THE UNIVERSITY OF QUEENSLAND

AUSTRALIA

Passenger Flow Prediction Based On Public Transport Smart Card Data

by Xiaoqi Zhuang

School of Information Technology and Electrical Engineering,

The University of Queensland.

Submitted for Master of Data Science Capstone 3

16th November, 2020

Abstract

Buses are an important part of public transportation in many countries and cities, including Australia and Brisbane. In order to better balance the cost and efficiency of the bus system, a model that can accurately predict the passenger flow of buses has become a hot research topic in recent years. To this concern, this project is decomposed into four tasks: exploratory data analysis; data processing; model building; web design. Firstly, exploratory data analysis help have some background knowledge about the given dataset and also provides advice on feature selection. Then, the original information is integrated into passenger flow data every 15 minutes. It is also combined with the auxiliary weather data set. During this process, the features which the final data set uses are determined through Pearson correlation coefficient, graph, etc. The model building process tries different regression models such as LSTM, GRU with different parameters to achieve the best accuracy. Finally, the finished model has been deployed to the web page. This webpage simulates the real passenger flow prediction demand. The input on webpage only needs date and it will automatically match the needed data and show both numerical and visualization prediction. The purpose of this is to allow those who do not have programming skills to make passenger flow prediction analysis. This report explains the details of the above steps and discusses the forecast results and the bus passenger flow in Brisbane. The final model prediction result obtains an RMSE of 27.58 in the data set from July 2018 to June 2019 and can predict the trend of passenger flow very accurately.

Table of Content

1. Introduction

This project aims to design a website for predicting every 15 minutes' bus passenger flow for Route 66 on UQ lake station in Brisbane. For UQ lake station, crowds and queues always appear while students are waiting to board the bus. Hence, passenger flow in this project is specifically referred to as on-board passenger flow. The original dataset in this project is collected by every passenger tapping their smart cards when getting on and off. Hence, the data set has more than 10 million records in total. In recent years, researchers have begun to predict passenger flow based on these extremely large datasets in order to solve the growing public transportation problem. With reference to the related work (section 1.2), the choice of model is locked in traditional machine learning methods, time series methods, and deep learning models such as random forest regression, ARIMA, LSTM.

In this chapter, section 1.1 describes the background of the project. Then, section 1.2 introduces the related work. Finally, section 1.3 demonstrates the objectives of this report.

1.1 Background

In this section, I will introduce the background for this project, including the current status of buses in Brisbane and the necessity of building a model for the passenger flow.

In Australia, buses are the fourth most popular mode of transportation for people. However, compared with the top three: private cars, trains, walking, the time that the bus needs to cover needs to be as long as possible, the frequency needs to be as short as possible, and should cover the entire city. Across the entire August in 2019 on the Brisbane bus network there were 238,298 total trips, and the number of full standing loads was 3228 (Lucy Stone, 2019). Another considerable example is that the route 66, which is the busiest bus in Brisbane across the University of Queensland and Brisbane city, has 6 percent of it reported as a “full standing load” even though this route has 282 daily buses (Lucy Stone, 2019). Generally, students can only get on the bus at the city and school stops during peak hours, while they may wait in line at other stops or have to stand in a fully loaded bus. The Brisbane government has also been paying attention to this issue and expects to add more routes to alleviate it. Therefore, a model that can predict bus passenger flow in a specific area and period will make resource usage more efficiently and also prevent waste. For example, for time intervals or stopes with high/low passenger flow, actions on increasing/reducing bus frequencies can be taken.

Since Route 66 and UQ Lake Station are the most demanding areas for buses (more details in section 2.1), this project focus on building a model that can predict the on-board passenger flow of Route 66 on UQ Lake Station every 15 minutes to balance demand. Once such a model is successfully implemented, it is easy to reuse it on other sites or other lines.

1.2 Related Work

In this section, I will present the existing techniques and compare the current applications. Researchers in recent years have developed plenty of techniques for passenger flow prediction.

At present, there are 3 common approaches for passenger flow prediction: different kinds of neural network models; traditional machine learning methods; time series models.

- Neural Networks: Recurrent Neural Network is an efficient model for time series analysis and Convolutional Neural Network is good at image recognition. Hence, a mixture model called LC-RNN is used to predict bus passenger flow (Lv, Z., Xu, J., Zheng, K., Yin, H., Zhao, P., & Zhou, X, 2018). It constructs a speed matrix formed by each road and its connected roads, then uses CNN to extract the spatial dependencies of adjacent "road speeds", and then further uses RNN to extract features from a time series perspective. LC-RNN shows good performance which has the lowest RMSE between 7 well-known existing methods.

- Traditional machine learning methods: Support Vector Machine (SVM) model has been used in the traffic prediction (Chen, Q., Li, W., & Zhao, J, 2011). The accuracy of the prediction by traditional machine learning methods can be adjusted by different parameters. A grey linear model also has been used for the off-board passenger flow in railway stations (Hou, L. M., & Ma, G. F, 2011).
- Time series models: An autoregressive integrated moving average (ARIMA) model for time series data has been used for the passenger flow prediction (Cai, C., Yao, E., Wang, M., & Zhang, Y. 2014). ARIMA model is a hybrid model including the autoregressive (AR) and moving average (MR) models and it can consider seasonality when predicting.

Additionally, exploratory data analysis (EDA) is a common way for researchers to have insights about the dataset by plenty of plots and feature engineering. A research for bus passenger flow prediction in Adelaide Metropolitan area used EDA to get the number of times only single person board from the bus stop, the busiest bus route, the bus stops which become more popular and so on (Gurvinder Singh, 2019). EDA also gives a more intuitive visualization for non-professionals such as government, executive departments to formulate the bus schedule. Therefore, I use EDA firstly to obtain deep background understanding and choose several models in this project.

1.3 Project Objectives

This project aims to build a model based on machine learning techniques, and the overall objectives can be divided into two parts:

- a model with high accuracy for bus passenger flow
- a website based on Flask which can interact with users

Given the challenging nature of the problems, I decomposed the process of model building into 5 parts: data processing; feature selection; exploratory data analysis; model building; benchmark evaluation. In order to improve the accuracy of passenger flow, this project will apply different approaches to achieve the following functionalities in each stage:

- Data processing by joining different datasets and data cleaning to fix unstructured data
- Feature selection by Pearson correlation coefficient and wrapper method (Guyon, I., & Elisseeff, A., 2003).
- Exploratory data analysis for data visualization
- Different model fitting for accurate prediction
- Benchmark evaluations for comparison with other peer models

Finally, since people in the public transportation department may lack background knowledge of machine learning models, the model with the lowest RMSE from the benchmark table has been chosen and deployed on the website for users to predict. The website can show visual charts and numerical passenger flow situation on the day which users provide.

2. Design

2.1 The description of dataset

The chosen dataset is collected by Translink, which includes information from July 2018 to June 2019. More specifically, the dataset records information for every passenger such as the boarding time and stop, the alighting time and stop. However, the target of this project is to predict passenger flow in a period, which does not exist in the dataset. This set of target values can be obtained according to the "GROUP BY" and "SUM" function.

The boarding time in the original data set is accurate to the second. However, for time period passenger flow prediction, it is necessary to normalize the boarding time to one time period. In this project, the normalized time period is set to 15 minutes, because Route 66 runs every 15 minutes.

Also, the raw dataset has missing or outlier values. For example, there are 535,848 (5%) records which their "Passengers" feature is 0 or negative values. In this project, all records that do not conform to the specification are deleted because their proportion in the data set is small.

Moreover, this dataset still lacks a lot of features required to predict bus passenger flow. The bus passenger flow should have a significant relationship with the temperature or other weather conditions, which can be proved in section 2.2. Hence, I also choose the "SILO" dataset about weather conditions from Queensland Government.

A preliminary processed data set is obtained by combining the passenger flow data set and the weather data set.

	Date	Hour	Period	Passengers	Weekday	Semester	TMax	TMin	Rain
0	2018-07-01	6	45	1.0	6	0	22.4	15.1	0.0
1	2018-07-01	7	15	1.0	6	0	22.4	15.1	0.0
2	2018-07-01	7	30	2.0	6	0	22.4	15.1	0.0
3	2018-07-01	7	45	3.0	6	0	22.4	15.1	0.0
4	2018-07-01	8	15	5.0	6	0	22.4	15.1	0.0

Figure 2.1 A sample of the final data set.

2.2 Exploratory data analysis

If the accurate passenger flow prediction model can be built for the busiest stops and busiest routes, the problem of bus supply and demand balance seems to be solved. Obviously, charts can help achieve these kinds of information.

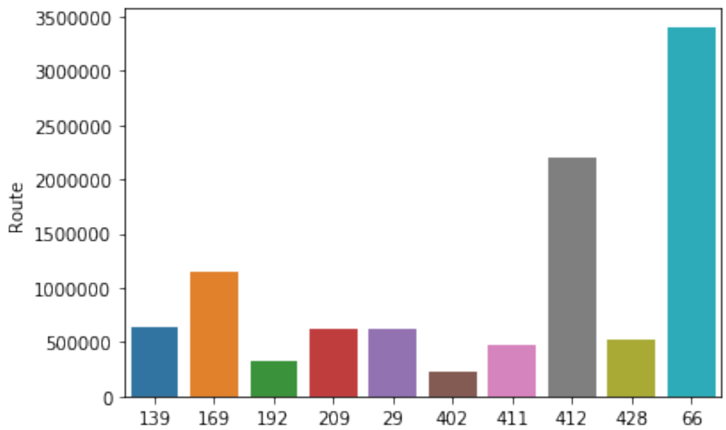


Figure 2.2 Bar plot of Routes

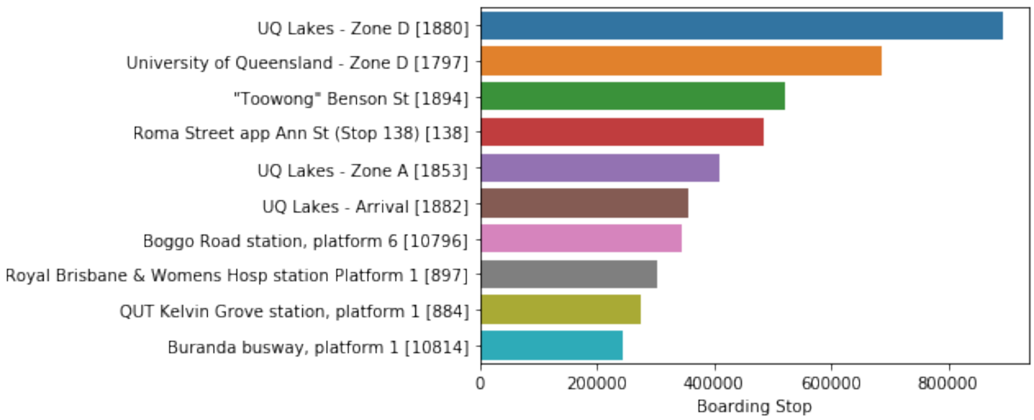


Figure 2.3 Bar plot of Boarding Stop

Figure 2.1, Figure 2.2 shows that the top ten routes are all related to UQ, and four of the top ten stops are on the UQ campus. Hence, this project aims on building a model that can predict the on-board passenger flow of Route 66 on UQ Lake Station every 15 minutes to balance demand. Once such a model is successfully implemented, it is easy to reuse it on other sites or other lines.

According to common sense, passenger flow should be closely related to time. Time series graphs can verify this.

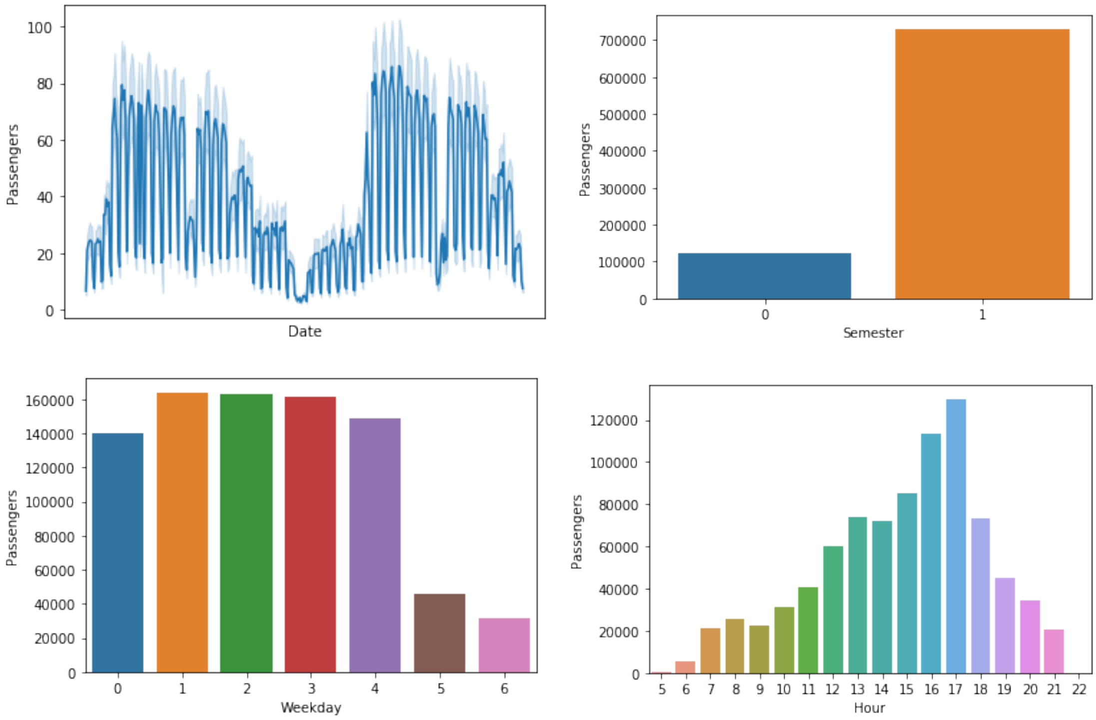
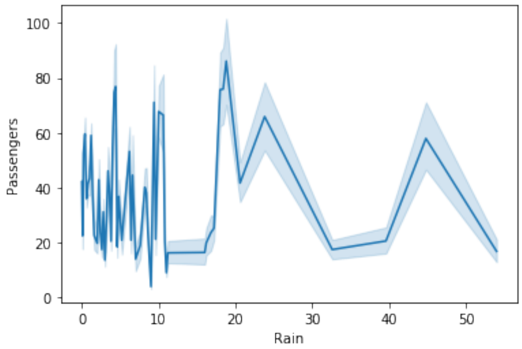


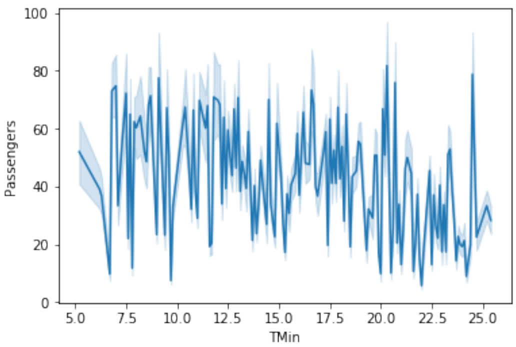
Figure 2.3 plot between Date, Semester, Weekday and Hour

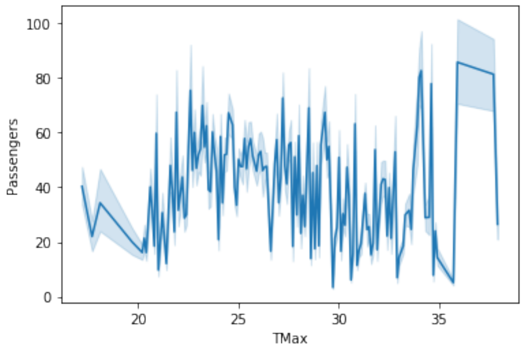
Figure 2.3 in the upper left corner shows that passenger flow is indeed a time series. And it has obvious seasonality property. According to the query, it is found that the time period when the average passenger flow is significantly higher is the period during the semester, and the lower is the period during the holiday. The figure in the upper right corner also proves this situation. The figure below also shows that different weekdays and different time periods have a great impact on passenger flow.

Moreover, EDA also can provide suggestion about feature selection.



Pearson Correlation = -0.07, P-value = 9.24e-26





Pearson Correlation = -0.08, P-value = 2.25e-27 Pearson Correlation = -0.12, P-value = 5.61e-70

Figure 2.4 correlation between weather information and passenger flow

Figure 2.4 shows that weather information such as max temperature, min temperature, rainfall has a negative impact on passenger flow, since all p-values are quite small and Pearson correlation coefficients are negative.

After this section, this project determines that the target value to be predicted is every 15 minutes passenger flow of Route 66 on UQ Lake Station. Also, it reflects the influence of attributes such as date and time period on passenger flow, which lays a good foundation for the decision to use the time series model and feature selection below.

2.3 Feature selection

The data set has reasons to include weather conditions according to their Pearson correlation coefficient in Figure 2.4. Moreover, according to the huge difference in passenger flow between semester and holiday and the huge difference in passenger flow between weekdays and weekends shown in Figure 2.3, the project decided to use two ways and choose the better one through evaluation.

The first way is to add two new features called ‘weekday’ and ‘semester’ into the dataset and the second way is to split the dataset into 4 parts: semester and workday; holiday and workday; semester and weekend; holiday and weekend.

The wrapper method is used to evaluate them. This method will measure whether to select this feature by comparing the prediction accuracy of the data set with increasing or decreasing features into a model. Putting the data into the model obtained in section 2.5, you can find that the RMSEs of dividing the data set into four parts are all lower than that of a data set with two new features.

	One dataset	**Semester Workday**	**Semester Weekend**	**Holiday Workday**	**Holiday Weekend**
RMSE	29.543	28.727	19.006	21.245	8.377

Table 1. RMSEs for different data sets

Hence, this project finally decided to have these 4 data sets and 4 models for them.

2.4 Model building

Section 2.2 and Section 2.3 show that the prepared data set is a multivariate time series sequence. Hence, this project will implement “Long Short-Term Memory” (LSTM) model (Hochreiter, S., & Schmidhuber, J., 1997), which is also the most popular model in this type of problem.

LSTM is inspired by Recurrent neural network (RNN). The architecture of RNN is shown in Figure 2.5. The recurrent neural network will remember the previous information and apply it to the calculation of the current outputs, and the inputs of the hidden layers include not only the outputs of the input layers but also the outputs of the hidden layers at the previous time.

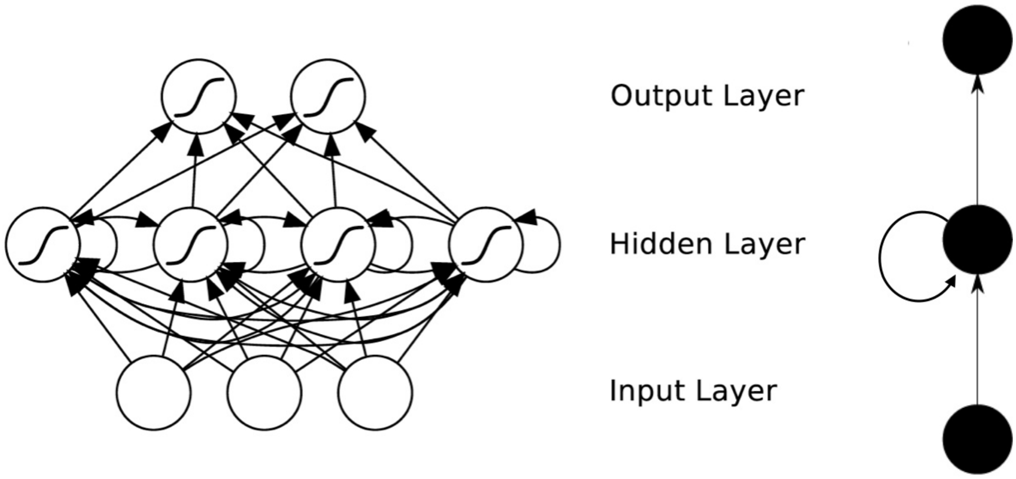


Figure 2.5. RNN architecture (Evan, 2018)

Based on the RNN structure, LSTM has created input gate, forget gate, and output gate. The input gate is used to determine whether to put the input into the memory. The forget gate judges whether to reset the memory and the output gate judges whether to output the number in the memory. This design makes the model closer to real life. In a time series sequence, data which time gap is too long has no effect on prediction. LSTM uses forget gate to reset memory in time to solve this problem.

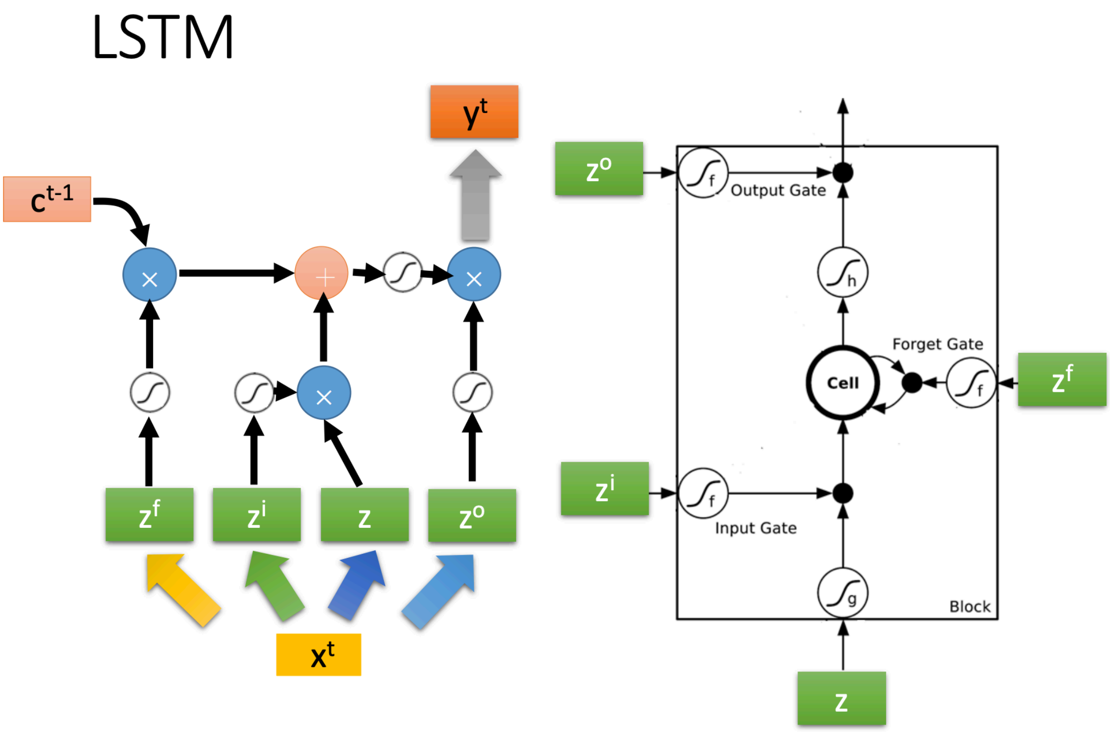


Figure 2.6. LSTM architecture (Hung-yi Lee, 2020)

Figure 2.6 is a neural structure of LSTM, which will return the value if the output gate is open.

After deciding which model to choose, how to apply the model to the real data set is also a problem.

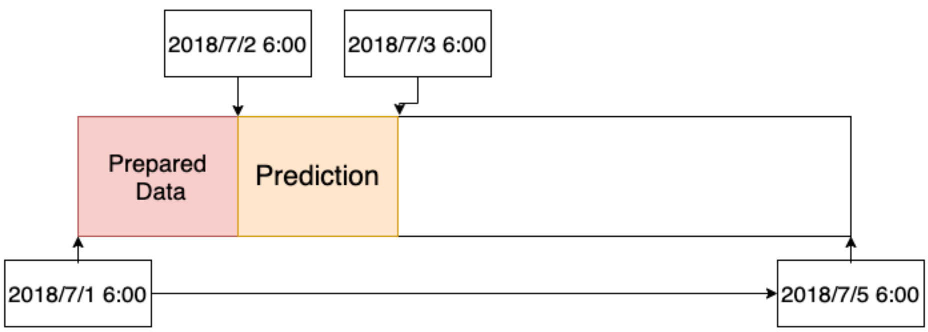


Figure 2.7. A prediction process example

Figure 2.7 is a simple prediction example which describes the prediction process of this project: using all the data of the previous day to predict the passenger flow data of the next day.

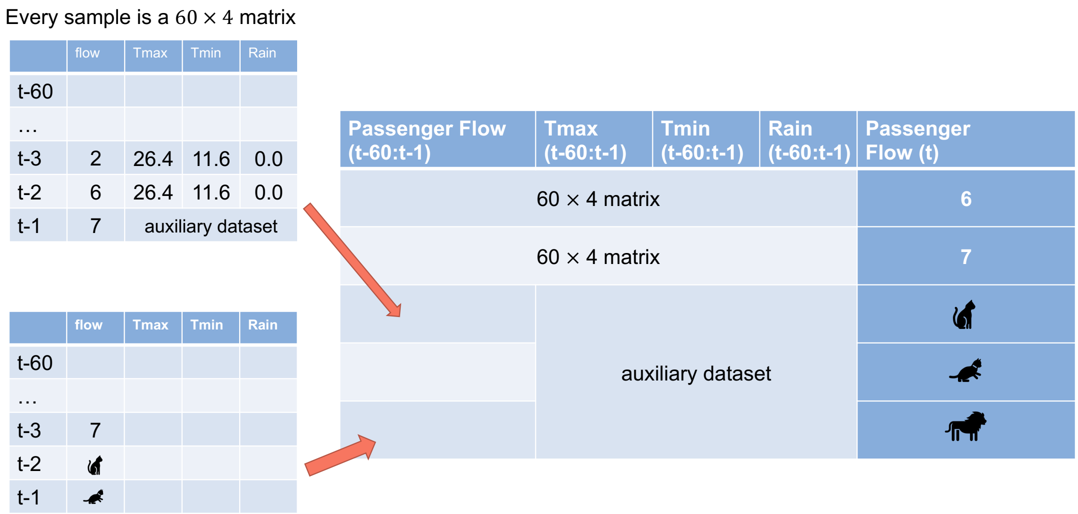


Figure 2.8. A detailed example

Figure 2.8 introduces how to apply our prediction process to the LSTM model. Each input is a 60 × 4 matrix containing information about the previous day. And the output is the passenger flow number of the next period. When the passenger flow at the first moment of the forecast day is predicted, use this data as the last row of the input matrix. Then delete the information in the first row of the original matrix to ensure the input matrix is always 60 × 4. Then use this input matrix to predict the next period's passenger flow until the passenger flow at the last period of the forecast day is predicted. The Input matrix is set to 60 × 4 because this project combined with the data set to make the assumption that Route 66 daily schedule is from 6:00 to 21:00, which has 15 hours, and each hour has four 15 minutes. Hence, every sample should have 60 time periods and 4 features: Max temperature, Min temperature, Rainfall, Passenger flow number.

In summary, the input shape of the LSTM model in this project will be [number of samples, 60, 4]. The number of LSTM layers and Neurons will be discussed in Section 2.5. The loss function is 'mean-square error, and the optimizer is 'ADAM' gradient descent.

2.5 Evaluation

This section will compare different models with different parameters to select the best model.

Firstly, the model that use a complete data set will be compared with models which separate weekdays and semesters. The model structure at this stage is unified into 2 LSTM layers, each LSTM layer has 256 neurons.

	One dataset	**Semester Workday**	**Semester Weekend**	**Holiday Workday**	**Holiday Weekend**
RMSE	29.543	28.727	19.006	21.245	8.377

Table 2. RMSEs for different data sets

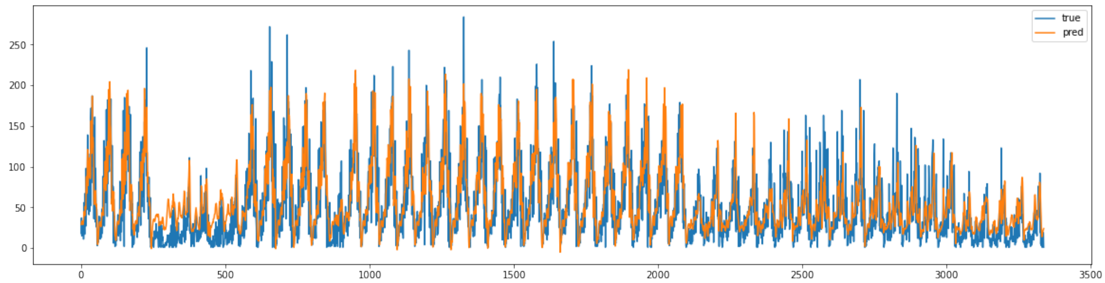


Figure 2.5 Prediction and Ground Truth plot of model for semester and workday dataset

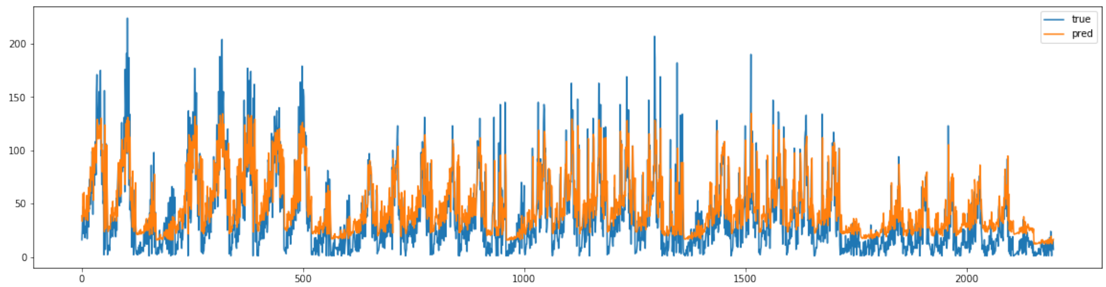


Figure 2.6 Prediction and Ground Truth plot of model for the whole dataset

In addition to RMSE, the comparison between prediction and ground truth values can also reflect the prediction effects of the two models. Figure 2.5 and Figure 2.6 show that the predictive effect of the semester-workday model is better than the model using the whole data set. This is reflected in the prediction of the time period for extremely small passenger flow and high passenger flow. Although both models can successfully simulate the trend of passenger flow, the model using the whole data set is difficult to predict very small passenger flow and extremely large passenger flow. This may be due to excess noise interfering with it. But for the project problem we need to solve: the balance of supply and demand for buses, the semester-workday model, or in other words, separates the data to build a model to predict more accurately.

Then, the model will compare the RMSE of different structures to get the most suitable structure. The structure will be adjusted from the number of neurons and the number of LSTM layers. Through empiricism, the number of neurons is selected as 128, 256, and 512. The number of layers of LSTM is set to 2 or 3 layers.

Dataset	Metric	Neuron numbers:	512	256	128
Holiday, weekend	RMSE		8.639	8.555	8.377
Holiday, workday			23.126	22.791	22.567
Semester weekend			19.006	19.402	19.491
Semester workday			32.888	33.955	32.491

Table 3. Benchmark table for different neuron numbers for 3-layer LSTM models

Dataset	Metric	Neuron numbers:	512	256	128
Holiday, weekend	RMSE		8.582	8.913	8.753
Holiday, workday			22.536	24.429	21.245
Semester weekend			19.333	19.406	19.745
Semester workday			32.812	28,727	30.369

Table 4. Benchmark table for different neuron numbers for 2-layer LSTM models

Table 3 and 4 introduces the performances of LSTM models with different structures. Model with the best performance (the lowest RMSE) has been finally chosen in this project. Moreover, this project also builds other models for comparison. Gated Recurrent Units (GRU) (Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y., 2014) is another variant of the RNN model. With the same parameters as LSTM, Table 5 shows that LSTM models are all better than GRU models.

Dataset	LSTM	GRU
Holiday, weekend	8.377	9.036 (+7.97%)
Holiday, workday	21.245	24.093 (+13.4%)
Semester weekend	19.006	20.121 (+5.87%)
Semester workday	28.727	31.678 (+10.27%)

Table 5. Benchmark table for LSTM and GRU performance

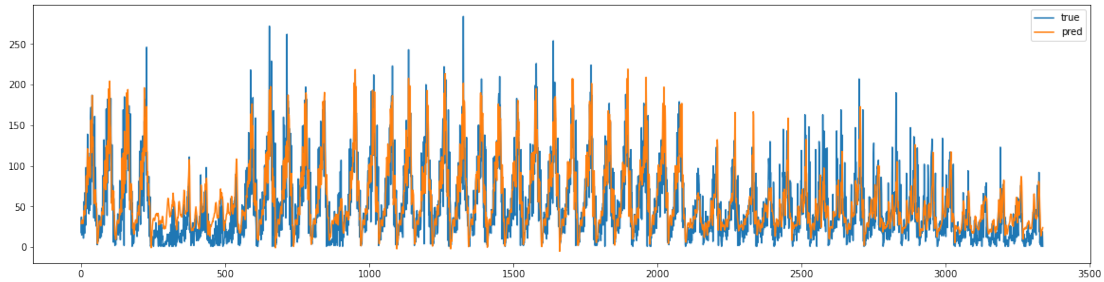


Figure 2.9. The Prediction and Ground Truth plot of Semester-Workday dataset

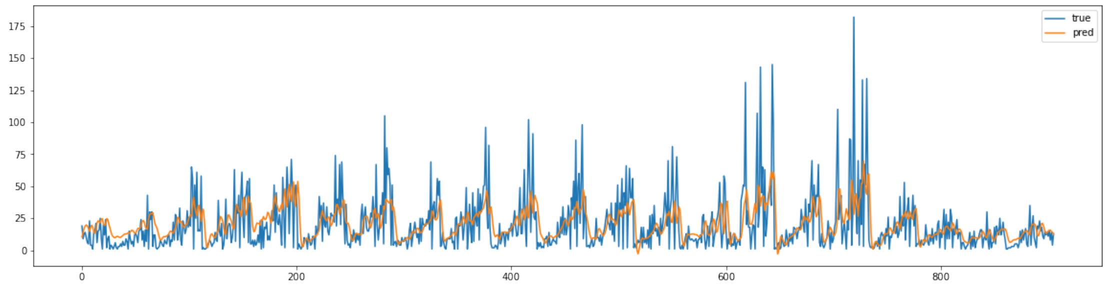


Figure 2.10. The Prediction and Ground Truth plot of Semester-Weekend dataset

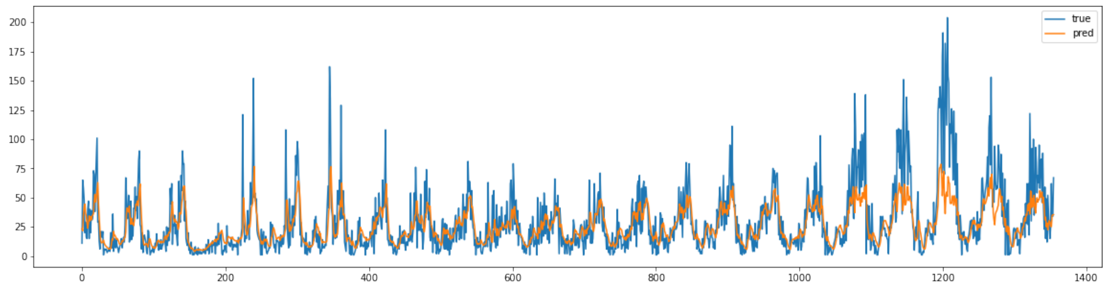


Figure 2.11. The Prediction and Ground Truth plot of Holiday-Workday dataset

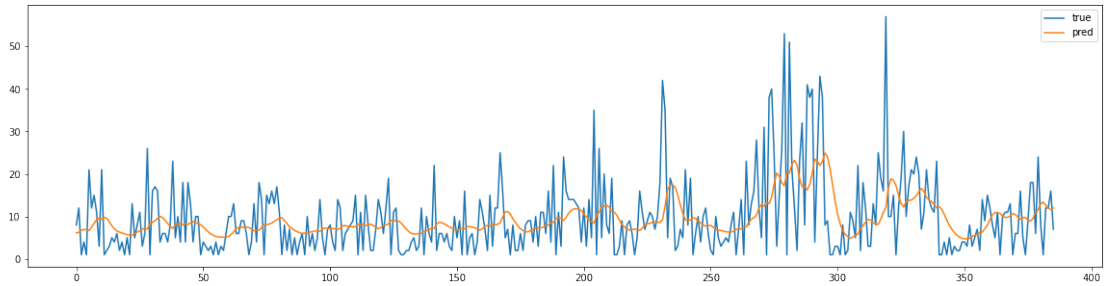


Figure 2.12. The Prediction and Ground Truth plot of Holiday-Weekend dataset

The above four figures compare the predicted results of the model with the true values. These figures show that all four models can accurately predict the trend of passenger flow. And the Semester-Workday dataset can predict extreme values very well. The other three are slightly conservative in predicting extreme values. This may be because the data sets are too small to train a neural network well. However, combined with the actual situation, the passenger flow during weekends or holidays is not important in our prediction. Because their extreme values are still within the tolerance of the bus station and the frequency is not high.

3. Application

3.1 Functionality

Training a suitable neural network model requires many attempts. This contains a lot of repeated steps. Therefore, this project also designed 3 functions to encapsulate these operations. This makes training LSTM easier and optimizes code readability.

The first part is the data processing part. As shown in the figure, the data shape required by the LSTM model is specific. You need to manually convert the original data into the shape required by LSTM.

```python
def dataProcessing(dayNum, dataset):
    """
    A method for users to make multivariate time series for LSTM
    @param: dayNum, the number of days you want to use as the training set.
    @param: dataset, the period you want to predict, s.t. study_weekend, study_weekday,
                                                          holiday_weekend, holiday_weekday
                     Dataset should be time series.
                     Dataset features(order important): PassengerFlow, TMax, TMin, Rain
    @return: X_train, y_train, X_test, y_test.  Training set 70%, Testing set 30%
    """
    time_step =  dayNum * 15 * 4
    scaler = MinMaxScaler(feature_range=(0, 1))
    scaled = scaler.fit_transform(dataset)
    a = np.zeros([scaled.shape[0]-time_step + 1, time_step ,4])
    for i in range(scaled.shape[0]-time_step+1):
        for j in range(time_step):
            a[i][j] = scaled[i + j]
    y = np.zeros(scaled.shape[0])
    for i in range(scaled.shape[0]):
        y[i] = scaled[i][0]
    size = round(a.shape[0]*0.7)

    #split train and test sets
    X_train = a[:size]
    X_test = a[size:-1]
    y_train = y[time_step:size+time_step]
    y_test = y[size+time_step:]

    return X_train, y_train, X_test, y_test
```

Figure 3.1. The code for data processing

The second part is model building. This function needs the number of layers, the number of neurons, batch size, epoch and other neural network parameters as inputs, and then return the LSTM model designed according to the parameters and give its loss plot for checking over-fitting or under-fitting.

```python
def myLSTM(X_train, y_train, layerNum, neuralNum, batchSize, epochs):
    """
    A well-designed method for users to create a LSTM model espicially for
    Route 66 UQ Lake Station Passenger Flow Prediction.

    @param: X_train, training features
    @param: y_train, training labels
    @param: layerNum, the number of LSTM layers. Recommended between 1 and 3.
    @param: neuralNum, the number of neurals for every LSTM layer. Recommended between 0 and 500.
    @param: batchSize, the number of batchSize
    @param: epochs, the number of epochs
    @return: model, An LSTM model specified based on your input
    """
    # design network
    model = Sequential()
    model.add(LSTM(neuralNum, activation='relu',return_sequences=True,
                   input_shape=([X_train.shape[1], X_train.shape[2]])))
    for i in range(layerNum-1):
        model.add(LSTM(neuralNum, return_sequences=False))
    model.add(Dense(1))
    model.compile(loss='mse', optimizer='adam')

    earlystop_callback = EarlyStopping(monitor='val_loss', min_delta=0.0001, patience=20)

    history = model.fit(X_train, y_train, epochs=epochs, batch_size=batchSize, validation_data=(X_test, y_test),
                        callbacks=[earlystop_callback],verbose=2, shuffle=False)

    plt.plot(history.history['loss'], label='train')
    plt.plot(history.history['val_loss'], label='test')
    plt.legend()
    plt.show()

    return model
```

Figure 3.2. The code for model building

The third part is the evaluation model. In the first part, in order to train LSTM models, data is normalized. However, when evaluating the model, data should be converted back to its original appearance to get the true RMSE value. At the same time, this function will also return a visualization of the true and predicted values.

```
def evaluateMyLSTM(model, X_test, y_test, dataset):
    """
    A method for users to evaluate their LSTM model by myLSTM().
    Showing a plot of prediction by model and real values for visualization.

    @param model, the model returned by myLSTM().
    @param X_test, testing features.
    @param y_test, testing labels.
    @param dataset, the dataset to tell scaler how to restore data
    @return rmse, the real Root-mean-square deviation value of data which has been restored.
    """

    scaler = MinMaxScaler(feature_range=(0, 1))
    scaled = scaler.fit_transform(dataset)

    # make a prediction
    yhat = model.predict(X_test)
    inv_yhat = np.concatenate((yhat, np.zeros([y_test.shape[0], 3])), axis=1)
    y_true = np.concatenate((y_test.reshape([y_test.shape[0],1]), np.zeros([y_test.shape[0], 3])), axis=1)
    y_true = scaler.inverse_transform(y_true)[:,0]
    y_pre =  scaler.inverse_transform(inv_yhat)[:,0]

    #plot
    fig = plt.figure(figsize=(20, 5))
    ax = fig.add_subplot(111)
    ax.plot(y_true, label='true')
    ax.plot(y_pre, label='pred')
    ax.legend()

    rmse = math.sqrt(mean_squared_error(y_true, y_pre))
    print('Test RMSE: %.3f' % rmse)

    return rmse
```

Figure 3.3. The code for model evaluation

Repeatedly run these three parts of the code, we will get the optimal LSTM model.

3.2 Website Design

After training the model, it is also very important to deploy the model on an interactive page for users. Therefore, this report has designed a web page based on flask framework. The following 2 screenshots show the main functions of this website.

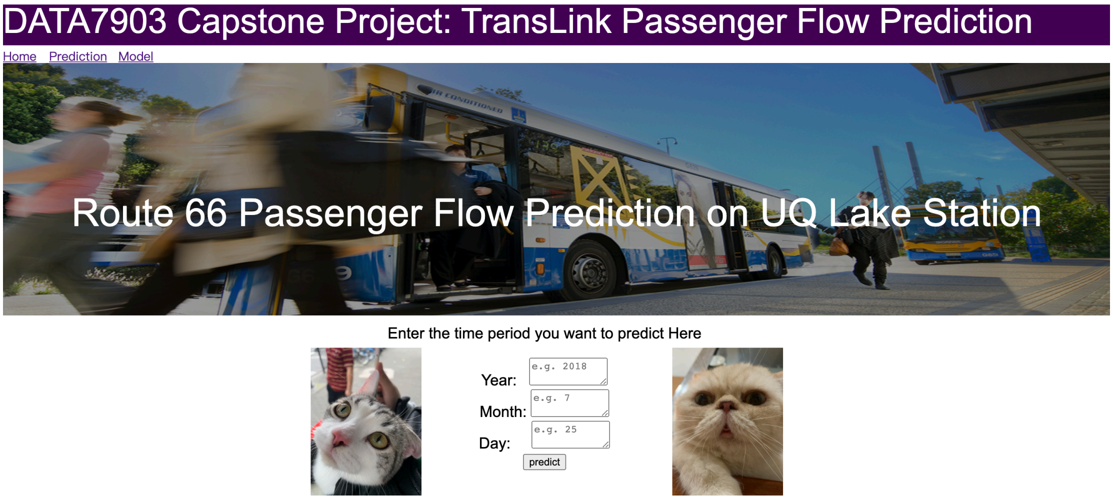


Figure 3.4. The prediction webpage

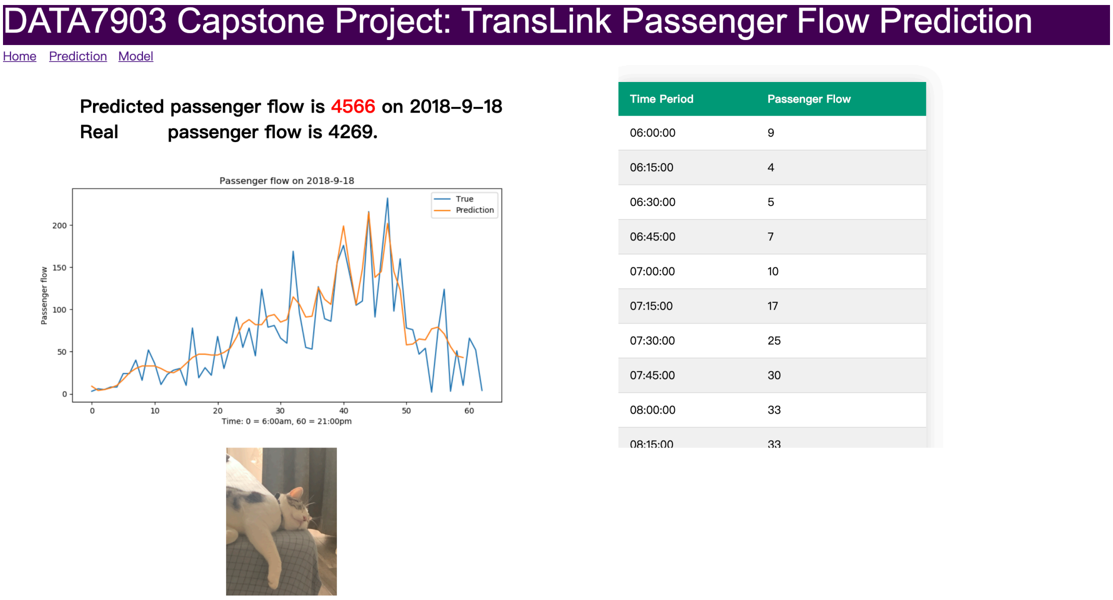


Figure 3.5. The result webpage

The operation of the web page is very simple. As shown in the Figure 3.4, users only need to enter the prediction date. The console will automatically match whether the date is a during the semester or holiday, weekday or weekend. Then it will choose the corresponding model for prediction. The user will get the predicted passenger flow every 15 minutes of the day, the total passenger flow throughout the day and a line chart. If the predicted date is the date in the data set, the web page will also provide ground truth data to help users check the reliability of the model as shown in Figure 3.5.

4. Conclusion

Expensive human resources make it difficult to dispatch additional buses unconditionally. The problem of insufficient bus demand during peak hours has also existed for a long time. Hence, it is very meaningful to build a model that can predict passenger flow over a period of time. This project successfully built an accurate prediction model for the passenger flow of Route 66 at UQ lake station every 15 minutes, which are the busiest route and station respectively. Moreover, this model only needs to adjust the data set to easily realize passenger flow prediction on other routes and stations.

In addition, this project also created an interactive website for users to analyze. The website can enable users who do not have programming skills to perform passenger flow prediction and analysis. This project also divides the model into three steps: data processing; building the model; evaluating the model. These steps are encapsulated in the function to make users train model more efficiently.

To summary, this project aims to solve the problem of supply and demand balance of Brisbane buses through passenger flow prediction models.

References

Lucy Stone. (2019). Brisbane's busiest buses revealed. Retrieved from https://www.brisbanetimes.com.au/national/queensland/brisbane-s-busiest-buses-revealed-and-how-many-people-were-left-behind-20190923-p52u38.html

Gurvinder Singh. (2019). Comprehensive Bus Boarding Analysis. Retrieved from https://www.kaggle.com/rednivrug/comprehensive-bus-boarding-analysis/output

Arthur Suilin. (2018). Web Traffic Time Series Forecasting. Retrieved from https://www.kaggle.com/c/web-traffic-time-series-forecasting/discussion/43795

Evan. (2018). Fully understand RNN and its different architectures. Retrieved from https://zhuanlan.zhihu.com/p/34152808

Chen, Q., Li, W., & Zhao, J. (2011). The use of LS-SVM for short-term passenger flow. prediction. Transport, 26(1), 5-10.

Lv, Z., Xu, J., Zheng, K., Yin, H., Zhao, P., & Zhou, X. (2018, July). Lc-rnn: A deep learning model for traffic speed prediction. In IJCAI (pp. 3470-3476).

Hou, L. M., & Ma, G. F. (2011). Forecast of railway passenger traffic based on a grey linear regression combined model. Computer Simulation, 7.

Cai, C., Yao, E., Wang, M., & Zhang, Y. (2014). Prediction of urban railway station's entrance and exit passenger flow based on multiply ARIMA model. Journal of Beijing Jiaotong University, 38(2), 135-140.

Wang, Y., Yin, H., Chen, H., Wo, T., Xu, J., & Zheng, K. (2019, July). Origin-destination matrix prediction via graph convolution: a new perspective of passenger demand modeling. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (pp. 1227-1235).

Sun, Y., Leng, B., & Guan, W. (2015). A novel wavelet-SVM short-time passenger flow prediction in Beijing subway system. Neurocomputing, 166, 109-121.

Liu, L., & Chen, R. C. (2017). A novel passenger flow prediction model using deep learning methods. Transportation Research Part C: Emerging Technologies, 84, 74-91.

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. Journal of. machine learning research, 3(Mar), 1157-1182.

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*.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.