Maximum Margin Clustering on Evolutionary Data

Xuhui Fan,
Longbing Cao, Xia Cui
Advanced Analytics Institute
Faculty of Engineering and
Information Technology
University of Technology
Sydney, Australia
Xuhui.Fan@student.uts.edu.au

Lin Zhu
Institute of Image Processing
and Pattern Recognition
Shanghai Jiao Tong University,
Shanghai, China
wxzhulin@yahoo.com.cn

Yew-Soon Ong
School of Computer
Engineering
Nanyang Technological
University
asysong@ntu.edu.sg

ABSTRACT

Evolutionary data, such as topic changing blogs and evolving trading behaviors in capital market, is widely seen in business and social applications. The time factor and intrinsic change embedded in evolutionary data greatly challenge evolutionary clustering. To incorporate the time factor, existing methods mainly regard the evolutionary clustering problem as a linear combination of *snapshot cost* and *temporal cost*, and reflect the time factor through the *temporal cost*. It still faces accuracy and scalability challenge though promising results gotten. This paper proposes a novel evolutionary clustering approach, evolutionary maximum margin clustering (e-MMC), to cluster large-scale evolutionary data from the maximum margin perspective. e-MMC incorporates two frameworks: *Data Integration* from the data changing perspective and *Model Integration* corresponding to model adjustment to tackle the time factor and change, with an adaptive label allocation mechanism. Three e-MMC clustering algorithms are proposed based on the two frameworks. Extensive experiments are performed on synthetic data, UCI data and real-world blog data, which confirm that e-MMC outperforms the state-of-the-art clustering algorithms in terms of accuracy, computational cost and scalability. It shows that e-MMC is particularly suitable for clustering large-scale evolving data.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning; I.5.3 [**Pattern Recognition**]: Clustering.

Keywords

Maximum Margin Clustering; Evolutionary data.

1. INTRODUCTION

Evolutionary data is ubiquitous, such as social networking data and capital market trading information, and is increasing exponentially with the widespread development and emergence of business and social applications. The key challenge of learning evolutionary data lies on its evolution nature with time development, for instance, the change of topics in blogging or the adjustment of trading behaviors [3]. Evolutionary clustering is a topic aiming at segmenting such time-varied data [4].

A critical factor in evolutionary data is the time factor t. It affects the accuracy, consistency and robustness of evolutionary clustering algorithms when the data presents dynamics in attribute values or interactions between data objects at different time points. This is much more challenging when learning a large scale of evolutionary data. A typical approach is to verify the modeling performance between data in a historical time window and the data currently learned, and to apply techniques like weighting to adjust the learning objective function.

Typical evolutionary clustering includes agglomerative clustering, k-means clustering [4], and spectral clustering [6]. For all these proposed approaches, the time factor t is involved as a smoothness control term in adjusting the clustering performance of the current data against that of historical one. It usually divides the objective function into two parts: *snapshot cost* (CS) defining the clustering quality of the current data, and *temporal cost* (CT) verifying the shift from historical records to current ones. It is reported that this can achieve rather promising outcomes, especially on the small size of data sets. They also face real-world data challenges, for example, the existing k-means evolutionary clustering algorithms suffer from complicated situations such as non-spherical datasets, and evolutionary spectral clustering algorithms perform unsatisfactorily on data with tens of thousands of objects.

This paper proposes a novel evolutionary clustering approach, Evolutionary Maximum Margin Clustering (e-MMC), for clustering evolutionary data. Unlike identifying centers of clusters in k-means evolutionary clustering in the common Euclidean Space or spectral clustering in Eigenspace, we employ the Maximum Margin Clustering (MMC) [20] algorithm to seek a hyperplane that best separates the data distribution in a pre-defined kernel space. This is motivated by the advantage of MMC through obtaining the maximum margin between two clusters to segment all possible clusters for a higher accuracy [20, 21] and even better computational performance [18]. e-MMC is challenged by identifying proper mechanisms for (1) incorporating the time influence into the MMC clustering process to obtain a time-smoothed data partition, and (2) tackling the large scale evolutionary data.

We incorporate the time factor into MMC and handle the time smoothness problem through two frameworks: *data integration* (DI) and *model integration* (MI). *Data integration* regards the historical data as the required records and measures the performance of the MMC-oriented margin partition on both current data and historical data with different weights. *Model integration* considers both the current data partition cost and the margin change in

CIKM'12, October 29–November 2, 2012, Maui, HI, USA.

terms of time. We employ an optimization strategy similar to the cutting plane MMC [23] to conduct e-MMC both efficiently and effectively. Three e-MMC based clustering algorithms are then proposed for evolutionary clustering. To the best of our knowledge, e-MMC is the first evolutionary version of the Maximum Margin Clustering algorithm to cluster the evolutionary data from the margin perspective, and can deal with the label assignments under the evolutionary framework adaptively.

We further verify the e-MMC approach, the two proposed frameworks, and the three algorithms through substantial experiments on synthetic dataset, UCI data and real-world blog data. Compared to the four baseline clustering algorithms, experimental results have shown that e-MMC can cluster evolutionary data with better accuracy, improved computational performance and scalability. It shows that e-MMC has a great potential for clustering large-scale evolutionary data, which is of high demand in the real world.

The rest of this paper is organized as follows. Section 2 introduces the related work on evolutionary clustering and MMC. Preliminaries and notations are introduced in Section 3. The evolutionary MMC approach and its two frameworks are described in Section 4. More discussion and analysis about them are available in Section 5. Extensive experimental evaluation is performed in Section 6. Section 7 concludes the work and discusses future work.

2. RELATED WORK

In contrast to static data set, various formats of non-static data set appear in business and social applications [2], such as social network linkage data and topic change in online news update, as well as behavioral data [3, 17]. Various effective methods have been put forward to tackle different characteristics of the evolving data, such as data stream clustering focusing on the one-scanned data [9, 1], and incremental clustering concentrating on updating the cluster parameters [10, 14, 16].

Focusing on the data attribute's evolving behavior, evolutionary clustering was first put forward by [4], where a framework was also defined for formulating the problem. Through exemplified algorithms of the bottom-up evolutionary hierarchical clustering and the evolutionary k-means clustering, they break the objective function into two parts: one focusing on the current data, and the other addressing the historical adjustment; both reaches satisfactory results on both current and historical data sets. In [6, 7], the two compositions are further extended to spectral clustering, enabling it to address more sophisticated situations. Both of the methods achieve successful results in capturing the evolving behaviors under some cases. However, according to their capacity of handling data size, data with larger than tens of thousands of objects would be challenging for them to process. What is more, both algorithms automatically assign the same label to current and previous time stamps in each data set. In practice, data objects often change so as to be associated with different labels during the evolution. This requests an adaptive way to assign labels.

Recently, [20] proposed Maximum Margin Clustering (MMC), which is inspired by the idea of Support Vector Machine (SVM). Similar to SVM, MMC also aims at seeking the maximum margin solution during unsupervised learning. This is to achieve the maximum margin between two clusters among all the possible cluster constitution. Experiments in [20, 21] show that MMC can achieve better performance on the clustering results, especially in the clustering accuracy. [18] further generalizes the MMC algorithm and reduces the corresponding computational load.

A most recent work in [11] involves MMC on time series data. It employs the MMC Algorithm to cluster a set of non-overlapping time series segments to overcome the intractable inference in generative models. However, it only segments time series data, without considering the data time evolving and there is no test on the large data scale case.

3. PRELIMINARIES

3.1 Notations

Here the evolutionary data is specialized by a discrete time factor t, and is denoted as $\{\boldsymbol{X}_t, t = 1, \cdots, T\}$. We use $\boldsymbol{X}_t = \{\boldsymbol{x}_{1,t}, \cdots, \boldsymbol{x}_{n,t}\} \in R^d$ to denote the whole n data points with d dimensions at time t. $\{\phi(\boldsymbol{x}_{i,t}), i = 1, \cdots, n\}$ are related to the corresponding kernel space used in our Maximum Margin Clustering framework, the same as used in Support Vector Machine, with Kernel matrix $K_t = \{k_{ij,t}\}^{n\times n} = \{\phi(\boldsymbol{x}_{i,t})\phi(\boldsymbol{x}_{j,t})\}^{n\times n}$. For a given similarity matrix, a Cholesky decomposition [13] of the kernel matrix $\boldsymbol{K} = \hat{\boldsymbol{X}}\hat{\boldsymbol{X}}^T$ is computed, and $\phi(\boldsymbol{x}_i)$ is taken as the corresponding set $(\hat{\boldsymbol{X}}_{i,1}, \cdots, \hat{\boldsymbol{X}}_{i,n})^T$.

3.2 Maximum Margin Clustering

Maximum Margin Clustering (MMC) [20, 21] is inspired by the idea of Support Vector Machine (SVM), a widely used classification method in machine learning.

Briefly speaking, SVM aims at finding an optimal hyperplane in a pre-defined kernel space $\{\mathcal{F} : \phi(\boldsymbol{x}) \in \mathcal{F}|\boldsymbol{\omega}\phi(\boldsymbol{x}) + b = 0\}$ that can best separate the data points with different labels $\{y_i\}_{i=1}^n$. Defining $\{\phi(\boldsymbol{x}_i)\}_{i=1}^n$ as the kernel coordinators of $\{\boldsymbol{x}_i\}_{i=1}^n$, C as the balancing parameter of the slack variables $\{\xi_i\}_{i=1}^n$, SVM seeks the optimal values of $\boldsymbol{\omega}, b, \xi$ for the optimization problem below:

$$\begin{aligned} \min_{\boldsymbol{\omega},b,\xi} \quad & \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + \frac{C}{n}\sum_{i=1}^{n}\xi_i \\ s.t. \quad & y_i(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_i) + b) \geq 1 - \xi_i; \\ & \xi_i \geq 0 \qquad i = 1, \cdots, n. \end{aligned} \tag{1}$$

With its success in real applications, MMC extends SVM to the unsupervised case, i.e., $\{y_i\}_{i=1}^n$ is unknown. That is to say, the goal of MMC is to find a labeling set $\{y_i\}_{i=1}^n$ that generates the largest margin among all the potential label assignments. More formally, the problem is defined as:

$$\begin{aligned} \min_{\boldsymbol{y}\in\{-1,+1\}^n} \min_{\boldsymbol{\omega},b,\xi} \quad & \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + \frac{C}{n}\sum_{i=1}^{n}\xi_i \\ s.t. \quad & y_i(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_i) + b) \geq 1 - \xi_i; \\ & \xi_i \geq 0 \qquad i = 1, \cdots, n; \\ & -l \leq \boldsymbol{e}^T\boldsymbol{y} \leq l \end{aligned} \tag{2}$$

where $l \geq 0$ is a constant controlling the clustering's balance and $\boldsymbol{e}$ is the all-one vector.

Comparing to SVM (Equation (1)), the only difference lies in the variables needed to be solved. MMC (Equation (2)) includes the label assignments $\{y_i\}_{i=1}^n$, while the SVM assumes to have the prior information. The label assignment in MMC makes it complicated to achieve the solutions.

The Cutting Plane MMC (CPMMC) Algorithm [23] is a recently proposed method to solve the MMC problem (Equation (2)) both efficiently and effectively. It is based on constructing a sequence of successively tighter relaxation of Equation (2); and during each of the intermediate tasks, a Wolfe dual form is utilized in the constrained concave-convex procedure.

In this paper, we use an optimization strategy similar as the CPMMC implementation to conduct MMC on evolutionary data. The details of its implementation are in [23] and Appendix A.

3.3 Evolutionary Data

Evolutionary data is closely related to stream data and incremental data. However, it differs from them since evolutionary data embeds data change as a major concern.

As in [4, 6], the evolutionary clustering clusters the current and historical data under the same clustering mechanism, although with a different weight in the objective cost function. A better performance of the cost function is expected to happen on both data sets rather than one of them only. This fact is embodied through the cost function by two compositions: *snapshot cost (CS)* measuring the proposed partition cost on the current data, and *temporal cost (CT)* scaling the temporal smoothness on historical data or historical partitions. More formally, the cost function for evolutionary clustering is defined as a linear combination of *CS* and *CT*, with a tuning parameter $\alpha(0 \leq \alpha \leq 1)$:

$$Cost = \alpha \cdot CS + (1-\alpha) \cdot CT \tag{3}$$

The larger α value, the more focus we put on the current data partition effect.

In [6, 7], evolutionary clustering is extended to the spectral clustering. Two frameworks called Preserving Cluster Quality (PCQ) and Preserving Cluster Membership (PCM) are proposed. Instead of taking the so-called "inter-cluster" cost as the cost function, the current partition cost function of evolutionary spectral clustering is set as the graph cut result, and the evaluation criterion of the historical data partition is categorized into the graph cut result on historical data in PCQ and partition shift from historical data to current data in PCM.

4. EVOLUTIONARY MMC FRAMEWORK

4.1 Basic Principle

Motivated by the evolutionary clustering strategies proposed in [4, 6], two novel Maximum Margin Clustering-based evolutionary frameworks are introduced to separate the evolving data distribution by a required hyperplane. More specifically, in order to avoid the confusion of different performance targets, we define two new evolutionary approaches: *snapshot margin(SM)* and *integration relaxation(IR)* to replace the *snapshot cost* and *temporal cost* in [4], respectively. The weighted linear combination function is defined to take both of the historical and current data into consideration.

$$\begin{aligned} &\text{Current Margin} = \alpha \cdot SM + (1-\alpha) \cdot IR \\ &SM, IR \text{ subject to some pre-defined constraints} \end{aligned} \tag{4}$$

Here, *SM* is formulated as the process of seeking objects with the maximum margins: $SM = \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + C \cdot \sum_{i=1}^{n} \xi_i$, while *IR* mainly considers the cost of incorporating historical records. Under the special case of $\alpha = 1$ with the initial constraints, the problem is converted into the common two-cluster MMC problem.

In this paper, we propose two novel frameworks corresponding to two different *IR* representatives. In the first framework, historical factor is represented by the historical data, termed *Data Integration (DI)*. *DI* utilizes the historical data and compares with the current data for data change and clustering performance variation. The data evolving behaviors in *DI* are categorized into *cohort data evolving* of the whole data set and *individual data evolving* of a single data object. The former captures the variation process of the whole data set, while the latter handles the particular movement of each data point. The second framework is named as *Model Integration* (MI) which focuses on margin evolving.

Figure 1 clearly depicts the structure of our proposed frameworks for MMC-based evolutionary clustering. Two frameworks are proposed from different points of view while encountering evolutionary data. One from the *data integration* perspective, which further handles two scenarios *cohort data evolving* and *individual data evolving* respectively. The other reflects *model integration*.

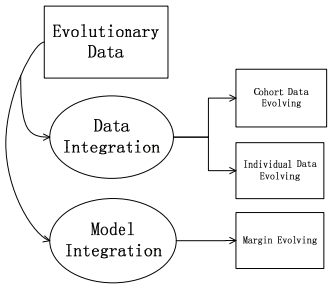

Figure 1: The evolutionary-MMC frameworks

4.2 Data Integration (DI)

In *data integration*, historical data is incorporated to represent the influence of time factor t. To simplify the algorithm, we mainly employ the data set of the current time t and that of the previous time $t-1$ to represent the whole data. A tuning parameter is set to differentiate the importance of the current and previous data sets.

By treating the data evolving behaviors from different perspectives, we deploy our framework into two different directions: *cohort data evolving* (CDE) in cohort analysis and *individual data evolving* (IDE) in individual observation. Below we explain them in detail.

4.2.1 Cohort Data Evolving(CDE)

In *cohort data evolving* (CDE), we treat the moving behavior of the whole data set as an evolving characteristic of the whole data set. This is reflected through the metrics such as mean and variation measuring the cluster shifting triggered by the evolution of its belonging data points. With this mechanism, each feasible solution in the MMC solution set is constrained by the specific time point. By denoting the MMC problem at time t as J_t and the *Integration Relaxation* (IR) as the MMC problem at time $t-1$, the CDE problem is simplified as:

$$\min_{\boldsymbol{y},\boldsymbol{\omega},b} \alpha \cdot J_t + (1-\alpha) \cdot J_{t-1} \tag{5}$$

According to the annotation for MMC in Equation (2), we specify our current margin problem (Equation (5)) as:

$$\begin{aligned} \min_{\boldsymbol{y}_t,\boldsymbol{y}_{t-1}\in\{-1,+1\}^n} \min_{\boldsymbol{\omega},b,\boldsymbol{\xi}_t,\boldsymbol{\xi}_{t-1}} & \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + \alpha \cdot \frac{C}{n}\sum_{i=1}^{n}\xi_{i,t} \\ & + (1-\alpha)\cdot\frac{C}{n}\sum_{i=1}^{n}\xi_{i,t-1} \\ s.t. & \; y_{i,t-1}(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_{i,t-1})+b) \geq 1-\xi_{i,t-1}; \\ & \; \xi_{i,t-1} \geq 0, i = 1,\cdots,n; \\ & \; y_{i,t}(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_{i,t})+b) \geq 1-\xi_{i,t}; \\ & \; \xi_{i,t} \geq 0, i = 1,\cdots,n. \end{aligned} \tag{6}$$

While the tuning parameter α satisfies $\alpha = 1-\alpha$, Equation (6) conveys a quite straightforward solution that seeks a margin to best separate the current data from historical data. Otherwise, the slack

variables are scaled according to its time stamp, leading to different weights in the solving process.

We employ the similar techniques in [23] to solve Equation (6). According to the results in [23]'s result, problem (6) is equally transformed into the following problem by eliminating the label $\{y_i\}_{i=1}^n$ temporally:

$$\begin{aligned}\min_{\boldsymbol{\omega},b,\boldsymbol{\xi}_t,\boldsymbol{\xi}_{t-1}} \quad & \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + \alpha\cdot\frac{C}{n}\sum_{i=1}^{n}\xi_{i,t} \\ & + (1-\alpha)\cdot\frac{C}{n}\sum_{i=1}^{n}\xi_{i,t-1} \\ s.t. \quad & |\boldsymbol{\omega}^T\phi(\boldsymbol{x}_{i,t-1}) + b| \geq 1 - \xi_{i,t-1}; \\ & \xi_{i,t-1} \geq 0, i = 1,\cdots,n; \\ & |\boldsymbol{\omega}^T\phi(\boldsymbol{x}_{i,t}) + b| \geq 1 - \xi_{i,t}; \\ & \xi_{i,t} \geq 0, i = 1,\cdots,n.\end{aligned} \tag{7}$$

To solve problem (7), we employ the same cutting plane strategy in [23]. Starting from an initialized constraint set Ω, we iteratively select the most violated label assignment $\boldsymbol{c} \in \{0,1\}^n$. Then we using a constraint concave convex procedure (CCCP) to optimize the solution until the pre-defined tolerable error contains the current most violated label assignment.

The Wolfe Dual form of the transformed CCCP problem is essential in the CCCP solving procedure. For simplicity, we show the related Wolfe Dual form and put it in Appendix B.1.

4.2.2 *Individual Data Evolving(IDE)*

In *individual data evolving* (IDE), we track the attribute change of each data point during the evolution. To make it clearer, we name the i-th data point from time t to $t-1$ as "time-paired data points" $\{\boldsymbol{x}_{i,t}, \boldsymbol{x}_{i,t-1}\}_{i=1}^n$. The following effective method is then proposed to find the structure of these "time-paired data points".

In respect to this time relation, the constraints of each time-paired data point have been shared (on ξ_i) here, reflected through the problem formalization that each time-paired data point employs the same slack variables. We again employ the tuning parameter α to unequally weight the data points at time points t and $t-1$, and thus convert the problem to the following:

$$\begin{aligned}\min_{\boldsymbol{y}\in\{-1,+1\}^n}\min_{\boldsymbol{\omega},b,\xi} \quad & \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + \frac{C}{n}\sum_{i=1}^{n}\xi_i \\ s.t. \quad & y_{i,t-1}(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_{i,t-1}) + b) \geq 1 - \frac{1}{1-\alpha}\cdot\xi_i; \\ & y_{i,t}(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_{i,t}) + b) \geq 1 - \frac{1}{\alpha}\cdot\xi_i; \\ & \xi_i \geq 0, i = 1,\cdots,n.\end{aligned} \tag{8}$$

Here *IR* does not have an explicit expression on the problem formalization; however, its influence reflects on the constraints. By this strategy, *IR* automatically chooses a slack variable that could cover the other's among the historical time point $t-1$ and the current time stamp t for all $i \in \{1,\cdots,n\}$.

Similar strategy as in *cohort data evolving* is used here to approximate problem (8). We put its Wolfe dual form in Appendix B.2 for consistency.

4.3 Model Integration (MI)

In *Model Integration*, historical records are mapped to previous hyperplane required in the learning. We target on a more stable margin result to keep the clustering result consistent. Also as predicted, a smaller shift is always preferable.

Firstly, *margin distance (md)* at time point t is introduced to better describe this shift. The *margin distance* is defined as the distance of i-th object to the required hyperplane:

$$md_{i,t} = \boldsymbol{\omega}_t\phi(\boldsymbol{x}_{i,t}) + b_T, i = 1,\cdots,n. \tag{9}$$

Here we should note that $\{md_{i,t}\}_{i=1}^n$ could be negative.

Further, we define the *margin shift (ms)* as the whole objects' difference between the current margin distance ($md_{i,t}$) and the previous margin distance ($md_{i,t-1}$).

$$ms = \sum_{i=1}^{n}(md_{i,t} - md_{i,t-1}) \tag{10}$$

Figure 2 illustrates the above concepts. M_{t-1} is the resulted hyperplane at the previous time point $t-1$. M_t^1 and M_t^2 are the two newly obtained hyperplanes without considering the historical records. $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are two data points distributed in different clusters. Each is associated with margin distances $d_{1,t-1}, d_{1,t}^1, d_{1,t}^2$ and $d_{2,t-1}, d_{2,t}^1, d_{2,t}^2$ corresponding to margins M_{t-1}, M_t^1, M_t^2, represented in dash line, solid line and dash dot line in the figure. As a result, we can easily obtain $\sum_{i=1}^2(d_{i,t}^1 - d_{i,t-1}) < \sum_{i=1}^2(d_{i,t}^2 - d_{i,t-1})$. If the performance of M_t^1 and M_t^2 on partitioning the current data is nearly the same, we should choose M_t^1 as the current margin according to Equations (9) and (10). It is also noticed that the variation of M_t^1 is smaller than that of M_t^2 compared to M_{t-1}.

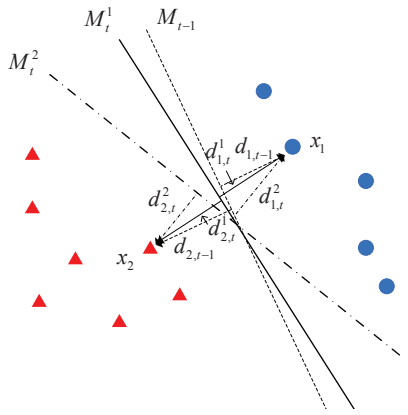

Figure 2: Model integration

In our *model integration (MI)* framework, *integration relaxation (IR)* in Equation (4) is the so-called *margin shift (ms)*. By incorporating the *IR* into the objective function, MI is formalized as:

$$\begin{aligned}\min_{\boldsymbol{y}\in\{-1,+1\}^n}\min_{\boldsymbol{\omega},b,\xi} \quad & \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + \frac{C}{n}\sum_{i=1}^{n}\xi_i + \alpha\cdot\sum_{i=1}^{n}\|\boldsymbol{\omega}_t\phi(\boldsymbol{x}_i) \\ & + b - (\boldsymbol{\omega}_{t-1}\phi(\boldsymbol{x}_i) + b_{t-1})\|_2^2 \\ s.t. \quad & y_i(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_{i,t}) + b) \geq 1 - \xi_i; \\ & \xi_i \geq 0 \qquad i = 1,\cdots,n.\end{aligned} \tag{11}$$

Here α is the tuning parameter as usual.

Due to the complexity of the problem-solving of Equation (11), we use an approximation of *IR* to simplify it. More specifically, *IR*

is separated into two terms: $\boldsymbol{\omega}$-penalty and b-penalty.

$$\begin{aligned}\min_{\boldsymbol{y}\in\{-1,+1\}^n}\min_{\boldsymbol{\omega},b,\xi}\quad & \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega}+\frac{C}{n}\sum_{i=1}^{n}\xi_i+\alpha\cdot\sum_{i=1}^{n}|\boldsymbol{\omega}\phi(\boldsymbol{x}_i)\\ & -\boldsymbol{\omega}_{t-1}\phi(\boldsymbol{x}_i)|_2^2+\alpha\cdot|b-b_{t-1}|_2^2 \qquad (12)\\ s.t.\quad & y_i(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_i)+b)\geq 1-\xi_i;\\ & \xi_i\geq 0 \qquad i=1,\cdots,n.\end{aligned}$$

The Wolfe dual form of this problem is given in Appendix B.3, with its further solution process in Appendix A.

4.4 Evolutionary MMC(e-MMC)

Inspired by the CPMMC Algorithm [23, 24, 19], we propose our algorithm for MMC-based evolutionary clustering both efficiently and effectively. As mentioned in Section 3, we need to provide a Wolfe dual form of the transformed problem during its problem solving process. The evolutionary Maximum Margin Clustering (e-MMC) algorithm is described in Algorithm 1.

Algorithm 1 Evolutionary Maximum Margin Clustering (e-MMC)

Require: violation parameter C; balance parameter l; error ε; parameter α;historic records $\boldsymbol{\omega}_{t-1}, b_{t-1}$ or $\{\phi(\boldsymbol{x}_{i,t-1})\}_{i=1}^n$; current kernel coordinator $\{\phi(\boldsymbol{x}_{i,t})\}_{i=1}^n$ initialized $\boldsymbol{\omega}_0, b_0$; initialized constraints set $\Omega=\emptyset$

Ensure: resulted data label $\{y_{i,t}\}_{i=1}^n$,$\{y_{i,t-1}\}_{i=1}^n$

1: choose a proper framework: CDE, IDE, MI;
2: select the most violated constraint $\boldsymbol{c}$ according to [23]
3: **while** $\frac{1}{n}\sum_{i=1}^n c_i - \frac{1}{n}\sum_{i=1}^n c_i|\boldsymbol{\omega}^T\phi(\boldsymbol{x}_i)+b| > \xi+\varepsilon$ **do**
4: $\Omega=\Omega\cup\boldsymbol{c}$
5: use quadratic programming technique to iteratively solve the Wolfe Dual form of the selected framework until converged
6: update $\boldsymbol{\omega}, b$
7: select the most violated constraint $\boldsymbol{c}$
8: **end while**
9: **return** corresponding hyperplane parameter $\{\boldsymbol{\omega}, b\}$

Details of the CPMMC method can be checked in Appendix A.

5. ANALYSIS AND EXTENSION

5.1 Comparison and Analysis

Here we re-formulate the original MMC problem from the *loss function* perspective to make a better comparison of the two strategies in the *data integration* framework.

$$\min_{\boldsymbol{y}\in\{-1,+1\}^n}\min_{\boldsymbol{\omega},b}\quad \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega}+C\cdot Loss \qquad (13)$$

where $Loss$ (different from *IR* as *IR* has constraints) is a pre-defined hinge loss function in the original MMC problem.

$$Loss=\sum_{i=1}^{n}\max\{0,1-y_i(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_i)+b)\} \qquad (14)$$

One advantage of $Loss$ is that this form exists without the introduction of the object's slack variables $\{\xi_i\}_{i=1}^n$.

Our CDE strategy is a linear combination of this original *Loss* at time stamps t and $t-1$.

$$\begin{aligned}Loss=\sum_{i=1}^{n}Loss_i=\sum_{i=1}^{n}[&\alpha\cdot\max\{0,1-y_{i,t}(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_{i,t})+b)\}\\ &+(1-\alpha)\cdot\max\{0,1-y_{i,t-1}(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_{i,t-1})+b)\}]\end{aligned} \qquad (15)$$

The *Loss function* of IDE is in the form as:

$$\begin{aligned}Loss=\sum_{i=1}^{n}\max\{0,&\alpha\cdot(1-y_{i,t}(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_{i,t})+b)),\\ &(1-\alpha)\cdot(1-y_{i,t-1}(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_{i,t-1})+b))\}\end{aligned} \qquad (16)$$

From the above *Loss function* analysis, we can easily obtain a deep understanding of the different evolutionary strategies incorporating the time factor t. In CDE, the hinge loss of the data cohort in different time stamps $(t, t-1)$ is linearly combined to form the whole loss, whereas the data violation in IDE is expressed through paired data object's comparison.

In the MI framework, the *margin shift* is used to smoothen the change of the model, where the change of label assignments can be regarded as another penetration point. Therefore, the difference between the current partition and historical partition is calculated to measure the continuity and robustness trend of evolutionary clustering. The time factor influence can be defined as:

$$df=\sum_{i=1}^{n}|y_{i,t}-y_{i,t-1}|^2 \qquad (17)$$

As we know, the signum function $sgn(\cdot)$ value of *margin distance* is the label prediction result both in SVM and MMC. Thus, the difference led by various data point allocations can be calculated as:

$$df=\sum_{i=1}^{n}|\mathrm{sgn}(\boldsymbol{\omega}_t\phi(\boldsymbol{x}_i)+b_t)-\mathrm{sgn}(\boldsymbol{\omega}_{t-1}\phi(\boldsymbol{x}_i)+b_{t-1})|_2^2 \qquad (18)$$

The function $sgn(\cdot)$ is inconvenient to be calculated during the derivation. We thus approximate the solution with the following relaxation function:

$$df=\sum_{i=1}^{n}|(\boldsymbol{\omega}_t\phi(\boldsymbol{x}_i)+b_t)-(\boldsymbol{\omega}_{t-1}\phi(\boldsymbol{x}_i)+b_{t-1})|_2^2 \qquad (19)$$

Note that df is the same as *margin shift* in the second framework in Section 4, which shares the same representation under this strategy.

Based on the above analysis, it is clear that *data integration* focuses on data consistency during the evolutionary clustering procedure with the historical data; whereas the *model integration* aims at margin consistency, which is more related to the existed historical maximum margin.

5.2 Object Inserting and Removing

During the data evolutionary procedure, the inserting and removing of data objects are different natural phenomena.

In the *data integration* framework, we fix the problem through the *Loss function* expression. More specifically, inserting and removing objects are regarded as a single term in calculating the object loss, i.e., $Loss_i=\alpha\cdot\max\{0,1-y_{i,T}(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_{i,T})+b)\}$ for a coming object i and $Loss_i=(1-\alpha)\cdot\max\{0,1-y_{i,T-1}(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_{i,T-1})+b)\}$ for a disappearing object i.

Another case is the *margin consistency* framework. An average value of the whole *margin distance* is calculated for those disappearing object's *margin distance*.

6. EXPERIMENTAL EVALUATION

The experimental evaluation is conducted on three types of datasets, which are grouped into synthetic dataset, UCI-benchmarking datasets [8] and NEC blog dataset [22]. All datasets are preprocessed by normalizing each feature on each dimension into the interval [0, 1]. Furthermore, the clustering process of the algorithms is repeated

for 50 times at each setting. All experiments were run on a computer with Intel Xeno (R) CPU 2.53-GHz, Microsoft Windows 7 with algorithms coded in Matlab.

6.1 Baseline and Experimental Settings

Our proposed methods are compared with four baseline algorithms to verify our algorithm performance: k-means clustering on accumulated historical data (ACC), k-means clustering on current individual data (IND), spectral evolutionary clustering with preserving cluster quality (PCQ) and the one with preserving cluster membership (PCM).

Parameters in these algorithms are set accordingly. In ACC and IND, we use the random initialization strategy to start the clustering; Euclidean distance and the RBF function $f(d) = \exp(-\frac{d^2}{2\delta^2})$ are used to construct the similarity matrix, and parameter δ ranges from 0.1 to 2.

Moreover, to accelerate the spectral clustering calculation and to better capture the similarity matrix, we employ the recently proposed spectral clustering implementation [5]. On the k-means clustering implementation, we directly use the existed function in the Matlab implementation toolbox.

In our algorithms, unless specified, the tuning parameter α is set to be 0.7 for all comparison algorithms.

6.2 Performance Metrics

For fair comparison, we use the k-means clustering cost function km-cost to measure the clustering performance of our proposed methods and the above baseline evolutionary clustering methods.

$$km\text{-cost} = \alpha \cdot \sum_{j=1}^{c_t}\sum_{i=1}^{C_{j,t}} \|\boldsymbol{x}_{i,t} - \frac{\sum_{l=1}^{C_{j,t}} \boldsymbol{x}_{l,t}}{C_{j,t}}\|^2 + (1-\alpha)\cdot \sum_{j=1}^{c_{t-1}}\sum_{i=1}^{C_{j,t-1}} \|\boldsymbol{x}_{i,t-1} - \frac{\sum_{l=1}^{C_{j,t-1}} \boldsymbol{x}_{l,t-1}}{C_{j,t-1}}\|^2 \tag{20}$$

where $\{C_j, j = 1, \cdots, c\}$ are the resultant cluster assignments given by the comparison methods, c is the cluster number and α is a pre-defined parameter tuning the importance weight between the *snapshot cost* and *temporal cost*. Obviously, the smaller the values are, the better the clustering performance is.

Besides the km-cost criterion, we also use the NMI (normalized mutual information) to study the performance of the clustering algorithms in the synthetic data learning, which is defined as:

$$NMI = \frac{\sum_{i=1}^{c}\sum_{j=1}^{c} \log(\frac{N \cdot n_{ij}}{n_i n_j})}{\sqrt{\sum_{i=1}^{c} n_i \log(\frac{n_i}{N}) \sum_{i=1}^{c} n_j \log(\frac{n_j}{N})}} \tag{21}$$

where n_{ij} is the number of agreements between clusters i and j, n_i is the number of data points in cluster i, n_j is the number of data points in cluster j, and N represents the number of data points in the whole dataset.

6.3 Experiments on Synthetic Data

6.3.1 Synthetic Dataset

In this experiment, we first adopt a synthetic dataset to investigate the performance of our proposed methods, where the dataset is derived from the same generation algorithm as that in [6]. 1500 2-dimensional data points are initially generated as described in Figure 3, with two Guassian Distributions generating 750 data points at locations $[3, 5]$ and $[3, 1]$ respectively. We then disturb the positions of data points by adding different noises to the origin points sequentially along the time line to simulate the data evolving process. To represent each cluster's individual evolving trend, the noisy points are set to be generated by uniform distributions with different parameters in different original clusters.

Figure 3 shows the data point distribution and its evolving appearance at time points $t-1$ and t. We set the evolving direction as $\Delta_1 = (0.5, -0.5)$ for the upper cluster and $\Delta_2 = (-0.5, 0.5)$ for the lower cluster, both scaling in 0.5 unity.

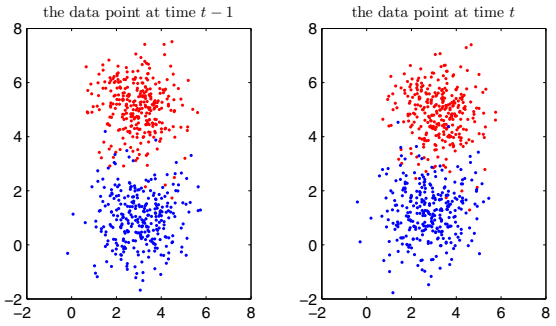

Figure 3: Data evolving

6.3.2 Clustering Result Comparison

A trial study on the synthetic evolutionary data is reported in Table 1. The synthetic data setting is provided above and we keep on adding the evolving noise to the previous data in 12 iterations. The boldface numbers in each column denote the best two values in correspondence to both km-cost and NMI value.

From Table 1, we can easily see that our e-MMC algorithms (CDE, IDE and MI) produce the smallest km-cost values during the time period from $t = 1$ to $t = 12$, comparing to that of the four baseline algorithms. For the NMI value, although PCQ and PCM are better than e-MMC algorithms in first three instances, e-MMC algorithms obtain better performance in the rest nine time stamps. Both ACC and IND cannot achieve a satisfied result in this study.

6.3.3 Evolving Range Study

We also conduct the evolving range learning. The experimental settings are almost the same as above, except the difference of the size of uniform distribution added to the data points. We set the uniform distribution size ranging from 0.1 to 1, and then observe the km-cost under these different situations. Figure 4 shows the detailed results. Since *CDE* and *IDE*'s value are approximate the same, their value lines overlap mostly.

From Figure 4, we can see that the km-cost grows with the increase of the range value. This is due to a larger diversity of the data set associated with a larger evolving range. Also, we can see that e-MMC algorithms can always obtain the smallest value from all the situations. This shows robustness of the proposed three e-MMC methods.

6.3.4 Computational Time Analysis

Figure 5 displays the running time of our methods against that of the baselines algorithms.

t scales in ms (margin shift) unity. It shows that our proposed methods CDE, IDE and MI can manipulate the data size at at least the million level. They runs much efficiently than the existed spectral evolutionary clustering, especially in large scale data. Although they are not as fast as the k-means implementations *ACC* and *IND*,

Table 1: Synthetic data experimental performance

Time	criteria	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$	$t=6$	$t=7$	$t=8$	$t=9$	$t=10$	$t=11$	$t=12$
CDE	KM-cost	**5.630**	**5.591**	**5.576**	**5.642**	**5.810**	**6.047**	**6.376**	**6.685**	**6.936**	**7.191**	**7.407**	**7.538**
	NMI	0.794	0.707	0.629	0.589	0.606	0.643	**0.714**	**0.785**	**0.854**	**0.907**	**0.956**	**0.983**
IDE	KM-cost	**5.630**	**5.590**	**5.575**	**5.639**	**5.806**	**6.046**	**6.376**	**6.685**	**6.935**	**7.191**	**7.407**	**7.538**
	NMI	0.796	0.707	0.633	**0.602**	**0.610**	**0.647**	**0.717**	**0.795**	**0.854**	**0.906**	**0.957**	**0.987**
MI	KM-cost	5.635	5.592	5.580	5.642	5.810	6.051	6.380	6.697	6.937	7.201	7.417	7.538
	NMI	0.799	0.699	0.652	**0.600**	**0.608**	**0.645**	0.713	0.790	0.851	0.896	0.957	0.987
PCQ	KM-cost	5.649	5.614	5.589	5.648	5.812	6.060	6.393	6.703	6.950	7.211	7.415	7.540
	NMI	**0.815**	**0.736**	**0.644**	0.589	0.598	0.637	0.711	0.786	0.837	0.890	0.94	0.973
PCM	KM-cost	5.653	5.627	5.595	5.666	5.833	6.075	6.419	6.742	6.963	7.223	7.431	7.544
	NMI	**0.826**	**0.757**	**0.659**	0.593	0.580	0.621	0.675	0.736	0.815	0.867	0.932	0.968
ACC	KM-cost	5.683	5.657	5.620	5.663	5.833	6.075	6.425	6.750	6.963	7.222	7.431	7.544
	NMI	0.756	0.688	0.602	0.583	0.573	0.619	0.668	0.730	0.812	0.872	0.928	0.968
IND	KM-cost	5.681	5.653	5.631	5.681	5.826	6.080	6.399	6.705	6.953	7.210	7.420	7.540
	NMI	0.754	0.662	0.586	0.573	0.616	0.629	0.703	0.742	0.812	0.892	0.928	0.979

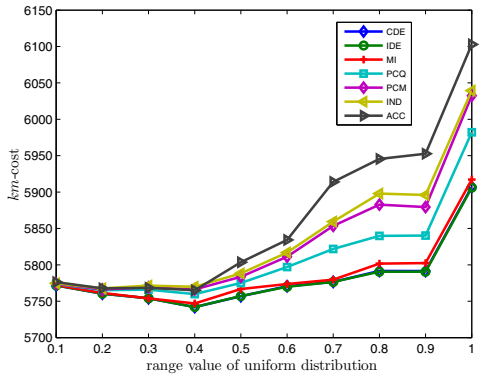

Figure 4: Uniform distribution performance

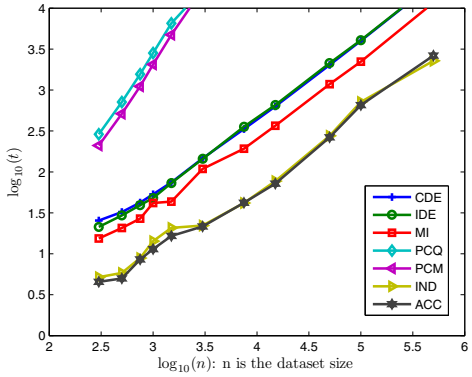

Figure 5: Running time

according to the previous experiments, our methods reach better clustering results than *ACC* and *IND*.

6.4 Experiments on Real World Data

In this section, we present outcomes on the UCI-benchmarking data sets and an NEC Blog data set.

6.4.1 UCI Data

First, we use a bunch of UCI data sets to test the performance of the three e-MMC methods CDE, IDE and MI. Each cluster in the datasets is set a fixed direction for adding the uniform distribution to simulate the evolving behavior of the data. Since the data attributes change randomly making it hard to determine each data point's belonging clusters, we use the km-cost rather than NMI to verify the performance. Table 2 displays our testing results. The boldface value in each row denotes the best in correspondence to the km-cost.

As we can see, our proposed three methods CDE, IDE and MI reach the best values in every dataset. This shows the clear advantage of the e-MMC approach compared to the baseline algorithms.

6.4.2 NEC Blogs Data

Here we further test the performance of our proposed evolutionary-MMC (e-MMC) frameworks on social network learning. We conduct experiments on a real Blog data set, which was collected by an NEC in-house blog crawler and had been used in [6, 15, 22] for evaluation. It contains 148, 681 entry-to-entry links among 407 blogs during 15 months, with a set of 303 blogs focusing on technology theme and a set of 104 blogs on politics in accordance with their contents.

Before using this Blog data set, we first pre-process the linked data by aggregating data in months 6 and 7 into the 6th time step, data in months 8-10 into the 7th time step, and data in months 11-15 into 8th time step. This is because the linked data entries reduced sharply as the data include less blogs towards the end of the time period. The links in each time step of data are shown in Table 3.

Table 3: Blog links in each time step

Time step	1	2	3	4	5	6	7	8
Link number	822	877	681	640	606	888	723	762

Figures 6, 7 and 8 depict the performance of our proposed methods CDE, IDE and MI compared to the baseline methods PCQ,

Table 2: km-cost on UCI datasets ($\times 10^2$)

	CDE	IDE	MI	PCQ	PCM	ACC	IND
$\mathbf{digits_{1,2}}$	**25.44**	25.46	25.51	25.58	25.55	25.50	25.62
$\mathbf{digits_{1,3}}$	24.07	**24.06**	24.11	24.28	24.18	24.30	24.19
$\mathbf{vowel_{1,2}}$	**1.204**	1.208	1.225	1.240	1.294	1.216	1.213
$\mathbf{satellite_{1,2}}$	40.19	**40.02**	40.39	40.46	40.46	40.44	40.41
breast	**4.989**	5.032	5.016	5.315	5.464	5.471	5.315
diabetes	**3.970**	3.971	4.012	4.200	4.213	4.222	4.202
liver	0.798	**0.797**	0.800	0.812	0.821	0.820	0.814
$\mathbf{pendigits_{0,1}}$	**21.74**	21.75	21.79	21.74	21.83	21.87	21.78

PCM, ACC and IND in terms of km-cost, *snapshot cost* and *historical cost*.

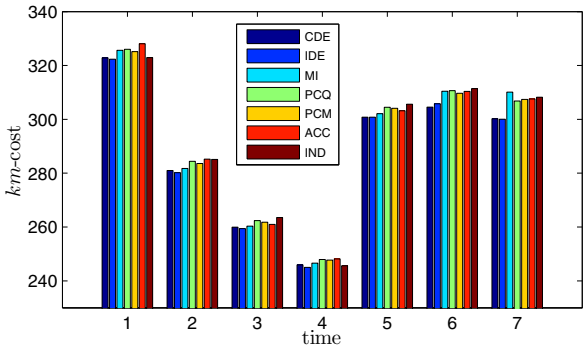

Figure 6: km-cost

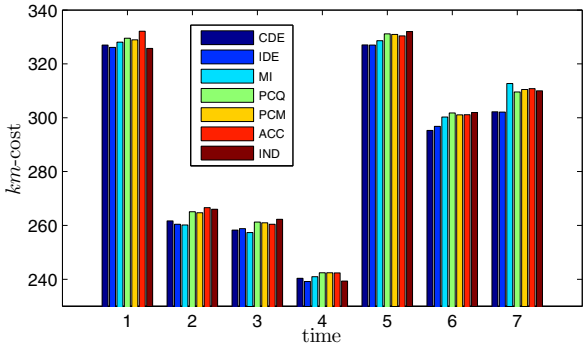

Figure 7: *Snapshot cost*

From Figures 6-8, we can see the proposed e-MMC methods always achieve better km-cost compared to the baseline algorithms. *MI* does not always perform perfectly on the time steps 6 and 7, this is due to that the outcomes are subject to the aggregation method.

7. CONCLUSIONS AND FUTURE WORK

Evolutionary data is ubiquitous in business and social applications. Maximum margin clustering (MMC) has demonstrated its power in achieving better accuracy by seeking maximum margin between two clusters. As the first work in the field, this paper has proposed the evolutionary MMC, and two corresponding frameworks, *data integration* and *margin consistency*, as well as three clustering algorithms for adapting MMC to learn unsupervised data.

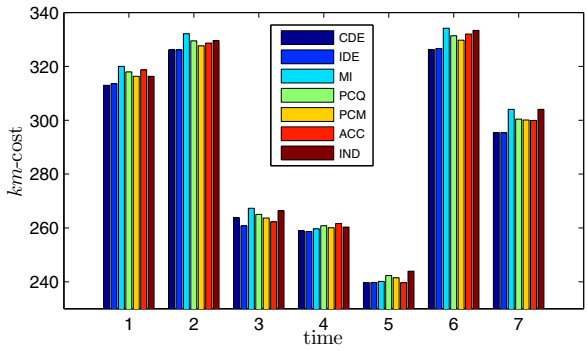

Figure 8: *Historic cost*

The design nicely incorporates time information into maximum margin learning. We have conducted substantial experiments on synthetic, UCI and real-life blog data to compare the accuracy, computation and scalability of our proposed three algorithms with four baseline algorithms. The outcomes clearly show that the proposed evolutionary MMC and the subsequent algorithms outperform the baselines in terms of achieving better accuracy, and are effective in tackling large scale of unlabeled data.

We are now working on expanding the evolutionary MMC to multiple cluster applications, towards multi-cluster maximum margin clustering algorithms on learning evolutionary data.[3][2][17]

8. ACKNOWLEDGEMENTS

This work is sponsored in part by Australian Research Council Discovery Grants (DP1096218) and ARC Linkage Grant (LP100200774). We appreciate the comments and help provided by Xiaodong Yue and Yiling Zeng.

9. REFERENCES

[1] C. Aggarwal, J. Han, J. Wang, and P. Yu. A framework for clustering evolving data streams. In *Proceedings of the 29th international conference on Very large data bases-Volume 29*, pages 81–92. VLDB Endowment, 2003.

[2] L. Cao. *Data mining for business applications*. Springer-Verlag New York Inc, 2008.

[3] L. Cao, Y. Ou, P. S. Yu, and G. Wei. Detecting abnormal coupled sequences and sequence changes in group-based manipulative trading behaviors. In *ACM SIGKDD*, KDD '10, pages 85–94, New York, NY, USA, 2010. ACM.

[4] D. Chakrabarti, R. Kumar, and A. Tomkins. Evolutionary clustering. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 554–560. ACM, 2006.

[5] W. Chen, Y. Song, H. Bai, C. Lin, and E. Chang. Parallel spectral clustering in distributed systems. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (99):1–1, 2010.

[6] Y. Chi, X. Song, D. Zhou, K. Hino, and B. Tseng. Evolutionary spectral clustering by incorporating temporal smoothness. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 153–162. ACM, 2007.

[7] Y. Chi, X. Song, D. Zhou, K. Hino, and B. Tseng. On evolutionary spectral clustering. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 3(4):17, 2009.

[8] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[9] S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan. Clustering data streams. In *Foundations of computer science, 2000. proceedings. 41st annual symposium on*, pages 359–366. IEEE, 2000.

[10] C. Gupta and R. Grossman. Genic: A single pass generalized incremental algorithm for clustering. In *Proceedings of the SIAM International Conference on Data Mining*, pages 137–153, 2004.

[11] M. Hoai and F. De la Torre. Maximum margin temporal clustering. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, 2012.

[12] J. Kelley. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712, 1960.

[13] A. LH21. The cholesky decomposition. *LINPACK: users' guide*, (8), 1979.

[14] Y. Li, J. Han, and J. Yang. Clustering moving objects. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 617–622. ACM, 2004.

[15] Y. Lin, Y. Chi, S. Zhu, H. Sundaram, and B. Tseng. Analyzing communities and their evolutions in dynamic social networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 3(2):8, 2009.

[16] H. Ning, W. Xu, Y. Chi, Y. Gong, and T. Huang. Incremental spectral clustering with application to monitoring of evolving blog communities. In *SIAM Int. Conf. on Data Mining*, pages 261–72, 2007.

[17] Y. Song, L. Cao, X. Wu, G. Wei, W. Ye, and W. Ding. Coupled behavior analysis for capturing coupling relationships in group-based market manipulations. In *ACM SIGKDD*, 2012.

[18] H. Valizadegan and R. Jin. Generalized maximum margin clustering and unsupervised kernel learning. *Advances in Neural Information Processing Systems*, 19:1417, 2007.

[19] F. Wang, B. Zhao, and C. Zhang. Linear time maximum margin clustering. *Neural Networks, IEEE Transactions on*, 21(2):319–332, 2010.

[20] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. *Advances in neural information processing systems*, 17:1537–1544, 2004.

[21] L. Xu and D. Schuurmans. Unsupervised and semi-supervised multi-class support vector machines. In *Proceedings of the 20th national conference on Artificial intelligence - Volume 2*, AAAI'05, pages 904–910. AAAI Press, 2005.

[22] T. Yang, Y. Chi, S. Zhu, Y. Gong, and R. Jin. Detecting communities and their evolutions in dynamic social networksŮa bayesian approach. *Machine learning*, 82(2):157–189, 2011.

[23] B. Zhao, F. Wang, and C. Zhang. Efficient maximum margin clustering via cutting plane algorithm. In *The 8th SIAM International Conference on Data Mining*, pages 751–762, 2008.

[24] B. Zhao, F. Wang, and C. Zhang. Efficient multiclass maximum margin clustering. In *Proceedings of the 25th international conference on Machine learning*, pages 1248–1255. ACM, 2008.

APPENDIX

A. CPMMC IMPLEMENTATION

[23][19] formulate the Maximum Margin Clustering problem as

$$\begin{aligned} \min_{\boldsymbol{\omega},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + \frac{C}{n}\sum_{i=1}^{n}\xi_i \\ s.t. \quad & |\boldsymbol{\omega}^T\phi(\boldsymbol{x}_i) + b| \geq 1 - \xi_i; \\ & \xi_i \geq 0, i = 1, \cdots, n. \end{aligned} \tag{22}$$

They proved that without the cluster-balance constraint, the solution to problem (22) is identical to problem (2) and made use of the *cutting plane* method[12] to solve the problem. However, since the problem is nonconvex with respect to $\boldsymbol{\omega}$, they first transformed it into a constraint concave convex procedure(CCCP).

One key step in the CCCP procedure is the below *quadratic programming*(QP) problem:

$$\begin{aligned} & \min_{\boldsymbol{\omega},\boldsymbol{b},\boldsymbol{\xi}} \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + C\xi \\ & s.t. \xi \geq 0 \\ & \quad -l \leq \sum_{i=1}^{n}(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_i) + b) \leq l \\ & \quad \forall \boldsymbol{c} \in \Omega : \frac{1}{n}\sum_{i=1}^{n} c_i - \xi - \frac{1}{n}\sum_{i=1}^{n} c_i \\ & \qquad \cdot \operatorname{sign}(\boldsymbol{\omega}_t^T\phi(\boldsymbol{x}_i) + b_t)[\boldsymbol{\omega}^T\phi(\boldsymbol{x}_i) + b] \leq 0 \end{aligned} \tag{23}$$

This problem could be solved in polynomial time, the Wolfe dual of problem is introduced to solve the problem more efficiently.

$$\begin{aligned} \max_{\lambda\geq 0,\mu\geq 0} & -\frac{1}{2}\sum_{k=1}^{|\Omega|}\sum_{l=1}^{|\Omega|}\lambda_k\lambda_l\boldsymbol{z}_k^T\boldsymbol{z}_l + (\mu_1 - \mu_2)\sum_{k=1}^{||\Omega|}\lambda_k\hat{\boldsymbol{x}}^T\boldsymbol{z}_k \\ & -\frac{1}{2}(\mu_1 - \mu_2)^2\hat{\boldsymbol{x}}^T\hat{\boldsymbol{x}} - (\mu_1 + \mu_2)l + \sum_{k=1}^{|\Omega|}\lambda_k\|\boldsymbol{c}_k\|_1 \\ s.t. & \sum_{k=1}^{|\Omega|}\lambda_k \leq C; \\ & (\mu_1 - \mu_2)n - \sum_{k=1}^{|\Omega|}\frac{\lambda_k}{n}c_{ki} \cdot \operatorname{sign}(\boldsymbol{\omega}_t^T\phi(\boldsymbol{x}_i) + b_t) = 0. \end{aligned} \tag{24}$$

Here the definition of $\|\boldsymbol{c}_k\|_1, \boldsymbol{z}_k, \hat{\boldsymbol{x}}$ is the following for simplic-

ity of the problem statement:

$$\begin{aligned}
&\|\boldsymbol{c}_k^1\|_1 = \frac{1}{n}\sum_{i=1}^{n} c_{ki} \\
&\boldsymbol{z}_k^1 = \frac{1}{n}\sum_{i=1}^{n} c_{ki}\text{sign}(\boldsymbol{\omega}_t^T\phi(\boldsymbol{x}_i)+b_t)\phi(\boldsymbol{x}_i); \\
&\hat{\boldsymbol{x}} = \sum_{i=1}^{n}\phi(\boldsymbol{x}_i).
\end{aligned} \tag{25}$$

The above dual form of the (QP) problem solving will continue until $\{\lambda_i\}_{i=1}^n, \mu$ converge. Details of the notations and algorithm solving could be checked in the original paper[23].

Attached is the detail CPMMC Algorithm.

Algorithm 2 Cutting Plane Maximum Margin Clustering[23]

Require: violation parameter C;
balance parameter l; initialized $\boldsymbol{\omega}_0, b_0$;error ε;
initial constraints set $\Omega = \emptyset$.
Ensure: resulted data label $\{y_i\}_{i=1}^n$
1: select the most violated constraint $\boldsymbol{c}$
2: **while** $\frac{1}{n}\sum_{i=1}^n c_i - \frac{1}{n}\sum_{i=1}^n c_i|\boldsymbol{\omega}^T\phi(\boldsymbol{x}_i)+b| > \xi+\varepsilon$ **do**
3: $\Omega = \Omega \cup \boldsymbol{c}$
4: use quadratic programming to iteratively solve the Wolfe Dual form (Eq. (24)) of CCCP problem until converged
5: update $\boldsymbol{\omega}, b, \xi$
6: select the most violated constraint $\boldsymbol{c}$
7: **end while**
8: **return** corresponding hyperplane parameter $\{\boldsymbol{\omega}, b\}$

B. VARIOUS WOLFE DUAL FORMS

Appendixes B.1-B.3 are the basic Wolfe dual forms required in [23] to employ the Cutting Plane Approximation, in correspondence to each proposed objective function.

B.1 Cohort Data Evolving

Under the *cohort data evolving* method, the problem can be transformed into the following Wolfe dual form:

$$\begin{aligned}
\max_{\lambda^1\geq 0,\lambda^2\geq 0,\mu\geq 0} & -\frac{1}{2}\sum_{k=1}^{|\Omega|}\sum_{m=1}^{|\Omega|}(\lambda_k^1\lambda_m^1\boldsymbol{z}_k^{1T}\boldsymbol{z}_m^1 + 2\lambda_k^1\lambda_m^2\boldsymbol{z}_k^{1T}\boldsymbol{z}_m^2 \\
&+ \lambda_k^2\lambda_m^2\boldsymbol{z}_k^{2T}\boldsymbol{z}_m^2) + (\mu_1-\mu_2)\sum_{k=1}^{|\Omega|}\hat{\boldsymbol{x}}^T(\lambda_k^1\boldsymbol{z}_k^1+\lambda_k^2\boldsymbol{z}_k^2) \\
&- \frac{1}{2}(\mu_1-\mu_2)^2\hat{\boldsymbol{x}}^T\hat{\boldsymbol{x}} - (\mu_1+\mu_2)l \\
&+ \sum_{k=1}^{|\Omega|}(\lambda_k^1\|\boldsymbol{c}_k^1\|_1 + \lambda_k^2\|\boldsymbol{c}_k^2\|_1) \\
s.t. & \sum_{k=1}^{|\Omega|}\lambda_k^1 \leq \alpha\cdot C; \sum_{k=1}^{|\Omega|}\lambda_k^2 \leq (1-\alpha)\cdot C; \\
& (\mu_1-\mu_2)(n_1+n_2) - \sum_{k=1}^{|\Omega|}\frac{\lambda_k^1}{n_1}\sum_{i=1}^{n_1} c_{ki}\cdot\text{sign}(\boldsymbol{\omega}_t^T\phi(\boldsymbol{x}_i) \\
& + b_t) - \sum_{k=1}^{|\Omega|}\frac{\lambda_k^2}{n_2}\sum_{i=n_1+1}^{n_1+n_2} c_{ki}\cdot\text{sign}(\boldsymbol{\omega}_t^T\phi(\boldsymbol{x}_i)+b_t) = 0.
\end{aligned} \tag{26}$$

Here the definitions of $\|\boldsymbol{c}_k^1\|_1, \|\boldsymbol{c}_k^2\|_1, \boldsymbol{z}_k^1, \boldsymbol{z}_k^1, \hat{\boldsymbol{x}}$ are below, for the simplicity of the problem statement:

$$\begin{aligned}
&\|\boldsymbol{c}_k^1\|_1 = \frac{1}{n_1}\sum_{i=1}^{n_1} c_{ki}; \|\boldsymbol{c}_k^2\|_1 = \frac{1}{n_2}\sum_{i=n_1+1}^{n_1+n_2} c_{ki}; \\
&\boldsymbol{z}_k^1 = \frac{1}{n_1}\sum_{i=1}^{n_1} c_{ki}\text{sign}(\boldsymbol{\omega}_t^T\phi(\boldsymbol{x}_i)+b_t)\phi(\boldsymbol{x}_i); \\
&\boldsymbol{z}_k^2 = \frac{1}{n_2}\sum_{i=n_1+1}^{n_1+n_2} c_{ki}\text{sign}(\boldsymbol{\omega}_t^T\phi(\boldsymbol{x}_i)+b_t)\phi(\boldsymbol{x}_i), k=1,\ldots,|\Omega|; \\
&\hat{\boldsymbol{x}} = \sum_{i=1}^{n_1+n_2}\phi(\boldsymbol{x}_i).
\end{aligned} \tag{27}$$

B.2 Individual Data Evolving

Under the *individual data evolving* method, the Wolfe Dual form is represented as:

$$\begin{aligned}
\max_{\lambda\geq 0,\mu\geq 0} & \frac{1}{1+2\beta}[-\frac{1}{2}\sum_{k=1}^{|\Omega|}\sum_{l=1}^{|\Omega|}\lambda_k\lambda_l\boldsymbol{z}_k^T\boldsymbol{z}_l + (\mu_1-\mu_2)\sum_{k=1}^{||\Omega|}\lambda_k\hat{\boldsymbol{x}}^T\boldsymbol{z}_k \\
& - \frac{1}{2}(\mu_1-\mu_2)^2\hat{\boldsymbol{x}}^T\hat{\boldsymbol{x}}] - (\mu_1+\mu_2)l + \sum_{k=1}^{|\Omega|}\lambda_k\|\boldsymbol{c}_k\|_1 \\
s.t. & \sum_{k=1}^{|\Omega|}\lambda_k \leq C; \\
& (\mu_1-\mu_2)n - \sum_{k=1}^{|\Omega|}\frac{\lambda_k}{n}c_{ki}\cdot\text{sign}(\boldsymbol{\omega}_t^T\phi(\boldsymbol{x}_i)+b_t) = 0.
\end{aligned} \tag{28}$$

where $\boldsymbol{z}_k, \hat{\boldsymbol{x}}, \|\boldsymbol{c}_k\|_1$ are the same as in Equation (25).

B.3 Model Integration

Due to the computational complexity, we take an approximation of the objective function and select the $\boldsymbol{\omega}$-penalty term into consideration. After the similar transformation as above, we get the following Wolfe dual formulation:

$$\begin{aligned}
\max_{\lambda\geq 0,\mu\geq 0} & \frac{1}{1+2\beta}[-\frac{1}{2}\sum_{k=1}^{|\Omega|}\sum_{l=1}^{|\Omega|}\lambda_k\lambda_l\boldsymbol{z}_k^T\boldsymbol{z}_l + (\mu_1-\mu_2)\sum_{k=1}^{||\Omega|}\lambda_k\hat{\boldsymbol{x}}^T\boldsymbol{z}_k \\
& - \frac{1}{2}(\mu_1-\mu_2)^2\hat{\boldsymbol{x}}^T\hat{\boldsymbol{x}}] + (\frac{2\beta}{1+2\beta}\boldsymbol{\omega}_{T-1}^T\hat{\boldsymbol{x}} - l)\mu_1 \\
& - (\frac{2\beta}{1+2\beta}\boldsymbol{\omega}_{T-1}^T\hat{\boldsymbol{x}} + l)\mu_2 - \sum_{k=1}^{|\Omega|}\lambda_k(\frac{2\beta}{1+2\beta}\boldsymbol{\omega}_{T-1}^T\boldsymbol{z}_k - \|\boldsymbol{c}_k\|_1) \\
s.t. & \sum_{k=1}^{|\Omega|}\lambda_k \leq C; \\
& (\mu_1-\mu_2)n - \sum_{k=1}^{|\Omega|}\frac{\lambda_k}{n}c_{ki}\cdot\text{sign}(\boldsymbol{\omega}_t^T\phi(\boldsymbol{x}_i)+b_t) = 0.
\end{aligned} \tag{29}$$

where $\boldsymbol{z}_k, \hat{\boldsymbol{x}}, \|\boldsymbol{c}_k\|_1$ are the same as in Equation (25)), and β is defined as:

$$\beta = \alpha\cdot\sum_{i=1}^{n}\|\phi(\boldsymbol{x}_i)\|_2^2 \tag{30}$$