground truth label of the prediction, we use the corresponding value of the number of check-in and check-out during the next hour of this duration. Since the length of the training set is less than one year, we remove the 'year' and 'month' attributes in input data to avoid overfitting.

6.1 Settings of proposed architectures

We observe that 95% of trips are taken within 50 min. Therefore, we use 12 RNN units for all architectures due to each RNN unit representing a time interval Δt_{in}, where the Δt_{in} is set to be 5 min. The parameter α in (4) is used to balance station-level loss and global-level loss. Due to the global-level loss is much smaller than the station-level loss, the parameter α is set to be 1.0. We use Adam [40] as our optimiser with a 128 batch size. The training epochs are set to be 50 for all architectures. Our proposed method is better compared to others because it considers the individual station level as well as the global level situation. In making use of both conditions we can see the problem from bigger picture but also at the details of the condition.

6.2 Baseline approaches

As our baselines, we pick three methods which support multiple output regression. Additionally, for comparison, we also use commonly used MAE instead of our proposed loss function. Some of the input methods which cannot accept time-series data as their input are trained using data with a given timestamp $\Delta t_{\text{in}} = 5$ min as separate inputs and then those models are trained with the same given timestamp $\Delta t_{\text{in}} = 5$ min but with summed values of the last hour inputs. Listed below are the three baseline methods:

i. Ordinary least-squares regression (OLS).
ii. Random forest [41] with 50 estimators (RF).
iii. Feedforward neural network, 4 layers with ReLu activations (FNN with ReLu).

After many experiments, we found that the value of 4 layers with ReLu activations in FNN and the value of 50 training epochs achieved a stable result within the minimal time period. We hypothesised the optimal configuration from similar works and the calculation of the method we utilised. Above configuration is the optimal one for the proposed method which we achieved after we tried to adjust the configuration in several prior experiments around the hypothesised values.

6.3 Evaluation metrics

We use four evaluation metrics shown in (5a)–(5d) for comparison. The MAE, root mean squared error (RMSE) and root mean squared logarithmic error (RMSLE) are used for station-level evaluation, while the MAPE and RMSLE are evaluated for the global level. The reason for the utilisation of these four metrics is because in our proposed method we need to properly evaluate the error from both station level and global level. On the station level, the value of MAE, RMSE, and RMSLE are to be compared to see how our model performs. On the global level, we conclude that the comparison of the error value of MAPE and RMSLE can better assess the evaluation

$$\text{MAE} = \frac{1}{M \times N} \sum_{i=1}^{N} \sum_{j=1}^{M} |\hat{y}_{ij} - y_{ij}| \tag{5a}$$

$$\text{RMSE} = \sqrt{\frac{1}{M \times N} \sum_{i=1}^{N} \sum_{j=1}^{M} (\hat{y}_{ij} - y_{ij})^2} \tag{5b}$$

$$\text{MAPE} = \frac{1}{N} \sum_{i=1}^{N} \frac{|\hat{y}_i - y_i|}{y_i} \tag{5c}$$

$$\text{RMSLE} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (\log(\hat{y}_i + 1) - \log(y_i + 1))^2} \tag{5d}$$

where M is the number of stations, N is the number of testing samples, $\hat{y}_i$ is the prediction and y_i is the ground truth.

7 Results

In this section, we present the results of our experiments. The results we obtain from several of these experiments provide underlying support for our decisions made during the construction

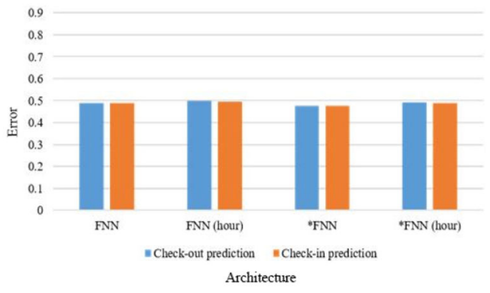

Fig. 16 *Using RMSLE to compare FNN*

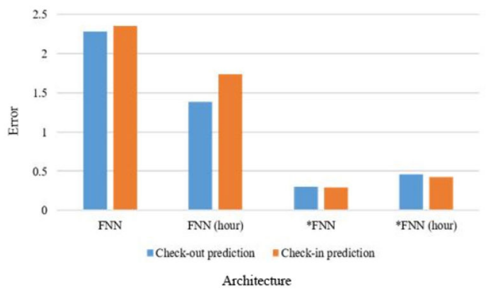

Fig. 17 *Using Global RMSLE to compare FNN*

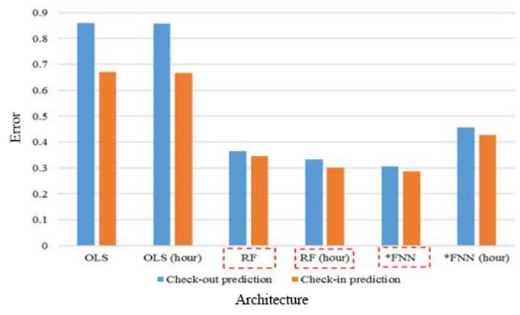

Fig. 18 *Using Global RMSLE to compare baselines*

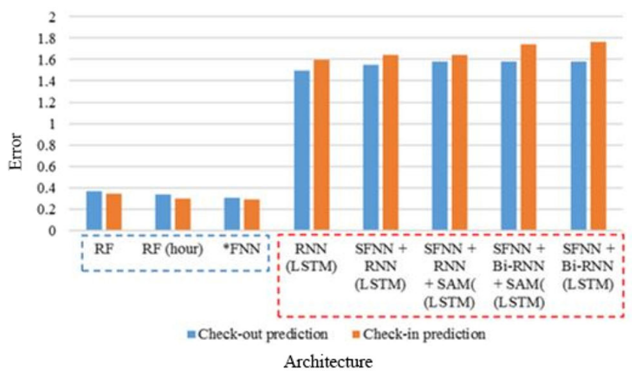

Fig. 19 *Using Global RMSLE to compare baselines and five proposed architectures based on LSTM*

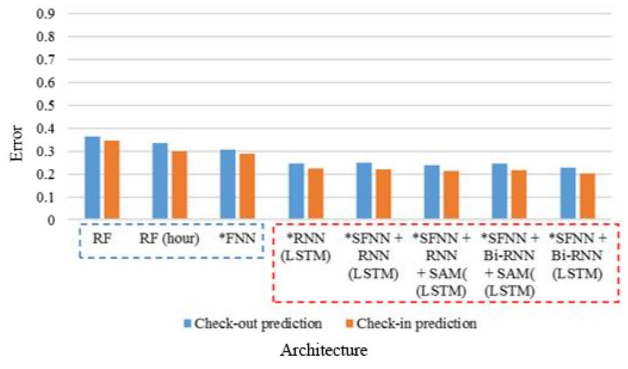

Fig. 20 *Using Global RMSLE to compare baselines and five proposed architectures based on LSTM (with loss function)*

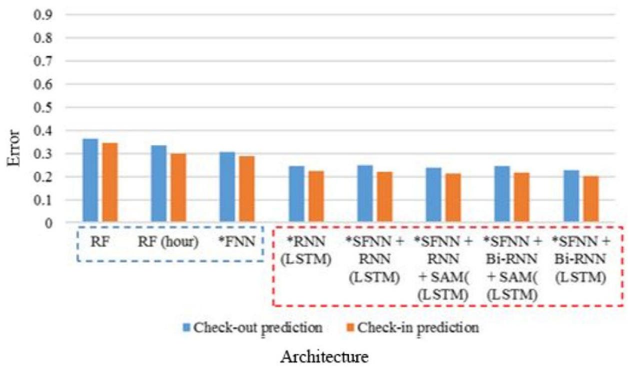

Fig. 21 *Using Global RMSLE to compare baselines and five proposed architectures based on GRU*

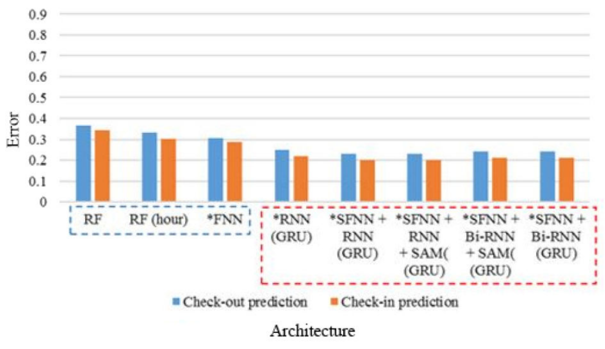

Fig. 22 *Using Global RMSLE to compare baselines and five proposed architectures based on GRU (with loss function)*

of our architectures. In each explanation of the figure, we try to describe the justification of why we pick a method and incorporate it into our system. This section also shows the final results of our architectures compared to the baselines, and how they compare to each other.

In Figs. 16 and 17, we use our loss function on FNNs, denoted as *FNNs, which generates fewer errors, comparing to the one using RMSLE and Global RMSLE. The asterisk (*) symbol in the figures indicates an architecture with our proposed loss function. From this result, it can be seen that our loss function can provide better performance than the previous version. Therefore, we only consider the FNNs with our loss function (*FNNs) to compare with other architectures. Although both RMSLE and Global RMSLE are useful to evaluate performances, we observe that Global RMSLE provides more discrimination.

In Fig. 18, RF, RF (hour), and *FNN are the top three baseline schemes from six baselines with Global RMSLE, which are highlighted in the figure. Therefore, we use RF, RF (hour), and *FNN as our comparison group of baselines.

Fig. 19 shows that using Global RMSLE on the result of LSTM without our loss function produce higher errors than that of the comparison group. On the other hand, using it on LSTM with our loss function produce fewer errors than the comparison group, which can be seen from Fig. 20. The comparison group of baselines are highlighted with the blue colour shown in the left side of Figs. 19 and 20, and the proposed architectures based on LSTM are highlighted with the red colour shown in the right side.

The proposed architectures based on GRU also have similar results. Shown in Fig. 21 are the errors of the proposed architectures based on GRU without our loss function that are higher than the errors of the comparison group when compared using Global RMSLE. Fig. 22 shows fewer errors in comparison for GRU that implements our loss function. The comparison group of baselines are highlighted with the blue colour shown in the left side of Figs. 21 and 22. And the proposed architectures based on GRU are highlighted with the red colour shown in the right side.

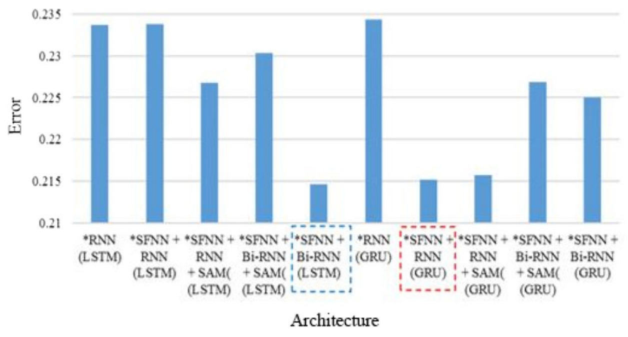

Fig. 23 *Average Global RMSLE for five proposed architectures (with loss function)*

It can be seen from Fig. 23 that the smallest errors are produced by SFNN + Bi-RNN(LSTM) and SFNN + RNN(GRU) compared to other architectures based on LSTM or GRU.

The error rate in the figure is obtained from averaging the error rate of check-in and check-out of our five proposed architectures with Global RMSLE.

8 Conclusion

This paper offers five architectures which implement RNN for predicting station-level pick-up demands in a bike-sharing network. Currently, the prediction for the entire bike demands is not accurate enough for the bike-sharing company to dispatch bikes efficiently. That is why we focus on predicting bike demands for each station and providing a better system for the department. To predict next hour pick-up and drop-off demands, our model incorporates high-dimensional time-series data obtained from each station, combined with hourly weather information simultaneously. The evaluation results show that the total demands of all stations are very high with MAPE and RMSLE evaluation metrics if the global pattern is included into the calculation, although within the MAE and RMSE evaluation metrics the results are satisfying. At station level, our proposed structures outperform the baseline methods, such as OLS (hour), RF, RF (hour), FNN and FNN (hour), and at a global level greatly outperform those methods. If both patterns are considered, the results of these methods are satisfying both globally and within the station level. Future works include combining the APP with our system to encourage subscribers to rent bikes from the station that has more bikes near the location of the subscribers or park the bikes at the station that has more empty docks although it is a little farther away from the destination point. Other possible improvement includes incorporating other trip information data such as starting station, duration, and user type to enhance current architectures.

9 References

[1] Larsen, J.: 'Bike-sharing programs hit the streets in over 500 cities worldwide'. Available at http://www.earth-policy.org/plan_b_updates/2013/update112, accessed 25 April 2013

[2] Midgley, P.: 'Bicycle-sharing schemes: enhancing sustainable mobility in urban areas', United Nations, Department of Economic and Social Affairs, 2011, pp. 1–12

[3] Chen, P.-C., Hsieh, H.-Y., Sigalingging, X.K., *et al.*: 'Prediction of station level demand in a bike sharing system using recurrent neural networks'. 2017 IEEE 85th Vehicular Technology Conf. (VTC Spring), Sydney, Australia, 2017

[4] Citi Bike. New York City Bike Sharing System Data. Available at https://www.citibikenyc.com/system-data

[5] Nair, R., Miller-Hooks, E., Hampshire, R.C., *et al.*: 'Large-scale vehicle sharing systems: analysis of Vélib'', *Int. J. Sustain. Transp.*, 2013, **7**, (1), pp. 85–106

[6] Burden, A.M., Barth, R.: 'Bike-share opportunities in New York city', Department of City Planning, New York, NY, USA, 2009

[7] LDA Consulting Washington: '2013 capital bike-share member survey report'. Technical Report 202, 2013

[8] Faghih-Imani, A., Eluru, N., El-Geneidy, A.M., *et al.*: 'How land-use and urban form impact bicycle flows: evidence from the bicycle-sharing system (BIXI) in Montreal', *J. Transp. Geogr.*, 2014, **41**, pp. 306–314

[9] DellOlio, L., Ibeas, A., Moura, J.L.: 'Implementing bike-sharing systems', *Proc. Inst. Civ. Eng. Munic. Eng.*, 2011, **164**, (2), pp. 89–101

[10] Prem Kumar, V., Bierlaire, M.: 'Optimizing locations for a vehicle sharing system'. Swiss Transport Research Conf. (Cited on pages 2, 8, and 24.), Ascona, Switzerland, 2012

[11] Martinez, L.M., Caetano, L., Eiro, T., *et al.*: 'An optimisation algorithm to establish the location of stations of a mixed fleet biking system: an application to the city of Lisbon', *Procedia-Social and Behavioral Sciences*, 2012, **54**, pp. 513–524

[12] Lin, J.-R., Yang, T.-H.: 'Strategic design of public bicycle sharing systems with service level constraints', *Transp. Res. E, Logist. Transp. Rev.*, 2011, **47**, (2), pp. 284–294

[13] Chen, L., Zhang, D., Pan, G., *et al.*: 'Bike sharing station placement leveraging heterogeneous urban open data'. Proc. of the 2015 ACM Int. Joint Conf. on Pervasive and Ubiquitous Computing, Osaka, Japan, 2015

[14] Garcia-Palomares, J.C., Gutierrez, J., Latorre, M.: 'Optimizing the location of stations in bike-sharing programs: a GIS approach', *Appl. Geogr.*, 2012, **35**, (1), pp. 235–246

[15] Bargar, A., Gupta, A., Gupta, S., *et al.*: 'Interactive visual analytics for multi-city bikeshare data analysis'. The 3rd Int. Workshop on Urban Computing (UrbComp 2014), New York, NY, USA, 2014

[16] Vogel, P., Greiser, T., Mattfeld, D.C.: 'Understanding bike-sharing systems using data mining: exploring activity patterns', *Procedia, Soc. Behav. Sci.*, 2011, **20**, pp. 514–523

[17] Borgnat, P., Robardet, C., Abry, P., *et al.*: 'A dynamical network view of Lyon's Velo'v shared bicycle system', in Mukherjee, A., Choudhury, M., Peruani, F., *et al.* (Eds.): *Dynamics on and of complex networks*, vol. **2** (Springer, New York, 2013), pp. 267–284

[18] Froehlich, J., Neumann, J., Oliver, N.: 'Sensing and predicting the pulse of the city through shared bicycling'. Proc. of the 21st Int. Joint Conf. on Artificial Intelligence, Pasadena, CA, USA, 2009, pp. 1420–1426

[19] Kaltenbrunner, A., Meza, R., Grivolla, J., *et al.*: 'Urban cycles and mobility patterns: exploring and predicting trends in a bicycle-based public transport system', *IEEE Pervasive Mob. Comput.*, 2010, **6**, pp. 455–466

[20] Yoon, J.W., Pinelli, F., Calabrese, F.: 'Cityride: a predictive bike sharing journey advisor'. 2012 IEEE 13th Int. Conf. on Mobile Data Management, Bengaluru, India, 2012

[21] Vogel, P., Mattfeld, D.C.: 'Strategic and operational planning of bike-sharing systems by data mining – a case study'. Int. Conf. on Computational Logistics, Hamburg, Germany, 2011

[22] Borgnat, P., Robardet, C., Rouquier, J.-B., *et al.*: 'Shared bicycles in a city: a signal processing and data analysis perspective', *Adv. Complex Syst.*, 2011, **14**, (3), pp. 415–438

[23] Li, Y., Zheng, Y., Zhang, H., *et al.*: 'Traffic prediction in a bike-sharing system'. Proc. of the 23rd SIGSPATIAL Int. Conf. on Advances in Geographic Information Systems, Seattle, WA, USA, 2015

[24] Yang, Z., Hu, J., Shu, Y., *et al.*: 'Mobility modeling and prediction in bike-sharing systems'. Proc. of the 14th Annual Int. Conf. on Mobile Systems, Applications, and Services, ser. MobiSys '16, Singapore, Singapore, 2016, pp. 165–178

[25] Zhang, J., Pan, X., Li, M., *et al.*: 'Bicycle-sharing system analysis and trip prediction', arXiv preprint arXiv:1604.00664, 2016

[26] Tal, R., Tzur, M., Forma, I.A.: 'Static repositioning in a bike-sharing system: models and solution approaches', *EURO J. Transp. Logist.*, 2013, **2**, (3), pp. 187–229

[27] Contardo, C., Morency, C., Rousseau, L.-M.: 'Balancing a dynamic public bike-sharing system'. Technical report, CIRRELT, September 2012

[28] Benchimol, M., Benchimol, P., Chappert, B., *et al.*: 'Balancing the stations of a self service 'bike hire' system', *RAIRO – Oper. Res.*, 2011, **45**, (1), pp. 37–61

[29] Chemla, D., Meunier, F., Wolfler-Calvo, R.: 'Balancing a bike-sharing system with multiple vehicles'. Proc. of Congress Annual de la Societe Francaise de Recherche Operationelle et d'aidea la Decision, ROADEF2011, Saint-Etienne, France, 2011

[30] Schuijbroek, J., Hampshire, R., van Hoeve, W.-J.: 'Inventory rebalancing and vehicle routing in bike sharing systems'. Technical report, Schuijbroek, February 2013

[31] Bengio, Y., Simard, P., Frasconi, P.: 'Learning long-term dependencies with gradient descent is difficult', *IEEE Trans. Neural Netw.*, 1994, **5**, (2), pp. 157–166

[32] Glorot, X., Bengio, Y.: 'Understanding the difficulty of training deep feedforward neural networks'. AISTATS, Sardinia, Italy, 2010, vol. 9

[33] Hochreiter, S., Schmidhuber, J.: 'Long short-term memory', *Neural Comput.*, 1997, **9**, (8), pp. 1735–1780

[34] Chung, J., Gulcehre, C., Cho, K., *et al.*: 'Empirical evaluation of gated recurrent neural networks on sequence modeling'. NIPS Deep Learning Workshop, Montreal, Canada, 2014

[35] YouBike: 'Taipei city bike sharing system data'. Available at http://data.taipei/opendata/datalist/datasetMeta?oid=8ef1626a-892a-4218-8344-f7ac46e1aa48, accessed 19 April 2016

[36] Motivate International, Inc.: 'Citi bike: system data'. Available at https://www.citibikenyc.com/system-data

[37] Nair, V., Hinton, G.: 'Rectified linear units improve restricted Boltzmann machines'. Proc. of the 27th Int. Conf. on Machine Learning (ICML10), Haifa, Israel, 2010

[38] Graves, A.: 'Generating sequences with recurrent neural networks', arXiv preprint arXiv:1308.0850, 2013

[39] Schuster, M., Paliwal, K.K.: 'Bidirectional recurrent neural networks', *IEEE Trans. Signal Process.*, 1997, **45**, (11), pp. 2673–2681

[40] Kingma, D., Ba, J.: 'Adam: a method for stochastic optimization', arXiv preprint arXiv:1412.6980, 2014

[41] Breiman, L.: 'Random forests', *Mach. Learn.*, 2001, **45**, (1), pp. 5–32