2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology

An Interactive Tool for Constrained Clustering with Human Sampling

Masayuki Okabe
Toyohashi University of Technology
Tenpaku 1-1, Toyohashi, Aichi, Japan
okabe@imc.tut.ac.jp

Seiji Yamada
National Institute of Infomatics
Chiyoda, Tokyo, Japan
seiji@nii.ac.jp

***Abstract*—This paper describes an interactive tool for constrained clustering that helps users to select effective constraints efficiently during the constrained clustering process. This tool has some functions such as 2-D visual arrangement of a data set and constraint assignment by mouse manipulation. Moreover, it can execute distance metric learning and k-medoids clustering. In this 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. Eventually we show a preliminary experiment in which human heuristics found through our GUI improve the clustering. This study provides fundamental technologies for interactive clustering of Web page and Web usages.**

I. 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" which is a pair of data that must be in the same cluster. The second one is called "cannot-link" which is a pair of data that must be in different clusters. There have been proposed several approaches to utilize these constraints so far. For example, a well-known constrained clustering algorithm the COP-Kmeans [1] uses these constraints as exceptional rules for the data allocation process in a k-means algorithm. A data may not be allocated to the nearest cluster center if the data and a member of the cluster form a cannot-link, or the data and a member of the other cluster form a must-link. Another studies [2], [3], [4] are based on supervised metric learning that utilizes the constraints to modify an original distance (or called "similarity", "kernel") matrix to satisfy the target distance or value of each constraint. Also hybrid method [5] is 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 a human user generally needs to label many constraints with "must-link" or "cannot-link", his/her cognitive cost seems very high. Thus we need an interactive system to help users cut down such an operation cost. The other problem is the effectiveness of the prepared constraints. Many experimental results in recent studies have shown clustering performance does not monotonically improve (sometimes deteriorates) as the number of applied constraints increases. The degree of performance improvement relies on the quality of constraints, not the amount. These results imply that constraints are not all useful, some are effective but some are not effective or even harmful to the clustering. We also need an interactive system to help users select only effective constraints that improve the clustering performance. The second problem is much related to active learning that has not been researched much in the field of constrained clustering so far.

There have been various studies on Web page clustering and Web usages clustering [6]. We consider this work for interactive clustering is one of promising approaches to be applied to such large-scaled clustering. Thus we think this work will provide fundamental technologies for Web clustering.

We approach these problems by developing an interactive tool that helps users to select effective constraints efficiently during clustering process. The main objectives to build the interactive 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 interactive system and users.

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

II. Overview of the System

In this section, we explain the process of interactive constrained clustering with our proposed tool. Figure 1 shows GUI (Graphical User Interface) of the tool, which consists of some buttons and a 2-D display area to visualize data distribution. Each data is represented by a colored circle in the 2-D display area. Users can interactively select additional constraints and reflect them to update the clustering result through the GUI. We briefly describe the interaction process between a user and our tool in the following.

978-0-7695-4191-4/10 $26.00 © 2010 IEEE
DOI 10.1109/WI-IAT.2010.274

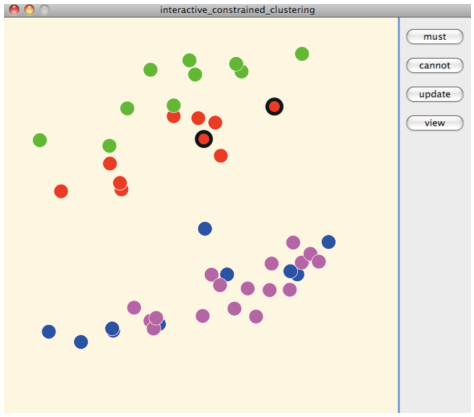
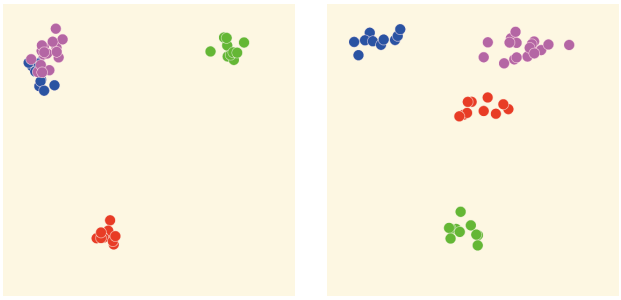


Figure 1. Graphical User Interface

1) A user loads a data set to be clustered to our tool. Each data must be represented by a feature vector with predefined format. The tool calculates the initial distance matrix from the feature vectors.
2) The tool runs modules of clustering (k-medoids) and multi-dimensional scaling (MDS) [7] to get the temporal clustering result and coordinates of the data set to display on the GUI. We explain the details of MDS in the next section. Then the tool displays and colors the data on the GUI according to the 2-D coordinates calculated by MDS and temporal clustering result.
3) If a user does not satisfy the clustering results, he/she can add constraints. The tool updates the distance matrix according to additional constraints. We describe the details of this update procedure in Section 4.
4) Repeat step 2 and step 3 until the user satisfies the clustering results.

Users can select a pair of data to assign a constraint by clicking colored circles. In Figure 1, 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 a k-medoids algorithm for a clustering process. Since we only update the distance matrix of a data set, not calculate each data vector from the modified distance matrix, we cannot use a normal k-means algorithm that uses ad-hoc centroids. In k-medoids, k representative data is called medoids and they are used substitutes for centroids in k-means.

Most of the studies in constrained clustering use labeled data sets in their experiments and prepare constraints at random. However, it is clear that random selection is very wasteful for a human user when he/she needs to determine many labels. Our tool reduces such 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 to find better selection strategies.



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

Figure 2. Examples of Data Arrangement (two combinations of two axises)

III. Data Arrangement by Multi-Dimensional Scaling

In this section, we describe a method of data arrangement. When we apply clustering to a data set, we generally use a high-dimensional feature vector to represent a data that cannot be displayed in our 2-D GUI. We need to display proximity relationships that reflects relations in the original space. We use multi-dimensional scaling (MDS) [7] to realize such 2-D visualization in our tool. MDS is a well-known technique that calculates spatial arrangement from a distance (or similarity) matrix of a data set. It does not need to care about the dimension of the feature vector, and also does not need row data but only a distance matrix. These advantages of MDS are 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 $\mathbf{v}$ is an eigen vector of S.

$$S\mathbf{v} = \lambda\mathbf{v}$$

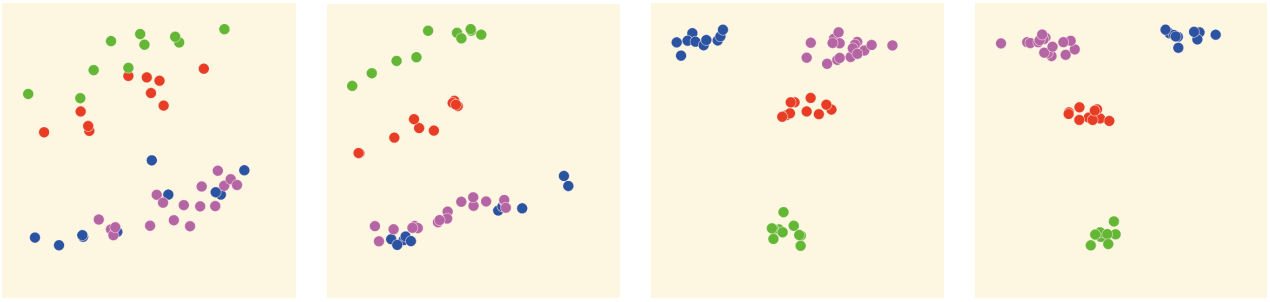
λ is an eigen value corresponding to $\mathbf{v}$. Then S can be factorized as

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

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

$$\begin{aligned} S &= V\Lambda V^T && (1)\\ &= V\Lambda^{1/2}\Lambda^{1/2}V^T && (2)\\ &= V\Lambda^{1/2}(V\Lambda^{1/2})^T && (3) \end{aligned}$$

The row of $V\Lambda^{1/2}$ is the coordinate of each data in MDS. Though we need only 2-D coordinate, we often need more dimensions to display a data set that has potentially more than three clusters. Thus we calculate n-D coordinate and display data based on arbitrary combinations of two axes. Figure 2(a) and Figure 2(b) shows the examples of such



(a) Result at 1 step (b) Result at 5 step (c) Result at 10 step (d) Result at 15 step

Figure 3. Results of Incremental Distance Learning

combinations. We used a "soybean-small" data set from UCI repository [8] for Figure 2. Two clusters (purple and blue) are overlapping in Figure 2(a), but are separated in Figure 2(b) because the pairs of axes are different.

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

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

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

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

IV. Algorithm for Incremental Learning of Distance Matrix

In this section, we describe an algorithm of distance learning adopted in our tool. This algorithm is proposed by Jain et al. [9] and is based on a framework of online learning. It repeatedly updates the distance matrix according to constraints given through an interaction process described in Section 2. It requires less computational cost than other constrained clustering techniques that need some optimization procedures. This advantage is very desirable for our tool because it needs quick responses in indicating updated results with given constraints to users.

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

$$d_A(\mathbf{u}, \mathbf{v}) = (\mathbf{u} - \mathbf{v})^T A (\mathbf{u} - \mathbf{v})$$

where A is initially the unit matrix and thus $d_A(\mathbf{u}, \mathbf{v})$ is initially the Euclid distance between feature vectors. The objective of the learning is to get a semi-definite matrix A that produces desirable $d_A(\mathbf{u}, \mathbf{v})$ for the constrained data pairs. Jain et al. considered a method to update incrementally $d_A(\mathbf{u}, \mathbf{v})$ and proposed an online algorithm that receives one constraint at a time [9]. We briefly describe how they introduced the update formula.

Let A_t be the distance matrix that is updated at t-th step, and $(\mathbf{u}_t, \mathbf{v}_t, y_t)$ be a constrained data pair given at that time. Here y_t is the target distance that $d_A(\mathbf{u}, \mathbf{v})$ must satisfy. If the data pair is must-link, y_t is 0. If it is a 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 $l(d_A(\mathbf{u}_t, \mathbf{v}_t), y_t)$ like the following.

$$A_{t+1} = \arg\min_{A \succ 0} D(A, A_t) + \eta l(d_A(\mathbf{u}_t, \mathbf{v}_t), y_t)$$

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

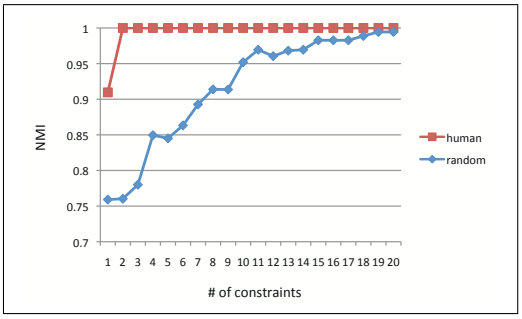
$$l(d_A(\mathbf{u}_t, \mathbf{v}_t), y_t) = (d_A(\mathbf{u}_t, \mathbf{v}_t) - y_t)^2$$

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

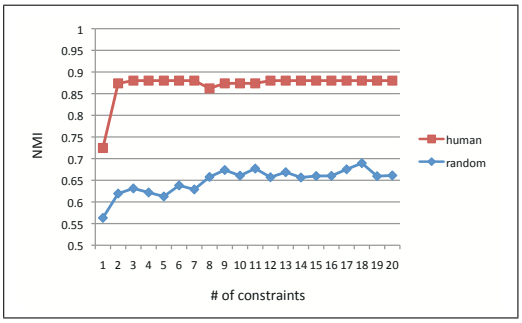
$$d_{A_{t+1}}(\mathbf{u}_t, \mathbf{v}_t) = \frac{\eta y_t \hat{y}_t - 1 + \sqrt{(\eta y_t \hat{y}_t - 1)^2 + 4\eta \hat{y}_t^2}}{2\eta \hat{y}_t}$$

$$\hat{y}_t = d_{A_t}(\mathbf{u}_t, \mathbf{v}_t)$$

Our tool can incrementally change the clustering result based on the distance matrix updated by the above formula. Figure 3 shows a series of cluster changes achieved by this incremental algorithm. The data set used in the Figure 3 is also the "soybean-small". Two clusters (green and red) is slightly overlapping in Figure 3(a), but are clearly separated in Figure 3(b). Relationship between two clusters (purple and blue) also changes from Figure 3(b) to Figure 3(c). The clusters in Figure 3(d) seems to be more condensed than Figure 3(c). We can see the clusters are gradually separated as the distance learning proceeds.



(a) Soybean-small



(b) Iris

Figure 4. Evaluation of Simple Selection Heuristics

V. Preliminary Experiments - Evaluation of Simple Selection Heuristics

We have developed a prototype of this tool. Through the test interactions with this tool, we found by chance a simple heuristics for selecting better constraints. The heuristics is, 1. to select a large cluster that may be unseparated from other clusters, 2. to find a must-link pair being apart as far as possible in the cluster. We compared the performance of this heuristics with random selection. Figure 4 shows the results, in which each axis means the number of constrains used in clustering and NMI (Normal Mutual Information) as evaluation measure. We used two data sets "Soybean-small" and "Iris" from UCI repository. In both data sets, selection with above heuristics by a human outperforms random selection, especially in early selection.

Although we need more experiments on various data sets, it is very interesting that heuristics found by human intuition works well in two data sets. This type of research is much related to "human active learning"[10]. Different from their experiments, our interest exists in "human sampling", where human only selects training examples and learning itself is done by machine. We consider this is more important to put machine learning to practical.

VI. Conclusion

This paper presented an interactive tool for constrained clustering that provides some basic functions such as the display of 2-D 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 user's cognitive load than that for clustering process under randomly selected constraints. In addition, selection bias of the constraints may help users find better selection strategies. We consider our proposed GUI is a promising approach for large-scaled applications like Web clustering.

The tool described in this paper is still a preliminary prototype. We have much work to do. For example, displaying data information is a very important function because users determine the labels of constraints based on the information. However a methods to display them depends on their data type. We need to implement different methods when displaying images data and document data. We also consider implementing the function of active learning that is important but rarely explored in constrained clustering, especially interactive constrained clustering. We think the active learning function may help users, or users may notice the drawback of the active learning algorithm. Eventually, we are currently planning to conduct user studies in larger scale in Web clustering to evaluate advantages of our proposed GUI.

References

[1] K. Wagstaff and S. Roger, "Constrained K-means Clustering with Background Knowledge," in *ICML'01* , pp.577-584, 2001.

[2] D. Klein, S. D. Kamvar and C. D. Manning, "From Instance-level Constraints to Space-Level Constraints: Making the Most of Prior Knowledge in Data Clustering," in *ICML'02*, pp.307-314, 2002.

[3] S. C. H. Hoi, R. Jin and M. R. Lyu, "Learning Nonparametric Kernel Matrices from Pairwise Constraints," in *ICML'07* , pp.361-368, 2007.

[4] S. Li, J. Liu and X. Tang, "Pairwise Constraint Propagation by Semidefinite Programming for Semi-Supervised Classification," in *ICML'08* , pp.576-583, 2008.

[5] S. Basu and et al., "A probabilistic Framework for Semi-Supervised Clustering," in *KDD ' 04*, pp. 59-68, 2004.

[6] C. Carpineto, S. Osiński, G. Romano and D. Weiss, "A Survey of Web Clustering Engines," *ACM Comput. Surver*, Vol.41, No.3, pp.1-38, 2009.

[7] I. Borg and P. Groenen, "Modern multidimensional scaling," *Theory and applications*, 1997.

[8] A. Asunction and D. J. Newman, "UCI Machine Learning Repository," Irvine, CA: University of California, School of Information and Computer Science, 2007.

[9] P. Jain and et al., "Online Metric Learning and Fast Similarity Search," *NIPS'08*, pp.761-768, 2008.

[10] R. Castro, C. Karish, R. Nowak, R. Qian, T. Rogers and Z. Zhu, "Human Active Leaning," *NIPS'08*, pp. 241-248, 2008.