Transportation Research Part C 107 (2019) 287–300

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Sequence to sequence learning with attention mechanism for short-term passenger flow prediction in large-scale metro system

Siyu Hao[a,*], Der-Horng Lee[a], De Zhao[a,b]

[a] Department of Civil & Environmental Engineering, National Unversity of Singapore, Singapore 117576, Singapore
[b] Jiangsu Key Laboratory of Urban ITS, Southeast University, Si Pai Lou #2, Nanjing 210096, China

ARTICLE INFO

Keywords:
Deep learning
Sequence to sequence
Attention
metro
Passenger flow

ABSTRACT

The accurate short-term passenger flow prediction is of great significance for real-time public transit management, timely emergency response as well as systematical medium and long-term planning. In this paper, we propose an end-to-end deep learning framework that can simultaneously make multi-step predictions for all stations in a large scale metro system. A sequence to sequence model embedded with the attention mechanism forms the backbone of this framework. The sequence to sequence model consists of an encoder network and a decoder network, making it good at modeling sequential data with varying lengths and the attention mechanism further enhances its ability to capture long-range dependencies. We use the proposed framework to predict the number of passengers alighting at each station in the near future, given the number of passengers boarding at each station in the last few short-term periods. The large quantities of real-world data collected from Singapore's metro system are used to validate the proposed model. In addition, a set of comparisons made among our model and other classical approaches evidently indicates that the proposed model is more scalable and robust than other baselines in making multi-step and network-wide predictions for short-term passenger flow.

1. Introduction

Predicting short-term passenger flow in a metro system is a crucial part of public transport management, which can help to better understand travel patterns, efficiently monitor and evaluate the system status, timely implement responsive measures in case of emergency or special incidents and also enhance the service quality of public transport systems (Wei and Chen, 2012).

Short-term traffic/passenger flow prediction tasks have been widely studied in history. In the earlier stage, the models for short-term prediction tasks are mainly based on various classical statistical methods, including integrated auto-regressive moving average (ARIMA) and exponential smoothing (ES) based models (Williams and Hoel, 2003; Van Der Voort et al., 1996; Tan et al., 2009; Williams et al., 1998; Smith and Demetsky, 1997). In recent years, the emerge of large amount of data generated from various sources in cities provides us a great opportunity to better understand the nature of the hidden dynamics in transport systems and has significantly transformed the way of predicting short-term traffic/passenger flow. Thanks to the massive amounts of data and the higher computational power, we have witnessed a dramatical leap from previous classical analytical models to the current rich variety of computational intelligence and data-driven techniques in many areas of transportation research (Vlahogianni et al., 2014; Cui et al., 2018).

The advanced computational intelligence approaches, especially the broader family of machine learning methods, including

[*] Corresponding author.
E-mail address: siyuhao@u.nus.edu (S. Hao).

https://doi.org/10.1016/j.trc.2019.08.005
Received 5 January 2019; Received in revised form 12 July 2019; Accepted 9 August 2019
Available online 23 August 2019
0968-090X/

support vector machine based models, tree based models, Bayesian based models as well as neural network based models, have attracted more and more attentions from researchers and gradually been studied and applied in the realm of short-term traffic/passenger forecasting. Wu et al. (2003) utilized support vector regression (SVR) to predict short-term travel time on highways based on loop detectors data. Support vector machine (SVM) based models were also demonstrated to be able to yield satisfactory results in predicting short-term traffic flow (Mingheng et al., 2013; Deshpande and Bajaj, 2016; Cong et al., 2016). Hou et al. (2015) investigated the short-term traffic flow forecasting in urban work zones using regression tree and random forest model. In addition, various types of Bayesian network based models, such as dynamic Bayesian network, linear conditional Guass Bayesian network and other adaptive Bayesian methods, have been widely adopted to predict traffic flow as well (Tebaldi and West, 1998; Zhu et al., 2016; Polson and Sokolov, 2015; Pascale and Nicoli, 2011; Sun et al., 2005).

Furthermore, utilizing artificial neural network (ANN) based models to investigate transportation domain problems have always been very active in history, from the early traditional feed-forward neural networks to today's various types of deep neural networks with much more complicated structures and better performance. Van Lint and Van Hinsbergen (2012) comprehensively reviewed neural networks and artificial intelligence (AI) applications to transportation issues. The neural networks used in short-term traffic/passenger flow forecasting had been mainly based on multi-layer perceptions and their simple variants (e.g. shallow feed-forward neural network Hua and Faghri, 1994; Smith and Demetsky, 1994; Dougherty and Cobbett, 1997), fuzzy neural network (Yin et al., 2002; Xiao et al., 2003), hybrid neural network (Zheng et al., 2006; Chen et al., 2001; Chan et al., 2012), until deep learning techniques were introduced into transportation issues in recent years.

Modern deep learning models are based on an artificial neural network with more complicated architectures, which aim at learning a high dimension complex function by a chain of non-linear transformations from input data to output labels (Polson and Sokolov, 2017). Deep learning methods have been widely applied for many tasks and shown a couple of superiorities, especially in computer vision, speech recognition and natural language precessing. In the domain of transportation, deep learning techniques have also shown superb prediction accuracy due to their capability of capturing complex non-linear relationship based on massive amounts of data (Ke et al., 2017). Among many different deep learning models, the most commonly used neural network architectures are fully connected deep neural networks (FCDNNs), convolutional neural networks (CNNs) and recurrent neural networks (RNNs). FCDNNs usually have deeper structure (more hidden layers) than traditional shallow ANN models, enabling them to detect more complex correlations. FCDNNs have been applied in a broad range of problems in transportation area. Yi et al. (2017) utilized a FCDNN model to identify traffic flow conditions based on probe vehicle data. Lv et al. (2015) demonstrated that a deep stacked autoencoder model has good performance in forecasting short-term traffic flow. Qu et al. (2019) utilized a FCDNN model to capture the relationship between the contextual factors (e.g. season, weather, time of day) and original traffic flow data in order to better forecast the daily traffic flow in Seattle. However, FCDNNs have limitations in characterizing the spatial-temporal correlations from transportation data. Fortunately, two important members of deep learning family, the convolutional neural networks (CNNs) and recurrent neural networks (RNNs), have shown their superior ability to capture the spatial-temporal structural information (LeCun et al., 1998; Zhang et al., 2017). Because of the parameter sharing mechanism and the sparsity of connections in CNN, it enables CNN models to better characterize spatial information. Ma et al. (2017) investigated the use of CNN for large scale transportation network traffic speed prediction. In his research work, the trajectory data on different road sections at different times are organized as time-space matrix and then converted to images to be fed into the CNN model and the experiment results showed that the CNN model outperforms many other machine learning methods in predicting short-term traffic speed in a large scale network. On the other hand, RNN models can use their internal state (memory) to process sequences of inputs which helps to better capture the temporal dependencies, especially LSTM (long short term memory), a more powerful version of RNN, can learn very long range connections in a sequence. Since LSTM has significant advantages in handling sequence dependencies, it has received many attentions in various short-term prediction tasks, including travel time prediction (Duan et al., 2016; Zhang et al., 2018; Hou and Edara, 2018), traffic flow and demand forecasting (Zhao et al., 2017; Jia et al., 2017; Xu et al., 2018) as well as traffic speed estimation (Ma et al., 2015; Cui et al., 2018; Wang et al., 2019). In addition, Zhang et al. (2017) proposed a deep spatio-temporal residual network to predict the flow of human mobilities in each region of city, in which the temporal closeness, period, and trend properties of crowd flow are modeled via separate deep residual networks and then fused together with external features to form an end-to-end flow forecasting framework. Zhang et al. (2019) investigated the use of a CNN-based model to predict the short-term traffic flow where the optimal input features were determined by an add-on feature selection algorithm. Yu et al. (2017) combined CNN and LSTM to predict short-term and long-term traffic in a large scale transportation network. This model consists of initial convolutional layers and upper LSTM layers. The convolutional layers at bottom can extract spatial characteristics and its output will then be fed into upper LSTM layers to learn temporal features. Ke et al. (2017) proposed a fusion conv-LSTM model to forecast short-term passenger demand under on-demand ride services. The proposed conv-LSTM model enabled convolutional operations in LSTM instead of just stacking CNN and LSTM, which could help to learn more complicated sptaio-temporal patterns.

Compared with the above-mentioned application scenarios in transportation domain, the exploration of leveraging deep learning techniques for passenger flow prediction in large-scale public transport systems started relatively late, while it is continuously gaining more and more attention in recent years. The deep learning models that have been studied for passenger flow prediction in public transport systems range from a variety of single models, including Stacked Autoencoder (Liu and Chen, 2017), Stacked Unidirectional LSTM (Toqué et al., 2016, Toqué et al., 2017, Hu et al., 2017), Graph Convolutional Network (Li et al., 2018), to hybrid models, such as combining Convolutional units with LSTM structure (Ma et al., 2018; Du et al., 2019). However, most of the proposed frameworks tend to stack naive LSTM units (many-to-many structure) for sequential modelling. Although LSTM modules are able to capture temporal dependency, but structuring the model by simply stacking multiple layers of LSTM units (many-to-many structure) has several critical limitations. For instance, under many-to-many structure, the length of target sequence can only be equal to (or less than) the length of input sequence, which seriously limiting the flexibility and generalization ability of the model, because it is very likely to have the target sequence and input sequence that are of different lengths. In addition, the plain many-to-many structure won't see all the input (whole sequence) when generating the middle outputs (output before the last step), which leads to the limitations and irrationality in many multistep-ahead prediction tasks, especially the unidirectional LSTM that is usually the default setting. Moreover, even though LSTM or GRU are the optimized variants of conventional naive RNN, their performance would inevitably decline to some extent if the input sequence becomes longer. Fortunately, the emergence of Sequence to sequence (Seq2seq)/encoder-decoder architecture and attention mechanism has exerted significant impacts on sequential modelling tasks in recent years. Since the Seq2seq/encoder-decoder architecture could provide a much more flexible and extendable framework and attention mechanism is able to resolve the bottleneck that simple Seq2seq suffers from modelling very long-range dependencies, it has not only been widely used in many traditional deep learning tasks but also gradually drawn more attention and in other domains, such as weather and pollution forecasting (Liu et al., 2018; Liang et al., 2018) as well as traffic speed prediction (Liao et al., 2018). However, passenger flow forecasting in public metro systems leveraging Seq2seq based frameworks has not attracted much attention. In order to address the above-mentioned gaps, in this paper, we propose an end-to-end deep learning framework for network-wide passenger flow prediction in a large scale metro system. The framework is built based on a Sequence to sequence (Seq2seq) learning model with attention mechanism. This model is designed to simultaneously predict the number of alighting passengers (outbound traffic) at each station in the near future according to the number of boarding passengers (inbound traffic) at each station in the last few short-term periods.

The remainder of this paper is organized as follows. In Section 2, the proposed framework and related methodologies are described in detail. In Section 3, the settings and data for conducting experiments are introduced and the results of experiments are analyzed. Finally in Section 4, the main findings and potential future work are summarized.

2. Methodology

In this section, we elaborate all the preliminary methodologies involved in the proposed model, from basic LSTM unit to seq2seq architecture with attention mechanism.

2.1. Long Short Term Memory (LSTM) networks

Recurrent Neural Networks (RNNs) can learn to use the past information to make decisions, but conventional simple RNN models have limitation in dealing with long sequences on account of suffering from vanishing gradient issues (Bengio et al., 1994). Therefore, LSTM, a more powerful version of basic RNN unit, has been developed to better capture long-range dependencies and helps a lot in fixing vanishing gradient problems (Hochreiter and Schmidhuber, 1997). As shown in Fig. 1, unlike basic RNNs, when mapping the input sequence x to the corresponding output sequence $\hat{y}$, there are several gates in LSTM units governing the information flow, which helps to capture long-range dependencies. Flowing equations illustrate how LSTM unit stores memory for a long time and optionally let information pass through.

$$u_t = \sigma(W_{ua}a_{t-1} + W_{ux}x_t + b_u) \tag{1}$$

$$f_t = \sigma(W_{fa}a_{t-1} + W_{fx}x_t + b_f) \tag{2}$$

$$o_t = \sigma(W_{oa}a_{t-1} + W_{ox}x_t + b_o) \tag{3}$$

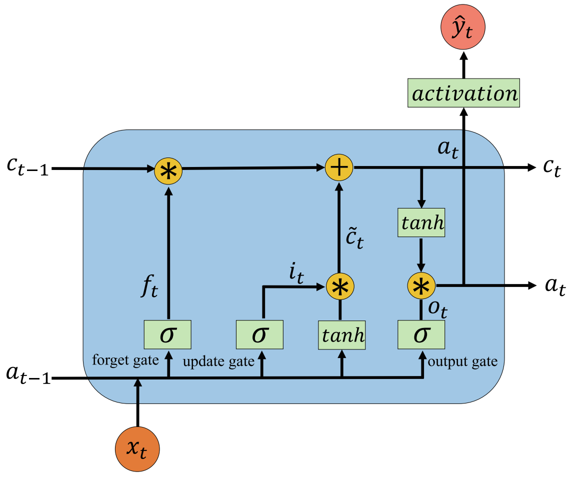


Fig. 1. Internal structure of a LSTM unit.

$$\widetilde{c}_t = tanh(W_{ca}a_{t-1} + W_{cx}x_t + b_c) \tag{4}$$

$$c_t = u_t * \widetilde{c}_t + f_t * c_{t-1} \tag{5}$$

$$a_t = o_t * tanh(c_t) \tag{6}$$

Eq. (1)–(3) refer to update gate control, forget gate control and output gate control, respectively. Eq. (4)–(6) demonstrate how the memory cell state c_t and output a_t are updated. W_{ua}, W_{ux}, W_{fa}, W_{fx}, W_{oa}, W_{ox}, W_{ca}, W_{cx}, b_u, b_f, b_o and b_c are all trainable parameters, where W_{ua}, W_{ux}, W_{fa}, W_{fx}, W_{oa}, W_{ox}, W_{ca} and W_{cx} are weighted matrices governing the connection from corresponding inputs to hidden layer while b_u, b_f, b_o and b_c are bias terms. σ refers to sigmoid function and *tanh* refers to hyperbolic tangent function and both of them are non-linear activation functions defined by following formulas:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{7}$$

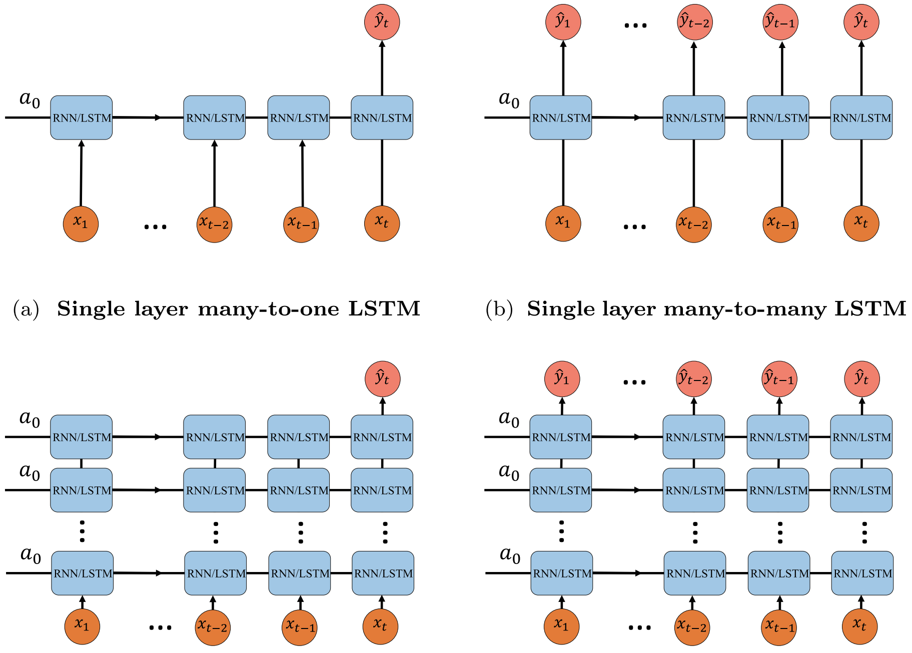
$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{8}$$

In the process of forget gate control, as shown in Eq. (2), it firstly gets the information from both the current step input x_t and the output from previous step a_{t-1}. Then, the combined information is passed to a sigmoid activation function to decide how much information will be thrown away from the cell state. The sigmoid function outputs a number between 0 and 1 for each number in the cell state c_{t-1}. Besides, the update gate aims at deciding what new information will be stored to the cell state where u_t acts as a filter and $\widetilde{c}_t$ is the candidate for replacing the memory cell c_{t-1}. The previous cell state c_{t-1} is multiplied by f_t firstly and then added to filtered candidate $\widetilde{c}_t * u_t$ to get new cell state c_t. Finally, the element-wise multiplication of filter o_t and updated cell memory c_t will generate the latest output a_t. This demonstrates how a single LSTM unit functions and why it can store long term memories.

2.2. Sequence to Sequence (seq2seq) model

For different applications, there are many different architectures of RNN models. As shown in Fig. 2, these are four most common architectures of RNN models. Both Fig. 2a and c represent many-to-one RNN models, which means that these models only have 1 output $\hat{y}$ at the last time step. Another common type is many-to-many architecture where the length of the input sequence T_x and the length of the output sequence T_y are identical, shown in Fig. 2b and d. In many-to-many architecture, the RNN/LSTM layers scan input sequence from x_1 to x_t and accordingly compute $\hat{y}_1$ so on up to $\hat{y}_t$. Moreover, Fig. 2a and b represent single layer RNN models while sometimes stacking multiple layers to build a deeper RNN model would be helpful to learn more complex functions, illustrated in Fig. 2c and d.

However, for many applications in practice, it is very possible that the length of the output sequence is greater than 1 and also



(a) **Single layer many-to-one LSTM** (b) **Single layer many-to-many LSTM**

(c) **Stacked many-to-one LSTM** (d) **stacked many-to-many LSTM**

Fig. 2. Different architectures of RNN models.

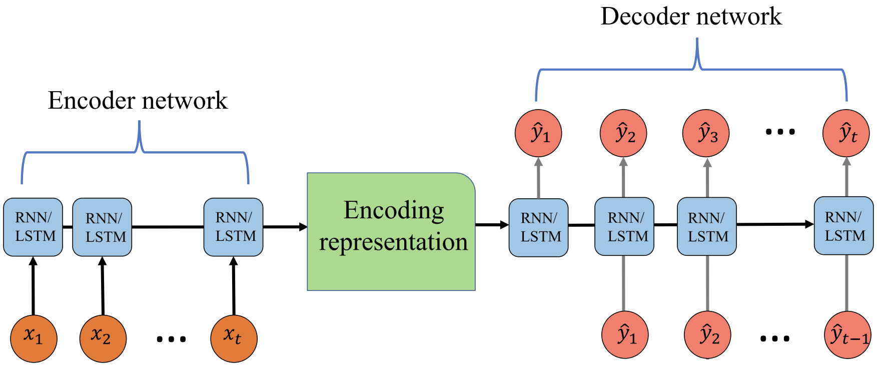


Fig. 3. Architecture of sequence to sequence model.

different from the length of the input sequence, for which the above-mentioned architectures might not be applicable. Therefore, a more flexible architecture that can handle any length of input and output sequence is needed. For the first time, Cho et al. (2014) pioneered the concept of RNN encoder-decoder network, which was also the prototype of sequence to sequence (seq2seq) model and it was further refined by Sutskever et al. (2014) who proposed a more mature seq2seq learning framework. As illustrated in Fig. 3, the seq2seq model consists of two RNN/LSTM networks, including an encoder network on the left side and a decoder network on the right side. The encoder network ingests and encodes the input sequence into a fixed-size vector representation denoted by c, which is often from the last step output of encoder network. On the other hand, the decoder network which is also built as a RNN/LSTM network will decode the encoding representation and generate the output sequence. In the process of decoding, the context vector from encoding representation c will be used as the initial hidden state of the decoder network, and the output value from the last time step $\hat{y}_{t-1}$ will also be fed into next RNN/LSTM unit as input to progressively make predictions. The seq2seq model is trained to maximize the conditional probability of a target sequence given an input sequence, which could be defined by Eq. (9):

$$p\left(y_1,\ ...,y_t \middle| x_1,\ ...,x_{t'}\right) = \prod_{t=1}^{t} p\left(y_t \middle| y_1,\ ...,y_{t-1},\ c\right) \tag{9}$$

The seq2seq model has been approved to be remarkably effective in various types of applications, from machine translation to speech recognition, especially when input sequence and output sequence are of varying lengths.

2.3. Attention mechanism

The seq2seq model has been continuously improved in past few years and the concept of attention mechanism firstly proposed by Bahdanau et al. (2014) is one of the most influential ideas in deep learning. Sometimes, when the input sequence is very long, it becomes difficult for basic seq2seq model to memorize all the input information, thus degrading the decoding and encoding performance. Fortunately, with the intervention of attention mechanism, it enables the model to learn to pay particular attention on specific parts of the input sequence when decoding and also relieve encoder network from squashing all the information into a fixed-size vector representation (Bahdanau et al., 2014). In attention architecture, the encoder network is usually built by a bi-directional LSTM network that can read input sequence in both the forward and backward directions. The activation from a bi-directional LSTM unit is denoted by $a_{t'}$, which is a concatenation or sum of the activation from forward LSTM $\overrightarrow{a_{t'}}$ and the activation from backward LSTM $\overleftarrow{a_{t'}}$. When making the prediction for output y_t, the amount of attention that y_t should pay to $a_{t'}$ from encoder network is denote by $\alpha_{tt'}$, which is defined as follows:

$$\alpha_{tt'} = \frac{exp(e_{tt'})}{\sum_{t'=1}^{T_x} exp(e_{tt'})} \tag{10}$$

In Eq. (10), T_x refers to the length of input sequence and $e_{tt'}$ represents the intermediate energy term which is computed as follows:

$$e_{tt'} = FC(s_{t-1},\ a_{t'}) \tag{11}$$

As illustrated in Eq. (11), s_{t-1} denotes the hidden state activation from previous time step in decoder LSTM and FC represents an operation of a fully connected feed-forward neural network to generate the scores of each time step state in the encoder network. Eq. (10) is a Softmax function for normalizing all corresponding scores to generate the probability distribution conditioned on target states $\alpha_{tt'}$. Then, the context vector c_t for predicting output y_t can be computed as the weighted average of $a_{t'}$, defined as below:

$$c_t = \sum_{t'=1}^{T_x} \alpha_{tt'} a_{t'} \tag{12}$$

Finally, the context vector c_t and the hidden state from last time step will be concatenated and then fed into the output layer to generate the target y_t.

Table 1
Periods categorization.

Period	Category
5 am–6 am	Pre Morning peak
6 am–9 am	Morning peak
9 am–10 am	After Morning peak
10 am–16 pm	Noon Off peak
16 pm–17 pm	Pre Evening peak
17 pm–19 pm	Evening peak
19 pm–21 pm	After Evening peak
21 pm–23 pm	Night Off peak

2.4. External features fusion

The real time passenger flow in an urban public transport system is not only affected by the previous flow but also by many other external features, such as weather, events as well as the time of day and the day of week. In order to integrate the external components into the prediction framework, we design a three-step fusion method to learn the representations of the original external data and then fuse them with the abstract representations of passenger flow sequential data. Due to the limited accessibility to various types of external data that match our historical training data, we only consider the metadata (e.g. *Time of day* and *Day of week*) in our study.

The first step of the fusion procedure is to transform the external data into needed structure. For *Time of day*h_t, we divide a single day into 8 time periods according to the normal travel pattern in Singapore, shown in Table 1. Because *Time of day*h_t and *Day of week*h_d are all categorical variables, thus we need to firstly convert them into one-hot encoding form. Subsequently, h_t and h_d will be concatenated and reshaped as tensors with the required shape($X_e \in \mathbb{R}^{B*T*O}$), where B, T, O denote batch size, length of sequence and dimension of one-hot vector, respectively. The first two dimensions (B and T) should be consistent with that of passenger flow sequential input $X_f \in \mathbb{R}^{B*T*I}$. In addition, the second step is to learn the abstract representations of external features through a small-scale Recurrent neural network as follow:

$$X_{er} = f(X_e) \tag{13}$$

$$A_{fused} = X_{er} + A_f \tag{14}$$

In Eq. (13), f refers to an operation of a Recurrent neural network and X_{er} is the abstract representations of external features. For this study, we utilize a single-layer many-to-many LSTM as f, which contributes to map the one-hot encoding external features from $X_e \in \mathbb{R}^{B*T*O}$ to $X_{er} \in \mathbb{R}^{B*T*D}$ that have the same shape with the activation of the encoder network for flow sequential input $A_f \in \mathbb{R}^{B*T*D}$ where the third dimension D equals to the number of hidden units in LSTM. The third step is to take the sum of $X_{er} \in \mathbb{R}^{B*T*D}$ and $A_f \in \mathbb{R}^{B*T*D}$ to get the fused activation $A_{fused} \in \mathbb{R}^{B*T*D}$ (Eq. (14)) and then feed it into the attention layer for subsequent decoding process.

As mentioned in previous section, the forecasting target in this paper is the demand of alighting passengers (outbound traffic) at each station in the future according to the demand of boarding passengers (inbound traffic) at each station in the last few short-term periods. The complete structure of the proposed framework is shown in Fig. 4. As illustrated, each single step of the flow sequential input is a vector representing the number of boarding passengers at each station, where $x_{t'}^i$ denotes the boarding demand at station i during time window t'. The flow sequential input will be firstly fed into a stacked bidirectional LSTM network for encoding and the external components will be fed into a small-scale RNN network for feature representation learning simultaneously, after which the external representations and activations of encoder network will be fused and transmitted to the attention extractors to generate the weighted context vectors. Finally, the decoder network will make multi-step predictions for all stations in the network according to the corresponding weighted context vector for each target time step along with the predicted output at previous time step, where $\hat{y}_t^i$ refers to the predicted alighting demand at station i during look-ahead time window t.

3. Experiments and results

3.1. Data preparation

The dataset we use covers one week's trip transactions from 19th March 2012 to 25th March 2012. In order to speed up the learning and convergence when training our model, the Min-Max normalization is performed to scale the data into the range of [0, 1] (shown in Eq. (15)) and then we use a sliding window approach to sample the scaled data and reshape them into the 3-D tensors with required form. Furthermore, the external features, such as *timeofday*, *dayofweek*, are tread as categorical variables and transformed to the one-hot encoding form.

$$z = \frac{x - min(x)}{max(x) - min(x)} \tag{15}$$

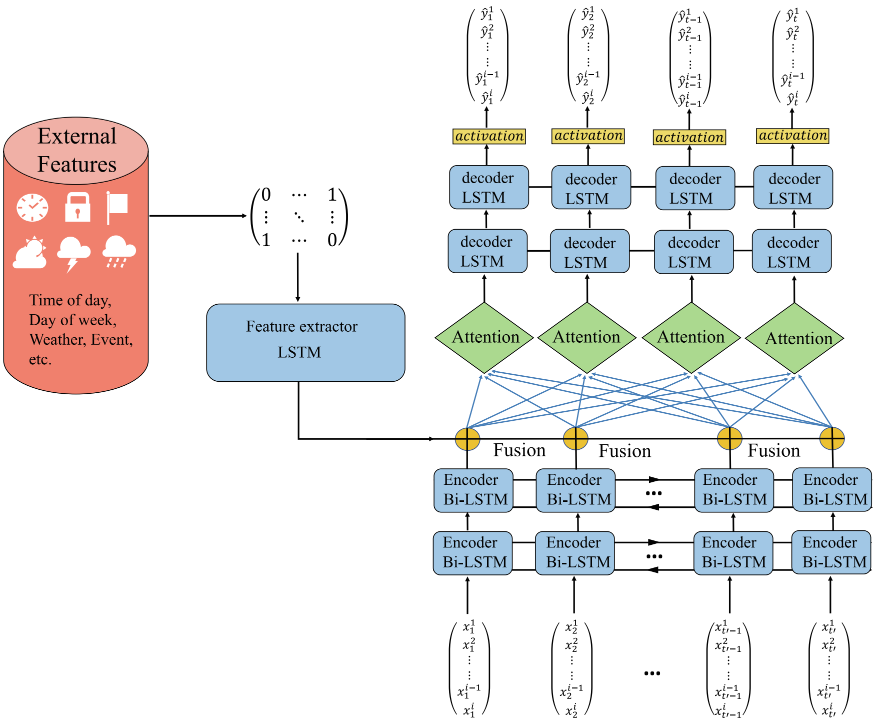


Fig. 4. Framework overview.

3.2. *Training model*

In the phase of training, we randomly select 85%of the data as training set, and the remaining 15% are used to validate the model performance. The model is trained only on the training set for 1000 epochs, while the early-stopping mechanism is introduced monitoring the loss on the validation set in order to avoid overfitting issue. Root Mean Square Error (MSE) is adopted as the evaluation metric for the proposed model, given by:

$$RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y^{(j)} - \hat{y}^{(j)})^2} \tag{16}$$

Besides, the efficient Adam optimization algorithm is used to optimize the loss function where the learning rate α is set to 0.001 and the exponential decay rates for the first and second moment estimates $\beta1$ and $\beta2$ are set to 0.9 and 0.999, respectively. All the experiments are conducted on an Ubuntu virtual machine from google cloud platform with 8 vCPU and 32 GB RAM.

3.3. *Results analysis*

3.3.1. *General comparison*

After the process of training our model and fine-tuning the hyperparameters, we compare the performance of the proposed framework (Attention_Seq2seq) with a set of classical approaches that have been widely used in traffic forecasting problems, such as auto-regressive moving average (ARIMA), support vector regression (SVR) and Bayesian ridge regression (BRR). In addition, several neural network models in the broader family of deep learning are also used as baseline models for performance comparisons, including fully connected neural networks (FCNNs) with different depths, naive LSTM and bidirectional LSTM (Bi_LSTM) with different stacking depths, Conv_LSTM, a set of basic sequence-to-sequence (S2S) models without attention mechanism as well as a set of attention sequence-to-sequence (AS2S) models with and with out external features. All the deep learning baseline models are implemented in Pytorch. Table 2 presents a detailed comparison of the prediction performance of each model.

As shown in Table 2, the best results (lowest RMSE) in each time window are highlighted in bold. For classical methods, such as ARIMA, SVR and BRR, they indeed show their effectiveness in predicting short-term passenger flow, especially BRR, which has good rankings in several time windows. However, these methods are less flexible due to their limitations in handling data with more than 2 dimensions, which means that they can not provide a network-wide end-to-end solution that are able to simultaneously make multi-step predictions for all stations in the network. Moreover, for FCNNs, models with different depths are tested under the same experiment conditions. Not surprisingly, deeper FCNN do have better prediction performance since more complex non-linear relationship can be captured (results listed in Table 2 is generated from FCNN with 5 hidden layers), but they still have a big gap with RNN models in time-series prediction tasks. In terms of LSTM models, Stacked Bidirectional LSTM (Bi_LSTM) performs better than

Table 2
Comparisons of the prediction performance with baseline models.

Model	0–15 mins (*RMSE*)	15–30 mins (*RMSE*)	30–45 mins (*RMSE*)	45–60 mins (*RMSE*)
ARIMA	25.33	25.79	26.16	27.03
SVR	42.49	42.44	43.13	44.67
BRR	21.88	22.11	22.95	23.75
FCNN	27.54	26.95	27.37	27.69
LSTM	24.82	24.16	24.09	24.50
Bi_LSTM	23.54	23.04	23.05	23.62
Conv_LSTM	27.34	25.31	25.65	25.17
S2S_1E1D	25.80	22.71	22.32	22.58
S2S_1E2D	25.62	22.44	22.24	22.51
S2S_1E3D	26.20	23.08	22.90	23.12
S2S_2E1D	27.11	23.86	23.61	23.96
S2S_2E2D	26.31	23.34	23.19	23.41
S2S_2E3D	27.81	24.11	23.89	24.17
S2S_3E1D	28.86	24.99	24.41	24.89
S2S_3E2D	27.82	24.34	24.20	24.53
S2S_3E3D	33.19	28.32	28.07	28.82
AS2S_1E1D (without external features)	24.39	23.53	23.43	24.06
AS2S_1E2D (without extenal features)	22.68	21.89	21.61	21.94
AS2S_1E3D (without external features)	22.73	21.76	21.49	21.74
AS2S_2E1D (without external features)	23.45	22.58	22.43	22.92
AS2S_2E2D (without external features)	22.56	21.82	21.61	21.89
AS2S_2E3D (without external features)	22.79	21.98	21.69	21.88
AS2S_3E1D (without external features)	23.43	22.63	22.39	23.14
AS2S_3E2D (without external features)	22.91	22.26	21.99	22.33
AS2S_3E3D (without external features)	23.29	22.45	22.31	22.45
AS2S_1E1D (with external features)	22.13	21.34	21.12	21.48
AS2S_1E2D (with external features)	20.58	19.74	19.41	19.61
AS2S_1E3D (with external features)	**20.37**	**19.37**	**19.15**	**19.26**
AS2S_2E1D (with external features)	21.79	21.19	21.02	21.24
AS2S_2E2D (with external features)	20.61	19.57	19.41	19.56
AS2S_2E3D (with external features)	21.09	19.95	19.72	19.83
AS2S_3E1D (with external features)	21.99	21.35	21.08	21.47
AS2S_3E2D (with external features)	21.91	21.12	20.87	21.06
AS2S_3E3D (with external features)	21.91	20.67	20.50	20.64

naive unidirectional LSTM in this multistep-ahead prediction, as Bi_LSTM would always take the complete input sequence into consideration when predicting the output of each step. However, Conv_LSTM doesn't perform very well for this task, even lag behind naive LSTM. In addition, we test different structures for both S2S and AS2S models, specifically the different combinations of depth of encoder and decoder. For example, S2S_1E2D refers to the basic Seq2seq model with 1 layer of encoder and 2 layers of decoder, in a similar way, AS2S_3E3D refers to the attention Seq2seq model with 3 layers of encoder and 3 layers of decoder. Overall, the Seq2seq models have much better prediction results, especially the proposed framework, Seq2seq with attention mechanism, which outperforms all other baselines in multiple look-ahead time windows prediction. The best result is yielded from AS2S_1E3D with external features fused in, which is 19.5% to 28.7% better than ARIMA, 52.1% to 56.9% better than SVR, 6.9% to 18.9% better than BRR, 26.0% to 30.4% better than FCNNs, 17.9% to 21.4% better than LSTM, 13.5% to 18.5% better than Bi_LSTM, 23.5% to 25.5% better than Conv_LSTM and 13.7% to 20.5% better than the best member from basic Seq2seq models. In addition, as illustrated in Fig. 5 the proposed model shows obvious advantages in forecasting the passenger flow in longer periods, especially in the next 30–60 min. The performance of the proposed model is very robust over the look-ahead time windows while most of the other baseline models are suffering from the decline of the accuracy as the prediction period becomes longer, which further proves that the Seq2seq structure and the attention mechanism indeed exerts a remarkable effect on capturing long-range dependencies.

3.3.2. Impact of the depth of encoder and decoder

The typical seq2seq model consists of an encoder network and a decoder network. In our study, different combinations of the number of layers of encoder and decoder have been tested to analyze the impact of network depth. We totally have 9 different combinations for both S2S models and AS2S models where the depth of encoder ranges from 1 layer to 3 layers and the depth of decoder also ranges from 1 layer to 3 layers. As illustrated in Fig. 6, we find that the depth of encoder and the depth of decoder have different effects on the prediction results for both S2S model and AS2S model. For most of the cases in our experiments, the prediction accuracy is more sensitive to the change of decoder network, which indicates that unilaterally stacking decoder layers tends to yield better prediction results than unilaterally stacking encoder layers. The prediction result normally becomes worse especially when

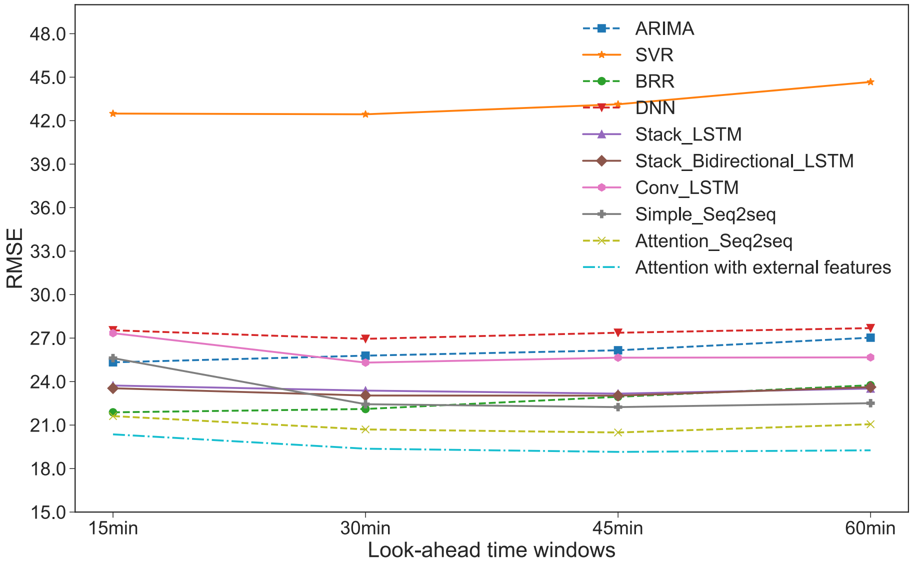
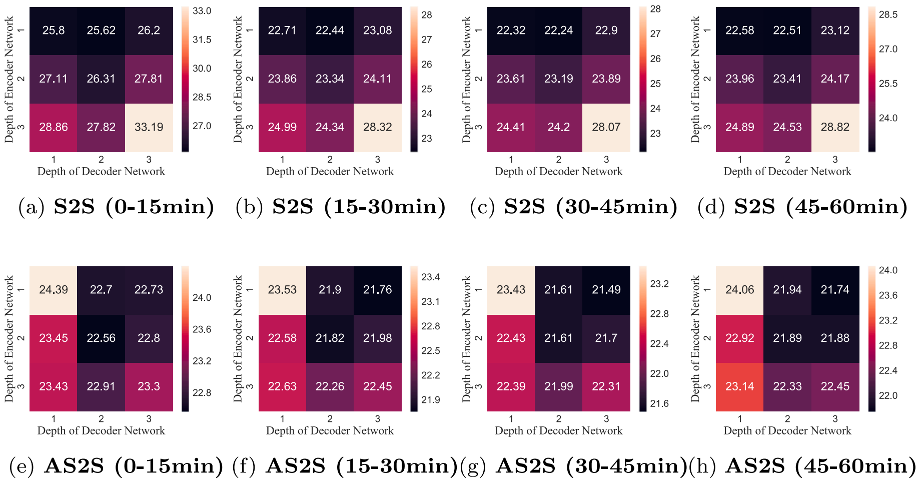


Fig. 5. Comparison of multi-step prediction.



(a) **S2S (0-15min)** (b) **S2S (15-30min)** (c) **S2S (30-45min)** (d) **S2S (45-60min)**

(e) **AS2S (0-15min)** (f) **AS2S (15-30min)** (g) **AS2S (30-45min)** (h) **AS2S (45-60min)**

Fig. 6. Impact of network depth.

encoder network has deeper structure than decoder network, the possible explanation for this phenomenon might be decoder network is closer to output layer and if the decoder is not complicated enough to decode the representations learned by encoder, it will make the model very hard to train. In addition, for S2S models, the prediction performance will drop significantly when increasing the complexity of model structure (the worst candidate is from 3 encoder layers and 3 decoder layers) while AS2S models perform relatively more steadily when the network becomes deeper thanks to the attention mechanism that contributes to relieve encoder network from squashing all the information into a fixed-size vector representation, thus enabling the model to have more complicated structure without much loss of training efficiency.

3.3.3. *Impact of attention mechanism*

Fig. 7 demonstrates the impact of attention mechanism with respect to different model structures. Compared with basic sequence-to-sequence models (S2S), with the help of attention mechanism, the prediction performance are improved by 5.4% to 29.8% for 0 min–15 min prediction, 2.4% to 20.7% for 15 min–30 min prediction, 2.8% to 20.5% for 30 min–45 min prediction as well as 2.5% to 22.1% for 45 min–60 min prediction, especially for models with more complex structure (e.g. deeper network), the gain of attention mechanism is more pronounced.

In addition, the attention matrix for two different input-target paired sequences is visualized in Fig. 8, with rows being the output steps (target sequence) and columns being the input steps. As shown in Fig. 8a and b, the correlation between each step of input sequence and output sequence could be clearly captured, indicating that the model learns to focus on different parts of input sequence when predicting a particular output step, which also could be interpreted as the aggregated travel time patterns are learned by attention mechanism.

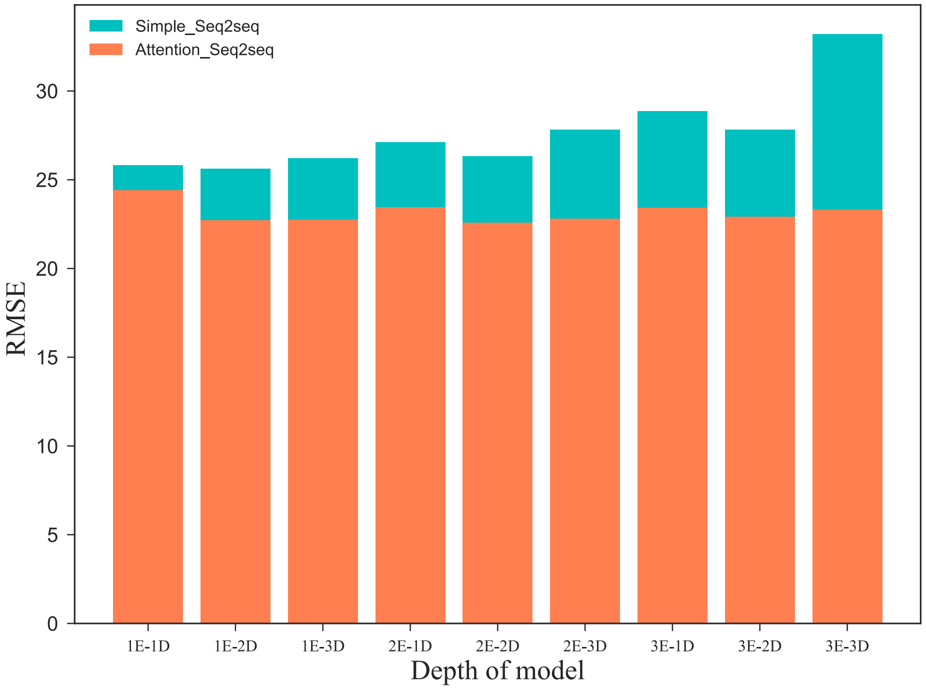
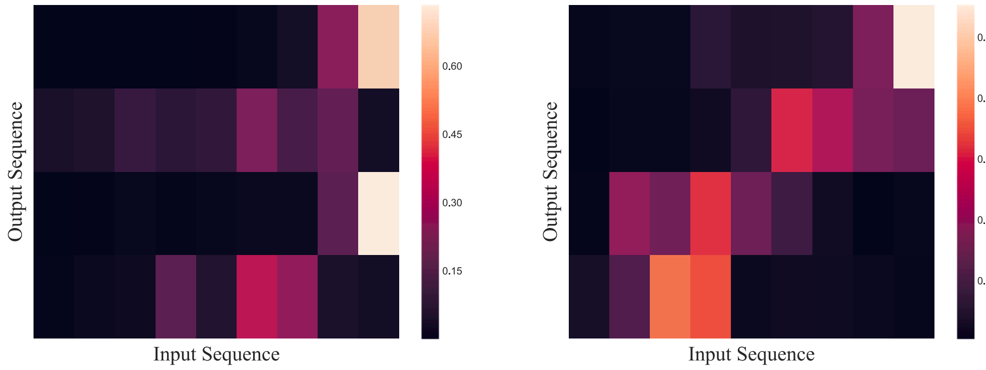


Fig. 7. impact of Attention Mechanism.



(a) **Attention matrix for pair 1**

(b) **Attention matrix for pair 2**

Fig. 8. Visualization of attention matrix.

3.3.4. Impact of external features

Fig. 9 indicates the impact of the fusion with external features. We witness a remarkable performance improvement in terms of both prediction accuracy and speed of convergence by fusing external data (e.g. *Time of day*, *day of week*) using the proposed fusion method. Specifically for training with 1000 epochs, the prediction result is improved by 4.4% to 10.4% for 0 min–15 min prediction, 5.1%-11.0% for 15 min–30 min prediction, 5.1% to 10.9% for 30 min–45 min prediction and 5.7% to 11.4% for 45 min–60 min prediction.

3.3.5. Case analysis

In order to show the prediction results in a more intuitive way, we select 4 stations with different characteristics as examples, of which both the true and predicted temporal alighting demand are plotted in Fig. 10. Raffles Place MRT station (Fig. 10a) and Clark Quay MRT station (Fig. 10b) are both located in bustling downtown area, where Raffles Place is a major working zone in Singapore while Clark Quay has more functions in entertainment. In addition, Yishun MRT station (Fig. 10c) and Haw Par Villa (Fig. 10d) are located out of downtown area and Yishun MRT station is surrounded by high density residential zones while Haw Par Villa MRT station is a small-scale station without much traffic. As shown in Fig. 10, our model is still capable of making reliable predictions simultaneously for multiple stations notwithstanding these stations have totally different demand patterns. Moreover, we also plot

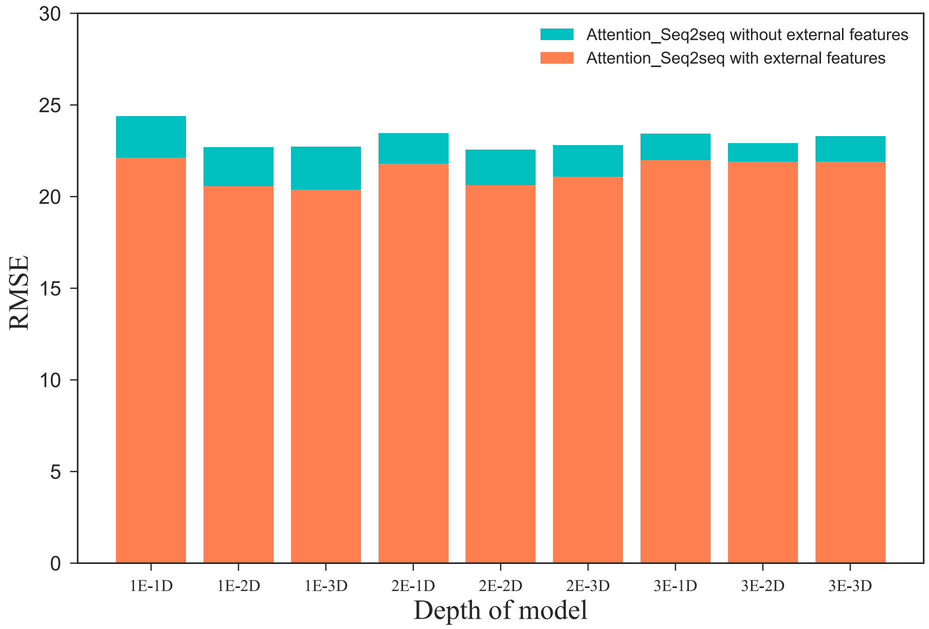
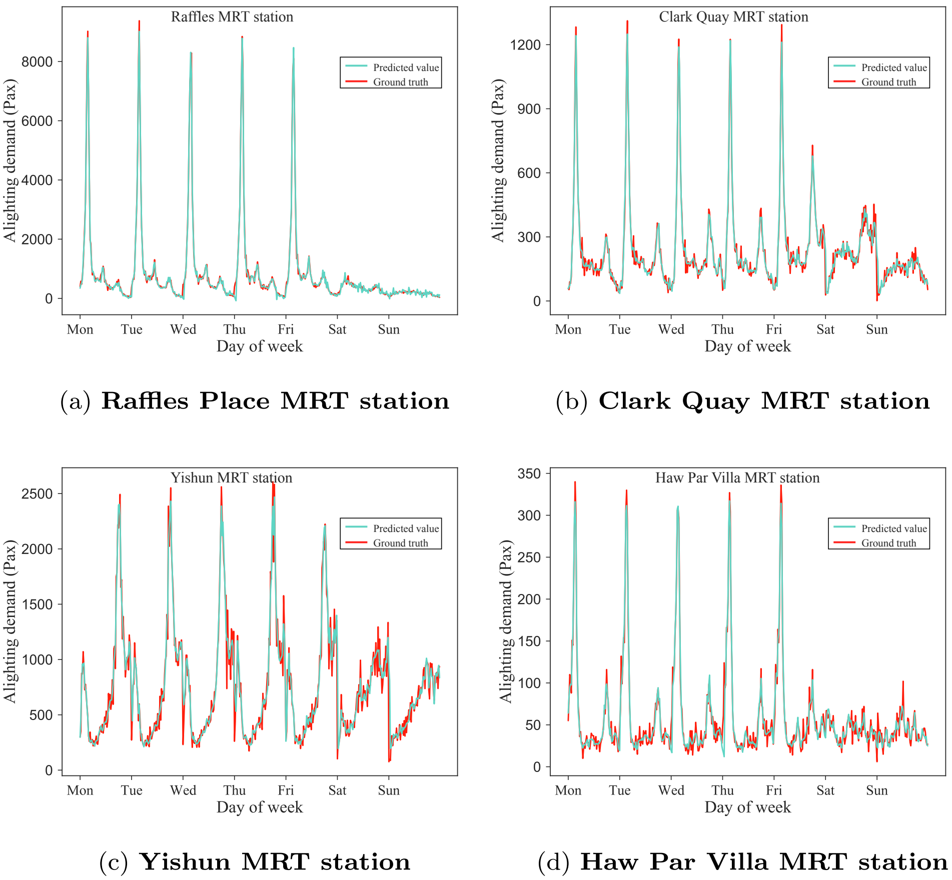


Fig. 9. Impact of external features.



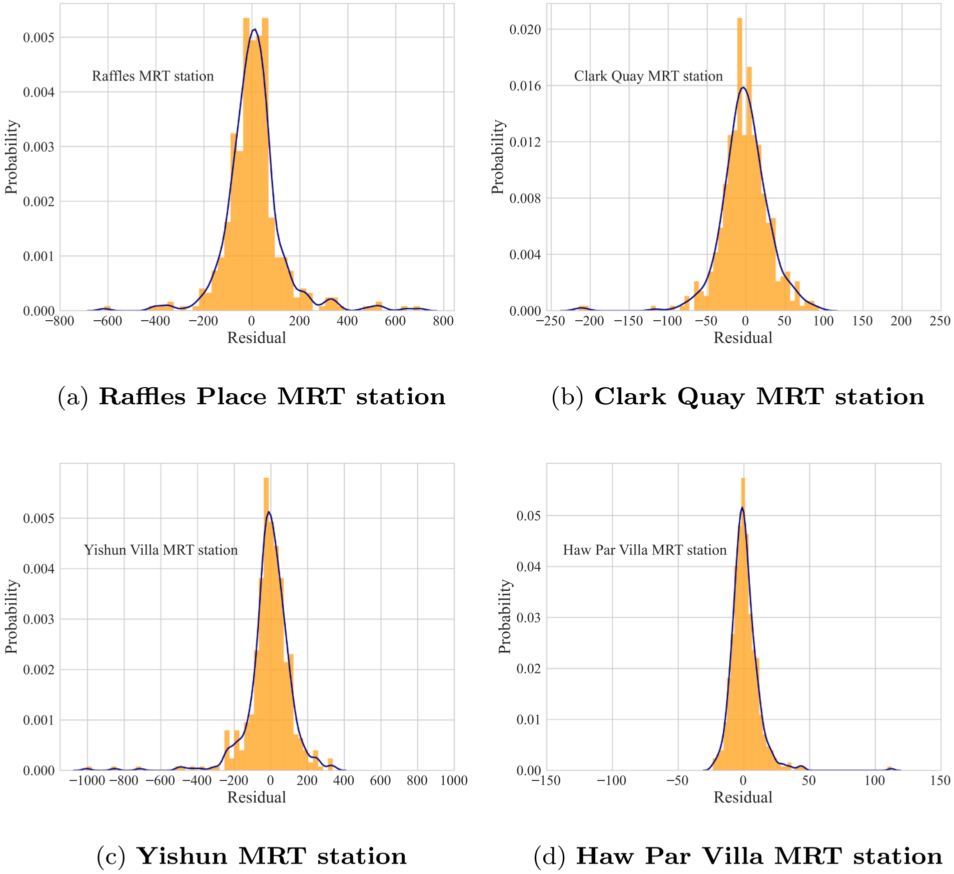
(a) **Raffles Place MRT station**

(b) **Clark Quay MRT station**

(c) **Yishun MRT station**

(d) **Haw Par Villa MRT station**

Fig. 10. Comparisons of the ground truth and predicted passenger alighting demand.



(a) **Raffles Place MRT station**

(b) **Clark Quay MRT station**

(c) **Yishun MRT station**

(d) **Haw Par Villa MRT station**

Fig. 11. Residual plots for selected stations.

out the residuals of the predictions, namely the difference between the ground truth and the predicted value. Residual is an important indicator to assess whether the regression model is systematically correct. As depicted in Fig. 11, the residuals of these 4 predictions are all basically follow normal distributions with zero mean, which indicates that there are no significant non-random patterns in the residuals and our model has already contained sufficient predictive information.

4. Summary and conclusion

In this paper, we propose a novel end-to-end framework to predict short-term passenger flow in a large scale urban metro system. The framework is based on a sequence to sequence learning model embedded with attention mechanism, which has a stacked bidirectional LSTM network as its encoder and another unidirectional LSTM network as its decoder. Sequence to sequence (seq2seq) model can be very practical for many applications in the domain of transportation, especially when the input sequence and output sequence are of different lengths. Besides, with the intervention of attention mechanism, it remarkably enhances the ability of the basic seq2seq model to capture long-range dependencies and greatly improves the prediction accuracy. In addition, we propose a fusion method for merging the external features which is demonstrated to improve the accuracy as well as speed up the convergence significantly. We use the proposed framework to predict the alighting demand at each metro station in the next 60 min by a 15-min interval, according to the boarding demand at each station in the last last few short-term periods, which is validated on large quantities of real-world data from Singapore's metro system. Compared with other state-of-the-art techniques that have been widely used in traffic forecasting related tasks, including ARIMA, support vector regression, Bayesian ridge regression and fully connected artificial neural networks, the proposed model predominates among all the candidates on the measurement of root mean square error (RMSE), especially when making predictions for longer periods (e.g. 30–60 min).

Although the proposed framework has shown its capability of making robust predictions for network-wide short-term passenger flow, there are still some limitations that need further improvement. Firstly, the model could be further trained on a larger dataset that covers a longer time span and some special conditions, so that the generalization ability could be improved and some anomalies in real world could be effectively detected. Secondly, combined with knowledges from passenger flow assignment models, whether the concept of the proposed model could help to better understand how passengers move inside the metro system will be studied in future work.

Acknowledgement

We would like to show our gratitude to Land Transport Authority of Singapore for providing the data. This research is partially supported by the National Natural Science Foundation of China [Grant No. 71701047].

References

Bahdanau, D., Cho, K., Bengio, Y., 2014. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv: 1409.0473.

Bengio, Y., Simard, P., Frasconi, P., 1994. Learning long-term dependencies with gradient descent is difficult. IEEE Trans. Neural Netw. 5, 157–166.

Chan, K.Y., Dillon, T.S., Singh, J., Chang, E., 2012. Neural-network-based models for short-term traffic flow forecasting using a hybrid exponential smoothing and Levenberg–Marquardt algorithm. IEEE Trans. Intell. Transp. Syst. 13, 644–654.

Chen, H., Grant-Muller, S., Mussone, L., Montgomery, F., 2001. A study of hybrid neural network approaches and the effects of missing data on traffic forecasting. Neural Comput. Appl. 10, 277–286.

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

Cong, Y., Wang, J., Li, X., 2016. Traffic flow forecasting by a least squares support vector machine with a fruit fly optimization algorithm. Procedia Eng. 137, 59–68.

Cui, Z., Ke, R., Wang, Y., 2018. Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction. CoRR abs/1801.02143. arXiv: 1801.02143.

Deshpande, M., Bajaj, P.R., 2016. Performance analysis of support vector machine for traffic flow prediction. In: Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC), 2016 International Conference on. IEEE, pp. 126–129.

Dougherty, M.S., Cobbett, M.R., 1997. Short-term inter-urban traffic forecasts using neural networks. Int. J. Forecast. 13, 21–31.

Du, B., Peng, H., Wang, S., Bhuiyan, M.Z.A., Wang, L., Gong, Q., Liu, L., Li, J., 2019. Deep irregular convolutional residual lstm for urban traffic passenger flows prediction. IEEE Trans. Intell. Transp. Syst.

Duan, Y., Lv, Y., Wang, F.Y., 2016. Travel time prediction with lstm neural network. In: Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on. IEEE, pp. 1053–1058.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9, 1735–1780.

Hou, Y., Edara, P., 2018. Network scale travel time prediction using deep learning. Transp. Res. Rec 0361198118776139.

Hou, Y., Edara, P., Sun, C., 2015. Traffic flow forecasting for urban work zones. IEEE Trans. Intell. Transp. Syst. 16, 1761–1770.

Hu, Z., Zuo, Y., Xue, Z., Ma, W., Zhang, G., 2017. Predicting the metro passengers flow by long-short term memory. In: Advances in Computer Science and Ubiquitous Computing. Springer, pp. 591–595.

Hua, J., Faghri, A., 1994. Apphcations of artificial neural networks to intelligent vehicle-highway systems. Transp. Res. Rec. 1453, 83.

Jia, Y., Wu, J., Xu, M., 2017. Traffic flow prediction with rainfall impact using a deep learning method. J. Adv. Transp. 2017.

Ke, J., Zheng, H., Yang, H., Chen, X.M., 2017. Short-term forecasting of passenger demand under on-demand ride services: a spatio-temporal deep learning approach. Transp. Res. Part C: Emerg. Technol. 85, 591–608.

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., 1998. Gradient-based learning applied to document recognition. Proc. IEEE 86, 2278–2324.

Li, J., Peng, H., Liu, L., Xiong, G., Du, B., Ma, H., Wang, L., Bhuiyan, M.Z.A., 2018. Graph cnns for urban traffic passenger flows prediction. In: 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI). IEEE, pp. 29–36.

Liang, Y., Ke, S., Zhang, J., Yi, X., Zheng, Y., 2018. Geoman: Multi-level attention networks for geo-sensory time series prediction. In: IJCAI, pp. 3428–3434.

Liao, B., Zhang, J., Wu, C., McIlwraith, D., Chen, T., Yang, S., Guo, Y., Wu, F., 2018. Deep sequence learning with auxiliary information for traffic prediction. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, pp. 537–546.

Liu, B., Yan, S., Li, J., Qu, G., Li, Y., Lang, J., Gu, R., 2018. An attention-based air quality forecasting method. In: 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA). IEEE, pp. 728–733.

Liu, L., Chen, R.C., 2017. A novel passenger flow prediction model using deep learning methods. Transp. Res. Part C: Emerg. Technol. 84, 74–91.

Lv, Y., Duan, Y., Kang, W., Li, Z., Wang, F.Y., et al., 2015. Traffic flow prediction with big data: a deep learning approach. IEEE Trans. Intell. Transp. Syst. 16, 865–873.

Ma, X., Dai, Z., He, Z., Ma, J., Wang, Y., Wang, Y., 2017. Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction. Sensors 17, 818.

Ma, X., Tao, Z., Wang, Y., Yu, H., Wang, Y., 2015. Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Transp. Res. Part C: Emerg. Technol. 54, 187–197.

Ma, X., Zhang, J., Du, B., Ding, C., Sun, L., 2018. Parallel architecture of convolutional bi-directional lstm neural networks for network-wide metro ridership prediction. IEEE Trans. Intell. Transp. Syst.

Mingheng, Z., Yaobao, Z., Ganglong, H., Gang, C., 2013. Accurate multisteps traffic flow prediction based on svm. Math. Problems Eng. 2013.

Pascale, A., Nicoli, M., 2011. Adaptive bayesian network for traffic flow prediction. In: Statistical Signal Processing Workshop (SSP), 2011 IEEE. IEEE, pp. 177–180.

Polson, N., Sokolov, V., et al., 2015. Bayesian analysis of traffic flow on interstate i–55: the lwr model. Ann. Appl. Stat. 9, 1864–1888.

Polson, N.G., Sokolov, V.O., 2017. Deep learning for short-term traffic flow prediction. Transp. Res. Part C: Emerg. Technol. 79, 1–17.

Qu, L., Li, W., Li, W., Ma, D., Wang, Y., 2019. Daily long-term traffic flow forecasting based on a deep neural network. Expert Syst. Appl. 121, 304–312.

Smith, B.L., Demetsky, M.J., 1994. Short-term traffic flow prediction models-a comparison of neural network and nonparametric regression approaches. In: Systems, Man, and Cybernetics, 1994. Humans, Information and Technology., 1994 IEEE International Conference on. IEEE, pp. 1706–1709.

Smith, B.L., Demetsky, M.J., 1997. Traffic flow forecasting: comparison of modeling approaches. J. Transp. Eng. 123, 261–266.

Sun, S., Zhang, C., Zhang, Y., 2005. Traffic flow forecasting using a spatio-temporal bayesian network predictor. In: International Conference on Artificial Neural Networks. Springer, pp. 273–278.

Sutskever, I., Vinyals, O., Le, Q.V., 2014. Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, pp. 3104–3112.

Tan, M.C., Wong, S.C., Xu, J.M., Guan, Z.R., Zhang, P., 2009. An aggregation approach to short-term traffic flow prediction. IEEE Trans. Intell. Transp. Syst. 10, 60–69.

Tebaldi, C., West, M., 1998. Bayesian inference on network tra c using link count data. J. Am. Stat. Assoc. 93, 557–573.

Toqué, F., Côme, E., El Mahrsi, M.K., Oukhellou, L., 2016. Forecasting dynamic public transport origin-destination matrices with long-short term memory recurrent neural networks. In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC). IEEE, pp. 1071–1076.

Toqué, F., Khouadjia, M., Come, E., Trepanier, M., Oukhellou, L., 2017. Short & long term forecasting of multimodal transport passenger flows with machine learning methods. In: 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC). IEEE, pp. 560–566.

Van Der Voort, M., Dougherty, M., Watson, S., 1996. Combining kohonen maps with arima time series models to forecast traffic flow. Transp. Res. Part C: Emerg. Technol. 4, 307–318.

Van Lint, J., Van Hinsbergen, C., 2012. Short term traffic and travel time prediction models, in artificial intelligence applications to critical transportation issues. Transportation Research Circular. National Academies Press, Washington DC Number E-C168.

Vlahogianni, E.I., Karlaftis, M.G., Golias, J.C., 2014. Short-term traffic forecasting: Where we are and where we're going. Transp. Res. Part C: Emerg. Technol. 43, 3–19.

Wang, J., Chen, R., He, Z., 2019. Traffic speed prediction for urban transportation network: a path based deep learning approach. Transp. Res. Part C: Emerg. Technol. 100, 372–385.

Wei, Y., Chen, M.C., 2012. Forecasting the short-term metro passenger flow with empirical mode decomposition and neural networks. Transp. Res. Part C: Emerg.

Technol. 21, 148–162.

Williams, B., Durvasula, P., Brown, D., 1998. Urban freeway traffic flow prediction: application of seasonal autoregressive integrated moving average and exponential smoothing models. Transp. Res. Rec. 1644, 132–141. https://doi.org/10.3141/1644-14.

Williams, B.M., Hoel, L.A., 2003. Modeling and forecasting vehicular traffic flow as a seasonal arima process: theoretical basis and empirical results. J. Transp. Eng. 129, 664–672.

Wu, C.H., Wei, C.C., Su, D.C., Chang, M.H., Ho, J.M., 2003. Travel time prediction with support vector regression. In: Intelligent Transportation Systems, 2003. Proceedings. 2003 IEEE. IEEE, pp. 1438–1442.

Xiao, H., Sun, H., Ran, B., Oh, Y., 2003. Fuzzy-neural network traffic prediction framework with wavelet decomposition. Transp. Res. Rec. 1836, 16–20.

Xu, C., Ji, J., Liu, P., 2018. The station-free sharing bike demand forecasting with a deep learning approach and large-scale datasets. Transp. Res. Part C: Emerg. Technol. 95, 47–60.

Yi, H., Jung, H., Bae, S., 2017. Deep neural networks for traffic flow prediction. In: Big Data and Smart Computing (BigComp), 2017 IEEE International Conference on. IEEE, pp. 328–331.

Yin, H., Wong, S., Xu, J., Wong, C., 2002. Urban traffic flow prediction using a fuzzy-neural approach. Transp. Res. Part C: Emerg. Technol. 10, 85–98.

Yu, H., Wu, Z., Wang, S., Wang, Y., Ma, X., 2017. Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks. Sensors 17, 1501.

Zhang, H., Wu, H., Sun, W., Zheng, B., 2018. Deeptravel: a neural network based travel time estimation model with auxiliary supervision. arXiv preprint arXiv: 1802.02147.

Zhang, J., Zheng, Y., Qi, D., 2017. Deep spatio-temporal residual networks for citywide crowd flows prediction. In: AAAI, pp. 1655–1661.

Zhang, W., Yu, Y., Qi, Y., Shu, F., Wang, Y., 2019. Short-term traffic flow prediction based on spatio-temporal analysis and cnn deep learning. Transp. A: Transp. Sci. 1–45.

Zhao, Z., Chen, W., Wu, X., Chen, P.C., Liu, J., 2017. Lstm network: a deep learning approach for short-term traffic forecast. IET Intel. Transp. Syst. 11, 68–75.

Zheng, W., Lee, D.H., Shi, Q., 2006. Short-term freeway traffic flow prediction: Bayesian combined neural network approach. J. Transp. Eng. 132, 114–121.

Zhu, Z., Peng, B., Xiong, C., Zhang, L., 2016. Short-term traffic flow prediction with linear conditional gaussian bayesian network. J. Adv. Transp. 50, 1111–1123.