World Automation Congress © 2010 TSI Press.

AN INTERACTIVE TOOL FOR CONSTRAINED CLUSTERING

MASAYUKI OKABE AND SEIJI YAMADA
Toyohashi University of Technology, Tempaku, Aichi, Japan
National Institute of Informatics, Chiyoda, Tokyo, Japan

ABSTRACT—This paper describes an interactive tool for constrained clustering that helps users to select effective constraints efficiently during the clustering process. This tool has some functions such as 2-dimensional visual arrangement of a data set and constraint assignment by mouse manipulation. Moreover, it can execute distance metric learning and k-medoids clustering. In the paper, we show the overview of the tool and how it works, especially in the functions of display arrangement by multi-dimensional scaling and incremental distance metric learning.

Keywords: visual interface, constrained clustering

1. INTRODUCTION

Constrained clustering is a promising approach for improving the accuracy of clustering by using some prior knowledge about data. As the prior knowledge, we generally use two types of simple constraints about a pair of data. The first constraint is called “must-link”. It is a pair of data that must be in the same cluster. The second one is called “cannot-link”. It is a pair of data that cannot be in the same cluster. There have been proposed several approaches to utilize these constraints so far. For example, a well-known constrained clustering algorithm the COP-Kmeans proposed by Wagstaff [1] uses these constraints as exceptional rules for the data allocation process in the k-means algorithm. A data may not be allocated to the nearest cluster center if the data and a member of the cluster is a pair of cannot-link, or the data and a member of other cluster is a pair of must-link. Another researches [2,3,4] are based on the supervised metric learning that utilize the constraints to modify an original distance (or similarity, kernel) matrix to satisfy the target distance (or value) of each constraint. Hybrid method [5] is also proposed.

Although the use of constraints is an effective approach, we have some problems in preparing constraints. One problem is the efficiency of the process. Because users are generally negative to the manual operation, i.e. labeling data pairs as “must-link” or “cannot-link”, we need a system to help users cut down the operation cost. The other problem is the effectiveness of the prepared constraints. Many experimental results in recent researches show clustering performance does not monotonically improve (sometimes decreases) as the number of applied constraints increases. The degree of performance improvement relies on a set of constraints, namely the quality of constraints. These results indicate that constraints are not all useful, some are effective but some are not effective or even harmful to the clustering. We need a system to help users select only effective constraints that improve the clustering performance. The second problem is much related to the active learning that has not been researched yet so far.

We approach these problems by developing an interactive tool that helps users to select effective constraints efficiently during the clustering process. The main objectives of the tool can be sum up as follows.

1. To provide an interactive environment in which users can visually recognize the proximity of data, and give constraints easily by mouse manipulation.
2. To provide hints for the better selection strategies through the interaction process between the system and users.

Besides 2-dimensional visual arrangement of a data set and constraint assignment function, our prototype tool has distance metric learning and k-medoids clustering that can be executed as the background process. Using these functions, users can compare the results of clustering before and after constraints addition. We consider such interactions help to provide hints for the better selection strategies.

In the following sections, we first explain the overview of our tool in Section 2. Then we describe two main functions of the tool - display arrangement by multi-dimensional scaling and incremental distance metric learning, in Section 3 and 4. Finally we conclude in Section5.

2. OVERVIEW OF THE SYSTEM

In this section, we explain the process of interactive constrained clustering with our proposed tool. Figure I shows the user interface (UI) of the tool, which consists of some buttons and display area for data distribution. Each data is represented by a colored circle in the figure. Users can interactively select additional constraints and reflect them to update the clustering result through the UI. We briefly describe the interaction process between a user and our tool.

1. A user loads a data set to be clustered to our tool. Each data must be represented by a feature vector with pre-defined format. The tool calculates the initial distance matrix from the feature vector.
2. The tool runs modules of clustering (k-medoids) and multi-dimensional scaling (MDS) [6] to get the temporal clustering result and coordinates of the data set to display on the UI. We explain the details of MDS in the next section. Then the tool displays and colors the data on the UI according to the coordinates calculated by MDS and temporal clustering result.
3. If the user does not satisfy the clustering result, he/she can add constraints. The tool updates the distance matrix using additional constraints. We describe the details of the update procedure in Section 4.
4. Repeat step no.2 and no.3 until the user gets a satisfactory clustering result.

Users can select a pair of data to assign a constraint by clicking colored circles. In Figure I, two red colored data with bold black circle are selected data. After selecting data, users can assign a constraint to it by clicking “must” or “cannot” button. Then they update distance matrix and re-clustering by clicking “update” button. We use the k-medoids algorithm for the clustering process. Since we only update the distance matrix of a data set (do not calculate each data vector from the modified distance matrix), we cannot use the normal k-means algorithm that uses ad-hoc centroids. In k-medoids, k representative data is called medoids. They are used substitutes for centroids in k-means.

Most of the researches in constrained clustering use labeled data sets in their experiments and prepare constraints at random. However, it is clear that random selection is much wasteful when human must determine the labels of constraints. Our tool helps to reduce the labeling cost. In addition, we have selection bias for the constraints because we can recognize the proximity relation between data. This functionality may help users find better selection strategies.

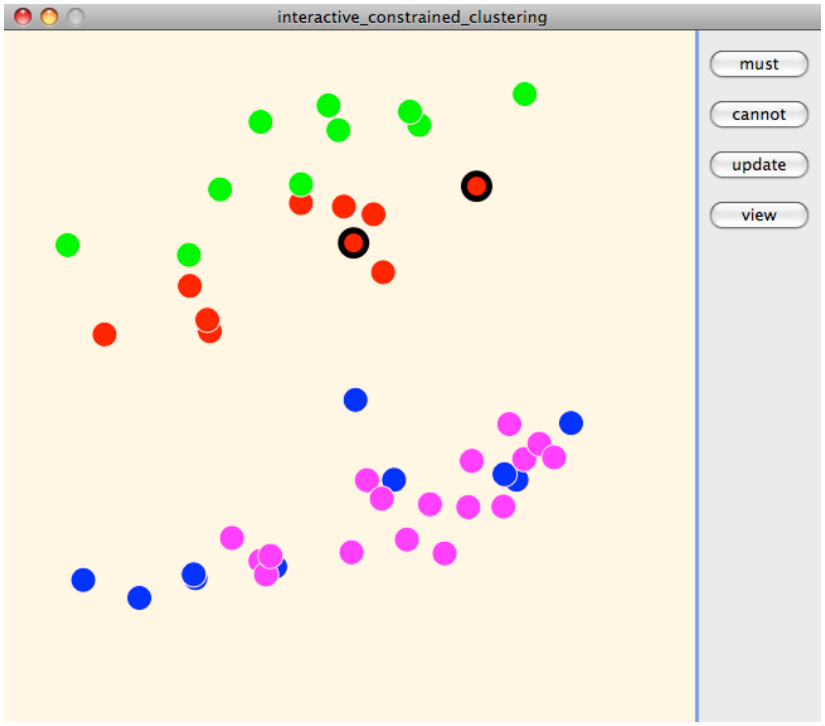


Figure I. User Interface of the Tool

3. DATA ARRANGEMENT BY MULTI-DIMENSIONAL SCALING

In this section, we describe the method of data arrangement. When we apply clustering to a data set, we generally use a multi-dimensional feature vector to represent a data that cannot be displayed in our 2-dimensional user interface. We need to display the proximity relationship that reflects the relation in the original space. We use multi-dimensional scaling (MDS) [6] to realize such function in our tool. MDS is a well-known technique that calculates spatial arrangement from a distance (or similarity) matrix of a data set. We do not need to care about the dimension of the feature vector. We do not need row data but only a distance matrix. This is very suitable for our environment as we describe in later sections. We describe a brief introduction of MDS in the following.

MDS is based on the eigen-decomposition that is a factorization technique of a square matrix. Let S is a square matrix and v is an eigen vector of S.

$$Sv = \lambda v$$

λ is an eigen value corresponding to v. Then S can be factorized as

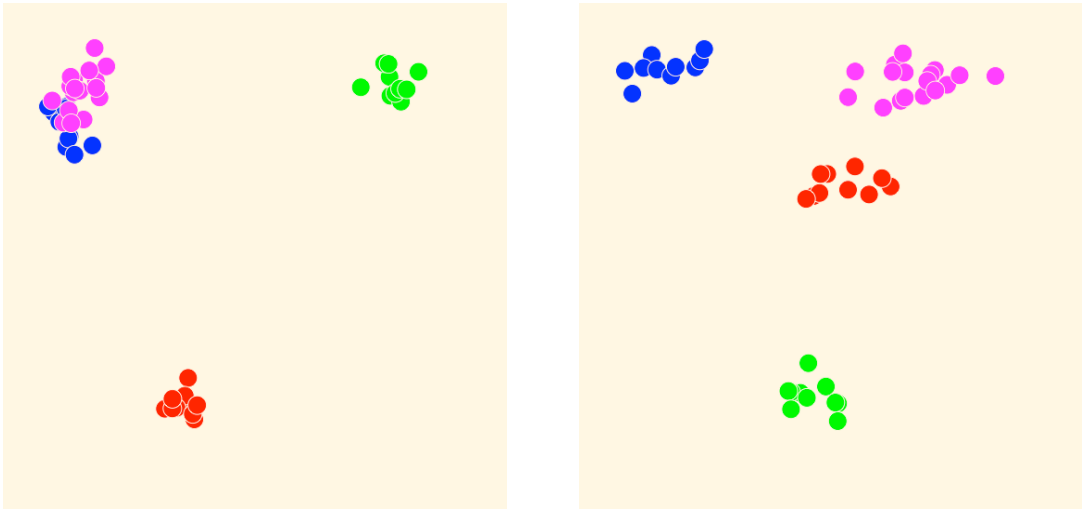
$$S = V\Lambda V^{-1}$$

where V is the square matrix whose columns are eigen vectors of S. Since we calculate S from a symmetric distance matrix D, S is also symmetric. Thus S can be factorized as

$$\begin{aligned} S &= V\Lambda V^T \\ &= V\Lambda^{1/2}\Lambda^{1/2}V^T \\ &= V\Lambda^{1/2}(V\Lambda^{1/2})^T \end{aligned}$$

The row of $V\Lambda^{1/2}$ is the coordinate of each data in MDS. Though we need only 2-dimensional coordinate, we often need more dimensions to display a data set that has potentially more than 3 clusters. Thus we calculate n-dimensional coordinate and display data using arbitrary combinations of two axes. Figure II(a) and Figure II(b) shows the examples of such combinations. We used the "soybean-small" data set from UCI repository [7] to make Figure II. Two clusters (purple and blue) are overlapping in Figure II(a), but are separated in Figure II(b).

S can be calculated from D that we repeatedly update though the interaction with our tool. Using centering matrix G_n, S is calculated as



(a) Combination No.1 **(b) Combination No.2**

Figure II. Examples of Data Arrangement (two combinations of two axises)

$$S = -\frac{1}{2} G_n D G_n^T ,$$

$$G_n = I_n - \frac{1}{n} 1_n 1_n^t$$

The centering matrix can remove the effect of the original point.

4. INCREMENTAL DISTANCE MATRIX LEARNING AND DEMONSTRATIONS

In this section, we describe an algorithm of distance learning adopted in our tool. This algorithm is proposed by Jain et al. [8] and is based on the framework of online learning. It repeatedly updates the distance matrix using constraints given through the interaction process described in Section 2. It requires less computational cost than other techniques that need some optimization procedures. This advantage is very desirable for our tool that needs quick reaction to users.

The algorithm is based on the problem of learning a Mahalanobis distance function. Given *n*-dimensional vectors ***u*** and ***v***, the squared Mahalanobis distance between them is defined as

$$d_A(u,v) = (u-v)^T A(u-v)$$

where $\boldsymbol{A}$ is initially the unit matrix and thus $d_A(u,v)$ is initially the Euclid distance between feature vectors. The objective of the learning is to get a semi-definite matrix $\boldsymbol{A}$ that produces desirable $d_A(u,v)$ for the constrained data pairs. Jain et al. considered to update $d_A(u,v)$ incrementally and proposed an online algorithm that receives one constraint at a time. We briefly describe how they introduce the update formula.

Let A_t be the distance matrix that is updated at *t*-th step, and (u_t, v_t, y_t) be a constrained data pair given at that time. Here y_t is the target distance that $d_A(u,v)$ must satisfy. If the data pair is must-link, y_t is 0. If cannot-link, y_t is 1. Jain et al. formalize an online learning problem to solve A_{t+1} by introducing a regularization function $D(A, A_t)$ and a loss function $l(d_A(u_t, v_t), y_t)$.

$$A_{t+1} = \arg\min_{A \succ 0} \{ D(A, A_t) + \eta l(d_A(u_t, v_t), y_t) \}$$

$$D(A, A_t) = tr(AA_t^{-1}) - \log\det(AA_t^{-1}) - d$$

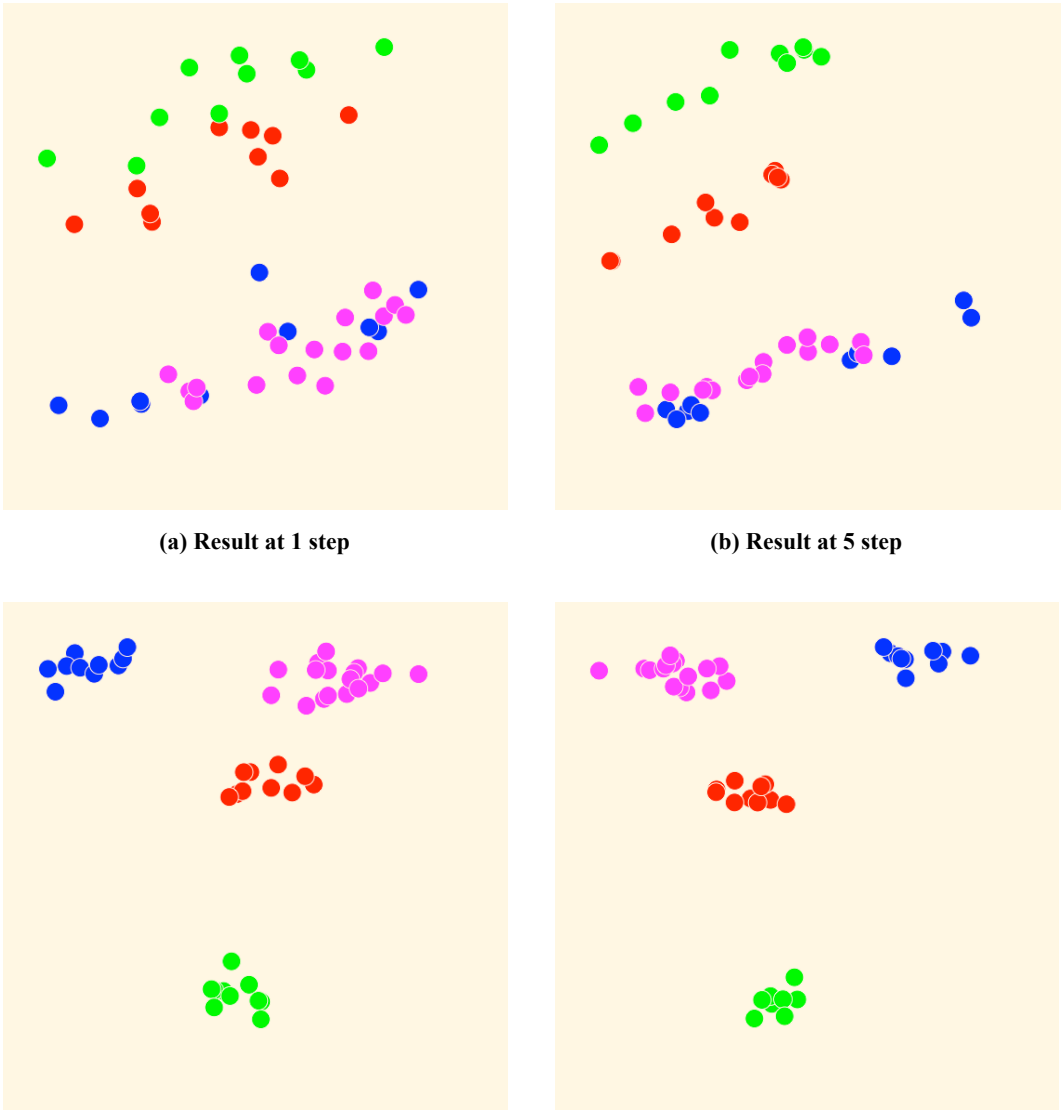
$$l(d_A(u_t, v_t), y_t) = (d_A(u_t, v_t) - y_t)^2$$

η is the regularization parameter that determines the degree of the influence of the constraint. In order to derive update formula analytically, $d_A(u_t, v_t)$ is approximated by $d_{A_t}(u_t, v_t)$. Though we omit the details of the introduction process, the update distance matrix can be solved analytically.

$$d_{A_{t+1}}(u_t, v_t) = \frac{\eta y_t \hat{y}_t - 1 + \sqrt{(\eta y_t \hat{y}_t - 1)^2 + 4\eta \hat{y}_t^2}}{2\eta \hat{y}_t}$$

$$\hat{y} = d_{A_t}(u_t, v_t)$$

Our tool can incrementally change the clustering result that is based on the distance matrix updated by the above formula. Figure III shows a series of cluster changes achieved by this incremental algorithm. The data set used in the Figure III is also the “soybean-small”. Two clusters (green and red) is slightly overlapping in Figure III(a), but are clearly separated in Figure III(b). Relationship between two clusters (purple and blue) also changes from Figure III(b) to Figure III(c). The clusters in Figure III(d) seems to be more condensed than Figure III(c). We can see the clusters are gradually separated as the distance learning proceeds.



(a) Result at 1 step **(b) Result at 5 step**

(c) Result at 10 step **(d) Result at 15 step**

Figure III. Results of Incremental Distance Learning

5. CONCLUSIONS

This paper presents an interactive tool for constrained clustering that provides some basic functions such as the display of 2-dimensional visual arrangement of a data set, constraint assignment by mouse manipulation, incremental learning of distance matrix and clustering by k-medoids. These functions helps users intervene the process of constrained clustering and finally get the satisfied clustering result with less work than the clustering process that uses randomly selected constraints. In addition, selection bias of the constraints may help users find better selection strategies.

The tool described in this paper is still a prototype. We have much work to do. For example, displaying data information is very important function because users determine the labels of constraints based on the information. However methods to display depends on the data type. We have to implement different methods when displaying images and news articles. We also consider implementing the function of active

learning that is important but rarely explored research area. The function may help users, or users may notice the drawback of the active learning algorithm.

REFERENCES

1. K. Wagstaff and S. Roger., "Constrained K-means Clustering with Background Knowledge," ICML'01, 2001, pp. 577-584.
2. D. Klein and et al., "From Instance-level Constraints to Space-level Constraints," ICML'02, 2002, pp. 307-314.
3. S. Hoi and et al., "Learning Nonparametric Kernel Matrices from Pairwise onstraints," ICML'07, 2007, pp. 361-368.
4. Z. Li and et al., "Pairwise Constraint Propagation by Semi-definite Programming for Semi-supervised Classification," ICML'08, 2008, pp. 576-583.
5. S. Basu and et al., "A probabilistic Framework for Semi-Supervised Clustering," KDD'04, 2004, pp. 59-68.
6. I. Borg and P. Groenen, "Modern multidimensional scaling," Theory and applications, 1997.
7. A. Asunction and D. J. Newman, UCI Machine Learning Repository[1]. Irvine, CA: University of California, School of Information and Computer Science, 2007.
8. P. Jain and et al., "Online metric learning and fast similarity search," NIPS'08, 2008, pp. 761-768.

[1] [http://www.ics.uci.edu/~mlearn/MLRepository.html]