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

Hani Ramadhan, 201793254 Hudzaifah E. Nursantio, 201983278

***Abstract*—The application of the machine learning-based indexing, called learned index, currently gains popularity because its efficiency and size outperforms 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 seach similar trajectory in a in-memory distributed system. We denote some problem transformation of the indexing procedures in the similar trajectory search to machine learning approach. We demonstrate the feasibility of the learned index on DITA system for searching similar trajectory large trajectory data of moving vehicles. The result proves that the learned partitioning models has the capability to perform the similar trajectory search.**

***Index Terms*—Learned Index, Trajectory, Similarity Search, Distributed System.**

I. Introduction

Thanks to the mass use of mobile device and location based service, especially ride-hailing apps, it is getting easier and more accurate to track moving objects which results in trajectory data. The abundant trajectory data may reveal interesting patterns when analyzed. However, to support this use case, it is important to consider similarity between trajectories. However, it is time consuming and challenging to compute the similarity between an input of single trajectory to a large set of trajectories, especially in 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 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. In addition, the application of machine learning in database has been extended to different features in SageDB [4], including multi-dimensional index. The multi-dimensional index seems appropriate for trajectory data, but it is only limited to range query, not the similarity. On the other hand, other approaches of learned index [5], [6] try to perform nearest neighbor search using probabilistic approach for distance computation using codebook and several data compression mechanism. Thus, it is interesting to incorporate the probabilistic distance modelling onto the similar trajectory search in distributed system.

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

II. Problem Definition

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

A trajectory T is a l-length sequence of positions $T = \{p_1, p_2, p_3, ..., p_l\}$ 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 a certain threshold τ. However, there are several distance measures for two trajectories. In this study, we use the Dynamic Time Warping distance (DTW) [7] distance.

We describe the DTW formula in Eq 1, where trajectory $T_A = \{t_{A_i}, 1 \leq i \leq n\}$ and $T_B = \{t_{B_j}, 1 \leq j \leq m\}$ consist of n and m points respectively, and $\texttt{dist}$ is the Euclidean distance between two points in 2D space. The subset trajectory T^{l-1} is the subset of trajectory of T where the sequence starts at p_1 and ends at p_{l-1}.

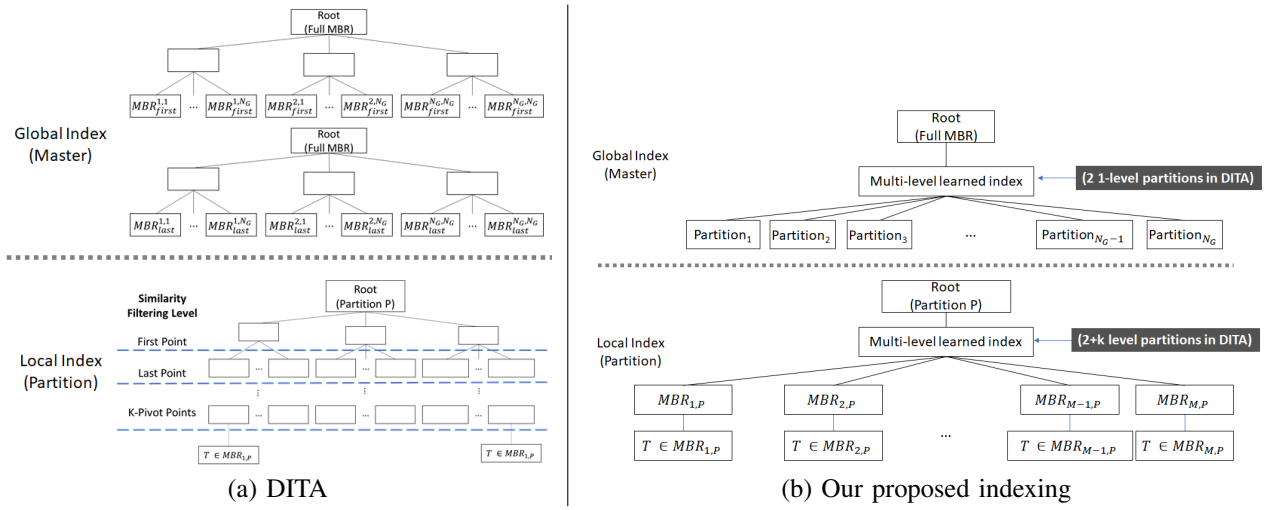
$$\texttt{DTW}(T_A, T_B) = \begin{cases} \sum_{i=1}^{n} \texttt{dist}(t_{A_1}, t_{B_1}), \text{ if } n = 1 \\ \sum_{j=1}^{m} \texttt{dist}(t_{A_1}, t_{B_j}), \text{ if } m = 1 \\ \texttt{dist}(t_m, q_n) + min(\texttt{DTW}(T_A^{n-1}, T_B^{m-1}), \\ \texttt{DTW}(T_A, T_B^{m-1}), \texttt{DTW}(T_A^{n-1}, T_B)), \text{otherwise} \end{cases} \tag{1}$$

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 an effective indexing to quickly search the 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. Hence, as problem, we would like to define the problem transformation for the effective indexing to machine learning approach and then perform similarity search using the learned approach.

III. Proposed System

We propose a system to substitute the similar trajectory search indexing in DITA by a learned index. The difference of 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.

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 the indexing. The first one is the probabilistic distance approach, which we mainly propose, while the second



(a) DITA (b) Our proposed indexing

Fig. 1: Difference of the indexing approach of DITA and ours

TABLE I: Example of reference set R

Trajectory	Global MBR	Local MBR
T_1	1	0
T_2	2	4
T_3	1	0
T_4	2	1
T_5	3	1

TABLE II: Example of result queries

Query Traj.	Dist. Threshold	Similar Traj.
Q_1	0.1	T_4
Q_1	0.5	T_4
Q_1	1	T_2, T_4, T_5
Q_2	0.1	T_1
Q_2	0.5	T_1, T_2
Q_2	1	T_1, T_2, T_3

one is the multi-label approach. But, before we discuss both approaches, we present another important part of this study: building the dataset.

A. Building Similar Trajectory 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 this as $sim(T_Q, \tau) = SimTS_\tau^{T_Q}$ and the concatenated $cSimTS_\tau^{T_Q} = \{\langle T_i, M_g^{T_Q}, M_{g,l}^{T_Q} \rangle | T_i \in SimTS_\tau^{T_Q}\}$. Then, we repeat these queries with different values of τ, given a threshold set $H = \{\tau_1, ..., \tau_{|H|}\}$, sorted in ascending fashion ($\tau_i > \tau_{i-1}$), for each query T_Q.

Example 1. An example of reference trajectory sample dataset with their respective global and local partition MBR is depicted in Table I. 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 similar trajectories in Table II.

Then, we transform the training dataset to be applicable to both of our approaches as follows.

B. Learned Index - Probabilistic Distance

As we acquire the training set with three different sets: query set $QS = \{Q_1, Q_2, ..., Q_{|l_Q|}\}$ with length l_Q, the sorted threshold set H, and $SimTS_\tau^{T_Q}, \tau \in H, T_Q \in QS$, we define the probabilistic distance in the following details.

Suppose that the most similar, or equal, trajectory has 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 equal 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 threshold value set H.

However, we cannot guarantee the similarity of any trajectories that captured by any threshold value larger than maximum value of H. Thus, we introduce a parameter p_{dis} 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 trajectory query T_Q and different set of similar trajectories $SimTS_\tau^{T_Q}, \tau \in H$ with the threshold set H, we model the probabilistic distance of between T_Q and a certain similar trajectory T_a in Eq. 2. Note that if we already get a similar trajectory T_a in a certain threshold τ, it is certain that T_a is always included if we search similar trajectories with threshold larger than τ.

$$p_{sim}(T_Q|T_a) = \begin{cases} 1, T_a \in SimTS_\tau^{T_Q} \bigwedge \tau = min(H) \\ p, T_a \notin SimTS_\tau^{T_Q} \bigwedge \tau \in H \\ \dfrac{max(H) - \tau_{i-1}}{max(H)} \times (1-p) + p, T_a \in SimTS_{\tau_i}^{T_Q} \\ \qquad \bigwedge T_a \notin SimTS_{\tau_{i-1}}^{T_Q} \end{cases} \tag{2}$$

On the other hand, we can also apply this probability concept from a query T_Q to an MBR M_a as $p(T_Q|M_a)$.

TABLE III: Probabilistic distances between query trajectories and the dataset elements

(a) Query and the dataset trajectories

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

(b) Query trajectories and global partitions

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

(c) Query trajectories and local partitions

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

We substitute the result trajectory $T_a \in SimTS_\tau^{T_Q}$ with $M_a \in cSimTS_\tau^{T_Q}$, given $\tau \in H$, where M_a is also applicable to either global or local partition.

Example 2. Given the previous examples, we provide the computation result $p_{sim}(T_Q|T_a)$ (similar trajectory), $p_{sim}(T_Q|MBR_{g_a})$ (global partitions), and $p_{sim}(T_Q|MBR_{g,l_a})$ (local partitions) in Table III (a), (b), and (c), respectively, with $p_{dis} = 0.05$.

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 $T_i \in SimTS_{\tau'}^{T_Q}$ with $p(T_Q|T_i) >= p(T_i|T_Q, \tau')$. The computation of $p(T_i|T_Q, \tau')$, given the threshold set H, is described in Eq. 3. Similar to before, this strategy can be applied also to the partitions.

$$p(T_i|T_Q, \tau') = \begin{cases} \frac{max(H)-\tau'}{max(H)} \times (1-p) + p, \tau' <= max(H) \\ \frac{max(H)}{\tau'} \times p, \tau' > max(H) \end{cases} \tag{3}$$

Example 3. Suppose we want to query similar trajectories to Q_2 but with DTW distance threshold of 0.3. Using the formula in Eq. 3, we get the $p(T_i|Q_1, \tau = 0.3) = 0.715$. Then, we simply acquire T_1 and T_2 as similar trajectories.

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

Next, we will discuss the training and prediction using learned index using the multi-label approach.

C. Learned Index - Multi-label Approach

The multi-label approach also uses the similar trajectory set $SimTS_\tau^{T_Q}$ given $T_Q \in QS$ as query set and $\tau \in H$ as threshold set to perform the training and prediction. However, we do not transform the set to probabilistic distance and directly use the similar trajectories/partitions as the target output for the learning process. Thus, we define the learning process as follows.

The learning process of index with multi-label approach takes pair of trajectory query and threshold (T_Q, τ) as input and directly outputs similar trajectories T_a or partitions that contain similar trajectories M_a $(T_a, M_a) \in cSimTS_\tau^{T_Q}$ as multi-label output. Thus, naturally, the input and the output instances be larger than the probabilistic approach but redundant. However, this may be beneficial as the redundant inputs and outputs are dependent to each other. But, the outputs tend to be less dependent to the query trajectory as the larger threshold should return more dissimilar trajectories even if the query trajectory are different.

Hence, we formalize the prediction of the similar trajectories/partitions given the multi-label approach. Given the unseen query trajectory and threshold pair (T'_Q, τ'), using the learned index, we output the similar trajectories/partitions set $\hat{cSimTS}_{\tau'}^{T'_Q}$.

D. Similar Trajectories Prediction

Thus, a machine learning model can 'learn' the similarity concept from the training set. This benefits the model to predict the similar trajectories of the unseen query trajectory in 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 to learn a huge set of trajectory probabilities as output. We can avoid this shortcoming by only learning the partitions that contain the similar trajectories, which number is significantly less than the number of trajectories. Then, we can remove trajectories that has distance higher than the threshold later using the strategy in DITA.

IV. Experiment

We perform the experiment using the learned model to predict the global partitions only, due to the time constraint. We still can acquire the results of similar trajectory by searching the 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 similar trajectories, thus we may prune some unnecessary partitions in the process of searching similar trajectories.

A. Dataset

We to conduct 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 text file. Then, we build the 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 picks 100 trajectories from the training set and 60 trajectories outside the training set with a threshold set of $H' = \{0.075, 0.03, 0.4, 6\}$. We name both

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

TABLE IV: Parameter Summary

Parameter Name	Values
Layers (Num. of Hidden Layers)	Dense(30)
Learning rate	0.001
Num. of Epochs	2000
Optimizer	Adam
p_{dis}	0.005
Num. Pivot (K)	1

test sets as dataset Test A and dataset Test B, respectively. We present the performance of our learned index given DITA query result as ground truth in terms of correctness using precision metrics.

B. 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 first point, last point, and K pivot points of the query trajectory. We perform this simplification as we would like to apply the simplest approach possible for the learned index to do the prediction, yet improving them later.

We execute the experiment using 4 different models: the probabilistic distance and the multi-label approach with 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 5 clusters. We use a simple 2-layers fully connected neural network to make the model as simple as possible. Table IV summarizes other parameters used in this study.

C. 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.

D. Result

We show the precision of queried similar global partitions using all of our learned partitioning methods and the actual partitions that contains similar trajectories using DITA. Note that, due to time limit again, we did not show the returned global partitions from DITA using its indexing scheme. We denote the displayed method names as Prob, Prob-Rec, ML, and ML-Rec for the probabilistic distance approach with single model and 2-level recursive model, then the multi-label approach with also single and 2-level recursive model, respectively.

We see that in Figure 2, the probabilistic distance approach outperforms the multi-label approaches. This superiority is due to the capability of the probabilistic distance to model the missing thresholds that are not included in the training set. We also see that the recursive/multi-level indexes slightly surpass the single-level index the probabilistic distance approach, but impressively in the multi-label approach. The multi-level index design signifies the approximation to the R-tree approach by the original index.

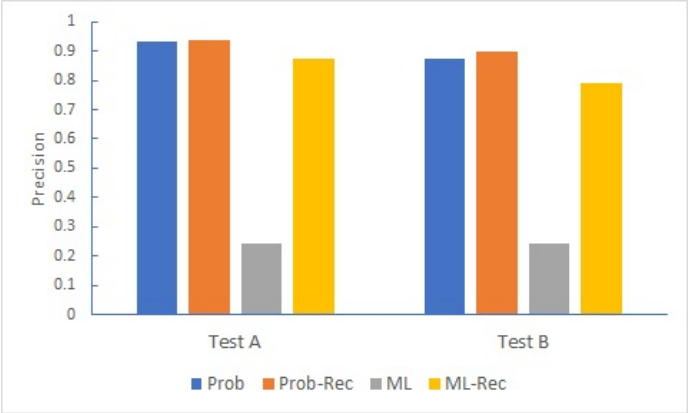


Fig. 2: The precision of querying similar trajectories using learned index

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 that may be influenced by the parameters of probabilistic distance and number of pivot points. On the other hand, as the trajectory data is the sequence of positions, it might be better to use sequence-based machine learning approach, such as Hidden Markov Model or LSTM network. However, we might suffer the burden of complexity using those models.

V. Conclusion

In this study, we demonstrate the feasibility of using machine learning model to learn the partitions indexing for searching similar trajectories. We propose two different approach 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 perfect as the ground truth. Thus, it is important to study the influencing parameters and other models to improve the performance of the learned index.

References

[1] D. Xie, F. Li, and J. M. Phillips, "Distributed trajectory similarity search," *PVLDB*, vol. 10, no. 11, pp. 1478–1489, 2017. [Online]. Available: http://www.vldb.org/pvldb/vol10/p1478-xie.pdf

[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 Conference 2018, Houston, TX, USA, June 10-15, 2018*, 2018, pp. 725–740. [Online]. Available: https://doi.org/10.1145/3183713.3183743

[3] T. Kraska, A. Beutel, E. H. Chi, J. Dean, and N. Polyzotis, "The case for learned index structures," in *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, 2018, pp. 489–504. [Online]. Available: https://doi.org/10.1145/3183713.3196909

[4] T. Kraska, M. Alizadeh, A. Beutel, E. H. Chi, A. Kristo, G. Leclerc, S. Madden, H. Mao, and V. Nathan, "Sagedb: A learned database system," in *CIDR 2019, 9th Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 13-16, 2019, Online Proceedings*, 2019. [Online]. Available: http://cidrdb.org/cidr2019/papers/p117-kraska-cidr19.pdf

[5] C. Chiu, A. Prayoonwong, and Y. Liao, "Learning to index for nearest neighbor search," *CoRR*, vol. abs/1807.02962, 2018. [Online]. Available: http://arxiv.org/abs/1807.02962

[6] A. Prayoonwong, C. Wang, and C. Chiu, "Learning to index in large-scale datasets," in *MultiMedia Modeling - 24th International Conference, MMM 2018, Bangkok, Thailand, February 5-7, 2018, Proceedings, Part I*, 2018, pp. 305–316. [Online]. Available: https://doi.org/10.1007/978-3-319-73603-7_25

[7] B. Yi, H. V. Jagadish, and C. Faloutsos, "Efficient retrieval of similar time sequences under time warping," in *Proceedings of the Fourteenth International Conference on Data Engineering, Orlando, Florida, USA, February 23-27, 1998*, 1998, pp. 201–208. [Online]. Available: https://doi.org/10.1109/ICDE.1998.655778