AttnTUL：Trajectory-User Linking via Hierarchical Spatio-Temporal Attention Networks

Anonymous

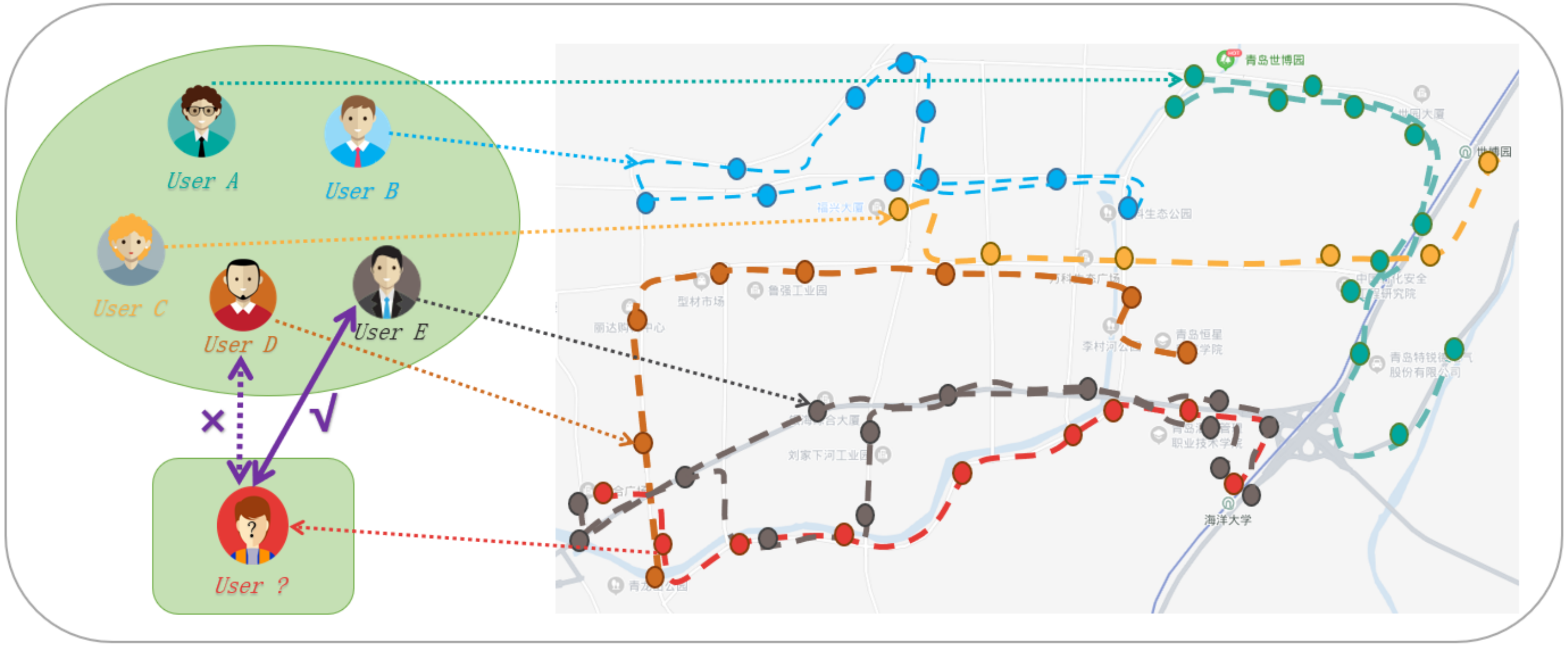
Anonymous

- **Outline**
  - **Introduction**
  - **Related work**
  - **Challenges**
  - **Preliminaries**
  - **Solution**
  - **Experiment Results**
  - **Summary**

Introduction

- **Define of Trajectory-User Linking (TUL)**

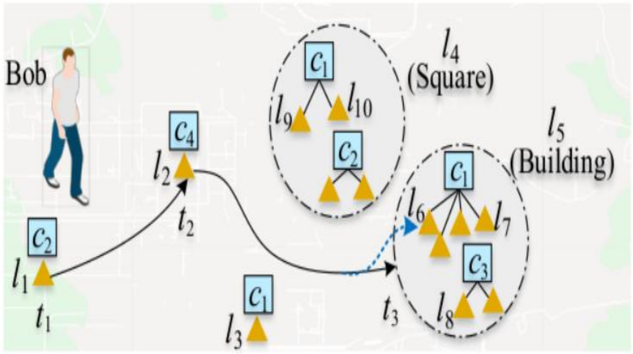
Linking the trajectory to the user who generated it

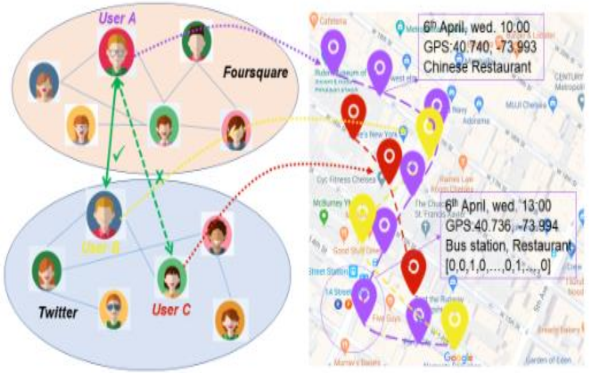


Describe of problem

Introduction

- **Application of Trajectory-User Linking**
  - **Understand movement intentions,** accurate and personalized travel recommendation [1]
  - **Improve traffic safety,** develop intelligent transportation systems [2]
  - **Linking users cross-platform,** obtain better business intelligence [3]





[1] Chen et al. “Curriculum Meta-Learning for Next POI Recommendation”. SIGKDD (2021).
[2] Wang et al. "Spatiotemporal Representation Learning for Driving Behavior Analysis: A Joint Perspective of Peer and Temporal Dependencies." (TKDE)(2019).
[3] Feng et al. “ DPLink: User Identity Linkage via Deep Neural Network From Heterogeneous Mobility Data “ WWW(2018)

Related work

- **Traditional models:**
  - **Similarity measures: DTW, LCSS[4], STLCD, SR[5]**
  - **Sequence modeling and machine learning method: MC, HMM, LDA[6], SVM， DT[7]**

  Curse of dimension / Large amount of calculation / Poor robustness …

- **Deep neural network models:**
  - **RNN and its variant: TULER[8](Bi-TULER, TULER-L, TULER-G)**
  - **Various methods based on RNN: TULVAE[9], TULSN, TULAR, DeepTUL[10], DPLink[3]**
  - **Trajectory representation learning method: t2vec, NeuTraj, T3S[11]**

  Limited mining information / Dense sequence forgetting / Focus more on trajectory similarity …

[4] Ying et al. "Mining user similarity from semantic trajectories." SIGSPATIAL (2010).
[5] Jin et al. “Trajectory-Based Spatiotemporal Entity Linking”. TKDE(2020).
[6] Hamid et al. “Spectral–spatial feature extraction using orthogonal linear discriminant analysis for classification of hyperspectral data.” European Journal of Remote Sensing (2017).
[7] Jiang et al. “A survey on spatial prediction methods.” TKDE(2018).
[8] Gao et al. "Identifying Human Mobility via Trajectory Embeddings. " IJCAI (2017)
[9] Gao et al. "Trajectory-User Linking via Variational AutoEncoder. " IJCAI (2018)
[10] Miao et al. "Trajectory-User Linking with Attentive Recurrent Network. " AAMAS (2020)
[11] Yang et al. “T3S: Effective Representation Learning for Trajectory Similarity Computation.” TKDE(2021).

Challenges

- **Data sparsity :**
    - **All existing method do not work well for low-sampling trajectories.**
- **Fail to capture long-term dependencieses[9] :**
    - **The complicated long-term sequence transformation law is easy to forget.**
- **Ignoring the global modelings :**
    - **All existing works focus on the local spatial sequence modeling of trajectory data.**
- **Fail to utilize rich contextual featurese :**
    - **previous approaches only utilize spatial feature and/or temporal feature.**

	Dataset type	Track length	Mining mode
Classical	LBSNs	Short	Sequential / Spatial
RNN-based	LBSNs	Short / Medium	Sequential / Spatial / Temporal
This work	LBSNs / LBS	Short / Medium / Long	Sequential / Temporal Multi-scale Spatial / State

Table 1: Advantages compared with existing methods

[9] Khandelwal et al. “Far Away: How Neural Language Models Use Context”. ACL (2018).

Preliminaries

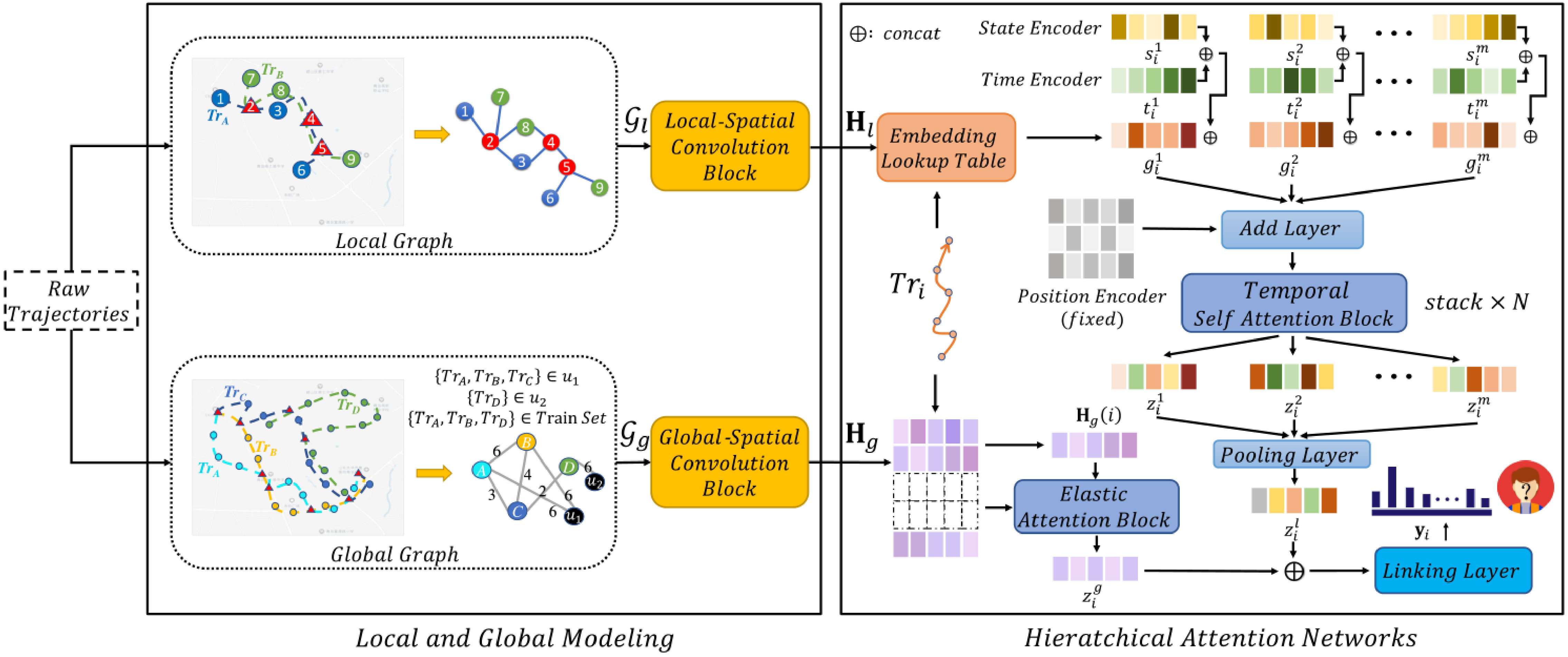
- **Definition**
  - **Definition 1: Spatio-Temporal Point**
    - $< t, l >$
  - **Definition 2: Trajectory**
    - $Tr_{u_{\mathrm{i}}}^{\tau} = (< t_1, l_1 >, < t_2, l_2 >, \ldots, < t_n, l_n >)$
  - **Definition 3: Linked trajectories**
    - $Tr_{u}^{\tau} = \{Tr_{u_{\mathrm{i}}}^{\tau} \mid u_i \in \mathcal{U} \wedge \tau \in T\}$

Preliminaries

- **Problem formulation**
  - **Given:**
    - unlinked trajectories $\overline{Tr}$
    - linked trajectories Tr_u
    - the set of user $\mathcal{U}$
  - **Goal:**

    provide a mapping function $f: \overline{Tr} \rightarrow \mathcal{U}$ that links the unlinked trajectory to users

Solution



The overview of the proposed framework

Solution

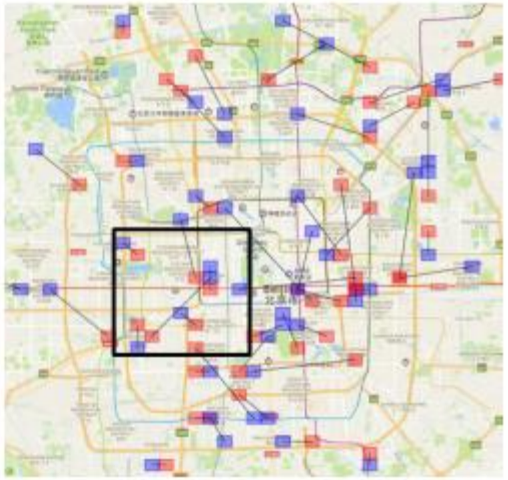
- **Local and Global Graph Modeling**
  - **Step1: Preprocessing**

Algorithm 1: Trajectories to Grids

Input:
The set of trajectory: **T**
The set area: $Lon_{min}, Lon_{max}, Lat_{min}, Lon_{max}$
The size of grid: d
Output:
Grid sequence corresponding to trajectory **T**: **L**

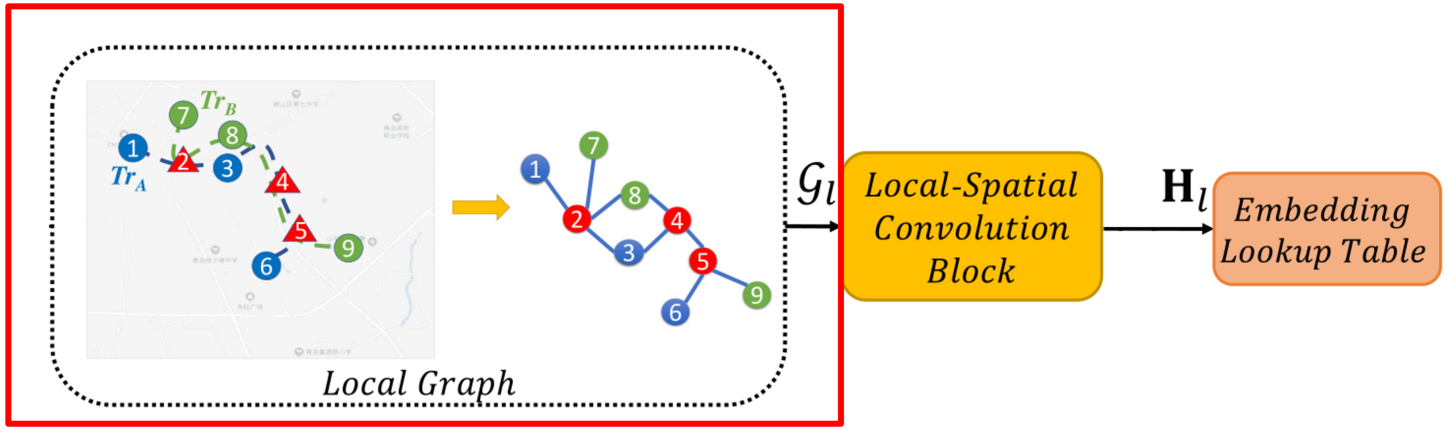
```
// Compute the distance of area by Haversine formula
length = Haversine(Lon_min, Lat_min, Lon_max, Lat_min)
width = Haversine(Lon_min, Lat_min, Lon_min, Lat_max)
// Compute the number of grids in longitude/latitude
N_lon = length / d
N_lat = width / d
// Compute the gap of a grid in longitude/latitude
g_lon = (Lon_max − Lon_min) / N_lon
g_lat = (Lat_max − Lat_min) / N_lat
for < lon, lat > in T do
    id = (lat − Lat_min) * N_lon / g_lat + (lon − Lon_min) / g_lon
    add id to list V
end for
List V drops duplicates and sorts to get a new list V_new
for v in V do
    find the index idx of v in the list V_new
    add idx to list L
end for
return L
```

grid mapping function
$f_g : l_i \rightarrow g_i$



Solution

- **Local and Global Graph Modeling**
  - **Step2: Local Spatial Graph Construction**



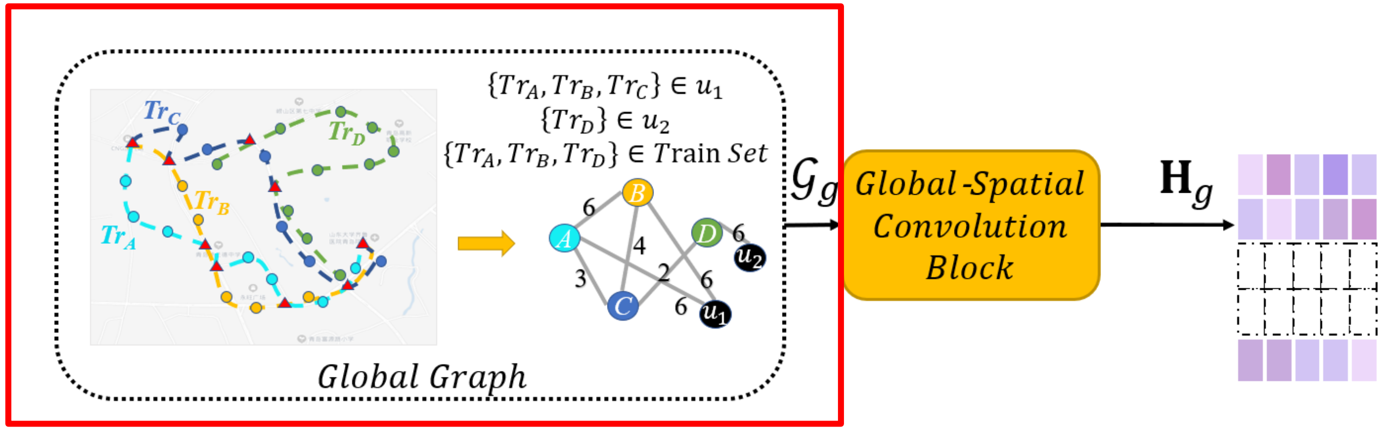
we first construct a local spatial graph $\mathcal{G}_l = (\mathcal{V}_l, \mathcal{E}_l)$

- grid is a node in $\mathcal{V}_l$
- $e_{i,j} \in \mathcal{E}_l$ is defined as the number of trajectories that contains the consecutive snippet

we use $\mathbf{A}_l$ and $\boldsymbol{X}_l$ to denote the adjacency matrix and feature matrix of local spatial graph, respectively.

Solution

- **Local and Global Graph Modeling**
  - **Step3: Global Spatial Graph Construction**



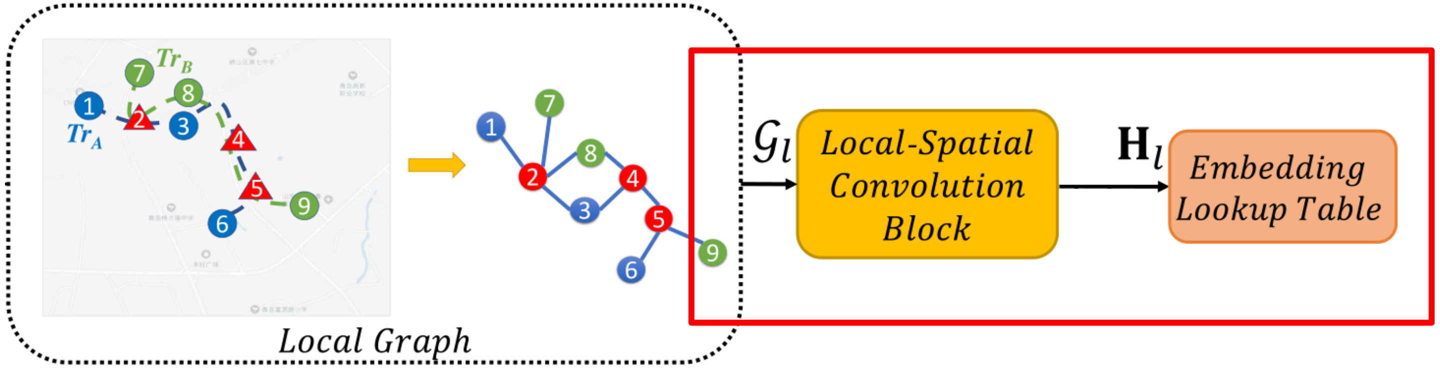
we next construct a global spatial graph $\mathcal{G}_g = (\mathcal{V}_g, \mathcal{E}_g)$

- each trajectory and user are treated as a node in $\mathcal{V}_g$
- $e_{i,j} \in \mathcal{E}_l$ is defined as shared grids in two trajectories/maximum weight between trajectory nodes

we use $\mathbf{A}_g$ and $\boldsymbol{X}_g$ to denote the adjacency matrix and feature matrix of global spatial graph, respectively.

Solution

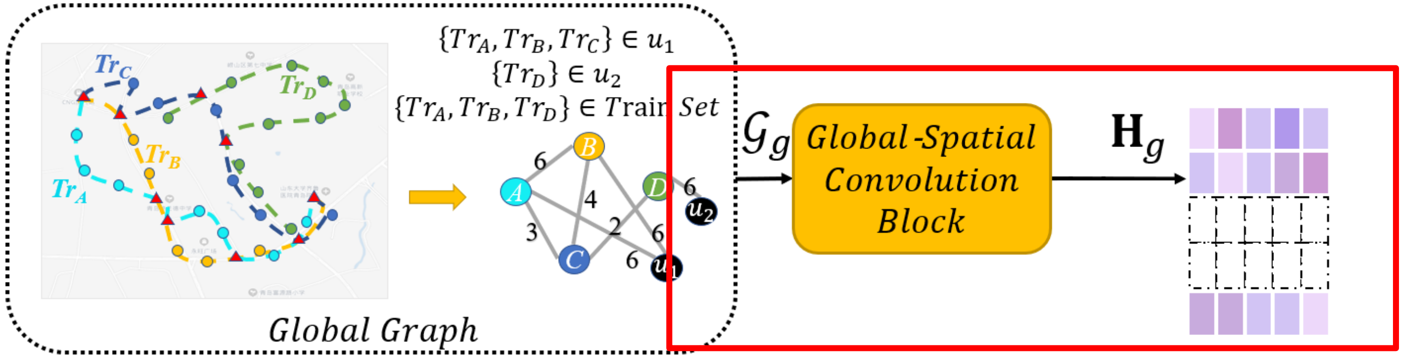
- **Spatial Graph Convolutional Networks**
  - **Step1: Local Graph Convolution**



- $\mathbf{H}_l^{(i+1)} = \mathrm{ReLU}(\tilde{\mathbf{D}}_l^{-\frac{1}{2}}\tilde{\mathbf{A}}_l\tilde{\mathbf{D}}_l^{-\frac{1}{2}}\mathbf{H}_l^{(i)}\mathbf{W}_l^{(i)})$
  - $\tilde{\mathbf{A}}_l = \mathbf{A}_l + I$
  - $\tilde{\mathbf{D}}_{l_{ii}} = \sum_j \tilde{\mathbf{A}}_{l_{ij}}$
  - $\mathbf{H}_l^{(0)} = \mathbf{X}_l$ —— *one hot encode*

Solution

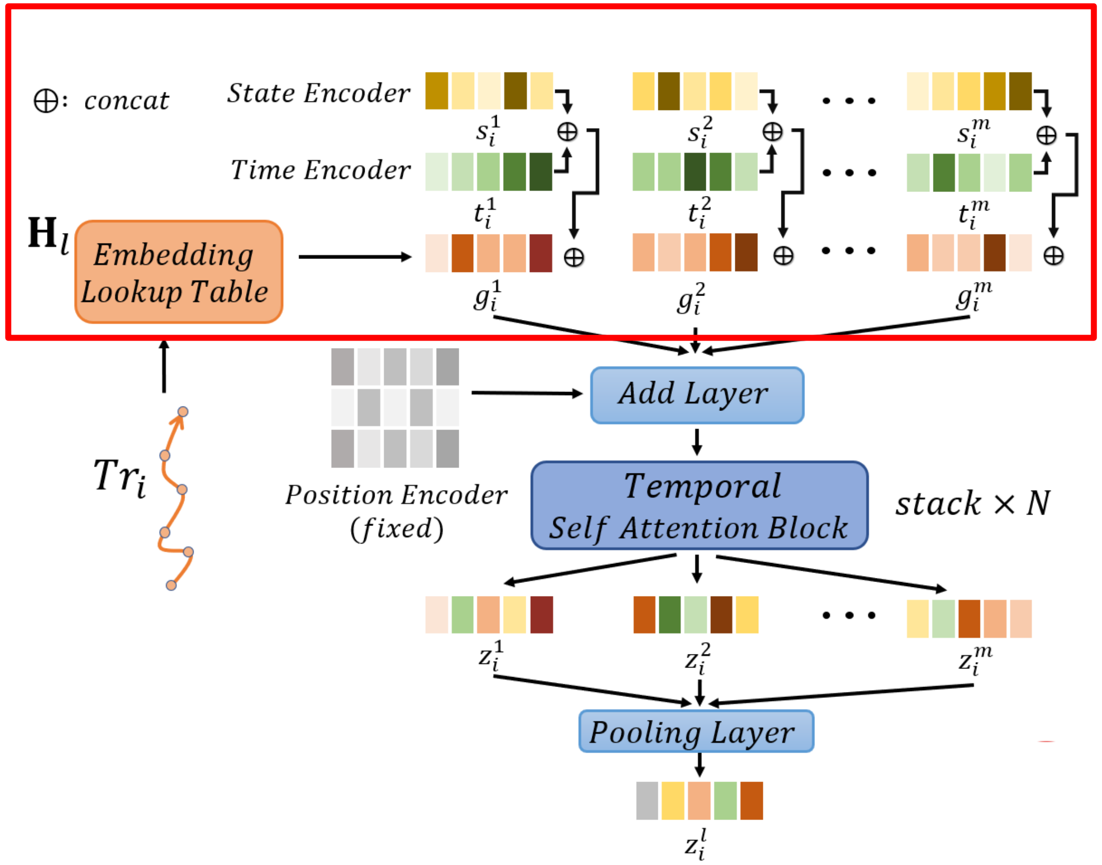
- **Spatial Graph Convolutional Networks**
  - **Step2: Global Graph Convolution**

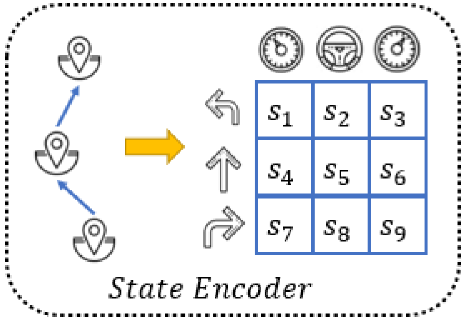


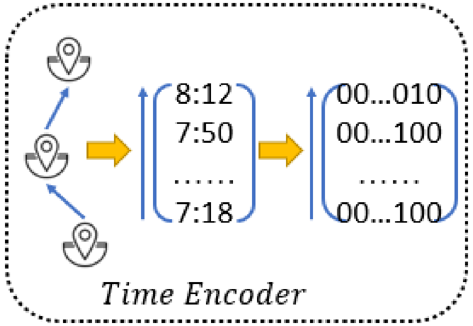
- $\mathbf{H}_g^{(i+1)} = \mathrm{ReLU}(\tilde{\mathbf{D}}_g^{-\frac{1}{2}} \tilde{\mathbf{A}}_g \tilde{\mathbf{D}}_g^{-\frac{1}{2}} \mathbf{H}_g^{(i)} \mathbf{W}_g^{(i)})$
  - $\tilde{\mathbf{A}}_g = \mathbf{A}_g + I$
  - $\tilde{\mathbf{D}}_{g_{ii}} = \sum_j \tilde{\mathbf{A}}_{g_{ij}}$
  - $\mathbf{H}_g^{(0)} = \mathbf{X}_g$ —— *multi hot encode*

Solution

- **Hierarchical Spatio-Temporal Attention Networks**
  - **Step1: Location Encoder**





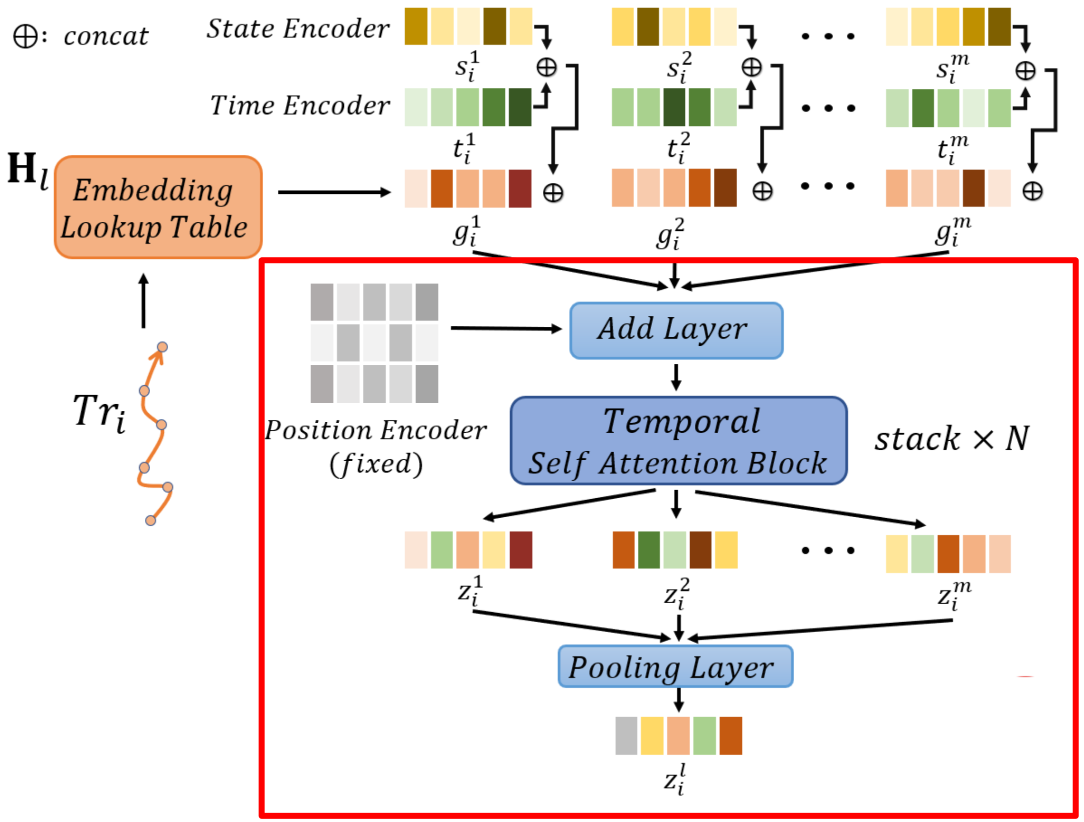


- **Location representation:**

$$x_i = \mathrm{Tanh}(FC([\mathbf{W}_t t_i + b_t; \mathbf{W}_s s_i + b_s; \mathbf{H}_l(g_i)]))$$

Solution

- **Hierarchical Spatio-Temporal Attention Networks**
  - **Step2: Temporal Self-Attention Encoder**



- **Add Layer:**

$$\mathbf{M}_i = \mathbf{X}_i + \mathbf{P}$$

$$\mathbf{P}_{pos,j} = \begin{cases} \sin(pos/10000^{j/d}) & if\ j\%2 = 0 \\ \cos(pos/10000^{j-1/d}) & else \end{cases}$$

- **Multi Head Self-Attention:**

$$(\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i)^\top = \mathbf{M}_i(\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V)^\top$$

$$\mathbf{Z}_i = \mathrm{Softmax}(\frac{\mathbf{Q}_i\mathbf{K}_i^\top}{\sqrt{d}})\mathbf{V}_i$$

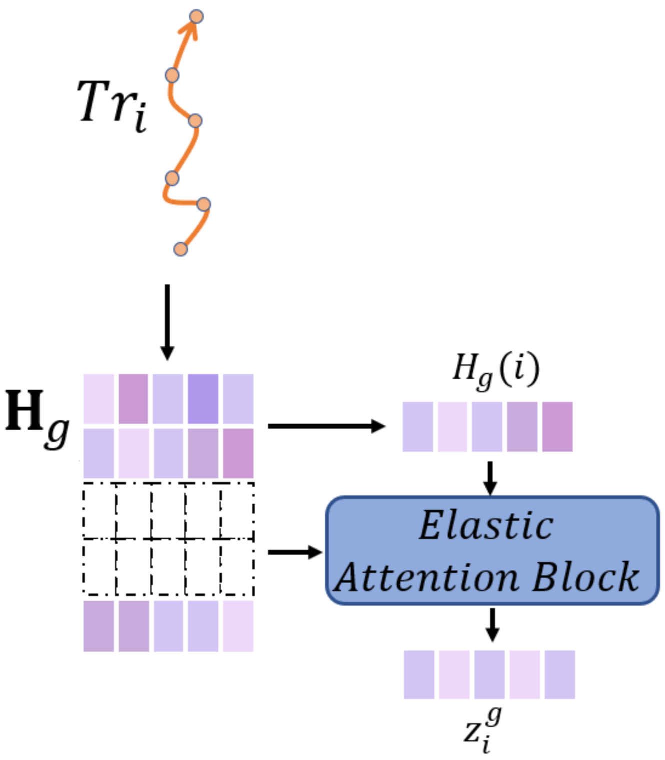
$$\mathbf{Z}_i = FC(\mathrm{concat}(\mathbf{Z}_i^{(1)}, \mathbf{Z}_i^{(2)}, \dots, \mathbf{Z}_i^{(\#head)}))$$

- **Pooling Layer:**

$$z_i^l = \mathrm{Pooling}(z_i^1, z_i^2, \dots, z_i^m)$$

Solution

- **Hierarchical Spatio-Temporal Attention Networks**
  - **Step3: Global Elastic Attention Encoder**



- **Score calculation:**

$$a_{i,j}^s = \frac{\mathbf{H}_g(i)\mathbf{H}_g(j)^\top}{\| \mathbf{H}_g(i) \| \| \mathbf{H}_g(j)^\top \|}$$

$$\mathbf{A}_i^s = \{a_{i,1}^s, a_{i,2}^s, \ldots, a_{i,\|Tr\|}^s\}$$

- **Weight calculation:**

$$\mathbf{W}_i^g = \text{Sparsemax}(\mathbf{A}_i^s)$$

  - **Simple:**

$$SparseMax(x_i) = \begin{cases} \frac{e^{s_i}}{\sum_{j\in\Omega_k} e^{s_j}}, i \in \Omega_k \\ 0, i \notin \Omega_k \end{cases} \qquad \Omega_k = top\_k(s_1, s_2, .., s_n)$$

  - **Complex:**

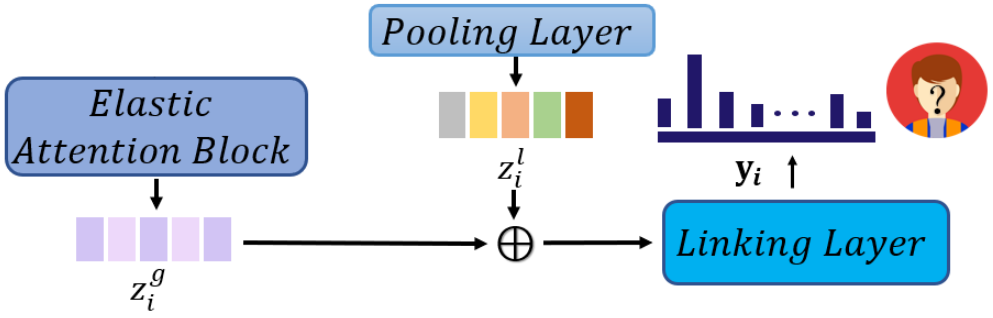
$$\text{Sparsemax}(\boldsymbol{x}) = \underset{\boldsymbol{p}\in\Delta^d}{\text{argmin}} \| \boldsymbol{p} - \boldsymbol{x} \|^2 \quad \Delta^d = \{p \in \mathbb{R}^d : p \geq 0 \| p \|_1 = 1\}$$

- **Weight summation :**

$$z_i^g = \mathbf{W}_i^g \mathbf{H}_g$$

Solution

- **Linking Layer**
  - **hierarchical information fusion and classification**



- **Information Fusion:**

$$\mathbf{y}_i = \left(W_c \begin{bmatrix} z_i^l \\ z_i^g \end{bmatrix} + b_c \right)$$

- **Object function: (negative log-likelihood+L2 regularization):**

$$\mathcal{L}(\Theta) = -\frac{1}{ll} \sum_{i=1}^{ll} c_i \log(\sigma(\mathbf{y}_i)) + \frac{\lambda}{2} \parallel \Theta \parallel^2$$

Experiment Result

- **Dataset:**

Table 1: Statistics of the datasets. $\overline{len}$: average length of trajectories, H: hour.

Datasets	#users	#traj.	#points	#POIs	τ	$\overline{len}$
Gowalla	109	6,242	23,501	8,400	6H	4
	83	5,102	19,794	7,610	6H	4
PrivateCar	71	4,178	135,085	10,493	1H	32
	42	2,797	87,719	7,514	1H	31
GeoLife	90	6,035	945,971	23,369	3H	157
	56	4,064	768,533	19,384	3H	189

- **Metric:**

$$ACC@K = \frac{\left|\{Tr_i \in \overline{Tr}: u^*(Tr_i) \in \mathcal{U}_K(Tr_i)\}\right|}{\left|\overline{Tr}\right|}$$

$$Macro\_P = \frac{1}{n}\sum_{1}^{n} P_i$$

$$Macro\_R = \frac{1}{n}\sum_{1}^{n} R_i$$

$$Macro\_F1 = \frac{1}{n}\sum_{1}^{n} \frac{2 \times P_i \times R_i}{P_i + R_i}$$

Experiment Result

- **Baseline:**
  - **Traditional methods:**
    - **LCSS** [4] **, SR** [5]
    - **LDA**[6]**, DT** [7]
  - **Deep neural network models:**
    - **TULER-L / TULER-G / Bi-TULER**[8]
    - **TULVAE**[9] **, DeepTUL**[10] **, DPLink**[3]
    - **T3S** [11]

Experiment Result

- **Performance comparison with deep neural networks:**

Table 2: Performance comparison with deep neural network models on three real-world datasets.

Dataset	Methods	ACC@1	ACC@5	Macro-P	Macro-R	Macro-F1	ACC@1	ACC@5	Macro-P	Macro-R	Macro-F1
		$\lvert\mathcal{U}\rvert = 83$					$\lvert\mathcal{U}\rvert = 109$				
Gowalla	TULER-L	39.15%	58.77%	29.83%	30.08%	27.47%	37.36%	57.86%	31.10%	29.94%	27.88%
	TULER-G	41.27%	60.76%	36.15%	32.74%	31.16%	39.54%	59.13%	35.04%	31.55%	30.47%
	Bi-TULER	42.29%	61.14%	38.91%	34.14%	33.50%	41.92%	60.56%	37.76%	33.97%	31.77%
	TULVAE	42.86%	62.99%	38.76%	35.53%	34.76%	42.44%	60.07%	39.49%	34.27%	35.06%
	DeepTUL	43.05%	63.17%	36.37%	38.31%	36.24%	42.13%	61.78%	37.66%	38.08%	37.24%
	DPLink	44.53%	64.32%	41.03%	38.60%	37.49%	45.26%	63.71%	39.98%	38.03%	37.18%
	T3S	45.30%	64.76%	43.83%	39.25%	39.22%	45.44%	63.45%	44.82%	39.98%	40.00%
	AttnTUL	**52.75%**	**70.16%**	**54.42%**	**48.98%**	**48.45%**	**50.61%**	**68.53%**	**49.67%**	**46.26%**	**46.12%**
		$\lvert\mathcal{U}\rvert = 42$					$\lvert\mathcal{U}\rvert = 71$				
PrivateCar	TULER-L	21.92%	44.90%	19.41%	21.05%	18.42%	15.43%	32.12%	15.20%	12.47%	12.16%
	TULER-G	21.92%	45.94%	19.35%	19.61%	18.01%	16.13%	32.77%	15.96%	14.37%	13.29%
	Bi-TULER	22.46%	47.79%	21.52%	21.54%	19.76%	16.57%	36.62%	15.76%	15.14%	13.80%
	TULVAE	23.98%	50.44%	25.34%	20.34%	20.45%	16.89%	31.81%	16.92%	15.38%	14.59%
	DeepTUL	22.99%	49.71%	22.95%	22.53%	20.16%	17.65%	33.09%	15.68%	16.97%	14.24%
	DPLink	23.47%	47.12%	25.36%	23.32%	22.23%	20.71%	41.79%	19.14%	16.83%	15.27%
	T3S	26.53%	47.90%	31.30%	25.07%	25.59%	21.28%	39.45%	24.13%	20.05%	20.01%
	AttnTUL	**35.11%**	**60.40%**	**33.24%**	**32.80%**	**31.49%**	**31.25%**	**54.74%**	**32.25%**	**32.24%**	**30.21%**
		$\lvert\mathcal{U}\rvert = 56$					$\lvert\mathcal{U}\rvert = 90$				
GeoLife	TULER-L	41.79%	71.81%	33.78%	34.94%	31.70%	36.84%	60.28%	32.91%	30.41%	29.32%
	TULER-G	43.93%	70.08%	37.09%	36.50%	33.33%	35.51%	61.24%	33.88%	31.73%	30.25%
	Bi-TULER	44.50%	74.09%	38.14%	35.82%	33.76%	37.99%	61.85%	35.82%	33.16%	32.12%
	TULVAE	46.04%	70.99%	42.32%	39.73%	36.87%	39.27%	64.30%	36.23%	33.83%	32.31%
	DeepTUL	51.23%	79.19%	45.84%	41.82%	39.36%	44.92%	69.93%	38.19%	38.35%	35.99%
	DPLink	53.80%	80.03%	48.23%	46.63%	45.03%	47.83%	73.95%	43.93%	40.80%	39.63%
	T3S	51.10%	77.40%	45.47%	43.84%	43.13%	44.33%	70.65%	40.18%	39.52%	38.18%
	AttnTUL	**61.37%**	**85.87%**	**59.20%**	**59.59%**	**58.53%**	**53.92%**	**79.47%**	**49.94%**	**50.29%**	**48.24%**

- **STAN achieves the best performance in terms of all metrics on three different categories of trajectories, especially in dense datasets.**
- **STAN achieves average gains of 25.73% ACC@1 and 40.75% Macro-F1 score in comparison to the best performed baseline across all datasets. This performance improvement is significant.**

Experiment Result

- **Performance comparison with classical models:**

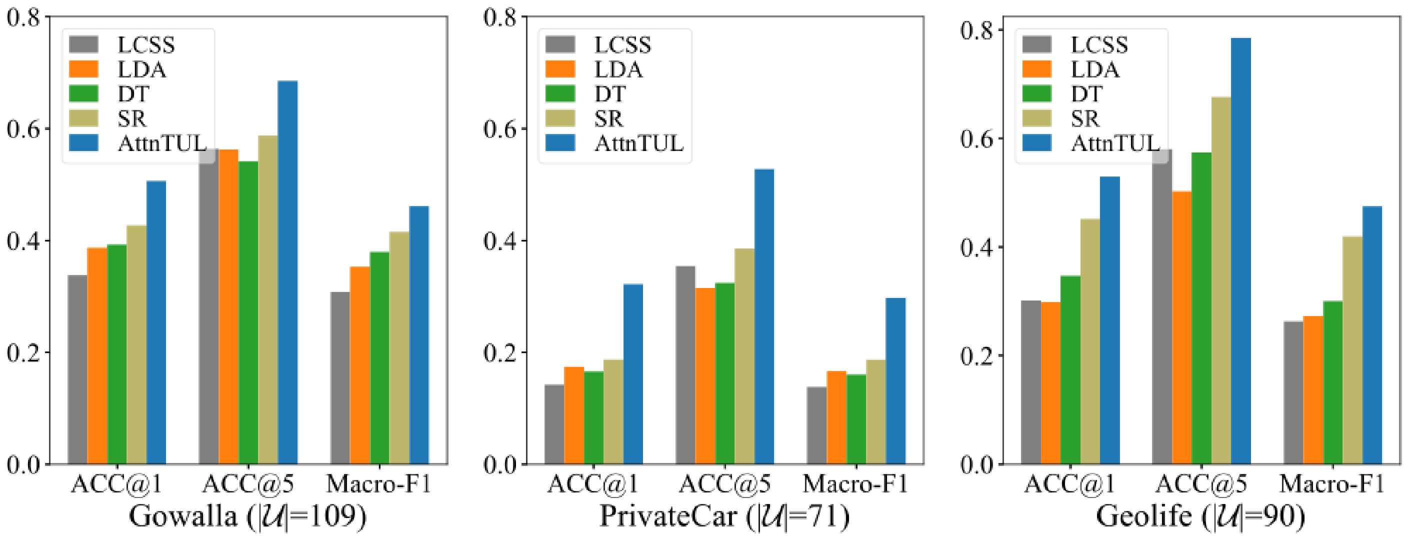


Figure 2: Performance comparison with classic models.

- **STAN presents significant performance improvement over classic methods**

Experiment Result

- **Ablation experiment:**

Table 3: Experimental results of ablation study.

Data	Methods	ACC@1	ACC@5	Mac-P	Mac-R	Mac-F1
Gowalla ($\|\mathcal{U}\|$=109)	TUL-L	50.37%	67.71%	47.58%	42.40%	42.14%
	TUL-G	45.37%	62.58%	43.74%	40.00%	39.24%
	TUL-SA	50.22%	67.38%	47.59%	41.57%	41.02%
	TUL-EA	44.39%	61.77%	43.04%	37.42%	37.08%
	TUL-TS	**51.45%**	68.45%	**50.70%**	**46.42%**	**46.50%**
	AttnTUL	50.61%	**68.53%**	49.67%	46.26%	46.12%
PrivateCar ($\|\mathcal{U}\|$=71)	TUL-L	19.83%	44.74%	18.26%	20.86%	16.01%
	TUL-G	30.09%	53.14%	27.50%	31.05%	27.16%
	TUL-SA	28.31%	53.92%	27.40%	27.67%	24.97%
	TUL-EA	28.49%	52.12%	28.36%	24.54%	23.64%
	TUL-TS	30.58%	53.20%	26.96%	31.52%	27.41%
	AttnTUL	**31.25%**	**54.74%**	**32.25%**	**32.24%**	**30.21%**
GeoLife ($\|\mathcal{U}\|$=90)	TUL-L	49.51%	78.43%	45.49%	45.48%	43.04%
	TUL-G	48.97%	75.46%	43.82%	43.61%	42.03%
	TUL-SA	51.06%	78.65%	47.01%	47.15%	44.01%
	TUL-EA	46.38%	74.35%	41.98%	40.56%	38.59%
	TUL-TS	52.07%	79.25%	47.58%	47.41%	45.02%
	AttnTUL	**53.92%**	**79.47%**	**49.94%**	**50.29%**	**48.24%**

- **our key components all contribute to performance improvement of the proposed model.**

- **Compared to STAN, STAN-L and STAN-G perform much worse demonstrating the importance of both local and global modelling in our model.**

- **Compared to STAN-L, STAN-SA achieves better performance on PrivateCar and GeoLife, indicating that our designed temporal self-attention block could capture long-term dependencies for long sequence trajectories.**

- **Compared to STAN-G, STAN-EA producing worse results demonstrates the crucial role of elastic attention of global module in extracting relevant global representation**

Experiment Result

- **Model robustness:**

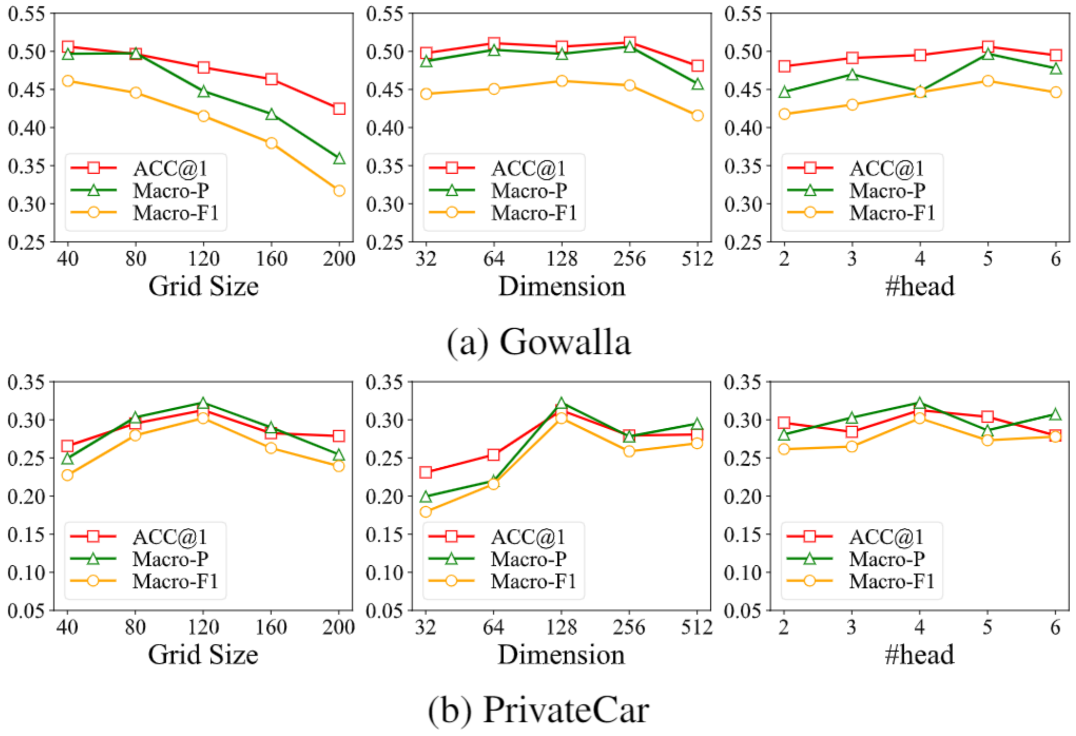


Figure 3: Parameter sensitivity *w.r.t.* s, d and $\#head$.

- If the grid is too large, the spatial information would be lost, which is reflected more obvious on sparse Gowalla than on dense PrivateCar. If the grid is too small, some spatial interaction among trajectories would not be reflected, which affects the prediction performance.

- Gowalla is very sparse, and thus few grids are involved. The lower embedding dimensionality has little effect on it, while PrivateCar is denser. When the dimensionality is very low, it has a greater impact.

Summary

- **We propose AttnTUL**
  - Our AttnTUL simultaneously model both local and globalspatial characteristics of users' mobility trajectories to realize high predictability.
- **We design a hierarchical attention network**
  - a temporal self-attention encoder to learn the local sequential dependencies in intra-trajectory, anda global elastic attention encoder to capture the complex correlations among inter-trajectory.
- **We conduct extensive experiments**
  - Results show that our model significantly outperforms state-of-the-art baselines by 11.82%~50.89% and 21.73%~97.84% improvements interms of ACC@1 and Macro-F1.

Thanks for watching!

Anonymous