TULSN: Siamese Network for Trajectory-user Linking

Yong Yu[1,2], Haina Tang[1,*], Fei Wang[2], Lin Wu[2],Tangwen Qian[2], Tao Sun[2], Yongjun Xu[2]
[1]*University of Chinese Academy of Sciences, China*
[2]*Institute of Computing Technology, Chinese Academy of Sciences, China*
yuyong171@mails.ucas.edu.cn
hntang@ucas.ac.cn
{wangfei, wulin, qiantangwen, suntao, xyj}@ict.ac.cn

***Abstract*—Trajectory-user linking (TUL), whereby a trajectory is linked to its owner in location-based social networks, is a fundamental and critical task in spatio-temporal data mining. It plays a key role in personalized recommendation, anomaly detection, and semantic trajectory mining. Existing methods for TUL are either rule-based methods, which link trajectories and users based on conventional trajectory similarities, or learning-based methods, which learn a classification model to map trajectories to their owners. However, rule-based methods ignore the semantic information in the trajectory sequence, and learning-based methods require retraining the model each time a new user is added. In this paper, we propose a Siamese network-based model for trajectory-user linking (TULSN), which uses a Siamese network to capture semantic information in the trajectory, and instead of retraining the model, it requires only a few labeled trajectories per user to identify the user category of the trajectory. The experimental results show that the TULSN outperforms existing baselines and state-of-the-art methods on real-world datasets.**

***Index Terms*—Siamese Network, Spatio-temporal data, User identification**

I. Introduction

With the rapid development of mobile computing, satellite positioning techniques, and location-based service (LBS), an increasing amount of geo-tagged mobility data is being generated, including vehicle mobility data, human mobility data, and animal migration data. Such massive mobility datasets contain valuable information that can help explore the migration laws of humans, vehicles, and animals, and analyze the movement characteristics of weather events such as hurricanes. Specifically, such data can be used for applications such as friend recommendations on location-based social networks [1], location prediction [2], [3], and user interest inference.

Linking trajectories to users who generate them is one of the key aspects for the development of many applications. For example, social networks, such as Foursquare and Airbnb, preserve detailed information about restaurants or shopping centers that have been sequentially visited by users. Ride-sharing apps also generate a large amount of trajectories to keep track of the places their users have been to [4]. Thus, users generate an increasing amount of mobility data as they access several online applications. In the task of trajectory semantic inference, for a particular user, it is necessary to combine mobility data under different accounts to form a sequence of complete mobility trajectory. Furthermore, since the spatial trajectory is discretized, the recorded mobility data are a sample of the user's real trajectory. The quality of the trajectory data is affected by the sampling rate, position uncertainty, and preprocessing method. The accuracy of location recommendation, which requires the mining of mobility data, is often affected by the problem of data sparsity. The accuracy can be improved by combining the mobility data generated by a single user from different sources. Therefore, it is particularly important to combine mobility data generated by a user from different sources.

A fundamental and critical task in spatio-temporal data mining is trajectory-user linking (TUL) [5]. The methods used to solve TUL problems can be classified into two types: rule-based methods and learning-based methods. The former measures the similarity in the trajectories using an algorithm such as the longest common sub-sequence (LCSS), dynamic time warping (DTW), or edit distance on real sequence (EDR). The latter makes use of an RNN-based approach to obtain the relationship between the trajectory and user.

However, the current solutions have the following challenges, which largely limit their performance: (1) Considering the large number of users, using conventional approaches to classify the trajectories is inefficient. (2) Rule-based methods cannot effectively capture the semantic information in the trajectories for modeling. (3) In existing learning-based methods, the number of users should be specified before the training process. Once a new user is added, the model needs to be retrained. In this study, we tackle the above challenges in identifying and linking the trajectories of users using a novel Siamese network for trajectory-user linking, termed TULSN. In the TULSN, we design a Siamese network-based framework for an efficient TUL by mining the semantic relationships in the trajectory data. More specifically, we utilize a Siamese recurrent neural network to convert the spatial-temporal trajectory data into an embedded representation space. The Siamese network can make the trajectory embedding distance of the same user as small as possible and the distance of different users as large as possible via backpropagation. We then utilize this embedded representation to determine the similarity

[*]Corresponding author of this work

between the trajectories, and apply the k-nearest neighbors (KNN) algorithm to cluster them into groups. Our model links the trajectories with the users by modeling the semantic relationships between unlabeled and labeled trajectories, so that the identification is possible even when the number of trajectories is small. Moreover, when a new user category is added, only a small amount of geotagged data is required for the TUL.

The following are the main contributions of this study:

- We introduce a novel idea to solve the TUL problem, i.e., a Siamese network that determines the semantic relationship between trajectories. This model combines the advantages of rule-based and learning-based methods, obtains the embedded representation of the trajectory with a learning-based process, and classifies the embeddings into different user categories using the KNN algorithm. This solution can improve the accuracy and efficiency of TUL, without having to retrain the model when a new user category is added.
- We introduce a self-attention mechanism to determine the importance of each point of interest (POI) in the sparse spatio-temporal data. We can make the learning process of the model to focus on the most relevant POI by assigning different weights to the POIs, thereby increasing the recognition accuracy.
- We propose a DR-DeepHash-based method to reduce the dimension of the embedded representation of the mobility data and to encode it into a string of binary encodings. These solutions improve the retrieval efficiency of the user categories of mobility data and reduce storage space.
- We evaluate our approach on two open real-world datasets. The experimental results demonstrate that the proposed method exhibits state-of-the-art performance for the TUL problem in comparison with several existing methods.

The rest of the paper is organized as follows. We give a brief overview of related studies in Section 2. Section 3 presents the definitions of the TUL and trajectory segmentation. Section 4 introduces the architecture and details of the proposed model. Section 5 reports the results of extensive experiments conducted on two real-world datasets. Section 6 summarizes the paper and outlines directions for future work.

II. Related Work

The methods used to solve TUL problems can be divided into two categories: rule-based methods and learning-based methods. The rule-based methods try to find the most relevant trajectory with a known user, and then link the trajectory to its user. The learning-based methods can build a relationship between the trajectories and their users and then classify the unlabeled trajectories into their user categories [4]. The key step in the rule-based approach is to design and define similarity measures for spatio-temporal trajectories. The distance between the semantic feature embeddings of the mobility data can be calculated by defining a distance function, and the corresponding similar trajectories of the query trajectory can be determined. Common distance measurement functions include DTW [6], LCSS [7], EDR [8], and Hausdorff distance [9]. However, these methods were proposed to measure spatio-temporal data; they cannot determine the similarity in the semantic information contained in spatio-temporal data. Therefore, they cannot be directly used to solve the TUL problem. Conventional classification models that can be used to deal with the TUL problem include the KNN [10] and support vector machine (SVM) [11]. These classifiers first encode trajectories into vectors, e.g., one-hot vectors [12] or bag-of-words vectors [13]. Subsequently, they treat different users as different labels and train classification models using training data [4]. However, the conventional methods cannot model the temporal and semantic information contained in spatio-temporal data. This makes them inefficient when it comes to solving the TUL problem.

To this end, a TUL [5] method is proposed, wherein an RNN-based model is used to learn the potential mobility information in the sparse trajectory data at the check-in level. To learn richer semantic information, the TLUTE [4] method has been proposed for TUL; this method takes the spatial, temporal, and categorical information of the trajectory as input and learns the mobility patterns of the user and trajectory. The recently introduced TULAE [14] method uses the variational AutoEncoder to solve the TUL problem in an semi-supervised manner and learns the hierarchical semantic information of the check-ins.

Siamese networks [15]–[17] have been successfully used in image matching and natural language similarity modeling. It can encode complex high-dimensional inputs into low-dimensional embedded expressions, which contain hidden semantics of the input data. However, this method has not been effectively applied to model sparse data of human movement.

III. Preliminaries

In the following, we define the TUL and trajectory segmentation.

A. Problem Definition

Let $T_{u_i} = \{p_{i1}, p_{i2}, \cdots, p_{in}\}$ represent a trajectory generated by a user u_i during a time interval, where $p_{ij} (j \in [1, n])$ is the check-in at time t_j for the user u_i. For spatio-temporal mobility data $T_k = \{p_1, p_2, \cdots, p_q\}$, we determine who generated it, which is called linked. We consider a set of users $\mu = \{u_1, u_2, \cdots, u_n\}$ and an unlinked trajectory dataset $\Gamma = \{T_1, T_2, \cdots, T_q\}$ produced by μ. The solution to the TUL problem provides a map that links the unlinked trajectories to the users: $\Gamma \mapsto \mu$. [5]

B. Trajectory Segmentation

For the ease of calculation, we divide the original trajectory into consecutive sub-trajectories (e.g., in days or months), each of which representing one trip. The objective is to investigate the characteristics of spatio-temporal movement and provide richer knowledge for semantic trajectory analyses such as sub-trajectory pattern mining. The main methods for trajectory segmentation can be classified into three basic strategies: time threshold, geometric topology, and trajectory semantics [18]. In this paper, we use a time threshold-based approach. We divide the original trajectory T_{u_i} into k consecutive subsequences $T_{u_i}^1, \cdots, T_{u_i}^k$ based on time intervals of 6 h [5].

IV. Proposed Method

In this paper, a TULSN algorithm is proposed to solve the TUL problem. Figure 1 shows an overview of the TULSN, which is composed of training and identification processes. With trajectory segments as input, the training process consists of three parts: (1) Two sub-networks in the Siamese network sharing the same weight use the LSTM to obtain the semantic representation of the trajectory segment. (2) A self-attention module is introduced to assign different weights to each POI. (3) A DR-DeepHash module is proposed to generate a binary hash code to reduce time and space complexities. The identification process consists of two parts: (1) using the Siamese network to obtain the hash embedding expression of the trajectory segment; (2) using a KNN as the classifier to identify the user category of the unknown trajectory segment. In the following, the implementation details of each part are presented.

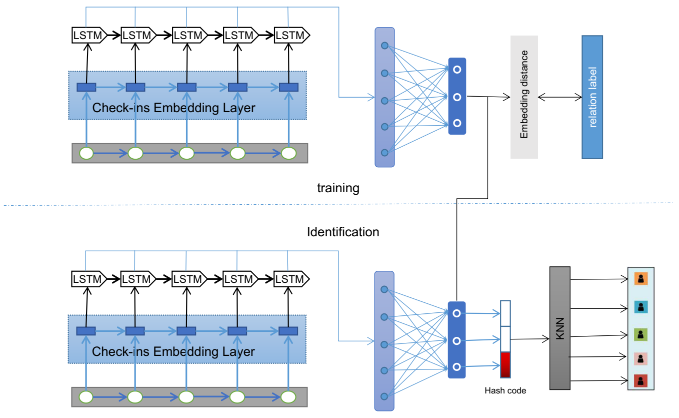

Fig. 1. Overview of TULSN: TULSN first uses semantic similarities between the trajectories to identify user categories. Top: The Siamese network is used to embed the trajectory pairs to learn the semantic relationship between the trajectories. Bottom: The sub-network in the Siamese network is used to embed the unlabeled trajectory, and the semantic similarity is compared with the embedded characterization of the labeled trajectory. The KNN is used for user identification of the trajectory.

A. Siamese Network

A Siamese network has two sub-networks, each of which encodes a value in a tuple. In this paper, a tuple consists of two trajectories, which belong to either the same user or different users. The two sub-networks share the same weights and the same architecture in the Siamese architecture. Each sub-network learns from variable-length original trajectory segments to fixed-length vector space. Specifically, each trajectory segment (POI vector sequence) is inputted to the sub-network, and the sub-network updates the hidden state of each unit. The final representation of the trajectory segment is coded by the last hidden state of the sub-network. Given a pair of trajectory segments, the semantic similarity in the trajectory segments is inferred by determining the similarity in the vector space.

The training samples used to train the Siamese network are in the form of tuples (x_1, x_2, y), The label y = 0 indicates that x_1 and x_2 are of different types, and y=1 indicates that x_1 and x_2 are of the same type. Two inputs x_1 and x_2 are received and then converted into vectors $v(x_1)$ and $v(x_2)$, respectively, using the LSTM or GRU. During the training of the Siamese network, the error of backpropagation comes from the similarity between $v(x_1)$ and $v(x_2)$ and the degree of deviation between the predicted similarity and the real label. During the training process, this will force the sub-network to capture the semantic differences in the trajectory segments. The distance D between the two output vectors is calculated using a certain distance metric, and the loss is calculated using the D value to train the Siamese network.

B. TULSN with a Self-attention Mechanism

Neural networks with attention mechanisms have achieved significant results in many natural language processing (NLP) tasks. In this study, we introduce a self-attention mechanism [19] into TULSN for a more accurate semantic understanding of the trajectory segments.

The LSTM model [20] (or GRU [21]) uses the hidden state of the last layer as the semantic embedded representation of the trajectory segment. However, this method considers that the semantic information in the different POIs is equally important for identifying the user's category. Moreover, the method ignores a significant amount of information about the POIs. Based on this, we utilize the self-attention mechanism to take all the information of the hidden state $H = \{H_1, H_2, \cdots, H_n\}$ as the output, and assign different weights to the different POIs. For example, different users have different preferences for different POIs. Here, we suppose that user u has a certain degree of preference for POI_1, POI_2, POI_3, and POI_4, and that given trajectory segments T1 and T2 are generated by u. With higher weights assigned to the preferred POIs, the self-attention mechanism can extract some information from the trajectories that can best distinguish whether T1 and T2 belong to the same user. The model is mathematically described as follows:

Suppose we have a sequence of trajectory segments whose length is p, then the trajectory segments can be represented as follows:

$$T = \{POI_1, POI_2, \cdots, POI_p\} \tag{1}$$

where POI_i is a POI in the trajectory segment, and the embedded expression of each POI is obtained using the word embedding method in NLP. We use the bidirectional LSTM to obtain the long-distance dependence information in the trajectory sequence:

$$h_{LR}(t) = LSTM_{LR}(h_{LR}(t-1), w(t), b(t)) \tag{2}$$

$$h_{RL}(t) = LSTM_{RL}(h_{RL}(t+1), w(t), b(t)) \tag{3}$$

To obtain more semantic information in the trajectory segments, $h_{LR}(i)$ and $h_{RL}(i)$ are connected to obtain h_i. We take all h_i as the output of the hidden layer:

$$H = \{h_1, h_2, \cdots h_n\} \tag{4}$$

In the self-attention mechanism, all the hidden layers in *H* are taken as the input, and the final expression of the trajectory sequence is calculated using the following formula:

$$M = tanh\left(w_1 H\right) \tag{5}$$

$$\alpha = softmax\left(w_2 M\right) \tag{6}$$

$$r = H\alpha \tag{7}$$

where w_1 is a weight matrix, which can be used to set the number of POIs related to the trajectory sequence, w_2 is the parameter vector, α is the weight of the different POIs in the trajectory sequence, and r is the final semantic embedding vector.

C. DR-DeepHash

We use the TULSN to represent the trajectories as embedded representations by extracting the semantic information from the trajectories. The challenge that needs to be overcome in the proposed method is the high time and space complexities in the semantic similarity retrieval of trajectories. In a TUL task, it is particularly important to quickly find the trajectory of the most relevant known user for the unknown user trajectory. To improve the recognition efficiency of the trajectories, the DR-DeepHash method is used to reduce the dimension of the embedded expression and convert the embedded expression into a binary coding. The objectives of the module are as follows: (1) The recognition efficiency should be effectively improved via dimension reduction. (2) The binary coding of the trajectories with similar semantics should be as close as possible, and the binary coding of the trajectories whose semantics are not similar should be kept as far as possible. (3) Through binary coding, the embedded expression of the trajectory has a smaller storage space, and this method improves the retrieval efficiency.

Given a trajectory for which the user is unknown, the objective of our method is to find a mapping relationship and map the semantic information of the trajectory to the hash space. To obtain the semantic information of the trajectory, we use LSTM or GRU to embed the trajectory, use dimension reduction to reduce the dimensionality of the embedded expression, and use DeepHash to quantify the embedded expression, so that the output of the network is a binary coding. The embedded expression of the trajectory can be described as follows: $b = \{0/1\}^p$. Next, we discuss the DeepHash method and DR method used in the training and identification processes in the TULSN.

1) *TULSN with Dimension Reduction:*

For the dimension reduction of the embedded expressions of the trajectories, we use a method similar to the fully connected layer in a CNN [22], and the recognition efficiency is improved. The principle of the method is as follows:

$$\tilde{v}_{u_i} = v_{u_i} w_{rd} + b_{rd} \tag{8}$$

where v_{u_i} is a high-dimensional semantic feature vector extracted by the LSTM network, w_{rd} is a weight matrix, and b_{rd} is an offset value. Its role is to map the high-dimensional semantic feature vector into the low-dimensional space while ensuring that it still contains rich semantic information.

Assuming a high-dimensional semantic feature vector $V = [v_1, v_2, v_3, \cdots v_m]$, we can obtain the semantic expression in a low-dimensional space via matrix calculation:

$$W = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ w_{21} & w_{22} & \cdots & w_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ w_{m1} & w_{m2} & \cdots & w_{mn} \end{bmatrix}$$

$$[\tilde{v_1}\, \tilde{v_2}\, \cdots\, \tilde{v_n}] = [v_1\, v_2\, v_3\, \cdots\, v_m] * W + [bv_1\, bv_2\, \cdots\, bv_n] \tag{9}$$

where m is the length of the high-dimensional semantic feature vector, and n is the length of the transformed low-dimensional feature vector, $m > n$.

2) *TULSN with DSH:*

To map the embedded representation of the trajectory into a binary code, the DSH method [23] is implemented in TULSN to improve the recognition efficiency, as follows:

In the network training process, the distance between the embedded expression of the trajectory segments is calculated using the formula reported in 10.

$$D\left(b_1, b_2\right) = \frac{\|b_1 - b_2\|_2^2}{\|b_1\|_2^2 + \|b_2\|_2^2} \tag{10}$$

where b_1 and b_2 are the semantic feature embeddings of two trajectory segments; the Euclidean distance is used to measure the distance between the embedded expressions; and in order to map the distance measure to the label, a normalized process is used for the distance measure. The label refers to whether the two trajectory segments belong to the same user.

Next, use the formula given in 11 to calculate the loss in the distance between the semantic feature embeddings of the two trajectories and the label.

$$\Gamma = \frac{1}{2} y D\left(b_1, b_2\right) + \frac{1}{2}\left(1 - y\right) max\left(\left(1 - D\left(b_1, b_2\right)\right), 0\right) \tag{11}$$

When y=0, the two trajectory segments are not similar, and the loss is $1 - D\left(b_1, b_2\right)$. When y=1, the two trajectory segments are similar, and the loss is $D\left(b_1, b_2\right)$.

To improve the recognition efficiency of the trajectory segment and reduce the storage consumption, a regularization method is used, and the formula is as shown in 12.

$$R = \frac{\alpha\left(\||b_1| - 1\|_1 + \||b_2| - 1\|_1\right)}{\|b_1\|_2^2 + \|b_2\|_2^2} \tag{12}$$

where α is the regularization weight.

Finally, the loss function of the network is expressed in equation 13.

$$L\left(b_1, b_2, y\right) = \Gamma + R \tag{13}$$

D. Training and Identification

When constructing the training dataset, the ratio of the trajectory pairs of the same user to those of different users must be 1:1, in order to ensure a data balance between the positive and negative samples in the dataset. The TULSN model does not have more preferences for the same user (or different users). To solve this problem, the original data are negatively sampled to obtain the training dataset and more semantic information. Further, we use the KNN to identify the unlabeled trajectory segments based on the similarity of the semantics embedded between the trajectory segments, and compare the unlabeled trajectory segments with a few trajectory segments that have already been labeled.

V. Experiments

We evaluate our proposed model on two real-world datasets. The experimental setup and parameter settings are rst introduced. We then compare the accuracy and efficiency of the TULSN with various baselines, and analyze the effects of different parameters on the experimental results.

A. Experiment setup

1) Dataset: To demonstrate the performance of TULSN, we conduct extensive experiments on two real-world datasets: Gowalla and Brightkite [24]. The datasets contain user categories and check-in trajectories generated by users. We randomly select 201 users from Gowalla and 92 users from Brightkite. For each user, we use the time-interval based approach to partition the original check-in trajectories. Table I lists the details of the two datasets [5].

TABLE I
Data description: U: number of users; L/UL: number of labeled and unlabeled trajectories (number of verification sets); C: number of POIs; R: average length of original trajectories; T: length range of sub-trajectories after segmentation

Dataset	U	L/UL	C	R	T
Gowalla	201	18,654/2,072	11,846	219	[1,131]
Brightkite	92	17,623/1,958	2,115	471	[1,184]

We split the datasets into two. The first 90% of each user is selected as the training datasets by taking random samples from the original datasets, and the remaining 10% is for the test datasets.

2) Baseline methods: We compare the TULSN model with several existing baselines, which can be categorized into rule-based methods and learning-based methods. The rule-based methods mainly use trajectory similarity metrics, whereas learning-based methods use trajectory classification techniques such as machine learning or deep learning. The existing solutions can be classified as follows:

(1) Rule-based methods include DTW, LCSS, Fréchet distance [25], and ERP [26]. These employ different similarity metrics to calculate the distance between two trajectories.

(2) Learning-based methods include linear discriminant analysis (LDA) [27], SVM, TULER and its variants, and TULVAE. These methods utilize classification models to learn the relationship between the trajectories and the corresponding users.

3) Evaluation Platform: Our method is implemented in Python 2.7 and TensorFlow 1.4.0. The platform runs on the Ubuntu 16.04 operating system with an Intel Xeon E5-1620 CPU and TiTan V GPU.

B. Parameter settings

TABLE II
Parameter value in TULSN training

parameters	Model use	Optional range
POI embedded dimension	250	100-300
Dropout rate	0.5	0-1
Number of hidden layers	300	250-1000
batch_size	128	32-256
Number of LSTM layers	1	>=1
LSTM layer structure	BiLSTM	BiLSTM,LSTM
Learning rate	0.01(decays with a rate of 0.96)	0.001-0.1

Table II lists the possible range of values of the different parameters and the values of the parameters used in the experiment.

C. Metrics

We adopt two metrics to evaluate the performance of our model, the ACC @ K and macro-F1. Among them, the ACC is used to evaluate the accuracy of TUL, and the formula is as follows:

$$ACC@K = \frac{correctly\ linked\ trajectories@K}{the\ number\ of\ trajectories}$$

The F1 value is the harmonic mean of the precision (macro-P) and recall (macro-R). The formula is as follows:

$$macro-F1 = 2 \times \frac{macro-P \times macro-R}{macro-P + macro-R}$$

D. Performance evaluation

1) Experiment 1: Table III and Table IV summarize the effects of various TUL solutions on the two datasets, respectively, where the best solution is boldfaced. We use the KNN as a classifier to perform user identification of the trajectories. Taking the first 90% of each user as the training datasets, we select some or all of the trajectory datasets in the training datasets as the labeled sample sets. Given an unlabeled trajectory, the semantic relationship between the unlabeled and labeled trajectories in the sample sets is determined, and the user category of the unlabeled trajectory is judged based on the user category of the trajectory in the sample sets, i.e., the category of the trajectory is obtained by one or more labeled trajectories in which the unlabeled trajectory is semantically similar in the sample sets. When calculating ACC@1, we set K in the KNN to 1, i.e., the user category of the labeled trajectory whose sample sets is closest to the unlabeled trajectory is the user category of the unlabeled trajectory. When calculating ACC@5, we set K in the KNN to 5, obtain the top-five labeled trajectories of the unlabeled trajectory that are closest in the sample sets, and determine whether the user category of the unlabeled trajectory is in the top-five trajectories. If so, it is considered correct.

TABLE III
PERFORMANCE ON THE GOWALLA DATASET

Method	ACC@1	ACC@5	Macro-F1
LDA	38.28	48.37	35.18
SVM	42.26	55.30	35.47
TULER-LSTM	47.13	62.05	37.07
TULER-LSTM-S	43.58	57.13	35.43
Bi-TULER	48.70	66.38	36.56
HTULER-L	43.30	60.90	35.92
TULVAE	49.50	64.39	36.41
TULSN	72.96	89.47	69.00
TULSN-A	**75.10**	**89.97**	**71.13**

TABLE IV
PERFORMANCE ON THE BRIGHTKITE DATASET

Method	ACC@1	ACC@5	Macro-F1
LDA	41.50	51.38	40.38
SVM	43.17	60.06	39.59
TULER-LSTM	45.10	63.04	37.18
TULER-LSTM-S	44.19	68.46	39.71
Bi-TULER	44.81	62.92	40.20
HTULER-L	45.16	64.55	39.89
TULVAE	47.98	68.48	45.32
TULSN	64.70	81.76	59.11
TULSN-A	**65.49**	**82.75**	**60.91**

The above experimental results show that the TULSN consistently outperforms the baselines in terms of the accuracy on the two datasets. The TULSN with the self-attention mechanism (TULSN-A) achieves the best results in terms of ACC@1, ACC@5, and Macro-F1. For example, on the Gowalla dataset, the TULSN-A method exhibits improvements of 25.6%, 23.59%, and 34.06% for the ACC@1, ACC@5, and Macro-F1 metrics, respectively, relative to the TULVAE method. Similarly, the evaluation results on the Brightkite dataset demonstrate the superiority of our model. Compared with the baselines, the advantage of the TULSN is that it utilizes semantic information to measure the similarity between trajectories, so that the user category can be recognized more accurately.

The TULSN-A outperforms the TULSN. This is because in TULSN-A, different weights are assigned to the POIs, thus ensuring that the model focuses on the most important semantic information of the trajectories for user category recognition. Moreover, the model has a higher convergence rate after introducing the self-attention mechanism.

2) Experiment 2: The TULSN solves the TUL problem by capturing the semantic relationship between the trajectories. Therefore, when a new user joins, we only need to obtain a few labeled trajectories for this user. With this, we can determine the user corresponding to this trajectory through the semantic similarity between the trajectories. Moreover, the TULSN method does not require retraining the new model when a new user arrives, and the model trained on the trajectory sets of other users can be used for identifying a new user. This is because the TULSN identifies the user category based on the internal semantic similarity between the trajectories, instead of judging the user category from the relationship between the trajectory and the user. New users often keep joining in the real world. If the model is retrained each time, the cost will be considerable, and when a new user joins, the original method requires a large amount of labeled data from the user to be retrained; however, labeled data are difficult to obtain because of privacy.

TABLE V
USER CATEGORY BETWEEN TRAINING AND VERIFICATION SETS ARE DIFFERENT

Method	ACC@1	ACC@5	Macro-F1
DTW-KNN	26.98	30.88	21.81
Fréchet-KNN	26.74	30.10	21.31
ERP-KNN	27.28	31.08	21.93
LCSS-KNN	29.27	32.27	24.70
TULSN	**73.89**	**89.33**	**72.06**

Table V shows the accuracy of TUL when the user categories between the training and verification sets are different. The model is trained using the trajectory data of the other 201 users on the Gowalla dataset as the training set. The latter 10% of the 201 users in experiment 1 are used as the verification set to verify the method. Existing RNN-based models cannot solve this problem. In this experiment, unsupervised methods, including DTW, Fréchet, ERP, and LCSS, are compared with the TULSN method. The unsupervised methods identify the category based on the similarity measure of the trajectories, whereas the TULSN method identifies the category based on the semantic similarity in the trajectories. The experimental results demonstrate the superiority of the TULSN method. Table V shows that the TULSN method achieves good experimental results without having to retrain the model when a new user arrives.

E. Parameter study

In this experiment, we investigate the effects of the K value in the KNN and the sampling rate of the labeled sample sets on the performance of the TULSN. The labeled sample sets contain trajectories of known users. These trajectories are used to identify the categories of trajectories of unknown users. In this experiment, the Gowalla dataset is used to study the related parameters. As shown in Figure 2, the K value increases, whereas the ACC@1 value decrease. Because of the similarity between the users, as the K value increases, it becomes easier to link the trajectory to a similar user rather than to the user who generated it. Moreover, the experimental results show that the value of ACC@1 increases with an increase in the number of labeled sample sets. When the proportion of the sample set is 30%, the value of ACC@1 on the verification set is still better than that obtained by TULVAE.

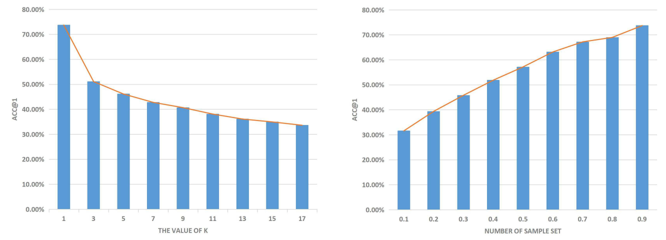

Fig. 2. Effect of K values of the KNN and the sampling rate of labeled sample sets on ACC@1

F. Effectiveness study

Although the TULSN achieves better accuracy, its time and space requirements can still be improved. To this end, we propose a DR-DeepHash method, which reduces the storage space and increases the recognition speed by reducing the dimensionality of the trajectory embedded expression and converting it into a binary code.

We further verify the efciency of the TULSN and TULSN with DR-DeepHash on the Gowalla dataset. In the experiment, we validate using the last 10% of the dataset as the validation set. The validation dataset contains a total of 2,153 trajectory records. As shown in Figure 3, the TULSN with DR-DeepHash has a significant reduction in both the time and space complexities compared to the TULSN. The TULSN requires 177 s to identify the user category in the verification set, whereas the improved TULSN with DR-DeepHash requires only 92 s. When using the TULSN, the storage space occupied by the embedded expression of the sample set is 171 M, and the storage space when using the improved method is 6.9 M. Although the recognition speed has been improved, there are some drawbacks compared to learning-based methods such as SVM, TULER, and TULVAE.

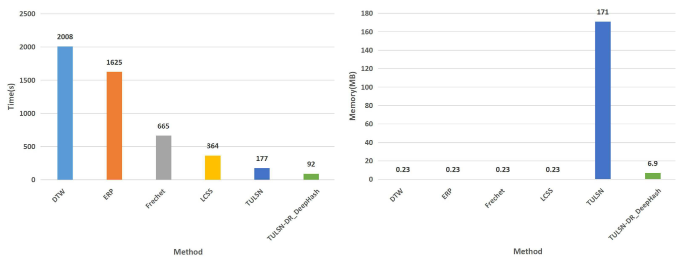

Fig. 3. Time and space complexities of different methods

VI. Conclusions

In this paper, we proposed a TULSN model to solve the TUL problem. Compared with existing methods that use either the conventional trajectory similarity measures or the RNN-based classification methods, the TULSN achieves better performance as it learns the semantic relationships between trajectories. We incorporated an attention mechanism into the TULSN in order to focus on the most relevant POIs in the input data. Furthermore, the time and space complexities of our solution can be significantly improved by introducing a binary hash mode in its design.

VII. Acknowledgments

This work was partially supported by NSFC under grant numbers 61902376 and 61702487. This work was also financially supported by National Key Research and Development Program of China under grant number 2018YFC1407400.

References

[1] X. Xiao, Y. Zheng, Q. Luo, and X. Xie, "Inferring social ties between users with human location history," *J. Ambient Intelligence and Humanized Computing*, vol. 5, no. 1, pp. 3–19, 2014. [Online]. Available: https://doi.org/10.1007/s12652-012-0117-z

[2] C. Cheng, H. Yang, M. R. Lyu, and I. King, "Where you like to go next: Successive point-of-interest recommendation," in *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*, 2013, pp. 2605–2611. [Online]. Available: http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6633

[3] Q. Liu, S. Wu, L. Wang, and T. Tan, "Predicting the next location: A recurrent model with spatial and temporal contexts," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, 2016, pp. 194–200. [Online]. Available: http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11900

[4] G. Wang, D. Liao, and J. Li, "Complete user mobility via user and trajectory embeddings," *IEEE Access*, vol. 6, pp. 72 125–72 136, 2018. [Online]. Available: https://doi.org/10.1109/ACCESS.2018.2881457

[5] Q. Gao, F. Zhou, K. Zhang, G. Trajcevski, X. Luo, and F. Zhang, "Identifying human mobility via trajectory embeddings," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, 2017, pp. 1689–1695. [Online]. Available: https://doi.org/10.24963/ijcai.2017/234

[6] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," in *Knowledge Discovery in Databases: Papers from the 1994 AAAI Workshop, Seattle, Washington, USA, July 1994. Technical Report WS-94-03*, 1994, pp. 359–370.

[7] M. Vlachos, D. Gunopulos, and G. Kollios, "Discovering similar multidimensional trajectories," in *Proceedings of the 18th International Conference on Data Engineering, San Jose, CA, USA, February 26 - March 1, 2002*, 2002, pp. 673–684. [Online]. Available: https://doi.org/10.1109/ICDE.2002.994784

[8] L. Chen, M. T. Özsu, and V. Oria, "Robust and fast similarity search for moving object trajectories," in *Proceedings of the ACM SIGMOD International Conference on Management of Data, Baltimore, Maryland, USA, June 14-16, 2005*, 2005, pp. 491–502. [Online]. Available: https://doi.org/10.1145/1066157.1066213

[9] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. J. Keogh, "Querying and mining of time series data: experimental comparison of representations and distance measures," *PVLDB*, vol. 1, no. 2, pp. 1542–1552, 2008. [Online]. Available: http://www.vldb.org/pvldb/1/1454226.pdf

[10] O. Kwon and J. Lee, "Text categorization based on k-nearest neighbor approach for web site classification," *Inf. Process. Manag.*, vol. 39, no. 1, pp. 25–44, 2003. [Online]. Available: https://doi.org/10.1016/S0306-4573(02)00022-5

[11] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995. [Online]. Available: https://doi.org/10.1007/BF00994018

[12] S. Lai, K. Liu, S. He, and J. Zhao, “How to generate a good word embedding,” *IEEE Intelligent Systems*, vol. 31, no. 6, pp. 5–14, 2016. [Online]. Available: https://doi.org/10.1109/MIS.2016.45

[13] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” in *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013. [Online]. Available: http://arxiv.org/abs/1301.3781

[14] F. Zhou, Q. Gao, G. Trajcevski, K. Zhang, T. Zhong, and F. Zhang, “Trajectory-user linking via variational autoencoder,” in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, 2018, pp. 3212–3218. [Online]. Available: https://doi.org/10.24963/ijcai.2018/446

[15] J. Mueller and A. Thyagarajan, “Siamese recurrent architectures for learning sentence similarity,” in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, 2016, pp. 2786–2792. [Online]. Available: http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12195

[16] *23rd International Conference on Pattern Recognition, ICPR 2016, Cancún, Mexico, December 4-8, 2016*. IEEE, 2016. [Online]. Available: https://ieeexplore.ieee.org/xpl/conhome/7893644/proceeding

[17] *Signature Verification Using a Siamese Time Delay Neural Network*, 1993. [Online]. Available: http://papers.nips.cc/paper/769-signature-verification-using-a-siamese-time-delay-neural-network

[18] Y. Zheng, “Trajectory data mining: An overview,” *ACM TIST*, vol. 6, no. 3, pp. 29:1–29:41, 2015. [Online]. Available: https://doi.org/10.1145/2743025

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” *CoRR*, vol. abs/1706.03762, 2017. [Online]. Available: http://arxiv.org/abs/1706.03762

[20] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[21] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” *CoRR*, vol. abs/1412.3555, 2014. [Online]. Available: http://arxiv.org/abs/1412.3555

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, 2012, pp. 1106–1114. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks

[23] H. Liu, R. Wang, S. Shan, and X. Chen, “Deep supervised hashing for fast image retrieval,” *International Journal of Computer Vision*, vol. 127, no. 9, pp. 1217–1234, 2019. [Online]. Available: https://doi.org/10.1007/s11263-019-01174-4

[24] E. Cho, S. A. Myers, and J. Leskovec, “Friendship and mobility: user movement in location-based social networks,” in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, August 21-24, 2011*, 2011, pp. 1082–1090. [Online]. Available: https://doi.org/10.1145/2020408.2020579

[25] A. D. Chouakria and P. N. Nagabhushan, “Improved fréchet distance for time series,” in *Data Science and Classification*, 2006, pp. 13–20. [Online]. Available: https://doi.org/10.1007/3-540-34416-0_2

[26] L. Chen and R. T. Ng, “On the marriage of lp-norms and edit distance,” in *(e)Proceedings of the Thirtieth International Conference on Very Large Data Bases, VLDB 2004, Toronto, Canada, August 31 - September 3 2004*, 2004, pp. 792–803. [Online]. Available: http://www.vldb.org/conf/2004/RS21P2.PDF

[27] J. Ye, R. Janardan, and Q. Li, “Two-dimensional linear discriminant analysis,” in *Advances in Neural Information Processing Systems 17 [Neural Information Processing Systems, NIPS 2004, December 13-18, 2004, Vancouver, British Columbia, Canada]*, 2004, pp. 1569–1576. [Online]. Available: http://papers.nips.cc/paper/2547-two-dimensional-linear-discriminant-analysis