Rebuilding City-Wide Traffic Origin Destination from Road Speed Data

Guanjie Zheng*
The Pennsylvania State University
University Park, PA 16802, USA
gjz5038@ist.psu.edu

Chang Liu*
Shanghai Jiao Tong University
Shanghai, China
only-changer@sjtu.edu.cn

Hua Wei
The Pennsylvania State University
University Park, PA 16802, USA
hzw77@ist.psu.edu

Chacha Chen
The Pennsylvania State University
University Park, PA 16802, USA
cjc6647@psu.edu

Zhenhui Li
The Pennsylvania State University
University Park, PA 16802, USA
jessieli@ist.psu.edu

***Abstract*—Understanding city-wide traffic problems may benefit many downstream applications, such as city planning and public transportation development. One key step to understand traffic is to reveal how many people travel from one location to another during one period (we call TOD, short for temporal origin-destination). With TOD, we can rebuild the city-wide traffic by simulating the volume and speed on each road segment.**

Frequently used mobility data, e.g., GPS trajectories, surveillance cameras, can only cover a subset of vehicles or selected regions of the city. Hence, we propose to use pervasive speed data to recover TOD, and use other mobility data as auxiliary data. To the best of our knowledge, we are the first to work on this challenging problem. It is highly challenging because the speed is generated from a complex process from TOD, and there exists multiple TOD distributions that may generate similar city-wide road speed observations. We propose a new method that models the complex process via separate modules and takes auxiliary data to eliminate infeasible solutions. Extensive experiments on synthetic and real datasets have shown the superior performance of our model over baselines.

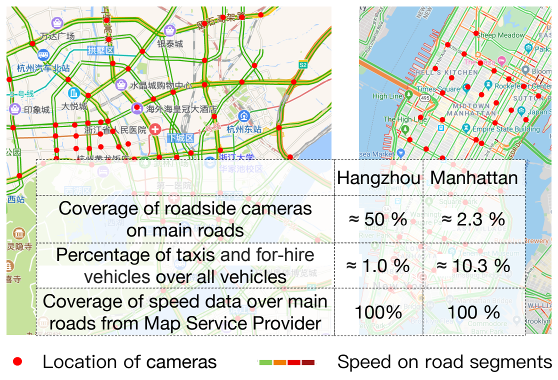
***Index Terms*—Mobility data, origin-destination estimation, urban computing**

I. Introduction

Traffic has been playing an essential role in urban city development, with impacts on many aspects of human life, e.g., commuting, shopping, and entertainment. Smooth and fast-moving traffic can effectively enable the city development in a larger region and people can enjoy the various venues around. In contrast, traffic congestion may lead to high economic loss. Therefore, urban cities are in urgent need of smart traffic policies and planning. Unfortunately, we can not directly develop these policies in the real world, due to its potential real cost (i.e., traffic jams).

To support and test city-wide traffic policy making, many efforts have been made [1], [2] in traffic simulations. For instance, these simulators usually take as input the trip counts of vehicles traveling from one location to another. Then, the simulator will simulate people's driving behavior (e.g., how

*Equal contribution.



	Hangzhou	Manhattan
Coverage of roadside cameras on main roads	≈ 50 %	≈ 2.3 %
Percentage of taxis and for-hire vehicles over all vehicles	≈ 1.0 %	≈ 10.3 %
Coverage of speed data over main roads from Map Service Provider	100%	100 %

Fig. 1. Spatial coverage of camera, trajectory, and speed data. Speed data has city-level coverage, while volume and TOD inferred from camera or trajectory data has low coverage.

people choose their routes, and how fast people can drive) according to some transportation models and assumptions. However, these involve two key issues. First, the trip counts data are not always available. Second, the pre-assumed transportation models and assumptions are not always true. For example, most micro-level simulations assume people will drive according to a pre-defined car-following model (e.g., accelerate as much as possible). However, in the real world, the traveling speed people can reach on a road segment might highly depend on the traffic volume. Macro-level traffic simulations may try to model the mapping between TOD, volume and speed, by assuming a linear mapping matrix [3]–[6].

To comprehensively understand how city-wide traffic operates, we briefly introduce the formation process of city-wide traffic as follows. (1) During a certain time interval, we can use a 2-D matrix F_t to represent people's travel intent, where i, j-th element of F_t represents the trip counts from origin region i to destination region j. (2) Then, people will head to their destinations according to their preferred routes. This causes the volume on each road segment to form. (3) Given the attributes of the road (e.g., number of lanes, speed limit), the volume on one road segment will determine the real-time speed of the vehicles running on this segment. In short, this process can be summarized as the impacting chain of the following three key quantities: temporal distribution of origin-destination (TOD) $\rightarrow$ volume $\rightarrow$ speed. OD(origin-destination) and TOD are different concepts, where OD is the combination of origin and destination pairs and TOD is the temporal distribution on OD pairs.

Thus, in this paper, we try to answer the following two key questions in building a real traffic simulation.

1) Recover the TOD distribution from various traffic data input.
2) Model the TOD $\rightarrow$ volume $\rightarrow$ speed chain.

More generally, we call this process rebuilding city-wide traffic. Note that, although researchers may have different opinions on how the city-wide traffic operates, here we just propose a possible generation process. It is easy to add more modules into this framework in order to incorporate other levels of observations (other than TOD, volume, and speed).

Thanks to the development of sensors and mobile devices, an increasing amount of mobility data are being collected, e.g., GPS trajectories, surveillance camera records, and mobile apps location records. However, the camera data usually only covers some central regions of the city, while the GPS trajectories may only be available on some cars, like taxis (as shown in Figure 1). Hence, it is difficult to directly convert these raw data to city-wide TOD or volume observations. In addition, it is difficult to acquire complete and sufficient TOD data because TOD data is usually collected by survey and needs a lot of human effort. This prevents the prediction method using historical TOD information from success. Fortunately, the average speed on a road segment can be easily probed by a few vehicles. Therefore, current navigation software like Google Maps can provide pervasive speed observations around the city, which can be used to help recover the TOD. Regrettably, map server companies including Google Maps do not care about people's real TOD, since they only need speed information to estimate travel time and recommend route.

Therefore, in this paper, we try to answer the previously mentioned two questions using the city-wide speed data as the major input, while traffic data in other levels (e.g., volume) can be used as auxiliary inputs.

To the best of our knowledge, this problem has never been investigated thoroughly in the literature. Early studies [7], [8] have tried to infer TOD from census data. However, the unrealistic simplified assumption (TOD is proportional to the population and inverse of the distance) and the static population count can not infer the dynamic TOD. Some other studies [6], [9] propose a linear mapping matrix between road segment volume and TOD. These models fail easily considering that different competing traffic on different roads will delay each other (the linear mapping matrix will not hold). Further, the volume data they used is usually not available for the city level. In addition, our problem is also different from the well-investigated traffic prediction problems, since these studies [10], [11] utilize historical traffic to predict future traffic, while we do not assume any TOD data is given for our problem. These studies do not reveal how the traffic operates and hence can not help rebuild the traffic system (i.e., building traffic simulation). This limits their application to predictions with known patterns, e.g., they can not answer how the traffic will change if the environment changes (e.g., a new bridge is built or road constructions are going on). We show in the experiments how our framework can simulate the traffic even the environment changes.

Rebuilding city-wide traffic impacting chain is challenging due to the following three reasons. First, *speed observations are the result of a complicated process.* Generally, this process can be described in the previously mentioned three stages: TOD $\rightarrow$ volume $\rightarrow$ speed. This complicated affecting chain makes this problem difficult to model. Second, *the available traffic data have different coverage and describe different quantities.* The GPS trajectories and camera data will put constraints on the TOD of certain groups of vehicles and the volume of certain road segments. How to combine them in a unified framework is a challenging problem. Third, *there exists multiple solutions of TOD distribution that may lead to a similar city-wide speed observation*. For instance, the high volume on major roads might be caused by travelers from several nearby communities. If the travelers from one community rise a bit while the travelers from one other decrease a bit, the volume or speed observation may stay almost unchanged. These may correspond to multiple possible TOD distributions, while only one of them matches the groundtruth.

In view of the aforementioned challenges, we propose a framework called OVS (**O**rigin-destination-**V**olume-**S**peed). It contains three modules to model the generation of TOD, volume, and speed respectively. Further, through this modular model design, OVS can incorporate the data from different levels (TOD level, volume level, or speed level) together. Specifically, though using speed as the major input, OVS can take auxiliary data (e.g., camera volume data, census data) to constrain the recovered solution (TOD, volume, and speed). Though spatially sparse or temporally static, these auxiliary data will effectively help to filter the unreasonable solutions and produce the most feasible one.

Our contribution can be summarized as below.

- We are the first to solve the problem of recovering city-wide TOD from pervasive speed data. We do not assume any TOD data is given. This problem can help us rebuild city-wide traffic (TOD $\rightarrow$ volume $\rightarrow$ speed).
- We propose a modular framework containing three modules: TOD generation, TOD-volume mapping, and volume-speed mapping. Our framework can also take other auxiliary data as input to eliminate infeasible solutions.
- Experiments on synthetic and real-world datasets have shown that OVS is significantly better than baseline methods. Case study results of OVS also match well with the real-world experiences.

II. Related Work

Building a real traffic simulator essentially requires accurate estimations of OD trip counts, volume, and speed [12]–[14]. There are two major groups of methods in solving these estimations, transportation methods, and data-driven methods.

Transportation Methods. Transportation methods usually simulate the traffic in the order that the causation relation forms: TOD - Volume - Speed. In order to infer the *TOD trip counts*, people have used search algorithms to search the TOD that satisfies the volume observation on the road links, e.g., genetic algorithm [15]–[17]. In addition, some studies propose the Gravity model [7], [8] which assumes OD flows are positively correlated to the product of the population of the origin and destination region and negatively correlated with the distance between two regions [18]. For *TOD-volume modeling*, traditional methods usually assume a linear assignment matrix mapping the TOD to the road link volume, and utilize statistical methods to estimate this assignment matrix. The often-used statistic methods include generalized least square [5], [6], maximum likelihood estimation [19], and bayesian models [9], [20]. Recently, some other methods further consider routing strategy [9], [21] and dynamic transition between consequent road link volumes [22], [23]. In addition, researchers have proposed several models to describe the *mapping between volume and speed*, e.g., fundamental diagram [24], [25]. Generally, when the volume reaches or surpasses the capacity of the road, speed will decrease as volume increases.

However, none of these methods have integrated the three generation steps together. Besides, their strong assumption (e.g., linear assign matrix between TOD and volume, TOD being proportional to population) might not hold in the real world. Therefore, we propose a data-driven pipeline to do these three tasks together.

Data-driven Methods. Recently, data-driven methods have been used to tackle the traffic prediction problem. People usually assume historical traffic is known. Problem-wise, people have worked on predicting TOD [10], [11], [26], volume [27], [28] and speed [29]. Method-wise, people have developed different methods, including ARIMA [28], non-parametric methods [28], graph neural nets [10], [11] and RNN [30]. However, all of these methods are doing spatial-temporal prediction using historical traffic data, rather than modeling the process of how traffic comes into being (TOD-volume-speed). Therefore, these methods do not apply to our problem. Our problem is much more challenging and requires a deep understanding of the generating model of observed link volumes and speed from the root cause of OD flows.

III. Preliminary

The problem scope is within a city which is divided into K regions and M links of road segments connecting the regions. We rebuild the traffic system through three stages: TOD $\rightarrow$ Volume$\rightarrow$Speed, as shown in Figure 2. Note that TOD is on the region level, while Volume and Speed are on the road level. Specifically, TOD reflects the trip count from every origin region to destination region, while Volume and Speed reflect

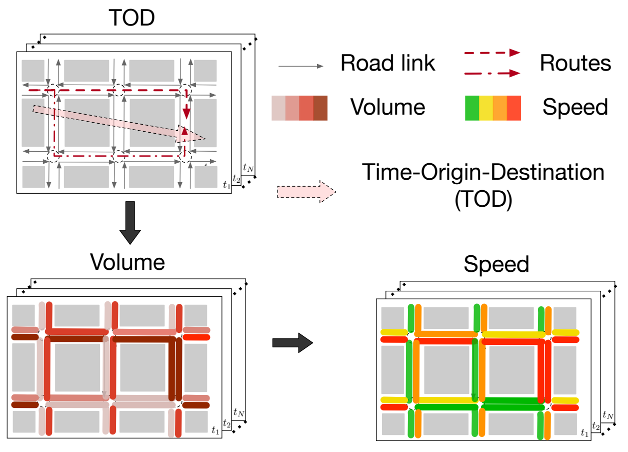


Fig. 2. Illustration of TOD, route, link and its corresponding volume and speed at time t.

TABLE I
Summary of Notations

Notation	Meaning
l_j	j-th link (road segment)
$v_{j,t}$, $q_{j,t}$	average speed and volume of link l_j at time t
r, R	region r, complete set of regions R
o, d	origin, destination
$\mathcal{V}$	average speed of the whole city
$\mathcal{F}$	3-D time-origin-destination tensor
$\mathcal{G}$	2-D time-origin-destination tensor (OD given)
T	total time span length

the volume and speed on each road. We define some basic concepts in our problem as follows.

- **Time interval.** The total time span is divided into T time intervals with a certain length (e.g., 10 minutes). We use t to represent the t-th time interval.
- **Link.** Each direction of one road segment is defined as a link l. Average speed $v_{j,t}$ and volume $q_{j,t}$ within the t-th time interval are defined to measure the congestion level on link l_j. In addition, we use $\mathcal{V}$ to represent the speed observation of the whole city, over the whole time interval T.
- **Region.** The whole city is divided into a set of smaller regions $R = \{r\}$ according to the information on OpenStreetMap . A region r can be as small as one block.
- **OD and TOD.** A trip is defined as a movement from an origin $o \in R$ to a destination $d \in R$. Then, the trips in a city can be represented as a 3-D tensor $\mathcal{F}$, with each cell $\mathcal{F}[t, o, d] = c$ representing that, at time t, the trip count from origin o to destination d is c. $\mathcal{F}$ can also be rewritten as a 2-D tensor $\mathcal{G}$, where $\mathcal{G}[i, t] = c$, and i is the index for OD pairs.
- **Routing.** Traveling from o to d, people may follow different routes, with each represented as a sequence of road links.
- **OD i contains link l_j.** We call OD i contains link l_j if link l_j is in one of the routes of OD i.

https://www.openstreetmap.org/

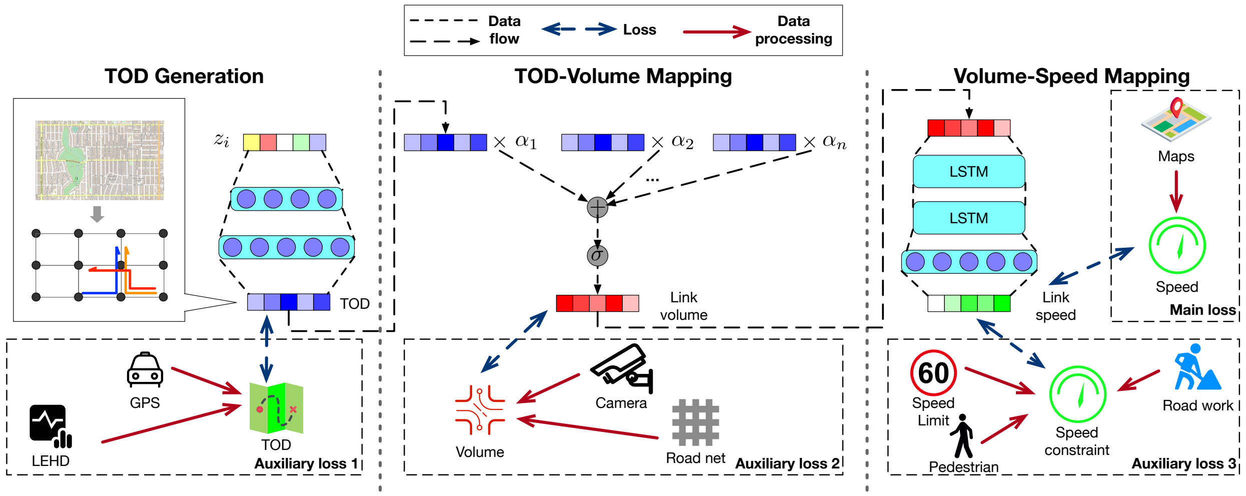


Fig. 3. Framework of OVS.

We formulate the problem as following.

Problem 1: Given N origin-destination (o, d) pairs, the real-time speed observation $\mathcal{V}$ on road links, the goal is to recover the time-origin-destination (TOD) tensor $\mathcal{G}$ and the mapping function $\hat{\mathcal{V}} = h(\mathcal{G})$, so that the reconstruction error is minimized, i.e., minimize $||\hat{\mathcal{V}} - \mathcal{V}||_2$.

Note that, though we use the link speed as the only basic input, we can also add other auxiliary input, such as sparse TOD tensor, volume tensor, census data, and POI data. Either partial view (corresponding to certain regions, link, or trip) or full observation can be used as auxiliary inputs.

IV. Method

A. Model Overview

In this section, we propose a method called OVS (**O**rigin-destination-**V**olume-**S**peed) to reconstruct the traffic system. Our model framework is shown in Figure 3. There are three components in our model, TOD Generation, TOD-Volume Mapping, and Volume-Speed Mapping. (1) The TOD Generation module generates a vector of TOD for one specific OD pair from random seeds. (2) Then, the TOD-Volume Mapping module will model how different TOD interact with each other and determine the link volumes. It takes TOD as input and outputs link volumes. (3) The Volume-Speed Mapping will take the link volume as input and output link speed. We define the main loss based on the gap between the final output link speed and the observed groundtruth link speed. (4) In addition, other data (e.g., GPS, LEHD, camera data and roadnet data) are converted into partial representations of TOD, volume or speed. They are compared against the predicted quantities (TOD, volume and speed) to construct the auxiliary loss. By minimizing a combination of the main loss and auxiliary loss, the model will be able to propagate the loss back to every module to optimize the parameters. We will introduce each module in the following sections.

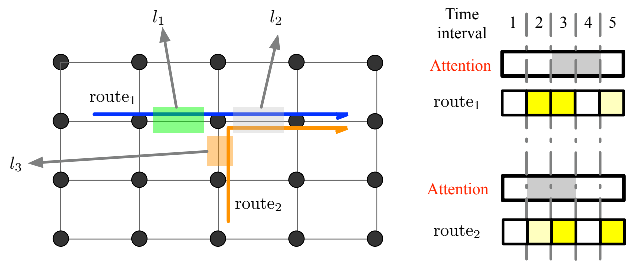


Fig. 4. The mapping relation between routes and link volume. The volume q_2 is affected by route$_1$ and route$_2$. However, this influence might be delayed due to the volume in the upstream links (i.e., q_1 and q_3 correspondingly).

B. TOD Generation

Following the convention in the literature [9], [20], we assume the TOD are generated from Gaussian priors. Following the tradition in deep belief networks (DBN) [31], we use sigmoid function as the activation function. Mathematically, for TOD i, we can compute the trip count g_i^{od} as following from a Gaussian sample input z_i. $\boldsymbol{W}_1^{od}$ and $\boldsymbol{W}_2^{od}$ are weight matrices and $\boldsymbol{b}_1^{od}$ and $\boldsymbol{b}_2^{od}$ are bias vectors.

$$\boldsymbol{h}_i^{od} = Sigmoid(\boldsymbol{W}_1^{od}\boldsymbol{z}_i + \boldsymbol{b}_1^{od}) \tag{1}$$
$$\boldsymbol{g}_i^{od} = Sigmoid(\boldsymbol{W}_2^{od}\boldsymbol{h}_i^{od} + \boldsymbol{b}_2^{od}) \tag{2}$$

C. TOD-Volume Mapping

The second component in our model is trying to describe how TOD generates link volume.

According to the routing policy π of people, one OD i may correspond to several routes. Mathematically, we can estimate the trip count on route k, $\boldsymbol{p}_k$, from the trip count on OD i, $\boldsymbol{g}_i^{od}$ by

$$\boldsymbol{p}_k = Sigmoid(\boldsymbol{W}_{i,j}^{route}\boldsymbol{g}_i^{od} + \boldsymbol{b}_{i,j}^{route}). \tag{3}$$

For the simplicity of illustration, for the following sections, we assume that people will choose the shortest or fastest route

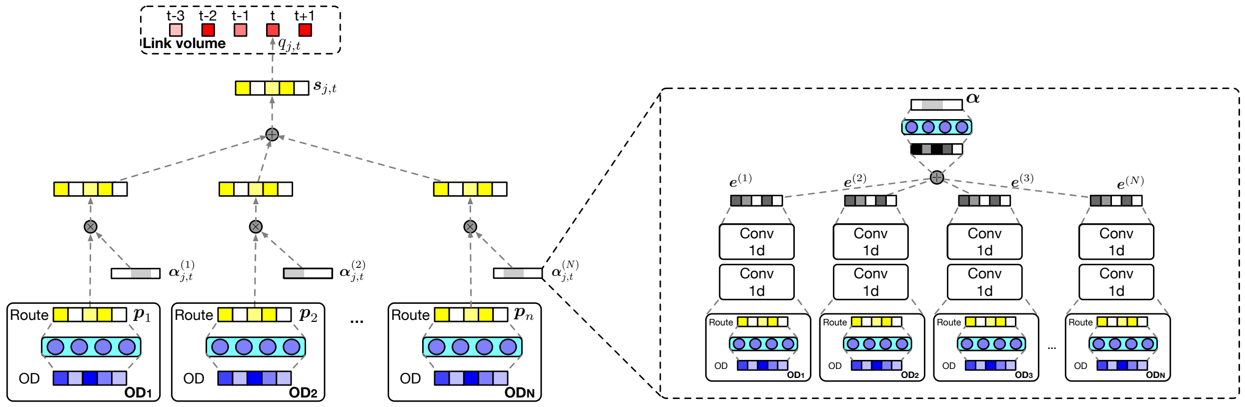


Fig. 5. The dynamic attention network in the TOD-Volume mapping network.

based on real-time traffic conditions. Therefore, one OD will only correspond to one route, and they will share the index i.

1) Dynamic Attention Network Maps Route Trip Count to Link Volume: The mapping relationship between link volume and route trip count is shown in Figure 4. Note that, route$_1$ and route$_2$ both contains link l_2. Hence, the volume $q_{2,t}$ of link l_2 at time t can be affected by the trip counts in route$_1$ and route$_2$ in current and previous time steps, i.e., $p_{1,t}$, $p_{1,t-1}$, ..., $p_{2,t}$, $p_{2,t-1}$, ... This is because the time it take for vehicles to arrive at l_2 from its origin via route$_1$ depends on the traffic congestion condition of the upstream links of l_2 in route$_1$, e.g., l_1. If traffic runs smoothly on l_1, then $q_{2,t}$ may be only affected by the route trip count a few time intervals back, e.g., $p_{1,t}$. If l_1 is congested, $q_{2,t}$ may be related to the route traffic further in the past, e.g., $p_{1,t-3}$.

Here, we propose to use a 2D attention network structure (among temporal dimension and among different routes) (shown in Figure 5) to model how route trip count generates the link volume.

Mathematically, we can model the volume $q_{j,t}$ as

$$q_{j,t} = \sum_{i \in N_j^{(r)}} \boldsymbol{\alpha}_i^{(j,t)} \cdot \boldsymbol{p}_i = \sum_{i \in N_j^{(r)}} \sum_{\tau=1}^{t} \alpha_{i,\tau}^{(j,t)} p_{i,\tau} \tag{4}$$

where $N_j^{(r)}$ is the set of all routes that contain link l_j. $\boldsymbol{\alpha}_i^{(j,t)}$ is a vector and the τ-th element of it is the coefficient on the route trip count $\boldsymbol{p}_i$ at τ-th time step.

Additionally, it is also related to p_2 (route$_2$) departed at time step $2, 3$ because there is heavier traffic on this route, so the traffic is delayed.this sentence need to be organize the structure. In short, the intensity of attention that each link volume should pay on the routes is highly dynamic and should be a function of the traffic situation of several time frames before the current time step. The number of time frames to look back is a hyperparameter.

According to the previous example, the intensity of attention that each link volume should pay on the routes is highly dynamic and should be a function of the traffic situation of several time frames before the current time. In next section, we will introduce how to calculate the dynamic attention $\boldsymbol{\alpha}$.

2) Attention Calculation: We learn an embedding from the whole road network to calculate $\boldsymbol{\alpha}_i^{(j,t)}$., as shown in the right bounding box of Figure 5. To avoid notation cluttering, we omit the index i, j and t in the attention layer.

Based on the intuition that the volume at time t can only be impacted by several time intervals ahead of t, e.g., $t, t-1, t-2, ...$, it is intuitive to apply 1-D convolution on the route trip count representation $\boldsymbol{p}_k$.

$$\boldsymbol{h}_1^c = \text{Conv}(\boldsymbol{p}_k) \tag{5}$$

$$\boldsymbol{e}^{(k)} = \text{Conv}(\boldsymbol{h}_1^c) \tag{6}$$

The convolution layers are configured with 1x3 filters, and stride of 1. Sigmoid activation is applied.

Further, the representations $\boldsymbol{e}^{(k)}$ are aggregated to obtain an overall representation of the system.

$$\boldsymbol{e} = \sum_{k=1}^{N} \boldsymbol{e}^{(k)} \tag{7}$$

Then, we add a fully-connected layer, followed by Softmax layer to get the attention

$$\boldsymbol{\alpha} = \text{Softmax}(\boldsymbol{W}_1^{att} \boldsymbol{e}^{att} + \boldsymbol{b}_1^{att}) \tag{8}$$

where $\boldsymbol{W}_1^{att}$ is a weigh matrix and $\boldsymbol{b}_1^{att}$ is a bias vector. By putting back the index i, j and t, we have obtained the attention $\boldsymbol{\alpha}_i^{(j,t)}$.

D. Volume-Speed Mapping

For the volume to speed mapping, we use LSTM layers to capture the relationship. Specifically, we have

$$\boldsymbol{h}_1^s = \text{LSTM}(\boldsymbol{q}_j) \tag{9}$$

$$\boldsymbol{h}_2^s = \text{LSTM}(\boldsymbol{h}_1^s) \tag{10}$$

$$\boldsymbol{v}_j = \text{FC}(\boldsymbol{h}_2^s) \tag{11}$$

Please note that the LSTM layers and fully connected layers (FC) are shared within all the different links.

As shown in Figure 3, the main loss is defined as

$$l_{main} = \sum_{j=1}^{M} \sum_{t=1}^{T} \|v_{j,t} - \hat{v}_{j,t}\|^2 \tag{12}$$

TABLE II
CATEGORIES OF AUXILIARY DATA.

	Static	Dynamic
TOD	POI, census (LEHD)	taxi trajectory
Volume	road network (# lanes)	surveillance camera
Speed	speed limit	road work, pedestrian

where $\hat{v}_{j,t}$ is the observed speed on link j at time t.

E. Auxiliary Loss

In addition to the speed data, other categories of data may also help infer the traffic, e.g., census data, camera data and road conditions. These data can be categorized as in Table II according to the quantities they may help to infer (TOD, volume and speed) and whether they are dynamic or static. Though they might be sparse in space (e.g., we may only have surveillance camera data for 10 intersections in a city), they can be used as auxiliary data to construct auxiliary loss. For instance, LEHD (Longitudinal Employer-Household Dynamics) data describes the number of people that live in one census unit and work in another unit, and therefore can constraint the TOD in one day as $l_{aux}^1 = \sum_{i=1}^{N} \left\| \sum_{t=1}^{T} g_{i,t} - f_1(\boldsymbol{x}_i) \right\|^2$, where $\boldsymbol{x}_i$ is the auxiliary feature for OD i, $f_1()$ is a function that converts feature x_i to trip counts for OD i. For instance, for LEHD data, $f_1(x_i)$ will represent the summation of all the people transiting following OD i.

Thus, the final overall loss function can be defined as

$$l = l_{main} + w_g \cdot l_g + w_q \cdot l_q + w_v \cdot l_v \tag{13}$$

where l_g, l_q and l_v are all weighted summation of the auxiliary loss introduced by the data in Table II. We will not enumerate all the forms how these loss functions are defined.

In addition, adding these auxiliary loss will help eliminate the unreasonable solutions. We will show this in the experiment part.

V. EXPERIMENT

A. Research Questions to be Validated

In this paper, we conduct experiments on both synthetic and real-world data to answer the following research questions:

- **RQ1**: Compared with baseline methods, how does our proposed method OVS perform in rebuilding the city-wide traffic (recovering TOD)?
- **RQ2**: Can we integrate other data sources to solve the multiple solution issue?
- **RQ3**: Can we avoid the influence of environment factors?
- **RQ4**: Can we provide an explanation of TOD tensors learnt from the real-world speed data?

TABLE III
DATASET INFORMATION.

Dataset	Intersections	# roads	# Trajectories
Hangzhou	46	63	9,656
Porto	70	100	2,576
Manhattan	100	180	1,242,408
State College	14	16	-

B. Datasets

We use both synthetic data and real-world data to conduct experiments. We use a microscopic traffic simulator CityFlow [2] which can simulate the behavior of each vehicle and traffic signal. The simulator takes TOD tensors as its input, simulates the movement of vehicles in single-vehicle level, and outputs the volume and speed tensors on each road.

Synthetic data. We use a road network with 3x3 intersections. The experiment is done on a 2-hour period with each time interval as 10 minutes. We generate five different TOD patterns as follows.

- Random: TOD tensor are generated with random values range from 1 to 20 vehicles/min.
- Increasing: Values in TOD tensor have an initial value of 5 vehicles/min, and will increase by 2 every 10 minutes. An extra random noise is added.
- Decreasing: Values in TOD tensor have an initial value of 20 vehicles/min, and will decrease by 2 every 10 minutes. An extra random noise is added.
- Gaussian: values in TOD tensor follow a Gaussian distribution with mean as 10 vehicles/min and variance as 4.
- Poisson: values in TOD tensor follow a Poisson distribution with rate λ as 3.

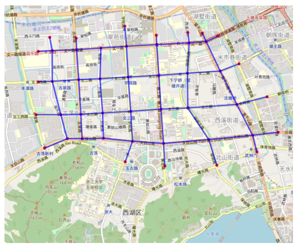
Real-world data. The statistics of the four real datasets **Hangzhou**, **Porto**, **Manhattan**, and **State College** are shown in Table III, and the coverage of the three datasets are illustrated in Figure 6. For Hangzhou, Porto and Manhattan, we collect the taxi trajectory data, scale them with city-specific factor (# all vehicles / # taxi) to represent the trajectories of all vehicles, and get the corresponding TOD tensors. Then, we input them into the simulator to get the speed tensors, which are used as the groundtruth observation. The goal of our methods is to recover the TOD tensors given the groundtruth speed tensors. We further collect the speed data from Google Maps in two cities to conduct case study. The first city Hangzhou is a big commercial city while the second city State College is a college town. As for the roadnet of each dataset, we collect them from OpenStreetMap , which is a well-know open map server. Admittedly, there are many map servers other than Google Map and OpenStreetMap. As long as they can provide the network structure of roads and intersections, and corresponding speed data on each road, we can use any map server in our experiments.

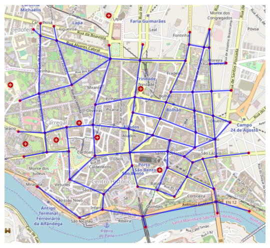
https://www.openstreetmap.org/

TABLE IV

SUMMARY OF NETWORK STRUCTURES. FC STANDS FOR FULLY CONNECTED LAYERS. N_{od} STANDS FOR THE NUMBER OF OD PAIRS. N_{link} IS THE NUMBER OF ROAD LINKS. N_{data} IS THE NUMBER OF DATA SAMPLES. ROUTE-e REPRESENTS ONE OF THE FOUR BRANCHES IN THE RIGHT PART NETWORK IN FIGURE 5. e-α REPRESENTS THE UPPER RIGHT PART OF THE NETWORK IN FIGURE 5, WHICH USES EMBEDDING e TO CALCULATE ATTENTION α.

Network	TOD Generation	TOD-Volume			Volume-Speed
		OD-Route	Route-e	e-α	
Input	(N_{od}, T)	(N_{od}, T)	(N_{od}, T)	(N_{od}, T)	(N_{link}, T)
Hidden layers	FC(16), FC(16)	FC(16)	Conv$_{1x3}$, Conv$_{1x3}$	FC(16)	LSTM(128), LSTM(128), FC(32)
Activation	Sigmoid, Sigmoid	Sigmoid	ReLU, ReLU	ReLU	Sigmoid, Sigmoid, Sigmoid
Output	(N_{od}, T)	(N_{od}, T)	(N_{od}, T)	(N_{od}, T)	(N_{link}, T)


(a) Hangzhou


(b) Porto

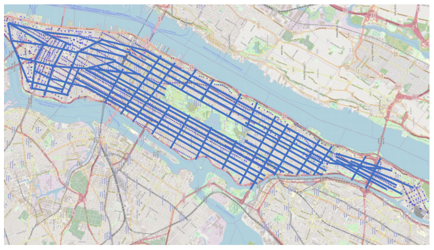

(c) Manhattan

Fig. 6. The areas of the three real datasets.

TABLE V

SUMMARY OF HYPERPARAMETERS

Hyperparameters	Value
Batch size	64
Learning rate	0.001
Dropout rate	0.3
Epoch Number	10000

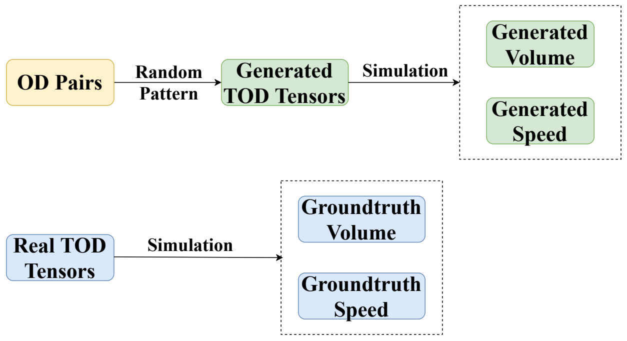


Fig. 7. Our data preprocess procedures. (1) In the training stage, we use the OD pairs of the real TOD tensors to randomly generate more TOD tensors. Then, we run simulation given the generated TOD tensors, to get the generated volume, speed respectively. (2) In the testing stage, we run simulation given the groundtruth TOD tensors, to get the groundtruth volume, speed respectively. For details, please see the following subsection.

C. Implementation

In this section, we will list the network structure and hyperparameters in the implementation of our proposed method OVS. The network structure and size of our model are shown in Table IV. As we mentioned in Section IV, our model contains three parts: TOD Generation, TOD-Volume Mapping, and Volume-Speed Mapping. Given random noise input, TOD Generation will output TOD tensors by two fully connected layers. The role of TOD-Volume Mapping is to turn TOD tensors to the volume tensors on road links. Specifically, there are three sub-module in TOD-Volume Mapping: OD-Route, Route-e, and e-α. OD-Route uses one fully connected layer to generate route from every given OD pair. Route-e uses two convolution layers to analyze the influences on every road links of every OD pair. e-α contains one fully connected layer and one softmax layer to get the corresponding attention values. Volume-Speed Mapping is used to predict speed from the volume on every road link, by two LSTM layers and one fully connected layer. Some of the key hyperparameters are shown in Table V.

D. Data Preprocess

In our experiments, we first process the data before training our model. For each dataset, we can acquire its real TOD tensors and the OD pairs of them. The choice of OD pairs is based on domain knowledge and widely accepted. In the training stage, based on the OD pairs, we randomly generate sufficient TOD tensors. The generation of TOD tensors follows the five different TOD patterns described before, with every 20% of TOD tensors have a specific pattern. We then put the generated TOD tensors into our traffic simulator to get corresponding volume and speed tensors. Now, we have the generated TOD, volume, and speed tensors that in line with the transportation of the roadnet. In the testing stage, we put the real TOD tensors into the simulator, and get the corresponding volume and speed tensors, which is regarded as the groundtruth volume and speed tensors. Figure 7 illustrates our data preprocess. We conduct this generation because the current available real-world TOD and speed data are usually not matched (i.e., the speed data is describing the average

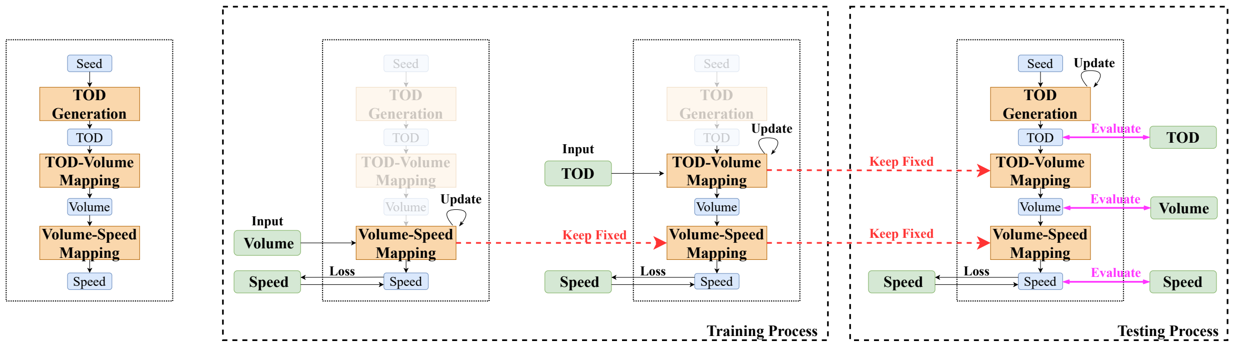


Fig. 8. The pipeline of our experiments. In the training process, we use generated TOD, volume, and speed to train the Volume-Speed Mapping and TOD-Volume Mapping. In the testing process, we first fine-tune our TOD Generation to fit the groundtruth speed data. Then, the results of the well-trained TOD Generation will be the recovered TOD tensors, which are used to evaluate the performance. We also use the volume and speed tensors as metrics. For illustrative purposes, we do not add auxiliary loss here.

speed of all the vehicles on the road, while the TOD only captures the taxi). Therefore, we use the scaled taxi TOD data to generate the groundtruth volume and speed, and hide them in the testing.

E. Pipeline

In this section, we will introduce the training and testing pipeline of our model. First, we train our proposed method OVS with two steps:

1) Feed the generated volume tensors to the neural network of the Volume-Speed Mapping, then use generated speed tensors to calculate loss and optimize the neural networks.
2) Fix the network parameters of the Volume-Speed Mapping, feed the generated TOD tensors to the neural network of the TOD-Volume Mapping, let the result of the TOD-Volume Mapping pass the Volume-Speed Mapping, then use generated speed tensors again to calculate loss and optimize the neural network of the TOD-Volume Mapping.

By the two-step training process, we can get well trained TOD-Volume Mapping and Volume-Speed Mapping. Note that we only use the main loss to update the parameters, which is the hardest case since we do not use any extra data or auxiliary loss. Most importantly, **we do not use any real TOD data in our training process.** Instead, our method only requires sufficient generated data to train.

In the testing process, we fix the network parameters of TOD-Volume Mapping and Volume-Speed Mapping. Then, we feed random seeds to our TOD Generation, and let the results of the TOD Generation pass the TOD-Volume Mapping and Volume-Speed Mapping. Then, we use the groundtruth speed tensors to calculate loss and optimize the parameters of the TOD Generation. After we get a well trained TOD Generation, the results of the TOD Generation will be the recovered TOD tensor we want. Here, real TOD data is only used to calculate metrics for evaluation. The testing process is actually to fine-tune our TOD Generation module to fit the observed speed data.

Above all, we finally succeed in using only real speed data to recover TOD tensors. The training and testing pipelines are illustrated in Figure 8.

F. Compared Methods

- **Gravity**: It is believed that the trip number from one region to another is determined by the census data [7], [8]. Mathematically, the total trip number from region i to j is calculated as $g_{i,j} = k\frac{p_i \cdot p_j}{d_{i,j}^2}$, where p_i, p_j is the population of region i and j, and $d_{i,j}$ is the distance between region i and j. k is tuned by grid search, and kept same across time intervals.
- **Genetic** algorithm [32] searches TOD trip counts that match speed observation best. This method iteratively picks the best several candidates and mutate until convergence.
- Generalized least square (**GLS**) [3]–[6]: These methods assumes a linear assignment matrix that maps TOD to link volume. A neural net is stacked behind to predict the speed.
- **EM** [19], [33]: This method will iteratively update the distribution of TOD and the distribution of the influence from TOD to corresponding road segments speed, and maximize the probability of the observed speed data.
- **NN** [34]: This method uses a neural network to predict the TOD, given the speed data on each road segment. This network contains two fully connected layers.
- **LSTM** [35]: This method regards speed data and TOD as sequential data. It uses two LSTM layers to predict TOD sequences based on speed sequences.

We admit that there are some traffic prediction methods that are better designed than the aforementioned baselines. However, the key idea of all traffic prediction methods is to do prediction by utilizing historical data, such as predicting today's traffic based on yesterday's traffic. Therefore, these methods can not work in our problem settings since we do not use historical data.

In addition, our goal is to illustrate the effectiveness of each module of our framework. Therefore, for each module, we use relatively simple models. Readers are encouraged to try to replace the models in the different modules in order for performance boosting. However, comparing each of the module with their corresponding baselines is off our goal.

TABLE VI
PERFORMANCE COMPARISON W.R.T RMSE (THE LOWER THE BETTER) IN REAL DATASETS. OVS ACHIEVES THE BEST PERFORMANCE. THE LAST ROW "IMPROVE" SHOWS THE RELATIVE IMPROVEMENT OVER THE BEST BASELINE.

	Hangzhou			Porto			Manhattan		
	TOD	vol	speed	TOD	vol	speed	TOD	vol	speed
Gravity	29.87	32.57	1.76	21.45	25.37	**1.30**	20.99	30.44	2.99
Genetic	**23.65**	**25.89**	**1.28**	35.67	35.01	3.04	36.16	31.54	2.81
GLS	27.65	30.50	1.50	19.38	23.74	1.35	38.18	29.57	2.46
EM	25.45	29.23	1.61	18.31	**16.78**	2.41	29.14	37.85	3.74
NN	31.28	39.98	1.85	18.98	26.90	2.04	20.62	22.98	2.18
LSTM	26.76	30.16	1.56	**15.88**	17.03	1.59	**16.17**	**20.52**	**2.09**
OVS	**9.98**	**12.14**	**0.56**	**10.39**	**13.07**	**1.14**	**11.23**	**14.22**	**1.89**
Improve	57.81%	53.09%	55.85%	34.56%	22.14%	12.52%	30.55%	30.72%	9.46%

TABLE VII
RUNNING TIME (IN SECOND) IN REAL DATASETS.

Dataset	Hangzhou	Porto	Manhattan
Time	235.37	433.59	1036.71

G. Metrics

We feed the recovered TOD tensors into the simulator and get the volume and speed tensors. We compute the *RMSE* (root mean squared error) of the recovered TOD, volume and speed against the corresponding groundtruth, which is given by

$$RMSE_{TOD} = \frac{1}{T}\sum_{t=1}^{T}\sqrt{\frac{1}{N}\sum_{i=1}^{N}(g_{i,t} - \hat{g}_{i,t})^2}$$

$$RMSE_{volume} = \frac{1}{T}\sum_{t=1}^{T}\sqrt{\frac{1}{M}\sum_{j=1}^{M}(q_{j,t} - \hat{q}_{j,t})^2}$$

$$RMSE_{speed} = \frac{1}{T}\sum_{t=1}^{T}\sqrt{\frac{1}{M}\sum_{j=1}^{M}(v_{j,t} - \hat{v}_{j,t})^2}$$

where g, q, v and $\hat{g}$, $\hat{q}$, $\hat{v}$ are the predicted and groundtruth TOD, volume and speed correspondingly. The groundtruth TOD are acquired from the dataset, while the groundtruth volume and speed are the result from the traffic simulator given the groundtruth TOD.

H. Performance Comparison (RQ1)

We compare OVS with baselines on both synthetic data and real data (results shown in Table VIII and VI respectively).

We come up with the following observations.(1) Our method OVS is consistently better than other baseline methods. In most of the cases, it is significantly better (more than 30% relative improvement). This is because our learning pipeline captures the generation mechanism of TOD-volume-speed. (2) Gravity performs quite well, probably because Gravity can learn the relation between OD count and population. However, Gravity can not model how the TOD varies over time. The performance of LSTM is also acceptable due to its ability in modeling the temporal change of TOD. However, without the carefully designed network like in OVS, both baseline methods (LSTM and Gravity) can not map speed to TOD accurately.

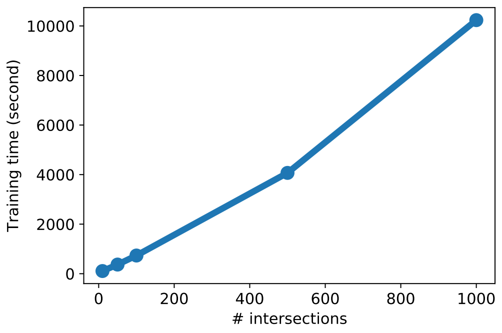


Fig. 9. Running time in synthetic dataset w.r.t. number of intersections. Five data points are with 10, 50, 100, 500 and 1000 intersections.

The ablation studies of OVS is shown in Table IX. In this experiment, we replace each module in OVS with the fully-connected neural network, e.g., "OVS − TOD" denotes the TOD Generation module in OVS is replaced by fully connected layers. We can observe all modules are working better than fully-connected layers in predicting TOD and vol. Since the speed is provided in testing, the speed error is a fitting error, so the slightly higher fitting error on speed of OVS is not a critical drawback.

We further investigate the scalability of our method. Note that, we are showing the training time for our method. The prediction process can be done in real time, because it only needs one fitting of the whole model and takes less than 0.1 second.

In Table VII, we report the running time of our method OVS on three real datasets. The running time is acceptable among all the datasets.

In Figure 9, we show the running time in synthetic environment with 10, 50, 100, 500 and 1000 intersections correspondingly. We can observe that the running time approximately scale linear with the number of intersections.

I. Adding Extra Data Sources (RQ2)

In this experiment, we show how extra data can be incorporated to constrain the solution and filter unreasonable

TABLE VIII
PERFORMANCE COMPARISON W.R.T RMSE (THE LOWER THE BETTER) IN SYNTHETIC DATASETS. OVS ACHIEVES THE BEST PERFORMANCE. THE LAST ROW "IMPROVE" SHOWS THE RELATIVE IMPROVEMENT OVER THE BEST BASELINE.

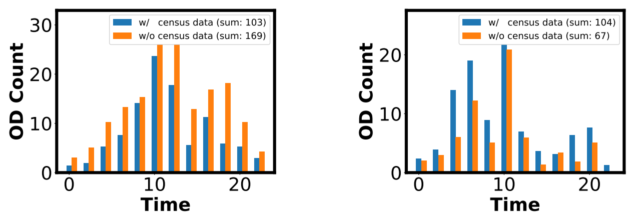
	Random			Increasing			Decreasing			Gaussian			Poisson		
	TOD	vol	speed	TOD	vol	speed	TOD	vol	speed	TOD	vol	speed	TOD	vol	speed
Gravity	**27.26**	**29.41**	**1.82**	**27.31**	**29.00**	**1.71**	**27.38**	**29.25**	1.77	21.27	34.13	**1.07**	**26.95**	**29.14**	**1.59**
Genetic	36.48	35.99	1.93	38.14	38.87	1.88	33.61	35.81	**1.75**	39.08	57.26	1.70	38.81	43.40	1.88
GLS	28.92	32.21	1.97	29.01	32.09	1.82	28.54	31.46	1.86	22.66	33.15	1.43	28.34	31.94	1.70
EM	38.96	40.82	2.58	39.75	42.00	2.48	40.76	42.38	2.68	27.53	36.08	1.77	43.42	45.74	2.30
NN	39.97	46.34	2.65	28.44	31.46	1.80	28.45	31.38	1.86	38.62	45.56	2.36	40.42	47.11	2.61
LSTM	28.51	31.92	1.95	28.45	31.44	1.80	28.47	31.39	1.86	**16.81**	**27.53**	1.24	28.12	31.61	1.70
OVS	**7.83**	**9.17**	**0.68**	**16.87**	**15.53**	**0.85**	**19.11**	**21.35**	**1.29**	**11.81**	**19.52**	**1.03**	**19.72**	**23.55**	**1.17**
Improve	71.3%	68.8%	62.6%	38.2%	46.4%	50.3%	30.2%	27.0%	26.3%	29.7%	29.1%	3.7%	26.8%	19.2%	26.4%

TABLE IX
ABLATION STUDY: "TOD", "TOD2V" AND "V2S" DENOTES TOD GENERATION, TOD-VOLUME MAPPING, AND VOLUME-SPEED MAPPING RESPECTIVELY.

Method	TOD	vol	speed
OVS	7.83	9.17	0.68
OVS − TOD	9.76	10.16	0.69
OVS − TOD2V	11.33	12.78	0.62
OVS − V2S	11.67	15.49	0.59

solutions. We use the roadnet data and generated speed data in Manhattan to conduct two sets of experiments for comparison: (1) with census data as a constraint (2) without census data as a constraint. Here, using census data as a constraint means there will be an auxiliary loss in TOD Generation, as the left-bottom corner of Figure 3 shows. We plot two recovered ODs in two different residential regions with a similar population (census results for both regions are normalized to 100), as Figure 10 shows.

We can see that when we use OVS directly, the sums of two recovered OD counts are a lot different, which is contradicted to the fact that the populations in the regions of these two ODs are similar. However, with the help of census data, OVS can recover TOD tensors with similar total counts, and also remains a similar distribution. Therefore, extra data sources, such as census data can help our model to be more reasonable.



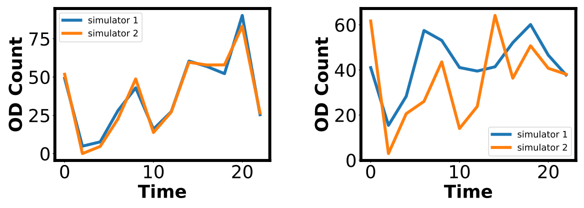
(a) Recovered TOD: Region1 (b)Recovered TOD: Region2

Fig. 10. Two different TOD tensors recovered in two ways: one is directly using OVS, the other is using OVS with census data as a constraint. The desired full-day sum of TOD is 100. It shows that adding census data as a constraint does push the recovered TOD close to the desired sum.

J. Consider Other Traffic Factors (RQ3)

As we all know, the road condition in cities often has some planned changes, such as road constructions. Under these circumstances, TOD tensor will not change drastically, since people still need to travel in the city for work or entertainment. However, the volume-speed mappings on influenced roads will be significantly different from those on other roads. Therefore, given the speed data generated by the same TOD in two scenarios (one is the regular scenario, the other is with road work), the method is expected to recover the same TOD in these two scenarios.

In our experiments, we use two traffic simulators to generate speed data: simulator 1 is the regular simulator while simulator 2 has irregular volume-speed mappings on some roads. The difference between the two simulators is the hypothesis that some roads are under maintenance, occurring traffic accidents, or other special cases. We can see the results in Figure 11. OVS predicts similar TOD tensors in two simulators, while the predictions of LSTM are different between the two simulators. These experiments illustrate that OVS can avoid the interference of road work factors and recover the correct TOD.

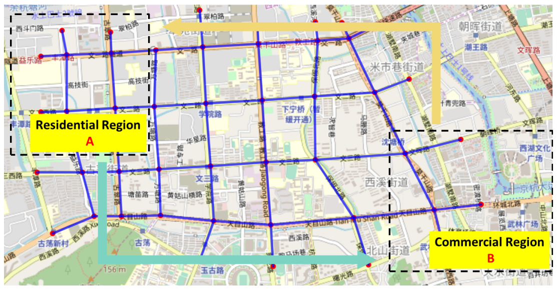


(a) OVS (b) LSTM

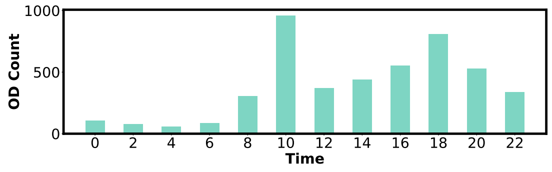
Fig. 11. Comparison of road work factor's influence to recovered TOD tensors. We can see that OVS remains a similar prediction while LSTM is largely affected, after adding road work factor.

K. Case Study of Learned TOD (RQ4)

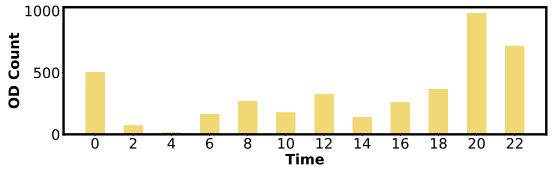
We collect real-world road network data in a big city in China and a university town in the USA, where the speed data of the road segments using Gaode Maps and Google Maps. We run a microscopic traffic simulation on the real-world road network by simulator CityFlow. We take the real-world speed data as input to estimate the TOD in the real world.



(a) Hangzhou Roadnet



(b) Recovered TOD: A $\rightarrow$ B



(c) Recovered TOD: B $\rightarrow$ A

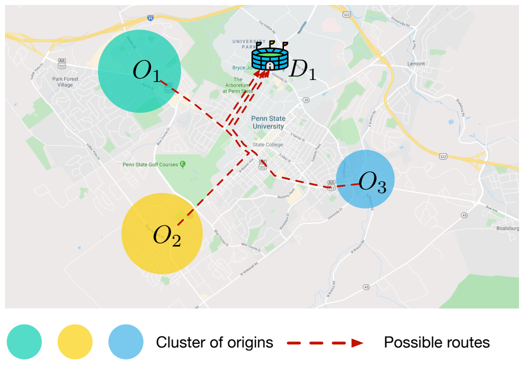
Fig. 12. Case study #1 in Hangzhou, China. (a): We use a roadnet in Hangzhou to conduct experiments, of which we notice two regions: residential region A and commercial region B. (b): We recover the TOD tensor by OVS, and we analyze the OD counts of (A $\rightarrow$ B) and (B $\rightarrow$ A) on a Sunday.

We show two case studies here. The first case reflects people's weekend commute between a residential region and a commercial region. The other case study shows the TOD on Saturday morning before a football game event.

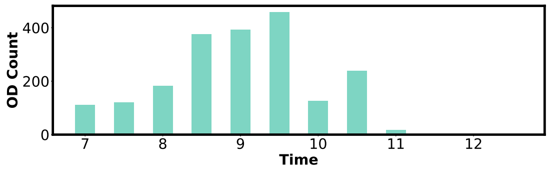
1) Case 1.: In this case, we use the pretrained model in Hangzhou roadnet from Table VI, and let it recover TOD tensors from real-speed data on 01/06/2019, which is a Sunday. We show the performance of different models on fitting the observed speed (result shown in the first column of Table X. Our method gets the lowest RMSE. Note that we can not show quantitative evaluation for volume and TOD because of lacking ground truth data on those variables.

We focus on two typical regions in Hangzhou roadnet: residential region A and commercial region B. We take a close look at the recovered OD counts between these two regions, as Figure 12 shows. We can see that the trips from residential region A to commercial region B have two peaks: one is around 10 am and the other is around 6 pm, which is corresponding to people's shopping habits on a Sunday. The trips from commercial region B to residential region A has one peak from 8 pm to 1 am, which is reasonable since people are used to going home late on weekends.

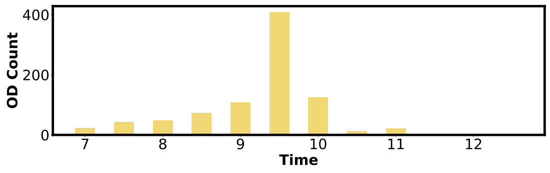
2) Case 2.: In this case, people drive to the stadium area on a Saturday morning before a football game. We first show the performance of different models on fitting the observed speed. We can see the result in the second column of Table



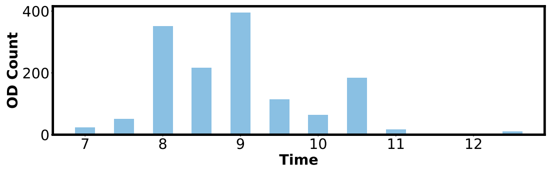
(a) State College Roadnet



(b) Recovered TOD: O1 $\rightarrow$ D1



(c) Recovered TOD: O2 $\rightarrow$ D1



(d) Recovered TOD: O3 $\rightarrow$ D1

Fig. 13. Case study #2 on a football event: (a) shows three OD that indicates people go to the stadium on football day; (b)(c)(d) illustrate the recovered TOD tensors by our model, we can see that most people go to the stadium at 9 am, which is approximately 2 hours before the game.

X. Our model still performs the best in this case.

In this case, the football game starts at noon. As Figure 13 shows, a higher TOD is transiting from north or south residential area to the stadium at around 9 am, which matches the schedule that people usually arrive 2 hours before the game which starts at noon. Besides, many people in other cities will come to the town for the football game. As a result, the number of trips for first and third OD is much larger than the second OD, because O_1 and O_3 are near the main exits of the highway #99 and highway #322 and O_2 is only a local residential area.

VI. Conclusion

In this paper, we propose to solve the problem of revealing underlying temporal origin-destination tensor from speed data. This problem differs from the literature in the following two folds. (1) This problem tries to recover the mechanism by which TOD generates traffic situation, rather than doing TOD

TABLE X
PERFORMANCE OF DIFFERENT MODEL W.R.T $RMSE_{speed}$ IN REAL-WORLD SCENARIOS.

Method	Case 1	Case 2
Gravity	2.81	2.55
Genetic	1.48	1.52
GLS	**1.21**	2.37
EM	1.45	2.08
NN	1.76	1.98
LSTM	1.28	**1.36**
OVS (ours)	**0.58**	**0.32**

prediction from historical TOD trip counts. (2) This problem takes easy-to-get speed data as input (rather than volume data), hence can apply to large-scale urban road networks.

We solve this problem by proposing a method called OVS, in which we model the TOD generation, TOD-Volume mapping, and Volume-Speed mapping in one unified model. This model, especially the dynamic attention network design, enables us to model the dynamic temporal and spatial dependency between link volume and TOD trip counts. Extensive experiments on synthetic datasets and real datasets are conducted. In experiments, we have shown that our proposed method OVS significantly improve the performance over the baseline methods. We also show two case studies to illustrate that the learned TOD patterns match with people's mobility.

In the future, our work can be further extended by better modeling the relation between routes and TOD, when people's routing strategy is considered, which is more challenging.

ACKNOWLEDGMENT

This work was supported in part by NSF awards #1652525, #1618448 and #1639150. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

REFERENCES

[1] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of SUMO - Simulation of Urban MObility," *International Journal On Advances in Systems and Measurements*, vol. 5, no. 3&4, pp. 128–138, December 2012.

[2] H. Zhang, S. Feng, C. Liu, Y. Ding, Y. Zhu, Z. Zhou, W. Zhang, Y. Yu, H. Jin, and Z. Li, "Cityflow: A multi-agent reinforcement learning environment for large scale city traffic scenario," in *The World Wide Web Conference*. ACM, 2019, pp. 3620–3624.

[3] N. L. Nihan and G. A. Davis, "Recursive estimation of origin-destination matrices from input/output counts," *Transportation Research Part B: Methodological*, vol. 21, no. 2, pp. 149–163, 1987.

[4] E. Cascetta, "Estimation of trip matrices from traffic counts and survey data: a generalized least squares estimator," *Transportation Research Part B: Methodological*, vol. 18, no. 4-5, pp. 289–299, 1984.

[5] M. G. Bell, "The estimation of origin-destination matrices by constrained generalised least squares," *Transportation Research Part B: Methodological*, vol. 25, no. 1, pp. 13–22, 1991.

[6] H. Yang, "Heuristic algorithms for the bilevel origin-destination matrix estimation problem," *Transportation Research Part B: Methodological*, vol. 29, no. 4, pp. 231–242, 1995.

[7] F. Calabrese, G. Di Lorenzo, L. Liu, and C. Ratti, "Estimating origin-destination flows using opportunistically collected mobile phone location data from one million users in boston metropolitan area," 2011.

[8] P. J. Jin, M. Cebelak, F. Yang, J. Zhang, C. M. Walton, and B. Ran, "Location-based social networking data: Exploration into use of doubly constrained gravity model for origin–destination estimation," *Transportation Research Record*, vol. 2430, no. 1, pp. 72–82, 2014.

[9] H. D. Sherali and T. Park, "Estimation of dynamic origin–destination trip tables for a general network," *Transportation Research Part B: Methodological*, vol. 35, no. 3, pp. 217–235, 2001.

[10] X. Xiong, K. Ozbay, L. Jin, and C. Feng, "Dynamic origin-destination matrix prediction with line graph neural networks and kalman filter," *arXiv preprint arXiv:1905.00406*, 2019.

[11] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," *arXiv preprint arXiv:1709.04875*, 2017.

[12] S. Bera and K. Rao, "Estimation of origin-destination matrix from traffic counts: the state of the art," 2011.

[13] J. Barros, M. Araujo, and R. J. Rossetti, "Short-term real-time traffic prediction methods: A survey," in *2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. IEEE, 2015, pp. 132–139.

[14] A. M. Nagy and V. Simon, "Survey on traffic prediction in smart cities," *Pervasive and Mobile Computing*, vol. 50, pp. 148–163, 2018.

[15] H. Kim, S. Baek, and Y. Lim, "Origin-destination matrices estimated with a genetic algorithm from link traffic counts," *Transportation Research Record*, vol. 1771, no. 1, pp. 156–163, 2001.

[16] S. Baek, H. Kim, and Y. Lim, "Multiple-vehicle origin–destination matrix estimation from traffic counts using genetic algorithm," *Journal of Transportation Engineering*, vol. 130, no. 3, pp. 339–347, 2004.

[17] O. Al-Battaineh and I. A. Kaysi, "Commodity-based truck origin–destination matrix estimation using input–output data and genetic algorithms," *Transportation research record*, vol. 1923, no. 1, pp. 37–45, 2005.

[18] J. E. Anderson, "A theoretical foundation for the gravity equation," *The American Economic Review*, vol. 69, no. 1, pp. 106–116, 1979.

[19] H. Spiess, "A maximum likelihood model for estimating origin-destination matrices," *Transportation Research Part B: Methodological*, vol. 21, no. 5, pp. 395–412, 1987.

[20] M. Maher, "Inferences on trip matrices from observations on link volumes: a bayesian statistical approach," *Transportation Research Part B: Methodological*, vol. 17, no. 6, pp. 435–447, 1983.

[21] K. Ashok and M. E. Ben-Akiva, "Estimation and prediction of time-dependent origin-destination flows with a stochastic mapping to path flows and link flows," *Transportation science*, vol. 36, no. 2, pp. 184–198, 2002.

[22] J. Park, Y. L. Murphey, R. McGee, J. G. Kristinsson, M. L. Kuang, and A. M. Phillips, "Intelligent trip modeling for the prediction of an origin–destination traveling speed profile," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 3, pp. 1039–1053, 2014.

[23] J. Barcelö, L. Montero, L. Marqués, and C. Carmona, "Travel time forecasting and dynamic origin-destination estimation for freeways based on bluetooth traffic monitoring," *Transportation research record*, vol. 2175, no. 1, pp. 19–27, 2010.

[24] N. Geroliminis and J. Sun, "Properties of a well-defined macroscopic fundamental diagram for urban traffic," *Transportation Research Part B: Methodological*, vol. 45, no. 3, pp. 605–617, 2011.

[25] C. F. Daganzo and N. Geroliminis, "An analytical approximation for the macroscopic fundamental diagram of urban traffic," *Transportation Research Part B: Methodological*, vol. 42, no. 9, pp. 771–781, 2008.

[26] H. Yao, F. Wu, J. Ke, X. Tang, Y. Jia, S. Lu, P. Gong, J. Ye, and Z. Li, "Deep multi-view spatial-temporal network for taxi demand prediction," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[27] Z. Zhao, W. Chen, X. Wu, P. C. Chen, and J. Liu, "Lstm network: a deep learning approach for short-term traffic forecast," *IET Intelligent Transport Systems*, vol. 11, no. 2, pp. 68–75, 2017.

[28] B. L. Smith, B. M. Williams, and R. K. Oswald, "Comparison of parametric and nonparametric models for traffic flow forecasting," *Transportation Research Part C: Emerging Technologies*, vol. 10, no. 4, pp. 303–321, 2002.

[29] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transportation Research Part C: Emerging Technologies*, vol. 54, pp. 187–197, 2015.

[30] Y. Wang, H. Yin, H. Chen, T. Wo, J. Xu, and K. Zheng, "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. ACM, 2019, pp. 1227–1235.

[31] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[32] W. Banzhaf, P. Nordin, R. E. Keller, and F. D. Francone, *Genetic programming: an introduction*. Morgan Kaufmann San Francisco, 1998, vol. 1.

[33] B. Li, “Bayesian inference for origin-destination matrices of transport networks using the em algorithm,” *Technometrics*, vol. 47, no. 4, pp. 399–408, 2005.

[34] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning internal representations by error propagation,” California Univ San Diego La Jolla Inst for Cognitive Science, Tech. Rep., 1985.

[35] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.