Hindawi Publishing Corporation
Journal of Sensors
Volume 2014, Article ID 824904, 12 pages
http://dx.doi.org/10.1155/2014/824904

Research Article

Nonnegative Matrix Factorization-Based Spatial-Temporal Clustering for Multiple Sensor Data Streams

Di-Hua Sun[1,2] **and Chun-Yan Sang**[1,2]

[1] *Key Laboratory of Dependable Service Computing in Cyber Physical Society of Ministry of Education, Chongqing University, Chongqing 400030, China*
[2] *College of Automation, Chongqing University, Chongqing 400030, China*

Correspondence should be addressed to Chun-Yan Sang; yysang2007@126.com

Received 10 March 2014; Revised 30 June 2014; Accepted 2 July 2014; Published 17 July 2014

Academic Editor: Athanasios V. Vasilakos

Cyber physical systems have grown exponentially and have been attracting a lot of attention over the last few years. To retrieve and mine the useful information from massive amounts of sensor data streams with spatial, temporal, and other multidimensional information has become an active research area. Moreover, recent research has shown that clusters of streams change with a comprehensive spatial-temporal viewpoint in real applications. In this paper, we propose a spatial-temporal clustering algorithm (STClu) based on nonnegative matrix trifactorization by utilizing time-series observational data streams and geospatial relationship for clustering multiple sensor data streams. Instead of directly clustering multiple data streams periodically, STClu incorporates the spatial relationship between two sensors in proximity and integrates the historical information into consideration. Furthermore, we develop an iterative updating optimization algorithm STClu. The effectiveness and efficiency of the algorithm STClu are both demonstrated in experiments on real and synthetic data sets. The results show that the proposed STClu algorithm outperforms existing methods for clustering sensor data streams.

1. Introduction

Cyber physical systems (CPS) have grown exponentially and have been attracting a lot of attention over the last few years [1]. Emerging applications include transportation, energy, manufacturing, environment monitoring, and many parts of our social infrastructure. It consists of a large number of sensors to monitor physical or environmental conditions [2, 3]. The incoming information in CPS needs to be considered as a stream and not so much as an ever-growing data set [4, 5].

To obtain interesting relationships and useful information from multiple data streams, a variety of methods have developed over the past decades. Beringer and Hüllermeier [6] used a discrete Fourier transformation (DFT) to summarize the data streams and presented an online version of the classic K-means clustering algorithm to cluster parallel data streams. To avoid recalculating the DFT coefficients and minimize processing time, an incremental update mechanism for clustering large number of data streams was proposed in [7]. To discover cross relationships among multiple data streams, a clustering-on-demand framework was presented in [8]. Yeh et al. [9] proposed a framework for clustering over multiple evolving streams by correlations and events, which monitored the distribution of clusters over multiple data streams based on their correlation. To find communities in a large set of interacting entities, Aggarwal and Yu [10] proposed an online analytical processing framework for community detection of data streams. Evolutionary clustering was a relatively new topic and was formulated by Chakrabarti et al. [11] in 2006. To discover communities and capture their evolution with temporal smoothness, incremental and evolutionary clustering technologies were designed to handle dynamic data streams [10, 12–14]. Wang et al. [15] proposed an ECKF framework for evolutionary clustering large-scale data based on low-rank kernel matrix factorization. Chi et al. [16] proposed two algorithms to incorporate temporal smoothness in evolutionary spectral clustering. Ning et al. [14] presented an incremental approach for spectral clustering to

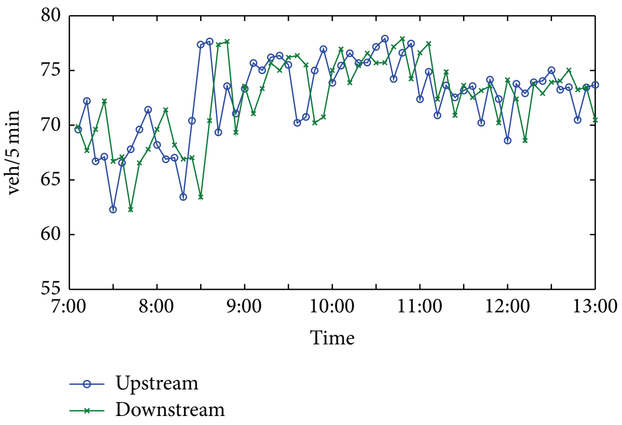


Figure 1: Time-series of upstream and downstream traffic flow.

handle not only insertion/deletion of data points but also similarity changes between existing points.

The values of a sensor in the temporal domain usually have spatial correlation features, meaning that the sensors' readings are influenced by near sensors. Such as in transportation systems, the congestion swiftly expands along the street and influences nearby sensors. A serious congestion usually lasts for a few hours and covers hundreds of sensors when reaching the full size. As time passes by, it shrinks slowly, eventually reduces the coverage, and finally disappears. As shown in Figure 1, we find that those records in an atypical event are spatially close and timely relevant. To retrieve the useful information from massive sensor data streams, Tang et al. [5] proposed a model of atypical cluster to describe the atypical events and retrieve them with spatial and temporal attribute. To explore the spatial correlation between data sampled by different sensors for sensor networks, α-local spatial clustering algorithm was proposed [17]. To explore the temporal locality features of sensor data, Wei and Peng [18] proposed an incremental clustering algorithm that clusters objects by inducing clustering results from previous time windows.

In this paper, we propose a spatial-temporal clustering algorithm (STClu) based on nonnegative matrix trifactorization by utilizing time-series observational data streams and geospatial relationship for clustering multiple sensor data streams. Our proposal was motivated by recent progress in nonnegative matrix factorization and its applications [19–25], with the aim being to discover clusters of objects with similar behavior and, consequently, discover potential anomalies and atypical events which may be revealed by the relationship evolving over time. We assume that the spatial feature is the summary of the atypical event in temporal dimension, and the temporal feature is the summary of the event in spatial dimension. STClu simultaneously incorporates the spatial correlation between two sensors in proximity and integrates the historical information into consideration. To maintain a consistent clustering result, there is a trade-off between the spatially close and timely relevant cost embedded in STClu and the benefit from current observations and the spatially close historical results. Thus, the clustering results at a given time step are determined by current snapshot, prior knowledge, and spatial correlation between two sensors in proximity.

The rest of this paper is organized as follows. Related research work is introduced in Section 2. Then, the preliminaries of STClu are given in Section 3. Section 4 presents the proposed algorithm in detail, while convergence and correctness proofs of the STClu are provided in the Appendix. Experimental results using synthetic and real world data sets are presented in Section 5. Finally, Section 6 gives the conclusion.

2. A Brief Review of Matrix Factorization and Its Applications

Graphs are used in a wide range of applications, such as transportation networks, sensor networks, and social networks. So, various problems are studied under graph mining. To find clusters in graphs is a new challenge if the graph is evolving over time. Matrix is a usual representation of a graph; the relationships between matrix factorization and K-means clustering have been explored in [26].

Singular value decomposition (SVD) has served as a building block for many important applications, such as PCA and LSI [27]. SVD factorizes a matrix with the general form of $A \approx SVD^T$, where S is a unitary basis consisting of left-singular vectors of A, D is a unitary basis consisting of right-singular vectors of A, and V is a diagonal matrix with singular values on the diagonal. In SVD, matrices S and D are allowed to have negative entries; the projected data might have negative values in spite of the original data being positive. That can prevent from the clustering results to be intuitive for applications such as documents or images that have a positive data input.

Nonnegative matrix factorization (NMF) [28, 29] is a linear, nonnegative approximate data representation technique. NMF focuses on the analysis of data matrices whose elements are nonnegative, a common occurrence in data sets derived from text and images. The nonnegative data matrix A is factorized into matrixes F and G as $A \approx FG^T$, with the constraints that $F \in \mathbb{R}^{d\times k}$ and $G \in \mathbb{R}^{n\times k}$ are nonnegative.

In particularly, the result of a K-means clustering run can be written as a matrix factorization $A = FG^T$, where A is a data matrix, F contains the cluster centroids, and G contains the cluster membership indicators. Nonnegative matrix factorization focuses on the analysis of data matrices whose elements are nonnegative [28].

To extend the applicable range of NMF methods, when the data matrix is unconstrained, Semi-NMF is motivated from the perspective of clustering [29], in which it restricts G to be nonnegative while placing no restriction on the signs of F. For reasons of interpretability, Convex-NMF constrains the basis vectors $F = (\mathbf{f}_1, \ldots, \mathbf{f}_k)$. The vectors defining F lie within the column space of A: $f_l = w_{1l}a_1 + \cdots + w_{nl}a_n = Aw_l$, or $F = AW$. It can be applied to both nonnegative and mixed-sign data matrices. This constraint could interpret the columns f_l as weighted sums of certain data points and these columns would capture a notion of centroids.

To expand the range of application of NMF, Semi-NMF and Convex-NMF algorithm have been proposed in [29]. Semi-NMF offers a low-dimensional representation of data points which lends itself to a convenient clustering interpretation. Convex-NMF restricts the columns of F to be convex combinations of data points in X. The advantage of this constraint is that the columns as weighted sums of certain data points could interpret. Based on NMF, a clustering algorithm has been proposed in [29].

However, the traditional NMF, Semi-NMF, and Convex-NMF are linear model and they may fail to discover the nonlinearities of data streams. In real world, the data streams have potential nonlinear structure. Kernel method is a powerful technique for extracting useful information form nonlinear correlations. To solve the problem of nonlinearities, the kernel method is to map the data nonlinearly into a kernel feature space. Then, the Convex-NMF method can be accomplished in the kernel feature space to process the nonlinear data.

Indeed, several important applications can be modeled as large sparse graphs, such as transportation network analysis and social network analysis. Low-rank approximation for the matrix of a graph is essential in finding patterns and detecting anomalies. And it can extract correlations and remove noise from matrix structured data. This has led to proposal of methods such as CUR [30], CMD [31], and the family of Colibri [32].

3. Preliminaries

To monitor the traffic status along a freeway, sensors are deployed and utilized to collect readings, such as vehicle speed and volume of traffic. It is assumed that data records arrive synchronously, which means that all data streams will be updated simultaneously. The sensors are the points in the graph and an edge is formed between each pair of sensors; these data records can be used to construct graphs of transportation networks periodically.

Given two sets of spatial objects $O^{(t-1)} = \{o_1^{(t-1)}, \dots, o_m^{(t-1)}\}$ and $O^{(t)} = \{o_1^{(t)}, \dots, o_n^{(t)}\}$, the spatial relationships are depicted as Figure 2. Let $G^{(t-1)} = (S^{(t)}, E^{(t)})$ be a graph associated with the sensor network of transportation system at timestamp t, where $s_i^{(t)} \in S^{(t)}$ represents the set of sensors associated with graph $G^{(t)}$ and $e_{ij}^{(t)} \in E^{(t)}$ represents the set of edges between $s_i^{(t)}$ and $s_j^{(t)}$. And similar definitions are depicted for $G^{(t-1)} = (S^{(t-1)}, E^{(t-1)})$ associated with the sensor network of transportation system at timestamp $t-1$.

Without loss of generality, we use the adjacency matrices $X^{(t-1)} \in \mathbb{R}^{m\times m}$ and $X^{(t)} \in \mathbb{R}^{n\times n}$ to represent two graphs $G^{(t-1)} = (S^{(t-1)}, E^{(t-1)})$ and $G^{(t)} = (S^{(t)}, E^{(t)})$, respectively. $X^{(i)}(i, j)$ is the element at the ith row and jth column of the matrix $X^{(i)}$; $X^{(i)}(:, j)$ is the jth column of $X^{(i)}$. Every row or column in $X^{(i)}$ corresponds to a node in $S^{(i)}$. If there is an edge from node $s_i^{(i)} \in S^{(i)}$ to node $s_j^{(i)} \in S^{(i)}$ with similarity $\rho_{ij}^{(i)}$, we set the value to $X^{(i)}(i, j)$. Otherwise, we set it to zero. The similarity matrix is computed as follows:

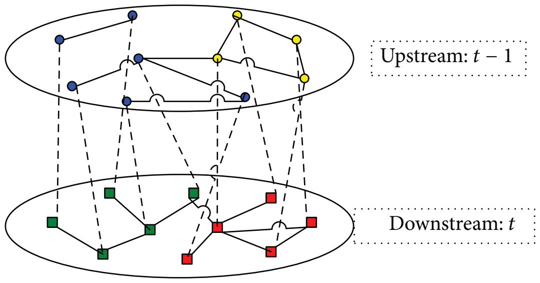


Figure 2: Spatial-temporal cluster: graph $G^{(t-1)} = (S^{(t-1)}, E^{(t-1)})$, graph $G^{(t)} = (S^{(t)}, E^{(t)})$ and the spatial relationship between timestamps $t-1$ and t.

$$\rho_{ij}^{(t)} = \exp\left(-\frac{\left\| s_i^{(t)} - s_j^{(t)} \right\|^2}{2\sigma^2}\right), \tag{1}$$

where σ is a scaling parameter controlling how rapidly similarity $\rho_{ij}^{(t)}$ reduces with the distance between $s_i^{(t)}$ and $s_j^{(t)}$.

The spatial relationships between two graphs are denoted as a set of edges connecting the nodes between $G^{(t)}$ and $G^{(t-1)}$. We use the adjacency matrix $X^{(t-1,t)} \in \mathbb{R}^{n\times m}$ to represent the spatial relationships. Given two objects $s_i^{(t-1)} \in S^{(t-1)}$ and $s_j^{(t)} \in S^{(t)}$, the locations of $s_i^{(t-1)}$ and $s_j^{(t)}$ are denoted as $L \cdot s_i^{(t-1)}$ and $L \cdot s_j^{(t)}$, respectively. The spatial relationship between $L \cdot s_i^{(t-1)}$ and $L \cdot s_j^{(t)}$ is defined as follows:

$$\begin{aligned} D\left(L \cdot s_i^{(t)}, L \cdot s_j^{(t-1)}\right) = \Big(& \left(L \cdot s_{i1}^{(t)} - L \cdot s_{j1}^{(t-1)}\right)^2 \\ & + \left(L \cdot s_{i2}^{(t)} - L \cdot s_{j2}^{(t-1)}\right)^2 \\ & + \cdots + \left(L \cdot s_{in}^{(t)} - L \cdot s_{jn}^{(t-1)}\right)^2 \Big)^{1/2}. \end{aligned} \tag{2}$$

At timestamp t, the objective of STClu is to create sets of partitions $\{\Delta_i^{(t)}\}_{i=1}^k$. $\forall i \in \{1, \dots, k\}$, $\Delta_i^{(t)}$ should meet the following conditions: $\bigcup_{i=1}^k \Delta_i^{(t)} = \{s_1^{(t)}, s_2^{(t)}, \dots, s_n^{(t)}\}$ and $\bigcap_{i=1}^k \Delta_i^{(t)} = \varnothing$. The community $\Delta_i^{(t)}$ is a set of similar data streams such that $\Delta_i^{(t)} = \{s_1^{(t)}, \dots, s_{|\Delta_i^{(t)}|}^{(t)}\}$, where $|\Delta_i^{(t)}|$ is the total number of streams in $\Delta_i^{(t)}$.

4. Our Approach

4.1. Objective Function of STClu. By utilizing time-series observational data streams and geospatial relationship for clustering multiple sensor data streams, the proposed method STClu takes the spatial correlation between two sensors in proximity and the historical information into consideration. The cost of spatially close and timely relevant is embedded in the objective function. The cluster partitions are obtained by decomposing the adjacency matrix representation of the graph into a product of its latent factors. The square of the Euclidean distance and the generalized Kullback-Leibler divergence are common metrics for the approximation error [24]. In STClu, we seek to minimize the distance function $\|A-B\|_F^2$ between two nonnegative matrices A and B, defined as

$$\|A-B\|_F^2 = \sum_{ij}\left(A_{ij}-B_{ij}\right)^2, \tag{3}$$

where $\|\cdot\|_F^2$ is the Frobenius norm. This is lower bounded by zero and clearly vanishes if, and only if, $A = B$.

The objective function can be written as

$$\begin{aligned} J_{\text{STClu}} = {} & \left\| X^{(t)} - C^{(t)}U^{(t)}R^{(t)^T} \right\|_F^2 \\ & + \left\| X^{(t-1)} - C^{(t-1)}U^{(t-1)}R^{(t)^T} \right\|_F^2 \\ & + \left\| X^{(t-1,t)} - R^{(t)}U^{(t-1,t)}R^{(t-1)^T} \right\|_F^2, \end{aligned} \tag{4}$$

where $C^{(t)}$ and $C^{(t-1)}$ are the representative column matrices of $X^{(t)}$ and $X^{(t-1)}$, respectively, $U^{(t)}$ and $U^{(t-1)}$ are the weight matrices, respectively, $R^{(t)} \in \mathbb{R}^{n\times k_1}$ and $R^{(t-1)} \in \mathbb{R}^{m\times k_2}$ are the corresponding cluster membership matrices for graphs $G^{(t)}$ and $G^{(t-1)}$, respectively, and $U^{(t-1,t)}$ reflects the correspondence between the subgroups derived from two time steps.

Specifically, assume cluster numbers k_1 and k_2 at time steps t and $t-1$, respectively. If $k_2 < k_1$, then the extra clusters at time t must be computed using only the current data and we let $U^{(t-1)} = [U^{(t-1)}, 0_{c\times(k_2+1:k_2)}]$. Otherwise, $k_2 > k_1$ and we simply remove the deleted cluster weight vectors in $U^{(t-1)}$.

4.2. Optimization of STClu. As can be seen, the objective function in (4) is minimized with respect to $C^{(t)}$, $U^{(t)}$, $R^{(t)}$, and $U^{(t-1,t)}$, and it is unrealistic to expect an algorithm to find the global minimum. In the following, we introduce an alternating schema to optimize the objective function, which can achieve a local minimum.

We optimize the objective function with respect to one variable while fixing the other variables. This iterative procedure repeats until convergence. Using the matrix properties $\operatorname{Tr}(AB) = \operatorname{Tr}(BA)$ and $\operatorname{Tr}(A) = \operatorname{Tr}(A^T)$, the objective function J in (4) can be rewritten as

$$\begin{aligned} & J_{\text{STClu}} \\ & = \operatorname{Tr}\left(\left(X^{(t)} - C^{(t)}U^{(t)}R^{(t)^T}\right)\left(X^{(t)} - C^{(t)}U^{(t)}R^{(t)^T}\right)^T\right) \\ & \quad + \operatorname{Tr}\left(\left(X^{(t-1)} - C^{(t-1)}U^{(t-1)}R^{(t)^T}\right) \right. \\ & \qquad \left. \times \left(X^{(t-1)} - C^{(t-1)}U^{(t-1)}R^{(t)^T}\right)^T\right) \\ & \quad + \operatorname{Tr}\left(\left(X^{(t-1,t)} - R^{(t)}U^{(t-1,t)}R^{(t-1)^T}\right) \right. \\ & \qquad \left. \times \left(X^{(t-1,t)} - R^{(t)}U^{(t-1,t)}R^{(t-1)^T}\right)^T\right) \\ & = \operatorname{Tr}\left(X^{(t)}X^{(t)^T}\right) - 2\operatorname{Tr}\left(X^{(t)}R^{(t)}U^{(t)^T}C^{(t)^T}\right) \\ & \quad + \operatorname{Tr}\left(C^{(t)}U^{(t)}R^{(t)^T}R^{(t)}U^{(t)^T}C^{(t)^T}\right) \\ & \quad + \operatorname{Tr}\left(X^{(t-1)}X^{(t-1)^T}\right) - 2\operatorname{Tr}\left(X^{(t-1)}R^{(t)}U^{(t-1)^T}C^{(t-1)^T}\right) \\ & \quad + \operatorname{Tr}\left(C^{(t-1)}U^{(t-1)}R^{(t)^T}R^{(t)}U^{(t-1)^T}C^{(t-1)^T}\right) \\ & \quad + \operatorname{Tr}\left(X^{(t-1,t)}X^{(t-1,t)^T}\right) \\ & \quad - 2\operatorname{Tr}\left(X^{(t-1,t)}R^{(t-1)}U^{(t-1,t)^T}R^{(t)^T}\right) \\ & \quad + \operatorname{Tr}\left(R^{(t)}U^{(t-1,t)}R^{(t-1)^T}R^{(t-1)}U^{(t-1,t)^T}R^{(t)^T}\right). \end{aligned} \tag{5}$$

To alternately update the entries of $C^{(t)}$, $U^{(t)}$, $R^{(t)}$, and $U^{(t-1,t)}$, we resort to a Lagrangian function. Let ζ_{ij}, ξ_{jl}, ψ_{kj}, and ϑ_{ij} be the Lagrange multiplier for constraints $C_{ij}^{(t)} \geq 0$, $U_{jl}^{(t)} \geq 0$, $R_{kj}^{(t)} \geq 0$, and $U_{ij}^{(t-1,t)} \geq 0$, respectively. Then the Lagrange function L is expressed as

$$\begin{aligned} L = {} & \operatorname{Tr}\left(X^{(t)}X^{(t)^T}\right) - 2\operatorname{Tr}\left(X^{(t)}R^{(t)}U^{(t)^T}C^{(t)^T}\right) \\ & + \operatorname{Tr}\left(C^{(t)}U^{(t)}R^{(t)^T}R^{(t)}U^{(t)^T}C^{(t)^T}\right) \\ & + \operatorname{Tr}\left(X^{(t-1)}X^{(t-1)^T}\right) - 2\operatorname{Tr}\left(X^{(t-1)}R^{(t)}U^{(t-1)^T}C^{(t-1)^T}\right) \\ & + \operatorname{Tr}\left(C^{(t-1)}U^{(t-1)}R^{(t)^T}R^{(t)}U^{(t-1)^T}C^{(t-1)^T}\right) \\ & + \operatorname{Tr}\left(X^{(t-1,t)}X^{(t-1,t)^T}\right) \\ & - 2\operatorname{Tr}\left(X^{(t-1,t)}R^{(t-1)}U^{(t-1,t)^T}R^{(t)^T}\right) \\ & + \operatorname{Tr}\left(R^{(t)}U^{(t-1,t)}R^{(t-1)^T}R^{(t-1)}U^{(t-1,t)^T}R^{(t)^T}\right) \\ & + \operatorname{Tr}\left(\zeta C^{(t)^T}\right) + \operatorname{Tr}\left(\xi U^{(t)^T}\right) \\ & + \operatorname{Tr}\left(\psi R^{(t)^T}\right) + \operatorname{Tr}\left(\vartheta U^{(t-1,t)^T}\right). \end{aligned} \tag{6}$$

Since the derivatives of L with respect to $U^{(t)}$, $C^{(t)}$, $R^{(t)}$, and $U^{(t-1,t)}$ must be zero, we have

$$\begin{aligned} \frac{\partial L}{\partial U^{(t)}} &= -2C^{(t)^T}X^{(t)}R^{(t)} + 2C^{(t)^T}C^{(t)}U^{(t)}R^{(t)^T}R^{(t)} + \xi, \\ \frac{\partial L}{\partial C^{(t)}} &= -2X^{(t)}R^{(t)}U^{(t)^T} + 2C^{(t)}U^{(t)}R^{(t)^T}R^{(t)}U^{(t)^T} + \zeta, \end{aligned}$$

$$\begin{aligned}\frac{\partial L}{\partial R^{(t)}} = & -2X^{(t)^T}C^{(t)}U^{(t)} + 2R^{(t)}U^{(t)^T}C^{(t)^T}C^{(t)}U^{(t)} \\ & -2X^{(t-1)^T}C^{(t-1)}U^{(t-1)} \\ & +2R^{(t)}U^{(t-1)^T}C^{(t-1)^T}C^{(t-1)}U^{(t-1)} \\ & -2X^{(t-1,t)}R^{(t-1)}U^{(t-1,t)^T} \\ & +2R^{(t)}U^{(t-1,t)}R^{(t-1)^T}R^{(t-1)}U^{(t-1,t)^T} + \psi, \\ \frac{\partial L}{\partial U^{(t-1,t)}} = & -2R^{(t)^T}X^{(t-1,t)}R^{(t-1)} \\ & +2R^{(t)^T}R^{(t)}U^{(t-1,t)}R^{(t-1)^T}R^{(t-1)} + \vartheta.\end{aligned} \tag{7}$$

Using the Karush-Kuhn-Tucker conditions [29, 33] and letting $\xi_{jl}U_{jl}^{(t)} = 0$, $\zeta_{ij}C_{ij}^{(t)} = 0$, $\psi_{kj}R_{kj}^{(t)} = 0$, and $\vartheta_{ij}U_{ij}^{(t-1,t)} = 0$, we obtain the following equations for $U^{(t)}$, $C^{(t)}$, $R^{(t)}$, and $U^{(t-1,t)}$:

$$\begin{aligned} & \left[-C^{(t)^T}X^{(t)}R^{(t)} + C^{(t)^T}C^{(t)}U^{(t)}R^{(t)^T}R^{(t)}\right]_{jl}U_{jl}^{(t)} = 0, \\ & \left[-X^{(t)}R^{(t)}U^{(t)^T} + C^{(t)}U^{(t)}R^{(t)^T}R^{(t)}U^{(t)^T}\right]_{ij}C_{ij}^{(t)} = 0, \\ & \Big[-X^{(t)^T}C^{(t)}U^{(t)} + R^{(t)}U^{(t)^T}C^{(t)^T}C^{(t)}U^{(t)} \\ & \quad - X^{(t-1)^T}C^{(t-1)}U^{(t-1)} \\ & \quad + R^{(t)}U^{(t-1)^T}C^{(t-1)^T}C^{(t-1)}U^{(t-1)} \\ & \quad - X^{(t-1,t)}R^{(t-1)}U^{(t-1,t)^T} \\ & \quad + R^{(t)}U^{(t-1,t)}R^{(t-1)^T}R^{(t-1)}U^{(t-1,t)^T}\Big]_{kj}R_{kj}^{(t)} = 0, \\ & \Big[-R^{(t)^T}X^{(t-1,t)}R^{(t-1)} \\ & \quad + R^{(t)^T}R^{(t)}U^{(t-1,t)}R^{(t-1)^T}R^{(t-1)}\Big]_{ij}U_{ij}^{(t-1,t)} = 0.\end{aligned} \tag{8}$$

According to (8), we present the following updating rules:

$$U_{jl}^{(t)} \longleftarrow U_{jl}^{(t)} \frac{\left[C^{(t)^T}X^{(t)}R^{(t)}\right]_{jl}}{\left[C^{(t)^T}C^{(t)}U^{(t)}R^{(t)^T}R^{(t)}\right]_{jl}}, \tag{9}$$

$$C_{ij}^{(t)} \longleftarrow C_{ij}^{(t)} \frac{\left[X^{(t)}R^{(t)}U^{(t)^T}\right]_{ij}}{\left[C^{(t)}U^{(t)}R^{(t)^T}R^{(t)}U^{(t)^T}\right]_{ij}}, \tag{10}$$

$$\begin{aligned}R_{kj}^{(t)} \longleftarrow R_{kj}^{(t)} \Bigg(\Bigg(& \Big[X^{(t)^T}C^{(t)}U^{(t)} + X^{(t-1)^T}C^{(t-1)}U^{(t-1)} \\ & + X^{(t-1,t)}R^{(t-1)}U^{(t-1,t)^T}\Big]_{kj}\Bigg) \\ & \times \Bigg(\Big[R^{(t)}U^{(t)^T}C^{(t)^T}C^{(t)}U^{(t)} \\ & + R^{(t)}U^{(t-1)^T}C^{(t-1)^T}C^{(t-1)}U^{(t-1)} \\ & + R^{(t)}U^{(t-1,t)}R^{(t-1)^T} \\ & \times R^{(t-1)}U^{(t-1,t)^T}\Big]_{kj}\Bigg)^{-1}\Bigg),\end{aligned} \tag{11}$$

$$U_{ij}^{(t-1,t)} \longleftarrow U_{ij}^{(t-1,t)} \frac{\left[R^{(t)^T}X^{(t-1,t)}R^{(t-1)}\right]_{ij}}{\left[R^{(t)^T}R^{(t)}U^{(t-1,t)}R^{(t-1)^T}R^{(t-1)}\right]_{ij}}. \tag{12}$$

Proofs of the convergence of update rules (9), (10), (11), and (12) are given in the Appendix.

4.3. Algorithm Description. To eliminate linearly dependent columns and construct the subspace used for low-rank approximation from data matrix X, a unique and independent subspace C is constructed using Colibri-S and Colibri-D [32]. Algorithm 1 summarizes the details of the iterative updating algorithm for (4). It is assumed that the adjacency matrices $X^{(t)}$, $X^{(t-1)}$, $X^{(t-1,t)}$, $R^{(t-1)}$, and $U^{(t-1)}$ are given. To improve the clustering efficiency, the variable factors are initialized by the previous clustering results instead of random values [29]. Applying the update rules in (9), (10), (11), and (12) until convergence, we obtain a new $C^{(t)}$, $U^{(t)}$, $R^{(t)}$, and $U^{(t-t,t)}$.

The computational cost of the proposed STClu algorithm is discussed as follows. To compute the complexity of the STClu algorithm, we need to inspect its main operations, namely, the decomposition matrix and multiplicative updates. In the decomposition step, the computation is actually done using the three small matrices, U, C, and R. Each of U, C, and R is multiplied by an update factor, which involves several matrix multiplications. We assume that k is the cluster number at time step t, c_1 and c_0 are the sizes of the data subspaces at time steps t and $t-1$, respectively, and n_1 and n_0 are the sizes of the input matrices at time steps t and $t-1$, respectively; the multiplicative update stops after T iterations, and the cost for STClu is $O(T(c^2k + cnk + cn^2 + c^2n))$, where $n = \max(n_0, n_1)$ and $c = \max(c_0, c_1)$.

5. Experiments and Results

In this section, we use several synthetic and real world data sets to evaluate the effectiveness and efficiency of the STClu algorithm.

5.1. Baseline Algorithms and Evaluation Methods. To demonstrate the efficiency of STClu, we compare it with the

```
Input: Matrices $X^{(t)}$, $X^{(t-1)}$, $X^{(t-1,t)}$, $R^{(t-1)}$, and $U^{(t-1)}$, cluster number $k^{(t)}$, and maximum iteration $T$
Output: Matrices $C^{(t)}$, $U^{(t)}$, $R^{(t)}$, and $U^{(t-1,t)}$
(1) use the family of Colibri methods to get $C^{(t)}$
(2) determine cluster number $k^{(t)}$
(3) initialize $U^{(t)}$, $R^{(t)}$, and $U^{(t-1,t)}$ using $R^{(t-1)}$, $U^{(t-1)}$, and $U^{(t-1,t)}$, respectively
(4) if we need to form a new partition
(5)     go to (8)
(6) else
(7)     let $R^{(t)} = R^{(t-1)}$ and $U^{(t)} = U^{(t-1)}$, and return
(8) while not converging and $t \leq T$ do
(9)     update $U^{(t)}$ by (9)
(10)    update $C^{(t)}$ by (10)
(11)    update $R^{(t)}$ by (11)
(12)    update $U^{(t-1,t)}$ by (12)
(13) end while
(14) return $U^{(t)}$, $C^{(t)}$, $R^{(t)}$, and $U^{(t-1,t)}$
```

Algorithm 1: STClu.

following three popular clustering algorithms. The first is K-means clustering in the principle component analysis (PCA) subspace. The other two algorithms are the typical spectral clustering algorithm normalized cut (Ncut) [34] and NMF-based clustering [19].

To evaluate clustering quality, all our comparisons are based on clustering accuracy (ACC) and normalized mutual information (NMI) measurements.

Clustering accuracy (ACC) discovers the one-to-one relationships between clusters and classes and measures the extent to which each cluster contains data points from the corresponding class. It is defined as [18, 19]

$$\mathrm{ACC} = \frac{\sum_{i=1}^{n} \delta\left(\mathrm{map}\left(r_i\right), l_i\right)}{n}, \quad (13)$$

where r_i denotes the cluster label, l_i denotes the true class label, n is the total number of documents, $\delta(x, y)$ is the delta function that is equal to one if $x = y$ and zero otherwise, and $\mathrm{map}(r_i)$ is the permutation mapping function that maps each cluster label r_i to the equivalent label from the data set.

Between two random variables i (category label) and j (cluster label), NMI is defined as [35]

$$\mathrm{NMI} = \frac{\sum_{i=1}^{k^{(a)}} \sum_{j=1}^{k^{(b)}} n_{i,j} \log\left(\left(n \cdot n_{i,j}\right) / \left(n_i \cdot n_j\right)\right)}{\sqrt{\left(\sum_{i=1}^{k^{(a)}} n_i \log\left(n_i/n\right)\right)\left(\sum_{j=1}^{k^{(b)}} n_j \log\left(n_j/n\right)\right)}}, \quad (14)$$

where n is the number of documents, n_i and n_j denote the number of documents in category i and cluster j, respectively, $n_{i,j}$ denotes the number of documents in category i as well as in cluster j, and $k^{(a)}$ and $k^{(b)}$ are the cluster numbers in the true and predicted clusters. The NMI score is 1 if the clustering results perfectly match the category labels whereas the score is 0 if the data are randomly partitioned. The higher the NMI score is, the better the clustering quality is.

Since there are no predefined categories in our data, we have to design an alternative way to carry out the evaluation. In this paper, we use the K-means clustering result of the combined feature data as the target labels for feature clustering. Similarly, the K-means clustering result of the combined user data is used as the target labels for user clustering. To randomize the experiments, we conduct the evaluation using different cluster numbers. Note that all the experiments ran on the same machine. At each time step, for each given cluster number, 30 test runs are carried out on different randomly chosen clusters. For each experiment, we run it 10 times and report the average.

5.2. Synthetic Data Sets. In this section, we use synthetic datasets to demonstrate that STClu can significantly cluster data for multiple evolving data streams over time. The number of nodes and number of clusters in each time step are given. The synthetic datasets are generated by two steps [18]. The first step generates the objects' geographic information, and the second step generates the values of the objects' nongeographic attributes. The objects' nongeographic attributes are generated by applying prototype systems used in [6]. It can be defined as follows: $f(t+\Delta t) = f(t)+f'(t+\Delta t)$, $f'(t+\Delta t) = f'(t) + u(t)$, in which $f(\cdot)$ is a stochastic process, $t = 0$, and $\Delta t, 2\Delta t, \ldots$ are independent random variables, uniformly distributed in an interval $[-a, a]$. So, the data stream $S(\cdot)$ can be defined by $S(t) = f(t + h(t)) + g(t)$, where $h(\cdot)$ and $g(\cdot)$ are stochastic processes that are generated in the same way as the prototype $f(\cdot)$. The constant a determines the smoothness of a data stream, which can be different for $p(\cdot)$, $h(\cdot)$, and $g(\cdot)$, such as 0.05, 0.1, and 0.2, respectively. To examine the quality of STClu for clustering multiple data streams, we generate random datasets with the number of streams ranging from 100 to 1000. Each stream has a length of 2,000 points.

We show the performance of four methods while the time window evolving over time and the number of clusters in the data varies from 2 to 20. The stream number is fixed at 400, and each of them contains 1,000 points. As can be seen in Table 1, the average ACC and NMI among four methods

Table 1: Comparison of ACC and NMI results for the four methods based on synthetic data sets.

k	ACC				NMI			
	K-means	Ncut	NMF	STClu	K-means	Ncut	NMF	STClu
2	0.3959	0.4954	0.5449	0.6542	0.3480	0.5130	0.5895	0.6749
4	0.3855	0.4740	0.5120	0.6436	0.4148	0.5233	0.4416	0.6967
5	0.3773	0.4685	0.5205	0.6757	0.3810	0.5016	0.5781	0.6122
6	0.3710	0.4866	0.5425	0.5603	0.4162	0.5430	0.6041	0.6059
8	0.3864	0.4636	0.4893	0.6038	0.4257	0.5247	0.5098	0.7022
10	0.3711	0.4540	0.4919	0.6231	0.3960	0.5154	0.4082	0.7238
12	0.3896	0.4802	0.5231	0.5868	0.4051	0.5088	0.5916	0.5421
15	0.3819	0.4728	0.5102	0.4834	0.4180	0.5207	0.5252	0.6281
18	0.4121	0.4827	0.5082	0.5293	0.4315	0.5786	0.5850	0.5823
20	0.4019	0.5275	0.6110	0.6516	0.4171	0.5823	0.4390	0.6299
Average	0.3873	0.4805	0.5254	0.6012	0.4053	0.5311	0.5272	0.6398

with the different numbers on synthetic dataset are shown. It can be clearly found that the STClu outperforms all other approaches and having highest NMIs at evolving time steps. The reason is that it incorporates the previous clustering results of the upstream sensors and the spatial relationship between the upstream and downstream sensors.

The average ACC and NMI value for the proposed algorithm varies with the varied number of clusters or data streams shown in Figures 3(a) and 3(b), respectively. As shown in Figure 3, we can observe that the performance of the proposed algorithm STClu is very robust to the number of clusters and data streams.

Finally, we evaluate the average processing time for a round of clustering multiple data streams. There are different factors affecting the execution time. In this experiment, we evaluate the effect of the window size and the number of data streams on the response time for the compared algorithms.

The following set of experiments evaluate the effect of window size on the execution time of these algorithms. Figure 4(a) shows that as the window size increases, the processing time of these algorithms increases. The y-axis shows the execution time, while the x-axis represents the window size $w(t)$ from 60 to 600. In Figure 4(b), the y-axis shows the execution time, while the x-axis represents the number of data streams being varied from 100 to 1000. The other parameters are fixed as follows: $w(t) = 300\,\text{s}$, $k = 4$. Note that as the number of streams increases, the execution time of these methods increases. However, the time of the STClu and NMF increases much more slowly than others. This is because of the execution efficiency of STClu and NMF for clustering multiple data streams in the approximate subspace. Although STClu requires more time than NMF, as can be seen from the experiments, it is able to process extremely large sensor data streams with spatial, temporal, and other multidimensional information.

5.3. Real World Data Sets. The real world data sets were obtained from the PeMS (http://pems.dot.ca.gov/) database. With sensor devices installed on road networks, a system monitors traffic flows on major US highways 24 hours a day, 7 days a week, thus acquiring huge volumes of data. We collected data from 963 sensors positioned along the freeway. We selected 15 months of daily data records giving the occupancy rates (between 0 and 1) of different car lanes on freeways in the San Francisco bay area over time. The data set covered records sampled every 10 min from 1 January 2008 to 30 March 2009. Data for public holidays were removed from the data set, as well as datva collected on two days with anomalies (8 and 9 March 2008), where all sensors were muted between 2:00 and 3:00 am.

First, we discuss the experiments on the PeMS data set with the number of clusters varying from 2 to 20. For each given cluster number k, the clustering results of all these algorithms on the PeMS data set are shown in Table 2; Ncut, NMF, and the proposed STClu algorithm achieve better performance than the K-means algorithm, since these three algorithms consider the geometrical structure information contained in the data.

Next, we discuss the experiments on the PeMS data set with different time steps. For each time step, the clustering results of all these algorithms on the PeMS data set are shown in Table 3. We observe that the STClu algorithm performs better than the other three algorithms for most time steps, except $t = 1$. The reason for this is that STClu integrates more historical knowledge and the spatial relationship between two sensors in proximity.

Finally, we report the experimental results on PeMS data set over a different number of time steps with a varying number of clusters. The clustering results of all these algorithms on the PeMS data set are shown in Table 4. STClu outperforms the other three algorithms in terms of both ACC and NMI, because it simultaneously considers the geometric structure and incorporates the prior knowledge and the spatial correlation between sensors in proximity.

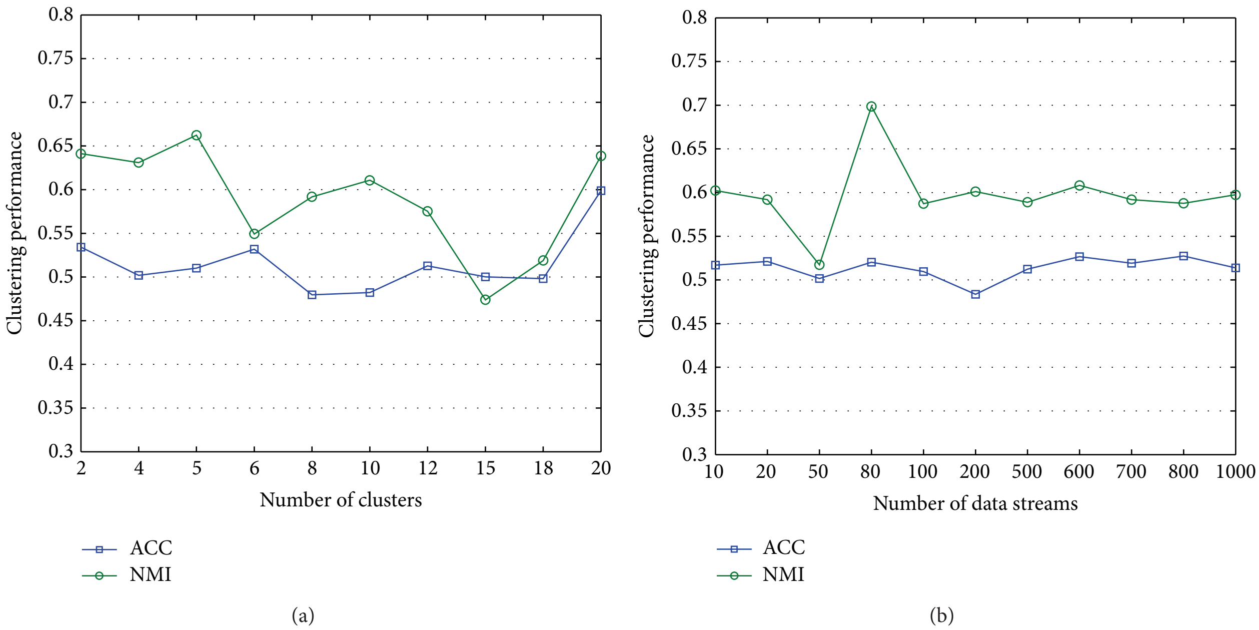


Figure 3: The performance of the proposed algorithm varying with the size of (a) clusters, (b) data streams, respectively.

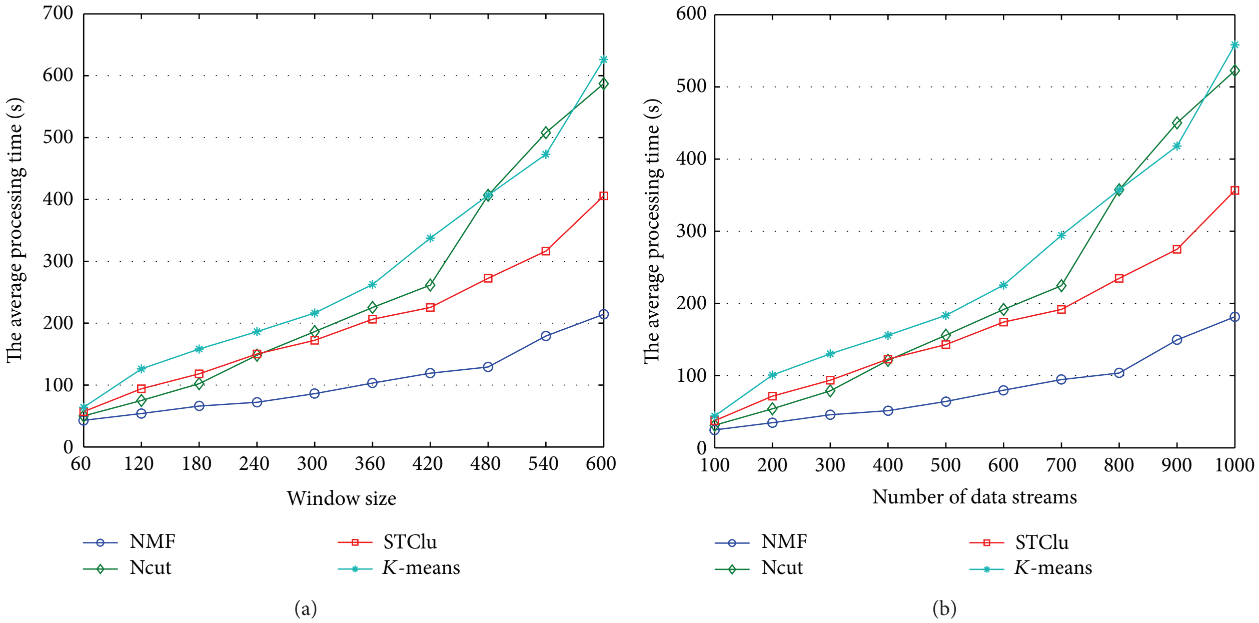


Figure 4: Average processing time: (a) while the window size $w(t)$ varies from 60 to 600, (b) while the number of streams varies from 100 to 1000, respectively.

6. Conclusions

Clustering multiple sensor data streams in CPS has been extensively studied in various applications, including transportation systems, sensor networks, and social networks. To extract and retain meaningful information from multiple sensor data streams, we assume that the spatial feature is the summary of the atypical event in temporal dimension, and the temporal feature is the summary of the event in spatial dimension. In this work, we have proposed a spatial-temporal clustering algorithm, called STClu. It utilizes time-series observational data streams and geospatial relationship for clustering multiple sensor data streams. It aims to discover clusters of objects with similar behavior and discover potential anomalies which may be revealed by the relationship evolving over time. STClu simultaneously incorporates the spatial relationship between two sensors in proximity and integrates the historical information into

Table 2: Comparison of the ACC and NMI results for the four methods on PeMS with the number of clusters varying from 2 to 20.

k	ACC				NMI			
	K-means	Ncut	NMF	STClu	K-means	Ncut	NMF	STClu
2	0.2716	0.3014	0.3163	0.5060	0.2808	0.3027	0.3260	0.5127
3	0.2685	0.2950	0.3064	0.5065	0.2762	0.2947	0.3197	0.5167
5	0.2660	0.2933	0.3089	0.5107	0.2741	0.2952	0.3256	0.5540
8	0.2641	0.2988	0.3155	0.5123	0.2759	0.3011	0.3119	0.5679
10	0.2687	0.2919	0.2996	0.5078	0.2748	0.2899	0.3105	0.5282
12	0.2642	0.2890	0.3004	0.5115	0.2711	0.2889	0.3137	0.5611
15	0.2697	0.2969	0.3097	0.5057	0.2777	0.2973	0.3129	0.5098
19	0.2674	0.2946	0.3059	0.5104	0.2754	0.2943	0.2959	0.5510
20	0.2764	0.2976	0.3052	0.5164	0.2813	0.2955	0.3023	0.6043
Average	0.2685	0.2954	0.3075	0.5097	0.2764	0.2955	0.3132	0.5451

Table 3: Comparison of the ACC and NMI results for the four methods on PeMS with different time steps.

Time step	ACC				NMI			
	K-means	Ncut	NMF	STClu	K-means	Ncut	NMF	STClu
1	0.2671	0.3033	0.3262	0.5052	0.2796	0.3085	0.3248	0.5052
2	0.2777	0.3157	0.3340	0.5053	0.2908	0.3184	0.3276	0.5065
3	0.2805	0.3102	0.3057	0.5100	0.2895	0.3019	0.3279	0.5477
4	0.2716	0.3074	0.2753	0.5083	0.2838	0.2856	0.3162	0.5323
5	0.2744	0.3054	0.3303	0.5140	0.2842	0.3115	0.3165	0.5836
6	0.2782	0.3090	0.3104	0.5028	0.2878	0.3035	0.3193	0.4839
7	0.2823	0.3263	0.3283	0.5077	0.2983	0.3208	0.3214	0.5277
8	0.2779	0.3275	0.2845	0.5091	0.2967	0.2999	0.3069	0.5395
9	0.2576	0.3409	0.3810	0.5027	0.2881	0.3796	0.3346	0.4834
10	0.2480	0.3256	0.3627	0.5086	0.2779	0.3759	0.3423	0.5350
11	0.2390	0.3118	0.3471	0.5072	0.2698	0.3678	0.3245	0.5227
12	0.2305	0.2989	0.3315	0.5120	0.2637	0.3656	0.3445	0.5653
Average	0.2654	0.3152	0.3264	0.5077	0.2842	0.3283	0.3255	0.5277

Table 4: Comparison of the ACC and NMI results for the four methods on PeMS with the number of clusters varying from 2 to 20 at different time steps.

Time step	ACC				NMI			
	K-means	Ncut	NMF	STClu	K-means	Ncut	NMF	STClu
1	0.2694	0.3024	0.3213	0.5056	0.2802	0.3056	0.3254	0.5090
2	0.2731	0.3054	0.3202	0.5059	0.2835	0.3066	0.3237	0.5116
3	0.2733	0.3018	0.3073	0.5104	0.2818	0.2986	0.3268	0.5509
4	0.2679	0.3031	0.2954	0.5103	0.2799	0.2934	0.3141	0.5501
5	0.2716	0.2987	0.3150	0.5109	0.2795	0.3007	0.3135	0.5559
6	0.2712	0.2990	0.3054	0.5072	0.2795	0.2962	0.3165	0.5225
7	0.2760	0.3116	0.3190	0.5067	0.2880	0.3091	0.3172	0.5188
8	0.2727	0.3111	0.2952	0.5098	0.2861	0.2971	0.3014	0.5453
9	0.2670	0.3193	0.3431	0.5096	0.2847	0.3376	0.3185	0.5439
10	0.2607	0.3184	0.3494	0.5107	0.2822	0.3466	0.3389	0.5536
11	0.2481	0.3093	0.3384	0.5085	0.2731	0.3398	0.3301	0.5344
12	0.2539	0.3044	0.3084	0.5129	0.2757	0.3286	0.3308	0.5731
Average	0.2654	0.3152	0.3264	0.5077	0.2842	0.3283	0.3255	0.5277

consideration. Experimental results on synthetic and real world data sets show that STClu achieves a better performance for clustering multiple sensor data streams evolving over time.

Appendix

In this section, we investigate the convergence and correctness of the objective function in (4) under update rules (9), (10), (11), and (12). Regarding these four update rules, we have the following theorem.

Theorem A.1. *For $X^{(t)}$, $C^{(t)}$, $U^{(t)}$, $R^{(t)}$, and $U^{(t-1,t)} \geq 0$, the update formulas for $U^{(t)}$, $C^{(t)}$, $R^{(t)}$, and $U^{(t-1,t)}$ given in* (9), (10), (11), *and* (12), *respectively, monotonically decrease the objective function in* (4).

The proofs are provided with the aid of auxiliary functions. Since the first term of the objective function J_{STClu} in (4) is only related to $U^{(t)}$, the first term of the objective function J_{STClu} in (4) is only related to $C^{(t)}$, and the third term of J_{STClu} in (4) is only related to $U^{(t-1,t)}$, we have exactly the same updating rule for $U^{(t)}$, $C^{(t)}$, and $U^{(t-1,t)}$as in [19], respectively. Thus, we can use the convergence proof in [19] to show that J_{STClu} is monotonically decreasing under the updating rules in (9), (10), and (12), respectively. Please see [19, 36] for details. Next, we prove convergence under the update rule for $R^{(t)}$ in (11).

Definition A.2. $Z(v, v')$ is an auxiliary function for $J(v)$ if the following two conditions are satisfied:

$$Z\left(v, v'\right) \geq J\left(v\right), \qquad Z\left(v, v\right) = J\left(v\right). \tag{A.1}$$

Lemma A.3. *For any v and v', if Z is an auxiliary function, J is monotonically decreasing under the formula*

$$v^{t+1} = \arg\min_{v} Z\left(v, v^{t}\right). \tag{A.2}$$

Proof. Consider

$$J\left(v^{t+1}\right) \leq Z\left(v^{t+1}, v^{t}\right) \leq Z\left(v^{t}, v^{t}\right) = J\left(v^{t}\right). \tag{A.3}$$

Thus, $Z(v^{t+1}, v^{t}) \leq Z(v^{t}, v^{t})$ and $J(v^{t+1}) \leq J(v^{t})$, which completes the proof. □

Next, we show that the update rule for $R^{(t)}$ in (12) is exactly the same as the update in (A.2) with a proper auxiliary function. Considering any element $R^{(t)}_{kj}$ in $R^{(t)}$, we let

$$\begin{aligned} J\left(R^{(t)}\right) = {} & \left\| X^{(t)} - C^{(t)}U^{(t)}R^{(t)^{T}} \right\|_{F}^{2} \\ & + \left\| X^{(t-1)} - C^{(t-1)}U^{(t-1)}R^{(t)^{T}} \right\|_{F}^{2} \\ & + \left\| X^{(t-1,t)} - R^{(t)}U^{(t-1,t)}R^{(t-1)^{T}} \right\|_{F}^{2}. \end{aligned} \tag{A.4}$$

Then, we get

$$\begin{aligned} & J\left(R^{(t)}\right)'_{kj} \\ & \quad = \left[\frac{\partial J\left(R^{(t)}\right)}{\partial R^{(t)}} \right]_{kj} \\ & \quad = \left[-2X^{(t)^{T}}C^{(t)}U^{(t)} + 2R^{(t)}U^{(t)^{T}}C^{(t)^{T}}C^{(t)}U^{(t)} \right. \\ & \qquad - 2X^{(t-1)^{T}}C^{(t-1)}U^{(t-1)} \\ & \qquad + 2R^{(t)}U^{(t-1)^{T}}C^{(t-1)^{T}}C^{(t-1)}U^{(t-1)} \\ & \qquad - 2X^{(t-1,t)}R^{(t-1)}U^{(t-1,t)^{T}} \\ & \qquad \left. + 2R^{(t)}U^{(t-1,t)}R^{(t-1)^{T}}R^{(t-1)}U^{(t-1,t)^{T}} \right]_{kj}, \\ & J\left(R^{(t)}\right)''_{kj} \\ & \quad = \left[2U^{(t)^{T}}C^{(t)^{T}}C^{(t)}U^{(t)} + 2U^{(t-1)^{T}}C^{(t-1)^{T}}C^{(t-1)}U^{(t-1)} \right. \\ & \qquad \left. + 2U^{(t-1,t)}R^{(t-1)^{T}}R^{(t-1)}U^{(t-1,t)^{T}} \right]_{kj}. \end{aligned} \tag{A.5}$$

Since our update rule operates elementwise, it is sufficient to show that each $J(R^{(t+1)})_{kj}$ is nonincreasing under the update formula in (12).

Lemma A.4. *The function*

$$\begin{aligned} & Z\left(R^{(t)}_{kj}, R^{(t)'}_{kj}\right) \\ & \quad = J_{kj}\left(R^{(t)'}_{kj}\right) + J'_{kj}\left(R^{(t)'}_{kj}\right)\left(R^{(t)}_{kj} - R^{(t)'}_{kj}\right) \\ & \qquad + \left(\left(\left[R^{(t)}U^{(t)^{T}}C^{(t)^{T}}C^{(t)}U^{(t)}\right.\right.\right. \\ & \qquad\qquad + R^{(t)}U^{(t-1)^{T}}C^{(t-1)^{T}}C^{(t-1)}U^{(t-1)} \\ & \qquad\qquad \left.\left. + R^{(t)}U^{(t-1,t)}R^{(t-1)^{T}}R^{(t-1)}U^{(t-1,t)^{T}}\right]_{kj}\right) \\ & \qquad \times \left.\left(R^{(t)'}_{kj}\right)^{-1}\right)\left(R^{(t)}_{kj} - R^{(t)'}_{kj}\right)^{2} \end{aligned} \tag{A.6}$$

is an auxiliary function for $J_{kj}(R^{(t)}_{kj})$.

Proof. We first get the Taylor series expansion of $J_{kj}(R^{(t)}_{kj})$:

$$\begin{aligned} & J_{kj}\left(R^{(t)}_{kj}\right) \\ & \quad = J_{kj}\left(R^{(t)'}_{kj}\right) + J'_{kj}\left(R^{(t)'}_{kj}\right)\left(R^{(t)}_{kj} - R^{(t)'}_{kj}\right) \\ & \qquad + \left\{\left[U^{(t)^{T}}C^{(t)^{T}}C^{(t)}U^{(t)} + U^{(t-1)^{T}}C^{(t-1)^{T}}C^{(t-1)}U^{(t-1)}\right.\right. \end{aligned}$$

$$+\ U^{(t-1,t)}R^{(t-1)^T}R^{(t-1)}U^{(t-1,t)^T}\Big]_{kj}\Big\}$$
$$\times\left(R_{kj}^{(t)}-R_{kj}^{(t)'}\right)^2. \tag{A.7}$$

Using (A.6) to find that $Z(R_{kj}^{(t)}, R_{kj}^{(t)'}) \geq J_{kj}(R_{kj}^{(t)})$ is equivalent to

$$\begin{aligned}&\Big(\Big[R^{(t)}U^{(t)^T}C^{(t)^T}C^{(t)}U^{(t)} + R^{(t)}U^{(t-1)^T}C^{(t-1)^T}C^{(t-1)}U^{(t-1)}\\&\quad + R^{(t)}U^{(t-1,t)}R^{(t-1)^T}R^{(t-1)}U^{(t-1,t)^T}\Big]_{kj}\Big)\left(R_{kj}^{(t)'}\right)^{-1}\\&\quad\geq\Big\{\Big[U^{(t)^T}C^{(t)^T}C^{(t)}U^{(t)} + U^{(t-1)^T}C^{(t-1)^T}C^{(t-1)}U^{(t-1)}\\&\qquad + U^{(t-1,t)}R^{(t-1)^T}R^{(t-1)}U^{(t-1,t)^T}\Big]_{kj}\Big\},\end{aligned} \tag{A.8}$$

we get

$$\begin{aligned}&\Big[R^{(t)}U^{(t)^T}C^{(t)^T}C^{(t)}U^{(t)} + R^{(t)}U^{(t-1)^T}C^{(t-1)^T}C^{(t-1)}U^{(t-1)}\\&\quad + R^{(t)}U^{(t-1,t)}R^{(t-1)^T}R^{(t-1)}U^{(t-1,t)^T}\Big]_{kj}\\&\quad=\sum_{l}^{K}R_{kl}^{(t)'}\Big[U^{(t)^T}C^{(t)^T}C^{(t)}U^{(t)}\\&\qquad + U^{(t-1)^T}C^{(t-1)^T}C^{(t-1)}U^{(t-1)}\\&\qquad + U^{(t-1,t)}R^{(t-1)^T}R^{(t-1)}U^{(t-1,t)^T}\Big]_{lj}\\&\quad\geq R_{kl}^{(t)'}\Big[U^{(t)^T}C^{(t)^T}C^{(t)}U^{(t)}\\&\qquad + U^{(t-1)^T}C^{(t-1)^T}C^{(t-1)}U^{(t-1)}\\&\qquad + U^{(t-1,t)}R^{(t-1)^T}R^{(t-1)}U^{(t-1,t)^T}\Big]_{jj}.\end{aligned} \tag{A.9}$$

Thus, (A.8) holds $Z(R_{kj}^{(t)}, R_{kj}^{(t)'}) \geq J_{kj}(R_{kj}^{(t)})$. Furthermore, $Z(R_{kj}^{(t)}, R_{kj}^{(t)'}) = J_{kj}(R_{kj}^{(t)})$ is obvious. □

Proof of Theorem A.1. Replacing $Z(R_{kj}^{(t)}, R_{kj}^{(t)'})$ in (A.2) by (A.6), we get

$$\begin{aligned}&R_{kj}^{(t)}\\&\quad=R_{kj}^{(t)'}-R_{kj}^{(t)'}\\&\qquad\times\left(\left(J_{kj}'\left(R_{kj}^{(t)'}\right)\right)\right.\\&\qquad\times\Big(2\Big[R^{(t)}U^{(t)^T}C^{(t)^T}C^{(t)}U^{(t)}\\&\qquad + R^{(t)}U^{(t-1)^T}C^{(t-1)^T}C^{(t-1)}U^{(t-1)}\\&\qquad + R^{(t)}U^{(t-1,t)}R^{(t-1)^T}R^{(t-1)}U^{(t-1,t)^T}\Big]_{kj}\Big)^{-1}\Big)\\&\quad=R_{kj}^{(t)'}\Big(\Big(\Big[X^{(t)}C^{(t)}U^{(t)} + X^{(t-1)}C^{(t-1)}U^{(t-1)}\\&\qquad + X^{(t-1,t)}R^{(t-1)}U^{(t-1,t)^T}\Big]_{kj}\Big)\\&\qquad\times\Big(\Big[R^{(t)}U^{(t)^T}C^{(t)^T}C^{(t)}U^{(t)}\\&\qquad + R^{(t)}U^{(t-1)^T}C^{(t-1)^T}C^{(t-1)}U^{(t-1)}\\&\qquad + R^{(t)}U^{(t-1,t)}R^{(t-1)^T}R^{(t-1)}U^{(t-1,t)^T}\Big]_{kj}\Big)^{-1}\Big).\end{aligned} \tag{A.10}$$

Since (A.6) is an auxiliary function for $J_{kj}(R_{kj}^{(t)})$, $J_{kj}(R_{kj}^{(t)})$ is nonincreasing under the update rule in (11). □

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors gratefully acknowledge the support provided for this research by the Research Fund for the Doctoral Program of Higher Education of China (Grant no. 20120191110047), Natural Science Foundation Project of CQ CSTC of China (Grant no. CSTC2012JJB40002), Engineering Center Research Program of Chongqing of China (Grant no. 2011pt-gc30005), Fundamental Research Funds for the Central Universities of China (Grant no. 106112014CDJZR178801), and Fundamental Research Funds for the Central Universities of China (Grant no. CDJXS10170004).

References

[1] J. Sztipanovits, X. Koutsoukos, G. Karsai et al., "Toward a science of cyber-physical system integration," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 29–44, 2012.

[2] L. Xiang, J. Luo, and A. Vasilakos, "Compressed data aggregation for energy efficient wireless sensor networks," in *Proceedings of the 8th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON '11)*, pp. 46–54, June 2011.

[3] L. Mo, L. Zhenjiang, and A. V. Vasilakos, "A survey on topology control in wireless sensor networks: taxonomy, comparative study, and open issues," *Proceedings of the IEEE*, vol. 101, no. 12, pp. 2538–2557, 2013.

[4] S. Geisler, C. Quix, S. Schiffer, and M. Jarke, "An evaluation framework for traffic information systems based on data streams," *Transportation Research C: Emerging Technologies*, vol. 23, pp. 29–55, 2012.

[5] L.-A. Tang, X. Yu, S. Kim et al., "Multidimensional sensor data analysis in cyber-physical system: an atypical cube approach," *International Journal of Distributed Sensor Networks*, vol. 2012, Article ID 724846, 19 pages, 2012.

[6] J. Beringer and E. Hüllermeier, "Online clustering of parallel data streams," *Data and Knowledge Engineering*, vol. 58, no. 2, pp. 180–204, 2006.

[7] Z. A. Aghbari, I. Kamel, and T. Awad, "On clustering large number of data streams," *Intelligent Data Analysis*, vol. 16, no. 1, pp. 69–91, 2012.

[8] B. Dai, J. Huang, M. Yeh, and M. Chen, "Adaptive clustering for multiple evolving streams," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 9, pp. 1166–1180, 2006.

[9] M. Yeh, B. R. Dai, and M. S. Chen, "Clustering over multiple evolving streams by events and correlations," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 10, pp. 1349–1362, 2007.

[10] C. C. Aggarwal and P. S. Yu, "Online analysis of community evolution in data streams," in *Proceedings of the 5th SIAM International Conference on Data Mining (SDM '05)*, pp. 56–67, April 2005.

[11] D. Chakrabarti, R. Kumar, and A. Tomkins, "Evolutionary clustering," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '06)*, pp. 554–560, August 2006.

[12] Y.-R. Lin, Y. Chi, S. Zhu, H. Sundaram, and B. L. Tseng, "Analyzing communities and their evolutions in dynamic social networks," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 3, p. 8, 2009.

[13] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E*, vol. 69, no. 2, Article ID 026113, 2004.

[14] H. Ning, W. Xu, Y. Chi, Y. Gong, and T. S. Huang, "Incremental spectral clustering by efficiently updating the eigen-system," *Pattern Recognition*, vol. 43, no. 1, pp. 113–127, 2010.

[15] L. Wang, M. Rege, M. Dong, and Y. Ding, "Low-rank kernel matrix factorization for large-scale evolutionary clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 6, pp. 1036–1050, 2012.

[16] Y. Chi, X. Song, D. Zhou, K. Hino, and B. L. Tseng, "Evolutionary spectral clustering by incorporating temporal smoothness," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '07)*, pp. 153–162, August 2007.

[17] Y. J. Ma, Y. K. Guo, X. C. Tian, and M. Ghanem, "Distributed clustering-based aggregation algorithm for spatial correlated sensor networks," *IEEE Sensors Journal*, vol. 11, no. 3, pp. 641–648, 2011.

[18] L.-Y. Wei and W.-C. Peng, "An incremental algorithm for clustering spatial data streams: exploring temporal locality," *Knowledge and Information Systems*, vol. 37, pp. 453–483, 2013.

[19] C. Ding, T. Li, W. Peng, and H. Park, "Orthogonal nonnegative matrix tri-factorizations for clustering," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '06)*, pp. 126–135, August 2006.

[20] D. Cai, X. He, J. Han, and T. S. Huang, "Graph regularized nonnegative matrix factorization for data representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1548–1560, 2011.

[21] Z. He, S. Xie, R. Zdunek, G. Zhou, and A. Cichocki, "Symmetric nonnegative matrix factorization: algorithms and applications to probabilistic clustering," *IEEE Transactions on Neural Networks*, vol. 22, no. 12, pp. 2117–2131, 2011.

[22] B. Vanluyten, J. C. Willems, and B. De Moor, "Structured nonnegative matrix factorization with applications to hidden Markov realization and clustering," *Linear Algebra and its Applications*, vol. 429, no. 7, pp. 1409–1424, 2008.

[23] N. D. Ho and P. Van Dooren, "Non-negative matrix factorization with fixed row and column sums," *Linear Algebra and its Applications*, vol. 429, no. 5-6, pp. 1020–1025, 2008.

[24] D. Kuang, C. Ding, and H. Park, "Symmetric nonnegative matrix factorization for graph clustering," in *Proceedings of the 12th SIAM International Conference on Data Mining (SDM '12)*, pp. 106–117, April 2012.

[25] A. Cichocki, R. Zdunek, A. H. Phan, and S. Amari, *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-Way Data Analysis and Blind Source Separation*, Wiley Publishing, 2009.

[26] C. Ding, X. F. He, and H. D. Simon, "On the equivalence of nonnegative matrix factorization and spectral clustering," in *Proceedings of the 5th SIAM International Conference on Data Mining (SDM '05)*, pp. 606–610, April 2005.

[27] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 2012.

[28] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[29] C. Ding, T. Li, and M. I. Jordan, "Convex and semi-nonnegative matrix factorizations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 45–55, 2010.

[30] P. Drineas, R. Kannan, and M. W. Mahoney, "Fast Monte Carlo algorithms for matrices III: computing a compressed approximate matrix decomposition," *The SIAM Journal on Computing*, vol. 36, no. 1, pp. 184–206, 2006.

[31] J. Sun, Y. Xie, H. Zhang, and C. Faloutsos, "Less is more: compact matrix decomposition for large sparse graphs," in *Proceedings of the 7th SIAM International Conference on Data Mining*, pp. 366–377, April 2007.

[32] H. Tong, S. Papadimitriou, J. Sun, P. S. Yu, and C. Faloutsos, "Colibri: fast mining of large static and dynamic graphs," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '08)*, pp. 686–694, August 2008.

[33] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, Cambridge, UK, 2004.

[34] J. B. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[35] A. Strehl and J. Ghosh, "Cluster ensembles—a knowledge reuse framework for combining multiple partitions," *The Journal of Machine Learning Research*, vol. 3, no. 3, pp. 583–617, 2003.

[36] Q. Gu and J. Zhou, "Co-clustering on manifolds," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09)*, pp. 359–367, July 2009.

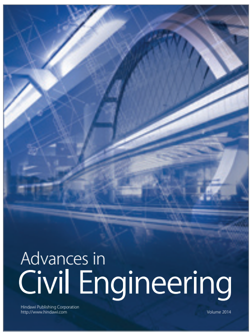
Journal of
Engineering

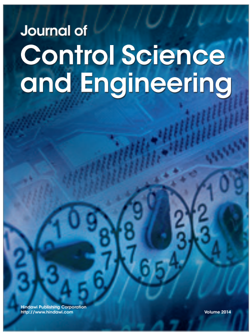
The Scientific World Journal

International Journal of
Rotating Machinery

Journal of
Sensors

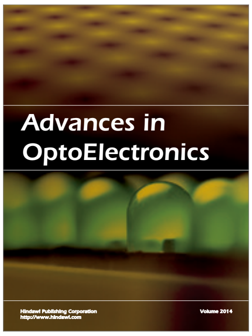
International Journal of
Distributed Sensor Networks


Advances in
Civil Engineering


Journal of
Control Science and Engineering

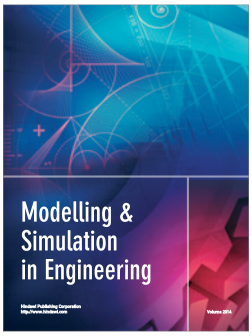
Journal of
Robotics

Journal of
Electrical and Computer Engineering

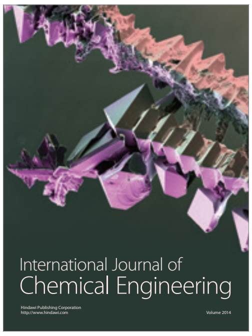

Advances in
OptoElectronics

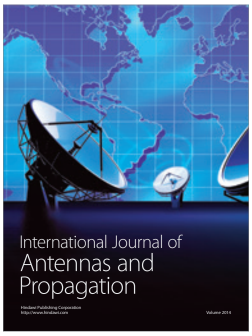
VLSI Design

International Journal of
Navigation and Observation

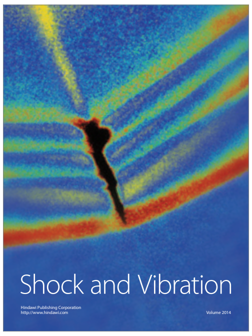

Modelling &
Simulation
in Engineering

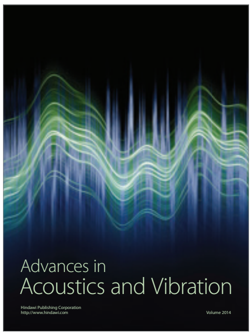
International Journal of
Aerospace Engineering


International Journal of
Chemical Engineering


International Journal of
Antennas and Propagation

Active and Passive
Electronic Components


Shock and Vibration


Advances in
Acoustics and Vibration