(심사용 작성양식) // 온라인 접수 시, 본 라인은 삭제한 후 제출해 주세요

분산 인-메모리 시스템에서 유사한 궤적 검색을위한 학습 된 파티션 인덱스

라마단 하니, 누르산티오 후드자이파 에디, 권준호
부산대학교
hani042@pusan.ac.kr, hudzaifah@pusan.ac.kr, jhkwon@pusan.ac.kr

Learned Index for Similar Trajectory Search in Distributed In-Memory System

요 약

The application of the machine learning-based indexing, called learned index, currently gains popularity because its efficiency and size outperform the traditional B-tree index. However, it is still not yet applied in the trajectory domain. In this study, we apply the learned index to a well-known distributed system for searching similar trajectories. We denote some problem transformation of the indexing procedures in a similar trajectory search to a machine learning approach. We plan to demonstrate the feasibility of the learned index on DITA system for searching similar trajectory large trajectory data of moving vehicles.

1. Introduction

Thanks to the massive use of the mobile device and location-based service, especially ride-hailing apps, it is getting easier and more accurate to track moving objects which result in trajectory data. The abundant trajectory data may reveal interesting patterns when analyzed. However, to support this use case, it is crucial to consider the similarity between trajectories.

However, it is time-consuming and challenging to compute the similarity between an input of a single trajectory to a large set of trajectories, especially in a distributed system. Some systems handled similarity trajectory search in distributed based on segmentation [1] and pivot points [2]. Those approaches utilize different distance computation schemes to query similar trajectories from an input trajectory. Although they are effective and efficient to find similar trajectories, it is interesting to see how the learned index [3] can improve the similarity search. The learned index is a machine learning-based indexing that performs faster and less space-consuming than traditional indexing. Also, the application of machine learning in the database has been extended to different features in SageDB [4], including a multi-dimensional index. The multi-dimensional index seems appropriate for trajectory data, but it is only limited to range queries, not the similarity. On the other hand, other learned-based approaches [5] try to perform the nearest neighbor search using a probabilistic approach for distance computation using codebook and several data compression mechanisms. Thus, it is interesting to incorporate the probabilistic distance modeling onto a similar trajectory search in a distributed system.

In this project, we demonstrate a learned index approach to model the distance of trajectories in a distributed system, benefiting the concept in [6] and DITA [2]. In this initial study, we restrict the case of a similar trajectory search with several limitations.

2. Problem Definition

In this section, we will provide standard notations and then formalize the problem for this study.

A trajectory T is an l-length sequence of positions $T = \{p_1, p_2, \dots p_l\}$ where p_i is the i-th position in the sequence. In this case, we have a trajectory set TS in 2D space. Thus, the scope of positions p is limited to the Cartesian coordinate $p_i = (x_i, y_i)$

Then, we need to define a measure for trajectory similarity. We denote two trajectories are similar where their distance is less than or equal to a certain threshold. However, there are several distance measures for two trajectories. In this study, we use the Dynamic Time Warping (DTW) distance, as described in [6].

However, it is exhaustive to compute the DTW distance between a query trajectory and all of the trajectories in the dataset, especially when the dataset is large. Thus, it is important to build effective indexing to search a similar trajectory to a query trajectory. On the other hand, given a machine learning approach, we can ”learn” the search scheme and the DTW distance approach. Based on the problem, we would like to define the problem transformation for the effective indexing to a machine learning approach and then perform a similarity search using the learned approach.

3. Proposed System

We propose a system to substitute a similar trajectory search indexing in DITA by a learned index. The difference between the proposed system and DITA is depicted in Figure 1. We describe the similar trajectory search and the learned index approach for that case as follows.

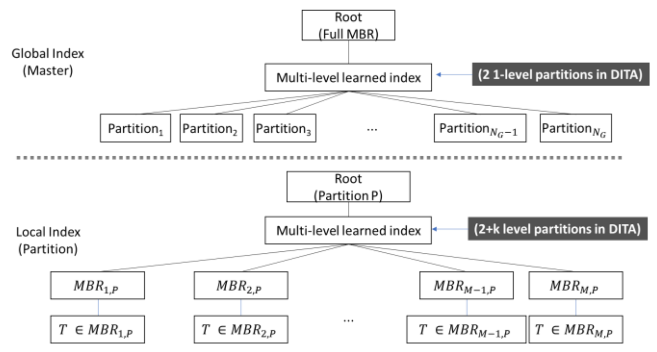

Figure 1 Difference between our approach and DITA

The main difference between our approach and DITA indexing is that we apply the multi-level learned index instead of R-tree. However, there are two different approaches that we can perform to do indexing. The first one is the probabilistic distance approach, which we mainly propose, while the second one is the multi-label approach. We only study this approach on the probabilistic distance approach. However, before we discuss the approaches in-depth, we present another essential part of this study: building the dataset.

3.1. Building Similar Trajectory Training Dataset

First, we build the training set by using DITA to output the similar trajectories set $SimTS$ in the reference dataset R for a query trajectory T_Q with a threshold τ. We concatenate the similar trajectories set with their respective global index MBR M_G and local index $M_{G,l}$. We formalize the returned similar trajectories to T_Q with a threshold τ as $SimTS_{\tau}^{TQ}$ τ and the concatenated version $cSimTS_{\tau}^{TQ} = \{\langle T_s, M_{G_S}, M_{G,l_S}\rangle, T_s \in SimTS_{\tau}^{TQ}\}$. Then, we repeat these queries with different values of τ, thus we have a threshold set $H = \{\tau_1, \dots \tau_h\}$ sorted in ascending fashion ($\tau_i > \tau_{i-1}$), for each query T_Q.

Example 1. An example of a reference trajectory sample dataset with its respective global and local partition MBR is depicted in Table 1. Then, given two sample trajectory queries $\{Q_1, Q_2\}$, we acquire the similar trajectories $SimTS$ with a threshold set $H = \{0.1, 0.5, 1\}$. We show those result of similar trajectories query in Table 2.

Then, we transform the training dataset to fulfill the probabilistic distance approach as follows.

Table 1 Example of reference dataset R

Trajectory	Global MBR (M_G)	Local MBR ($M_{G,l}$)
T_1	1	0
T_2	2	4
T_3	1	0
T_4	2	1
T_5	3	1

Table 2 Example of similarity queries result

Query Traj. T_Q	Dist. Threshold τ	Similar Traj. $SimTS_{\tau}^{TQ}$
Q_1	0.1	T_4
Q_2	0.5	T_4
Q_3	1	T_2, T_4, T_5
Q_1	0.1	T_1
Q_2	0.5	T_1, T_2
Q_3	1	T_1, T_2, T_3

3.2. Learned Index - Probabilistic Distance

As we acquire the training set with three different sets: query set $\mathrm{QS} = \{\mathrm{Q}_1, \mathrm{Q}_2, \dots, \mathrm{Q}_{\mathrm{lq}}\}$ with length lq, the sorted threshold set H, and $SimTS_{\tau}^{TQ}$, $\tau \in H$, $\mathrm{T_Q} \in \mathrm{QS}$, we define the probabilistic distance in the following details.

Suppose that the most similar, or same, trajectories have the distance 0 and the most dissimilar trajectory has the distance approaching infinity, we can model this distance concept to probability, which is bounded from 0 to 1. We assume the same trajectory's similarity probability as 1 and the most dissimilar trajectory as 0. Then, we can model the similarity measure in between using the help of the threshold value set H.

However, we cannot guarantee the similarity of any trajectories that captured by any threshold value larger than the maximum value of H. Thus, we introduce a parameter p that models the similarity/dissimilarity probability in the range $(max(H), \infty)$, where it captures all available trajectories in dataset (including the most dissimilar one). Then, given a query $\mathrm{T_Q}$ and a set of similar trajectories $SimTS_{\tau}^{TQ}$, and the threshold set H, we model the probabilistic distance of between $\mathrm{T_Q}$ and a certain similar trajectory $\mathrm{T_C} \in \mathrm{R}$ in Eq. (1). Note that if we already get a similar trajectory $\mathrm{T_C}$ in a certain threshold τ, it is certain that $\mathrm{T_C}$ is always included if we search similar trajectories with threshold larger than τ. Similarly, we can also apply this probability concept from a query $\mathrm{T_Q}$ to an MBR $\mathrm{M_C}$ (either global or local partition) as $d_p(T_Q | M_C)$. We substitute the condition $T_C \in SimTS_{\tau}^{TQ}$ with $M_C \in cSimTS_{\tau}^{TQ}$.

$$d_p(T_Q|T_C) = \quad (1)$$

$$\begin{cases} 1, & T_C \in SimTS_{\tau}^{T_Q}, \tau = \min(H) \\ p, & T_C \notin SimTS_{\tau}^{T_Q}, \tau = max(H) \\ p + \frac{max(H) - \tau_{i-1}}{max(H)} \times (1-p), & T_C \in SimTS_{\tau}^{T_Q} \wedge T_C \notin SimTS_{\tau-}^{T_Q} \end{cases}$$

Example 2. Given the previous example, we provide the computation result of $d_p(T_Q|T_C)$ (similar trajectory), $d_p(T_Q|M_{G_C})$ (global partitions), and $d_p\left(T_Q\middle|M_{G,l_C}\right)$ (local partitions) in Table 3 (a), (b), and (c), respectively, with $p = 0.05$.

Table 3 Probabilistic distances between query trajectory and elements of the reference dataset

(a) Similar Trajectories

T_Q	$d_p(T_Q\|T_1)$	$d_p(T_Q\|T_2)$	$d_p(T_Q\|T_3)$	$d_p(T_Q\|T_4)$	$d_p(T_Q\|T_5)$
Q_1	0.05	0.48	0.05	1	0.48
Q_2	1	0.905	0.48	0.05	0.05

(b) Global Partitions

T_Q	$d_p(T_Q\|M_1)$	$d_p(T_Q\|M_2)$	$d_p(T_Q\|M_3)$
Q_1	0.05	1	0.48
Q_2	1	0.905	0.05

(c) Local Partitions

T_Q	$d_p(T_Q\|M_{1,0})$	$d_p(T_Q\|M_{2,1})$	$d_p(T_Q\|M_{2,4})$	$d_p(T_Q\|M_{3,1})$
Q_1	0.05	1	0.48	0.48
Q_2	1	0.05	0.48	0.05

Then, if we search a similar trajectory similar in the query set $T_Q \in QS$ with a different threshold τ', we can return similar trajectories $T_i \in SimTS_{\tau'}^{TQ}$ where $d_p(T_Q|T_i) \geq d_p(T_i|T_Q, \tau')$, which formula is described in Eq.(2). This strategy can also be applied to the partitions.

$$d_p(T_i|T_Q, \tau') = \begin{cases} \frac{max(H) - \tau'}{max(H)} \times (1-p), & \tau' \leq max(H) \\ \frac{max(H)}{\tau'} \times p, & \tau' > max(H) \end{cases} \quad (2)$$

Example 3. Suppose we want to query similar trajectories to Q_2 but with DTW distance threshold of 0.3. Using Eq.(2); we get the $d_p(T_i|Q_2, \tau' = 0.3) = 0.715$. Then, we simply acquire T_1 and T_2as similar trajectories by comparing with data from Table 3 (a).

Finally, we describe the learning of our approach. Given the query set QS and its respective probabilistic distance set $d_p(T_Q|T_C)$ or $d_p(T_Q|M_C)$ where $T_Q \in QS$; T_C or $M_C \in cSimTS_{\tau}^{TQ}$; and $\tau \in H$, we feed the model using QS and H as input, p as a parameter, and probabilistic distance set as output. Then, given an unseen query trajectory T'_Q and a new threshold τ', we predict the probabilistic distance of that trajectory and then infer the similar trajectories/partitions $\widehat{cSimTS}_{\tau'}^{T'Q}$.

3.3. Similar Trajectories Prediction

Thus, a machine learning model can 'learn' the trajectory similarity concept from the training set. This concept benefits the model to predict the similar trajectories of the unseen query trajectory in the training set, even without performing the actual query processing with R-tree index nor seeing all the points in the trajectories dataset. However, while it is feasible to output similar trajectories, it suffers from learning a huge set of trajectory probabilities as output. We can avoid this shortcoming by only learning the partitions that contain similar trajectories, whose number is significantly less than the number of trajectories. Then, we can remove trajectories that have distance higher than the threshold later using a similar strategy in DITA.

4. Experiment

We perform the experiment using the learned model to predict the global partitions only as an initial study. We still can acquire the results of a similar trajectory by searching similar trajectories candidates in the predicted global partitions. On the other hand, DITA can still go through a global partition that does not contain any similar trajectory. While our approach learns the associated partitions from the actual returned similar trajectories, thus we may prune unnecessary partitions in the process of searching similar trajectories.

4.1. Dataset

We conducted the experiment on the default DITA example trajectories of taxi driving and the DITA project[1]. The dataset consists of 10,000 trajectories stored in the text file. Then, we build similar trajectory training dataset using 1,381 trajectories that belongs to the default dataset and a threshold set $H = \{0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 2.5, 5, 7.5, 10\}$. For the test dataset, we randomly pick 60 trajectories outside the training set with a new threshold set of $H' = \{0.075, 0.03, 0.4, 6\}$. We present the performance of our learned index given the DITA query result as ground truth in terms of correctness using precision.

4.2. Parameter Setup

As described before, the inputs for the models are the trajectories. However, in this study, we limit the complexity of the inputs to some points, which are the first point, last point, and K pivot points of the query trajectory. We perform this simplification as we would like to apply the most straightforward

[1] https://github.com/TsinghuaDatabaseGroup/DITA

approach possible for the learned index to perform the prediction yet improving them later.

We execute the experiment using two different models: the probabilistic distance with a single model and 2-level recursive models, as described in [3]. The 2-level recursive models use clusters based on the agglomerative hierarchical clustering based on the similar trajectories with threshold $\tau = 5$ with 10 clusters. We use a simple 2-layers fully connected neural network to make the model as simple as possible.

4.3. Hardware and Software Setup

We perform this study in Windows 10 64-bit machine with 16 GB RAM and Intel(R) Core(TM) 3.60GHz. To run the experiments and the program, we use Hadoop-2.6 for Windows, Spark 2.2.0, TensorFlow, and TensorFlow Java API 1.13.1, and Python 3.6. We train the TensorFlow models in Python first, then load it in the Scala program (default in DITA). Thus, we can perform the partition prediction inside the DITA.

4.4. Result

We show the precision of queried similar global partitions using our learned partitioning methods. We denote the displayed method names as Single-Prob and Level-Prob, denoting the single model and 2-level recursive model probabilistic distance approach, respectively.

We see that in Figure 2, the recursive/multi-level indexes slightly surpass the single-level index model. The multi-level index design signifies the approximation to the R-tree approach by the original index.

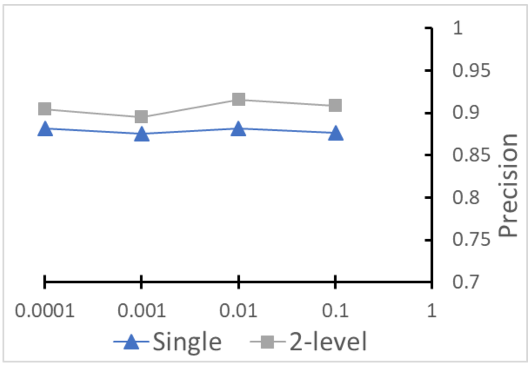

Figure 2 Result of the learned partitioning index

Still, we did not get a perfect approximation to the ground truth, as it is a requirement in the retrieval case. It is important to get the correct answer while we can easily prune out the incorrect answer. Thus, further, we would like to see the improvements by using a sequence-based machine learning approach, such as LSTM neural network. However, we might suffer the burden of complexity using those models. Thus, exploring a suitable compression for the data and the model should be interesting.

5. Conclusion

In this study, we demonstrate the feasibility of using a machine learning model to learn the partitions indexing for searching similar trajectories. We propose two different approaches to model the similarity of the trajectories using the probabilistic distance and multi-label classification. However, the precision of the learned index is still not as perfect as the ground truth. Thus, it is essential to the other models to improve the performance of the learned index.

6. Acknowledgment

This work was partly supported by Capacity Enhancement Program for Scientific and Cultural Exhibition Services through the National Research Foundation of Korea (NRF) funded by Ministry of Science and ICT (NRF-2018X1A3A1069642) and by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2018R1A5A7059549).

REFERENCES

[1] D. Xie and F. P. J. M. Li, "Distributed Trajectory Similarity Search," *PVLDB,* vol. 11, no. 10, pp. 1478-1489, 2017.

[2] Z. Shang, G. Li, and Z. Bao, "DITA: Distributed In-Memory Trajectory Analytics," in *Proceedings of the 2018 International Conference on Management of Data (SIGMOD)*, Houston, TX, 2018.

[3] T. Kraska, A. Beutel, E. Chi, J. Dean, and N. Polyzotis, "The Case for Learned Index Structures," in *Proceedings of the 2018 International Conference on Management of Data (SIGMOD)*, Houston,TX, 2018.

[4] T. Kraska, M. Alizadeh, A. Beutel, E. Chi, A. Kristo, G. Leclerc, S. Madden, H. Mao, and V. Nathan, "SageDB: A Learned Database System," in *9th Biennial Conference on Innovative Data Systems Research*, Asilomar, CA, 2019.

[5] C.-Y. Chiu, A. Prayoonwong, and Y.-C. Liao, "Learning to Index for Nearest Neighbor Search," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* pp. 1-15, 2019.

[6] B.-K. Yi, H. Jagadish and C. Faloutsos, "Efficient Retrieval of Similar Time Sequences Under Time Warping," in *Fourteenth International Conference on Data Engineering (ICDE)*, Florida, 1998.