mathematics

MDPI

Article

Non-Convex Metric Learning-Based Trajectory Clustering Algorithm

Xiaoyan Lei [1] and Hongyan Wang [2,*]

[1] Network Engineering School, Zhoukou Normal University, Zhoukou 466001, China; 20031035@zknu.edu.cn
[2] School of Computer Science and Technology, Zhejiang Sci-Tech University, Hangzhou 310018, China
* Correspondence: wanghongyan@zstu.edu.cn

Abstract: To address the issue of suboptimal clustering performance arising from the limitations of distance measurement in traditional trajectory clustering methods, this paper presents a novel trajectory clustering strategy that integrates the bag-of-words model with non-convex metric learning. Initially, the strategy extracts motion characteristic parameters from trajectory points. Subsequently, based on the minimum description length criterion, trajectories are segmented into several homogeneous segments, and statistical properties for each segment are computed. A non-convex metric learning mechanism is then introduced to enhance similarity evaluation accuracy. Furthermore, by combining a bag-of-words model with a non-convex metric learning algorithm, segmented trajectory fragments are transformed into fixed-length feature descriptors. Finally, the K-means method and the proposed non-convex metric learning algorithm are utilized to analyze the feature descriptors, and hence, the effective clustering of trajectories can be achieved. Experimental results demonstrate that the proposed method exhibits superior clustering performance compared to the state-of-the-art trajectory clustering approaches.

Keywords: trajectory clustering; bag-of-words model; metric learning; trajectory segmentation

MSC: 90C26



Academic Editor: José Antonio Sanz

Received: 28 December 2024
Revised: 20 January 2025
Accepted: 21 January 2025
Published: 24 January 2025

Citation: Lei, X.; Wang, H. Non-Convex Metric Learning-Based Trajectory Clustering Algorithm. *Mathematics* **2025**, *13*, 387. https://doi.org/10.3390/math13030387

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

With the rapid development of global positioning systems (GPS), radio frequency identification (RFID), wireless sensor networks (WSN), and other related positioning technologies, the moving trajectory can be acquired and processed to extract information such as social attributes and physical properties [1]. As an important part of the spatiotemporal trajectory mining task, trajectory clustering aggregates similar trajectories to form trajectory clusters to characterize potential behavioral features of mobile targets and then infer travel intentions, mine movement patterns, predict locations, and detect anomalies; thus, it is of great importance in practical applications [2].

Trajectory clustering can be classified into two primary categories: distance-based and density-based clustering [3]. Density-based clustering groups trajectories according to the density of data points in the trajectory space, identifying clusters as regions of high density separated by areas of low density [4]. This approach is advantageous for discovering clusters of arbitrary shapes and effectively handling outliers. Distance-based clustering, on the other hand, emphasizes measuring the spatial distances between trajectories to group similar ones together, which typically relies on geometric distances, aiming to minimize intra-cluster variances while maximizing inter-cluster differences [5]. Density-based clustering is particularly useful for identifying similar vehicle trajectories because the vehicle

trajectory data encompass spatiotemporal motion attribute parameters. Consequently, density-based clustering is considered here for classifying vehicle trajectories. The density-based trajectory clustering model comprises two key steps: similarity measurement and a clustering algorithm.

On the basis of the methodologies and procedures, trajectory similarity measurement algorithms can be classified into traditional and learning-based approaches. Among these, the Hausdorff distance effectively addresses inconsistencies arising from variations in point density along the trajectories and demonstrates robustness against minor perturbations [6]. Nevertheless, it should be acknowledged that the Hausdorff distance can be sensitive to noise in the data. The Fréchet distance uniquely captures the continuity and order of points, rendering it highly sensitive to the actual shapes of the trajectories [7]. However, it is important to note that, due to its reliance on the maximum point-to-point distances under optimal reparameterization, the Fréchet distance can be susceptible to outliers or anomalies in the data. The one-way distance (OWD) framework quantifies the directional discrepancy from one trajectory to another and is inherently asymmetric [8]. However, the average of the OWD distances in both directions between two trajectories yields a symmetric measure. Dynamic time warping (DTW) effectively identifies similar trajectories after local time scaling, addressing inconsistencies arising from varying sampling rates and time scales [9]. However, the calculation of the DTW distance requires continuous trajectory sampling points, making it sensitive to noise. The longest common subsequence (LCSS) distance is particularly advantageous for comparing high-dimensional time series or spatiotemporal trajectories, as it demonstrates robustness against noise due to its reliance on a threshold-based similarity criterion [10]. However, determining the optimal thresholds poses a significant challenge. The edit distance on real sequence (EDR) is particularly advantageous because it can be normalized to a value between 0 and 1, facilitating comparative analysis [11]. Nevertheless, the accuracy of the EDR measurement is highly dependent on the appropriateness of the chosen threshold. The edit distance with real penalty (ERP) integrates both Lp-norms and the edit distance, thereby enhancing the measurement of sequence similarity by accounting for local shifts and the treatment of gaps [12].

In recent years, deep representation learning models such as word2vec, Glove [13], and latent data assimilation (LA) [14] have significantly influenced tasks like part-of-speech tagging and machine translation. These models address various limitations of traditional similarity measurement approaches and effectively resolve multiple challenges faced by conventional trajectory similarity measures. For example, sequence-to-sequence (Seq2Seq) models have increasingly been applied in the domain of road traffic, particularly for measuring trajectory similarities [15]. Originating from natural language processing, these models have been effectively adapted to the spatiotemporal domain to encode and decode sequences of locations, thereby capturing the intricate movement patterns in urban environments. The convolutional auto-encoder (CAE) model, initially developed for image processing, has been extended to analyze spatial trajectories [16]. By transforming sequences of locations into image-like representations, the CAE model facilitates the identification of movement patterns that are valuable in urban and maritime mobility applications. These models encode trajectories with different lengths and sampling rates in various regions into fixed-length feature vectors and then measure trajectory similarity using Euclidean distance. However, since the Euclidean metric only takes into account the differences in each characteristic, the resultant similarity measurement error is relatively large, thus reducing the clustering accuracy. In view of this, a large margin nearest neighbor (LMNN)-based metric learning method was developed in [17], which fully utilizes the sample labeling information to construct a Mahalanobis distance metric matrix for measuring sample similarity. However, it fails to consider the substantial increase in computational

complexity caused by the semi-positive definite constraint of the metric matrix and the overfitting resulting from metric matrix learning. Regarding this, a low-rank metric learning method based on LMNN was put forward in [18], which adds a low-rank constraint to the algorithm of [17] to regularize the learning model, thereby reducing the computational complexity and avoiding overfitting. Nevertheless, because it approximates the rank of the metric matrix using the kernel norm, and the kernel norm has a poor approximation performance compared with the rank of the non-convex function, the accuracy of the regularization model is decreased, and the clustering performance is impaired.

In clustering, a set of objects is partitioned into subgroups based on a specified similarity measure without prior knowledge of the dataset. Clustering algorithms are categorized into five main types according to the techniques used to define clusters. The primary function of a partitional algorithm is to determine a division into k clusters that optimizes the selected criteria. The K-means algorithm is a prominent partitional algorithm [19]. Hierarchical clustering algorithms generate a collection of nested clusters that form a hierarchical tree structure, such as balanced iterative reducing and clustering using hierarchies (BIRCH) [20]. Grid-based methods quantize the object space into a finite grid structure. Examples of multi-level grid-based clustering algorithms include statistical information grid (STING) [21] and WaveCluster [22]. Model-based approaches employ statistical models to determine the best fit for the data. For instance, COBWEB hypothesizes a model for each cluster to identify the optimal fit [23]. The density-based spatial clustering of applications with noise (DBSCAN) algorithm clusters objects into meaningful subclasses based on density [24]. In this approach, the density threshold is defined by the maximum radius of the neighborhood (ε, Eps) and the minimum number of points required in an ε-neighborhood of a given object (MinPts). Ordering points to identify the clustering structure (OPTICS) is built upon the DBSCAN algorithm, extending its capabilities to identify nested clusters and reveal the hierarchical structure of clusters [25]. Unlike DBSCAN, OPTICS differs in the order in which objects are processed within the dataset. Clustering using references and density (CURD) captures the shape and extent of clusters using reference points and subsequently analyzes the data based on these references [26]. CURD is capable of discovering arbitrarily shaped clusters and exhibits robustness against noise. Its high efficiency renders it particularly suitable for mining large-scale datasets. Agrawal et al. developed and validated an enhanced spatiotemporal clustering algorithm, Spatiotemporal-OPTICS (ST-OPTICS), by extending the OPTICS algorithm [27]. Husch et al. developed correlation-based clustering of big spatiotemporal datasets (CorClustST) [28], a clustering algorithm that leverages the concept of correlation for analyzing large spatiotemporal datasets.

Focusing on the abovementioned problems, a trajectory clustering method with the bag-of-words model and metric learning is developed in this paper. The proposed method first extracts the motion parameters of trajectory points. Subsequently, the trajectory is partitioned into several optimal homogeneous trajectory segments under the principle of minimum description length (MDL), and the statistic features of each segment are then computed to form a comprehensive feature set representing these trajectory segments. In the following sections, the metric learning method with the non-convex low-rank constraint is proposed to improve the similarity metric performance. Additionally, the segmented trajectories are encoded into fixed-length vectors to derive their feature descriptors using both the bag-of-words model and the developed metric learning approach. Finally, the segmented trajectory feature descriptors are aggregated by exploiting K-means and the proposed metric learning method to achieve trajectory clustering. The experimental results demonstrate that the proposed approach outperforms state-of-the-art techniques in terms of clustering efficacy.

Along the abovementioned lines, the main contributions of this work can be summarized as follows:

(1) Based on the remarkable property in which the Laplace norm can effectively approximate the rank of a metric matrix, the non-convex metric learning method with nearest neighbor structure preservation and low-rank constraints is developed to optimize the metric matrix to improve the accuracy of the sample similarity metric in the process of trajectory feature encoding and clustering. The resultant non-convex issue can be efficiently addressed by leveraging the difference of convex functions algorithm (DCA) and alternating direction method of multipliers (ADMM) approaches.
(2) Raw trajectories can be divided into several homogeneity sub-trajectories based on the extracted kinematic parameters of trajectory points under the minimum description length principle, and its feature set is obtained by calculating the statistic characteristics of sub-trajectories.
(3) The segmented trajectory can be encoded as a fixed-length vector by leveraging the bag-of-words model, along with the developed metric learning method, to obtain its feature descriptor. Subsequently, all feature descriptors can be clustered using the K-means clustering algorithm in conjunction with the proposed metric learning approach, thereby facilitating effective trajectory clustering.

The remainder of this work is organized as follows: Section 2 presents the metric learning model incorporating manifold and low-rank constraints, along with an effective approach utilizing DCA and ADMM to address the resultant non-convex problem. In Section 3, we detail the extraction of trajectory point motion parameters, segment feature extraction, trajectory feature encoding, and clustering processes. The effectiveness of the proposed method is verified via numerical examples in Section 4. Finally, conclusions are drawn in Section 5.

2. Non-Convex Low-Rank Metric Learning

The overall framework of the proposed method is depicted in Figure 1. The proposed algorithm mainly comprises the following five modules: Firstly, based on the outstanding property in which Laplace norm can considerably approximate the rank of the metric matrix, the non-convex low-rank metric learning method is proposed to optimize the metric matrix, thereby enhancing the accuracy of the sample similarity metric during the trajectory feature encoding and clustering process; Furthermore, the motion parameters of trajectory points are extracted to obtain the features of each trajectory point; Moreover, the trajectory can be divided into several optimal homogeneous segments by applying the unsupervised segmentation algorithm, and the statistical features of each segment are extracted. In addition, the trajectory features can be encoded by employing the proposed metric learning method and bag-of-words model. Finally, trajectory clustering can be accomplished by using the proposed metric learning method and the K-means method.

The Laplace norm is initially introduced in this section. Subsequently, a metric learning model based on nearest neighbor structure preservation and low-rank constraints is elaborated. Finally, the DCA and ADMM approaches can be utilized to address the resulting optimization problem.

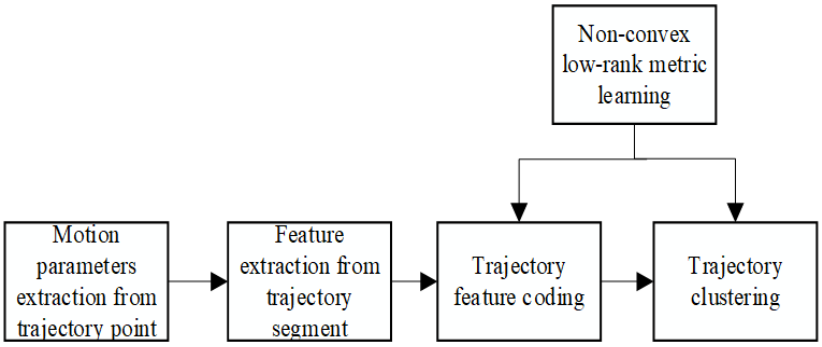


Figure 1. Trajectory clustering method based on bag-of-words model and metric learning.

2.1. Laplace Norm

Assuming that the singular value decomposition (SVD) of the matrix $\boldsymbol{X} \in \mathbb{R}^{m\times n}$ can be expressed as $\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\mathrm{T}}$, in which, $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_{r_\mathbf{X}}]$ and $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_{r_\mathbf{X}}]$ are the left and right singular matrices, respectively, $\sum = \mathrm{diag}(\sigma_1, \sigma_2, \cdots, \sigma_{r_\mathbf{X}})$ is the diagonal matrix, $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{r_X} \geq 0$, $r_\mathbf{X}$ is the rank of $\mathbf{X}$, and $\sigma_i(\mathbf{X})$ is the ith singular value of $\mathbf{X}$, then the Laplace norm of the matrix $\mathbf{X}$ can be defined as follows:

$$\|\mathbf{X}\|_\gamma = \sum_{i=1}^{r} \phi(\sigma_i(\mathbf{X})) \tag{1}$$

where $\phi(\sigma_i(\mathbf{X})) = 1 - e^{\frac{-\sigma_i(\mathbf{X})}{\gamma}}$ is the penalty function, and $\gamma > 0$ is the order of the Laplace norm.

It is known that the Laplace norm, $\|\mathbf{X}\|_\gamma$, is a pseudo-norm of the approximate L_0 norm [29], which has the following excellent properties: (1) rank approximation: $\lim_{\gamma\to 0}\|\mathbf{X}\|_\gamma = rank(\mathbf{X})$; (2) unitary invariance: for any unitary matrix $\mathbf{U} \in \mathbb{R}^{m\times m}$, $\mathbf{V} \in \mathbb{R}^{n\times n}$, $\|\mathbf{X}\|_\gamma = \|\mathbf{UXV}\|_\gamma$ is satisfied; and (3) positivity: $\|\mathbf{X}\|_\gamma \geq 0$, when and only when $\mathbf{X} = 0$ and $\|\mathbf{X}\|_\gamma = 0$.

It is evident that the issue associated with the rank function, $rank(\mathbf{X})$, is a complicated non-convex problem that is difficult to solve directly, and therefore, the approximation method is usually used to achieve an effective solution [30]. The rank function approximation can be divided into the following two categories: convex and non-convex approximation. Among these, the commonly used convex approximation function is the kernel norm, and the kernel norm optimization can be considered a convex problem and, thereby, can be solved directly. Nevertheless, the kernel norm is a biased estimator, which might over-penalize large singular values, subsequently resulting in the inability of the kernel norm optimization to achieve the optimal solution. Meanwhile, the commonly exploited non-convex approximation functions include the Laplace norm, logarithmic determinant function, Geman norm, etc., which can be regarded as nearly unbiased estimations of the rank function, thus enhancing the accuracy of the solution to the rank optimization problem. To acquire a superior approximation of the rank function, Figures 2 and 3 present the comparison curves of the approximation effects of the aforementioned functions. It can be observed from Figure 2 that the Laplacian norm can approximate the rank function more precisely with the increase in singular values compared to the kernel norm, and the approximation effect becomes better when $\gamma = 0.1$. It is discernible from Figure 3 that the approximation effect of the Laplacian norm ($\gamma = 0.1$) to the rank function is superior to that of other non-convex functions. Accordingly, the Laplace norm with $\gamma = 0.1$ can be selected to approximate the rank function.

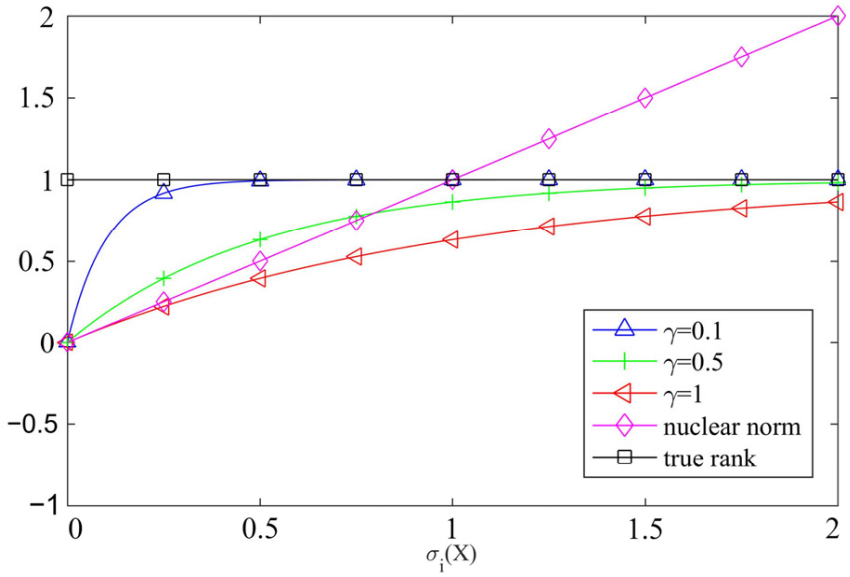


Figure 2. Comparison of different-order Laplace and kernel parametric rank function approximation.

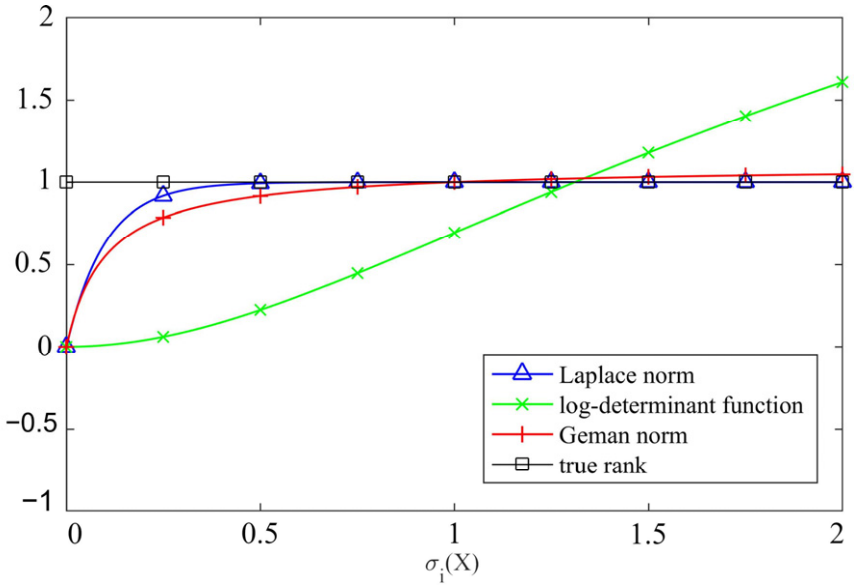


Figure 3. Comparison of common non-convex rank approximation functions.

To facilitate solving the non-convex problems associated with the Laplace norm, the definition of the subgradient of the Laplace norm can be given as follows:

$$\partial\|\mathbf{X}\|_{\gamma} = \left\{\mathbf{U}diag(\mathbf{X})\mathbf{V}^{\mathrm{T}} : \mathbf{X}_i = \frac{d(\phi(\sigma_i(\mathbf{X}))}{d(\sigma_i(\mathbf{X}))}, i = 1, 2, \ldots, r_X\right\} \tag{2}$$

Based on Equation (1), the equation above can be rewritten as follows:

$$\partial\|\mathbf{X}\|_{\gamma} = \left\{\mathbf{U}diag(\mathbf{X})\mathbf{V}^{\mathrm{T}} : \mathbf{X}_i = e^{\frac{-\sigma_i(\mathbf{X})}{\gamma}}/\gamma, i = 1, 2, \cdots, r_X\right\} \tag{3}$$

2.2. Metric Learning Manifold Constraints

It is well-known that metric learning plays a significant role in tasks such as classification, clustering, and information retrieval [2]. Existing metric learning methods can be classified into the following three categories: supervised, unsupervised, and semi-supervised metric learning. Among these, supervised metric learning utilizes all training sample label information to constrain the learning model, while unsupervised metric learning does not require any label information and aims at mining the potential low-dimensional manifold of the data to preserve the local topology to the greatest extent possible; thus, it can be expressed from the perspective of dimensionality reduction and manifold learning. Semi-supervised metric learning combines the advantages of both supervised and unsupervised metric learning by leveraging the geometric information of unlabeled samples and the label information of the limited labeled samples to learn distance metrics [31]. Considering that the training samples in clustering tasks usually do not contain label information, manifold learning can be exploited to measure sample similarity in this section. The manifold learning methods mainly include Laplacian eigenmaps (LE), locality preserving projections (LPP), and locally linear embedding (LLE), which can fully preserve the local structure information and maintain the nearest neighbor relationship of the data after dimensionality reduction [32]. Additionally, the following manifold assumption is proposed in [33]: if two samples have the nearest neighbor relationship in Euclidean space, the two also maintain this relationship in Mahalanobis space. Based on the mentioned above, metric learning with manifold constraints can be exploited here to improve the trajectory clustering performance.

Given a dataset, $\mathbf{X}_{\text{sa}} = \{\mathbf{X}_i\}_{i=1}^{n_{sa}} \subset \mathbb{C}^D$, where n_{sa} is the number of samples and D is the data dimension, the Mahalanobis distance between $\mathbf{X}_i$ and $\mathbf{X}_j$ can be expressed as follows:

$$d_{\mathbf{M}}^2(\mathbf{X}_i, \mathbf{X}_j) = (\mathbf{X}_i - \mathbf{X}_j)^{\mathrm{T}}\mathbf{M}(\mathbf{X}_i - \mathbf{X}_j) = \left\|\mathbf{X}_i - \mathbf{X}_j\right\|_{\mathbf{M}}^2 \tag{4}$$

where $\mathbf{M} \in \mathbb{C}^{D\times D}$ is the semi-positive definite metric matrix [17].

Accordingly, the unsupervised metric learning under the manifold constraint can be illustrated as follows:

$$\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(\mathbf{X}_i - \mathbf{X}_j)^{\mathrm{T}}\mathbf{M}(\mathbf{X}_i - \mathbf{X}_j)\mathbf{W}_{ij} \tag{5}$$

where $\mathbf{W} = \left[\mathbf{W}_{ij}\right]_{n\times n}$ is the weighted matrix, which characterizes the sample association relationship, i.e., the sample space manifold, and can be expressed as follows:

$$\mathbf{W}_{ij} = \begin{cases} 1, \mathbf{X}_i \textit{ and } \mathbf{X}_j \textit{ have connect} \\ 0, \mathbf{X}_i \textit{ and } \mathbf{X}_j \textit{ do not have connect} \end{cases} \tag{6}$$

Based on the relevant property of the matrix trace, Equation (5) can be reformulated as follows:

$$\begin{aligned} &\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(\mathbf{X}_i - \mathbf{X}_j)^{T}\mathbf{M}^{T}(\mathbf{X}_i - \mathbf{X}_j)\mathbf{W}_{ij} = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(\mathbf{M}(\mathbf{X}_i - \mathbf{X}_j))^{T}(\mathbf{X}_i - \mathbf{X}_j)\mathbf{W}_{ij} \\ &= \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} tr(\mathbf{M}(\mathbf{X}_i - \mathbf{X}_j)(\mathbf{X}_i - \mathbf{X}_j)^{T})\mathbf{W}_{ij} = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} tr(\mathbf{M}\mathbf{C}_{ij})\mathbf{W}_{ij} \end{aligned} \tag{7}$$

where $\mathbf{C}_{ij} = (\mathbf{X}_i - \mathbf{X}_j)(\mathbf{X}_i - \mathbf{X}_j)^{\mathrm{T}}$, $tr(\cdot)$ is the matrix trace operator.

2.3. Metric Matrix Low-Rank Constraint

It is known from [34,35] that the computational complexity of the metric learning algorithm is $O(D^2)$ at least, where D is the data dimension. This indicates that the high-dimensional metric matrix significantly increases the computational complexity. Additionally, full-rank metric learning is prone to causing model overfitting. Therefore, low-rank constrained metric matrix regularization is proposed in [18] to mitigate the computational complexity and prevent overfitting. However, a biased kernel norm rank approximation is utilized in [18], thereby limiting the improvement of metric performance. It is clear from the abovementioned that the Laplace norm can approximate the matrix rank more accurately and thus improve the accuracy of the regularized model. Consequently, the metric performance can be enhanced in this section based on the low-rank constraint of the Laplace norm.

With Equation (1), the Laplace norm of the matrix $\mathbf{M}$ can be depicted as follows:

$$\|\mathbf{M}\|_{\gamma} = \sum_{i=1}^{n} \phi(\sigma_i(\mathbf{M})) \tag{8}$$

As a consequence, the objective function associated with the Laplace norm of the metric matrix under the low-rank constraint can be constructed as follows:

$$\min_{\mathbf{M}} \|\mathbf{M}\|_{\gamma} \tag{9}$$

2.4. Solving Metric Learning Issue with Low-Rank and Manifold Constraints

Based on (7) and (9), the metric learning issue under the low-rank and manifold constraints can be constructed as follows:

$$\min_{\mathbf{M}} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} tr(\mathbf{M}\mathbf{C}_{ij})\mathbf{W}_{ij} + \lambda \|\mathbf{M}\|_{\gamma} \tag{10}$$

where λ is the regularization parameter.

Obviously, model (10) is a rather complicated non-convex problem associated with $\mathbf{M}$, which is difficult to solve directly by using traditional convex optimization methods. According to [36], DCA can linearize non-convex problems into convex ones and then solve them iteratively with global convergence, while ADMM can handle non-smooth issues with strong convergence. Therefore, the abovementioned complicated non-convex issue can be solved by using DCA and ADMM, and the proposed method consisting of DCA outer loop and ADMM inner loop can be illustrated as follows.

Outer loop: DCA decomposes the non-convex problem $f(\mathbf{M})$ into the difference of two convex functions, i.e., $g(\mathbf{M}) - h(\mathbf{M})$. In addition, it is known from the definition of a subgradient that if $h(\mathbf{M})$ exists a subgradient, it must satisfy the following conditions:

$$h(\mathbf{M}) \leq h(\mathbf{M}_k) + \langle \partial h(\mathbf{M}_k), \mathbf{M} - \mathbf{M}_k \rangle \tag{11}$$

where k is the number of iterations, and $\partial h(\mathbf{M}_k)$ denotes the subgradient of $h(\mathbf{M})$ at $\mathbf{M}_k$.

In one iteration, a linear approximation to $h(\mathbf{M})$ in Equation (11) can be used to acquire a convex expression, and then the problem can be minimized and solved. Specifically, $f(\mathbf{M})$ can be solved iteratively using the following model:

$$\mathbf{M}_{k+1} = \operatorname*{argmin}_{\mathbf{M}} g(\mathbf{M}) - (h(\mathbf{M}_k) + \langle \partial h(\mathbf{M}_k), \mathbf{M} - \mathbf{M}_k \rangle) \tag{12}$$

Thus, problem (10) can be solved iteratively based on the following function:

$$\begin{cases} g(\mathbf{M}) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} tr(\mathbf{M}\mathbf{C}_{ij})\mathbf{W}_{ij} \\ h(\mathbf{M}) = -\lambda \|\mathbf{M}\|_{\gamma} \end{cases} \tag{13}$$

The optimization variables are iteratively updated as follows:

$$\mathbf{M}_{k+1} = \underset{\mathbf{M}}{\operatorname{argmin}} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} tr(\mathbf{M}\mathbf{C}_{ij})\mathbf{W}_{ij} + \lambda \left\langle \partial \|\mathbf{M}_k\|_{\gamma}, \mathbf{M} \right\rangle + \mathbf{C} \tag{14}$$

where $C = \lambda(\|\mathbf{M}_k\|_{\gamma} - \left\langle \partial \|\mathbf{M}_k\|_{\gamma}, \mathbf{M}_k \right\rangle)$.

Since $\mathbf{M}_k$ is the last iterative solution, C is a constant; thus, the solution to issue (10) can be obtained by solving the following convex problem:

$$\mathbf{M}_{k+1} = \underset{\mathbf{M}}{\operatorname{argmin}} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} tr(\mathbf{M}\mathbf{C}_{ij})\mathbf{W}_{ij} + \lambda \left\langle \partial \|\mathbf{M}_k\|_{\gamma}, \mathbf{M} \right\rangle \tag{15}$$

It can be seen that problem (15) is non-smooth, and thus, no closed-form solution can be obtained directly [37], while it is known from the mentioned above that ADMM can solve the non-smooth problem effectively; thus, ADMM can be employed to solve this issue iteratively.

The termination condition of DCA can be expressed as follows [38]:

$$\text{RelErr1} = \frac{\|\mathbf{M}_{k+1} - \mathbf{M}_k\|_2}{\max\{\|\mathbf{M}_k\|_2, 1\}} \leq \varepsilon_{\text{outer}} \tag{16}$$

where $\|\cdot\|_2$ is the Euclidean norm, $\max\{\cdot\}$ is the taking maximum operator, and $\varepsilon_{\text{outer}} > 0$ is the outer loop convergence threshold.

Inner loop: To facilitate solving the issue, the auxiliary variables, $\mathbf{N}$, are introduced, and then issue (15) can be transformed into the following optimization problem with equation constraints:

$$\underset{\mathbf{M}}{\operatorname{argmin}} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} tr(\mathbf{N}\mathbf{C}_{ij})\mathbf{W}_{ij} + \lambda \left\langle \partial \|\mathbf{M}_k\|_{\gamma}, \mathbf{M} \right\rangle \ s.t.\ \mathbf{M} - \mathbf{N} = 0 \tag{17}$$

The augmented Lagrangian function of the previous problem can be formulated as follows:

$$L(\mathbf{M}, \mathbf{N}, \mathbf{Y}, \mu) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} tr(\mathbf{N}\mathbf{C}_{ij})\mathbf{W}_{ij} + \lambda \left\langle \partial \|\mathbf{M}_k\|_{\gamma}, \mathbf{M} \right\rangle + \langle \mathbf{Y}, \mathbf{M} - \mathbf{N} \rangle + \frac{\mu}{2} \|\mathbf{M} - \mathbf{N}\|_F^2 \tag{18}$$

where $\mathbf{Y}$ is the Lagrange multiplier, and $\mu > 0$ is the penalty parameter.

Issue (18) can be solved iteratively based on ADMM, which can be illustrated in detail as follows:

(1) Fixing $\mathbf{M}$, the following optimization problem can be solved to update $\mathbf{N}$:

$$\mathbf{N}_{l+1} = \underset{\mathbf{N}}{\operatorname{argmin}} L(\mathbf{M}_l, \mathbf{N}, \mathbf{Y}_l, \mu_l) \tag{19}$$

that is,

$$\mathbf{N}_{l+1} = \underset{\mathbf{N}}{\operatorname{argmin}} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} tr(\mathbf{N}\mathbf{C}_{ij})\mathbf{W}_{ij} + \lambda \left\langle \partial \|\mathbf{M}_k\|_{\gamma}, \mathbf{M}_l \right\rangle + \langle \mathbf{Y}_l, \mathbf{M}_l - \mathbf{N} \rangle + \frac{\mu_l}{2} \|\mathbf{M}_l - \mathbf{N}\|_F^2 \tag{20}$$

Discarding the terms independent of the optimization variables, (20) can be recast as follows:

$$\mathbf{N}_{l+1} = \underset{\mathbf{N}}{\operatorname{argmin}} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} tr(\mathbf{N}\mathbf{C}_{ij})\mathbf{W}_{ij} + \frac{\mu_l}{2} \left\| \mathbf{M}_l - \mathbf{N} + \frac{\mathbf{Y}_l}{\mu_l} \right\|_F^2 \tag{21}$$

Taking the derivative of the abovementioned object function and making it zero, one can obtain the following equation:

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mathbf{C}_{ij}\mathbf{W}_{ij} - \mathbf{Y}_l - \mu_l(\mathbf{M}_l - \mathbf{N}) = 0 \tag{22}$$

Then, the following equation can be obtained:

$$\mathbf{N}_{l+1} = \mathbf{M}_l + {\mu_l}^{-1}(\mathbf{Y}_l - \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mathbf{C}_{ij}\mathbf{W}_{ij}) \tag{23}$$

(2) Fixing $\mathbf{N}$, the following optimization issue can be solved to update $\mathbf{M}$:

$$\mathbf{M}_{l+1} = \underset{\mathbf{M}}{\operatorname{argmin}} L(\mathbf{M}, \mathbf{N}_{l+1}, \mathbf{Y}_l, \mu_l) \tag{24}$$

that is,

$$\mathbf{M}_{l+1} = \underset{\mathbf{M}}{\operatorname{argmin}} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} tr(\mathbf{N}_{l+1}\mathbf{C}_{ij})\mathbf{W}_{ij} + \lambda \left\langle \partial \|\mathbf{M}_k\|_\gamma, \mathbf{M} \right\rangle + \langle \mathbf{Y}_l, \mathbf{M} - \mathbf{N}_{l+1} \rangle + \frac{\mu_l}{2} \|\mathbf{M} - \mathbf{N}_{l+1}\|_F^2 \tag{25}$$

Discarding the term unrelated to the optimization variables, we have the following equation:

$$\mathbf{M}_{l+1} = \underset{\mathbf{M}}{\operatorname{argmin}} \lambda \left\langle \partial \|\mathbf{M}_k\|_\gamma, \mathbf{M} \right\rangle + \frac{\mu_l}{2} \|\mathbf{M} - \mathbf{N}_{l+1} + {\mathbf{Y}_l}/{\mu_l}\|_F^2 \tag{26}$$

Similarly, one can obtain the following equation:

$$\mathbf{M}_{l+1} = \mathbf{N}_{l+1} - {\mu_l}^{-1} \left[\mathbf{Y}_l + \lambda \partial \|\mathbf{M}_k\|_\gamma \right] \tag{27}$$

(3) Fixing $\mathbf{M}$ and $\mathbf{N}$, $\mathbf{Y}$ can be updated using the following equation:

$$\mathbf{Y}_{l+1} = \mathbf{Y}_l + \mu_l(\mathbf{M}_{l+1} - \mathbf{N}_{l+1}) \tag{28}$$

Similar to DCA, the ADMM convergence condition can be set as follows:

$$\text{RelErr2} = \frac{\|\mathbf{M}_{l+1} - \mathbf{M}_l\|_2}{\max\{\|\mathbf{M}_l\|_2, 1\}} \leq \varepsilon_{\text{inner}} \tag{29}$$

where $\varepsilon_{\text{inner}} > 0$ is the inner loop convergence threshold. Meanwhile, to speed up the iteration, the penalty parameter μ can be updated using the following equation:

$$\mu_{l+1} = \min\{\beta * \mu_l, \mu_{\max}\} \tag{30}$$

where $\beta \in (0, \frac{1+\sqrt{5}}{2})$ is the magnification, and $\mu_{\max}$ is the upper bound of μ_l.

In summary, the specific steps of the proposed non-convex low-rank metric learning method are presented in Algorithm 1.

Algorithm 1. Non-convex low-rank metric learning method

Input: training samples $\mathbf{X}_{sa}$, $\varepsilon_{\text{inner}}$, $\varepsilon_{\text{outer}}$, $\mu_{\max}$, β, λ
Initialization: $\mathbf{M}_0 = \mathbf{I}$ ($\mathbf{I} \in \mathbb{R}^{D\times D}$ for unit matrix), $k = 0$, $l = 0$, $\mathbf{Y}_0 = 0$, μ_0
while RelErrl $> \varepsilon_{\text{outer}}$ do
1. Calculate $\partial\|\mathbf{M}_k\|_\gamma$ with (3)
while RelErr2 $> \varepsilon_{\text{inner}}$ do
2. Fixing $\mathbf{M}_k$, update $\mathbf{N}_l$ with (23)
3. Fixing $\mathbf{N}_l$, update $\mathbf{M}_l$ with (27)
4. Fixing $\mathbf{M}_l$ and $\mathbf{N}_l$, update $\mathbf{Y}_l$ with (28)
5. Update penalty parameter μ_l with (30)
6. $l = l + 1$
end while
7. $\mathbf{M}_{k+1} = \mathbf{M}_{l+1}$
8. $k = k + 1$
end while
Output: $\mathbf{M}_{k+1}$

3. Trajectory Clustering with the Bag-of-Words Model and Metric Learning

3.1. Trajectory Point Motion Parameters Extraction

Given a trajectory dataset, $X_{tr} = \{\mathbf{Tr}_i\}_{i=1}^{n_{tr}}$, where n_{tr} is the number of trajectories, the trajectory, $\mathbf{Tr}_i$, is composed of a finite number of position vectors: $\mathbf{Tr}_i = \left[\mathbf{p}_{i,t_1}, \cdots, \mathbf{p}_{i,t_k}, \cdots, \mathbf{p}_{i,t_n}\right]^{\mathrm{T}}$, where $\mathbf{p}_{i,t_k} = (x_{i,t_k}, y_{i,t_k})$ is the ith target position at the t_k moment. Each trajectory point can calculate the corresponding motion parameters based on the time stamp and position information, i.e., velocity s_{i,t_k}, acceleration a_{i,t_k}, and steering angle r_{i,t_k}. Concretely, we obtain the following equations:

$$s_{i,t_k} = \frac{\sqrt{(x_{i,t_k} - x_{i,t_{k-1}})^2 + (y_{i,t_k} - y_{i,t_{k-1}})^2}}{t_k - t_{k-1}} \tag{31}$$

$$a_{i,t_k} = \frac{s_{i,t_k} - s_{i,t_{k-1}}}{t_k - t_{k-1}} \tag{32}$$

$$r_{i,t_k} = arctan\frac{y_{i,t_k} - y_{i,t_{k-1}}}{x_{i,t_k} - x_{i,t_{k-1}}} \tag{33}$$

The steering angle of the first trajectory point of each trajectory can be set as $r_{i,t_1} = 0$ here.

3.2. Trajectory Segment Feature Extraction

As a fundamental step in motion data analysis, trajectory segmentation can divide the trajectory into several meaningful subsequences. Effective trajectory segmentation methods not only offer high-quality features to characterize the behavior of the moving target but also significantly reduce the computational complexity. Common trajectory segmentation methods can be categorized into the following three types: supervised, unsupervised, and semi-supervised trajectory segmentation. Supervised trajectory segmentation depends on predefined rules, labels, and thresholds. Nevertheless, the segmentation criteria cannot be specified in some cases and depend on the characteristics of the trajectory dataset. Unsupervised trajectory segmentation does not require target information and achieves segmentation only through the constructed cost function. Semi-supervised trajectory segmentation is a combination of supervised and unsupervised trajectory segmentation [39].

Since the trajectory data of the clustering task typically lacks label information, the trajectory can be segmented by employing the unsupervised trajectory segmentation algorithm GRASP-UTS proposed in [40]. Minimal distortion and maximal compression are exploited in GRASP-UTS to determine the optimality of the homogeneous segments. Specifically, achieving minimal distortion in trajectory segmentation requires that the trajectory segments display a high degree of homogeneity with respect to their characteristics, ensuring that points within each segment exhibit substantial similarity. Conversely, achieving maximal compression in trajectory segmentation entails minimizing the number of resultant trajectory segments, thereby optimizing the selection of landmarks. Consequently, the concepts of minimal distortion and maximal compression are orthogonal, that is, selecting all trajectory points as landmarks minimizes distortion but sacrifices compression efficiency. On the contrary, choosing a single landmark for the entire trajectory maximizes compression but introduces significant distortion.

From the description above, it is clear that the homogeneity and number of segments are dependent on the distortion and compression, while the clustering relies on the similarity between segments, which is influenced by the homogeneity; hence, the clustering results are inevitably affected by segmentation. In order to mitigate this effect as much as possible, the MDL principle is carefully designed in GRASP-UTS to achieve homogeneity in the segments.

According to the methods mentioned above, the main steps of the trajectory segmentation are briefly described below.

(1) Several sample trajectory reference points (representative feature points) can be randomly selected, and several continuous trajectory segments are generated in the vicinity of the reference points.

(2) A cost function is constructed under the MDL principle based on the characteristics of minimum distortion and maximum compression segmentation, and the resultant cost function is minimized to adjust the trajectory segment boundaries.

(3) The position and number of reference points can be iteratively updated, and steps (1)–(2) are executed until the best homogeneous trajectory segments can be obtained.

Several best homogeneous trajectory segments will be generated after segmentation, and each segment contains several trajectory points. Subsequently, the statistical features of each segment can be extracted based on the motion parameters of the acquired trajectory points, namely, minimum, maximum, mean, median, and standard deviation, to obtain the global features of the trajectory segments; that is, each segment can be represented by a 15-dimensional feature vector. Finally, each trajectory is characterized by the corresponding segment features. So far, all the trajectories in the X_{tr} can acquire the corresponding feature set, $TS = \{\mathbf{L}_i\}_{i=1}^{n_{tr}}$, after segmentation and statistic feature extraction, where $\mathbf{L}_i$ is the ith trajectory feature. The number of segments for each trajectory, $\mathbf{L}_i$, may be different in TS, and the segments of each trajectory can comprise a segment feature set, $LD = \{\mathbf{C}_{ij}\}$, in which $\mathbf{C}_{ij}$ is the ith segment statistical feature of the jth trajectory, $i = 1, \cdots, n_{tr}$. The value range of each trajectory, j, is different, which can be determined on the basis of the number of the corresponding segments. Additionally, TS and LD will be used to encode the trajectory feature.

3.3. Trajectory Feature Encoding

The bag-of-words model was originally proposed as a feature coding method for text retrieval, and nowadays, it is widely used in data mining, image processing, vision detection, and other fields [41]. The bag-of-words model can encode all trajectories with different lengths into fixed-length feature vectors, and the process is straightforward to implement. Therefore, trajectory features can be encoded by utilizing the improved bag-of-

words model, which mainly consists of K-means-based dictionary generation and nearest neighbor search-based trajectory feature quantization. The dictionary generated by the former will be utilized in the search process of the latter for quantization. The encoding steps can be elaborated in detail as follows.

Dictionary generation:

(1) With the trajectory segment feature set, LD, the proposed low-rank metric learning method is used to obtain the metric matrix suitable for LD.

(2) Some LD samples can be randomly selected as the initial dictionary, i.e., the initial clustering prime.

(3) The Mahalanobis distance from other samples to the dictionary words can be calculated with the obtained metric matrix, and then they can be divided into clusters represented by the words under the principle of minimum distance, thereby obtaining several clusters.

(4) The mass center of each cluster can be recalculated as words.

(5) The sum of squared distances from each sample within the cluster to the mass center of the corresponding cluster, i.e., the sum of squared intra-cluster errors, is calculated. If it is less than the given threshold, the words of each cluster can be combined to construct the dictionary; otherwise, steps (2)–(4) need to be repeated.

Trajectory feature quantification:

(1) Initialize the trajectory feature vector as $\mathbf{0}$, and the length of the vector is identical to the number of words in the obtained dictionary.

(2) Search all trajectory segments, L_i, $i = 1, \cdots, n_{tr}$, which is essentially a nearest neighbor search process. Specifically, the dictionary words can be regarded as the reference point, the distances between trajectory segments and all words are calculated, and each segment is replaced with the corresponding word according to the nearest neighbor principle, and then the number of similar segments can be counted on the basis of the word order to acquire the feature histogram associated with $\mathbf{L}_i$, i.e., feature vector $\mathbf{z}_i$.

(3) All trajectory features within TS can be quantized to construct an encoding trajectory feature set, $Z = \{\mathbf{z}_i\}_{i=1}^{n_{tr}}$, which will be exploited for the trajectory clustering.

3.4. Trajectory Clustering

The trajectories can be clustered after feature extraction and encoding, the main steps of which can be delineated as follows:

(1) The metric matrix suitable for Z can be obtained by using the developed non-convex low-rank metric learning algorithm with the trajectory encoding feature set Z.

(2) The trajectory similarity can be measured by using the obtained metric matrix, and the trajectory clusters $G = \{g_i\}_{i=1}^{m}$ can be achieved by exploiting the K-means algorithm, where g_i is the centroid of the ith trajectory class, and m is the number of classes.

According to the description above, the proposed clustering algorithm with the bag-of-words model and metric learning is shown in Algorithm 2.

Algorithm 2. Bag-of-words model and metric learning-based clustering approach

Input: Trajectory dataset X_{tr}

1. Extract trajectory point motion parameters based on Equations (31)–(33).
2. Segment trajectories by using GRASP-UTS, and extract trajectory segment features.
3. Encode trajectory features based on the bag-of-words model and the metric matrix obtained via the proposed low-rank metric learning method.
4. Cluster trajectories by employing K-means and the metric matrix acquired using the developed low-rank metric learning method.

Output: Trajectory class clusters G

3.5. Analysis of Computational Complexity

The computational complexity of the proposed method mainly involves the computation of (3), (23), (27), (28), and (30). Because (23) and (28) involve only basic matrix addition and subtraction, their computational cost is negligible. Therefore, the primary computational burden of the proposed algorithm lies in the calculation of the sub-gradients for (3) and (27). According to [42], the computational complexity of calculating the subgradient is $O(D_r^2)$, where D_r is the rank of the data matrix. Therefore, the overall computational complexity of the proposed algorithm can be estimated as $O(D_r^2)$. As a comparison, all of the computational complexities of the comparative approaches, that is, Hausdorff, DTW, and LCSS, can be denoted as $O(D^2)$, where D is the dimension of the data matrix [43]. Due to the rank of the matrix being less than or equal to its dimension, the computational complexity of the proposed method is reduced compared to that of the comparison algorithms.

4. Experimental and Simulation Analysis

4.1. Experimental Dataset and Environment

The Federal Highway Administration (FHWA) launched the next-generation simulation (NGSIM) program in 2005 to collect microscopic traffic data during different time periods of four road sections, including I-80, us101, lankershim, and Peachtree, which are captured by an overhead camera with a sampling interval of 0.1 s to obtain a vehicle trajectory dataset [44]. The trajectory data mainly includes information such as vehicle number, position, type, speed, and acceleration. The vehicle types include motorcycles, small cars, and trucks. Additionally, the speed limit of the lankershim section is 35 mph, the collection time of the lankershim dataset is from 8:30 a.m.–9:00 a.m., and the collection is situated at the intersection [45].

In this section, the Lankershim dataset is filtered according to the following requirements: (1) the ranges of the horizontal and vertical coordinates are $X \in (-80, 80\text{ft})$ and $Y \in (300, 500\text{ft})$, respectively, (2) small cars are taken as the study object, (3) several consecutive trajectory points contain different location information, (4) trajectory points with speed exceeding 35 mph are excluded, (5) data on each vehicle shall not be less than 10, and (6) trajectories outside the intersection area and abnormal trajectories are ruled out.

A total of 95 trajectories of straight, left-turn, and right-turn types are generated after filtering. Therefore, 20 trajectories of each type can be obtained, and the trajectories are smoothed with the Savitzky–Golay filter to filter out noise. The real trajectory dataset after processing is shown in Figure 4. The experimental environment is as follows: Intel Core i7-7700 3.60 Ghz processor and 8 GB of memory. The parameter values of the developed algorithm can be set as follows: $\gamma = 0.1$, $\lambda = 0.1$, $\beta = 1.618$, $\mu_0 = 10^{-3}$, $\mu_{\max} = 10^{10}$, and $\varepsilon_{\text{inner}} = \varepsilon_{\text{outer}} = 10^{-3}$.

4.2. Evaluation Metrics

Precision, recall, and accuracy are employed herein as evaluation metrics to verify the effectiveness of the proposed algorithm, and the relevant evaluation metrics are defined as follows [46].

It is known that the dataset, X_{tr}, can be divided into k classes, i.e., $\mathcal{T} = \{T_1, T_j, \cdots T_k\}$, X_{tr} can be clustered to produce r class clusters, i.e., $\mathcal{C} = \{C_1, C_i, \cdots, C_r\}$, and the relationship between r and k is indefinite. With this, one can obtain the following evaluation metrics:

The accuracy rate of the class cluster C_i, Prec_i, can be formulated as follows:

$$\text{Prec}_i = \frac{1}{n_i} \max_{j=1}^{k} \{n_{ij}\} = \frac{n_{ij_i}}{n_i} \tag{34}$$

where $n_i = |C_i|$ is the number of trajectories of C_i, $n_{ij} = |C_i \cap T_j|$ is the number of overlapping trajectories between C_i and T_j, and n_{ij_i} is the maximum number of trajectories of T_j belonging to C_i and T_j.

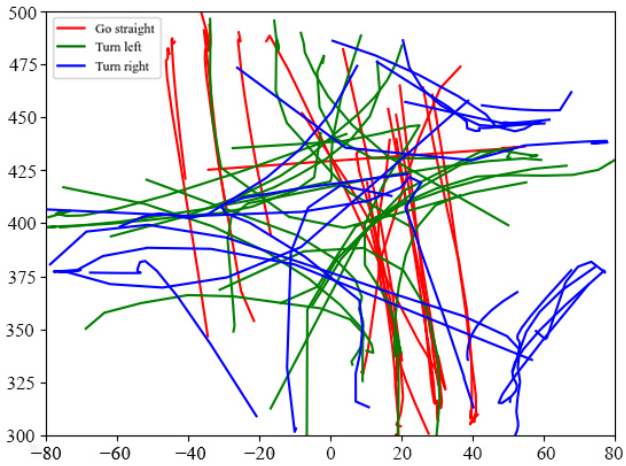


Figure 4. Visualization of real trajectory dataset.

The recall rate of the class cluster C_i, recall_i, can be depicted as follows:

$$\text{recall}_i = \frac{n_{ij_i}}{|T_{ji}|} = \frac{n_{ij_i}}{m_{ji}} \tag{35}$$

where $m_{ji} = |T_{ji}|$ is the number of trajectories of n_{ij_i} belonging to class T_j.

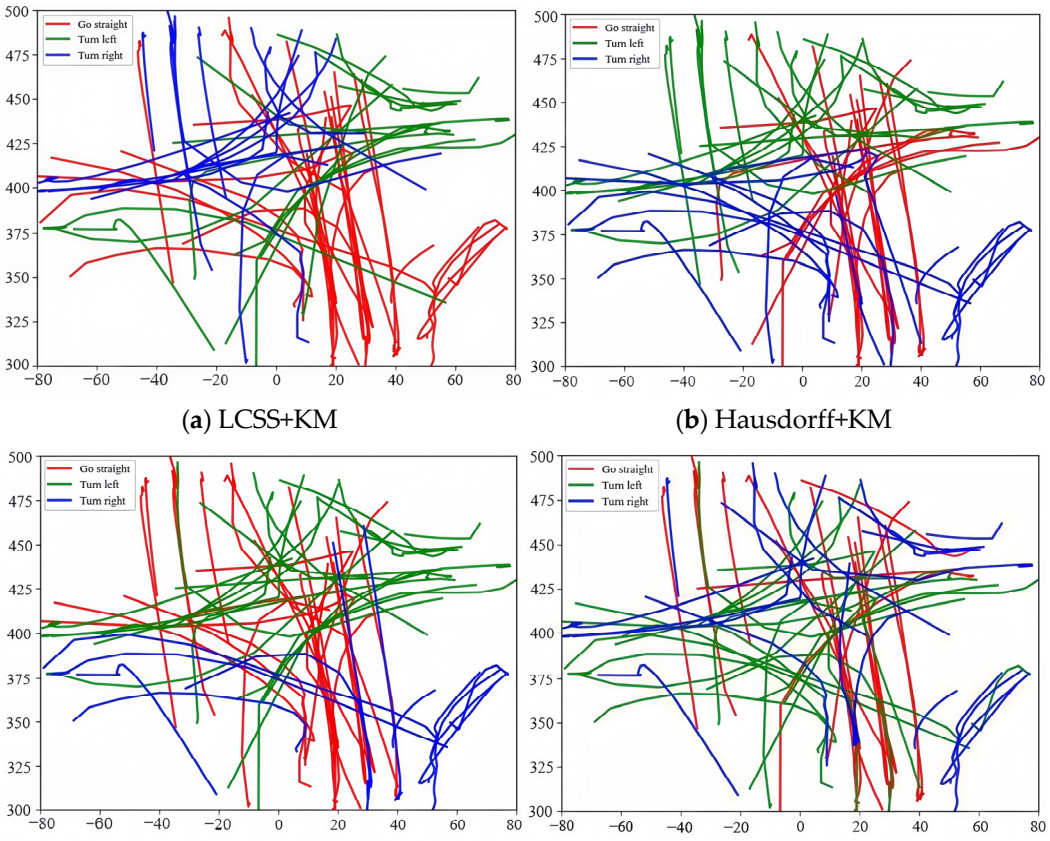
The cluster accuracy rate of X_{tr} can be expressed as follows:

$$\text{Accuracy} = \frac{\sum_{j=1} n_{ij_i}}{n_{tr}} \tag{36}$$

4.3. Simulation Results and Analysis

The effectiveness of the proposed algorithm is verified by comparing it with three types of trajectory clustering methods based on distance measurement, including the LCSS distance [10], SSPD distance [12], and Hausdorff distance [6], to measure trajectory similarity, and then the clustering can be achieved by using the K-Medoids (KM) approach with better noise robustness. The clustering results acquired by each algorithm are shown in Figure 5. It is obvious from Figure 5a–c that the upper left-turn trajectory (green) is clustered using LCSS + KM as the right-turn trajectory (blue), while the lower left-turn trajectory (blue) is clustered as the straight turn trajectory (red), the middle left-turn trajectory (green) is clustered using Hausdorff + KM as the straight turn trajectory (red), and the upper left-turn trajectory (blue) is clustered using SSPD + KM as the left-turn trajectory (green). It can be seen from Figure 5d that the proposed algorithm can more accurately gather the trajectories within different regions, such as the upper right and lower right-turn trajectories (blue trajectories). The findings mentioned above can be attributed to the following fact: the compared algorithms only extract some features, such as trajectory direction and shape, for clustering, and thus, the clustering performance is poor, while the developed approach exploits the optimized metric matrix and bag-of-words model to promote the feature coding

performance and lifts the clustering effect based on the optimized metric matrix; therefore, the trajectory clustering performance of different regions can be improved significantly.



(**a**) LCSS+KM (**b**) Hausdorff+KM

(**c**) SSPD+KM (**d**) The developed method

Figure 5. Visualization of clustering results.

Table 1 presents the precision and recall rates (columns 2 to 4), as well as the accuracy rates (column 5), obtained by each algorithm. It is clear from Table 1 that the clustering accuracy of the compared algorithms is rather low due to the fact that LCSS + KM, Hausdorff + KM, and SSPD + KM only consider trajectory direction, position, and shape information, respectively. As for the developed method, it takes motion parameters, such as velocity, into account and optimizes the metric matrix, which can improve the clustering accuracy and thereby improve the ability to distinguish different types of trajectories. Moreover, compared to the comparison algorithms, the proposed approach has a higher recall rate for straight trajectories, which indicates that the clustering method with trajectory feature coding and shape information can achieve excellent clustering performance for linear trajectories. Furthermore, compared with straight trajectories, the clustering accuracy rate acquired via each algorithm in the cases of the left- and right-turn trajectories is poor because the left and right-turn trajectories are rather similar. According to the description above, it is known that the proposed algorithm has higher accuracy and recall rates compared to the comparison algorithms in the scenarios of straight and left-turn, as well as right-turn, trajectories.

Table 1. Comparison of clustering performance for real data sets.

Clustering Method	Straight	Left Turn	Right Turn	Accuracy
LCSS + KM	0.70/0.48	0.40/0.53	0.45/0.56	51.67%
Hausdorff + KM	0.65/0.65	0.45/0.39	0.60/0.71	56.67%
SSPD + KM	0.85/0.68	0.65/0.59	0.45/0.69	65.00%
The proposed algorithm	0.90/0.95	0.57/0.60	0.55/0.55	71.67%

5. Conclusions

A trajectory clustering approach with the bag-of-words model and metric learning was developed in this paper based on trajectory feature encoding and metric learning theory. The trajectory was segmented via the proposed method into several optimal homogeneous trajectory segments under the MDL principle, and then the statistic features of each segment were calculated to obtain the feature set of trajectory segments. Subsequently, a metric learning method was proposed under manifold and low-rank constraints to improve the similarity metric performance. Additionally, the segmented trajectory was encoded into a fixed-length vector by using the bag-of-words model and the proposed metric learning method to acquire its feature descriptor. Finally, the segmented trajectories were clustered with K-means and the proposed metric learning method to achieve effective trajectory clustering. The experimental results show that the proposed algorithm can cluster trajectories more preferably than the LCSS, SSPD, and Hausdorff distance metric-based methods in different regions, especially in the case of linear trajectory, and distinguish different types of trajectories more accurately. It is important to note that comparative experiments indicate a lower clustering accuracy rate for the proposed algorithm in scenarios involving left–right rotation trajectories, owing to the high degree of similarity between such trajectories. Furthermore, the presence of noise and data loss in trajectory datasets results in the clustering accuracy rate of the proposed algorithm reaching only 71%. Deep learning, due to its superior feature extraction capabilities, can effectively address these complex scenarios. Consequently, future research will integrate non-convex low-rank metric learning with deep learning to construct a non-convex low-rank deep metric learning model, thereby enhancing trajectory clustering accuracy in challenging environments.

Author Contributions: Investigation, formal analysis, methodology, software, writing, validation, and visualization—X.L.; Conceptualization, funding acquisition, resources, and supervision—H.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China under Grant 61301258, the Key projects of the Natural Science Foundation of Zhejiang Province under Grant LZ21F010002, the China Postdoctoral Science Foundation under Grant 2016M590218, and the Foundation of State Key Laboratory of Complex Electromagnetic Environment Effects on Electronics and Information System under Grant CEMEE2023K0301.

Data Availability Statement: The raw data supporting the conclusions of this article are available upon request from the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Xie, J.; Liu, X.; Wang, M. SFKNN-DPC: Standard deviation weighted distance based density peak clustering algorithm. *Inf. Sci.* **2024**, *653*, 119788. [CrossRef]
2. Jiang, J.; Pan, D.; Ren, H.; Jiang, X.; Li, C.; Wang, J. Self-supervised trajectory representation learning with temporal regularities and travel semantics. In Proceedings of the 2023 IEEE 39th International Conference on Data Engineering (ICDE), Anaheim, CA, USA, 3–7 April 2023; pp. 843–855.

3. Yang, Y.; Cai, J.; Yang, H.; Zhang, J.; Zhao, X. TAD: A trajectory clustering algorithm based on spatial-temporal density analysis. *Expert Syst. Appl.* **2020**, *139*, 112846. [CrossRef]
4. Yang, J.; Liu, Y.; Ma, L.; Ji, C. Maritime traffic flow clustering analysis by density based trajectory clustering with noise. *Ocean Eng.* **2022**, *249*, 111001. [CrossRef]
5. Yu, Q.; Luo, Y.; Chen, C.; Chen, S. Trajectory similarity clustering based on multi-feature distance measurement. *Appl. Intell.* **2019**, *49*, 2315–2338. [CrossRef]
6. Sousa, R.S.D.; Boukerche, A.; Loureiro, A.A. Vehicle trajectory similarity: Models, methods, and applications. In *ACM Computing Surveys (CSUR)*; Association for Computing Machinery: New York, NY, USA, 2020; pp. 1–32.
7. Niu, X.; Chen, T.; Wu, C.Q.; Niu, J.; Li, Y. Label-based trajectory clustering in complex road networks. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 4098–4110. [CrossRef]
8. Besse, P.C.; Guillouet, B.; Loubes, J.-M.; Royer, F. Review and perspective for distance-based clustering of vehicle trajectories. *IEEE Trans. Intell. Transp. Syst.* **2016**, *17*, 3306–3317. [CrossRef]
9. Kumar, D.; Wu, H.; Rajasegarar, S.; Leckie, C.; Krishnaswamy, S.; Palaniswami, M. Fast and scalable big data trajectory clustering for understanding urban mobility. *IEEE Trans. Intell. Transp. Syst.* **2018**, *19*, 3709–3722. [CrossRef]
10. Ma, D.; Fang, B.; Ma, W.; Wu, X.; Jin, S. Potential routes extraction for urban customized bus based on vehicle trajectory clustering. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 11878–11888. [CrossRef]
11. Tao, Y.; Both, A.; Silveira, R.I.; Buchin, K.; Sijben, S.; Purves, R.S.; Laube, P.; Peng, D.; Toohey, K.; Duckham, M. A comparative analysis of trajectory similarity measures. *GISci. Remote Sens.* **2021**, *58*, 643–669. [CrossRef]
12. Wang, S.; Bao, Z.; Culpepper, J.S.; Cong, G. A survey on trajectory data management, analytics, and learning. *ACM Comput. Surv.* **2021**, *54*, 39. [CrossRef]
13. Yao, D.; Zhang, C.; Zhu, Z.; Huang, J.; Bi, J. Trajectory clustering via deep representation learning. In Proceedings of the 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, USA, 14–19 May 2017; pp. 3880–3887.
14. Zhong, C.; Cheng, S.; Kasoar, M.; Arcucci, R. Reduced-order digital twin and latent data assimilation for global wildfire prediction. *Nat. Hazards Earth Syst. Sci.* **2023**, *23*, 1755–1768. [CrossRef]
15. Taghizadeh, S.; Elekes, A.; Schäler, M.; Böhm, K. How meaningful are similarities in deep trajectory representations? *Inf. Syst.* **2021**, *98*, 101452. [CrossRef]
16. Liang, M.; Liu, R.W.; Li, S.; Xiao, Z.; Liu, X.; Lu, F. An unsupervised learning method with convolutional auto-encoder for vessel trajectory similarity computation. *Ocean Eng.* **2021**, *225*, 108803. [CrossRef]
17. Michelioudakis, E.; Artikis, A.; Paliouras, G. Online semi-supervised learning of composite event rules by combining structure and mass-based predicate similarity. *Mach. Learn.* **2024**, *113*, 1445–1481. [CrossRef]
18. Sun, P.; Yang, L. Low-rank supervised and semi-supervised multi-metric learning for classification. *Knowl. Based Syst.* **2022**, *236*, 107787. [CrossRef]
19. Macqueen, J. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*; University of California Press: Berkeley, CA, USA, 1967; Volume 233, pp. 281–297.
20. Zhang, T.; Ramakrishnan, R.; Livny, M. BIRCH: An Efficient Data Clustering Databases Method for Very Large Databases. In Proceedings of the ACM SIGMOD International Conference on Management of Data, Montreal, QC, Canada, 4–6 June 1996; pp. 103–114.
21. Wang, W.; Yang, J.; Muntz, R. STING: A Statistical Information Grid Approach to Spatial Data Mining. In Proceedings of the 23rd International Conference on Very Large Data Bases, Athens, Greece, 25–29 August 1997; pp. 186–195.
22. Sheikholeslami, G.; Chatterjee, S.; Zhang, A. WaveCluster: A wavelet-based clustering approach for spatial data in very large databases. *VLDB J.* **2000**, *8*, 289–304. [CrossRef]
23. Fisher, D. Knowledge acquisition via incremental clustering. *Mach. Learn.* **1987**, *2*, 139–182. [CrossRef]
24. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, Portland, OR, USA, 2–4 August 1996; pp. 226–231.
25. Ankerst, M.; Breunig, M.M.; Kriegel, H.; Sander, J. OPTICS: Ordering Points To Identify the Clustering Structure. In Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data, Philadelphia, PA, USA, 1–3 June 1999; Volume 28, pp. 49–60.
26. Ma, S.; Wang, T.; Tang, S.; Yang, D.; Gao, J. A New Fast Clustering Algorithm Based on Reference and Density. In *Advances in Web-Age Information Management*; Springer: Berlin/Heidelberg, Germany, 2002; pp. 214–225.
27. Agrawal, K.P.; Garg, S.; Sharma, S.; Patel, P. Development and validation of OPTICS based spatio-temporal clustering technique. *Inf. Sci.* **2016**, *369*, 388–401. [CrossRef]
28. Hüsch, M.; Schyska, B.U.; Bremen, L.V. CorClustST-Correlation-based clustering of big spatio-temporal datasets. *Future Gener. Comput. Syst.* **2020**, *110*, 610–619. [CrossRef]

29. Huang, L.; Xu, Z.; Zhang, Z.; He, Y.; Zan, M. A fast iterative shrinkage/thresholding algorithm via laplace norm for sound source identification. *IEEE Access* **2020**, *8*, 115335–115344. [CrossRef]
30. Greenacre, M.; Groenen, P.; Hastie, T.; d'Enza, A.I.; Markos, A.; Tuzhilina, E. Principal component analysis. *Nat. Rev. Methods Primers* **2022**, *2*, 1–100. [CrossRef]
31. Yuan, C.; Yang, L. An efficient multi-metric learning method by partitioning the metric space. *Neurocomputing* **2023**, *529*, 56–79. [CrossRef]
32. Lei, C.; Zhu, X. Unsupervised feature selection via local structure learning and sparse learning. *Multimedia Tools Appl.* **2018**, *77*, 29605–29622. [CrossRef]
33. Islam, A.; Radke, R. Weakly supervised temporal action localization using deep metric learning. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 3–8 January 2020; pp. 547–556.
34. Sun, G.; Cong, Y.; Wang, Q.; Xu, X. Online low-rank metric learning via parallel coordinate descent method. In Proceedings of the 2018 24th International Conference on Pattern Recognition (ICPR), Beijing, China, 20–24 August 2018; pp. 207–212.
35. Chen, S.; Shen, Y.; Yan, Y.; Wang, D.; Zhu, S. Cholesky Decomposition-Based Metric Learning for Video-Based Human Action Recognition. *IEEE Access* **2020**, *8*, 36313–36321. [CrossRef]
36. Ma, X.; Li, G.; Wang, Y.; Li, H.; Yang, W. Seismic Deconvolution Based on a Non-Convex L1-L2 Norm Constraint. In Proceedings of the 81st EAGE Conference and Exhibition, London, UK, 3–6 June 2019; pp. 1–5.
37. Tono, K.; Takeda, A.; Gotoh, J. Efficient DC algorithm for constrained sparse optimization. *arXiv* **2017**, arXiv:1701.08498.
38. Wang, J.; Zhang, F.; Huang, J.; Wang, W.; Yuan, C. A nonconvex penalty function with integral convolution approximation for compressed sensing. *Signal Process.* **2019**, *158*, 116–128. [CrossRef]
39. Junior, A.S.; Times, V.C.; Renso, C.; Matwin, S.; Cabral, L.A. A semi-supervised approach for the semantic segmentation of trajectories. In Proceedings of the 2018 19th IEEE International Conference on Mobile Data Management (MDM), Aalborg, Denmark, 25–28 June 2018; pp. 145–154.
40. Etemad, M.; Júnior, A.S.; Hoseyni, A.; Rose, J.; Matwin, S. A Trajectory Segmentation Algorithm Based on Interpolation-based Change Detection Strategies. In Proceedings of the EDBT/ICDT Workshops, Lisbon, Portugal, 26 March 2019; pp. 51–58.
41. Nivash, S.; Ganesh, E.N.; Harisudha, K.; Sreeram, S. Extensive analysis of global presidents' speeches using natural language. In *Sentimental Analysis and Deep Learning: Proceedings of ICSADL*; Springer: Singapore, 2022; pp. 829–850.
42. Mateos-Nunez, D.; Cortes, J. Distributed saddle-point subgradient algorithms with Laplacian averaging. *IEEE Trans. Autom. Control* **2016**, *62*, 2720–2735. [CrossRef]
43. Liang, M.; Liu, R.W.; Gao, R.; Xiao, Z.; Zhang, X.; Wang, H. A survey of distance-based vessel trajectory clustering: Data pre-processing, methodologies, applications, and experimental evaluation. *arXiv* **2024**, arXiv:2407.11084.
44. Feng, X.; Cen, Z.; Hu, J.; Zhang, Y. Vehicle trajectory prediction using intention-based conditional variational autoencoder. In Proceedings of the 2019 IEEE Intelligent Transportation Systems Conference (ITSC), Auckland, New Zealand, 27–30 October 2019; pp. 3514–3519.
45. Zhang, H.; Fu, R. A Hybrid Approach for Turning Intention Prediction Based on Time Series Forecasting and Deep Learning. *Sensors* **2020**, *20*, 4887. [CrossRef]
46. Ghazal, T.M.; Hussain, M.Z.; Said, R.A.; Nadeem, A.; Hasan, M.K.; Ahmad, M.; Khan, M.A.; Naseem, M.T. Performances of k-means clustering algorithm with different distance metrics. *Intell. Autom. Soft Comput.* **2021**, *29*, 735–742. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.