Traffic Network Flow Estimation Based On Social Network Influence Model

Midterm Report

11812804 董　正
11810419 王焕辰
11811305 崔俞崧

Supervisior: 宋轩

Department of Computer Science and Engineering

Dec. 2021

Contents

1 Preliminaries

1.1 Review

In this semester, we will try to build a traffic flow estimation system based on graph neural network and social network influence model. We have changed the system design a little. Currently, the structure of the whole system is

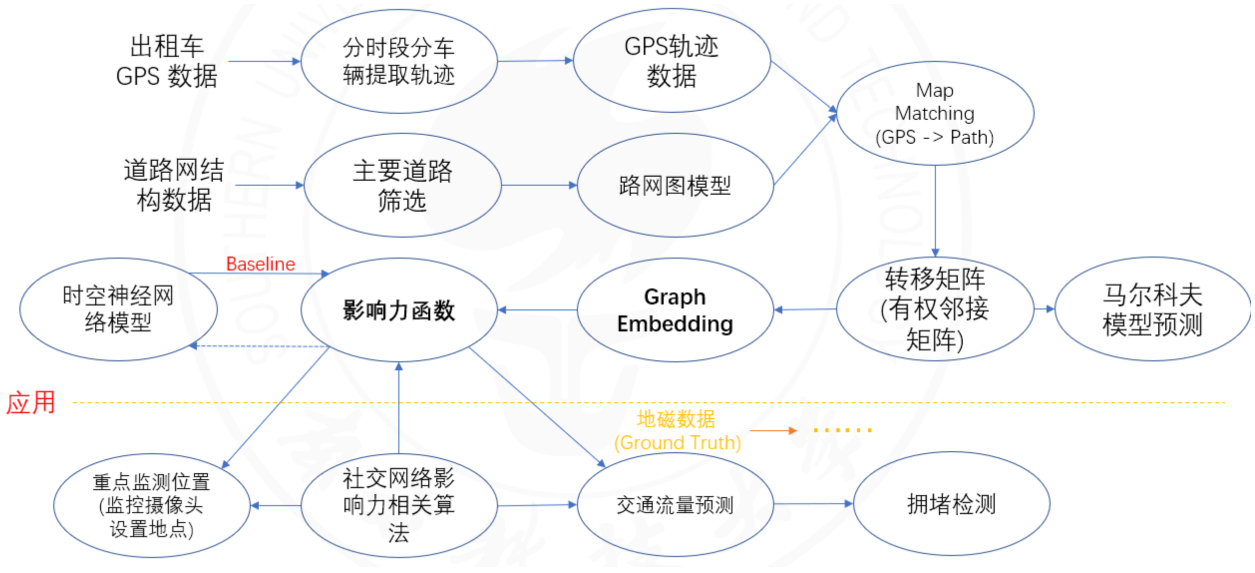


Figure 1: System Structure

1. Process taxi GPS data to get tracks
2. Process road network data to get a basic graph model
3. Match tracks to each road and get the adjacent matrix of the graph
4. Try a simple prediction based on Markov model
5. **Graph embedding**
6. **Influence function design**
7. Combine spatial-temporal models and use them as baseline
8. Combine social network influence algorithms to predict traffic network flow, use geomagnetic data as one of the ground truth
9. Applications: traffic surveillance camera position and traffic jam detection

1.2 Report Contents

Breifly, we will state our work in this report as

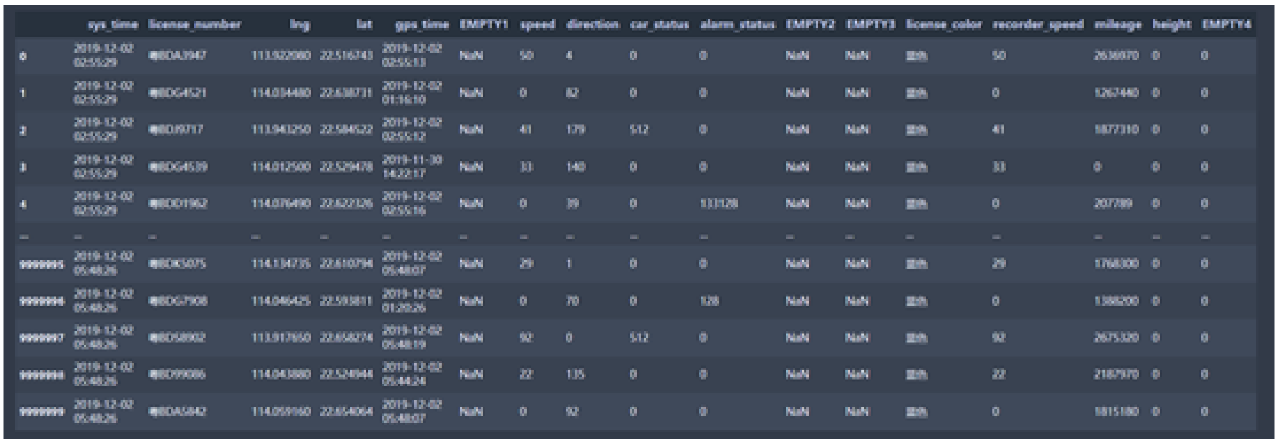
- Transition Matrix and Graph Embedding Analysis by 董正 & 崔俞崧

- Geomagnetic Data Mining 王焕辰
- Future Plan by 董正

2 Transition Matrix and Graph Embedding Analysis

2.1 Dataset

- Data source: Shenzhen Municipal Government
- Region: Shenzhen
- Time: 2019-12-01 to 2019-12-13
- Content: Taxi vehicle trajectory data
  - License number
  - Longitude and latitude
  - Speed
  - License type



	sys_time	license_number	lng	lat	gps_time	EMPTY1	speed	direction	car_status	alarm_status	EMPTY2	EMPTY3	license_color	recorder_speed	mileage	height	EMPTY4
0	2019-12-02 02:55:29	粤BDA3947	113.922080	22.516743	2019-12-02 02:55:13	NaN	50	4	0	0	NaN	NaN	[illegible]	50	2636970	0	0
1	2019-12-02 02:55:29	粤BDG4521	114.034480	22.638731	2019-12-02 01:16:10	NaN	0	82	0	0	NaN	NaN	[illegible]	0	1267440	0	0
2	2019-12-02 02:55:29	粤BDJ9717	113.943250	22.584522	2019-12-02 02:55:12	NaN	41	179	512	0	NaN	NaN	[illegible]	41	1877310	0	0
3	2019-12-02 02:55:29	粤BDG4539	114.012500	22.529478	2019-11-30 14:22:17	NaN	33	140	0	0	NaN	NaN	[illegible]	33	0	0	0
4	2019-12-02 02:55:29	粤BDD1962	114.076490	22.622326	2019-12-02 02:55:16	NaN	0	39	0	133128	NaN	NaN	[illegible]	0	207789	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9999995	2019-12-02 05:48:26	粤BDK5075	114.134735	22.610794	2019-12-02 05:48:07	NaN	29	1	0	0	NaN	NaN	[illegible]	29	1768300	0	0
9999996	2019-12-02 05:48:26	粤BDG7908	114.046425	22.593811	2019-12-02 01:20:26	NaN	0	70	0	128	NaN	NaN	[illegible]	0	1388200	0	0
9999997	2019-12-02 05:48:26	粤BD58902	113.917650	22.658274	2019-12-02 05:48:19	NaN	92	0	512	0	NaN	NaN	[illegible]	92	2675320	0	0
9999998	2019-12-02 05:48:26	粤BD99086	114.043880	22.524944	2019-12-02 05:44:24	NaN	22	135	0	0	NaN	NaN	[illegible]	22	2187970	0	0
9999999	2019-12-02 05:48:26	粤BDA5842	114.059160	22.654064	2019-12-02 05:48:07	NaN	0	92	0	0	NaN	NaN	[illegible]	0	1815180	0	0

Figure 2: Dataset

2.2 Transition Matrix

In the previous period, we did map matching to get taxi trajectory data and a transition matrix.

- Size: 16153×16153
- Represents the transition probability between adjacent roads
- We splited different time intervals in every 3 hours for accuracy
- The data we use is from 12.02(Mon.) to 12.07(Sat.), and use 12.08(Sun.) as a test dataset.

Next, we gave an flow analysis on this transition matrix:

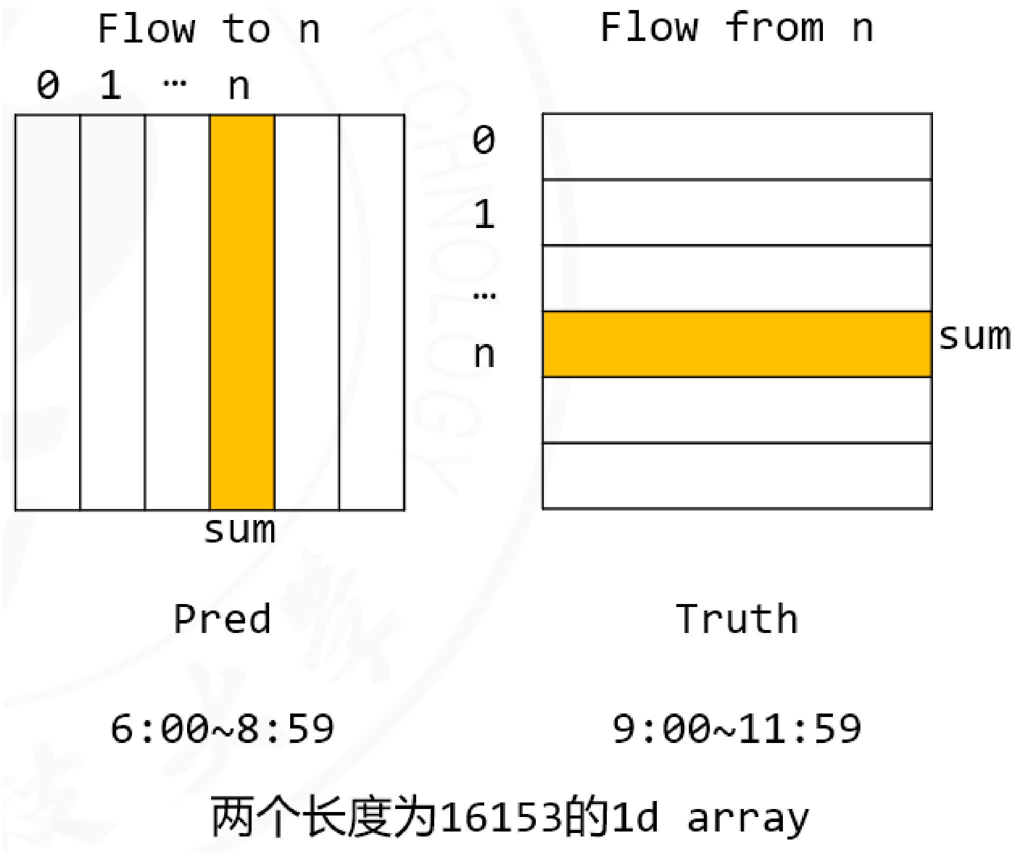


Figure 3: Matrix Verification

Here we choose the 6:00 ~8:59 transition matrix of first 6 days and 9:00 ~11:59 transition matrix of Sunday. In the left matrix, the sum of column `n` means how many cars turned to $road_n$ (i.e. traffic flow into $road_n$) in this time interval. And in the right matrix, the sum of row `n` means how many cars turned to other roads from $road_n$ (i.e. traffic flow from $road_n$). Therefore, in the ideal situation, these two sum should be fairly close.

$$\text{Flow to } road_n \text{ in the } (t-1)^{th} \text{ time interval} \approx \text{Flow from } road_n \text{ in the } t^{th} \text{ time interval}$$

Expanding it to the whole matrix, we can get an array of sum. Also compute it on test matrix. And we start analysis on these two sum.

After sum on rows and sum on columns, we will get two 1D arrays whose length are 16153. Then we first calculate their absolute values of difference directly. Remember to divide 6 on the left matrix.

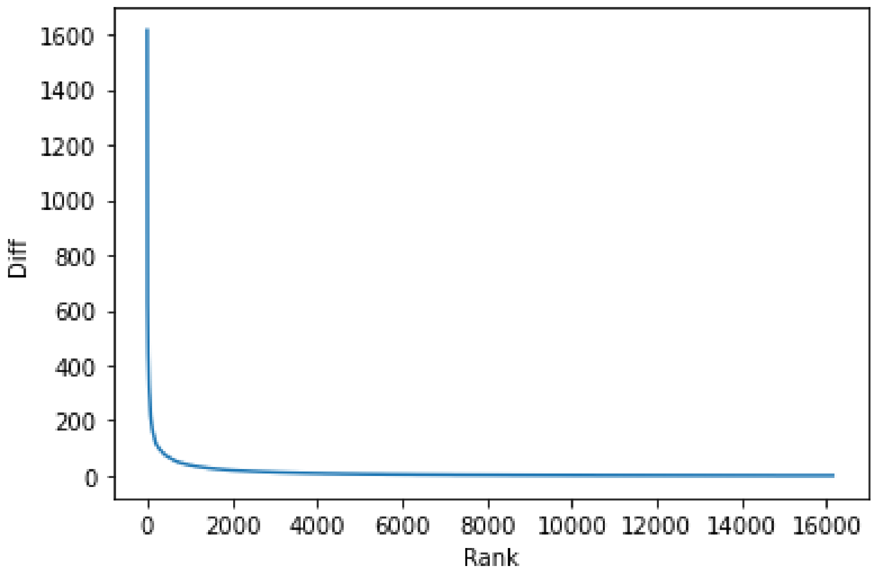


Figure 4: Difference of two sum arrays

From the chart above we can see that some of the roads have a very large difference. Digging into it a bit we found that it is because we counted self to self (diagonal) in the matrix. After cleared the diagonal, the differences are:

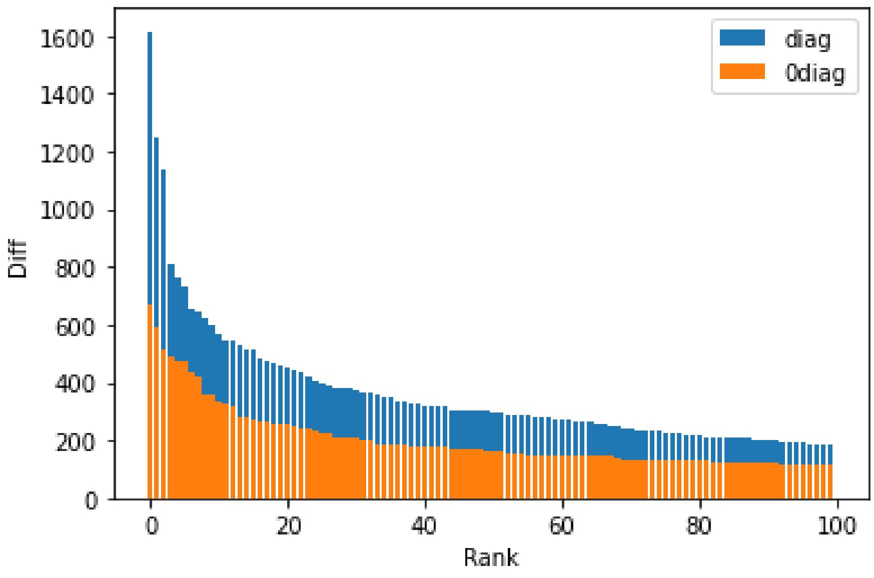


Figure 5: Difference of two sum arrays after cleared diagonal

Still, there are some roads have a large difference. We speculate that this is because we used Sunday as test dataset. Traffic on holidays has a very different pattern compared to weekdays. Therefore, in the next period, we plan to try to use Monday to Thursday as train set, and Friday as test set.

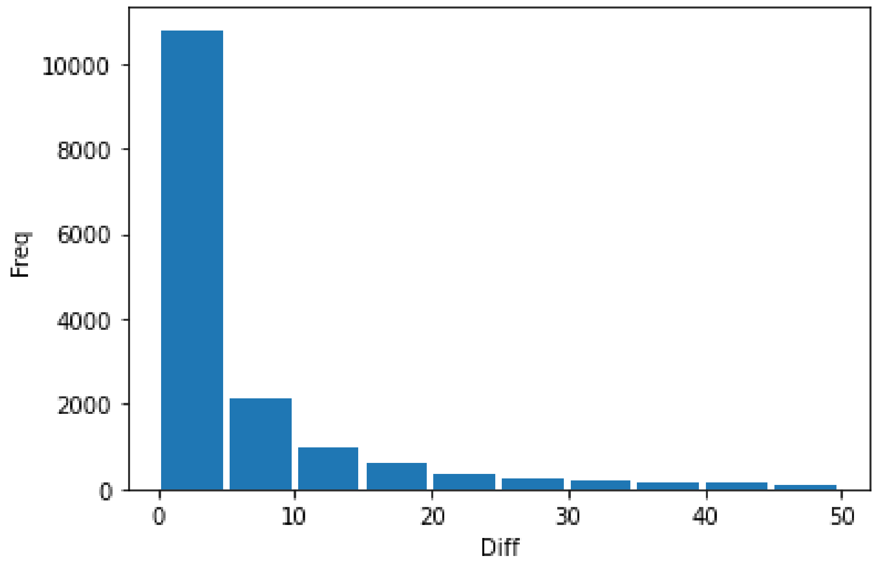
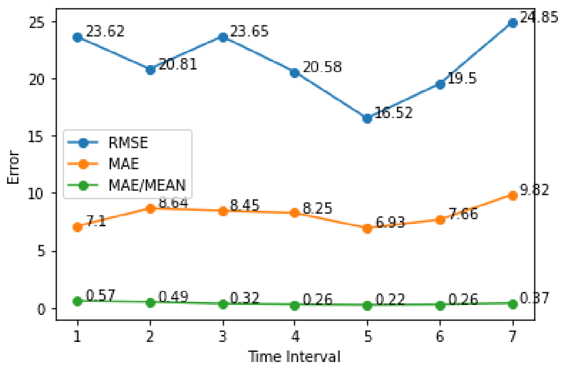


Figure 6: Distribution of difference

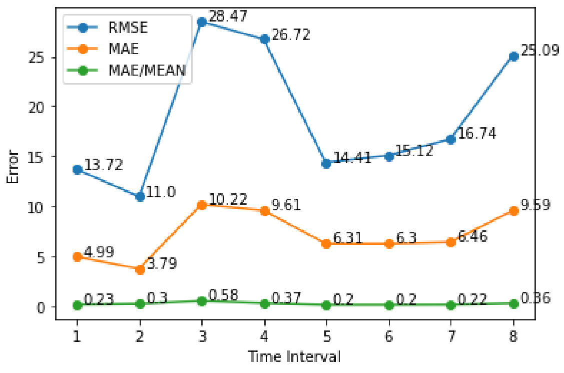
The figure above shows the distribution of differences. We can see that 97% of roads is less than 50, and 80% of roads is less than 10, which indicates the accuracy of transition matrix.

However, counting differences is not enough. Then we use some metrics to test the error on the whole matrix. Here we used RMSE, MAE, and MAE/Mean of test matrix. This is because the two transition matrices is very sparse, i.e. most of them are zeros. Therefore, MAPE cannot be applied because it will divide zero.

We used a controlled experiment. The comparing group calculated errors on same time interval, and both sum on rows. The results are shown as following:



(a) Errors



(b) Errors when comparing same time interval

From the figures above, we can indicate that it is better to use the former time interval to predict the next interval on rush hours. However, the performance on night hours is worse than direct prediction. This is because traffic flow is not stable on night hours. It will increase at the beginning of the day, and decrease at the end of the day. Direct prediction has a very high error on daytime, which is not acceptable.

2.3 Graph Embedding

The most common way to embed a graph is *DeepWalk* with *Word2Vec*.

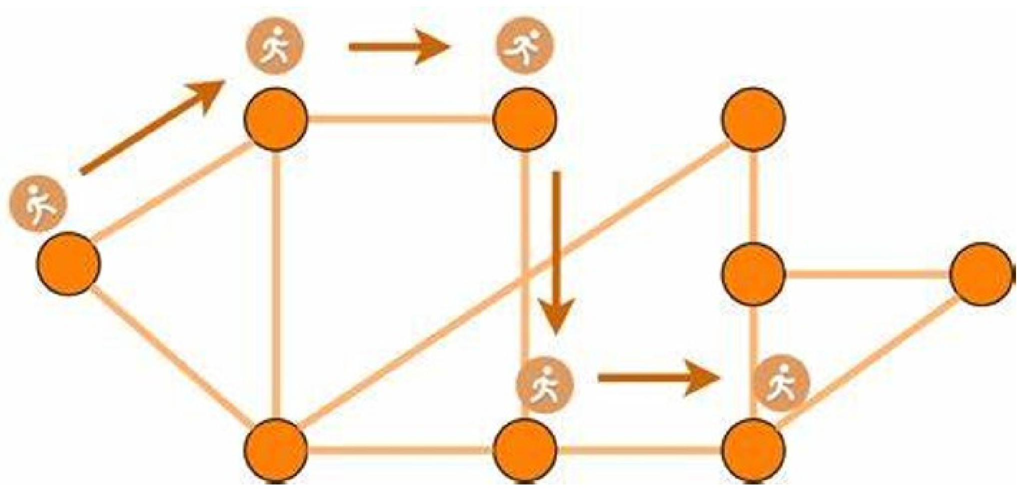


Figure 7: DeepWalk

In *DeepWalk* algorithm, we will generate fixed length random path based on the graph model. Then treat these paths as sentences, use *Word2Vec* technology from NLP to transfrom them to vectors.

However, since map matching gave us a series of taxi trajectories, we can directly apply them to *Word2Vec*. The advantages are:

- Trajectories keeped road connection information.
- Trajectories can indicated the transition probability between roads.
- No need to set walk length.
- Save time, because random walk generation take a lot.

What's more, compared to transition matrix, *Word2Vec* can

- Represent the transition probability (called similarity) between any two roads.
- Have high-dimensional transition information. Because just a 3D transition matrix will be too big to store.
- Can deduce influence between roads (next step of our experiment).

The figure shows the embedding result on a small dataset, reduced dimension by TSNE algorithm.

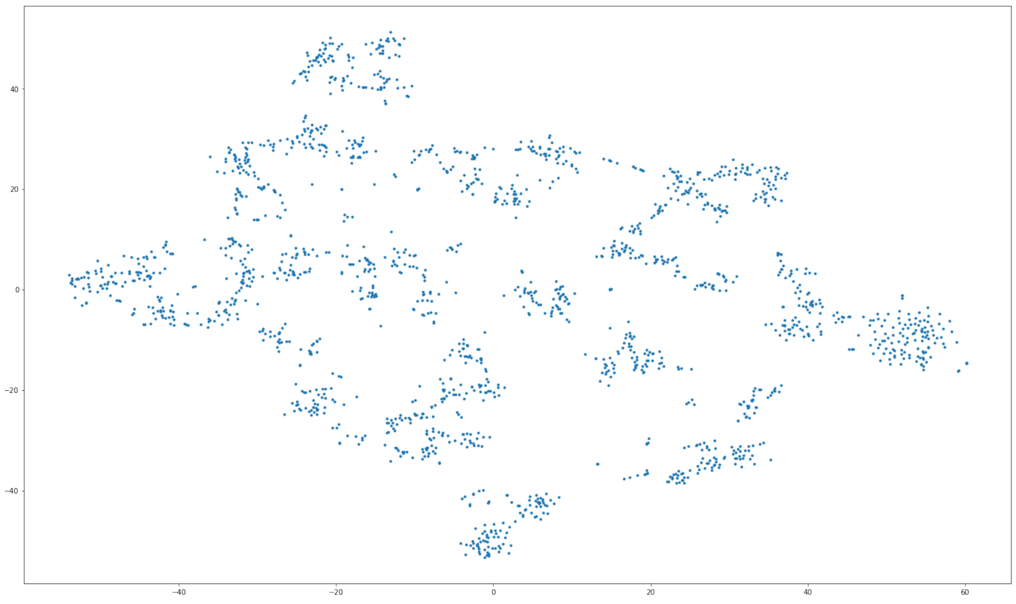
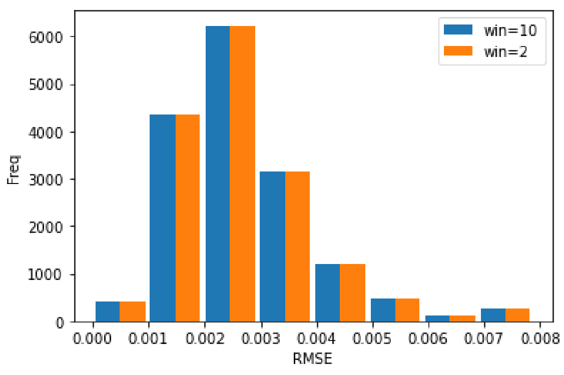
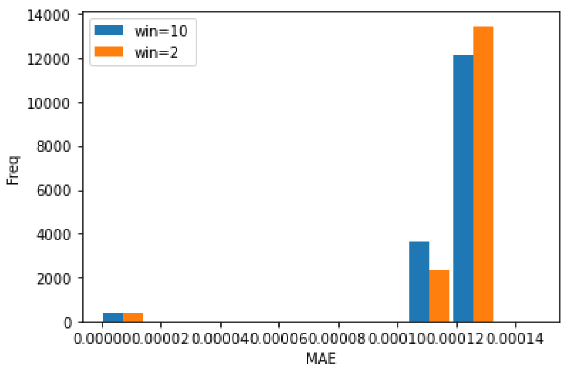


Figure 8: Graph Embedding Result After TSNE

The distance between vectors shows the transition probability between corresponding roads. From the picture we can tell the road connectivity, since there are obviously many different clusters.



(a) Errors



(b) Errors when comparing same time interval

Then we compare the difference between embedded vectors and transition matrix. The chart above calculated their RMSE and MAE. First, for each road, devide every element (in transition matrix, it is transition count, and in embedded vectors, they are similarity) by their sum to get a from of probability. We can know that they have a very small error, which infers that *Word2Vec* keeped the transition probability information in transition matrix. In addition, it can also represent high-dimensional transition.

In *Word2Vec*, there is a parameter `window`, which means a word can influence how far in the sentence. To compare, we set this parameter to 2 (default 10). And the results shows that it had very little effect on embedding. What's more, it will affect high-dimensional transition, thus, we decided to use default parameters.

3 Geomagnetic Data Mining

3.1 Processing of Geomagnetic Detection Point Drift

Since observing the position of each detection point by previous results, all points are found to be offset in the same direction.

It is speculated that the base map does not match the coordinate data, so the *coord_convert* library is used to convert the coordinate data from BD-09 (Baidu coordinate system) to WGS84 (default coordinate system for the map).

After the transformation of coordinates, the coordinates are successfully matched to the corresponding road network on the map, which can be used for the subsequent construction of geomagnetic detection point graph network.

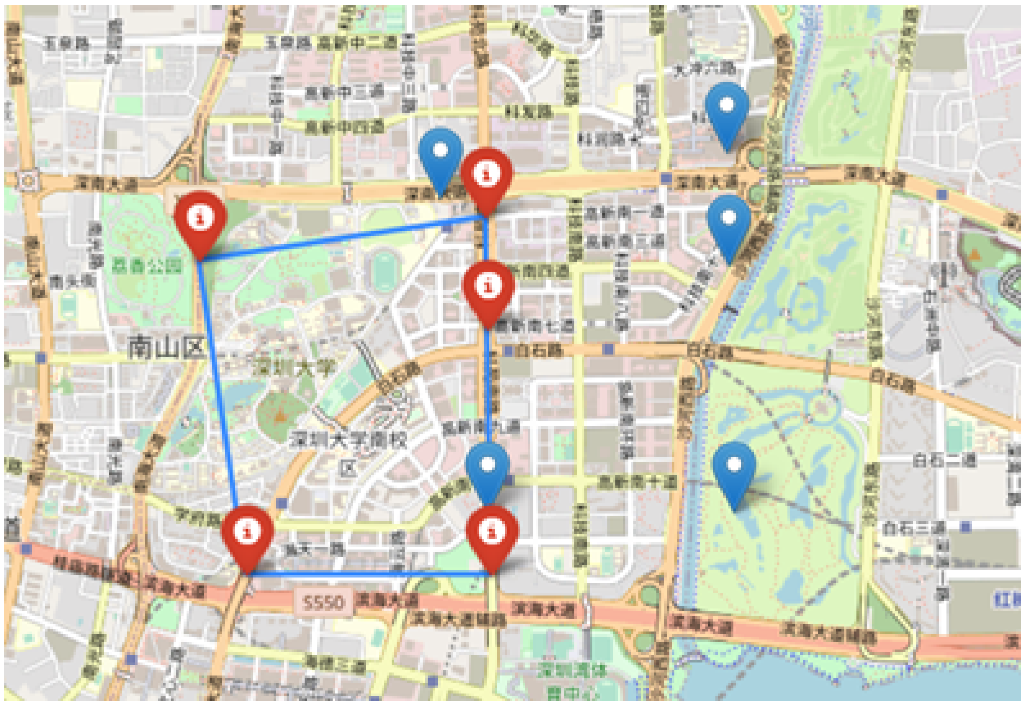


Figure 9: Coordinates on the Map before and after Transformation

3.2 Vehicle Type Classification

Using geomagnetic data to provide vehicle types and other features to classify vehicles, and then better used for traffic state, flow prediction

Due to the uneven distribution of the four types of vehicles, the down sampling process is conducted and the available features, such as road, speed and time, are extracted for preliminary classification

SVM and lightgbm were used for classification successively, and the results are as follows. However, the results of both are not ideal because of insufficient features provided by data

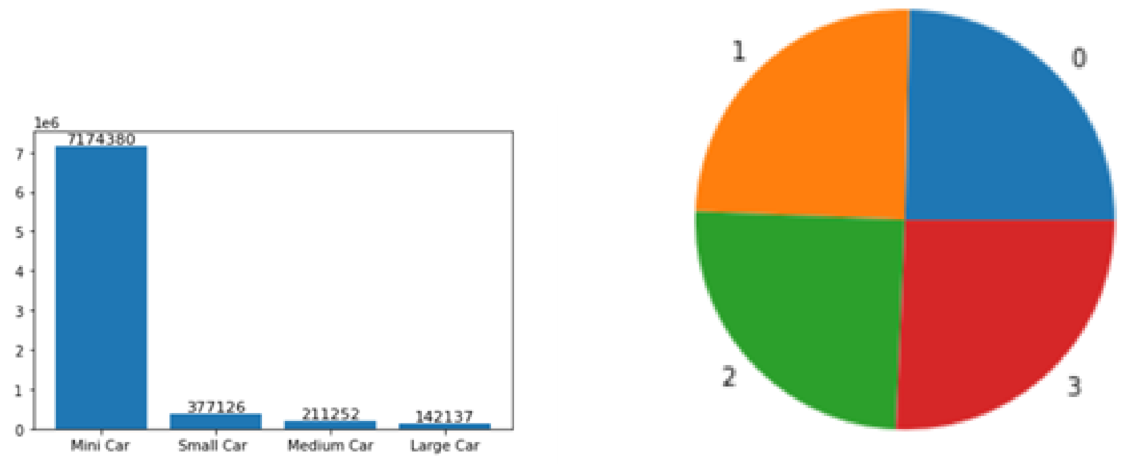


Figure 10: The Number of Various Models and the Proportion after Down Sampling

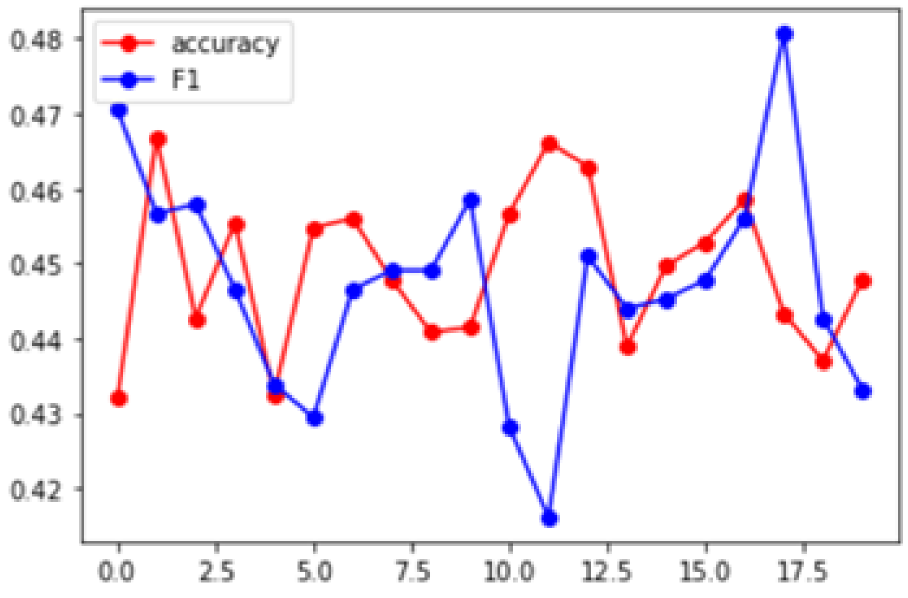


Figure 11: Several Scores of Training Results by lightgbm

3.3 Traffic Flow Data Further Processing

In view of the poor effect of vehicle classification and the limitations of geomagnetic data. Therefore, we divide each road vehicle into inflow and outflow according to existing literature and models.

Inspired by the literature, we're going to find out a standard to establish a real graph as far as possible for the unrealistically connected road geomagnetic detection points, and reduce the inflow to outflow error (dividing the map into several districts and construct each graph).

Training only through traffic flow data and graph building through each detection point. As well as predicting traffic flows, it can also serve as a benchmark to evaluate other presented models.

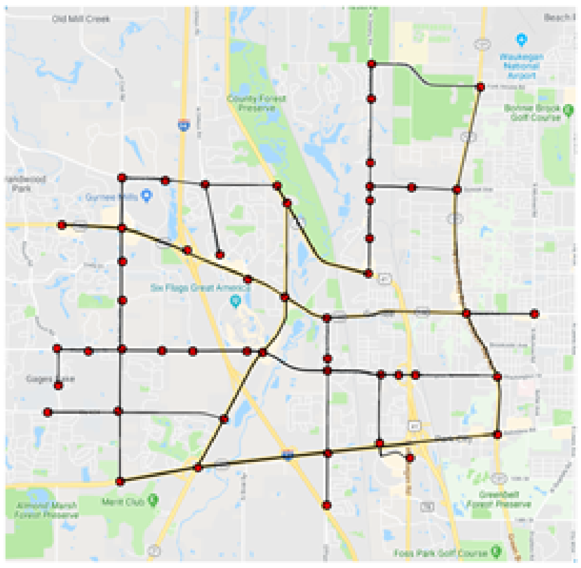
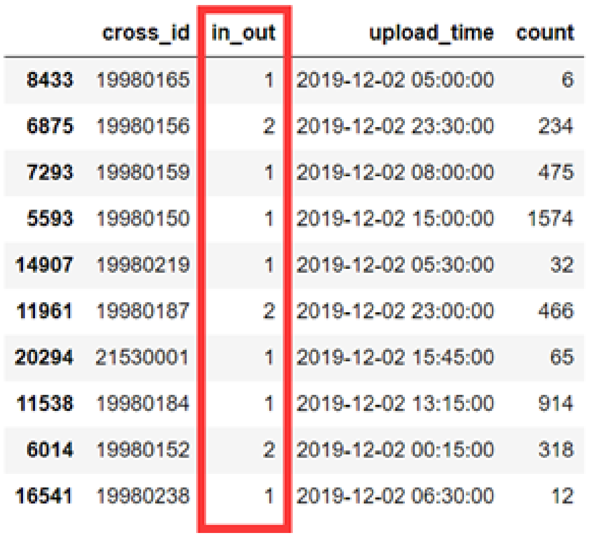


Figure 12: Graph of a Community Detection Point



	cross_id	in_out	upload_time	count
8433	19980165	1	2019-12-02 05:00:00	6
6875	19980156	2	2019-12-02 23:30:00	234
7293	19980159	1	2019-12-02 08:00:00	475
5593	19980150	1	2019-12-02 15:00:00	1574
14907	19980219	1	2019-12-02 05:30:00	32
11961	19980187	2	2019-12-02 23:00:00	466
20294	21530001	1	2019-12-02 15:45:00	65
11538	19980184	1	2019-12-02 13:15:00	914
6014	19980152	2	2019-12-02 00:15:00	318
16541	19980238	1	2019-12-02 06:30:00	12

Figure 13: Traffic Data after Grouping

4 Future Plan

In the next period, we will learn to use *LibCity* to run baseline algorithms on our model.

LibCity

Figure 14: LibCity

From LibCity's GitHub (https://github.com/LibCity/Bigscity-LibCity):

LibCity is a unified, comprehensive, and extensible library, which provides researchers with a credible experimental tool and a convenient development framework in the traffic prediction field. Our library is implemented based on PyTorch and includes all the necessary steps or components related to traffic prediction into a systematic pipeline, allowing researchers to conduct comprehensive experiments. Our library will contribute to the standardization and reproducibility in the field of traffic prediction.

LibCity currently supports the following tasks:

- Traffic State Prediction
- Traffic Flow Prediction
- Traffic Speed Prediction
- On-Demand Service Prediction
- Origin-destination Matrix Prediction
- Traffic Accidents Prediction
- Trajectory Next-Location Prediction
- Estimated Time of Arrival
- Map Matching
- Road Network Representation Learning

After that, we will start writing paper and also read papers about social network to design our influence function.