2021 IEEE 37th International Conference on Data Engineering (ICDE) | 978-1-7281-9184-3/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICDE51399.2021.00222

TrajForesee: How limited detailed trajectories enhance large-scale sparse information to predict vehicle trajectories?

Kangjia Shao†, Yang Wang †*, Zhengyang Zhou†, Xike Xie†, Guang Wang◇
†University of Science and Technology of China, China, ◇Rutgers University, NJ, US
Email: {angyan*, xkxie}@ustc.edu.cn, {sa517285,zzy0929}@mail.ustc.edu.cn, gw255@cs.rutgers.edu

***Abstract*—Foreseeing detailed vehicle future trajectories collectively enables a large scope of urban applications such as route planning and commercial advertising. Existing methods focused on predicting future trajectories of urban vehicles with their own fine-grained historical trajectories. Unfortunately, in real-world scenarios, fine-grained trajectories provided by GPS are limited to obtain due to privacy issues and business competitions. In this paper, our solution enables the ubiquitous but coarse-grained location-based surveillance information to predict the fine-grained trajectories of all vehicles with limited number of fine-grained trajectories. We first capture the vectorized semantic representation of trajectories by training the spatiotemporal embedding in large coarse trajectory set. Then, we propose a new measurement to calculate the trajectory similarity, which combines the vehicles' historical behavior similarity and short-term trajectory similarity. The obtained trajectory similarity is then seamlessly embedded into the dynamic graph convolution network in the manner of spatial attention. The dynamic graph convolution sequence-to-sequence module and the fully-connected layer are devised to generate final sequential trajectory predictions. The whole process is to train in a multi-task framework. Extensive experiments on real-world datasets show the excellent performance of our method.**

***Index Terms*—Spatiotemporal data, trajectory prediction, urban computing.**

I. Introduction

Foreseeing future trajectories for all types of vehicles in a city is of great significance for intelligent transportation applications, e.g., commercial advertisement pushing [15] and hit-and-run vehicle capturing [18].

Existing studies mostly focus on forecasting future trajectory of an individual vehicle with analyzing its own fine-grained historical trajectories in time series. We can summarize existing works into two categories, *machine learning based methods* [5], [12], [13], [16] and *deep learning based methods* [1], [3], [4], [7], [10]. Machine learning-based methods calculate the possibilities of all possible future trajectories for individual urban vehicles by analyzing their own historical trajectories with probabilistic models and eventually predict the future trajectories for these vehicles. Nevertheless, such predictions are implemented based on long-term data accumulations of the fine-grained historical position information of these predicted vehicles which cannot be well-obtained in most real scenarios. Deep learning-based methods analyze and extract the trajectory patterns of individual urban vehicles in time series by feeding their own historical trajectories into deep neural networks, and then predict future trajectories of individual urban vehicles with corresponding trained neural networks. Without exception, due to the lacking of fully addressing non-linear and interactive influences among trajectories, road networks, and individual driving preferences, these deep learning-based methods can only predict future trajectories of urban vehicles at the same granularity of their own historical trajectories.

In summary, previous works on vehicle future trajectory forecasting assume the pre-deployment of dash-mounted GPS and networking devices, and so as the availability of fine-grained historical position information of predicted vehicles. It is not clear how to take advantage of limited detailed trajectories to enable fine-grained future trajectory predictions with large-scale sparse surveillance information. In this work, to tackle these challenges, we propose a novel multi-task learning framework, TrajForesee, with a novel trajectory similarity measurement inspired by Natural Language Processing (NLP) strategies and a dynamic Graph Convolution Network (GCN) based Sequence2Sequence framework. Specifically, we first generate a semantic word embedding dictionary with for all urban intersections by feeding all fine-grained vehicle trajectories into an NLP model, and with the generated dictionary, we embed all coarse-grained trajectories in both spatial and temporal perspectives. After that, we propose a novel similarity measurement to measure the overall similarities among coarse-grained trajectories with considering both the similarities among trajectories themselves and the behavior similarities among the corresponding vehicles that generate these trajectories. Next, A novel Dynamic GCN based Sequence2Sequence (DGCS2S) framework is proposed to predict fine-grained future trajectories for urban vehicles.

To our best knowledge, this is the first work for fine-grained urban vehicle trajectory prediction by taking full advantages of large-scale sparse position-based information and very limited detailed GPS trajectories with well-designed learnable similarities. The technique proposed in this paper offers a new perspective to make full use of large-scale sparse and low-

✉ Prof. Yang Wang is the corresponding author: angyan@ustc.edu.cn.

978-1-7281-9184-3/21/$31.00 ©2021 IEEE
DOI 10.1109/ICDE51399.2021.00222

quality urban sensing data to benefit a series of spatiotemporal analysis and calculations.

II. Related work

Machine learning based methods: [13] uses a two-years trace of the mobility patterns of over 6,000 users within a campus to evaluate the future trajectory prediction performances of several traditional location predictors, and discovers that low-order Markov predictor performed as well or better than some more complex compression-based predictors with a smaller space-consuming. [16] and [12] utilize different Gaussian mixture models to approximate historical moving patterns of moving objects, and subsequently evaluate the probabilities of different future possible trajectories, then regressively predict the maximum likely future trajectories of moving objects by using Gaussian process. [5] constructs a Bayesian model based spatio-temporal model to approximate historical trajectories and predict future trajectories with considering both the spatial correlations and the temporal periodic patterns of historical trajectories. Coincidentally, this kind of machine learning based methods relies on the assumption of the known of fine-grained historical trajectories of predicted vehicles which cannot well-obtained in real scenario due to privacy issues.

Deep Learning based methods: [7] feeds the sequence of vehicles' coordinates obtained from sensor measurements to the LSTM network to analyze the temporal behaviors of urban vehicles and produces the probabilistic information on the future location of the vehicles to offer good estimations of future trajectories. [10] employs the encoder-decoder architecture which analyzes the pattern underlying in the fine-grained past trajectories of urban vehicles with an LSTM-based encoder and generates several most likely future trajectory candidates of corresponding vehicles. [4] leverages an LSTM network to anticipate the individual vehicles' driving patterns with their own sequential historical trajectories and the factors such as lane division, road structure, traffic lights' status and motion characteristics of vehicle detector real-time perception, and the learned patterns are then used to guide a low-level optimization-based context reasoning process for future trajectory prediction. [1] proposes an LSTM model to learn general human movement and predict their future trajectories based on their own fine-grained past position information. Nevertheless, none of the existing deep learning based methods makes prediction with coarse-grained historical trajectories, thus fall short in forecasting the future trajectories for all urban vehicles with only sparse stationary road surveillance systems.

III. Problem definition

We formally define the basic concepts as well as the problem studied in this work.

*Definition 1 (**Road network**): Urban road network can be formalized as a directed graph $G(\mathcal{V}, \mathcal{E})$ where vertex $v_i \in \mathcal{V}$ corresponds to an urban intersection v_i and edge $e_{ij} \in \mathcal{E}$ denotes the directed road segment from intersection v_i to v_j.*

*Definition 2 (**Coarse-grained trajectory set**): The coarse-grained trajectory set is defined as coarse-grained trajectories of all urban vehicles. This kind of trajectory is captured and generated by the stationary road surveillance systems, and can be denoted as $\mathbb{C} = \{C_1, C_2, \cdots, C_N\}$ where N is the total number of all urban vehicles. Here $C_i = \{c_1^i, \cdots, c_M^i\}$ indicates all historical coarse-grained trajectories of the ith urban vehicle, c_j^i indicates the jth coarse-grained historical trajectory of the ith urban vehicle, and $c_j^i = \{(v_1^{i,j}, t_1^{i,j}), \cdots, (v_K^{i,j}, t_K^{i,j})\}$ where $v_p^{i,j} \in \mathcal{V}_s (p \in \{1, \cdots, K\})$ and $t_p^{i,j}$ indicates the time that the ith urban vehicle goes through intersection $v_p^{i,j}$ in trajectory c_j^i.*

*Definition 3 (**Detailed trajectory set**): The detailed trajectory set is defined as fine-grained trajectories of paritial urban vehicles. This kind of trajectory is captured and uploaded by the dash-mounted GPS and networking devices of these vehicles, and can be denoted as $\mathbb{F} = \{F_1, F_2, \cdots, F_n\}$ where n is the total number of GPS device equipped urban vehicles. Here $F_i = \{f_1^i, \cdots, f_m^i\}$ indicates all historical fine-grained trajectories of the ith GPS device equipped urban vehicle, f_j^i indicates the jth fine-grained historical trajectory of the ith GPS device equipped urban vehicle, and $f_j^i = \{(v_1^{i,j}, t_1^{i,j}), \cdots, (v_k^{i,j}, t_k^{i,j})\}$ where $v_p^{i,j} \in \mathcal{V} (p \in \{1, \cdots, k\})$, $v_p^{i,j}$ and $v_{p+1}^{i,j}$ are directly connected with a road segment, and $t_p^{i,j}$ indicates the time that the ith GPS device equipped urban vehicle goes through intersection $v_p^{i,j}$ in trajectory f_j^i.*

Assuming the i_1th GPS device equipped vehicle is the i_2th urban vehicle, we have $|C_{i_2}| = |F_{i_1}|$ and $c_j^{i_2} \subseteq f_j^{i_1}$. If we denote the coarse-grained trajectory set of all GPS equipped vehicles and the rest GPS free vehicles as $\mathbb{C}_G$ and $\overline{\mathbb{C}_G}$, these two sets should satisfy $\mathbb{C}_G \bigcap \overline{\mathbb{C}_G} = \emptyset$ and $\mathbb{C}_G \bigcup \overline{\mathbb{C}_G} = \mathbb{C}$.

*Definition 4 (**Vehicle trajectory prediction**): Given the coarse-grained historical trajectory set $\mathbb{C}$ of all urban vehicle and the fine-grained trajectory set $\mathbb{F}$ of partial vehicles, our task is to predict fine-grained future q-step trajectory $\widehat{f_*^i} = \{(\widehat{v_1^{i,*}}, \widehat{t_1^{i,*}}), \cdots, (\widehat{v_q^{i,*}}, \widehat{t_q^{i,*}})\}$ for any urban vehicle as accurate as possible.*

IV. Future vehicle trajectory predictions with coarse-grained surveillance information

The overview of the proposed solution is illustrated in Figure 1, which consists of four major parts: i) spatiotemporal embedding for historical trajectories, ii) similarity measurement of trajectories, iii) dynamic GCN based sequence2sequence framework for trajectory prediction, and iv) multi-task learning framework for DGCS2S training.

A. Spatiotemporal embedding for historical trajectories

1) Word embedding with limited fine-grained historical trajectories: We find there exists obvious inverse phenomenon between the appearing frequency and the frequency ranking of individual segments which obeys the Zipf's Law [9] in NLP [1]. So we employ the Skipgram [2] model, which aims

[1]Zipf's law says that, the appearing frequency ranking of a word is inversely proportional to the appearing frequency itself of this word in a natural language corpus.

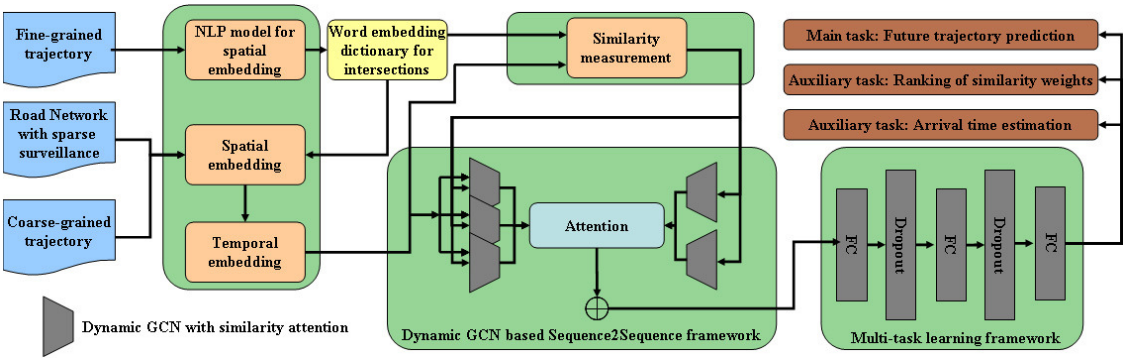


Fig. 1. Solution overview

at achieving context prediction in NLP by generating corresponding vectors for each individual words, to generate a word embedding dictionary for all urban intersections by learning all context features from fine-grained historical trajectories. Specifically, inspired by [11], the main target of utilizing Skipgram here is to find a vector space Φ that maximizes the probability of any intersection appearing in the trajectory context. If we consider the appearance of each individual road segment as an independent event, and given any road segment v_i, the optimization problem can be written as,

$$\min_{\Phi} \{-\log \Pr(\{v_{i-w}, \cdots v_{i+w}\} \setminus v_i | \Phi(v_i))\} \quad (1)$$

where Φ demotes the projection function which maps all road intersections into a word embedding vector, and w is the length of context. With all historical fine-grained trajectories and the above mentioned optimization target, we then train the optimized project function Φ, and we have $\mathcal{S} = \Phi(\mathcal{V})$.

2) Spatial embedding for coarse-grained historical trajectories: With the learned word embedding dictionary $\mathcal{S}$ for all urban intersections, the coarse-grained trajectories also can be embedded as spatial embedded vectors.

3) Temporal embedding for coarse-grained historical trajectories: The coarse-grained trajectories of vehicles can exhibit significant weekly patterns. Considering this kind of weekly and daily patterns in coarse-grained trajectories, we first construct a time grid for each day of the week. Given a node in this grid, the left and right neighbors of this node in a same row correspond to the time points that are one minute before and after the time point of this point respectively, and the above and under neighbors of this node in a same column are the time points that are one week before and after the time point of this point respectively. Intuitively, we can generate 7 different time grids for each day of the week, and for one specific day, we can generate a random number of different routes by selecting a random start and walking a random number of random-direction steps iteratively. Based on all generated routes, we can also project all time points of 7 different days of the week into a word embedding dictionary $\mathcal{R} = \{r_1, \cdots, r_{10080}\}$ where 10080 indicates the number of time points in 7 days [2].

[2]Each time point corresponds to one single minute in real circumstance. For 7 days of the week, the total number of time points is $60 \times 24 \times 7 = 10080$.

B. Trajectory similarity measurement

1) Measurement of vehicle behavior similarity: We here introduce Term Frequency-Inverse Document Frequency (TF-IDF) [17] method to help evaluate the similarity among two vehicles based on their historical coarse-trajectories. Given the ith and jth urban vehicles and their historical coarse-grained trajectories, we first consider all their historical trajectories as two independent corpora and globally calculate p-crucial intersections for these two coarse-trajectories as $cru(i)$ and $cru(j)$ respectively, and then calculate the behavior similarity among two vehicles by

$$\mathrm{Sim}_{veh}(i,j) = \cos\{\Phi(cru(i)), \Phi(cru(j))\} \quad (2)$$

2) Measurement of individual trajectory similarity: Similarly, we also consider each individual trajectory c_α and c_β as two independent corpora, and then the similarity between them can be measured by cosine similarity of their spatiotemporal embedded $\Psi(\Phi(c_\alpha))$ and $\Psi(\Phi(c_\beta))$,

$$\mathrm{Sim}_{traj}(c_\alpha, c_\beta) = \cos\{\Psi(\Phi(c_\alpha)), \Psi(\Phi(c_\beta))\} \quad (3)$$

3) Measurement of overall similarities of urban trajectory: Given αth coarse-grained trajectory of ith vehicle and βth coarse-grained trajectory of jth vehicle, the overall similarity between them can be written as,

$$\mathrm{Sim}_{all}(c_\alpha^i, c_\beta^j) = \gamma \mathrm{Sim}_{veh}(i,j) + \tau \mathrm{Sim}_{traj}(c_\alpha^i, c_\beta^j) \quad (4)$$

where γ and τ are the adjustable weights of both vehicle behavior and individual trajectory similarities.

C. Dynamic GCN based Sequence2Sequence framework for future trajectory prediction

To fully extract all spatial correlations in urban road network, we modify all traditional convolution operations in the Convolution Sequence2-Sequence framework [6] with a novel dynamic GCN. The particular architecture of our DGCS2S is illustrated in Figure 2.

1) Seq2Seq Framework for future trajectory prediction:

Encoder: We first combine a spatiotemporal embedded coarse-grained trajectory with the sequence embedded vectors [3]. Given the jth coarse-grained trajectory of the ith urban

[3]Given the jth coarse-grained trajectory of the ith urban vehicle, $c_j^i = \{(v_1^{i,j}, t_1^{i,j}), \cdots, (v_K^{i,j}, t_K^{i,j})\}$ where $v_p^{i,j} \in \mathcal{V}_s (p \in \{1, \cdots, K\})$, the corresponding sequence embedded vector can be denote as $\mathcal{G} = \{g_1, \cdots, g_K\}$ where g_p is the embedded sequence number of intersection $v_p^{i,j} (p \in \{1, \cdots, K\})$.

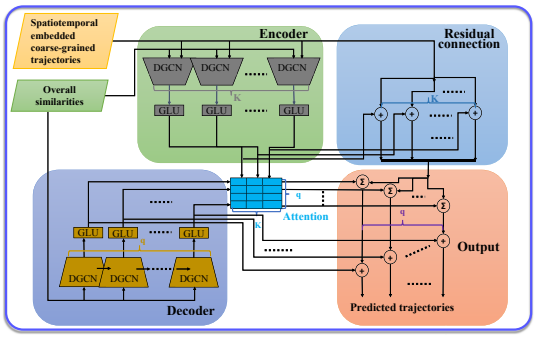


Fig. 2. Architecture of DGCS2S framework

vehicle and its corresponding sequence embedded vecor $\mathcal{G}$, the final embedded vector can be formalized by $\mathcal{Z} = \{((s_1^{i,j} + g_1), r_1^{i,j}), \cdots, ((s_K^{i,j} + g_K), r_K^{i,j})\}$. Next, we utilize K Dynamic GCNs (DGCN) and K Gated Linear Unit (GLU) activation functions to extract features from the inputted sequence $\mathcal{Z}$, and the output of the encoder component can be denoted as $\mathbb{E} = \{\epsilon_1, \cdots, \epsilon_K\}$. Notice that the number of DGCNs and GLUs in the encoder component equals to the number of the former step K in the inputted coarse-grained trajectory.

Decoder: We use the output previous-round-predicted trajectory as the input of the decoder component, and the framework predicts in an iterative manner. For the first round, the input of the decoder component is initialized by a padding of specific symbols. Notice that the number of DGCNs and GLUs in the decoder component equals to the number of the subsequent step q in the predicted future trajectory. The output of the Decoder component can be denoted as $\mathbb{D} = \{d_1, \cdots, d_q\}$.

Attention: To investigate the weight of each element within the inputted sequence by executing dot product for the outputs of both the encoder and decoder components, we generate an attention matrix $\mathcal{A}_{IO} \in R^{q \times K}$, where $a_{ij} = \frac{\exp(d_i \cdot \epsilon_j)}{\sum_{p=1}^{K} \exp(d_i \cdot \epsilon_p)}$.

Output: The output is the predicted future q-step sequence, we only select the first step $v_{q_1}^{i,j}$ in this q-step sequence as the predicted next step, and iteratively predict the future trajectory by using the $\{(v_2^{i,j}, t_2^{i,j}), \cdots, (v_K^{i,j}, t_K^{i,j}), (v_{q_1}^{i,j}, t_{q_1}^{i,j})\}$ as the input.

2) Dynamic GCN: The future trajectory of an urban vehicle is strongly related with its K previous step trajectory, the current intersection and the driving behavior pattern of the corresponding vehicle. For a prediction of an urban vehicle with specific historical coarse-grained trajectory at a specific intersection, we can calculate the transition probabilities for all possible intersections by using the overall similarities and the attention mechanism to map historical coarse-grained trajectories to fine-grained trajectories, and then calculate the dynamic adjacent matrix for DGCN.

Considering the spatiotemporal embedded $\Psi\left(\Phi(c_j^i)\right) = \{(s_1^{i,j}, r_1^{i,j}), \cdots, (s_K^{i,j}, r_K^{i,j})\}$ of the jth coarse-grained trajectory of the ith urban vehicle c_j^i, we filter all trajectories which include intersection $v_K^{i,j}$ from $\mathbb{C}_G$ and denote all selected trajectories as $\mathbb{C}_G^{v_K^{i,j}}$. For each coarse-grained individual trajectory in $\mathbb{C}_G^{v_K^{i,j}}$, we calculate the similarities among this trajectory and all other coarse-grained trajectories in $\overline{\mathbb{C}_G}$, and subsequently calculate the transition probabilities for all neighboring intersections of $v_K^{i,j}$ in each trajectory in $\overline{\mathbb{C}_G}$. We carry out this transition probability updating process for each individual trajectory in $\mathbb{C}_G^{v_K^{i,j}}$, and we then obtain the final transition probabilities for urban road network with regard to a specific vehicle and a specific historical coarse-grained trajectory at intersection $v_K^{i,j}$.

After dynamically updating all corresponding values in $\mathcal{A}$, we then generate the normalized transition probability matrix $\widetilde{TM}$ and $\Theta = I_{|\mathcal{V}|} + \widetilde{TM}$, where $I_{|\mathcal{V}|}$ is the identity matrix in the same dimension of Θ. Finally we calculate the adjacent matirx $\mathcal{L} = P^{-\frac{1}{2}} \Theta P^{-\frac{1}{2}}$ by using a symmetric normalization on Θ, where P is a diagonal matrix, and p_{ii} is the sum of all elements in the ith row of matrix Θ. After finishing a future step and updating the inputted trajectory, we then dynamically update the matrix $\mathcal{L}$ for predicting the subsequent future steps.

D. Multi-task learning framework for trajectory prediction

For task-wise regularization and enhance the overall performance, we settle the trajectory prediction as the main task, and the arrival time as the auxiliary one.

Accuracy loss for trajectory prediction: The trajectory loss for ith urban vehicle can be evaluated by the cross entropy (CE) between its real future trajectory f_*^i and predicted one $\widehat{f_*^i}$ spatiotemporal embedded vectors by,

$$Loss_{traj} = CE(STEmbed(f_*^i), STEmbed(\widehat{f_*^i})) \quad (5)$$

Arrival time loss for trajectory prediction: The arrival time loss can be defined by,

$$Loss_{time} = \sum_{j=1}^{q} \left| \widehat{t_j^{i,*}} - t_j^{i,*} \right| \quad (6)$$

where $\widehat{t_j^{i,*}}$ and $t_j^{i,*}$ are the time that the ith urban vehicle arrives at the jth intersection in the predicted future trajectory and the actual trajectory respectively. The overall loss can be formalized as,

$$Loss = \lambda_1 Loss_{traj} + \lambda_2 Loss_{time} \quad (7)$$

V. Experiments

A. Dataset

We collect the urban traffic-related records during 2017 from traffic adminstrative agencies of SIP and Shenzhen. For training and testing splits, we use the dataset during January and February for training, and datasets in March for testing. For evaluation, these records are divided into two perspectives in our experiments as follows.

- **Fine-grained trajectory**. We utilize the GPS records of running taxis as the fine-grained trajectory. In SIP and Shenzhen, these records only include the GPS information of 4,367 and 8,572 taxicabs with the average sampling rate of 20 seconds, respectively.
- **Coarse-grained trajectory**. We obtain traffic surveillance records of all camera-equipped intersections in SIP and

Shenzhen as the coarse-grained trajectory data. This kind of information is more comprehensive and reliable with all-type vehicles incorporated. We also divide taxis in our datasets into two parts: taxis with fine-grained trajectories known, and taxis with fine-grained trajectories unknown.

B. Baseline

Task-related baselines:

(1) TR-TP: This an alternative solution which combines a trajectory recovery method Sharededge [19] and our DGCS2S component in TrajForesee.

(2) LSTM-ED: An LSTM-based encoder-decoder model proposed in [10].

(3) SimTrack-CNN: A sequence to sequence prediction which is based on CNN proposed in [6].

(4) GMTP: A prediction method based on Gaussian mixture model with excellent performance in [12].

Ablative variants baselines:

(1) SimTrack-GCN: We replace the convolution method DGCN in DGCS2S model with a plain GCN proposed in [8] to test whether DGCN is more suitable for the current problem.

(2) OtherSim-DGCN: The trajectory measurement in TrajForese is replaced with the linear weighted similarity in spatial and temporal perspectives [14]. It is designed to verify the effectiveness of our vehicle trajectory similarity measurement.

C. Hyperparameter setting

The dimensions of temporal embedding and spatial embedding are 15 and 25, respectively. In LSTM-ED, the hidden dimensions of LSTM for encoder and decoder are both 256. The convolution layers of encoder and decoder in SimTrack-CNN are both 5, and the width of convolution kernel is set as 3. In the similarity measurement module, we fix the number of crucial intersections p as 20. We fine-tune these hyperparameters carefully but omit here for space limitation.

D. Performance metrics

Assuming the ith urban vehicle real future trajectory is $f_*^i = \{(v_1^{i,*}, t_1^{i,*}), \cdots, (v_q^{i,*}, t_q^{i,*})\}$ and the predicted one is $\widehat{f_*^i} = \{(\widehat{v_1^{i,*}}, \widehat{t_1^{i,*}}), \cdots, (\widehat{v_q^{i,*}}, \widehat{t_q^{i,*}})\}$, the accuracy of future trajectory prediction can be evaluated by the metrics of Accuracy by Number (AN) and Accuracy by Length (AL),

$$\mathrm{AN}(f_*^i, \widehat{f_*^i}) = \frac{\sum_{j=1}^{q} \sigma_j}{q} \tag{8}$$

$$\mathrm{AL}(f_*^i, \widehat{f_*^i}) = \frac{\sum_{j=0}^{q-1} \sigma_j \sigma_{j+1} \mathrm{length}(e_{j(j+1)})}{\sum_{j=1}^{q-1} \mathrm{length}(e_{j(j+1)})} \tag{9}$$

where σ_j equals 1 if $v_j^{i,*} = \widehat{v_j^{i,*}}$, otherwise it is 0; $\mathrm{length}(e_{j(j+1)})$ represents the length of road segment $e_{j(j+1)}$.

E. Experimental result

Comparison results on metrics of both AN and AL are illustrated in Table I-IV. Overall, we observe that our framework can consistently outperform both task-related baselines and ablative variant methods on all time steps. Our work targets bridging the gap between the coarse-grained and fine-grained trajectories for future prediction by taking advantage of the similarity patterns among coarse-grained trajectories.

AN: The AN values of TrajForesee on SIP dataset in Table I are 9.4%-12.9% and 7%-7.9% higher than that of SimTrack-CNN and SimTrack-GCN, respectively. It reveals that our integrated framework can perform better than original CNN and GCN. The AN values of TrajForesee are 2.6% - 6.7% higher than those of OtherSim-DGCN in SIP, which demonstrates our similarity measurement is superior to others'.

TABLE I
AN UNDER DIFFERENT PREDICTION STEPS IN SIP

Prediction steps	1	3	5	10	20
TR-TP	0.62	0.45	0.21	0.130	0.02
LSTM-ED	0.746	0.54	0.414	0.282	0.199
SimTrack-CNN	0.814	0.606	0.506	0.334	0.275
GMTP	0.712	0.503	0.388	0.251	0.144
SimTrack-GCN	0.838	0.649	0.539	0.366	0.304
TrajForesee	**0.908**	**0.718**	**0.611**	**0.463**	**0.375**
otherSim-DGCN	0.841	0.673	0.545	0.435	0.349

TABLE II
AN UNDER DIFFERENT PREDICTION STEPS IN SHENZHEN

Prediction steps	1	3	5	10	20
TR-TP	0.657	0.469	0.291	0.144	0.09
LSTM-ED	0.752	0.577	0.473	0.291	0.203
SimTrack-CNN	0.824	0.641	0.574	0.391	0.288
GMTP	0.734	0.512	0.391	0.283	0.169
SimTrack-GCN	0.840	0.701	0.577	0.401	0.359
TrajForesee	**0.913**	**0.744**	**0.620**	**0.482**	**0.399**
otherSim-DGCN	0.859	0.688	0.560	0.445	0.350

AL: Compared with the best task-related basline, our method obtains an increase of 9.4%-14.9% in SIP and 8.9%-16.4% in Shenzhen on AL metric, and it also significantly outperforms its own ablative variants from Table III and IV.

TABLE III
AL UNDER DIFFERENT PREDICTION STEPS IN SIP

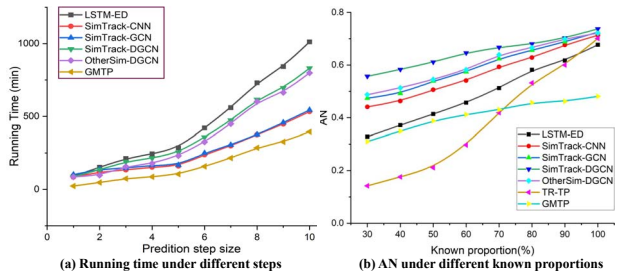
Prediction steps	1	3	5	10	20
TR-TP	0.62	0.404	0.168	0.063	0.008
LSTM-ED	0.746	0.484	0.360	0.213	0.131
SimTrack-CNN	0.814	0.561	0.465	0.277	0.221
GMTP	0.712	0.463	0.346	0.199	0.113
SimTrack-GCN	0.838	0.613	0.507	0.318	0.248
TrajForesee	**0.908**	**0.692**	**0.589**	**0.426**	**0.341**
OtherSim-DGCN	0.841	0.657	0.523	0.389	0.315

Running time: The models based on convolutions run faster than RNN-based LSTM-ED in Figure 3(a) on SIP dataset. This is because the convolution based framework can parallelly extract input features, while the RNN-based model operates serially. With the increase of prediction steps, the difference

TABLE IV
AL UNDER DIFFERENT PREDICTION STEPS IN SHENZHEN

Prediction steps	1	3	5	10	20
TR-TP	0.657	0.435	0.192	0.093	0.021
LSTM-ED	0.752	0.497	0.377	0.221	0.149
SimTrack-CNN	0.824	0.579	0.469	0.285	0.250
GMTP	0.734	0.471	0.359	0.201	0.129
SimTrack-GCN	0.840	0.637	0.512	0.343	0.260
TrajForesee	**0.913**	**0.717**	**0.601**	**0.449**	**0.381**
OtherSim-DGCN	0.859	0.676	0.540	0.406	0.342

between the two types of models is increasing, this is because convolution-based models have one more processes to calculate the trajectory similarity and LSTM-ED uses the serial mode to extract features, leading to a linear time complexity with the increase of prediction steps. Moreover, SimTrack-CNN (GCN) is more effective due to dynamic GCN for extracting similarity every round.



(a) Running time under different steps (b) AN under different known proportions

Fig. 3. Tests of running time and robustness

Robustness: We also test the robustness of our TrajForesee with regard to different proportions of involved known fine-grained trajectory vehicles. It can be seen that the AN values of TrajForesee and GMTP do not fluctuate much while other baselines vary greatly with the growth of the known part proportion in Figure 3 (b), indicating that it is very important to comprehensively understand the historical trajectory of the vehicle for predicting the future trajectory.

VI. CONCLUSION

In this paper, our solutions enable large-scale sparse traffic information to forecast fine-grained vehicle trajectories with limited detailed trajectories. Specifically, we first propose a spatiotemporal trajectory embedding technique to learn high-quality trajectory mapping functions to refine coarse-grained trajectory. Then we design a novel overall similarity measurement to simultaneously extract the similarities of both long-term behavior and short-term individual patterns from ubiquitous coarse-grained trajectories. With similarity measuring, we propose a novel encoder-decoder structure DGCS2S to forecast following fine-grained trajectories, where the embedded DGCN dynamically learns the attention scores between future fine-grained trajectories and embedded coarse-grained trajectories. Finally, we introduce some auxiliary tasks to regularize our learning process and further boost the performance of the main task. Extensive experiments on two real-world datasets demonstrate our TrajForesee can exactly outperform all baselines and improve the efficiency of trajectory forecasting tasks.

VII. ACKNOWLEDGEMENTS

This paper is partially supported by the Anhui Science Foundation for Distinguished Young Scholars (No.1908085J24), NSFC (No.61672487, No.61772492, No. 62072427, No.62072428), Jiangsu Natural Science Foundation (No.BK20171240, BK20191193) and CAS Pioneer Hundred Talents Program.

REFERENCES

[1] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, and S. Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
[2] G. C. Cheng W and W. M. From n-gram to skipgram to concgram. *International journal of corpus linguistics*, 11(4):411–433, 2006.
[3] N. Deo and M. M. Trivedi. [ieee 2018 ieee intelligent vehicles symposium (iv) - changshu (2018.6.26-2018.6.30)] 2018 ieee intelligent vehicles symposium (iv) - multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms. pages 1179–1184.
[4] W. Ding and S. Shen. Online vehicle trajectory prediction using policy anticipation network and optimization-based context reasoning. *arXiv preprint arXiv:1903.00847*, 2019.
[5] H. Gao, J. Tang, and H. Liu. Mobile location prediction in spatio-temporal context. 2012.
[6] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1243–1252. JMLR. org, 2017.
[7] B. D. Kim, C. M. Kang, S. H. Lee, H. Chae, J. Kim, C. C. Chung, and J. W. Choi. Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network. 2017.
[8] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.
[9] Newman and MEJ. Power laws, pareto distributions and zipfs law. *Contemporary Physics*, 46(5):323–351.
[10] S. H. Park, B. Kim, C. M. Kang, C. C. Chung, and J. W. Choi. Sequence-to-sequence prediction of vehicle trajectory via lstm encoder-decoder architecture. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1672–1678. IEEE, 2018.
[11] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.
[12] S. J. Qiao, K. Jin, N. Han, C. J. Tang, Gesangduoji, and L. A. Gutierrez. Trajectory prediction algorithm based on gaussian mixture model. *Journal of Software*, 2015.
[13] L. Song, D. Kotz, R. Jain, and X. He. Evaluating location predictors with extensive wi-fi mobility data. In *IEEE INFOCOM 2004*, 2004.
[14] E. Tiakas and D. Rafailidis. Scalable trajectory similarity search based on locations in spatial networks. In *Model and Data Engineering*, pages 213–224. Springer, 2015.
[15] C. Wang, J. Li, Y. He, K. Xiao, and H. Zhang. Destination prediction-based scheduling algorithms for message delivery in iovs. *IEEE Access*, 8:14965–14976, 2020.
[16] J. Wiest, M. Hoffken, U. Kresel, and K. Dietmayer. Probabilistic trajectory prediction with gaussian mixture models. In *Intelligent Vehicles Symposium*, 2012.
[17] H. C. Wu, R. W. P. Luk, K. F. Wong, and K. L. Kwok. Interpreting tf-idf term weights as making relevance decisions. *Acm Transactions on Information Systems*, 26(3):1–37.
[18] W. Yang, W. Chen, Z. Wei, H. He, and H. Liu. Tracking hit-and-run vehicle with sparse video surveillance cameras and mobile taxicabs. In *2017 IEEE International Conference on Data Mining (ICDM)*, 2017.
[19] Y. Yang, F. Zhang, and D. Zhang. Sharededge: Gps-free fine-grained travel time estimation in state-level highway systems. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2(1):1–26, 2018.