Efficient Trajectory Retrieval

By Hasan Pourmahmoodaghababa (Group 10)
uID: u1255635

1 The Problem

The problem under consideration in this project is trajectory similarity problem from information retrieval perspective. We include a formal definition of a trajectory below:

Definition 1.1. *A spatial trajectory is a sequence of way-points $T = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, where (x_i, y_i) shows the latitude and longitude of a moving object. (x_i, y_i) are called waypoints.*

A trajectory can show the spatial location of a moving object like vehicles [2], humans [6] and animals [1] that have arisen through collection of GPS traces carried by users. Time series [3] and other shapes such like letters [5] are other sorts of examples.

The goal of this project is that given a query trajectory we should return the most similar trajectories from data to the query. This is somehow similar to a term-document task. Two types of queries are considered here:

- The first type is a kNN query, i.e. a pair $q = (T, k)$, where T is a trajectory and k is a positive integer. The search machine should return k most similar trajectories to T from data in an ordered way.

- The second kind is a range query, i.e. a pair $q = (T, r)$, where T is a trajectory and r is a positive real number showing the range we are willing to get an ordered list of trajectories that lie in distance at most r from T in data.

2 Motivation

The trajectory similarity problem has many applications in different fields such as transportation, biology and stock market. So, studying trajectory similarity problem one can extract users' behavior, detect transportation modes, do route extraction task, and so on. Generically, however, this is a difficult problem because of the complexity of the shape of trajectories. Indeed, famous distances defined on trajectories, such as Fréchet, discrete Frćhet and dynamic time warping, are usually expensive to utilize as they suffer from quadratic complexity in terms of the number of waypoints of trajectories. Thus, when dealing with long trajectories even in a medium-sized data sets employing these measures will not be useful and efficient in practice.

Moreover, to get the similar trajectories from data to a query trajectory needs computing the distance between the query and all trajectories in data. Therefore, utilizing those metrics will not be appropriate.

Therefore, the question is if there is an efficient and effective way to get similar trajectories from data?

3 Approach

3.1 Related Work and Novelty

We will essentially propose the method studied in the research paper [4] with some modifications and new ideas, which is an information retrieval perspective; in fact, building a vector space model. We will apply novel vectorization techniques and evaluate our approach on 3 real world datasets not only on Precision and Recall (like [4]) but also based on MAP and nDCG, which will be designed from their corresponding definitions in IR on text data.

3.2 Definitions

Considering the fact that all trajectories live in a rectangular region R, we divide R into a number of standard grids (tiny rectangles). Say we have d grids. Then according to the fact that if a trajectory passes through a grid, we can get a vectorized representation of a trajectory. We have considered two types of vectorizations:

Definition 3.1. *A binary vectorization of a trajectory T living in the region R is a d-dimensional vector $\tilde{T} = (a_1, \ldots, a_d)$, where $a_i = 1$ if T touches the i-th grid and $a_i = 0$ otherwise.*

Definition 3.2. *A frequency-preserving vectorization of a trajectory T living in the region R is a d-dimensional vector $\tilde{T} = (a_1, \ldots, a_d)$, where a_i is the number of times that T touches the i-th grid.*

So, by applying any type of vectorization on a dataset $D = \{T_i\}_{i=1}^n$ of size n, we can get a data matrix $\tilde{D}$ of size $d \times n$ by putting each $\tilde{T}_i$ in the i-th column of $\tilde{D}$.

In order to construct a suitable distance between trajectories, we will use a *similarity matrix*.

Definition 3.3. *A cell similarity matrix, denoted by C, is an $n \times n$ symmetric matrix, where $C[i, j]$ shows the similarity of i-th and j-th cell according to a similarity function in $\mathbb{R}^2$. Indeed, $0 \leq C[i, j] \leq 1$ and $C[i, i] = 1$.*

We can now define the *grid aggregate distance*

Definition 3.4. *The Grid Aggregate Distance between two trajectories T_i and T_j corresponding to a cell similarity matrix C is the Mahalanobis distance:*

$$\mathrm{GAD}(T_i, T_j) = \sqrt{(\tilde{T}_i - \tilde{T}_j)^t C (\tilde{T}_i - \tilde{T}_j)}.$$

Since C is symmetric, using linear algebra, we can decompose it as $C = P \Delta P^t$, where P is an orthogonal matrix (i.e. $T^{-1} = P^t$) and Δ is a diagonal $d \times d$ matrix, where $\Delta[i, i]$ is an eigenvalue of C. Let Δ_0 be the diagonal matrix such that $\Delta_0[i, i] = \Delta[i, i]$ if $\Delta[i, i] \geq 0$ and $\Delta_0[i, i] = 0$ otherwise. We will consider the approximated version of C ($P \Delta_0 P^t$) as C. So,

$$\mathrm{GAD}(T_i, T_j) = \sqrt{[\Delta_0^{1/2} P^t (\tilde{T}_i - \tilde{T}_j)]^t [\Delta_0^{1/2} P^t (\tilde{T}_i - \tilde{T}_j)]}.$$

3.3 Latent Semantic Indexing (LSI)

Let $\tilde{\tilde{D}} = \Delta_0^{1/2} P^t \tilde{D}$. Since we will choose the number of grids to be high in order to get a better resolution of vectorization, we would like to reduce the dimension of vectorized trajectories. To do so, let $\tilde{\tilde{D}} = USV^t$ be its singular value decomposition (SVD). Let say we would like reduce the dimension of embedded trajectories to e. It can be done by considering only e columns of U, say U_e. Therefore, the reduced dimensional embedded data would be $\tilde{\tilde{D}}_e = U_e^t \cdot \tilde{\tilde{D}}$, which is considered as the index of our document/data.

Now the grid aggregate distance between two trajectories T_1 and T_2 can be efficiently approximated by the Euclidean distance between $U_e^t \cdot \tilde{\tilde{T}}_1$ and $U_e^t \cdot \tilde{\tilde{T}}_2$, i.e.

$$\mathrm{GAD}(T_1, T_2) \approx \|U_e^t \cdot \tilde{\tilde{T}}_1 - U_e^t \cdot \tilde{\tilde{T}}_2\|.$$

3.4 Query Processing

First of all, we store the matrix U_e. Then given a new query trajectory T_q, we first vectorize it to get its d-dimensional vectorized grid representation $\tilde{T}_q$. Then we calculate $\tilde{\tilde{T}}_q = \Delta_0^{1/2} \cdot P^t \cdot \tilde{T}_q$. Lastly, using U_e, we can get the compact representation $\tilde{\tilde{T}}_{qe} = U_e^t \cdot \tilde{\tilde{T}}_q$. Now we can utilize the index $\tilde{\tilde{D}}_e$ (by just evaluating the Euclidean distance of $\tilde{\tilde{T}}_{qe}$ and each column of $\tilde{\tilde{D}}_e$) to return the appropriate trajectories depending on the type of query.

4 Experiments

Three trajectory data sets are used for evaluation.

Preprocessing. In all datasets, we first removed stationary points from all trajectories. Then we removed all duplicated trajectories as well as trajectories with less than 10 waypoints. Moreover, in Geolife dataset we removed all unusual trajectories like outliers or airplane trajectories.

Experimental Setup. In [4], the cosine and dot product similarity functions are used to get the cell similarity matrix. But we applied the Gaussian kernel as a similarity function. Indeed, if c_i is the center of i-th cell, the similarity of i-th and j-th cells is given by

$$k(c_i, c_j) = \exp(-\|c_i - c_j\|^2/\sigma^2),$$

where σ is chosen to be 3 times the length of a grid. The reason we chose the Gaussian kernel is that it is positive definite and so the obtained matrix C is positive definite. Therefore, there is no need for approximation by Δ_0. This guarantees more accuracy in results.

Evaluation For all datasets we have used Gaussian kernel as a cell similarity function. We have done the experiments with Car-Bus and Characters data sets both with GAD and DTW as ground truth distances.

Data Sets Overview. Table 1 shows the statistics of data sets used in experiments.

Data	Size	Cleaned	Sample	Grids	Reduced dim
Car-Bus	163	105	105	2500	25
Characters	2858	2858	300	2500	25
Geolife	17621	14187	1000	2500	100

Table 1: Overview of data sets

4.1 Car-Bus Dataset

Car-Bus dataset from UCI Machine Learning Repository[1]. There are 87 car and 76 bus trajectories in this data, which are recorded in Aracuja, Brazil (see Figure 1 (left)). The statistics with GAD (DTW) as ground truth distance are given in Tables 2 and 3 (4 and 5).

k	Precision	Recall	MAP	nDCG@5	nDCG@10
5	0.8781	0.8781	0.8929	0.9400	0.9646
10	0.9143	0.9143	0.8996	0.7902	0.9588
20	0.9576	0.9576	0.9240	0.5907	0.8210

r	Precision	Recall	MAP	nDCG@5	nDCG@10
0.002	0.8634	0.9657	0.8061	0.7491	0.6313
0.005	0.8775	0.9162	0.7228	0.8955	0.8895
0.01	0.9171	0.9171	0.7274	0.9400	0.9588

Table 2: Performance of Car-Bus dataset with binary representation

k	Precision	Recall	MAP	nDCG@5	nDCG@10
5	0.9314	0.9314	0.9579	0.9809	0.9942
10	0.9343	0.9343	0.9496	0.8210	0.9856
20	0.9700	0.9700	0.9530	0.6024	0.8298

r	Precision	Recall	MAP	nDCG@5	nDCG@10
0.002	0.9914	1	0.9914	0.5175	0.3599
0.005	0.9603	1	0.9603	0.5393	0.3792
0.01	0.9056	0.9990	0.8914	0.6260	0.4737

Table 3: Performance of Car-Bus data with frequency representation

k	Precision	Recall	MAP	nDCG@5	nDCG@10
5	0.5600	0.5600	0.6580	0.7391	0.8088
10	0.5381	0.5381	0.6022	0.5991	0.7170
20	0.5090	0.5090	0.5569	0.4513	0.5862

r	Precision	Recall	MAP	nDCG@5	nDCG@10
0.002	0.5752	0.9952	0.4961	0.5957	0.4550
0.005	0.3647	0.9514	0.1624	0.6543	0.5305
0.01	0.3705	0.8763	0.0634	0.6913	0.5860

Table 4: Performance of Car-Bus dataset with DTW as the ground truth distance, and binary representation

Comparing Tables 2 and 3 one can easily observe that the frequency-preserving vectorization performs better than

[1] `https://archive.ics.uci.edu/ml/datasets/GPS+Trajectories`

k	Precision	Recall	MAP	nDCG@5	nDCG@10
5	0.5429	0.5429	0.6646	0.7309	0.7964
10	0.5333	0.5333	0.5977	0.5975	0.7053
20	0.5281	0.5281	0.5623	0.4607	0.5881
r	**Precision**	**Recall**	**MAP**	**nDCG@5**	**nDCG@10**
0.002	1	0.7643	0.9817	0.5932	0.4363
0.005	0.9857	0.6374	0.9524	0.6416	0.5072
0.01	0.8687	0.5452	0.7736	0.6782	0.5632

Table 5: Performance of Car-Bus dataset with DTW as the ground truth distance, and frequency representation

the binary representation, although both have a high performance. Considering the performance of the proposed method with DTW as the ground truth distance (Tables 4 and 5), one can conclude that the retrieval performance decreases in comparison with GAD.

4.2 Characters Trajectory Dataset

This dataset is also taken from UCI Machine Learning Repository[2]. It consists of handwritten characters captured using a WACOM tablet (see Figure 2). The average performance over 300 (about 10% of data) trajectories with GAD (DTW) as ground truth distance are given in Tables 6 and 7 (8 and 9).

k	Precision	Recall	MAP	nDCG@5	nDCG@10
5	0.9107	0.9107	0.9224	0.9667	0.9878
10	0.9400	0.9400	0.9284	0.8176	0.9838
20	0.9538	0.9538	0.9396	0.6025	0.8334
r	**Precision**	**Recall**	**MAP**	**nDCG@5**	**nDCG@10**
0.002	0.7720	1	0.7656	0.5624	0.4010
0.005	0.8372	0.9680	0.5450	0.9351	0.9067
0.01	0.9400	0.9400	0.6567	0.9667	0.9838

Table 6: Performance of Characters dataset with GAD as the ground truth distance, and binary representation

k	Precision	Recall	MAP	nDCG@5	nDCG@10
5	0.8373	0.8373	0.8816	0.9327	0.9694
10	0.8303	0.8303	0.8552	0.7833	0.9356
20	0.8458	0.8458	0.8497	0.5834	0.7878
r	**Precision**	**Recall**	**MAP**	**nDCG@5**	**nDCG@10**
0.002	1	1	1	0.4868	0.3337
0.005	0.9733	1	0.9733	0.4868	0.3337
0.01	0.8174	1	0.8174	0.5067	0.3494

Table 7: Performance of Characters dataset with GAD as the ground truth distance, and frequency representation

k	Precision	Recall	MAP	nDCG@5	nDCG@10
5	0.6060	0.6060	0.6934	0.7711	0.8488
10	0.6287	0.6287	0.6557	0.6504	0.7792
20	0.6158	0.6158	0.6440	0.5084	0.6639
r	**Precision**	**Recall**	**MAP**	**nDCG@5**	**nDCG@10**
0.002	0.9534	0.2180	0.7903	0.7696	0.7746
0.005	0.6541	0.6005	0.3171	0.7711	0.7791
0.01	0.6287	0.6287	0.2189	0.7711	0.7791

Table 8: Performance of Characters dataset with DTW as the ground truth distance, and binary representation

k	Precision	Recall	MAP	nDCG@5	nDCG@10
5	0.7147	0.7147	0.7950	0.8578	0.9120
10	0.6930	0.6930	0.7453	0.7203	0.8451
20	0.6363	0.6363	0.7040	0.5451	0.7049
r	**Precision**	**Recall**	**MAP**	**nDCG@5**	**nDCG@10**
0.002	1	0.9933	1	0.4900	0.3362
0.005	0.9933	0.7089	0.9933	0.6293	0.4721
0.01	0.9909	0.3102	0.9272	0.8341	0.7674

Table 9: Performance of Characters dataset with DTW as the ground truth distance, and frequency representation

Unlike Car-Bus data, for kNN queries as shown in Tables 6 and 7, in Characters data the binary representation tends to do a bit better job than the frequency-preserving representation. Also according to Tables 8 and 9 we observe that the performance of the proposed method with DTW as the ground truth distance for retrieval task is lower than using GAD as the ground truth distance.

4.3 Geolife Trajectory Dataset

This big dataset, Geolife trajectory dataset[3], which was released by Microsoft in 2012, consists of trajectories of 182 users from 2007 to 2012 which are mostly recorded in Beijing, China (see Figure 1 (right)). The average statistics over 1000 randomly sampled trajectory from data as query using GAD as ground truth distance are given in Tables 10, 11.

k	Precision	Recall	MAP	nDCG@5	nDCG@10
5	0.8676	0.8676	0.8632	0.9182	0.9443
10	0.8921	0.8921	0.8751	0.7822	0.9394
20	0.9074	0.9074	0.8904	0.5823	0.7989
r	**Precision**	**Recall**	**MAP**	**nDCG@5**	**nDCG@10**
0.001	0.9275	0.9549	0.8116	0.7260	0.6559
0.005	0.8896	0.9023	0.2025	0.8893	0.8952
0.01	0.8883	0.8892	0.0376	0.9182	0.9390

Table 10: Performance of Geolife dataset with GAD as the ground truth distance, and binary representation

As it can be viewed from Tables 10 and 11, generically, both featurization techniques perform well on a big and

[2] https://archive.ics.uci.edu/ml/datasets/Character+Trajectories

[3] https://msropendata.com/datasets/d19b353b-7483-4db7-a828-b130f6d1f035

k	Precision	Recall	MAP	nDCG@5	nDCG@10
5	0.9480	0.9480	0.9597	0.9822	0.9930
10	0.9590	0.9590	0.9577	0.8233	0.9886
20	0.9653	0.9653	0.9604	0.6029	0.8330

r	Precision	Recall	MAP	nDCG@5	nDCG@10
0.001	0.9990	1	0.9990	0.4906	0.3367
0.005	0.9894	1	0.9973	0.5291	0.3748
0.01	0.9765	0.9978	0.9582	0.5735	0.4276

Table 11: Performance of Geolife dataset with GAD as the ground truth distance, and frequency representation

quit challenging dataset. However, the binary featurization obtains a low performance for MAP with range queries but nDCG values are reasonable and recall is high.

Performance Analysis. According to the tables one can easily see that by increasing k in kNN queries, in all data sets, either with binary or frequency representation, average AP and MAP generally increase but nDCG decreases when we use GAD as the ground truth distance. However, average AP, Recall and MAP tend to decrease but nDCG increases for range queries by increasing the range size.

Remark. For kNN queries Precision and Recall are the same in this context since we can always retrieve k trajectories from data and there are k relevant ones in data.

5 Conclusion

According to the experiments, we can conclude that the LSI technique for trajectory retrieval, with either binary or frequency-type vectorization, is a quite successful method. Indeed, for all data sets using the GAD distance as the ground truth measure, mostly we got values above 90% for MAP, Recall and nDCG. As we can see from tables, on Car-Bus and Geolife data, the frequency-type representation tended to perform better while on Characters binary representation did a better job.

Considering the effect of ground truth distance between trajectories, as it is evident from tables, the LSI technique generally performs well with GAD distance in comparison to the popular distance Dynamic Time Warpping (DTW). The reason is clear, I guess. That is because the LSI technique comes by approximating the embedded trajectories in a latent space. So, obviously, the euclidean distance between two reduced dimensional embedded trajectories is an approximate of their GAD distance. Therefore, it is expected to get a high performance considering the GAD distance rather than others like DTW.

6 Contribution

My contributions are:

- Applying a kernel-based similarity measure,
- Using different 3 datasets as a benchmark including a big dataset,
- Utlizing binary vectorization of trajectories,
- Using term-frequency type trajectory vectorization,
- Applying DTW as a ground truth distance,
- Creating a user-friendly interactive search tool for kNN and range queries of trajectories.

7 Implementation

Implementation of the computational part of the project is done in Python. However, the user interface is done in JavaScript and HTML. These backend and frontend are connected together with an API in Python with Flask. Codes and datasets are in the following GitHub repository: `https://github.com/aghababa/Information_Retrieval/tree/main/Project/User%20Interface`

8 Visualizing Data Sets.

Figures 1 and 2 show an overview of datasets used here.

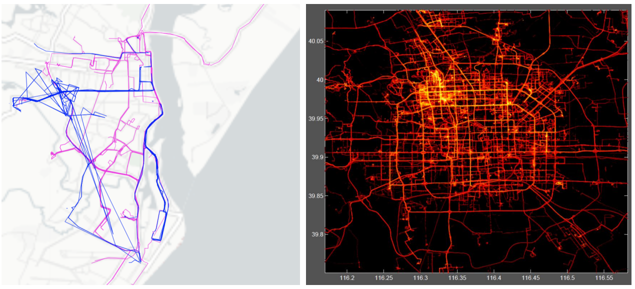

Figure 1: Car-Bus (left) and Geolife (right) data sets.

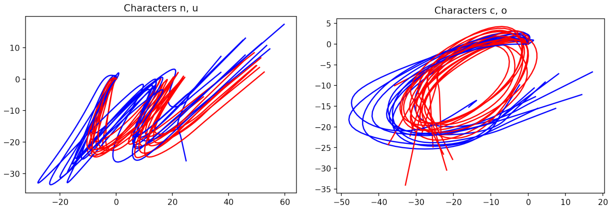

Figure 2: Characters data. Letters n, u (left), c, o (right).

References

[1] Kevin Buchin, Anne Driemel, Natasja van de L'Isle, and André Nusser. klcluster: Center-based clustering of trajectories. In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 496–499, 2019.

[2] Michael O. Cruz, Hendrik Macedo, R. Barreto, and Adolfo Guimaraes. *GPS Trajectories Data Set*, February 2016.

[3] Anne Driemel, Amer Krivosija, and Christian Sohler. Clustering time series under the Frechet distance. In *ACM-SIAM Symposium on Discrete Algorithms*, 2016.

[4] Apostolos N. Papadopoulos. Trajectory retrieval with latent semantic analysis. In *SAC'08 March 16-20*, pages 1089–1094.

[5] Ben H Williams, Marc Toussaint, and Amos J Storkey. A primitive based generative model to infer timing information in unpartitioned handwriting data. In *IJCAI*, pages 1119–1124, 2007.

[6] Yu Zheng, Hao Fu, Xing Xie, Wei-Ying Ma, and Quannan Li. *Geolife GPS trajectory dataset - User Guide*, July 2011.