Traffic Prediction for New York City Based on Graph Neural Network

Haoran Zhao
Drexel University
Philadelphia, United States

Yi Pan*
Drexel University
Philadelphia, United States
yp369@drexel.edu

Yantian Ding
Drexel University
Philadelphia, United States

2023 IEEE International Conference on Image Processing and Computer Applications (ICIPCA) | 979-8-3503-1467-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICIPCA59209.2023.10257965

***Abstract*—The real-time speed of each roadway segment in a network is essential in many urban traffic environments, from transportation route planning in freight transportation to congestion avoidance in advanced traveler information systems. Therefore, accurate real-time average road speed prediction is of utmost importance. Vehicle speeds and travel times can vary considerably in urban environments due to traffic congestion caused by accidents or adverse weather conditions. At another level, one can also predict the future of each road segment by the speed variation of the surrounding road segments and the impact of the surrounding road segments on that road segment. This paper presents a project to predict the real-time average speed of selected road segments in New York City using graph neural networks. It focuses on predicting road segment speeds using Movement data from Uber vehicles and road network data collected over a considerable period as input. In this paper, different machine learning and data mining methods are applied, and computational results are reported on actual data to demonstrate the accuracy of the obtained predictions and the model's superiority.**

Keywords—Deep Learning, Prediction, Traffic Speed, Graph Neural Network, Time-series Data

I. Introduction

Travel speed prediction is a significant problem in most urban traffic models, regardless of whether they involve the transport of people or goods. Most travel speed variations and travel time variations are caused by traffic congestion, which can be classified as recurrent and non-recurrent, i.e., due to accidents, construction, emergencies, special events, and bad weather. Therefore, it is necessary to accurately predict the travel speed (time) under recurrent and non-recurrent congestion. Notably, better forecasting may greatly assist individuals and transportation companies operating in urban areas to plan their activities and may lead to significant reductions in actual travel times and greenhouse gas emissions.

The paper aims to predict travel speeds on significant roadways in New York City to avoid congestion through real-time prediction of speeds at each location. This paper constructs the potential possible driving speed predictions using speed and geolocation data collected from the Uber APP on the Uber Driver's phone. The data used in this study comes from Uber Technologies Inc, a company that develops mobile applications to connect riders and drivers in a sharing economy service that provides passenger vehicle rentals and matchmaking rideshare [1]. Uber provides access to their data on their platform Uber Movement, and their purpose is to "provide data and tools for cities to more deeply understand and address urban transportation challenges."[2]

Subsequent sections of the paper are arranged as follows. We first demonstrate the preprocessing part of raw data. Then, an exploratory analysis is performed to understand the data's insight. Subsequently, it describes the neural network model used to predict vehicle driving and reports the computational results in real-world situations. Finally, we compare our model with other solutions and analyzes the advantages and disadvantages of our model.

II. Theory Introduction

A. Graph Attention Networks

The core idea of GAT, i.e., Graph attention networks, is to introduce an attention mechanism into the graph convolutional network. Since the input of our project is graph-based traffic speed data, which is often a sequence of temporal data, the general graph convolutional network (GCN) cannot make effective extraction of temporal information from the nodes. In this regard, we use a GAT network consisting of a Graph Attentional Layer, which aggregates the information of neighbor nodes in spatial latitude, and later aggregates the information of neighbor nodes in temporal latitude through a Multi-Head-Attention mechanism, to train the model and make predictions.[3]

B. Attention Mechanism

From the nomenclature of the Attention Mechanism, it is clear that it is originated from the Human Visual Attention Mechanism. The Human Visual Attention Mechanism is a signal processing mechanism in the brain critical to human vision. After rapidly surveying the entire image to identify the target area, the human eye then concentrates more intensely on that area to gather additional detailed information, while ignoring irrelevant information. The Attention Mechanism in deep learning is essentially modeled after this human visual attention mechanism. The core goal is to select the more critical information to the current task goal from a large amount of information[4] , as shown in Figure 1.

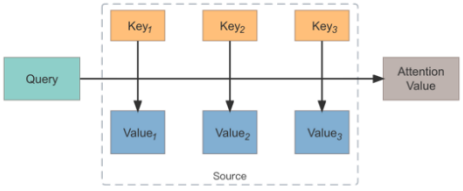

Fig. 1. Attention Mechanism

One way to conceptualize the Attention Mechanism is as follows (as illustrated above): think of the individual components in the source as a set of <*Key, Value*> pairs. At this time, given the *Query* of an element in Target, the weight

coefficient of Value corresponding to each *Key* is obtained by calculating the similarity between the *Query* and each *Key*. Then the *Value* is weighted and summed to obtain the Attention value. In essence, the Attention Mechanism involves computing a weighted sum of the values associated with each element in the source, where the query and key are utilized to determine the weighting coefficients for the corresponding value. The formula is

$$Attention(\text{ Query, Source }) = \sum_{i=1}^{L_x} Similarity(Query, \text{Key}_i) * \text{ Value}_i \quad (1)$$

where $L_x = |Source|$, that is L_x represents the length of the source. In this project, the set of Source is the set of neighboring points of the traffic road network, and the key is the same as the value, which is the speed of each node at a specific time.

As for the specific calculation process of the Attention Mechanism in this project, we can summarize it as follows:

a) We calculate the similarity between the two in the set of neighboring points based on the speed value.

$$s_i = Similarity(\text{ Query, Key}_i) = \text{ Query} \cdot \text{Key}_i \quad (2)$$

b) Normalize the raw scores in step 1 to derive the weighting coefficients

$$a_i = Softmax(Sim_i) = \frac{e^{Sim_i}}{\sum_{j=1}^{L_x} e^{Sim_j}} \quad (3)$$

c) Weighted summation of speed value according to weighting coefficients, resulting in attention value

All steps are shown in the figure 2 below:

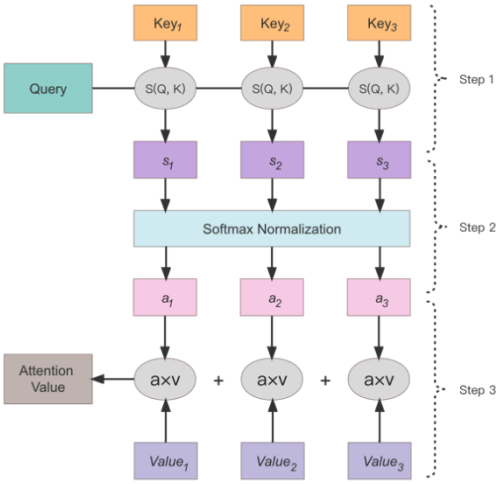

Fig. 2. Weight Summation

C. Multi-Headed-Attention mechanism

It is not enough to use the Attention Mechanism mentioned above to handle the information between nodes. This is because the attention mechanism only considers the spatial connectivity information of nodes, while ignoring the temporal connectivity information of nodes. Also, according to the above introduction, we can find that if we use the attention mechanism alone, the model will focus too much on its own position when encoding the current situation. Therefore, we use the Multi-Head-Attention Mechanism, which allows the model to process the data temporally and consider the information of neighboring nodes. The Multi-Head-Attention structure is shown in the following figure 3. [5]

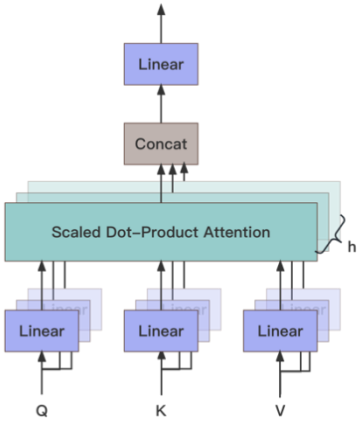

Fig. 3. Multi-Headed-Attention mechanism

We can see that the Multi-Head-Attention mechanism is actually a multi-group attention process for the original input sequence; then, the attention value of each group is stitched together for a linear transformation to obtain the final output. The calculation formula is:

$$MultiHead(Q,K,V) = Concat(head_1, \ldots, head_h)W^O \quad (4)$$

$$head_i = Attention\left(QW_i^Q, KW_i^K, VW_i^V\right) \quad (5)$$

, where $W_i^Q \in R^{d_{\text{mdl}} \times d_k}, W_i^K \in R^{d_{\text{mdl}} \times d_k}, W_i^V \in R^{d_{\text{mdl}} \times d_v}, W^O \in R^{hd_v \times d_{\text{mdl}}}$

D. Graph Attentional Layer

The Graph Attentional Layer is the basis of the whole GAT model. The input of a single Graph Attentional Layer is a set of temporal vectors of nodes.

$$h = \left\{\overrightarrow{h_1}, \overrightarrow{h_2}, \ldots, \overrightarrow{h_N}\right\}, \overrightarrow{h_i} \in R^F \quad (6)$$

where N denotes, the number of neighboring nodes of the node and F denotes the corresponding latitude of the timing vector.

The output of each layer is a new set of feature vectors for the nodes:

$$h' = \left\{\overrightarrow{h_1'}, \overrightarrow{h_2'}, \ldots, \overrightarrow{h_N'}\right\}, \overrightarrow{h_i'} \in R^{F'} \quad (7)$$

where F' denotes the new feature vector latitude of the node (not equal to F).

The structure of a graph attentional layer is shown in the following Figure 4.

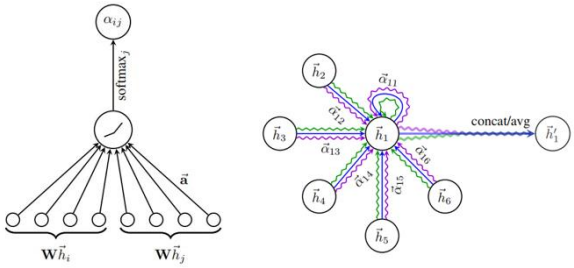

Fig. 4. Graph Attentional Layer[3]

Specifically, the Graph Attentional Layer first performs attention processing based on the input set of temporal vectors of neighboring nodes to calculate the correlation degree between nodes. It then uses the multi-head-attention mechanism to perform differentiated information aggregation on neighboring nodes. The specific steps are as follows.

a) Model Input

b) Linear Transformation

To obtain sufficient expressiveness, the input features are transformed into higher-level features. We use a learnable linear transformation $\boldsymbol{W} \in R^{F' \times F}$, to map the model input to a higher dimensional space

c) Attention Mechanism

After using the linear transformation, we employ an attention function to obtain the attention distribution.

$$e_{ij} = a\left(\overrightarrow{Wh_i}, \overrightarrow{Wh_j}\right) (8)$$

, where e_{ij} stand for the importance of features for node j to node i. The above equation is a general case where the graph's structure is not considered, and each node participates in the attention computation of other nodes. In fact, in this project, we use masked attention (using the adjacency matrix as a mask) to introduce information about the structure of the graph-we compute e_{ij} only for node j in the set N_i of neighboring nodes belonging to node i.

The attention scoring function we use is specified by splicing the high-dimensional spatial representations of node i and node j, followed by a linear transformation $\overrightarrow{\boldsymbol{a}} \in R^{(2F')}$, and finally a nonlinear activation function LeakyReLU:

$$e_{ij} = LeakyReLU\left(\overrightarrow{a}^T \left[\overrightarrow{Wh_i} | \overrightarrow{Wh_j}\right]\right) (9)$$

d) Softmax Normalization

To make the weight coefficients on different nodes easy to compare, we normalize the weights of each node to obtain the attention distribution.

$$\alpha_{ij} = Softmax_j\left(e_{ij}\right) = \frac{\exp\left(e_{ij}\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(e_{ik}\right)} (10)$$

e) Model output

f) Compute weighted average

Using the attention distribution obtained, the feature representations of neighboring nodes are weighted and averaged. The final output is then generated by assembling each sub-attention structure through a Multi-Headed-Attention Mechanism.

III. Establishment of Prediction Model

A. Data

1) Uber Movement

Every hour of every day, people are using Uber to travel through more than 450 cities worldwide. From Sydney to New York, Uber has been working to understand these cities to make them cleaner, more efficient, and less congested. Along the way, Uber has found local leaders, urban planners, and civic communities working to decipher their cities' commuting problems and figure out the best ways to invest in new infrastructure. [6]

In January 2017, Uber released the "Uber Campaign," a tool for urban planners and researchers to study improving urban transportation. Specifically, the dataset includes more than 2 billion Uber trips in cities such as Bogotá, Boston, Johannesburg, Manila, Paris, Sydney, and Washington, D.C.. It includes the arithmetic mean, geometric mean, and standard deviation of travel times aggregated over a selected date range between each pair of zones1 in these cities. Uber Movement is available to the public and can be downloaded directly from the [Uber Movement website] to download in .csv format, as shown in Figure 5.

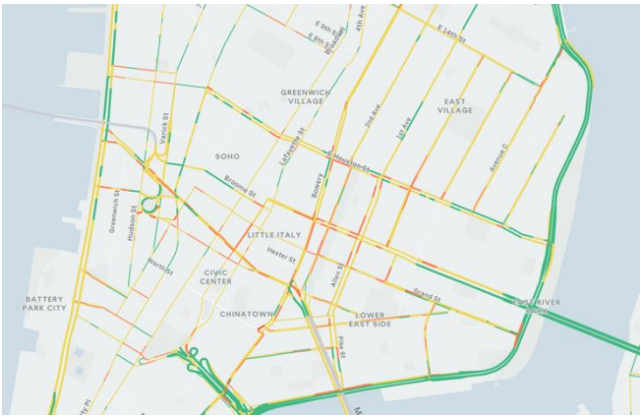

Fig. 5. The Uber Web interface colors roads in the New York city based on the average travel speed

In this paper, we chose to use the January 2020 data as our dataset. In this dataset, velocity data in hourly units for 31 days and 24 hours are included. Table I presents the information contained within this dataset.

TABLE I. ATTRIBUTES IN UBER MOVEMENT DATA

Attribute	Description	Data Type
Year	Year(Local city time)	Int
Month	Month 1-12(local city time)	Int
Day	Day 1-31(local city time)	Int
Hour	Hour 0-23(local city time)	Int
Utc_timestamp	Unix time for start of hour (UTC)	Int
Segment_id	Movement ID that maps to a specific road segment.	Text
start_junction_id	Movement ID that maps to start intersection of traversal.	Text
end_junction_id	Movement ID that maps to end intersection of traversal.	Text
Osm_way_id	Corresponding OpenStreetMap Way ID for this segment.	Bigint
osm_start_node_id	Corresponding OpenStreetMap Node ID for this junction.	Bigint
osm_end_node_id	Corresponding OpenStreetMap Node ID for this junction.	Bigint
speed_mph_mean	Average speed of Uber vehicles on this road segment in mph	Float
speed_mph_stdev	Standard deviation of speeds on this road segment in mph.	Float

2) OpenStreetMap

OpenStreetMap is a collaborative project for building free content online maps to create a world map with free content that all can edit and easy navigation options for available mobile devices. [7]

The data format used by OSM is a topographic data structure, which consists of four core elements [8].

a) **node**: stores the latitude and longitude, indicating the location, but not the actual size of the node on the map, such as an attraction or a mountain, or a store or a restaurant, or as part of a path. Nodes can be attached to paths and relationships.

b) **way**: An ordered arrangement of nodes, presented as a dash, can also loop back to the starting node to form a closed path and can be presented as a circular path or as a polygonal area. This source material can be used to present linear material, such as a street, river, etc., or a polygonal area, such as farmland, a park, a parking lot, a building, a campus, or a lake or a lake forest. Paths must have nodes displayed on the map and can be attached to relationships. Path information can be calculated as the length, area, or perimeter of a polygon.

c) **Relation**: Nodes, paths, and relations are ordered (the three types of primitives are collectively referred to here as "members"), where each member optionally has a "role" (string) that determines the nature of that member in the relation. Relationships are used to represent the relationship of individual primitives (nodes, paths, and relationships), such as the turn restrictions of a road, a boundary formed by different paths, a national, provincial, or railroad route, or multiple polygons in the middle of a region (e.g., an atrium surrounded by a circular building, or an island in a body of water), where a "role" string can be used to describe the relationship between them.

d) **tag**: A key-value pair (all keys are strings) that stores the metadata (type, name, and physical characteristics of the object) of the objects on the map, giving meaning to the OSM information and representing the existence of something in the real world, and information about it. Tags cannot exist independently. They must be attached to an existing object, i.e., a node, path, or relationship.

With the data provided by Uber, we can obtain the OSM IDs corresponding to each road segment's start and endpoints, so we can build a graph to model the basic urban structure in which the trip occurs. Using the data provided by Uber Movement, we can quickly build a basic city structure on OpenStreetMap, as shown in Figure 6.

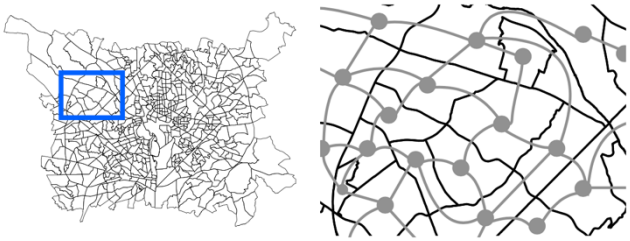

Fig. 6. The OSM System contains a set of polygons on which we'll zoom in (left). Looking closely at the region in the blue rectangle, we define a graph where every node represents a region and two regions are linked if they are adjacent (right)[9]

B. Exploratory Data Analysis

The overall process of our Exploratory Data Analysis is presented roughly, as shown in the figure 7 below.

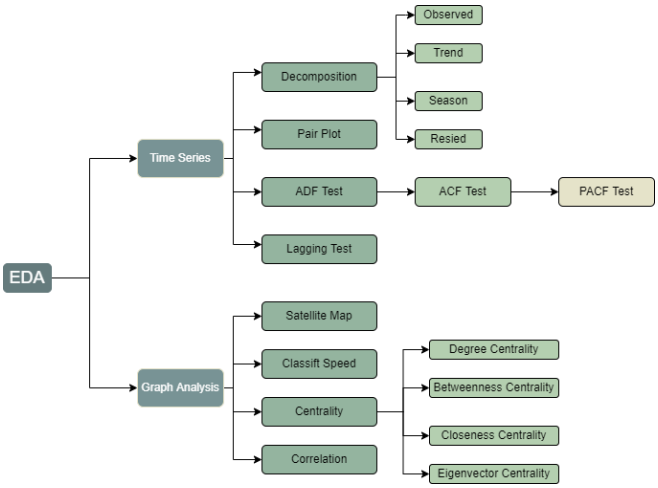

Fig. 7. EDA Steps

Our exploratory data analysis is carried out in two main aspects. One aspect is to analyze the data in terms of time series to explore the distribution of the data over time. On the other side, we explore the data by using graphs to determine corresponding features in the network results. The reason for dividing the EDA into two parts is that our data is road data with a network structure. Based on this network data, we collect information about the speed of vehicles and the time during a month, so we divide the exploration process into two parts: temporal and spatial

1) Time Series Data

We analyze the data from four aspects regarding the Time Series: Decomposition, Pair Plot, ADF Test and Lagging Test.

a) Decomposition

In Decomposition, we investigate whether the data has a cyclical pattern of change. We divided Decomposition into four small phases for analysis. We first drew four decomposition graphs to analyze the time-series relationship of each hour of the road network. As shown in the figure 8 below

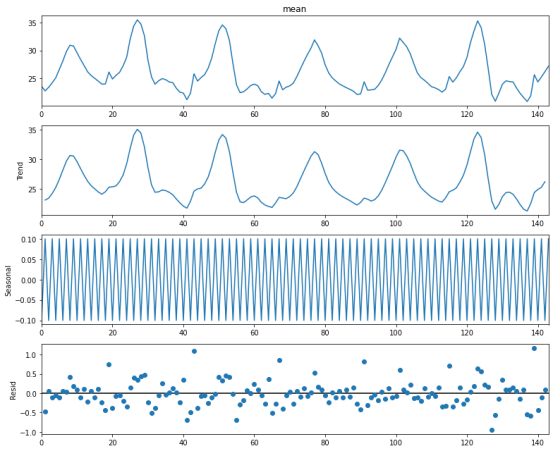

Fig. 8. Hourly Speed Decomposition

The first graph is a line plot that depicts the observed hourly average speed. This line plot reflects a smooth trend, with periodic increases and decreases over time, and no significant unusual changes. The second graph displays the overall trend change data, indicating that the average speed data trend is smooth, with no significant increase or decrease. The third graph exhibits the seasonal data of the time-series, revealing a periodic variation of the average velocity data with a steady frequency, confirming our assumption that the time-series data is seasonal. The fourth graph represents the residual data, which highlights the random variation in the time-series that we need to focus on predicting. From these images, we observe that the residual values of the average velocity data are symmetrically distributed around 0, suggesting that there may be a linear correlation, which we will explore further later.

In a similar manner, we utilized the four decomposition plots to investigate the temporal relationship in the daily average speed data for all road segments. This is illustrated in the combined plot below, as shown in Figure 9.

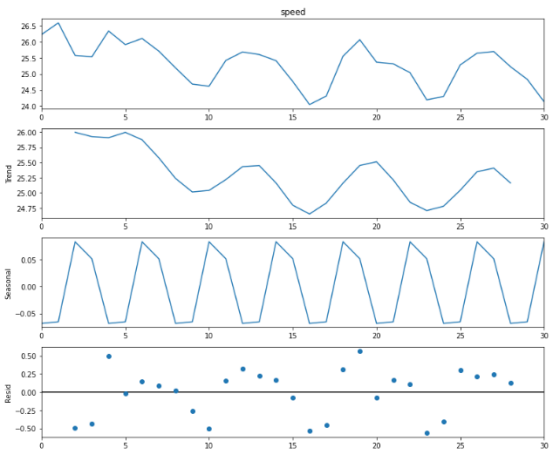

Fig. 9. Daily Speed Decomposition

The first graph is a line graph of the actual observed daily average velocity. We used the line graph to reflect the curve of the data. You can see that there is a downward trend here, indicating that people's driving speed showed a downward trend in January 2020, probably due to weather, such as cold weather with snow causing icy roadsides or snow blocking roads, etc. If weather data is available, we can compare it for deeper exploration. The second graph shows the overall trend change data. From the graph, we can find that the trend of the average speed data is smooth, offering a gradual decline, which fits with the first graph. The third graph shows the seasonal data exhibited by the whole series. By looking at the images, we find that the average velocity data varies periodically with a steady frequency. Therefore, we consider our time-series data to be seasonal. Every so often, the velocity will show the same trend, for example, every week, with the Monday of each week showing the same velocity. The fourth graph is a plot of the residual data. The residuals represent the random variation in the time series, which we must focus on predicting. By looking at the images, we find that the residual values of the average velocity data are regularly distributed on both sides of 0, indicating that there may be a linear correlation in velocity as well.

Additionally, we applied decomposition analysis to the daily number of travel records to examine the corresponding time-series relationship. This is depicted in the combined graph below, as shown in Figure 10.

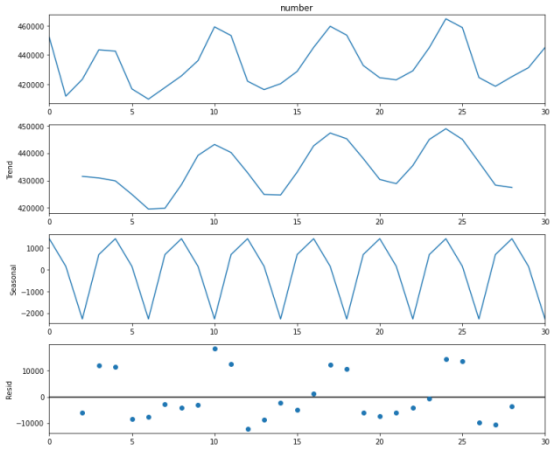

Fig. 10. Daily Records Number Decomposition

The first graph is a line graph of the actual values we observed. We use the line graph to reflect the trend of the data. We can see an upward trend, indicating that more and more uber vehicle trip data records are collected as January progresses, showing an opposite change to the decrease in average daily speed. The second graph shows the overall trend change data. From the graph, we can see that the trend of the transportation data is smooth and offers a gradually increasing trend of change. The third graph shows the seasonal data exhibited by the whole series. We find that our data vary periodically with a steady frequency by looking at the pictures. Therefore, we consider our time-series data to be seasonal; for example, more recorded data may be collected weekly on Mondays than Fridays. The fourth graph is a plot of the residual data. The residuals represent the random variation in the time series, which we need to focus on predicting. By looking at the images, we find that the residual values of the data are regularly distributed on both sides of 0, as analyzed in the previous two graphs, indicating that there may be a corresponding linear relationship here as well.

b) Correlation Analysis

We perform the correlation analysis of the pair plot after decomposition. As shown in the following pair plot, as shown in Figure 11.

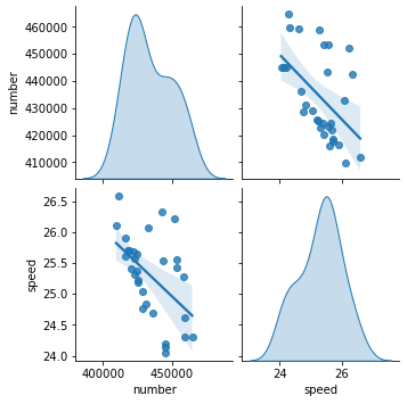

Fig. 11. Correlation Pair Plot

To examine the relationship between two variables, we used a pair plot to compare daily collected vehicle data records with daily average speed. Figures 1 and 4 show the corresponding linear plots indicating the concentrated distribution of the data, with the overall distribution of the daily collected uber data volume falling between 40,000 and 48,000, while the daily average speed is primarily distributed between 24 and 26. Figures 2 and 3 display a scatter plot of the two features, revealing a negative linear correlation. We explain this correlation by suggesting that an increase in the amount of data collected leads to a decrease in average speed, as more vehicles on the road increase the likelihood of traffic congestion, thus reducing overall average speed. This conclusion is supported by the results of the previous decomposition data, which indicates a negative linear correlation between the amount of daily uber driver data collected and the average speed.

Subsequently, we conducted an ADF test on our series data, which revealed a p-value of 0.008689, below the confidence level of 0.05. The ADF test assumes the existence of a unit root, and by comparing the p-value to the confidence level, we determine whether to reject the original hypothesis. In this case, the p-value was less than 0.05, indicating that there is no unit root, and the data is smooth and not random. This allows us to proceed with the ACF Test and PACF Test, and ensures that our information is not randomly selected.

Since we have illustrated that our data is stable by ADF Test, we are pleased to perform ACF Test. as shown in the figure 12 below.

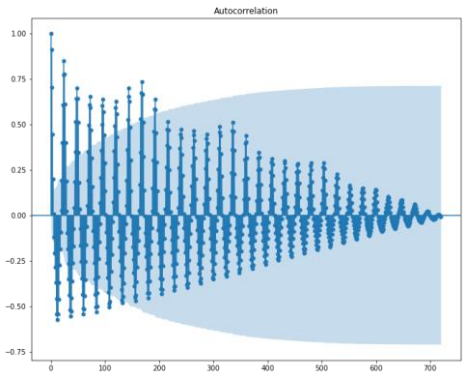

Fig. 12. ACF Diagram

Here we use ACF Test to test the stability of the time series. We plot the Autocorrelation image as shown above. For a stable time, series, autocorrelation means that it is possible to predict the future using historical data. The blue area is the error range. The data in this region are considered insignificant. The pattern of decreasing ACF amplitude generally implies the presence of autocorrelation, as we have found and demonstrated in our previous time series exploration. From the figure, we can see that the ACF is stable as its values gradually converge and approach zero, thus demonstrating the stability of the time series, which means the series is of no randomness. Because both ADF Test and ACF Test illustrate the strength of the data, we can continue to explore deeper and analyze our data in terms of PACF.

We plotted the partial correlation image as shown in the figure 13 below.

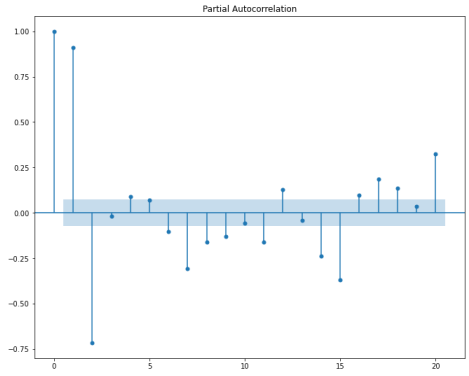

Fig. 13. PACF Diagram

PACF considers only cycle-specific correlations, which gives us a good starting point for determining the autocorrelation cycle, so in PCAF, we can find which processes are autocorrelation. In the ACF plot, the ACF is a trailing tail that gradually tends to 0. In PACF, it is very close to 0 when the lag is 3, so we can consider using a model like ARMA (3,0) to form a time series model to use past data to make predictions about possible future data.

After completing the above stability test, our time series data are smooth and not randomly distributed but show regular variation, consistent with the decomposition phenomenon. At the end of the time series exploration, we performed a lagging test, which compares the current data after a delay period to determine whether they have a corresponding relationship. As shown in the figure below, we set the delays as 1, 2, 3, and 4hour to analyze the relationship between the two, as shown in Figure 14.

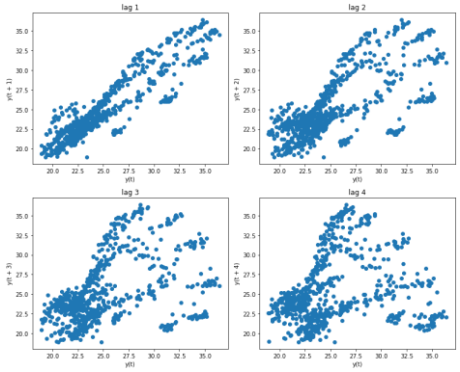

Fig. 14. Lagging Test

As shown in the figure, when lag is 1, we can find a clear positive linear correlation, and as the lag value gets larger, the divergence of the scatter plot increases. The linear correlation decreases, but it still exists. Because we use the hourly speed data of road network sections, our data has both temporal and spatial attributes. When congestion occurs on a road section, the road sections around this road section are likely to receive the impact of that road section and experience congestion. And as time goes by, this congestion will gradually disappear, so when the lag value is small, indicating that it is not far from the current data time, the two will show a very, very strong linear correlation, as shown in the lag1 figure, the scattered points almost overlap to form a thick line. And as the lag value increases, as time passes, the two gradually diverge, and the linear correlation gradually decreases but still has some degree of linear correlation. This is a significant finding; the performance of the data will show a positive linear correlation within a specific time difference, while the correlation gradually decreases until it disappears as time goes on

2) Graph Data

a) Data Visualization

By using the JOSM editor, we have generated the respective network on an accurate New York City map, as shown in Figure 15.

Fig. 15. Graph Data Visulization

In the above figure, the highlighted road segments are the ones we have selected. The selected road segments are mainly located in Manhattan, Brooklyn, and Queens, most of which are in the busy areas of New York and have a high practical significance. At the same time, we also observed that some road sections in our selected subplot are in remote locations, which are some distance away from the busy areas of New York. These road segments may be the relative outliers of our prediction task in this project and the data itself.

b) Congestion Level Division

Because we need to predict traffic congestion based on speed. We calculate the specific banner value among different classes According to Level of Service (LOS) with Average Speed and Traffic Conditions for Arterial Roads [10]. We divide into four zones according to different speeds as follows.

- [33.80259, ∞): Unblocked
- [20.87807, 33.80259): Mild Congestion
- [15.9071, 20.87807): Moderate Congestion
- [0, 15.9071): Serious Congestion

The histogram of these four road sections as a percentage of the total number of road sections is shown below. We can see that congestion accounts for the largest share of them, over 10,000 out of about 21,000 data. The bar chart shows that New York City is at least mildly congested in most cases, as shown in Figure 16.

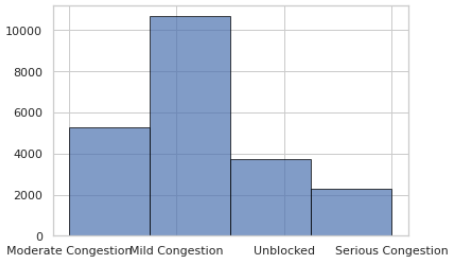

Fig. 16. Histogram of Congestion Level Distribution

c) Centrality

we put four different centralities and speed together to plot a heat map to explore a correlation between these four centralities and the speed, as shown in Figure 17.

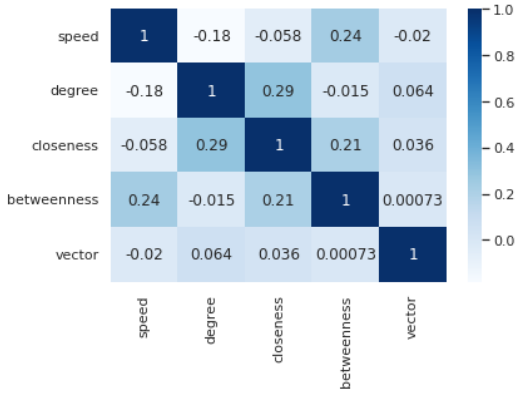

Fig. 17. Correlation between Centralities and Speed

The graph shows that speed and betweenness are positively correlated with a correlation coefficient of 0.24 since betweenness reflects whether a node is a "bridge" or not. This is consistent with our view that "bridges" tend to be suburban roads, and "bridges" tend to be "open." Speed is negatively correlated with the degree, with a correlation coefficient of -0.18. The higher the degree of a node, the more likely it is to be a "junction." Speed is negatively correlated with closeness, with a correlation coefficient of -0.058. The higher the closeness of a node, the more likely it is to be a "junction." The higher the closeness of a node, the closer it is to an urban area, and the more likely it is to be congested. Therefore, the higher the closeness, the lower the traffic speed. The correlation coefficient of the vector is approximately equal to 0. This is consistent with our view that vector is not related to traffic speed.

C. Model structure design

1) Model Input

Traffic data processing is not quite the same as standard data processing. We use the Sliding Window approach since our traffic data is a series of time-series data. The following are the specific processing steps, as shown in Figure 18.

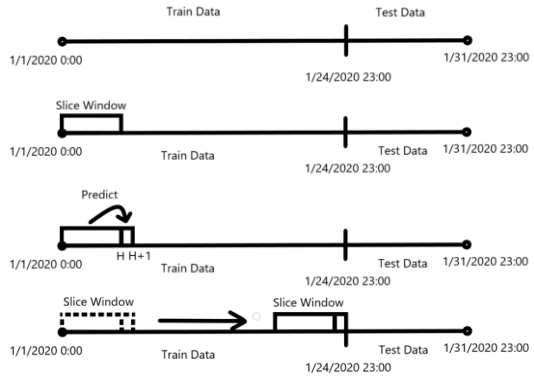

Fig. 18. Sliding Window on data

a) The data for each hour from January 1, 2020 at 00:00 to January 31, 2020 at 23:00 (744 hours in total) were segmented by using the first 24 days (576 hours) as the training set and the last 7 days (168 hours) as the test set.

b) Artificially define a Sliding Window size, in this project, the window size is 24 hours

c) Using the data within the window (called historical data, H), predict the traffic flow at the moment H+1

d) Repeat step 3 and move this window by one time unit at a time until all training samples are obtained

2) Model Structure

Our whole GAT model consists of two parts. In the first part, the attention value of each node is calculated using Graph Attentional Layer. And in the second part, the attention value of each node is aggregated and output using Graph Attentional Layer according to the Multi-Head-Attention Mechanism.

IV. Model Comparison and Analysis

A. Model Comparison

We used the Prophet statistical model and Random Forest Model to compare and evaluate the GAT model.

1) Prophet

Prophet is published by Facebook's Core Data Science team. It depends on a contribution model where non-linear trends are fit with weekly and yearly seasonality and plus holidays. PROPHET is strong to missing data, capturing the shifts in the trend and large outliers[11]. In addition, it gets a reasonable estimate of the mixed data without spending manual effort [12].

2) Random Forest

Random Forest is a machine learning algorithm that utilizes Ensemble Learning, a prominent branch of machine learning, to combine multiple decision trees via blending techniques. Each decision tree is a classifier (assuming we are talking about a classification problem), so N trees will have N classification results for one input sample. The random forest integrates all the classification votes and designates the category with the most votes as the final output, one of the most straightforward Bagging ideas. [13]

3) Evaluation

By comparing the original data, and prediction results (as Fig. 19). The red color indicates the expected value, and the blue is the actual value. The horizontal coordinates indicate the time series of the test set (every 24 hours for the last 7 days), and the vertical coordinates are the velocity values. As we can see, the model can learn the temporal trend of the data relatively well, and the predicted speed is close to the actual value. The model performs well on the test set.

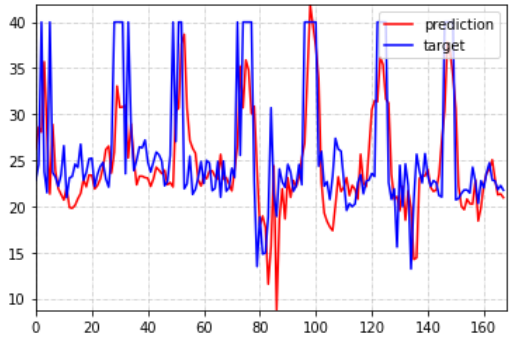

Fig. 19. Comparison of predicted results

TABLE II. Model Evaluation Matric

	MAE	MAPE	RMSE
GAT	7.1	0.29	9.55
Prophet	142.7	452.4	215.0
Random Forest	14.4	35.9	93.4

By comparing Table II, we can find that GAT has better performance when predicting traffic in New York City.

B. Model Analysis

1) Advantages

We believe that the GAT model has several advantages.

GAT is efficient. GAT does not need to use complex matrix operations such as eigenvalue decomposition compared with other graph models. The time complexity of the Graph Attentional Layer is $O\left(|V|FF' + |E|F'\right)$, which is the same as that of GCN. In the expression, |V| and |E| denote the number of points and the number of edges in the graph, respectively. Since our dataset is relatively large (a total of 21947 nodes, 74783 edges, and 744 data on each node), therefore, the time spent on training would be astronomical if other graph models were used.

GAT has a stronger representation capability. Compared with GCN, the importance of each node in GAT can be different, which fits well with one of the characteristics of our dataset: different sub-maps contribute differently to the whole map (different speeds of vehicles in different sections). At the same time, GAT can extract information about the data in terms of time latitude, which is also in line with the characteristics of traffic flow data - a sequence of temporal data.

The GAT model considers all neighboring nodes and does not assume the internal order of these nodes. All nodes of the domain in GAT do not need to be ordered and share the convolution kernel parameters.

2) Limitations

At the same time, we believe that the GAT has several disadvantages.

Unlike GNN, GraphSAGE, GAT directly selects first-order adjacent nodes as domain nodes. This is not very similar to reality; when traffic congestion occurs, it often impacts the first-order neighbor nodes, and the second-order, third-order are ignored.

The GAT calculation time is still too long. We are using an A40 graphics card, but it still takes nearly 20 minutes to run an epoch, making it difficult to adjust the optimal parameters.

V. Conclusion

Through this project, we understood and explored the task of analyzing and predicting traffic flow data, understood and applied the knowledge of graph neural networks, built proper GAT neural networks to cope with the task of traffic speed prediction in New York City, and the model has good performance. However, our model still has some limitations that need to be further overcome in the subsequent research work.

Acknowledgments

During the composition of this paper, I have been fortunate to receive substantial support and assistance.

I would first like to thank my professor, Professor Bhupesh Shetty, whose expertise was invaluable in formulating the research questions and methodology. Your insightful feedback pushed me to sharpen my thinking and brought my work to a higher level.

Also, I would particularly like to acknowledge my group mate for their wonderful collaboration and patient support.

References

[1] Goode Lauren. "Worth It? An App to Get a Cab." *The Wall Street Journal* 17 (2011).

[2] Uber "Uber Movement: Let's Find Smarter Ways Forward, Together." *Uber Movement*, movement.uber.com/?lang=en-US

[3] Veličković, Petar, et al. "Graph attention networks." *arXiv preprint arXiv:1710.10903* (2017).

[4] Niu Zhaoyang, Guoqiang Zhong, and Hui Yu. "A review on the attention mechanism of deep learning." Neurocomputing 452 (2021): 48-62.

[5] Vaswani Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

[6] Gilbertson Jordan. "Introducing Uber Movement." *Uber Newsroom*, 8 Jan. 2017, www.uber.com/newsroom/introducing-uber-movement-2/.

[7] Mark Anderson. "Global positioning tech inspires do-it-yourself mapping project." *National Geographic* (2006).

[8] Wiki Contributors. "Map Features." *Map Features - OpenStreetMap Wiki*, https://wiki.openstreetmap.org/wiki/Map_features.

[9] Pearson Mackenzie, Javier Sagastuy, and Sofia Samaniego. "Traffic flow analysis using uber movement data." *htts://snap. stanford. edu/class/projects* (2017).

[10] Cecaj Alket, et al. "Comparing deep learning and statistical methods in forecasting crowd distribution from aggregated mobile phone data." *Applied Sciences* 10.18 (2020): 6580.

[11] Abaya Ernesto B., et al. Instantaneous Fuel Consumption Models of Light Duty Vehicles and a Case Study on the Fuel Consumption at Different Traffic Conditions in Metro Manila Using Shepard's Interpolation Method. No. 2018-01-0075. SAE Technical Paper, 2018.

[12] Taylor Sean J., and Benjamin Letham. "Forecasting at scale." The American Statistician 72.1 (2018): 37-45

[13] Biau Gérard, and Erwan Scornet. "A random forest guided tour." Test 25 (2016): 197-227.