Prediction of urban business district based on trajectory time series feature recognition

Qingyang LI; Jiahao Chen

Summary and Abstract

Urban business district has the functions of work, consumption, leisure and entertainment, and plays an important role in the daily life of urban residents. In recent years, with the continuous progress of urbanization, traffic congestion has occurred in different degrees in big cities, especially in the surrounding areas of business districts. If we can learn and model the change law of urban business district, so as to accurately predict the change of business district boundary, then we can design a better algorithm to avoid congestion based on this, and guide vehicles to avoid the congested business district area. Based on the track data of taxis and online car hailing, this paper establishes a mathematical model to discover and predict the change of business circle boundary in a day. This paper takes Chengdu City of China as the research object, takes the time series of Chengdu Regional Grid traffic matrix as the input, uses the convolution long-term and short-term memory network for modeling and training, and realizes the time series prediction of business circle boundary. The accuracy rate, recall rate and F value are used as evaluation indexes, and the benchmark methods such as SVM and GBDT are used for comparative experiments.

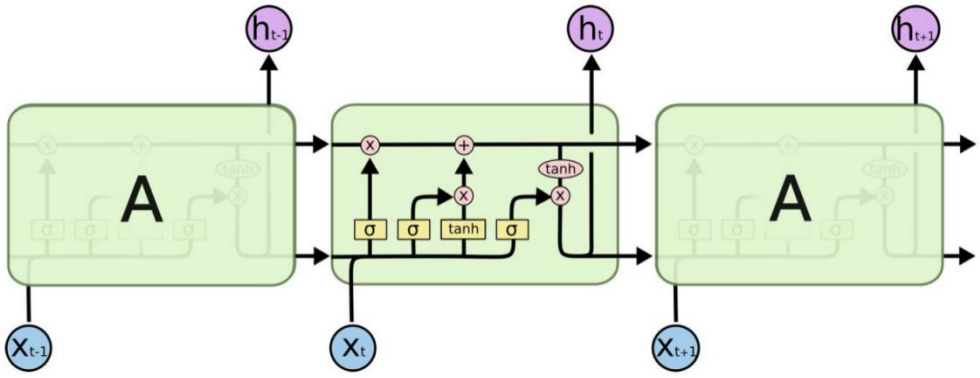
Approach

In this project, we are going to use Convolution Long Short-Term Memory (Convolution LSTM) to be our neural network, and we will use some time series data feature recognition methods.

For the data pre-processing, we divided the map of Chengdu into grids and extract pick-up and drop-off locations and times of taxicab trajectories. Then we generate the number of accesses of each grid in different timeslots.

Convolution Long-Short Term Memory model (LSTM) was proposed in 2015. It is a network structure that introduces convolution operation commonly used in feature extraction of plane image into LSTM network to make it suitable for feature extraction of image sequence (such as time series of urban precipitation distribution image). Compared with the traditional LSTM network, a significant advantage of convolution LSTM is that it extracts the spatial information from the image data by convolution operation, rather than treating each point as an isolated data.

The mathematical principles of LSTM and revolution LSTM are briefly described below.



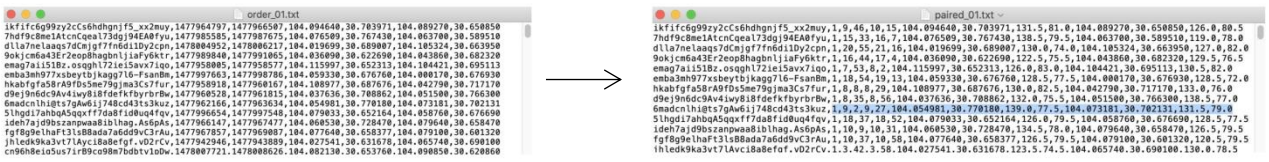
$$i_t = \sigma(W_i \cdot [h_{t-1}; x_t] + b_i)$$
$$f_t = \sigma(W_f \cdot [h_{t-1}; x_t] + b_f)$$
$$o_t = \sigma(W_o \cdot [h_{t-1}; x_t] + b_o)$$

The formula and picture above show the mathematical principle of gate switch, the core mechanism of LSTM network. Each network layer contains three gate switches, namely forget gate f, input gate i and output gate o. Inside each "gate" is the vector [Ht-1; Xt] is multiplied by the weight matrix W, and then goes through a sigmoid layer (with σ Represents). The convolution LSTM replaces the multiplication operation with two-dimensional convolution operation to extract spatial features.

Implementation detail

The very first thing we need to do is extracting and preprocessing the original data. We get taxi data from DIDI, and Chengdu's POI information from Qingyang's University senior. These data we can find in the data2 direction. The original data contains: ID of taxis, the time for passengers to get on and off, and latitude and longitude coordinates of each order. We translate the time data into (day,hour,min) form, and translate the latitude and longitude coordinates into 2-d coordinates.

We associate the location with the nearest function area (from POI), and then we can build a heat map recording how many times of one area become a get on or get off place. We save this information as numpy matrix by generating npy files.



We can see from the two pictures above, before processing, the data contain: id, get on/off time, get on/off latitude and longitude. After processing, the data contains: id, day, get on hour and minute, get off hour and minute, get on/off latitude and longitude and get on/off XY coordinate. For example, a record of a taxi trip would be:

‘ikfifc6g99zy2cCs6hdhgnjf5_xx2muy,1477964797,1477966507,104.094640,30.703971,104.089270,30.650850’

The first column is the ID of the taxicab, which is not useful for this project and can simply be ignored. The following two columns are the Unix timestamps for “get on” and “get off”. The last four columns consists of floating point numbers, which are the latitude of get on, longitude of get on, latitude of get off, longitude of get off, respectively.

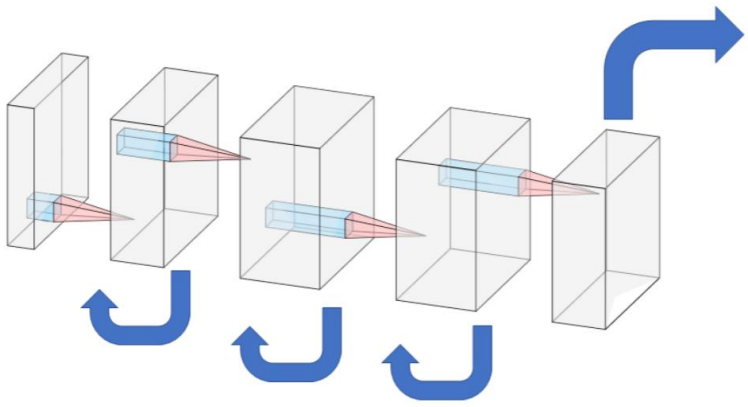
After our processing, this line of data will become:

”ikfifc6g99zy2cCs6hdhgnjf5_xx2muy,1,1,46,2,15,104.094640,30.703971,131.5,81.0,104.089270,30.650850,126.0,80.5”

We can see that these data are based on the original data, but there are some more numbers. “1,1,46,2,15” represents that this trip is happened in 1:46 of day 1, and the passenger get off at 2:15 of the same day. We assume that all of these instances happen in the same day. The number “131.5,81.0” is the 2-d coordinate we transferred from the latitude and longitude data. Then we still need to standardize the 2-d coordinate and fit them into a 40*40*3 numpy matrix.

The next part is how we build the model. As we mentioned before, this model is a convolution LSTM model.

In this project, we divided the day into 144 copies, each of which collected 10 minutes of boarding and alighting data. The input of neural network should be 144 elements time series (s [0], s [1], ..., s [n - 1]). Each element s [i] of the sequence is a three-dimensional vector, whose shape is 40 * 40 * 2. The two feature vector elements are the out degree and in degree of the mesh. The output of neural network is also a time series with 144 elements (t [0], t [1], ..., t [n]). Each element of the sequence t [i] is a three-dimensional tensor. Each data element of the output tensor represents the probability that the grid belongs to the category. In this problem, the grid is divided into four categories: non business district, low heat business district, medium heat business district and high heat business district, so the value is 4. The output classification results will be compared with the classification supervision information.



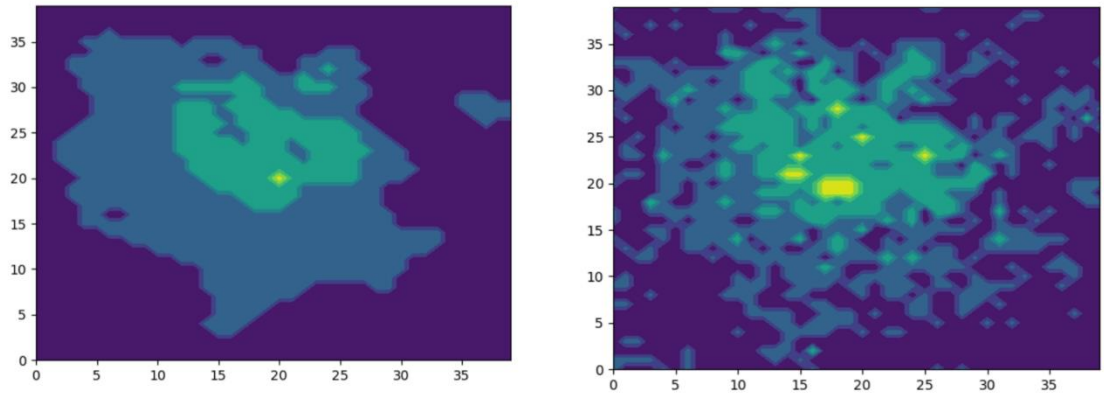
The picture above shows the detail of the network of Conv LSTM, the leftmost layer is the input layer, and the rightmost layer is the output layer. It contains four middle hidden layers, and the number of output channels is 4, 8, 8 and 4 respectively. When a group of data (an access graph in a time series) propagates forward, the network iterates 9 times, updating the cell state and hidden state of each hidden layer each time to participate in the next iteration. After the last iteration, the network takes the hidden state of layer 4 as the output data.

In this project, we use a Macbook and a virtual machine. We use the Google Cloud Platform.

We wrote our code in jupyter notebook with python 3.7.

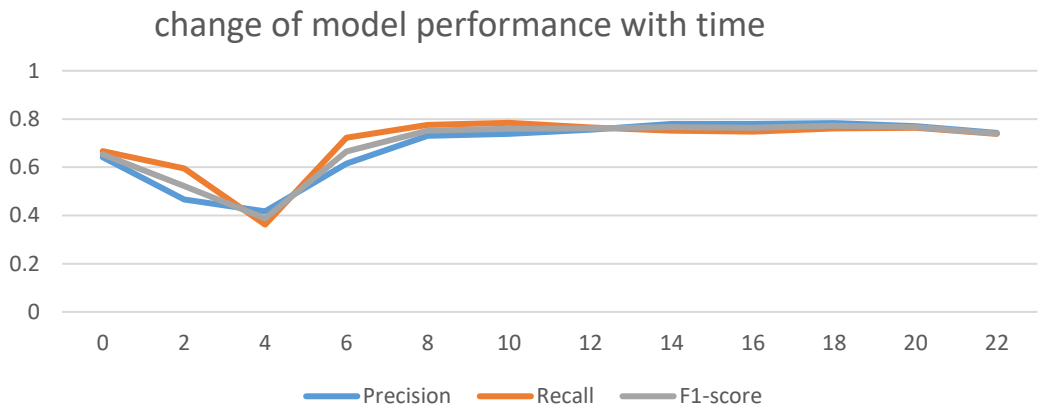
Prediction and Comparison

We labeled each point in the heat map from 0 to 4, and draw the heat map like below:



The left figure is the result of our prediction, and the right one is the ground truth. We can see that our model is kind of underfitting, because the prediction missed some details, but we can see the outline of the business district from the predicted picture. We think we can complex our model like deepen the network, add more parameters and get more data from taxicabs to fit the model.

We compare the precision, recall and f1 score in this model, using different days of taxicab data and the outcome is below:



We can see if we use few days like 3 or 4, the performance is clearly bad. Thus we finally choose 20 as the parameter of input. Moreover, we compared our model with another four classifications, and the following form shows the result:

	Precision	Recall	F1-score
Convolution LSTM	0.750494	**0.683332**	0.715339
kNN	0.759345	0.547244	0.636079
DT	0.780998	0.670517	**0.721553**
SVM	0.768431	0.679399	0.721178
GBDT	**0.788682**	0.646284	0.710417

Actually, each model has its own advantage and disadvantage, but our model has the best performance overall.

Conclusion

In conclusion, this project mainly completes the following work:

(1) In the aspect of data preprocessing, we select the area of 20km * 20km around the center of Chengdu City for research. We collected the trajectory data of taxicabs in Chengdu for about 20 days in November 2016, and extracted the starting and ending points, boarding and alighting time and other information. In space, we build a 40 * 40 city traffic matrix by corresponding the geographic coordinates of the start and end points to the map grid. In terms of time, we divide the daily visit matrix into 144 element time series according to the slot of 10 minutes as the input data of the model.

(2) In the aspect of model construction, we build a model based on Convolution LSTM to discover and predict the boundary of urban business district. The model combines the advantages of LSTM network in recognizing temporal features and convolution neural network in recognizing spatial features, and can efficiently extract the spatio-temporal features of input data.

(3) In the aspect of comparative verification, we take the accuracy rate, recall rate and F value as the indicators to measure the performance of the model. We select k-nearest neighbor, general decision tree, support vector machine and GBDT as benchmark methods. Through cross validation experiments and experiments with different data sets, we compare the experimental results and test the prediction performance of the model.

Reference

[1] Shi X, Chen Z, Wang H, et al. Convolutional LSTM Network: a machine learning approach for precipitation nowcasting[C]. neural information processing systems, 2015: 802-810.

[2] Zhang J, Zheng Y, Qi D, et al. Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction[C]. national conference on artificial intelligence, 2016: 1655-1661.

[3] Zheng Y, Li Q, Chen Y, et al. Understanding mobility based on GPS data[C]. ubiquitous computing, 2008: 312-321.

[4] Kwella B, Lehmann H. Floating car data analysis of urban road networks[C]. International Conference on Computer Aided Systems Theory. Springer, Berlin, Heidelberg, 1999: 357-367.

[5] Yuan J, Zheng Y, Xie X, et al. Discovering regions of different functions in a city using human mobility and POIs[C]. knowledge discovery and data mining, 2012: 186-194.

Acknowledgements

The implementation of ConvLSTM in PyTorch was found in this Github repo:

https://github.com/automan000/Convolutional_LSTM_PyTorch

The 20-day trajectory dataset was provided by DIDI (the Chinese version of Uber). The POI data of Chengdu was provided by Zhuoran Li, Qingyang's university senior.