A Fast Unsupervised Spatial Temporal Trajectory Similarity Measure Based on Trajectory Image Structure Matching

Xiaolin Chang
Beijing University of Technology
Chaoyang Qu, Beijing Shi, China
changxl@emails.bjut.edu.cn

Shaofu Lin
Beijing University of Technology
Chaoyang Qu, Beijing Shi, China
linshaofu@bjut.edu.cn

Xiliang Liu*
Beijing University of Technology
Chaoyang Qu, Beijing Shi, China
liuxl@bjut.edu.cn

ABSTRACT

In order to perform the spatial temporal trajectory similarity measure quickly and accurately, for the characteristics of large amount of trajectory data, existence of spatial temporal heterogeneous distribution and obvious noise, a fast computational method TISM-CAE for trajectory image structure matching combined with convolutional auto-encoder is proposed. Firstly, the given spatial temporal trajectory slices are remapped into a two-dimensional matrix. Secondly, the low-dimensional features of the trajectory images are obtained in an unsupervised learning manner using a convolutional auto-encoder model. Finally, the trajectory similarity is equivalent to compare the similarity between the low-dimensional features. The real floating vehicle dataset in Shanghai and the artificially simulated similar trajectory dataset are used for experimental analysis. Final results show that the proposed method has a large improvement in accuracy and time complexity compared with the current commonly used methods LCSS and EDR, providing a feasible path for fast and accurate analysis of massive spatial temporal trajectories.

KEYWORDS

spatial temporal trajectory, similarity measure, convolutional auto-encoder, floating vehicle data

ACM Reference Format:
Xiaolin Chang, Shaofu Lin, and Xiliang Liu. 2022. A Fast Unsupervised Spatial Temporal Trajectory Similarity Measure Based on Trajectory Image Structure Matching. In *Proceedings of of 3st ACM SIGKDD Workshop on Deep Learning for Spatiotemporal Data, Applications, and Systems (DeepSpatial '22).* ACM, New York, NY, USA, 8 pages. https://doi.org/0.0

1 INTRODUCTION

With the increasing maturity of wireless communication technology, positioning technology, sensor technology and real-time video acquisition technology, the acquisition of spatial temporal trajectory data has become more and more convenient [31]. These data integrate spatial information, temporal information and attribute information, and contain rich and diverse semantic information and behavioral patterns. Therefore mining and analysis of spatial temporal trajectory data has become an important research topic in the field of data mining. Spatial temporal trajectory data mining is mainly divided into four categories [26], such as spatial temporal trajectory clustering [21] [22], concomitant pattern mining [10], frequent pattern mining [13] and trajectory classification. Spatial temporal trajectory clustering refers to dividing spatial temporal objects with similar behaviors into the same group based on similarity measure, so that the difference between groups is as large as possible and the difference within groups is as small as possible. Trajectory classification refers to building a model based on the similarity measure between training trajectory data, by which the class of a trajectory can be predicted. One of the most critical methods for these studies is the trajectory similarity measure [28].

As an important tool for trajectory data mining, trajectory similarity measure is widely used in various fields of real-world trajectory computing. For example, in the field of intelligent recommendation, merchants can analyze the hobbies and interests of certain types of users by finding similar activity trajectories that satisfy certain spatial and temporal constraints, so as to make targeted recommendations and improve users' experience and stickiness [19]. In the field of smart traveling, users' travel time can be reasonably planned by drawing on their similar trajectories, which provides the possibility of smart travel [23]. In the field of infectious disease prevention and control, especially in the period of local outbreaks and sporadic distribution of the COVID-19, the trajectory similarity measure plays an important role in finding the spatial and temporal concomitants of confirmed cases in the massive trajectory data [20]. However, the trajectory data from various sources are affected by spatial and temporal distribution, sampling equipment, communication failures and other factors leading to uneven sampling, large data volume, the presence of noise and other characteristics, which bring great challenges to the efficiency and accuracy of similarity measure between two trajectories [27] [14].

For different application scenarios and actual needs, a series of different similarity measures have been proposed. Various trajectory similarity measures have their own different delineations for trajectory similarity [12]. With the idea of similar image matching in computer vision, this paper proposes a fast computational method of trajectory image structure matching based on convolutional auto-encoder(TISM-CAE), which can evaluate trajectory similarity measure with lower time complexity and higher accuracy on trajectory data sets under scenarios of uneven sampling, large data volume, the presence of noise and so on. The main contributions of this paper are as follows.

*

DeepSpatial '22, August, 2022, Virtual

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/0.0

1. We map the specified spatial temporal trajectory slices into trajectory images and employ the idea of image processing to measure the similarity between trajectories. The trajectory images have stronger ability to recognize the geometry and position of trajectories.

2. We use convolutional auto-encoder network to rapidly extract trajectory image features with an unsupervised learning approach, without the need for complex deep learning network frameworks and training data with labels.

3. We perform extensive experimental analysis using real floating vehicle dataset of Shanghai as well as a manually simulated trajectory dataset, the results show that the proposed method has lower time complexity and higher accuracy.

The rest of this paper is organized as follows: Section 2 describes the related works on trajectory similarity measure. Section 3 introduces the research methodology proposed in this paper. Section 4 discusses experimental data as well as extensive experimental analysis. Section 5 gives the discussion and conclusion.

2 RELATED WORKS

Trajectory similarity measure is the basis of trajectory data mining. As spatial temporal trajectory data integrates temporal and spatial information, the similarity between trajectories cannot be directly measured by common similarity measure methods such as cosine similarity, Jaccard similarity and Euclidean distance. Although [1] utilizes the Euclidean distance to calculate the trajectory similarity, it requires a high quality of trajectories, which requires the same number of trajectory points as well as the same sampling interval. Experts and scholars home and abroad have conducted a lot of research on trajectory similarity measure. Since trajectories are mainly stored as trajectory points, when similarity measure methods are performed on trajectories, the most intuitive way is to use the distance between corresponding points in two trajectories to measure the similarity between trajectories. The most commonly used ones in the existing literature include the Dynamic Time Warping algorithm (DTW) [5], the Longest Common Subsequence algorithm (LCSS) [4] and the Edit Distance on Real sequence algorithm (EDR) [3]. Dynamic Time Warping algorithm (DTW) achieves local stretching or scaling of the trajectories by copying trajectory points, so that the similarity can be measured for trajectories with different sampling rates or different lengths, and in order to improve the computational efficiency, DDTW [8] and ACDTW [9] are proposed by segmenting the trajectories; Longest Common Subsequence algorithm (LCSS) mainly considers the similar part between trajectories as the trajectory similarity measure, so there is no requirement on the trajectory sampling rate and trajectory length. Edit Distance on Real sequence algorithm (EDR) eliminates the effect of noise by quantizing the distance between trajectory points as two values of 0 or 1. However, most of these similarity measures need to match each point in the trajectory, and the time complexity of these methods is $O(m * n)$, where m and n are the number of trajectory points in two trajectories. In addition, more improvements based on several basic algorithms mentioned above [2] [6] are used to solve the problems of high time complexity, sensitivity to uneven sampling and noise in these similarity measure algorithms.

With the successful application of deep learning in image processing, voice recognition, natural language processing and other fields, trajectory similarity measure methods based on deep learning have been gradually developed. [11] introduces word vectors in natural language processing into the representation of trajectories and proposes a method t2vec based on seq2seq representation of trajectories for the first time, which can achieve accurate and efficient trajectory search in uneven sampling and noisy trajectory data. However this method is built on top of recurrent neural network (RNN), and the construction of the network framework is complicated and the learning rate is slow. [25] proposes a model framework called at2vec to represent the similarity between two trajectories, but the model needs to introduce POI semantic information.

In summary, the trajectory similarity measure based on trajectory points is affected by uneven sampling and noisy points. When the number of sampling points between two trajectories is inconsistent due to uneven sampling, it is challenging to align the trajectory points with their corresponding matching points. When there are noisy points, the final result of similar trajectories is affected by the distance threshold, although they can be eliminated by setting the distance threshold. The trajectory similarity measure based on deep learning is robust to uneven sampling and noisy sampling points, but the network framework is complex to construct, requiring a large amount of labeled data for training, and has a slow learning rate. Inspired by deep learning, this study uses a convolutional auto-encoder network to quickly, efficiently and accurately match similar trajectories in an unsupervised learning manner without complex deep learning network framework and labeled training data.

3 METHODOLOGY

3.1 Trajectory Image Construction

A trajectory is a series of time-stamped point records used to describe the motion of moving objects such as people, vehicles, animals and natural phenomena. Theoretically, since the motion of moving objects is continuous, the resulting trajectories should also be continuous records. However, in reality, the limitations of positioning technology lead to the fact that the current position of moving objects can only be collected periodically, so the phenomenon of continuous motion of moving objects can only be collected and stored in the form of discrete points. These discrete point records are represented by geographic coordinate system (GCS), longitude and latitude. The symbolic concept of trajectory in GCS is defined in this paper as follows:$T_{GCS} = \{(Point_i, time_i)|1 \leq i \leq n\}$ where n represents the number of trajectory points contained in the trajectory, $time_i$ represents the moment of the i^{th} trajectory point, and $Point_i$ represents the position of the i^{th} trajectory point. $Point_i = (lon_i, lat_i)$, where lon_i represents the longitude of the i^{th} trajectory point, lat_i represents the latitude of the i^{th} trajectory point.

Since the Earth is approximately an ellipsoid, the geographical coordinates expressed by longitude and latitude cannot be calculated directly in terms of area, distance or direction using the plane geometry formula. Firstly, the geographical coordinates need to be projected on the map. Considering that the study area Huangpu

District of Shanghai is in a low latitude area, Web Mercator is used for calculation in this paper [7], as shown in the following equation:

$$X_i = R * \Pi * lon_i/180 \tag{1}$$

$$Y_i = R/2 * log(\frac{(1 + sin(lat_i/180 * \Pi))}{(1 - sin(lat_i/180 * \Pi))}) \tag{2}$$

Where the radius of the earth $R = 6378137m$, X_i represents the projection coordinates corresponding to the geographic coordinate of longitude in the i^{th} trajectory point, Y_i represents the projection coordinates corresponding to the geographic coordinate of latitude in the i^{th} trajectory point, the trajectory in the geographic coordinates corresponding to the trajectory T_{GCS} in Web Mercator projection coordinates is represented as $T_{MCT} = \{(mctXY_i, time_i)|1 \leq i \leq n\}$, where $mctXY_i = (X_i, Y_i)$.

Next, the Web Mercator projection coordinates are mapped into the set two-dimensional matrix with certain mathematical rules to generate the trajectory image, as shown in Fig. 1.

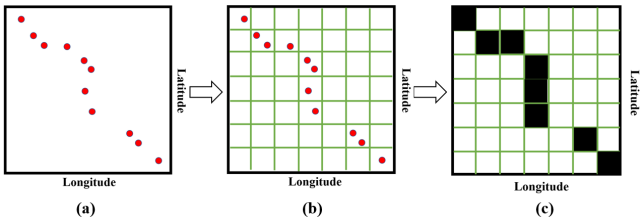

Figure 1: Schematic diagram of geographic coordinates to projection coordinates.

In the first step, the selected spatial region is divided into a limited number of grids, as shown in Fig. 1(b). The study area is divided into grids with the single size of the same length and height as m, where m denotes the resolution of the trajectory image. The second step is to convert the Web Mercator projection coordinate points into coordinates in the divided grid according to a certain mathematical formula, which is equivalent to the coordinates in a two-dimensional matrix. In order to ensure that the trajectory image generated by the two-dimensional matrix is visually consistent with the direction in the actual situation, the origin of coordinates is set as the upper left corner of the two-dimensional matrix and noted as $O(X_0, Y_0)$ in this paper. The row and column values of coordinates in the two-dimensional matrix are customarily expressed by i and j, as shown in Fig. 2(b). Then the symbolic formula for converting any projected coordinates (X_m, Y_m) to row values I_m and column values J_m in a two-dimensional matrix is as follows:

$$\begin{cases} I_m = (Y_m - Y_0)/DY + 1 \\ J_m = (X_m - X_0)/DX + 1 \end{cases} \tag{3}$$

Where represents rounding down and$DXDY$ represents the real length of each grid. In this paper, the trajectory image is considered as a two-dimensional matrix containing only 0 and 1.

As shown in Fig. 1(c), it can be seen that some adjacent time-stamped points in the two-dimensional matrix correspond to incoherent features. In order to better represent the structure of the trajectory, it is necessary to interpolate two adjacent coordinate points in the two-dimensional matrix. In this paper, we adopt the

Algorithm 1:Digital Differential Analyzer

Input: Grid matrix L
Start point (I_s, J_s)
End point (I_e, J_e)
Output: Grid matrix L
/*Calculate step size of row and column*/
$DI = I_e - I_s DJ = J_e - J_s$
steps=max(|DI|,|DJ|)
for i to steps do
  $I_m = I_s + Isteps * i$
  $J_m = J_s + Jsteps * i$
  add (I_m, J_m) to L and assign a value of 1
End for
return L

Digital Differential Analyzer (DDA), as shown in Fig. 2(b). Let the coordinates of the starting point of the line to be drawn be (I_s, J_s) and the coordinates of the ending point be (I_e, J_e), the pseudo-code corresponding to the steps of Digital Differential Analyzer (DDA) is shown in Algorithm 1.

Fig. 2(c) shows the image of a real GPS trajectory data in the real geographic base map and the trajectory image generated with the algorithm.

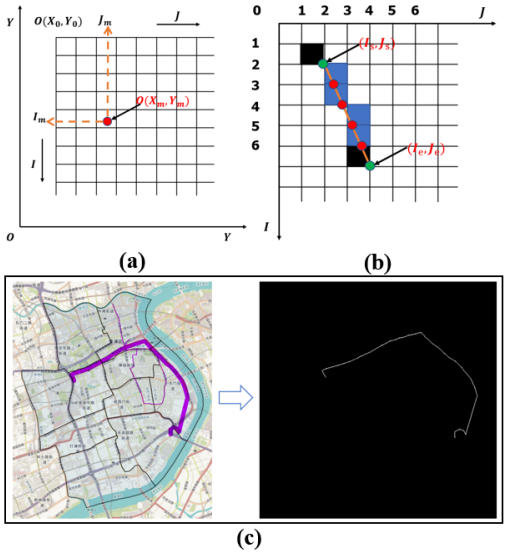

Figure 2: Schematic diagram of trajectory image generation.

3.2 Convolutional Auto-encoder Model

Auto-encoder network is a feed-forward neural network for data compression and feature extraction, which is widely used in unsupervised learning [16]. However, in the classical auto-encoder network structure, all neurons are fully connected with the previous layer, which requires a lot of parameter adjustment during the learning process and may lead to gradient disappearance during

back propagation, thus resulting in the loss of local spatial structure information of the image. The proposed convolutional neural network (CNN) makes image feature extraction more reasonable and greatly reduces the number of parameters [24]. Thanks to the idea of convolutional neural network (CNN), the convolutional auto-encoder network (CAE) adds a convolutional layer for feature extraction before feature encoding and adds a deconvolutional layer for reconstruction operation in feature decoding [30].

The TISM-CAE proposed in this paper combines the advantages of convolutional operations in convolutional neural network and unsupervised training in auto-encoder network, enabling it to process trajectory image data based on convolutional operations during unsupervised learning for feature extraction as well as feature dimensionality reduction. Compared with traditional auto-encoder network, convolutional auto-encoder network can learn the spatial representation of images better, retaining more local details of images and making the model easy to train and reconstruct better. By exploring several successful convolutional auto-encoder networks, the proposed convolutional auto-encoder network is designed as follows. The network structure includes a convolutional encoder and a convolutional decoder, as shown in Fig. 3. The convolutional encoder has four convolutional layers and four pooling layers. Given a gridded trajectory image I, the convolutional encoding operation is defined as follows.

$$m_i = (I * C^n + b) \tag{4}$$

Where m_i represents the activation function, $*$ represents the two-dimensional convolution operation, C^n represents the n^{th} convolution kernel, and b represents the bias term. The size of the convolutional kernels in each convolutional layer is set to 9*9, 7*7, 5*5 and 3*3, the number of corresponding convolutional kernels is set to 16, 16, 8 and 4. A maximum pooling layer with a grid size of 2*2 is added between the different convolutional layers, and the activation function uses the ReLU function commonly used in image processing. After the convolutional coding layer, the data size becomes 4 two-dimensional matrices, which are treated as feature vectors of the trajectory image after spreading.

Next, the trajectory image is reconstructed using the convolution decoder, which uses edge zero padding in order to maintain spatial resolution, and the convolution decoder operation is defined as follows.

$$o_i = (m_i * \tilde{C^n} + \tilde{b}) \tag{5}$$

Where o_i represents the output of the i^{th} trajectory image after reconstruction, represents the n^{th} convolution kernel in the inverse convolution process, and represents the bias term in the inverse convolution process. There are four inverse convolution layers in the convolution decoder defined in this paper. The convolution kernel size of each inverse convolution layer is 3*3, 5*5, 7*7 and 9*9, and the corresponding number of convolution kernels are set to 4, 8, 16 and 16 respectively. The activation function uses the ReLU function.

The loss function in this paper uses the mean square error (MSE) and the loss function formula is defined as follows:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(X_i - Y_i)^2 \tag{6}$$

Where N represents the number of batches, X_i represents the 2D matrix corresponding to the i^{th} original trajectory image in the batch, Y_i represents the generated trajectory image corresponding to the i^{th} original trajectory image in the batch. When the mean square error is smaller, it indicates that the generated trajectory image is closer to the original trajectory image, which means that the low-dimensional feature vector generated by the convolutional encoder can better represent the original trajectory image. Therefore, the similarity between trajectories can be well represented by comparing the Euclidean distance of the feature vectors corresponding to different trajectory images between two.

3.3 Trajectory Similarity Evaluation Criteria

There is no universally accepted standard for trajectory similarity measure, and the results and similarity criteria calculated by different trajectory similarity algorithms are not consistent, so it is impossible to quantitatively assess the accuracy of the results of different trajectory similarity algorithms.

For artificially simulated trajectory datasets, [18] proposes various methods for artificially simulated similar trajectory datasets and the evaluation criteria for the accuracy of the calculation results of corresponding different similarity measure algorithms, which is based on the principle that the list $list_T^{before}$ consisting of k most similar trajectories queried by trajectory T in the original trajectory dataset and the list $list_T^{after}$ consisting of k most similar trajectories queried by trajectory T in the artificially similar trajectory dataset are close to each other. By calculating the Spearman's rank correlation coefficient of the two lists, when the result is closer to 1, it indicates that the corresponding trajectory similarity measure algorithm has higher accuracy.

Since it is difficult to obtain the real trajectory datasets with similar trajectory labels, the trajectory clustering results are generally used for real trajectory datasets to indirectly assess the accuracy of different trajectory similarity measure algorithms, because clustering results are highly dependent on similarity calculations, and high quality clustering results can be obtained if trajectory similarity can be accurately measured. [2] proposes a way to evaluate real trajectories using hierarchical clustering, through two perspectives of inter-class distance and intra-class distance. Firstly, it is necessary to approximately find the trajectory center T^{ex} of the real trajectory dataset T, finding the trajectory that has the smallest sum of distances to other trajectories in a class, and the symbolic definition of T^{ex} is as follows:

$$T_T^{ex} = \min_{T^i, i\in[0,\ldots,n^T]} \left\{ \sum_{j=1, j\neq i}^{n^T} D(T^i, T^j) \right\} \tag{7}$$

Assuming that $C_1, \ldots, C_K$ are clustering centers after clustering the real trajectory dataset T, the intra-class distance (WC) and inter-class distance (BC) are defined symbolically as follows:

$$BC = \sum_{K=1}^{K} D(T_T^{ex}, T_{C_K}^{ex}) \tag{8}$$

$$WC = \sum_{K=1}^{K} \frac{1}{|C_K|} \sum_{T^i \in C_K} D(T_{C_K}^{ex}, T^i) \tag{9}$$

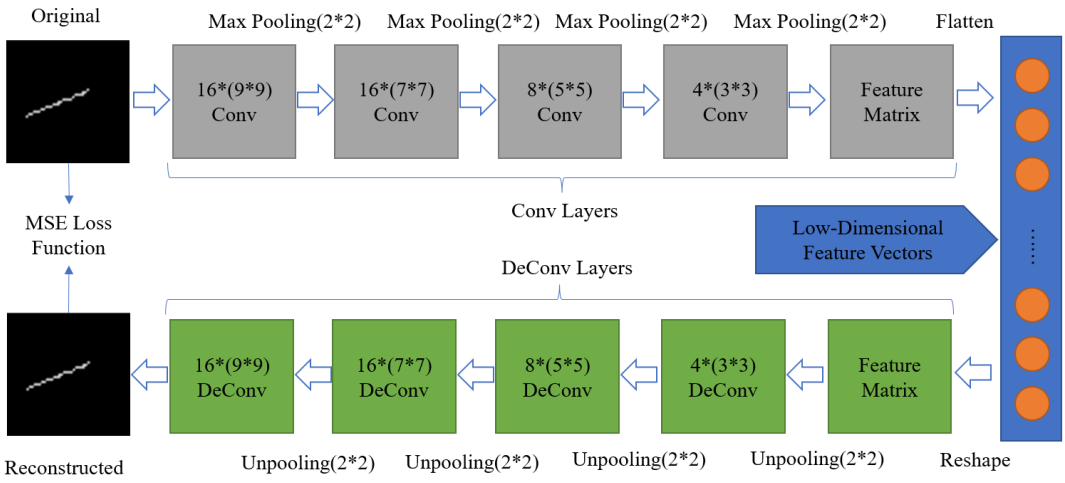

Figure 3: Convolutional Auto-Encoder Model.

Let $AC = WC/(WC + BC)(0, 1)$, in general, the intra-class distance decreases as the number of clusters increases and the inter-class distance increases as the number of clusters increases, so the AC value decreases when the number of clusters increases, the lower AC value represents the higher accuracy of the corresponding trajectory similarity algorithm.

4 EXPERIMENTS

4.1 Data Sets and Operating Environment

The data set used in this paper is real floating vehicle GPS records, which collected 11,870 floating vehicle travel records generated in one day in Huangpu District, Shanghai, with a trajectory sampling frequency of 1s, and each trajectory includes vehicle ID, the longitude and latitude coordinates of trajectory point, sampling time and other information. Since the trajectories of floating vehicles are limited by the road network, they may pass the same road section several times in a day, so each trajectory is segmented to get a trajectory close to a straight line, avoiding the trajectory of turning back and circling. The principle of trajectory segmentation in this paper is that when the angle formed by three consecutive sampling points in a trajectory is less than 145°, the middle sampling point will be regarded as an inflection point, and all the inflection points in a trajectory will be found and the trajectory between two adjacent inflection points will be regarded as a trajectory close to a straight line. Take the logarithm of longitude and latitude points contained in each track as the track length,the lengths of the different straight-line segments obtained by segmenting all the trajectories in Huangpu District in a day are as follows.

It can be seen from Fig. 4 that the length of the trajectory after segmentation is concentrated below 20, which means that there are more short-circuit sections in the road network in Huangpu District, Shanghai, and the longer the length of the trajectory means that the floating vehicle travels on the highway, first-class highway and other road networks, and it is more likely that the trajectory length appears in the same range is the trajectory of different floating vehicles walking on the same road network. In order to balance

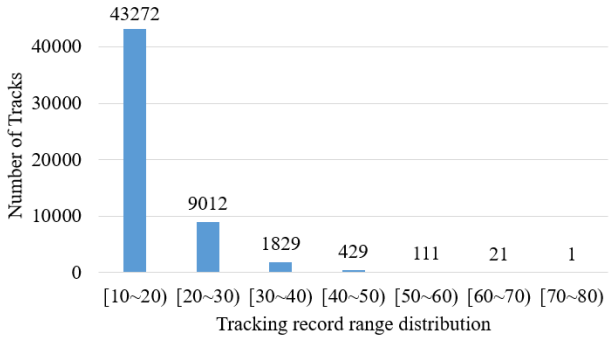

Figure 4: Track length range distribution.

the computational cost as well as to select longer linear trajectory segments, 429 trajectories with lengths between 40 and 50 are selected for analysis in this paper.

The algorithm in this paper is implemented in PyCharm software using python 3.6 programming language. The program runs on Windows 10 operating system, and the computer's central processing unit (CPU) is a 2.3GHz Intel i5-8300H processor with 16GB of running memory.

4.2 Convolutional Auto-encoder Network Parameter Setting

In order to improve the generalization and robustness of the convolutional auto-encoder network model proposed in this paper, various parameters are discussed and analyzed in this section. According to the algorithm proposed in this paper, the target area needs to be gridded. The smaller the grid size, the lower the resolution of the trajectory image, which easily leads to the loss of small-scale geometric features in the original trajectory; The larger the grid size is, the higher the resolution of the trajectory image, which easily leads to the incoherent features corresponding to adjacent timestamp points in the original trajectory and the higher

Table 1: The Imapct Of Cell Size And Batch Size On Loss

Cell Size	32	64	128
Batch Size=200	0.0438	0.0113	0.0299
Batch Size=300	0.0416	0.0076	0.0108
Batch Size=400	0.0608	0.0183	0.0120

information loss rate and longer computation time during the training process. In order to determine the grid size, we make the grid size 32, 64 and 128 with the same learning rate, and the batch size 200, 300 and 400 respectively. And the training results are evaluated according to the mean square error loss function. The loss values at different resolutions and different batch sizes are shown in Table 1.

From Table 1, we can see that the mean square error loss value can be optimized when the batch size is 300 and the grid size is 64, therefore, in this paper, we choose a batch size of 300 and a grid size of 64. On this basis, the learning rates are 0.005, 0.001, 0.0005 and 0.0001, and the training iterations are 4000 times under different learning rates. Fig. 5 shows the effect of different learning rates on the loss values, and the model proposed in this paper can get better results when the learning rate is 0.0005.

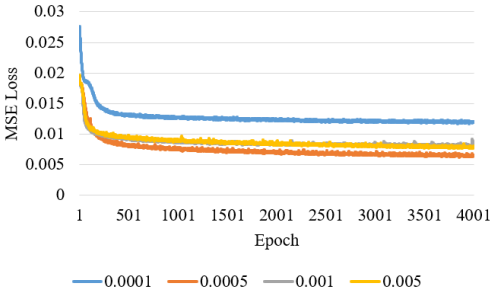

Figure 5: Effect of different learning rates on loss value.

4.3 Experimental Analysis of Artificially Simulated Trajectory Dataset

This section implements two trajectory transformation operations, as downsampling and adding noise points, by drawing on several methods for constructing similar trajectory datasets proposed in [18]. The trajectory transformation refers to the execution of several types of transformations on the original trajectory in a controllable way for any trajectory in the trajectory dataset, the controllable way refers to the adjustment parameters. There are two parameters set in this paper, which are ratio and distance. The ratio is used to specify the percentage of transformed points to the number of original trajectory points. For example, the length of the trajectory to be transformed is 100, Ratio=0.1 means there are 100*0.1=10 trajectory points to perform the corresponding operation. The distance is used to specify how far the trajectory point to perform the transformation operation is from the original position, for example, distance=10m means the trajectory point after the operation is 10m away from its original position. The two trajectory transformation

Table 2: Classification Of Trajectory Transformation Operations

Transformation Type	Adjustment Parameters
downsampling	ratio
adding noise points	ratio,distance

operations set in this paper are shown in Table 2, downsampling means deleting the specified ratio number of trajectory points for any trajectory in the real trajectory data set, when the trajectory Adding noise points means adding a certain number of anomalies to any trajectory in the real trajectory data set, using ratio to control the number of anomalies added to the original trajectory and using distance to control how far the anomalies can be shifted.

4.3.1 Results of Trajectory Similarity Evaluation After Downsampling Process. In order to verify the effect of removing trajectory points on the similarity measure results of TISM-CAE and the comparative algorithms LCSS and EDR, this paper generates similar trajectory datasets by adjusting the ratio parameter, which is used to test the Spearman rank correlation coefficients between $list_T^{after}$ and $list_T^{before}$ generated by different similarity measure algorithms. The ratio parameters are set to 10%, 20% and 30% respectively. To ensure the reliability of the results, 10 trajectories are randomly selected among 429 original trajectories, the averaged Spearman rank correlation coefficients of each trajectory under different similarity measure algorithms are calculated. The correlation coefficient values of different trajectory similarity measure algorithms are shown in Fig. 6.

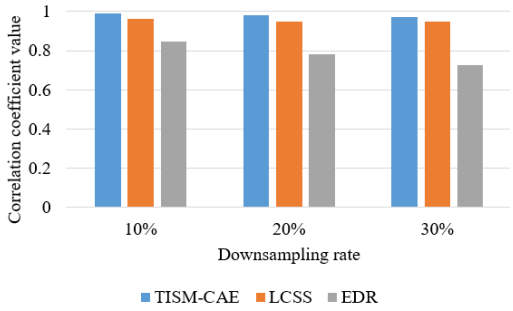

Figure 6: Trajectory similarity evaluation for downsampling processing.

As can be seen from Fig. 6, the results of different similarity measure algorithms have the same trend when dealing with trajectories with missing sampling points, the correlation coefficient values all decrease as the downsampling rate increases. The algorithm TISM-CAE proposed in this paper still has the highest correlation coefficient values at different downsampling rates, which means a higher accuracy, with an average improvement of 3.1% over the best-performing LCSS algorithm.

4.3.2 Results of Trajectory Similarity Evaluation After Adding Noise Points. In order to verify the influence of adding noise points on

the similarity measure results of TISM-CAE and the comparison algorithms LCSS and EDR, this paper only adds some significant anomalous sampling points on the original trajectory data set, and sets the ratio parameter to 10% and 20%, and the distance parameter to 10m and 50m. In order to ensure the reliability of the results, 10 trajectories are randomly selected among 429 original trajectories, and the averaged Spearman rank correlation coefficients between $list_T^{after}$ and $list_T^{before}$ are calculated for each trajectory under different similarity measure algorithms. The correlation coefficient values of different trajectory similarity measure algorithms are shown in Fig. 7.

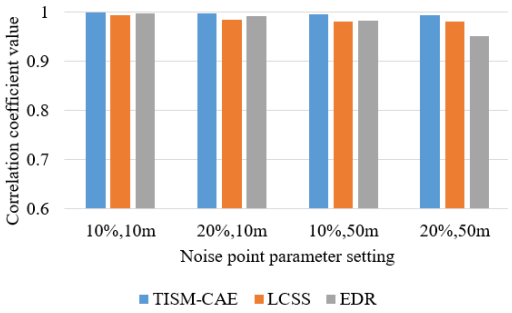

Figure 7: Trajectory similarity evaluation for adding noise point.

As can be seen from Figure 7, the algorithm TISM-CAE proposed in this paper has the highest correlation coefficient value above the trajectories processed with different added noise points, which means a higher accuracy, with an average improvement of 1.1% over the best performing LCSS algorithm.

4.4 Experimental Analysis of Real Trajectory Dataset

In order to verify the accuracy of the similarity measure results of the algorithm TISM-CAE and the comparison algorithms LCSS and EDR on the real trajectory dataset, in view of the absence of a reliable evaluation criterion to quantitatively assess the similarity results of different algorithms, this section uses the trajectory clustering results to indirectly assess the accuracy of different algorithms. Firstly, 100 trajectories are randomly selected in the real trajectory dataset, then the T^{ex} of the approximate central trajectory is calculated according to (7) using different similarity measure algorithms, finally the intra-class distance (WC) and inter-class distance (BC) are calculated according to (8) and (9), and the AC value is obtained. The clusters are set to 1 90 in this paper, the clustering results of different algorithms are shown in Fig. 8.

From Fig. 8, it can be seen that with the increase of the number of clusters, the AC values of all types of algorithms are decreasing and are approaching the value of 0. Among them, the AC values of the algorithm TISM-CAE in this paper are significantly lower than those of LCSS and EDR, indicating that the algorithm in this paper has better clustering effect.

In order to compare the time complexity between different algorithms, this paper tends to measure the execution time of similarity

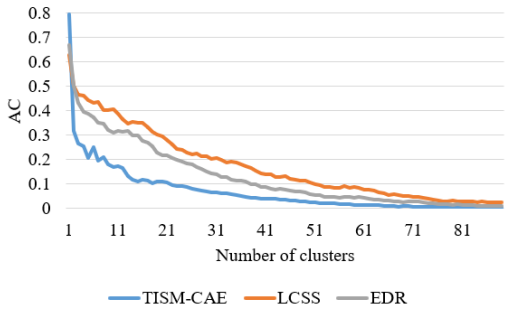

Figure 8: Clustering results of different similarity measure algorithms.

computation for different number of trajectories. In particular, in order to obtain more fair experimental results, 100, 200, 300 and 400 trajectories are randomly selected in the original real 429 trajectory dataset in this paper, and each similarity algorithm computation experiment is run 50 times for different number of datasets to obtain the average results. The computation time and trend of TISM-CAE and the comparison algorithms LCSS and EDR are visualized in Fig. 9. In comparison, the algorithm proposed in this paper can greatly improve the computational efficiency.

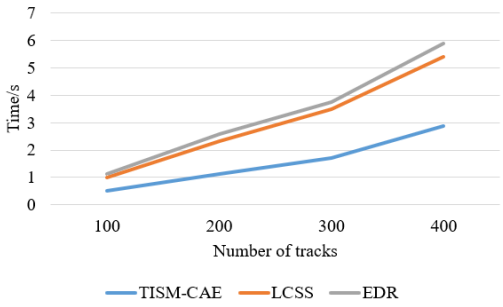

Figure 9: Running time of different similarity measure algorithms.

5 DISCUSSION

In this paper, we employ real floating vehicle trajectory data in Huangpu District, Shanghai, segment all trajectories, and select trajectories with medium length for experiments. The proposed TISM-CAE shows better results both in accuracy and time complexity. However, some limitations in the setting of experiments should not be neglected.

Firstly, limited by the floating car data resource, the research area mainly covers Huangpu District in Shanghai city. Since TISM-CAE provides an open framework for trajectory similarity estimation, further experiments can be carried out with the help of suitable data resource at urban or regional level.

Next, the training process of the convolutional auto-encoder model is limited by the floating car data with different trajectory lengths. In order to reflect the general situation, in the preprocessing step, trajectories whose number of GPS track points are

less than 40 are filtered, and trajectories with long coverage are firstly segmented within a medium number of GPS track points (i.e., the number of GPS track points between 40 and 50) according to literature review[15] [29] [17]. The medium-length trajectories can better grasp the internal rules of spatial-temporal mobility, showing a better experimental effect.

What's more, the sampling frequency of floating car data also affects the processing time of trajectory similarity estimation. The time complexity of traditional algorithms LCSS and EDR is $O(n^2)$, where n is the GPS record number in a given trajectory. The time complexity of the proposed TISM-CAE is not affected by the sampling frequency since raster data can be processed in a linear manner. This characteristic makes TISM-CAE more suitable for parallelization, showing a valuable avenue for epidemic prevention and control.

6 CONCLUSION

In this paper, we propose a fast computational method TISM-CAE for trajectory image structure matching combined with convolutional auto-encoder. The accuracy of TISM-CAE outperforms LCSS and EDR on similar trajectory datasets with uneven sampling and noise in artificial simulations. The accuracy of different trajectory similarity measure algorithms is indirectly evaluated using trajectory clustering results on real trajectory datasets, where TISM-CAE clustering is the best, while the time complexity between the algorithms is compared using different number of trajectories, TISM-CAE has the best computational efficiency. This model is accurate and efficient, with good generalization, and provides a new solution for trajectory similarity measure method.

REFERENCES

[1] Rakesh Agrawal, Christos Faloutsos, and Arun Swami. 1993. Efficient similarity search in sequence databases. In *International conference on foundations of data organization and algorithms*. Springer, 69–84.

[2] Philippe C Besse, Brendan Guillouet, Jean-Michel Loubes, and François Royer. 2016. Review and perspective for distance-based clustering of vehicle trajectories. *IEEE Transactions on Intelligent Transportation Systems* 17, 11 (2016), 3306–3317.

[3] Lei Chen, M Tamer Özsu, and Vincent Oria. 2005. Robust and fast similarity search for moving object trajectories. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*. 491–502.

[4] James W Hunt and Thomas G Szymanski. 1977. A fast algorithm for computing longest common subsequences. *Commun. ACM* 20, 5 (1977), 350–353.

[5] Eamonn J Keogh and Michael J Pazzani. 2000. Scaling up dynamic time warping for datamining applications. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*. 285–289.

[6] Eamonn J Keogh and Michael J Pazzani. 2000. Scaling up dynamic time warping for datamining applications. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*. 285–289.

[7] Changchun Li, Bogen Cai, Guanwei Shang, and Jian Wang. 2012. Research and implementation of map algorithm based on Web Mercator projection. *Computer Application Research* 29, 12 (2012), 4793–4796.

[8] Hailin Li, Chonghui Guo, and Wangren Qiu. 2011. Similarity measure based on piecewise linear approximation and derivative dynamic time warping for time series mining. *Expert Systems with Applications* 38, 12 (2011), 14732–14743.

[9] Huanhuan Li, Jingxian Liu, Zaili Yang, Ryan Wen Liu, Kefeng Wu, and Yuan Wan. 2020. Adaptively constrained dynamic time warping for time series classification and clustering. *Information Sciences* 534 (2020), 97–116.

[10] Xi Li. 2019. Trajectory accompanying pattern mining based on spatial temporal segmentation and word vector similarity. *Journal of Zhongshan University: Natural Science Edition* 58, 5 (2019), 17–25.

[11] Xiucheng Li, Kaiqi Zhao, Gao Cong, Christian S Jensen, and Wei Wei. 2018. Deep representation learning for trajectory similarity computation. In *2018 IEEE 34th international conference on data engineering (ICDE)*. IEEE, 617–628.

[12] Ying Li, Li Zhao, Xiangmo Zhao, and Ke Chen. 2020. Effectiveness of trajectory similarity measures based on truck GPS data. *China Journal of Highway and Transport* 33, 2 (2020), 146–157.

[13] Rayanoothala Praneetha Sree, Durvasula VLN Somayajulu, and S Ravichandra. 2020. A Novel Approach for Mining Time and Space Proximity-based Frequent Sequential Patterns from Trajectory Data. *Journal of Information & Knowledge Management* 19, 04 (2020), 2050040.

[14] Sayan Ranu, Padmanabhan Deepak, Aditya D Telang, Prasad Deshpande, and Sriram Raghavan. 2015. Indexing and matching trajectories under inconsistent sampling rates. In *2015 IEEE 31st International Conference on Data Engineering*. IEEE, 999–1010.

[15] Arun Sharma, Zhe Jiang, and Shashi Shekhar. 2022. Spatiotemporal Data Mining: A Survey. *arXiv preprint arXiv:2206.12753* (2022).

[16] B She, F Tian, and W Liang. 2018. Fault diagnosis based on a deep convolution variational autoencoder network. *Yi Qi Yi Biao Xue Bao/Chinese Journal of Scientific Instrument* 39, 10 (2018), 27–35.

[17] Shashi Shekhar, Zhe Jiang, Reem Y Ali, Emre Eftelioglu, Xun Tang, Venkata MV Gunturi, and Xun Zhou. 2015. Spatiotemporal data mining: A computational perspective. *ISPRS International Journal of Geo-Information* 4, 4 (2015), 2306–2338.

[18] Han Su, Shuncheng Liu, Bolong Zheng, Xiaofang Zhou, and Kai Zheng. 2020. A survey of trajectory distance measures and performance evaluation. *The VLDB Journal* 29, 1 (2020), 3–32.

[19] Lun Wu, Liu Yang, Zhou Huang, Yaoli Wang, Yanwei Chai, Xia Peng, and Yu Liu. 2019. Inferring demographics from human trajectories and geographical context. *Computers, Environment and Urban Systems* 77 (2019), 101368.

[20] Jiangang Yu, Yihao Guo, Xinning Zhu, Yifan You, and Dinghe Xiao. 2020. Discovery of Travelling Companions from Trajectories with Different Sampling Rates. In *Proceedings of the 4th International Conference on Computer Science and Application Engineering*. 1–8.

[21] Qingying Yu, Yonglong Luo, Chuanming Chen, and Shigang Chen. 2019. Trajectory similarity clustering based on multi-feature distance measurement. *Applied Intelligence* 49, 6 (2019), 2315–2338.

[22] Guan Yuan, Penghui Sun, Jie Zhao, Daxing Li, and Canwei Wang. 2017. A review of moving object trajectory clustering algorithms. *Artificial Intelligence Review* 47, 1 (2017), 123–144.

[23] Haitao Yuan and Guoliang Li. 2021. A survey of traffic prediction: from spatio-temporal data to intelligent transportation. *Data Science and Engineering* 6, 1 (2021), 63–85.

[24] Fu Zhang, Nian Cai, Jixiu Wu, Guandong Cen, Han Wang, and Xindu Chen. 2018. Image denoising method based on a deep convolution neural network. *IET Image Processing* 12, 4 (2018), 485–493.

[25] Yifan Zhang, An Liu, Guanfeng Liu, Zhixu Li, and Qing Li. 2019. Deep representation learning of activity trajectory similarity computation. In *2019 IEEE International Conference on Web Services (ICWS)*. IEEE, 312–319.

[26] Zhujun Zhao and Genlin Ji. 2017. Advances in spatial temporal trajectory classification research. *Journal of Geoinformation Science* 19, 3 (2017), 289–297.

[27] Kai Zheng, Yu Zheng, Xing Xie, and Xiaofang Zhou. 2012. Reducing uncertainty of low-sampling-rate trajectories. In *2012 IEEE 28th international conference on data engineering*. IEEE, 1144–1155.

[28] Yu Zheng. 2015. Overview of urban computing. *Journal of Wuhan University (Information Science Edition)* 1 (2015), 1–13.

[29] Yu Zheng. 2015. Trajectory data mining: an overview. *ACM Transactions on Intelligent Systems and Technology (TIST)* 6, 3 (2015), 1–41.

[30] Qian Zhou, Yimin Qiu, and Zhenyu Wu. 2020. Image retrieval research based on convolutional auto-encoder and hash algorithm. *Instrumentation Technology and Sensors* 11 (2020), 105–110.

[31] Xingxing Zhou, Genlin Ji, and Shuliang Zhang. 2018. A review of spatial temporal trajectory similarity metrics. *Geoinformation World* 25, 4 (2018), 11–18.