Explainable Trajectory Representation through Dictionary Learning

Yuanbo Tang[1], Zhiyuan Peng[1], Yang Li[1*]
[1] Tsinghua-Berkeley Shenzhen Institute, Tsinghua University

ABSTRACT

Trajectory representation learning on a network enhances our understanding of vehicular traffic patterns and benefits numerous downstream applications. Existing approaches using classic machine learning or deep learning embed trajectories as dense vectors, which lack interpretability and are inefficient to store and analyze in downstream tasks. In this paper, an explainable trajectory representation learning framework through dictionary learning is proposed. Given a collection of trajectories on a network, it extracts a compact dictionary of commonly used subpaths called "pathlets", which optimally reconstruct each trajectory by simple concatenations. The resulting representation is naturally sparse and encodes strong spatial semantics. Theoretical analysis of our proposed algorithm is conducted to provide a probabilistic bound on the estimation error of the optimal dictionary. A hierarchical dictionary learning scheme is also proposed to ensure the algorithm's scalability on large networks, leading to a multi-scale trajectory representation. Our framework is evaluated on two large-scale real-world taxi datasets. Compared to previous work, the dictionary learned by our method is more compact and has better reconstruction rate for new trajectories. We also demonstrate the promising performance of this method in downstream tasks including trip time prediction task and data compression.

CCS CONCEPTS

• **Information systems → Data mining**.

KEYWORDS

Trajectory representation learning, hierarchical pathlet learning

ACM Reference Format:
Yuanbo Tang[1], Zhiyuan Peng[1], Yang Li[1*], [1] Tsinghua-Berkeley Shenzhen Institute, Tsinghua University . 2023. Explainable Trajectory Representation through Dictionary Learning . In *The 31st ACM International Conference on Advances in Geographic Information Systems (SIGSPATIAL '23), November 13–16, 2023, Hamburg, Germany.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3589132.3625607

1 INTRODUCTION

The development of information technology and the widespread use of mobile devices have produced a large amount of GPS trajectory data. Raw trajectory data typically appears as variable-length ordered sequences, which cannot be directly input into common data mining algorithms. Trajectory representation learning, which means transforming a trajectory into an embedding vector, can standardize trajectory data, extract valuable information from redundant original data, and benefit various downstream tasks including trajectory compression, trip time estimation [1].

Recently, various deep learning based models for trajectory representation learning has been developed. For example, Yang et al. [2] introduced a model based on self-attention (T3S) that automatically adjusts the importance of spatial and structure information for different similarity measures. And they showed the effectiveness for trajectory similarity computation. In addition, in [3] the authors proposed a trajectory encoder-decoder network based on graph attention mechanism to obtain trajectory embedding and evaluate in vehicle trajectories prediction task. Before the emergence of these deep learning based methods, researchers also attempted to explore this field using traditional algorithms, including [4], wherein the authors introduce a pipelined algorithm that extract frequent underlying paths called corridor from trajectories and evaluate it using Minimum Description Length (MDL) score. Besides that, Zou et al. [5] extracted middle level features from trajectories for clustering using a cluster specific Latent Dirichlet Allocation Model. However, the representations generated by previous methods are usually dense vector whose dimensions lack semantic meanings. As a result, it is difficult to interpret the learned representation.

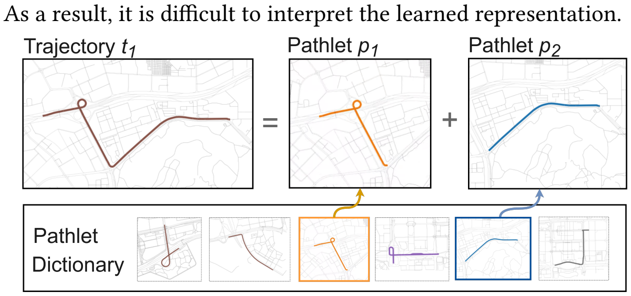

Figure 1: Illustration of pathlet learning: A pathlet dictionary is learned from dataset and each trajectory can be represented by concatenating pathlets chosen from this dictionary.

In this paper, we introduce an explainable trajectory representation method through dictionary learning for trajectories on a network. The network is usually a road map for vehicle trajectories or a grid network for unstructured trajectories, on which trajectory can be projected using map matching [6]. The basic idea is demonstrated in Figure 1. Given a collection of trajectories on a network, it extracts a compact dictionary of commonly used subpaths called "pathlets". Each trajectory can then be reconstructed by concatenating pathlets from the dictionary, similar to the process of constructing a sentence by assembling a group of words. The resulting trajectory representation is a sparse binary vector, where

SIGSPATIAL '23, November 13–16, 2023, Hamburg, Germany

ACM ISBN 979-8-4007-0168-9/23/11.
https://doi.org/10.1145/3589132.3625607

Full text version of this paper is available at github.com/tangyuanbo1/pathlet-learning each dimension corresponds to a pathlet in the learned dictionary and each binary variable indicates whether the corresponding pathlet is used to reconstruct the trajectory. Such design is motivated by the observation that people's travel behavior exhibits remarkable regularity, enabling us to reconstruct majority of trajectories using a small set of movement patterns.

The pathlet representation of trajectories was first explored by Chen et al. [7], who formulate the pathlet learning problem as a combinatorial optimization problem. Solved approximately using dynamic programming, the original formulation is costly to compute and lacks theoretical guarantee. We propose an algorithm using a novel dictioanry learning formulation that provides better optimality and scalability for large trajectory datasets. Specifically, in our formulation, the objective function minimizes the size of the pathlet dictionary and the average number of pathlets required to reconstruct each trajectory at the same time. We propose an efficient solution to this integer programming problem, by first solving its relaxed version and find the integer solution using randomized rounding. To ensure the scalability to large-scale road networks, we further propose a hierarchical representation scheme that compute pathlets of different granularity in multi-scale spatial partition of the map. This algorithm is evaluated on two real-world taxi datasets and some frequent mobility patterns are visualized. We also demonstrate the promising performance of this method in downstream tasks. For example, our method outperforms neural-network based methods by 4.7% in prediction accuracy on trip time prediction.

2 PRELIMINARY

Terminology. Given a dataset T and a roadmap that can be formed as a directed graph $G = (E, V)$, a trajectory $t \in T$ is defined as a sequence of edges e on G. For each t, a path p on G is a candidate pathlet if p is a subpath of t. We denote the set of all candidate pathlets traversed by T as $\overline{P}$.

Given a pathlet dictionary P and a trajectory t, P_{sub} is a subset of P so that t can be represented by concatenating $p \in P_{sub}$. This process is denoted by $t = c(P_{sub})$. Furthermore, the representation cost $rc(t, P)$ refers to the minimal number of elements required to represent t , which is defined as: $rc(t, P) = \min_{P_{sub} \subset P, t = c(P_{sub})} |P_{sub}|$

Problem definition. The goal is to find an optimal dictionary P that minimizes the following two objectives at the same time: 1) the size of the dictionary, as a smaller dictionary contains less redundant information and is therefore more desirable. 2) the average number of elements required to reconstruct trajectories. We use hyperparameter λ to control the trade-off between these two objectives. Therefore, similar to [7], in this paper the pathlet dictionary learning problem is defined as:

$$\min_{P \subset \overline{P}} \; size(P) + \lambda * \sum_{t \in T} rc(t, P) \tag{1}$$

$$s.t. \forall t \in T, \exists P_{\text{sub}} \subset P : t = c(P_{\text{sub}}) \tag{2}$$

3 METHODOLOGY

3.1 Problem Formulation

To formulate the problem defined above using vector notations, we introduce three matrices M, D, R to record the cover relationship among trajectories, edges, and candidate pathlets respectively. Matrix M has dimensions of $|E|$ by $|T|$, where each element $M_{i,j}$ is equal to 1 when the i-th trajectory passes through the j-th edge and 0 otherwise. Matrix D with a size of $|E| \times |\overline{P}|$ is constructed in the same way for the relationship between all candidate pathlets and edges. Similarly, matrix R is a $|\overline{P}| \times |T|$ decision matrix, each entry $R_{i,j} = 1$ if p_i is used to represent t_j, and $R_{i,j} = 0$ otherwise. Based on these definition, the problem can be formulated as follows:

$$\min_{R_{i,j} \in \{0,1\}} \; C(R) = \sum_{i=1}^{|\overline{P}|} max(R_{i,:}) + \lambda * \sum_{i=1}^{|\overline{P}|} \sum_{j=1}^{|T|} |R_{i,j}|$$
$$s.t. DR = M$$

Here $max(R_{i,:})$ refers to the maximum value of i th row of R, which is equal to 1 if any trajectory utilizes p_i to represent itself. In other words, $max(R_{i,:}) = 1$ means that candidate pathlet p_i is selected as an element of the dictionary. We reuse the notation P to represent the matrix form of the dictionary, which is a submatrix formed by selected columns of D, $P = D[:, \{i \mid max(R_i, :) = 1\}]$ and therefore $size(P) = \sum_{i=1}^{|\overline{P}|} max(R_{i,:})$. The constraint $DR = M$ corresponds to the setting that each trajectory should be reconstructed using pathlets. In this optimization problem, the dictionary and the assignment relationship will be optimized at the same time. It is worth noting that the pathlet learning problem described above is NP-hard in most cases. Therefore, an effective algorithm is required to obtain good approximated solutions.

3.2 Pathlet Dictionary Learning with Randomized Rounding

The proposed algorithm consists of two main steps. Firstly, we relax the binary constraint, which transforms the original optimization problem into a convex optimization problem. Therefore, the global optimal solution R^* can be found easily by the projected gradient descent algorithm. Then a randomized rounding step is carried out to obtain the final solution R^r. The whole procedure is shown in the following pseudocode of Algorithm 1.

Algorithm 1 Pathlet dictionary learning by randomized rounding

Require: M: trajectory matrix; D: pathlet matrix; R_0: initial solution; ϵ, θ: hyperparameters; C: objective function;
Ensure: Optimal binary matrix R^r

```
# Step1, we compute the fractional solution R*.
initial R_0 = 0;
repeat
    compute gradient directions g_k = ∇C(R_k);
    update the decision matrix R_{k+1} = R_k − α g_k;
    clip the result to make sure 0 ≤ R_k ≤ 1;
until (|C(R_k) − C(R_{k−1})| < ϵ)
#Step2, we compute rounded solution R^r based on R*.
Sample R^r with P(R^r_{i,j} = 1) = min(1, θ R*_{i,j})
```

Probabilistic Bound. We claim that R^r satisfies

$$P[C(R^r) \le 2\theta \frac{\lambda + 1}{\lambda} C(R^*) \; and \; DR^r \ge M] \ge \frac{1}{2} - |T| e^{-\theta}$$

In practice, $|\overline{P}|$ can be quite large. We pre-filter out less frequently used candidates to alleviate computational burden. Please refer to Appendix D for details.

This inequality means that the probablity that a solution with low cost can be found and all trajectories will be covered at the same time is lower bounded by a positive constant. Therefore, we can repeat the randomized rounding process to get a series of $\{R_1^r, R_2^r...\}$ until find a satisfactory solution. The proof can be found in appendix part A.

3.3 Hierarchical Pathlet Learning

Candidate pathlet space consists of all segments of trajectories from dataset, whose size is usually huge in real-world dataset and make it time-consuming to get the solution. On the other hand, Multi-scale dictionaries of pathlets and trajectory representations can help people gain a deeper understanding of traffic characteristic. To enhance the scalability of the original algorithm, we introduce a hierarchical method called "pathlet of pathlets" to reduce the computation complexity and generate multi-scale trajectory representations.

Specifically, we first partition the roadmap into different levels of granularity using axis-aligned binary space partitioning. Starting from the bottom of the partition tree, we compute the k-th level pathlet dictionary P_k as the union of dictionaries computed in all k-th level cells. Next, we use the k-th level pathlet representation of each trajectory as the input, and compute the $(k-1)$-th level pathlet dictionaries. This iterative process can be repeated to generate multi-scale pathlets that capture movement patterns.

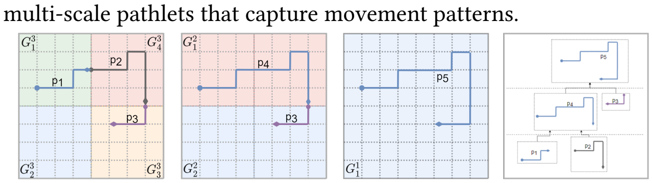

Figure 2: Illustration of the hierchical pathlet representation. Here G_i^k refers to the i-th region of the k-th layer.

3.4 Representing New Trajectories

Once we obtain a set of dictionaries at multiple scales, we can use them together in representing new trajectories. We define a unified dictionary matrix P' by concatenating the dictionary matrices by column. The size of dictionary P' is therefore equal to the number of columns of P'.

For any new trajectory, it can be mapped to a new representation space using P' . To be specific, representation vector is obtained by solving:

$$\min_{r_i \in \{0,1\}} \sum_{i=1}^{size(P')} |r_i| \quad s.t. P'r = m$$

Here m represents the vector recording the edges covered by a new trajectory, and r denotes the representation vector that we aim to solve for. This problem can be viewed as a simplified version of the original problem because the dictionary is fixed at this moment. We solve it using the same strategy described before: first compute the optimal fractional solution r^* using gradient descent and then round it to get the final binary solution.

4 EXPERIMENTS

4.1 Numerical Performance

4.1.1 The Performance Comparison with Previous Work. Our research largely follows the problem formulation described in [7] but we adopt different formulation and method. In that paper, the authors first relaxed $max(R_{i,:})$ to $R_{i,j}$, and then solved it using dynamic programming, which is simple and effective. However, this relaxation operation resulted in an redundant dictionary, providing us with room for improvement especially when λ is small.

In this experiment, hyperparameter λ and θ are set as 0.1 and $\frac{1}{4}ln(2|T|)$ respectively, and we only randomly sample 3 times using strategy described before. As is shown in Table 1, our approach generates a more compact and effective dictionary compared to dynamic programming methods, reducing the dictionary size by 43.01% and 36.36% respectively on two datasets and the representation cost is relatively lower. At the same time, it is observed that the cover ratio is very close to 1, here θ is set as $\frac{1}{4}ln(2|T|)$ instead of $ln(4|T|)$ because in the experiment we found that the method can still produce a feasible solution with low cost within 3 random sampling cycles, which further validates the effectiveness of previously derived probability bound.

4.1.2 Reconstruction using Multi-scale Dictionary. The hierarchical framework enables us to learn multi-level pathlet dictionaries on arbitrarily large maps and datasets with limited computational resources. In this section, we validate the above statement by comparing the performance of the dictionary directly learned on the whole map (denoted using P_{direct}) and the dictionaries generated by hierarchical framework on test data. Specifically, we randomly selected 10,000 trajectories from the Futian district as train set to learn the dictionary and tested it on another 10,000 trajectories. In Table 2, P_2 represents dictionaries learned on regions of the 2-th layer and $P_1 + P_2$ refers to multi-scale dictionaries. The performance of P_{direct} can be considered as ground truth to some extent, although it comes with significant computational resource consumption. We can observe that compared to only using P_2, the reconstruction cost is much lower when using the multi-scale dictionary. The performance of the multi-scale dictionary is closer to that of P_{direct}, but consumes only 54% of the GPU memory resources compared to the training of P_{direct} and the computation time is reduced by 20%.

4.2 Visualization of Pathlet Dictionary

Some frequent pathlets are visualized in Figure 3 to intuitively verify whether the algorithm finds common mobility patterns or not. For example, Figure 3 (c) is a pathlet corresponding to turning left on the overpass. Figure 3 (e) depicts Praça Mouzinho de Albuquerque, which is one of the famous attractions in Porto. These pathlets have semantic meaning consistent with our cognition in life and reveal common mobility patterns shared by numerous trajectories.

4.3 Performance on Downstream Tasks

4.3.1 Application in Trip Time Prediction. To demonstrate the effectiveness and usability of the representation vector, we utilize a simple GBDT model to predict trajectory time whose input is the combination of trajectory embedding vector and the time encoding vector. We use mean absolute error (MAE) between the predicted result and the ground truth (in seconds) as the metric to train the simple GBDT model. The performance of all evaluated models are summarized in the table 3. It can be observed that our proposed algorithm ensures explainability of the results without compromising accuracy. One possible reason why our method outperforms others

Due to the space limit, details of experiments setup can be found in appendix part B.

Table 1: The performance comparison with previous work.

Dataset	Method	Train Phrase			Test Phrase	
		Dictionary size/|T|	Representation cost	Cover ratio	Representation cost	Cover ratio
Porto	DP	1.79	2.14	100%	4.19	88.03%
	our method	1.02(-43.01%)	2.00(-6.54%)	99.6%	2.75	93.6%(+5.57%)
Shenzhen	DP	1.21	2.88	100%	3.01	93.9%
	our method	0.91(-36.36%)	2.75 (-4.51%)	99.1%	3.02	95.4% (+1.5%)

Table 2: The performance using different dictionaries.

Dictionary	Size	Represent-ation Cost	GPU Memory *	Running Time
P_{direct}	13076	5.21	42.8G	1.5h
P_2	10470	6.05	23.1G	0.9h
$P_1 + P_2$	12631	5.33	23.1G	1.2h

* GPU memory here refers to the size of GPU memory needed for training instead of storage of dictionary.

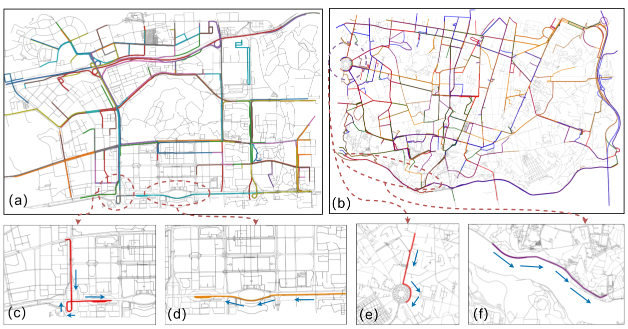

Figure 3: Top 300 frequent pathlets in two grids.

is that our vectors are naturally sparse, which makes it more robust on the test set and easier to train the model. This demonstrates the simplicity and effectiveness of our method, as well as its broad prospects in the field of application.

Table 3: The performance comparison with previous work.

dataset	method	MAE	MAPE	RMSE	RMSLE
Porto	[8]	-	-	-	0.41
	[9]	171.97	-	-	-
	Ours	163.9	26.2	199.74	0.35
Porto (short trips)	[10]	39.25	14.74	52.35	-
	Ours	32.87	9.37	37.59	0.08

4.3.2 *Application in Data Compression.* Learning a dictionary and reconstructing trajectories using elements from this dictionary can also be considered as a process of data compression. In [4] the authors described an evaluation method based on Minimum Description Length (MDL) to measure the compression performance:

$$MDL \text{ score } = \frac{\mathcal{L}(C) + \mathcal{L}(\mathcal{D} \mid C)}{\mathcal{L}(\mathcal{D})}$$

Here $\mathcal{L}(.)$ refers to the size of a data collection in bits. $\mathcal{D}$ and C are used to denote the dataset and the corridor set, a concept similar to pathlets. $\mathcal{D} \mid C$ refers to the representation of the original trajectory using corridor. Compared to the previous method's score of 0.27 reported in [4], our method achieved a score of 0.21. One possible reason is our objective function and MDL score are consistent, whereas method in [4] based on LDA does not optimize the MDL score explicitly. This experiment indicates that transforming trajectories into pathlets form can effectively compress data, facilitating easier storage and transmission.

5 CONCLUSION AND FUTURE WORK

In this study, we reformulated the problem of pathlet learning from a collection of trajectories and solved it using a novel dictionary learning based method, resulting in a hierarchical and explainable representation of trajectories with theoretical probability bound. We tested our algorithm on two large-scale datasets. The output dictionary of pathlets provides us with deeper insight into mobility patterns. We also demonstrate how the pathlet could benefit downstream tasks such as trip time estimation and trajectory compression. In future work, we will adapt our algorithm to represent trajectories in other domains, improve the numerical optimization, and further advance the theoretical analysis.

ACKNOWLEDGMENTS

This study is supported by the Tsinghua SIGS Scientific Research Start-up Fund (Grant QD2021012C).

REFERENCES

[1] T. H.-N. CAO Han-Lin, "Survey on Trajectory Representation Learning Techniques," *Journal of Software*, vol. 32, no. 5, pp. 1461–1479, Jan. 2021.
[2] P. Y. e. al, "T3S: Effective Representation Learning for Trajectory Similarity Computation," in *2021 IEEE 37th International Conference on Data Engineering (ICDE). IEEE, 2021.*, 2021.
[3] Y. Liang and Z. Zhao, "NetTraj: A Network-Based Vehicle Trajectory Prediction Model With Directional Representation and Spatiotemporal Attention Mechanisms," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 14 470–14 481, Sep. 2022.
[4] N. Zygouras and D. Gunopulos, "Corridor Learning Using Individual Trajectories," in *2018 19th IEEE International Conference on Mobile Data Management (MDM)*, Jun. 2018, pp. 155–160.
[5] J. Zou, Y. Cui, F. Wan, Q. Ye, and J. Jiao, "A cluster specific latent dirichlet allocation model for trajectory clustering in crowded videos," in *2014 IEEE International Conference on Image Processing (ICIP)*, Oct. 2014, pp. 2348–2352.
[6] P. Newson and J. Krumm, "Hidden Markov map matching through noise and sparseness," in *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems.* Seattle Washington: ACM, Nov. 2009, pp. 336–343.
[7] C. Chen, H. Su, Q. Huang, L. Zhang, and L. Guibas, "Pathlet learning for compressing and planning trajectories," in *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems.* Orlando Florida: ACM, Nov. 2013, pp. 392–395.
[8] H. T. Lam, E. Diaz-Aviles, A. Pascale, Y. Gkoufas, and B. Chen, "(Blue) Taxi Destination and Trip Time Prediction from Partial Trajectories."
[9] T.-Y. Fu and W.-C. Lee, "Trembr: Exploring Road Networks for Trajectory Representation Learning," in *ACM Transactions on Intelligent Systems and Technology*, vol. 11, Feb. 2020, pp. 1–25.
[10] G. Jin, M. Wang, J. Zhang, H. Sha, and J. Huang, "STGNN-TTE: Travel time estimation via spatial–temporal graph neural network," *Future Generation Computer Systems*, vol. 126, pp. 70–81, Jan. 2022.