KCK-means: A Clustering Method based on Kernel Canonical Correlation Analysis

Chuan-Liang Chen[1], Yun-Chao Gong[2], and Ying-Jie Tian[3, †]

[1]Department of Computer Science, Beijing Normal University, Beijing 100875, China
[2]Software Institute, Nanjing University, Nanjing, China
[3]Research Centre on Fictitious Economy & Data Science, Chinese Academy of Sciences, 100080, Beijing, China
C.L.Chen86@gmail.com, gyc05@software.nju.edu.cn, tianyingjie1213@163.com

Abstract. Kernel Canonical Correlation Analysis (KCCA) is a technique that can extract common features from a pair of multivariate data, which may assist in mining the ground truth hidden in the data. In this paper, a novel partitioning clustering method called KCK-means is proposed based on KCCA. We also show that KCK-means can not only be run on two-view data sets, but also it performs excellently on single-view data sets. KCK-means can deal with both binary-class and multi-class clustering tasks very well. Experiments with three evaluation metrics are also presented, the results of which reflect the promising performance of KCK-means.

Key words: Kernel Canonical Correlation Analysis, K-means clustering, Similarity Measure, Clustering Algorithm

1 Introduction

Clustering is one of the most commonly techniques which is widely applied to extract knowledge, especially when lacking any a priori information (e.g., statistical models) about the data. Generally, the problem of clustering deals with partitioning a data set consisting of n points embedded in m-dimensional space into k distinct set of clusters, such that the data points within the same cluster are more similar to each other than to data points in other clusters [3]. There are two main approaches of clustering algorithms, hierarchical (e.g., agglomerative methods) and partitional approaches (e.g., k-means, k-medoids, and EM). Most of these clustering algorithms are based on elementary distance properties of the instance space [4].

In some interesting application domains, instances are represented by attributes that can naturally be split into two subsets, either of which suffices for learning [5], such as web pages which can be classified based on their content as well as based on the anchor texts of inbound hyperlinks. Intuitively, there may be some projections in these two views which should have strong correlation with the ground truth. Kernel

† Corresponding Author.

Canonical Correlation Analysis (KCCA) is such a technique that can extract common features from a pair of multivariate data, which can be used as a statistical tool to identify the correlated projections between two views. Therefore, KCCA is expected to be used to measure the similarity between data points excellently. In this paper, we propose two algorithms based on KCCA which can improve the performances of traditional clustering algorithms—K-means, namely KCK-means for two-view data sets and single-view data sets that could not be split naturally. The results of experiments show that their performances are much better than those of the original algorithms. Our empirical study shows that these two algorithms can not only perform excellently on both two-view and single-view data, but also be able to extract better quality clusters than traditional algorithms.

The remainder of this paper is organized as follows. We demonstrate KCCA and propose the algorithms in Sect. 2. Performance measures, experiment results and their analysis are presented in Sect. 3. Finally, Sect. 4 presents the main conclusions.

2 KCK-means Method

2.1 Canonical Correlation Analysis

Firstly, we briefly review Canonical Correlation Analysis (CCA), then its kernel extension—Kernel Canonical Correlation Analysis (KCCA).

CCA is computationally an eigenvector problem. It attempts to find two sets of basis vectors, one for each view, such that the correlation between the projections of these two views into the basis vectors are maximized. Let $X = \{x_1, x_2, \ldots, x_l\}$ and $Y = \{y_1, y_2, \ldots, y_l\}$ denote two views, i.e. two attribute sets describing the data. CCA finds projection vectors w_x and w_y such that the correlation coefficient between $w_x^T X$ and $w_y^T Y$ is maximized. That is [12],

$$\rho = \arg\max_{w_x, w_y} \left(\frac{w_x^T C_{xy} w_y}{\sqrt{w_x^T C_{xx} w_x w_y^T C_{yy} w_y}} \right), \quad w.r.t \begin{cases} w_x^T C_{xx} w_x = 1 \\ w_y^T C_{yy} w_y = 1 \end{cases}, \tag{1}$$

where C_{xy} is the between-sets covariance matrix of X and Y, C_{xx} and C_{yy} are respectively the within-sets covariance matrices of X and Y. The maximum canonical correlation is the maximum of ρ with respect to w_x and w_y.

Assume that C_{yy} is invertible, then

$$w_y = \frac{1}{\lambda} C_{yy}^{-1} C_{yx} w_x \ , \tag{2}$$

and

$$C_{xy}C_{yy}^{-1}C_{yx}w_x = \lambda^2 C_{xx}w_x \ . \tag{3}$$

By first solving for the generalized eigenvectors of Eq. 3, we can therefore obtain the sequence of w_x's and then find the corresponding w_y's using Eq. 2.

However, in complex situations, CCA may not extract useful descriptors of the data because of its linearity. In order to identify nonlinearly correlated projections between the two views, kernel extensions of CCA (KCCA) can be used [12]. Kernel CCA offers an alternative solution by first projecting the data into a higher dimensional feature space, i.e. mapping x_i and y_i to $\phi(x_i)$ and $\phi(y_i)$ respectively (i = 1, 2, … , l). And then $\phi(x_i)$ and $\phi(y_i)$ are treated as instances to run CCA routine. Let S_x = $\{(\phi(x_1), \phi(x_2), ..., \phi(x_l))\}$ and S_y = $\{(\phi(y_1), \phi(y_2), ..., \phi(y_l))\}$. Then the directions w_x and w_y can be rewritten as the projection of the data onto the direction α and β ($\alpha, \beta \in \Re^l$): $w_x = S_x\alpha$ and $w_y = S_y\beta$. Let $K_x = S_x^T S_x$ and $K_y = S_y^T S_y$ be the kernel matrices corresponding to the two views. Substituting into Eq. 1 we can obtain the new objective function

$$\rho = \max_{\alpha,\beta} \frac{\alpha^T K_x K_y \beta}{\sqrt{\alpha^T K_x^2 \alpha \cdot \beta^T K_y^2 \beta}} \ . \tag{4}$$

α can be solved from

$$(K_x + \kappa I)^{-1} K_y (K_y + \kappa I)^{-1} K_x \alpha = \lambda^2 \alpha \ , \tag{5}$$

where κ is used for regularization. Then β can be obtained from

$$\beta = \frac{1}{\lambda}(K_y + \kappa I)^{-1} K_x \alpha \ . \tag{6}$$

Let $\mathcal{K}_x(x_i, x_j) = \phi_x(x_i)\phi_x^T(x_j)$ and $\mathcal{K}_y(y_i, y_j) = \phi_y(y_i)\phi_y^T(y_j)$ are the kernel functions of the two views. Then for any for any x^* and y^*, their projections can be obtained from $P(x^*) = \mathcal{K}_x(x_i, X)\ \alpha$ and $P(y^*) = \mathcal{K}_y(y_i, Y)\ \beta$ respectively.

A number of α and β (and corresponding λ) can be solved from Eq. 5 and Eq. 6. If the two views are conditionally independent given the class label, the most strongly correlated pair of projections should be in accordance with the ground-truth [9]. However, in real-world applications the conditional independence rarely holds, and therefore, information conveyed by the other pairs of correlated projections should not be omitted [9].

So far we have considered the kernel matrices as invertible, although in practice this may not be the case [20]. We use Partial Gram-Schmidt Orthogonolisation (PGSO) to approximate the kernel matrices such that we are able to re-represent the correlation with reduced dimensionality [12]. In PGSO algorithm, there is a precision parameter—η, which is used as a stopping criterion. For low-rank approximations, we need keep eigenvalues greater than η and the number of eigenvalues we need to

consider is bounded by a constant that depends solely on the input distribution [20]. Since the dimensions of the projections rely on the $N \times M$ lower triangular matrix output by PGSO which relies on this stopping criterion, we discuss the influence of η to our algorithm in Sect. 3. More detail about PGSO is described in [20].

2.2 Two KCK-means Algorithms

In our method, the similarity between data points is measured partly by the projections obtained by KCCA and extends the K-means algorithm.

In [7], Balcan et al. showed that given appropriately strong PAC-learners on each view, an assumption of expansion on the underlying data distribution is sufficient for co-training to succeed, which implies that the stronger assumption of independence between the two views is not necessary, and the existence of sufficient views is sufficient. Similarly, the distance function f_{sim} described below is also calculated based on the assumption that X and Y are sufficient to describe the data respectively, which is the same as the assumption of expansion about the co-training method. Actually, our method is intuitively derived from co-training [10]. Since the two views are sufficient to describe the data, both of them may be consist of some projections correlate with the ground truth. So we intend to measure the similarity between instances using information from two views of data. KCCA is an excellent tool that can carry out this task. Therefore, measuring by the use of KCCA may be a promising way of solving the problem of traditional distance measures.

Let m denote the number of pairs of correlated projections that have been identified, then x^* and y^* can be projected into $P_j(x^*)$ and $P_j(y^*)$ (j = 1, 2, … ,m). Let f_{sim} denote distance functions, which is L_2-norm $\|\bullet\|^2$ in this paper. Of course, other similarity distance functions also could be. Based on the projections obtained by KCCA, a new similarity measure can be defined as follows,

$$f_{sim}(x_i, x_j) = \mu \left\| x_i - x_j \right\|^2 + \sum_{k=1}^{m} \left\| P_k(x_i) - P_k(x_j) \right\|^2 , \tag{7}$$

where μ is a parameter which regulates the proportion of the distance between the original instances and the distance of their projections. Based on this similarity measure, we propose the first algorithm as follows.

Input: X and Y, two views of a data set with n instances
k, the number of clusters desired

Output: C_1 and C_2, two vectors containing the cluster indices of each point of X and Y.

Process:

1. Identify all pairs of correlated projections, obtaining α_i, β_i by solving Eqs. 5 and 6 on X and Y.
2. **for** i = 1, 2, …, l **do**
   Project x_i and y_i into m pairs projections and obtain $P(x_i)$ and $P(y_i)$.
3. Get the new data sets by unite X and $P(X)$, Y and $P(Y)$, i.e. $M_x = X \cup P(X)$, $M_y = Y \cup P(Y)$.

Cluster M_x and M_y respectively as follows:

4. Randomly assign each instance of M_x (M_y) to one cluster of the k clusters.
5. Calculate the cluster means, i.e., calculate the mean value (both the original value and the projections' value) of the instance of each cluster.
6. **repeat**
7. (re)assign each instances to the cluster to which the instance is the most similar by calculating Eq. 7.
8. update the cluster means.
9. **until** no change.

Fig. 1. KCK-means Algorithm for two-view data sets.

However, two-view data sets are rare in real world, which is the cause that though co-training is a powerful paradigm, it is not widely applicable. In [6], it points out that if there is sufficient redundancy among the features, we are able to identify a fairly reasonable division of them, and then co-training algorithms may show similar advantages to those when they perform on the two-view data sets. Similarly, in this paper, we try to randomly split the single-view data set into two parts and treat them as the two views of the original data set to perform KCCA and then KCK-means.

Input: X, a single-view data set with n instances
k, the number of clusters desired
Output: C, a vector containing the cluster indices of each point of X.
Process:

1. Randomly spilt X into two views with the same attributes, X_1 and X_2.
2. Identify all pairs of correlated projections, obtaining α_i, β_i by solving Eqs. 5 and 6 on X_1 and X_2.
3. **for** $i = 1, 2, \ldots, l$ **do**
   Project $x_{1,i}$ and $x_{2,i}$ into m pairs projections and obtain $P(x_{1,i})$ and $P(x_{2,i})$.
4. Unite $P(X_1)$ and $P(X_2)$ into $P(X)$, i.e. $P(X) = P(X_1) \cup P(X_2)$.
5. Get the new data sets by unite X and $P(X)$, i.e. $M_x = X \cup P(X)$.

Cluster M_x:

6. Randomly assign each instance of M_x to one cluster of the k clusters.
7. Calculate the cluster means, i.e., calculate the mean value (both the original value and the projections' value) of the instance of each cluster.
8. **repeat**
9. (re)assign each instances to the cluster to which the instance is the most similar by calculating Eq. 7.
10. update the cluster means.
11. **until** no change.

Fig. 2. The KCK-means Algorithm for single-view data sets.

3 Experiments and Analysis

Two standard multi-view data sets are applied to evaluate the effectiveness of the first version of KCK-means. They are

Course: The *course* data set has two views and contains 1,051 examples, each corresponding to a web page, which is described in [10]. 200 examples are used in this paper and there are 44 positive examples.

Ads: The *url* and *origurl* data sets are derived from the *ads* data set which is described in [16] and has two categories. 300 examples are used in this paper, among which 42 examples are positive. In this paper, we construct a two-view dataset by using the *url* view and *origurl* view.

In order to find out how well the second version of KCK-means performs on single-view data sets, we use three single-view data sets‡.

A3a: The *a3a* is a single-view data set derived from Adult Data Set of UCI, which is described in [11]. It has two categories and 122 features. 3,185 examples are used and there are 773 positive examples.

W1a: The *w1a* is a single-view data set derived from *web page* dataset which is described in [9]. It has two categories and 300 sparse binary keyword attributes. 2,477 examples are used, among which 72 examples are positive.

DNA: The DNA is a single-view data set which is described in [8]. It has three categories and 180 attributes. 2,000 examples are used, among which 464 examples are 1st class, 485 examples are 2nd class, and 1,051 examples are 3rd class.

We use three performance measures, *Pair-Precision*, *Intuitive-Precision* and *Mutual Information*, to measure the quality of the clusters obtained by the KCK-means.

Pair-Precision: The evaluation metric in [2] is used in our experiments. We evaluate a partition i.e. the correct partition using

$$accuracy = \frac{num(correct \quad decisions)}{n(n-1)/2} \quad .$$

Mutual Information: Though entropy and purity are suitable for measuring a single cluster's quality, they are both biased to favor smaller clusters. Instead, we use a symmetric measure called *Mutual Information* to evaluate the overall performance. The *Mutual Information* is a measure of the additional information known about one when given another [1], that is

$$MI(A,B) = H(A) + H(B) - H(A,B) \ ,$$

where $H(A)$ is the entropy of A and can be calculated by using

$$H(A) = -\sum_{i=1}^{n} p(x_i)\log_2(p(x_i)) \ .$$

Intuitive-Precision: We choose the class label that share with most samples in a cluster as the class label. Then, the precision for each cluster A is defined as:

$$P(A) = \frac{1}{|A|}\max(\left|\{x_i \mid label(x_i) = C_j\}\right|) \ .$$

‡ On http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets, all these single-view data sets can be downloaded.

In order to avoid the possible bias from small clusters which have very high precision, the final precision is defined by the weighted sum of the precision for all clusters, as shown in the following equation

$$P = \sum_{k=1}^{G} \frac{|A_k|}{N} P(A_k) \ ,$$

where G is the number of categories (classes) and N is the total number of instances.

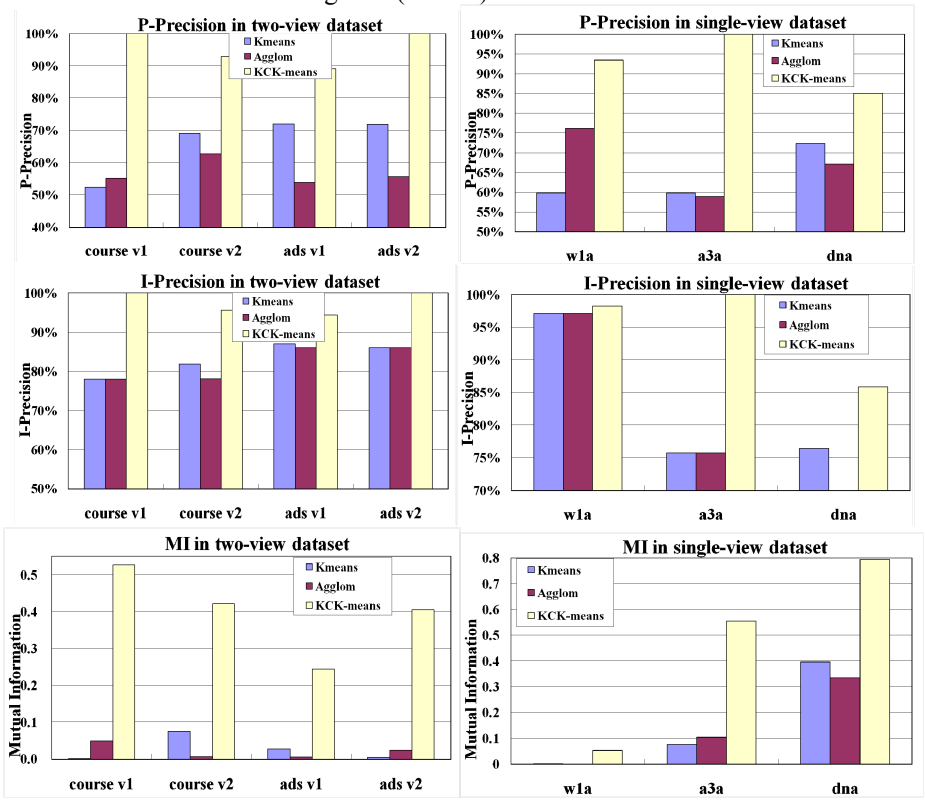

Fig. 3. Clustering results on two two-view data sets (*course* and *ads*, on the left column) and three single-view data sets (*a3a*, *w1a* and *DNA*, on the right column) using KCK-means comparing with two traditional clustering algorithms, K-means and Agglom (agglomerative hierarchical clustering) with three performance measures, P-Precision (*Pair-Precision*), I-Precision (*Intuitive-Precision*), and MI (*Mutual Information*).

The comparison among between KCK-means and K-means, agglomerative hierarchical clustering, are performed. In order to better reflect the performance of the three algorithms, for all experiments demonstrated below with the two partitioning algorithms, K-means and KCK-means, the diagrams are based on averaging over ten clustering runs to compensate for their randomized initialization. And that is also beneficial for measuring the performance of the second version of KCK-means on the single-view data sets for its randomly splitting these data sets. The performances of the three algorithms are showed in Fig. 3.

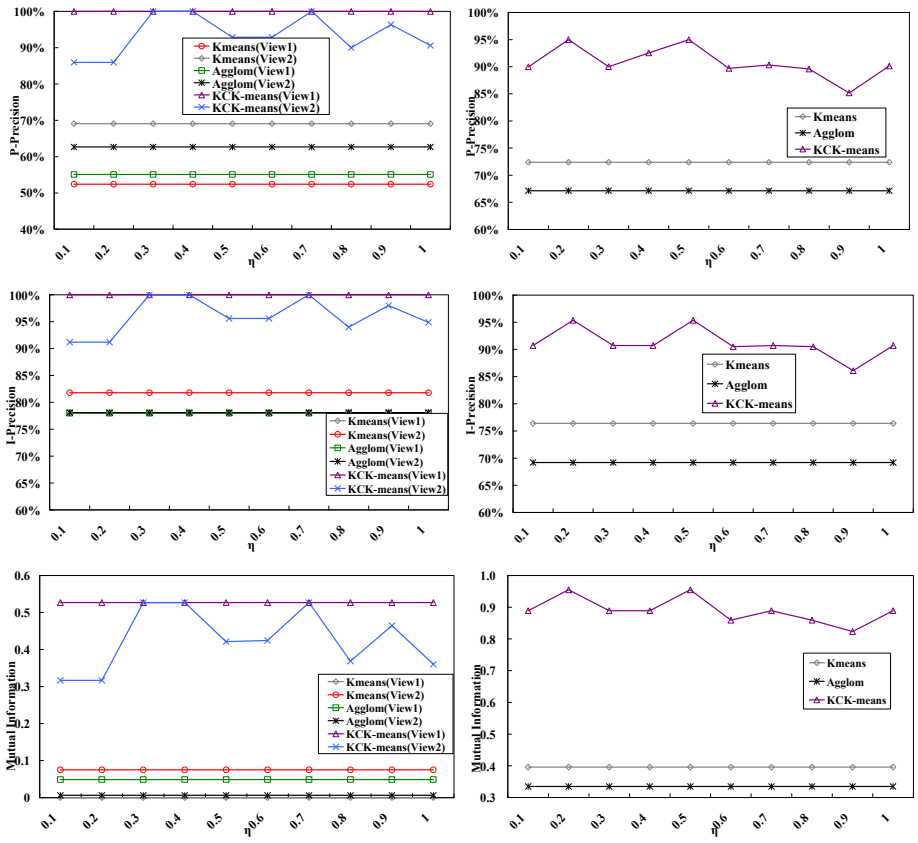

Fig. 4. The influence of η on the performance of KCK-means on the two-view data set *course* and the single-view data set *DNA*, where η changes from 0.1 to 1.0, all of the three evaluation metrics, *Pair-Precision*, *Intutitive-Precision* and *Mutual Information*, are used.

In Fig. 3, the performances of KCK-means are much better than those of other two traditional clustering algorithms. On some data sets such as *a3a*, the *Pair-Precision* and *Intuitive-Precision* of the results of KCK-means are both almost 100%, but *Pair-Precision* and *Intuitive-Precision* of the results of K-means and agglomerative hierarchical clustering are 59.74%, 75.73% and 58.87%, 75.73% respectively. KCK-means also performs excellently on the multi-class data set—*DNA* and gets 85.03% *Pair-Precision*, for K-means and agglomerative hierarchical clustering 72.39% and 67.13% respectively. For other two evaluation metrics, KCK-means is also much better than those of the others'.

In our experiments, we also note that when the proportion parameter μ is set to be very small or even zero, the performance of KCK-means is the best, which means using the projections obtained from KCCA the similarity between instances already can be measured good enough. The μ in the experiments described in this paper is all set to be 10^{-6}.

In Sect. 2.1 we have stated that there is a precision parameter (or stopping criterion)—η in the PGSO algorithm, on which the dimensions of the projections rely. Now we demonstrate its influence on the performance of KCK-means. In order to better measure such influence, we use two data sets, *course* and *DNA*, in the experiments described below. Because *course* is a two-view data set with two classes, and *DNA* is a single-view data set with three classes, then we can combine the measure of the KCK-means on two-view data set and single-view data set simultaneously. The results are derived on more than ten clustering showed in Fig. 4.

In Fig. 4 we can find that follow the change of η, the performance of KCK-means changes a little. Furthermore, even considering the influence, the performances of KCK-means on both data sets are also much better than the other two clustering algorithms.

However, in the experiments we find when η is larger than some threshold which depends on given data set the performance of KCK-means descends very much even worse than those of K-means and agglomerative hierarchical clustering. After carefully observation, we find in such situations the number of dimensions of projections is always very small, sometimes even only one dimension. Just as what we have described in Sect. 2.1, in real-world applications the conditional independence rarely holds, and therefore, information conveyed by the other pairs of correlated projections should not be omitted [9]. Therefore, this performance descending may be caused by lacking information conveyed by the other projections.

4 Conclusion

In this paper, we propose a novel partitioning method, i.e. KCK-means, based on KCCA and inspiration from co-training. By using KCCA which mines the ground truth hidden in the data, KCK-means measures the similarity between instances. On two two-view data sets, *course* and *ads*, and three single-view data sets, *a3a*, *w1a* and *DNA*, the experiments are performed using three performance measures, *Pair-Precision*, *Intuitive-Precision* and *Mutual Information*. The results reflect that by using KCK-means, much better quality of clusters could be obtained than those obtained from K-means and agglomerative hierarchical clustering algorithms.

However, we also observe that when the number of dimensions of the projections obtained from KCCA is very small, the performance of KCK-means descends very much even worse than those of the two traditional clustering algorithms. This reflects that in real-world applications, we need to consider the information conveyed by the other pairs of correlated projections obtained from KCCA, instead of only considering the strongest projection or very few stronger projections. That is, the number of dimensions of projections obtained from KCCA and then used in KCK-means must be enough.

Acknowledgments. The research work described in this paper was supported by grants from the National Natural Science Foundation of China (Project No. 10601064, 70531040, 70621001).

References

1. Butte, A.J., Kohane, I.S.: Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. In: Pacific Symposium on Biocomputing, pp. 415--426. Hawaii (2000)
2. Wagstaff, K., Claire, C.: Clustering with Instance-level Constraints. In: the 17th International Conference on Machine Learning, pp. 1103--1110. Morgan Kaufmann press, Stanford (2000)
3. Khan, S.S., Ahmadb, A.: Cluster center initialization algorithm for K-means clustering. Pattern Recognition Letters 25, vol. 25, pp. 129--1302 (2004)
4. Kirsten, M., Wrobel, S.: Relational distance-based clustering. In: the 8th International Conference on Inductive Logic Programming, pp. 261--270. Springer press, Madison (1998)
5. Bickel, S., Scheffer, T.: Multi-View Clustering. In: the 4th IEEE International Conference on Data Mining, pp. 19--26. IEEE press, Brighton (2004)
6. Nigam, K., Ghani, R.: Analyzing the effectiveness and applicability of co-training. In: the 9th international conference on Information and knowledge management, pp. 86--93. ACM press, McLean (2000)
7. Balcan, M.F., Blum, A., Yang, K.: Co-training and expansion: Towards bridging theory and practice. In: the 18th Annual Conference on Neural Information Processing Systems, pp. 89--96. MIT press, Vancouver (2005)
8. Hsu, C.W., Lin, C.J.: A comparison of methods for multi-class support vector machines. IEEE Transactions on Neural Networks 13, vol. 13, pp. 415--425 (2002)
9. Zhou, Z.H., Zhan, D.C., Yang, Q.: Semi-supervised learning with very few labeled training examples. In: the 22nd AAAI Conference on Artificial Intelligence, pp.675--680. AAAI press, Vancouver (2007).
10. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: the Conference on Computational Learning Theory, pp. 92--100. Morgan Kaufmann press, Madison (1998)
11. Kohavi, R.: Scaling Up the Accuracy of Naive-Bayes Classifiers: a Decision-Tree Hybrid. In: the Second International Conference on Knowledge Discovery and Data Mining, pp. 202--207. AAAI press, Oregon (1996)
12. Hardoon, D.R., Szedmak, S., Shawe-Taylor, J.: Canonical correlation analysis; An overview with application to learning methods. Technical report, Department of Computer Science Royal Holloway, University of London (2003)