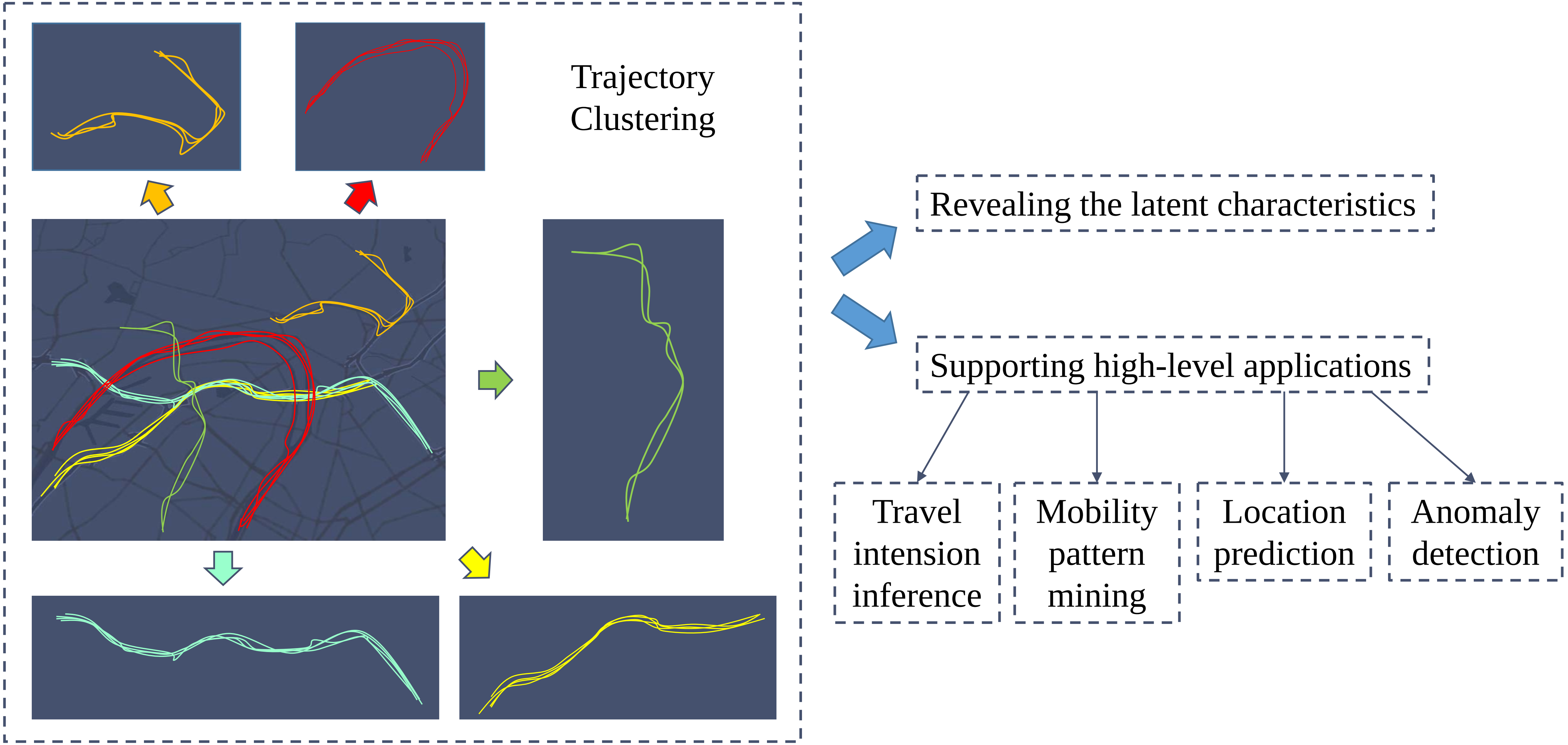
Trajectory Clustering, Classification and Anomaly Detection

Trajectory Clustering via Deep Representation Learning (Di Yao, Chao Zhang, Zhihua Zhu, Jianhui Huang, Jingping Bi) (IJCNN)

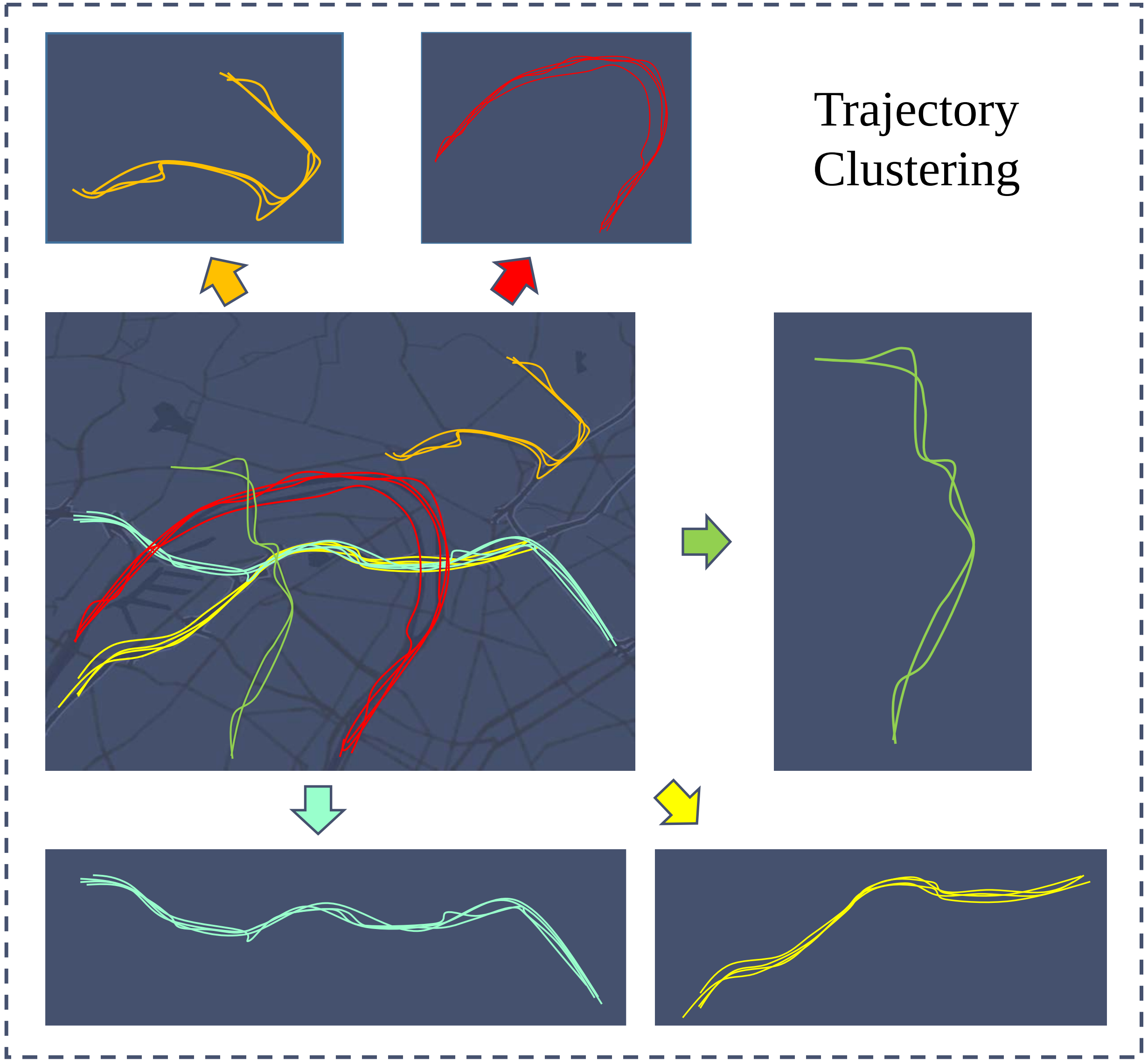
Motivation

- Trajectory clustering is one of the important trajectory analysis tasks.



Motivation

- Current measures can only group trajectories in a fixed region and time period.



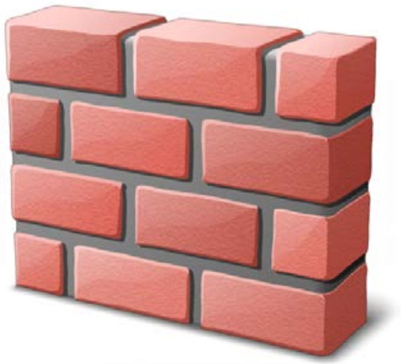
Current Measures

Clustering Algorithm → K-means, DBSCAN, Spectral clustering

Similarity Measure → DTW, EDR, LCSS

Limitations

Fixed Geographical Region



Fixed Time Period

Problem Formulation

- ❑ To generate a clusters set based on movement pattern which is time- and space- invariant.

Moving objects $O = \{o_1, \boldsymbol{o_2}, \dots, o_L\}$

For each object

GPS records $S_o = \{\boldsymbol{x_1}, x_2, \dots, x_M\}$

For each record

Tuple $x_i = (t_x, l_x, a_x, o_x)$

t_x: Timestamp

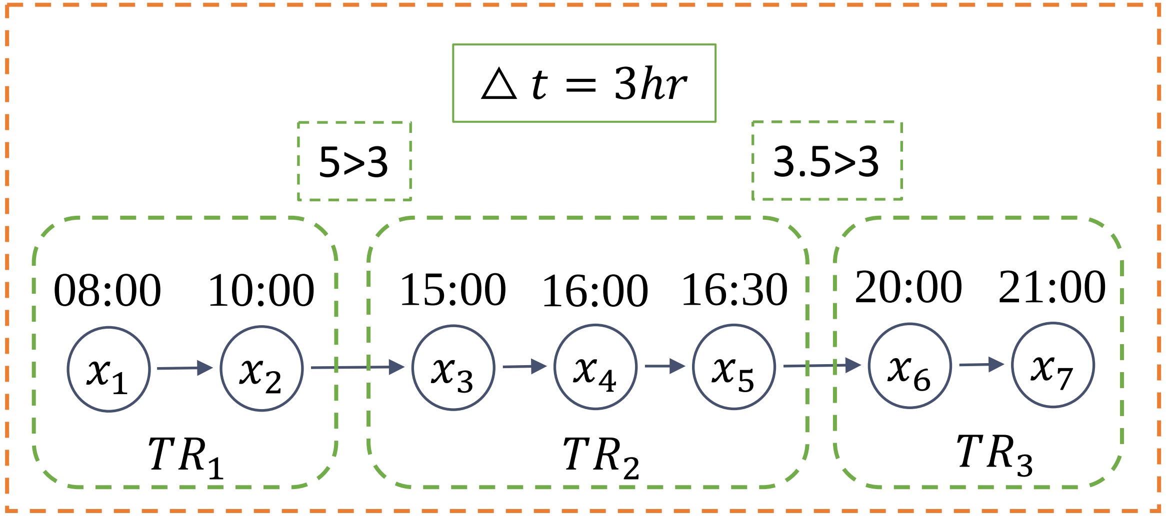
l_x: Two dimensional vector (latitude & longitude)

a_x: Set of attributes

o_x: Object ID

Segmentation

Gap threshold $\triangle t$



For all objects

Trajectory Set $\mathcal{T} = \{TR_1, TR_2, \dots, TR_N\}$

GOAL: Generating a set of clusters (space- and time- invariant)

Clusters $\mathrm{C} = \{C_1, C_2, \dots, C_K\}$

Framework Overview

- The framework is an **unsupervised** approach with four layers.

GPS records $S_o = \{\boldsymbol{x_1}, x_2, \ldots, x_M\}$

↓ Input

1st Trajectory Pre-processing Layer — Remove low-quality records + Segmentation (temporal continuity)

↓

2nd Moving Behaviour Feature Extraction Layer — Feature extraction algorithm → feature sequence

↓

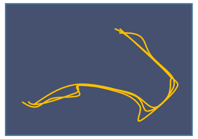
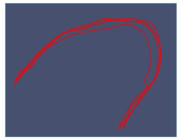
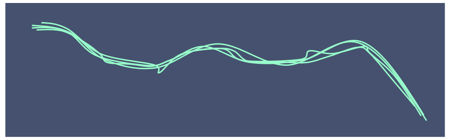
3rd Seq2Seq Auto-Encoder Layer — Embed feature sequence to a fixed-length vector

↓

4th Cluster Analysis Layer — Classic clustering algorithm

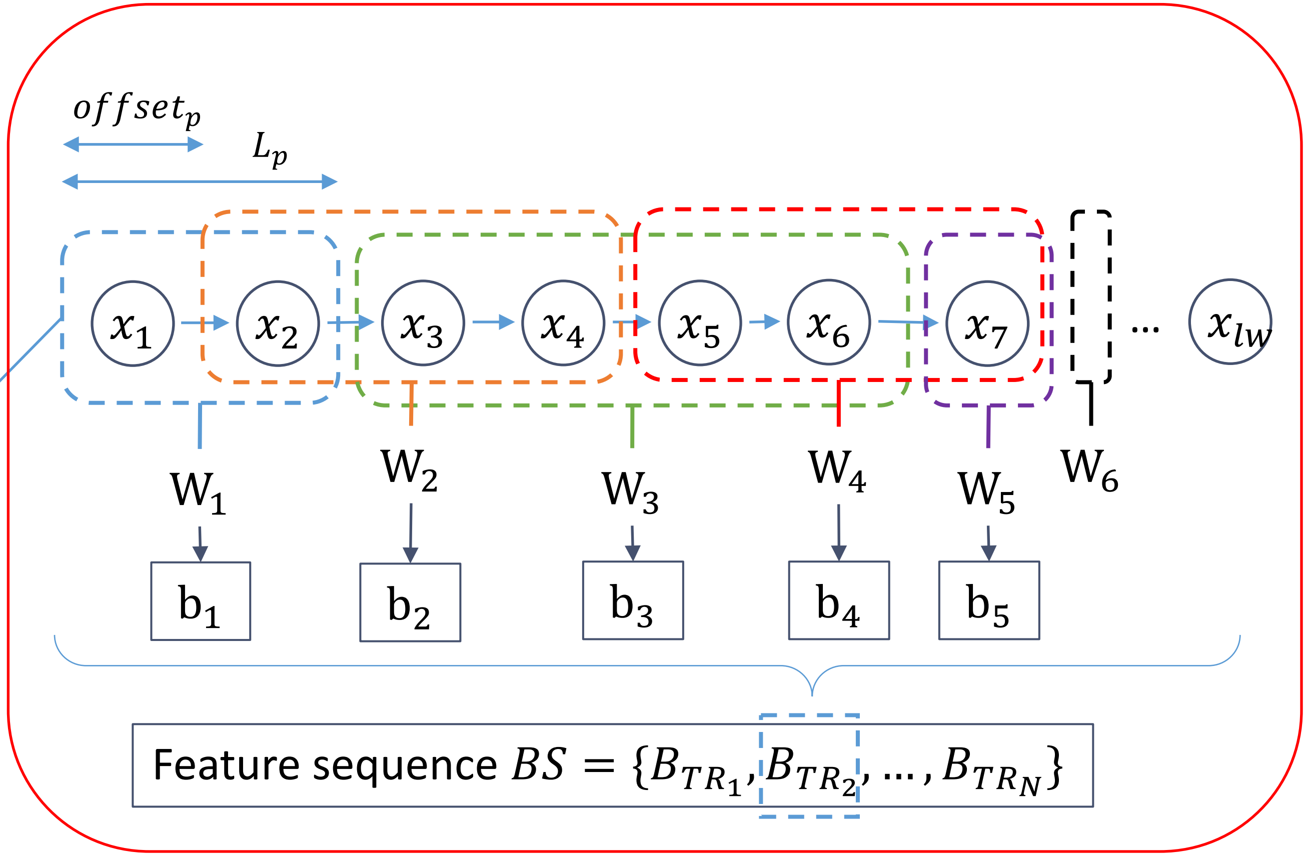
↓ Output

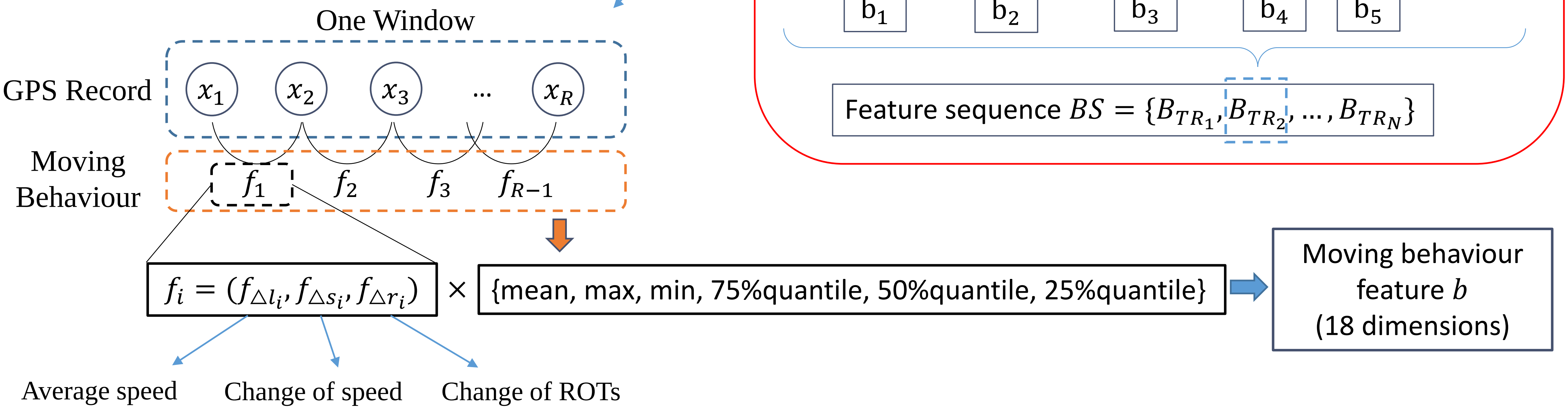
Clusters $C = \{C_1, C_2, \ldots, C_K\}$

Feature Extraction

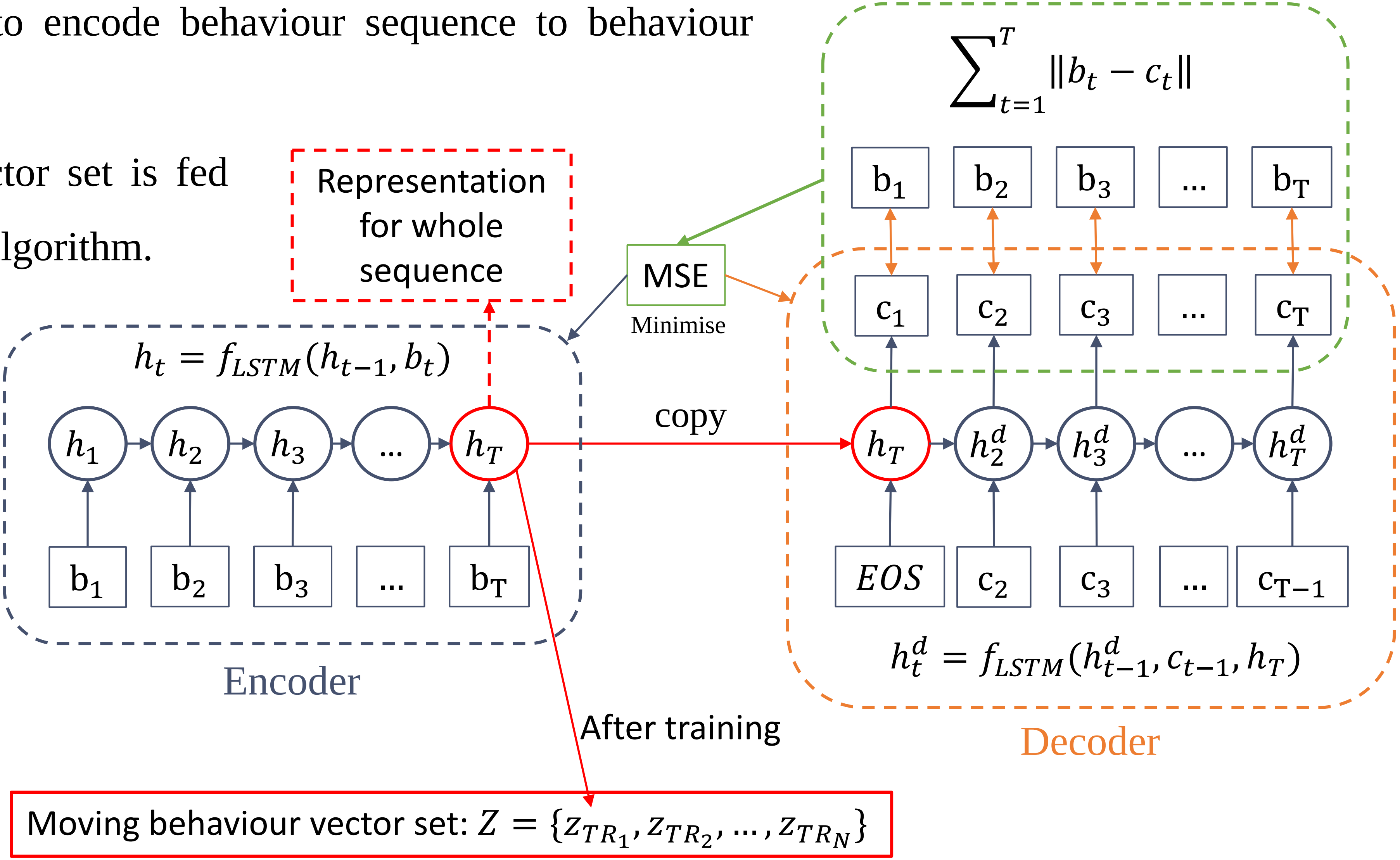
- Adopt sliding window to traverse the records and extract features.
- L_p: Width of window; $offset_p = L_P/2$.
- Each record is assigned to two windows.
- For each trajectory, a moving behaviour sequence is generated.





LSTM Seq2seq Auto-encoder

- LSTM is adopted to learn long-term dependencies.
- The seq2seq is train to encode behaviour sequence to behaviour vector.
- Moving behaviour vector set is fed into classic clustering algorithm.



Experiment

- Both synthetic and real datasets are used.
- Four compared methods: LCSS, DTW, EDR and Hausdorff Distance with K-Medoids clustering algorithm.
- The results are measured in precision, recall and accuracy.

Results

$$Precision = \frac{TP}{TP + FP}$$

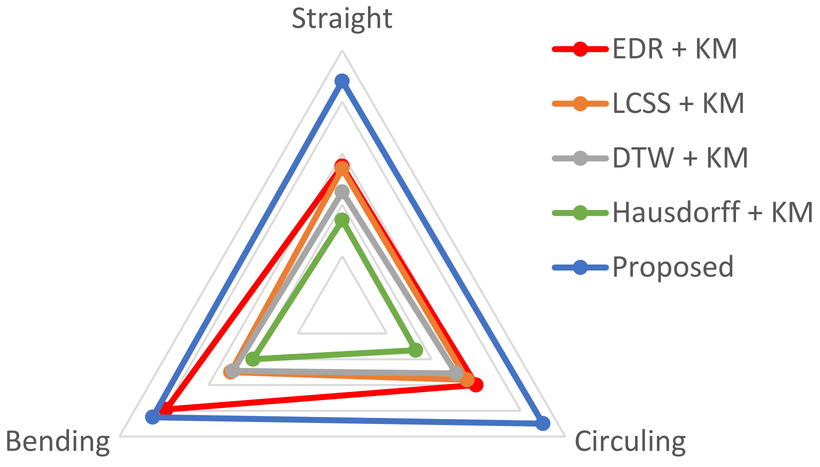
$$Recall = \frac{TP}{TP + FN}$$

$$Accuracy = \frac{all\ TP}{Number\ of\ trajectories}$$

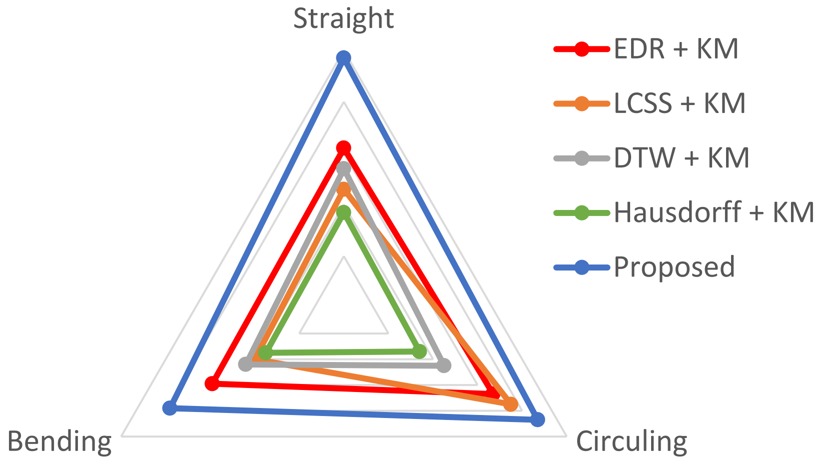
Truth / Result	Positive	Negative
Positive	TP	FP
Negative	FN	TN

Results

- Proposed method outperforms other baselines in synthetic data.

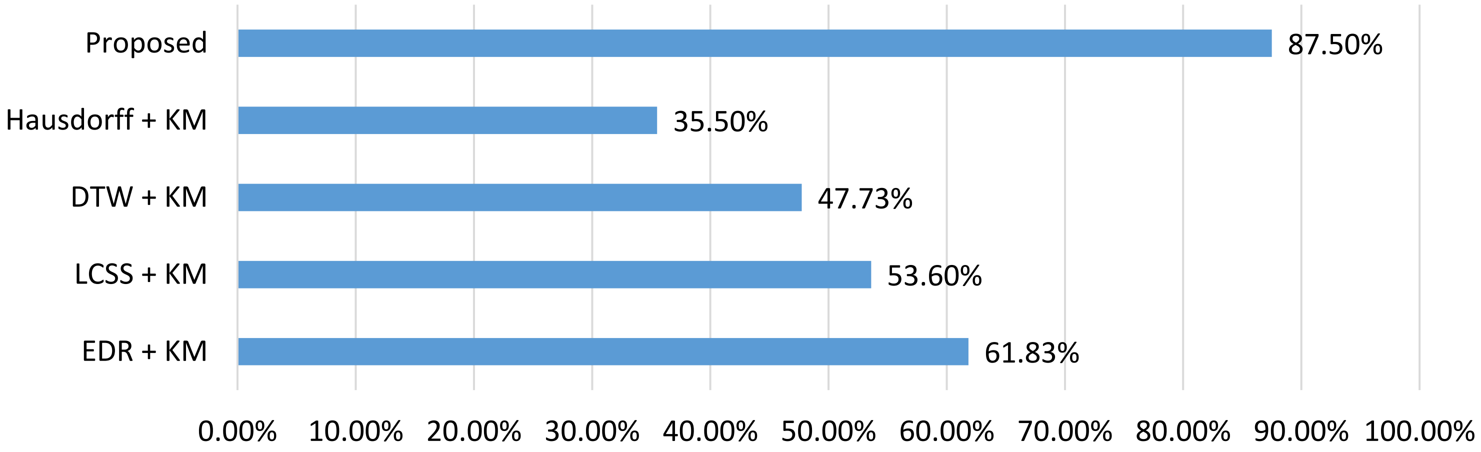


Precision



Recall

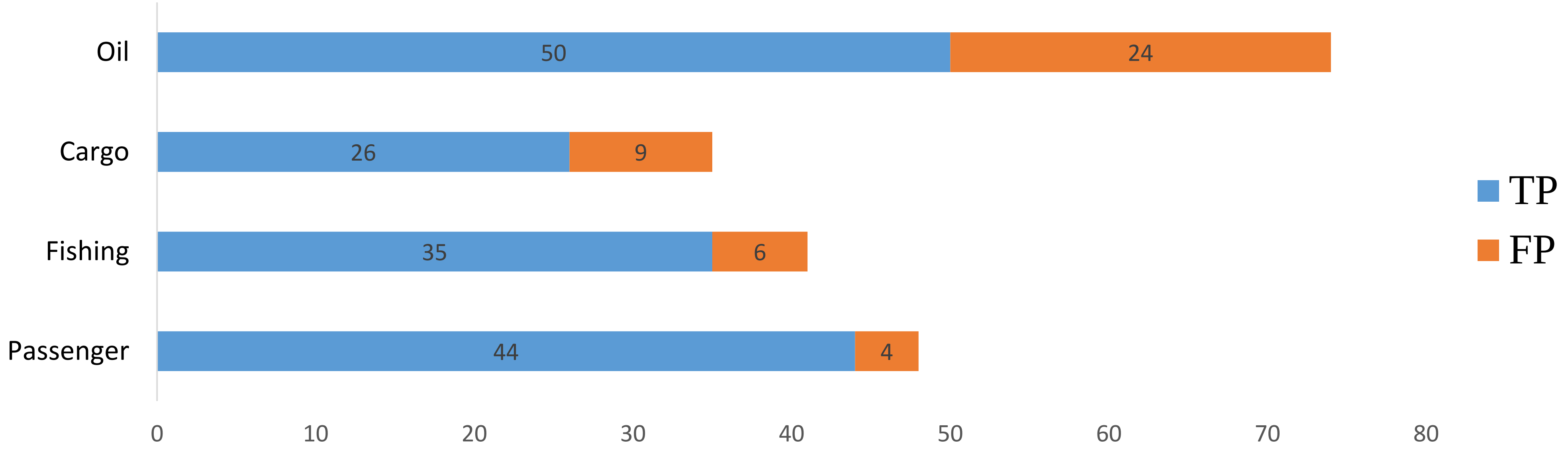
Accuracy



Results

- ❑ Proposed method works well in real data set.

	Passenger	Fishing	Cargo	Oil
Total Number	50	50	50	50
Precision	44/48=0.91	35/41=0.85	26/37=0.7	50/74=0.67
Recall	0.88	0.7	0.52	1.0
Overall accuracy: (44+35+26+50)/200 = **0.78**				



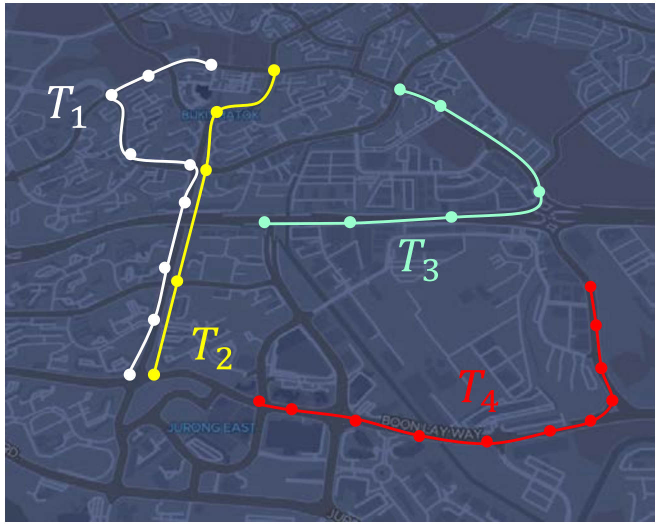
--------END--------

3. Trajectory Clustering, Classification and Anomaly Detection

Identify Human Mobility via Trajectory Embedding (Qiang Gao, Fan Zhou, Kunpeng Zhang, Goce Trajceski, Xucheng Luo, Fengli Zhang) (IJCAI-17)

Motivation

- Trajectory User Linking (TUL): Identify and link trajectories to users who generated them in the LBSN.
- More personalized recommendations and help in identifying terrorists/criminals.
- A semi-supervised model TUL via Embedding and RNN (TULER) is proposed.



A set of unlinked trajectory:
$\mathcal{T} = \{T_1, T_2, \dots, T_m\}$

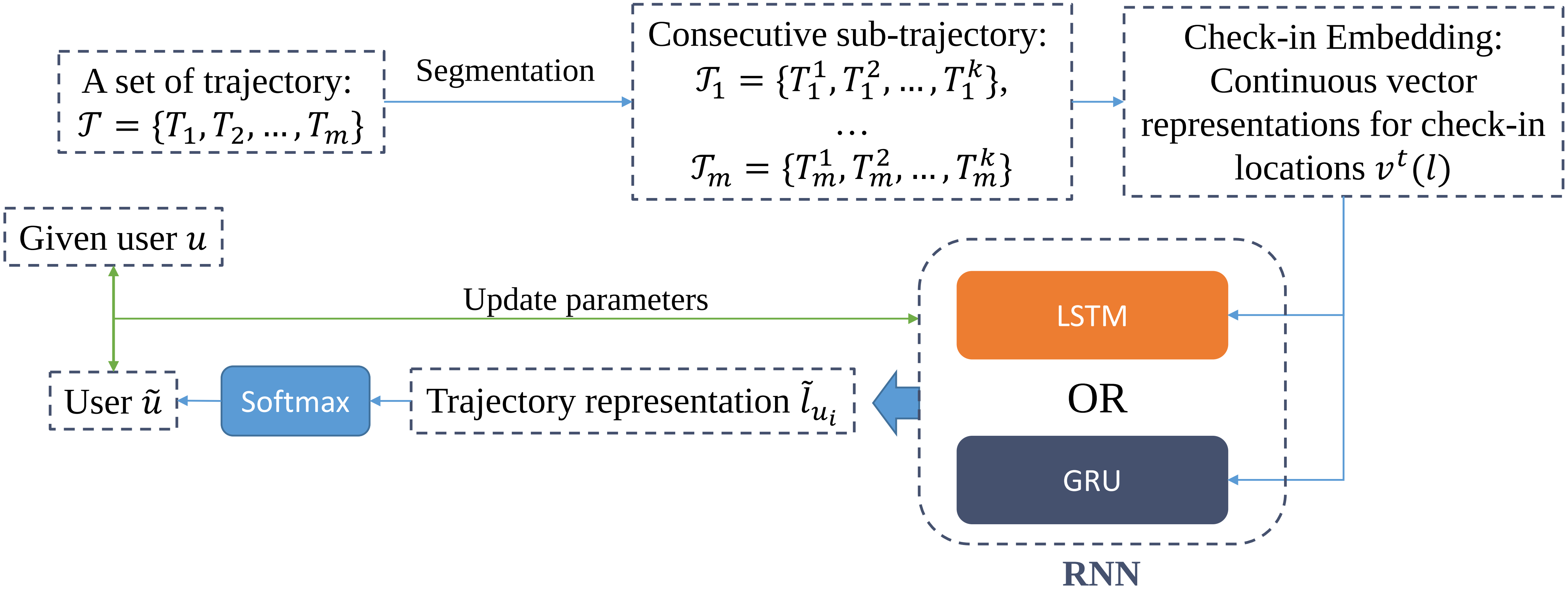
TULER

1. Divided into consecutive sub-trajectories
2. Trajectory embedding
3. RNN model

A set of users:
$\mathcal{U} = \{u_1, u_2, \dots, u_n\}$
$(m \gg n)$

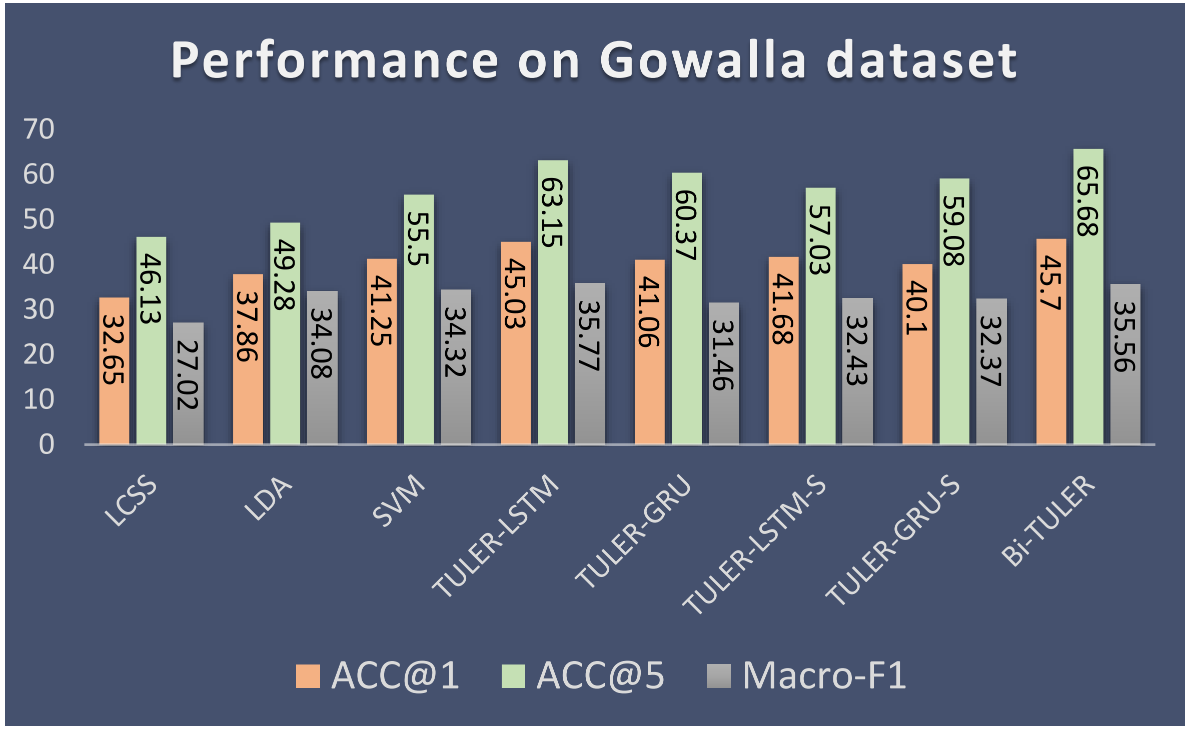
TULER

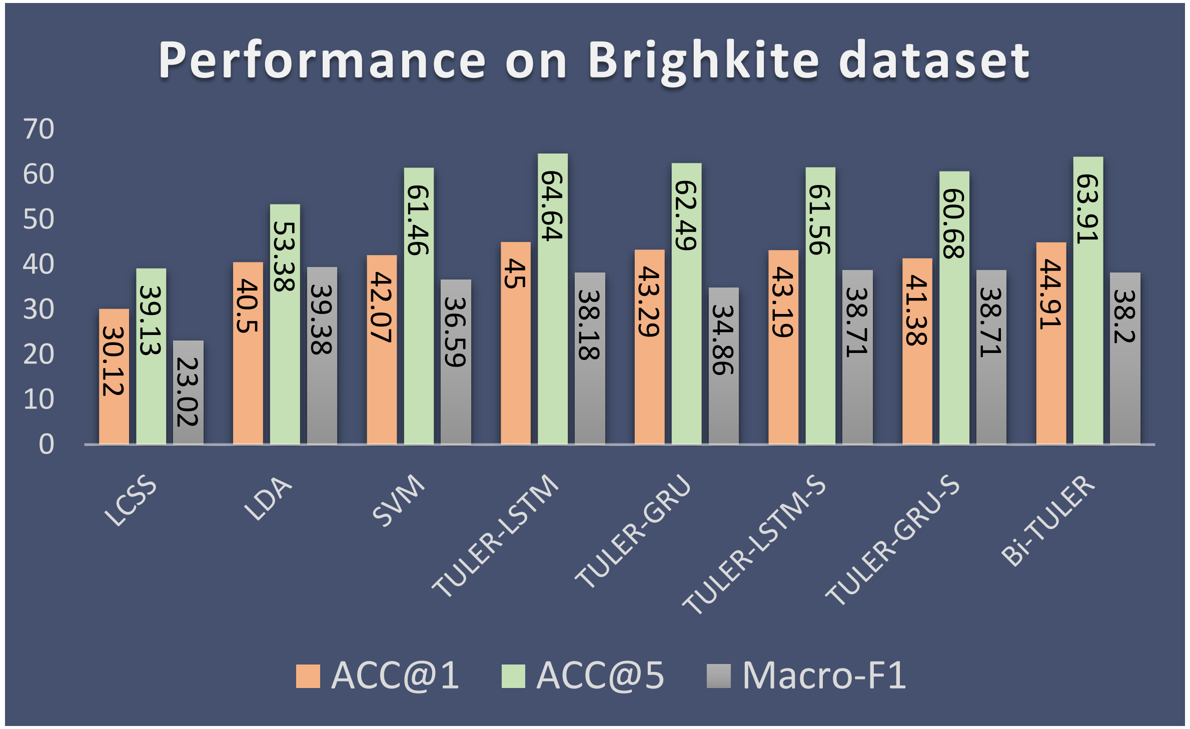
- To reduce the computation complexity, original trajectories are divided into k sub-sequences.
- Different types of RNN models are adopted to control the input and output of trajectory embedding.
- During trajectory embedding, some check-ins are randomly dropped to prevent overfitting.



Evaluation

- Two LBSN datasets are used to evaluate the performance of the TULER.
- TULER is compared with LCSS, SVM, LDA and its variants.





--------END--------

3. Trajectory Clustering, Classification and Anomaly Detection

Online Anomalous Trajectory Detection with Deep Generative Sequence Modeling (Yiding liu, Kaiqi Zhao, Gao Cong, Zhifeng Bao) (ICDE-20)

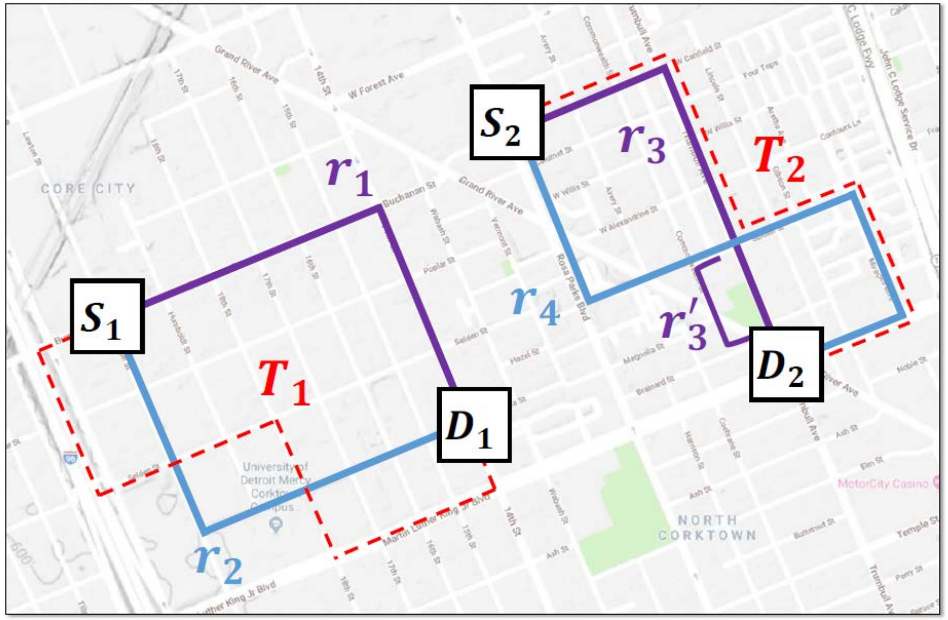
Definition

- **Trajectory**:
  - A sequence of chronologically ordered GPS points, represents the trace of a moving object (e.g., taxi trajectory).

- **Anomalous trajectory detection**:
  - Detecting trajectories that do not follow the normal routes of a specific travel itinerary (i.e., source and destination).
- **Applications**:
  - Preventing taxi frauds.
  - Detecting road constructions or accident.

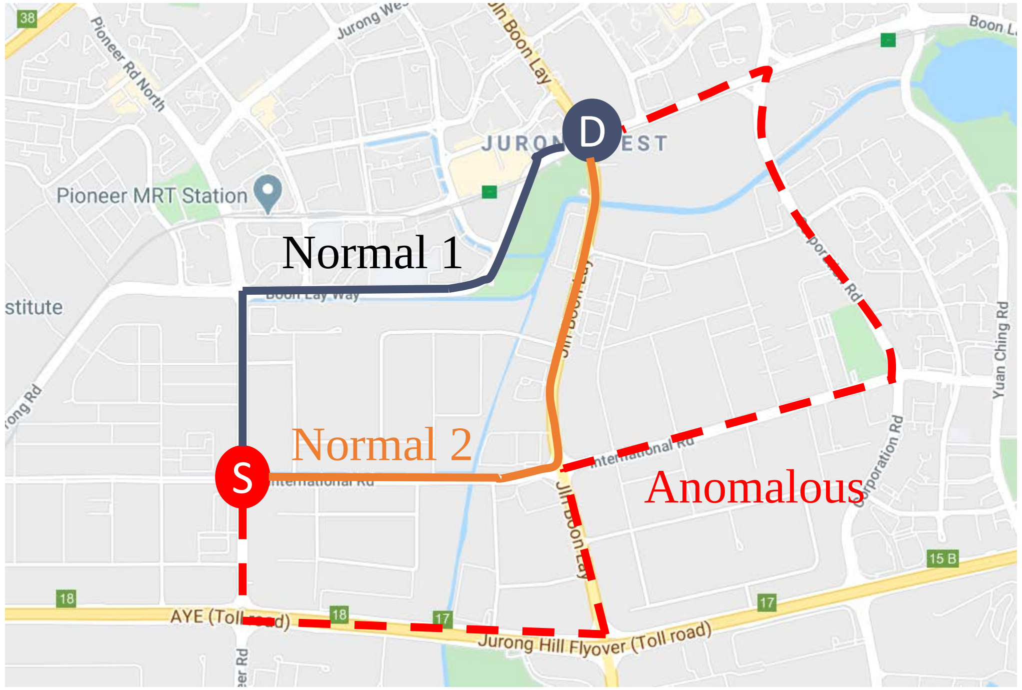
Definition

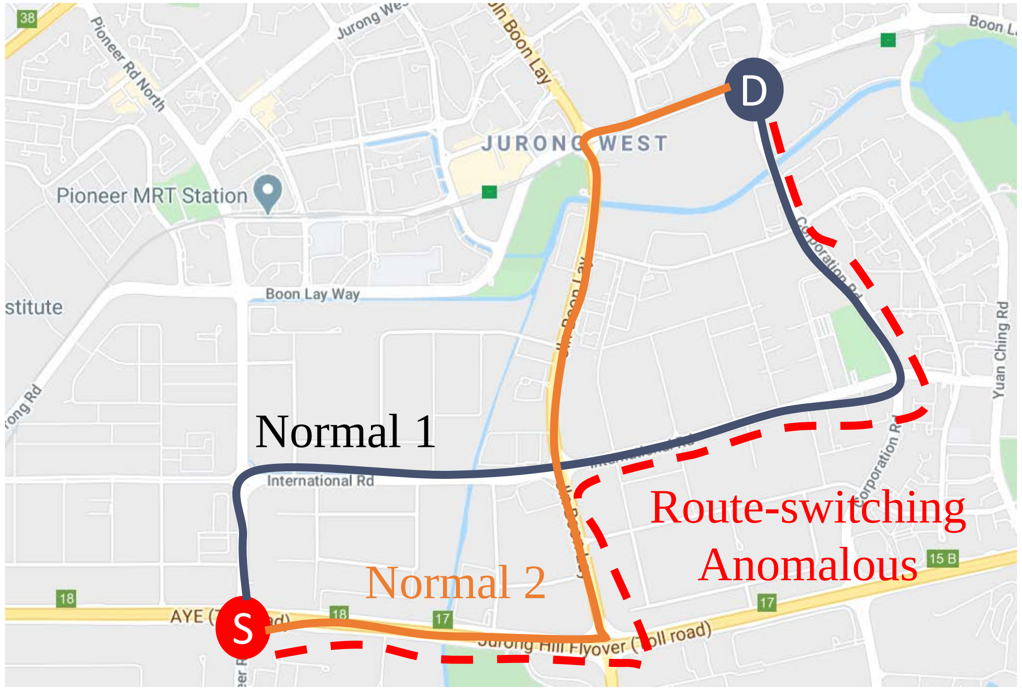
- Two examples:
  - r_1, r_2: routes followed by most trajectories between S_1 and D_1.
  - r_3, r_4: routes followed by most trajectories between S_2 and D_2.
  - T_1: **Detour** trajectory.
  - T_2: **Route-switching** trajectory.



Motivation

- Automatically detect anomalous trajectories has become a critical concern.
- Existing studies cannot do the following simultaneously:
  - Handle the complexity and variety of trajectory data to discover normal route.
  - Support efficient detection in an online manner.
- A novel model Gaussian Mixture Variational Sequence AutoEncoder (GM-VSAE) is proposed.





Challenge

- **Discovering normal routes from massive data.**
  - **Complexity**: trajectories vary across different places due to the complexity of transportation systems.
  - **Sequential correlation**: Capturing sequential correlations between route segments for detecting route-switching anomalies.

- **Efficient online detection.**
  - **Efficient detection**: quickly answering whether a trajectory is anomalous.
  - **Online detection**: detecting whether a trajectory is anomalous at the same time as it is being sequentially generated.

Challenge

- Existing studies: **a two-step framework**
  - Step 1: defining “normal routes” with **representative trajectories**.
  - Step 2: comparing a target trajectory with representative trajectories based on **distance or density metrics**.

- Limitations
  - Heuristic hand-crafted definition of representative trajectories and distance / density metrics.
  - Route-switching anomalies cannot be detected.
  - Efficient online detection is not supported.

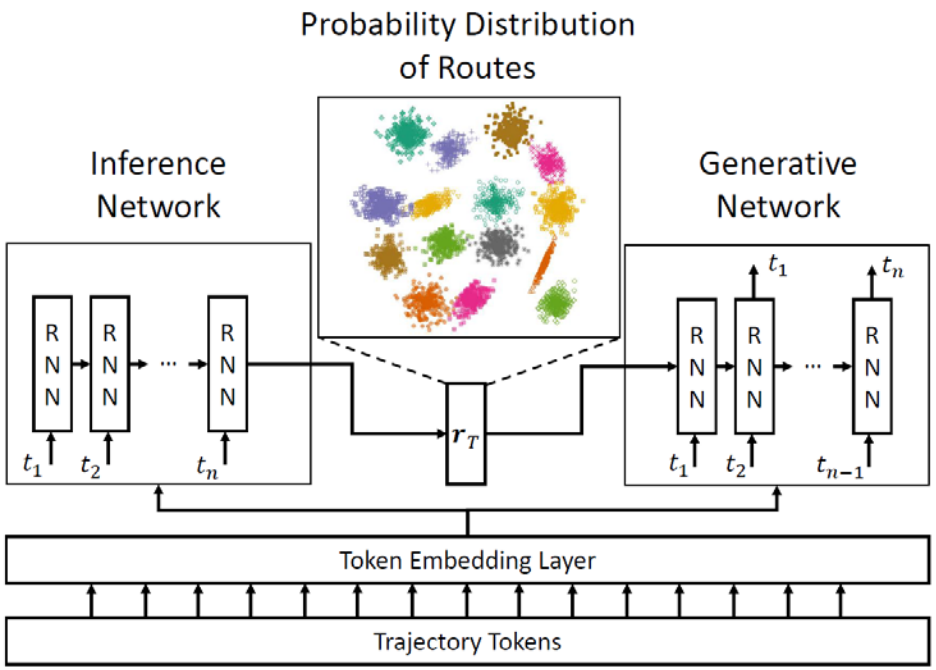
Novelty & Contribution

- To support more effective & efficient online anomalous trajectory detection:
  - We develop a **deep generative model**, which captures sequential information of trajectories, and encodes their route representations.
  - We jointly use a **Gaussian mixture distribution** to model the non-uniformly distributed route representations, which allow us to discover different types of normal routes.
  - We propose a novel paradigm, **detection-via-generation**, for anomalous trajectory detection using GM-VSAE, where the time complexity is linear to the trajectory length.
  - We propose an **approximate posterior inference** method that largely reduces the time cost of GM-VSAE and enables a more efficient online detection.

NANYANG TECHNOLOGICAL UNIVERSITY

Method

- Gaussian Mixture Variational Sequence Auto-Encoder
  - **Goal**: encoding trajectories as latent route representations.
  - **Components**:
    - **Inference network (RNN)**: encoding trajectory.
    - **Latent route distribution**: Gaussian mixture distribution.
    - **Generative network (RNN)**: decoding trajectory.



NANYANG TECHNOLOGICAL UNIVERSITY

Method

- Online Anomalous Trajectory Detection Framework
  - Using the latent route distribution and the generative network.
  - **Definition 1**: *normal routes* are those routes which are more likely to be traveled by trajectories.

$$p_\gamma(\mathbf{r}|c) = \mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\sigma}_c^2\mathbf{I})$$

The PDF of a Gaussian component

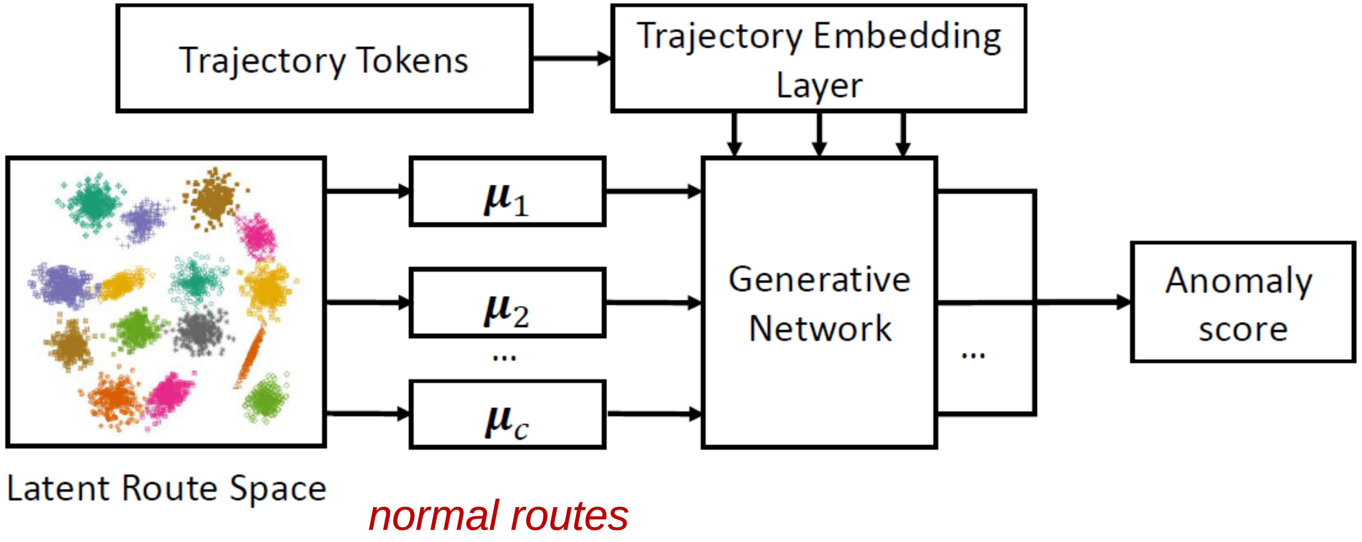
    - Definition of normal routes: . $\mathbf{r}_*^c \simeq \boldsymbol{\mu}_c$
  - **Definition 2**: *anomalous trajectories* cannot be well-generated from normal routes.
    - Definition of anomaly score:
    - Online detection:

$$s(T) = 1 - \arg\max_c \exp\left[\frac{\log p_\theta(T|\boldsymbol{\mu}_c)}{n}\right]$$

$$s(t_{\le i+1}) = 1 - \arg\max_c \exp\left[\frac{\log p_\theta(t_{\le i}|\boldsymbol{\mu}_c) p_\theta(t_{i+1}|t_{\le i}, \boldsymbol{\mu}_c)}{i+1}\right]$$

Method

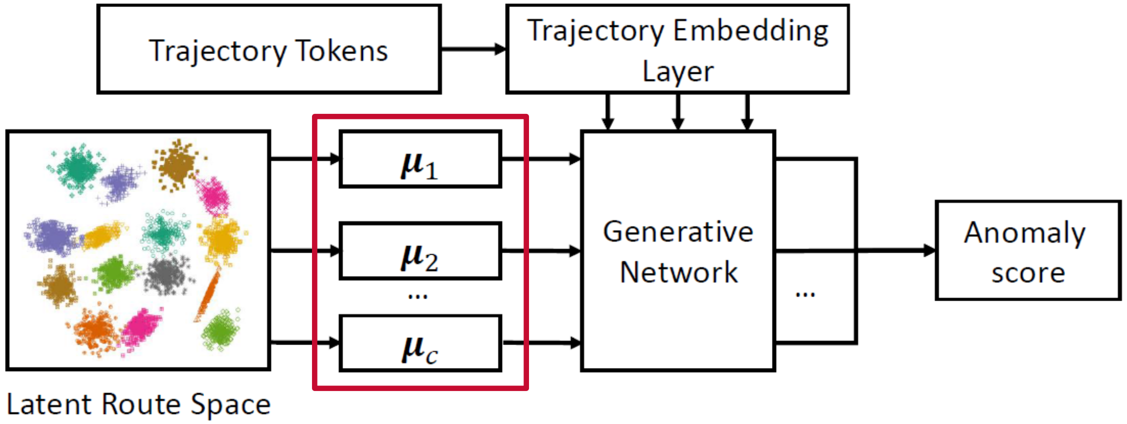
- Online Anomalous Trajectory Detection Framework
  - Using the latent route distribution and the generative network.
  - **Time complexity:**
    - Full trajectory detection: $O(n)$.
    - Score Updating: $O(1)$.



The GM-VSAE detection framework

Method

- Problem of the GM-VSAE detection framework



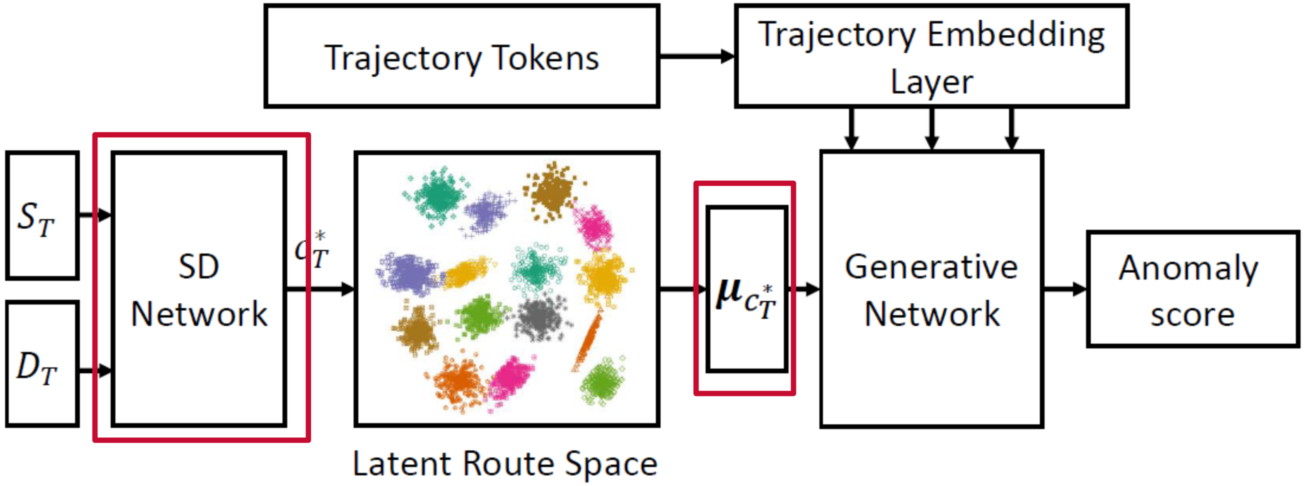
The GM-VSAE detection framework

- If the number of normal routes is large, the detection would be slow.

- **Definition**: c_T^* the real normal routes that T should travel.
- **Question**: Can we infer c_T^* beforehand?

Method

- Improving efficiency with SD-VSAE
  - Leveraging the **source** and **destination**.
  - **Intuition**: the normal routes between a specific source-destination pair usually have the same or similar route type.
  - SD-network: An auxiliary MLP for normal route inference:



The SD-VSAE detection framework

Time cost: c times faster than GM-VSAE!

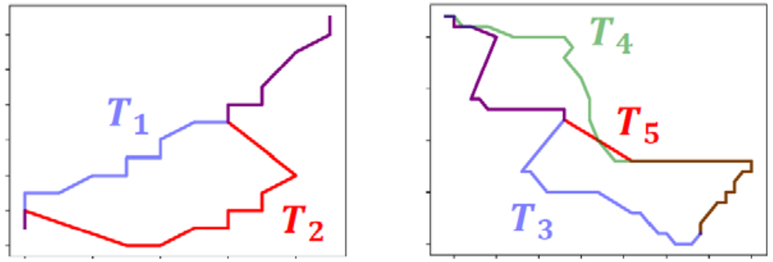
NANYANG TECHNOLOGICAL UNIVERSITY

Experiments

- Dataset

Dataset	#Points	#Trajectories	$\#(S, D)$	Avg. n
Porto	11,635,104	262,574	4,567	44.31
Beijing	909,642	52,497	6,752	17.32

- Anomalies



(a) Detour (D). (b) Route-switching (RS).

- Implementation details
  - Embedding size: 32.
  - Hidden state size: 256.
  - Dim of latent space: 256.
  - Batch size: 128
  - Non-linearity: ReLU.
  - Learning rate: 1e-4.
  - Optimizer: Adam.

- Baselines
  - TRAOD.
  - T-DBSCAN.
  - iBAT.
  - DBTOD-embed.
  - Sequence Auto-Ecoder (SAE).
  - Variational SAE.

Experiments

- Online anomalous trajectory detection
  - Metric: Precision-Recall AUC.
  - Anomaly proportion: 5%.
  - Parameter for online detection:
    - ρ: the proportion of a trajectory being observed for detection

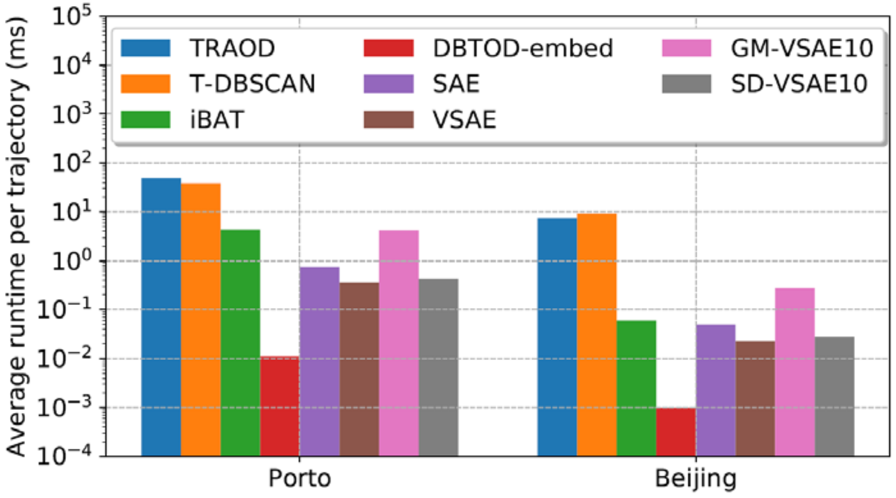
  - Overall comparison on Porto data:

PERFORMANCE COMPARISON FOR ANOMALOUS TRAJECTORY DETECTION IN TERMS OF PR-AUC ON PORTO DATASET.

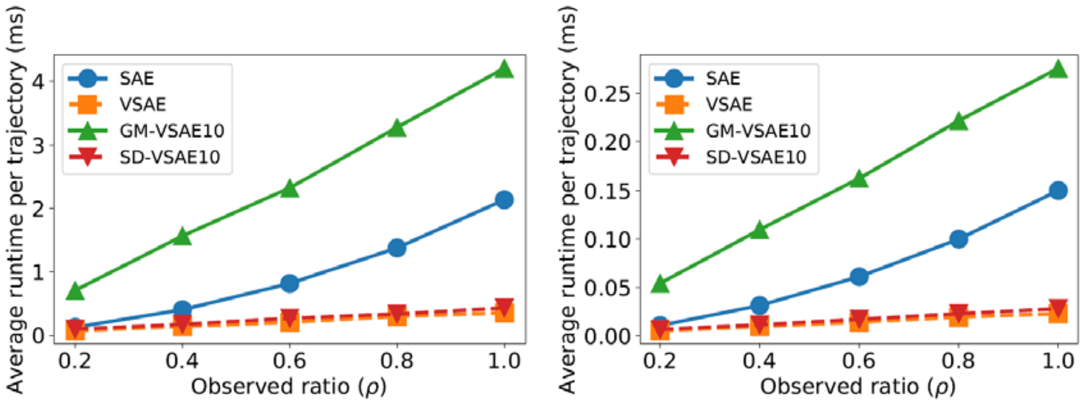
	Detecting detour anomalies (D)									Detecting route-switching anomalies (RS)					
Perturb params	d=3; $\alpha=0.1$			d=5; $\alpha=0.1$			d=3; $\alpha=0.3$			$\beta=0.3$		$\beta=0.5$		$\beta=0.7$	
Observed ratio (ρ)	0.1	0.5	1.0	0.1	0.5	1.0	0.1	0.5	1.0	0.5	1.0	0.7	1.0	0.9	1.0
TRAOD	-	-	0.212	-	-	0.311	-	-	0.161	-	0.189	-	0.183	-	0.179
T-DBSCAN	-	-	0.231	-	-	0.305	-	-	0.253	-	0.240	-	0.245	-	0.201
iBAT	0.156	0.184	0.181	0.159	0.196	0.193	0.182	0.371	0.406	0.200	0.179	0.177	0.173	0.177	0.175
DBTOD-embed	0.148	0.146	0.277	0.148	0.141	0.279	0.159	0.297	0.384	0.138	0.285	0.171	0.292	0.240	0.292
SAE	0.152	0.46	0.717	0.154	0.502	0.824	0.176	0.666	0.921	0.469	0.396	0.459	0.424	0.440	0.426
VSAE	0.170	0.455	0.701	0.174	0.501	0.819	0.197	0.660	0.910	0.614	0.566	0.619	0.571	0.605	0.591
GM-VSAE10	0.230	0.473	0.773	0.234	0.513	0.842	0.253	0.702	0.961	0.640	0.597	0.636	0.606	0.634	0.618
GM-VSAE20	0.334	0.494	**0.811**	0.335	0.522	**0.868**	**0.361**	0.711	**0.969**	**0.717**	**0.662**	0.721	0.678	**0.720**	0.706
GM-VSAE50	**0.338**	**0.498**	**0.811**	**0.337**	**0.523**	**0.868**	0.358	**0.714**	0.963	0.715	0.660	**0.725**	**0.705**	0.718	**0.707**

Experiments

- Overall efficiency:



- Scalability:

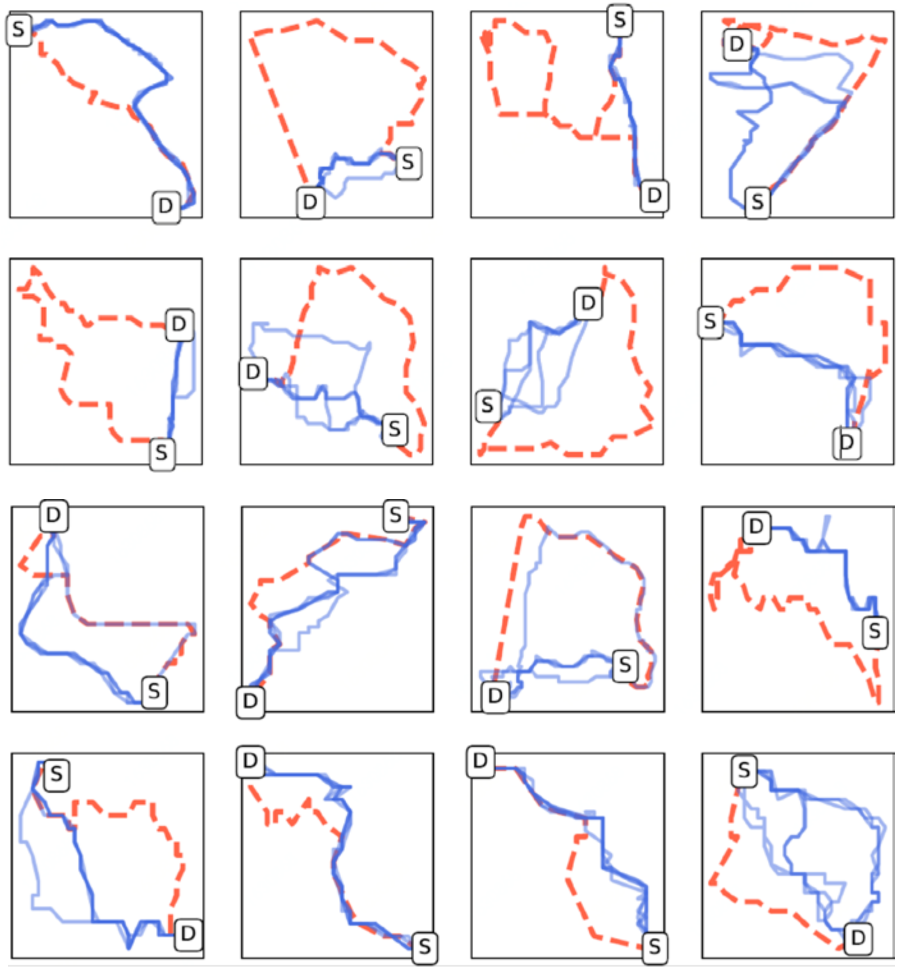


(a) Porto (b) Beijing.

NANYANG TECHNOLOGICAL UNIVERSITY

Experiments

- Visualization
  - Case study of the detected anomalous trajectory:



NANYANG TECHNOLOGICAL UNIVERSITY

Trajectory Prediction for Human Mobility Analytics

Yile Chen, Cheng Long, Gao Cong, Chenliang Li Context-aware Deep Model for Joint Mobility and Time Prediction, WSDM 2020

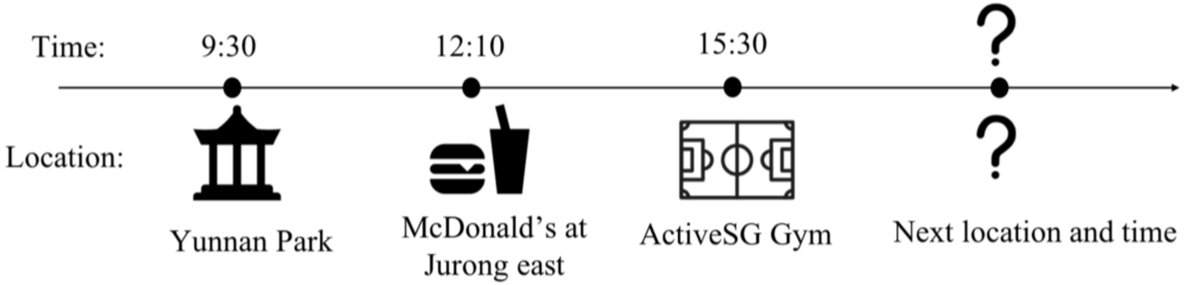
Introduction

- With the help of geo-positioning technologies, we are able to collect a huge amount of location data that records our mobility.

- Data source includes:
  - GPS records
  - Telco data
  - Location based social network (Instagram, Facebook, etc.)

Introduction

- Existing methods only focus on predicting the next location a user will arrive but ignore the corresponding arrival time.
- We are interested in predicting both the next location and its arrival time of a user, given his/her current mobility trajectory and historical mobility trajectories.



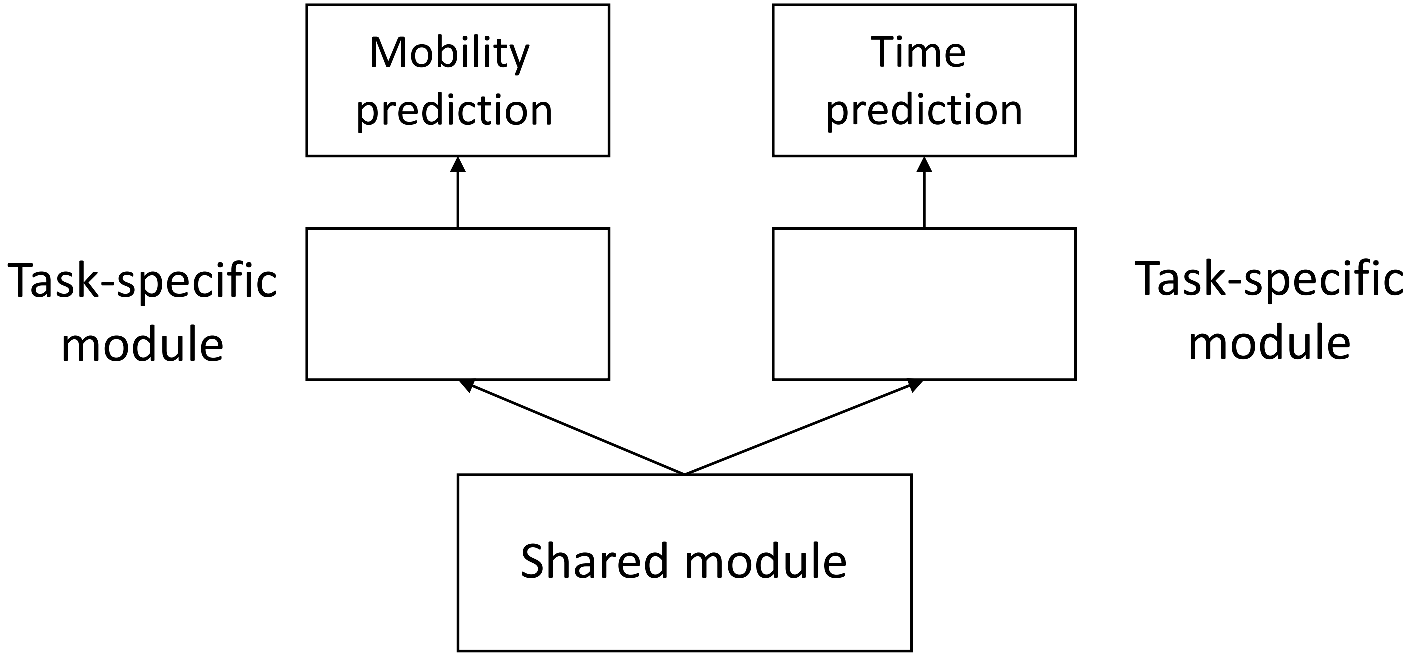
- Applications:
  - congestion control, targeted advertisement.

Challenges

- Joint prediction
  - ➢Combine mobility and time prediction in a unified way such that they can complement each other.
- Context awareness
  - ➢Consider not only the mobility transition patterns, but also rich context information unique in mobility records.
- Data Sparsity
  - ➢Some users have limited mobility records.

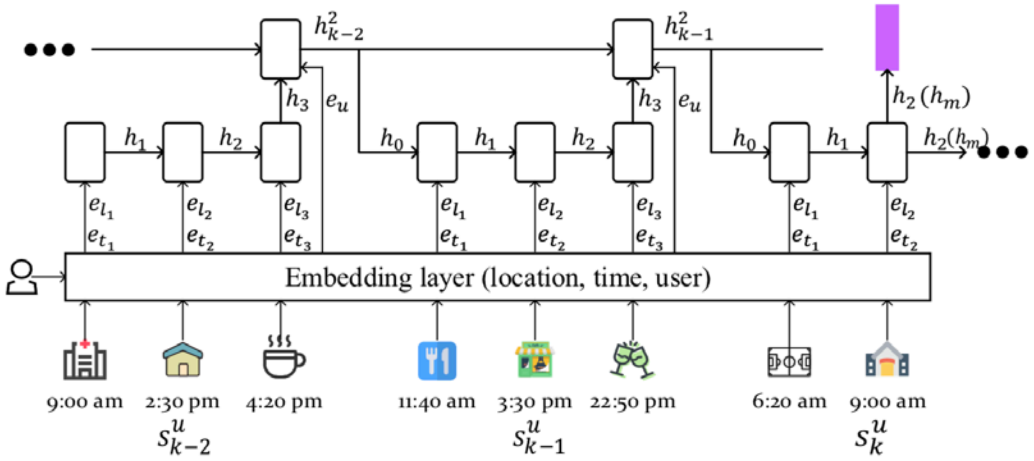
DeepJMT Model

- Joint mobility and time modeling
  - Shared module + Task-specific modules.



DeepJMT Model

- Joint mobility and time modeling
  - Mobility trajectory is a kind of sequential data and it is suitable to be modelled by RNN
  - Hidden representation of RNN encodes the sequential patterns that indicate the mobility regularities and temporal patterns at each location, and it could be further processed as shared input for both mobility and time prediction tasks.



DeepJMT Model

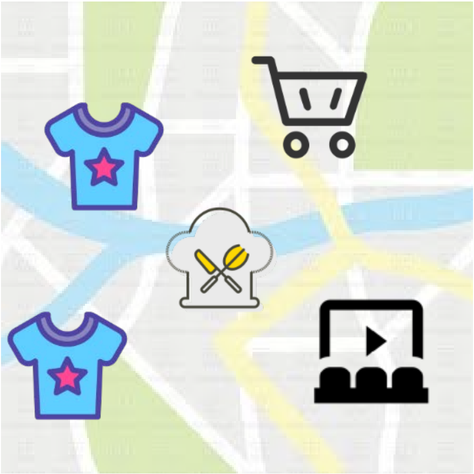
- Joint mobility and time modeling
  - Mobility prediction module: feed forward neural network with hidden representation as input.
  - Time prediction module: conditional density function of time could be derived from intensity function λ(t) that models the rate of event happening

$$f(t) = \lambda(t) \exp\left(-\int_{t_m}^{t} \lambda(\tau) d\tau\right)$$

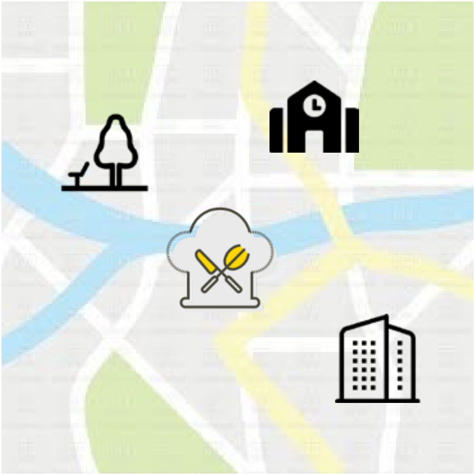
    - Intensity function λ(t) could be parameterized by a neural network where the input is also the hidden representation.
  - In this way, mobility and time prediction could be coupled in a unified framework.

DeepJMT Model

- Spatial context extraction
  - identify the semantics of each location
  - spatial neighbors could provide some information about the functionality of the target location
  - example:



Semantics: shopping center, shopping/entertainment activity



Semantics: residential area, having a meal

DeepJMT Model

- Spatial context extraction
  - The spatial neighbors of a location could be considered as a graph and we apply similar technique as Graph Attention Network (GAT) to derive the spatial context.

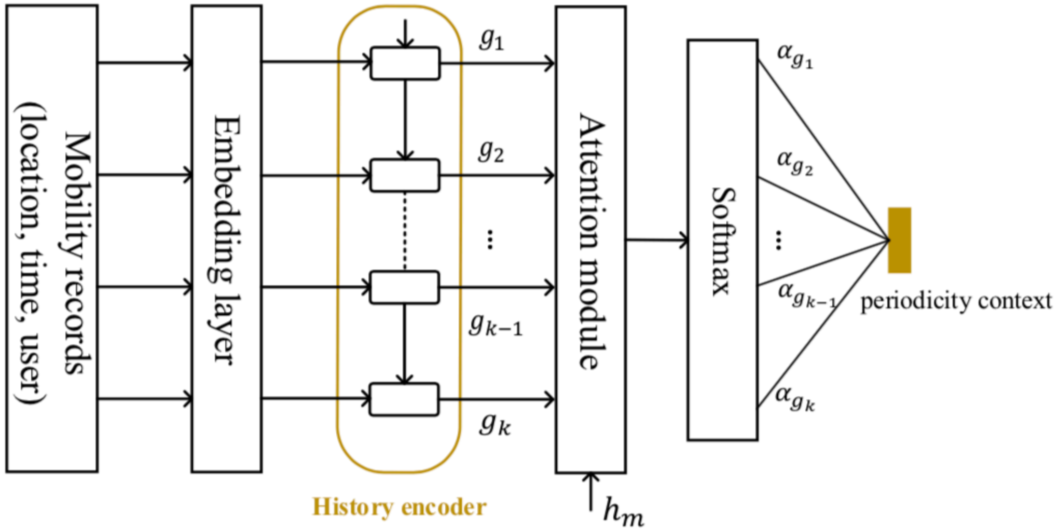
$$c_l = g_l\left(\sum_{l_i \in C(l)} \alpha_{l_i} e_{l_i}\right)$$

  - The weight of each spatial neighbor is calculated considering both dynamic influence and geographical distance

$$\alpha_{l_i} = \frac{\exp\left(q^s_{l_i} \cdot D(l_i, l)\right)}{\sum_{l_\tau \in C(l)} \exp\left(q^s_{l_\tau} \cdot D(l_\tau, l)\right)} \qquad D(l_x, l_y) = \exp\left(-\frac{dist(l_x, l_y)}{\beta}\right)$$

DeepJMT Model

- Periodicity context extraction
  - human mobility often manifests multi-level periodicity.
  - pay different attention to historical mobility records to extract periodicity representation.
  - use attention mechanism to derive the representation



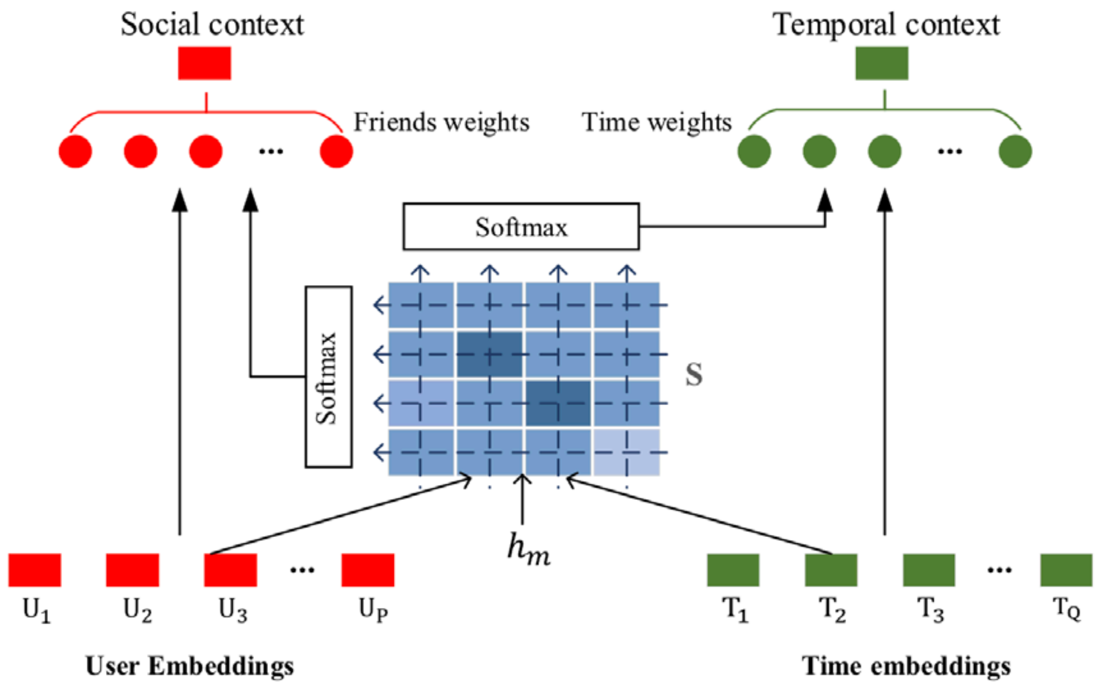
$$\alpha_{h'_i} = \frac{\exp\left(q^h_{h'_i}\right)}{\sum_{\tau=1}^{k} \exp\left(q^h_{h'_\tau}\right)}, c_p = \sum_{i=1}^{k} \alpha_{h'_i} h'_i$$

DeepJMT Model

- Alleviate sparsity issue
  - a user usually has similar behaviors and shares similar interests with his/her friends. Therefore we could extract more evidence about the mobility and temporal patterns from friends.
  - apply co-attention mechanism to jointly reason about the weights of different user-time pairs considering current mobility status.

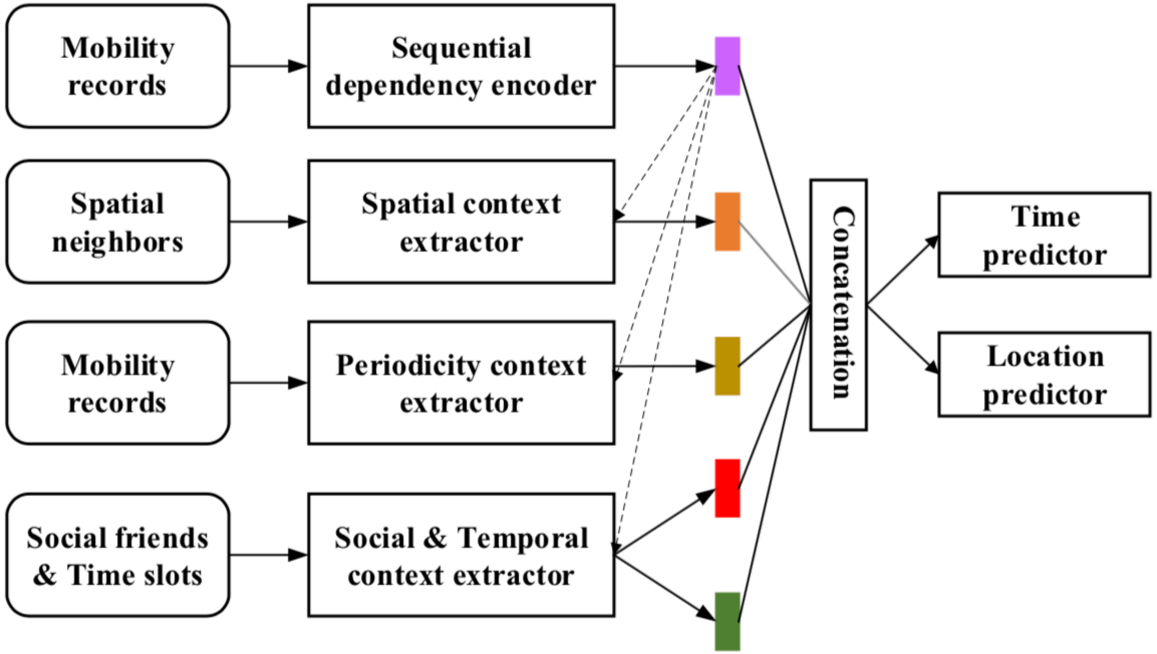
DeepJMT Model

- Alleviate sparsity issue
  - social context: aggregated influence on a user from friends
  - temporal context: capture a user’s preference on different time slots



DeepJMT Model

- Multi-task learning
  - Maximize the log likelihood of both time and location prediction:



$$L = -\sum_{s_t^u \in S_u} \sum_{m=1}^{|s_t^u|-1} \left(\log P\left(l_{m+1} | \mathcal{H}_{t_m}\right) + \log f\left(t_{m+1}\right) \right)$$

Conditional probability for mobility

Conditional pdf for time

Experiment

- Statistics of the datasets

Dataset	#users	#locations	#trajectories	Avg. traj. for a user
NYC	1069	8,358	34,439	32.2
IST	7960	13,844	179,751	22.6
TKY	4662	11,747	156,982	33.7

- Three types of baselines:
  - ➢Mobility prediction
  - ➢Time prediction
  - ➢Joint mobility and time prediction

Experiment

- Mobility prediction
  - PRME (Feng et al., IJCAI’ 15)
  - GRU
  - STRNN (Liu et al., AAAI’ 16)
  - DeepMove (Feng et al., WWW’ 18)
- Time prediction
  - Hawkes process
  - Self-correcting process
- Joint mobility and time prediction
  - RMTPP (Du et al., KDD’ 16)
  - IntensityRNN (Xiao et al., AAAI’ 17)
  - DeepJMT (ours)

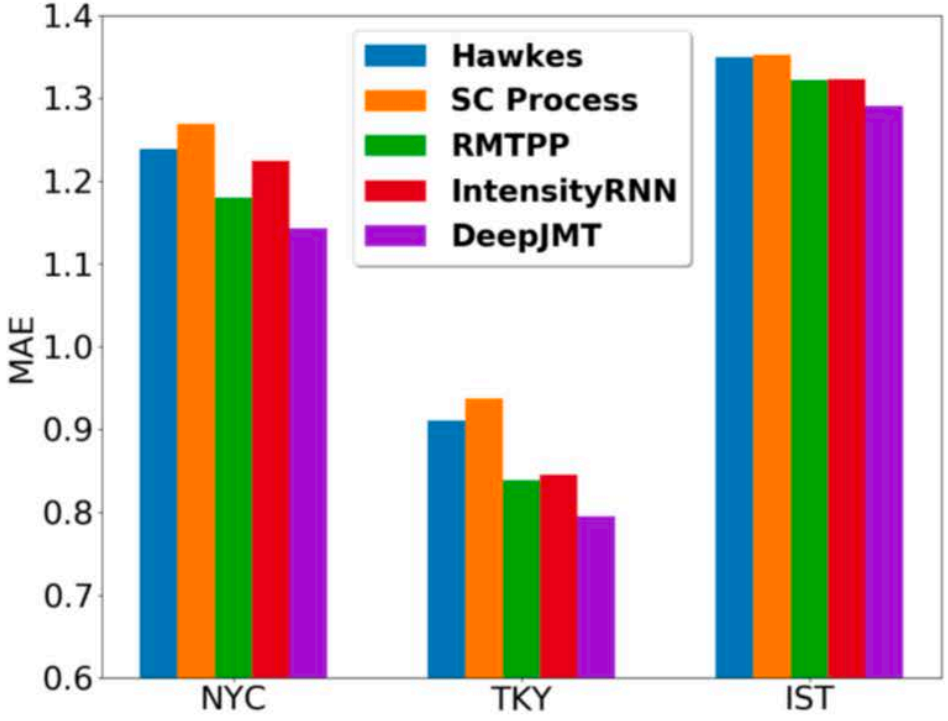
Experiment

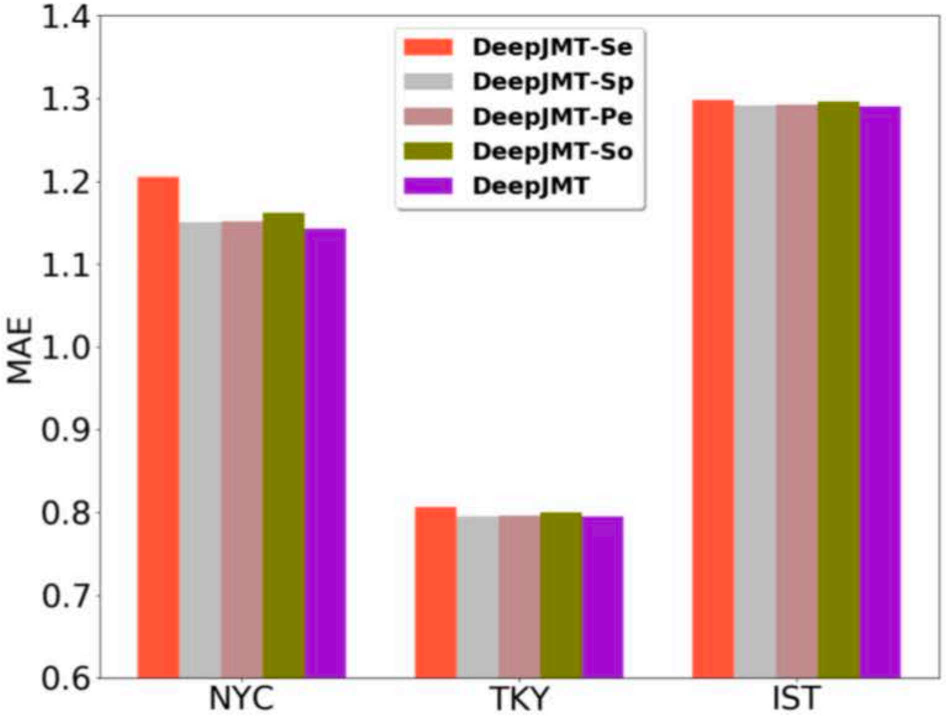
- Result of mobility prediction

Methods	NYC				TKY				IST			
	HR@5	HR@10	HR@20	MAP	HR@5	HR@10	HR@20	MAP	HR@5	HR@10	HR@20	MAP
PRME	0.2023	0.2696	0.3140	0.0679	0.2210	0.2738	0.3312	0.0411	0.0789	0.1325	0.1968	0.0121
GRU	0.2927	0.3416	0.3834	0.1242	0.2747	0.3433	0.4142	0.0879	0.1524	0.2108	0.2778	0.0581
STRNN	0.2771	0.3365	0.3789	0.1183	0.2808	0.3428	0.4085	0.0744	0.1481	0.2168	0.2766	0.0551
DeepMove	0.3847	0.4605	0.5271	0.1483	0.3642	0.4481	0.5379	0.1106	0.2386	0.3077	0.3726	0.0832
RMTPP	0.3532	0.4253	0.4871	0.1424	0.3452	0.4191	0.4869	0.1067	0.1829	0.2385	0.3028	0.0696
IntensityRNN	0.3552	0.4244	0.4832	0.1419	0.3412	0.4117	0.4789	0.1047	0.1740	0.2269	0.2900	0.0657
DeepJMT-So	0.4093	0.4882	0.5496	0.1595	0.4027	0.4843	0.5597	0.1224	0.2498	0.3184	0.3959	0.0913
DeepJMT-Se	0.3891	0.4418	0.5059	0.1375	0.3821	0.4677	0.5366	0.1103	0.2343	0.3010	0.3706	0.0764
DeepJMT-Sp	0.4021	0.4806	0.5381	0.1540	0.3843	0.4601	0.5306	0.1142	0.2345	0.3008	0.3678	0.0874
DeepJMT-Pe	0.4066	0.4853	0.5420	0.1602	0.4012	0.4838	0.5588	0.1220	0.2504	0.3201	0.3968	0.0903
DeepJMT	**0.4122**	**0.4924**	**0.5536**	**0.1623**	**0.4050**	**0.4876**	**0.5622**	**0.1240**	**0.2541**	**0.3226**	**0.3997**	**0.0928**

Experiment

- Result of time prediction

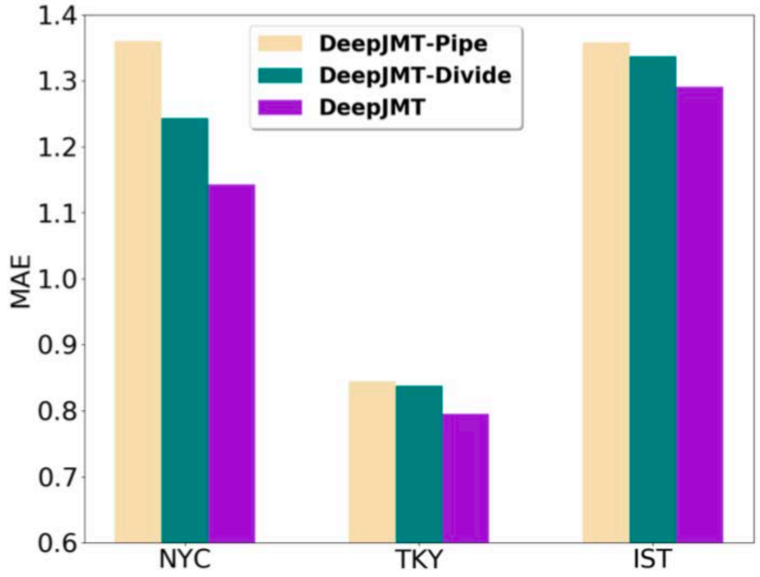




Experiment

- Result of joint learning paradigm

Methods	NYC				TKY				IST			
	HR@5	HR@10	HR@20	MAP	HR@5	HR@10	HR@20	MAP	HR@5	HR@10	HR@20	MAP
DeepJMT-Pipe	0.4069	0.4901	0.5508	0.1580	0.4014	0.4822	0.5585	0.1220	0.2505	0.3187	0.3981	0.0905
DeepJMT-Divide	0.4020	0.4830	0.5443	0.1550	0.3985	0.4809	0.5560	0.1208	0.2450	0.3136	0.3917	0.0882
DeepJMT	**0.4122**	**0.4924**	**0.5536**	**0.1623**	**0.4050**	**0.4876**	**0.5622**	**0.1240**	**0.2541**	**0.3226**	**0.3997**	**0.0928**



Summary

- We propose to solve joint mobility and time prediction task in a unified framework.
- We incorporate rich context information that is unique in mobility trajectory into our model and alleviate the sparsity problem by leveraging evidence from friends.
- We conduct experiments to show that our model can achieve better results on both tasks compared with baseline models.