2009 Ninth IEEE International Conference on Data Mining

Maximum Margin Clustering with Multivariate Loss Function

Bin Zhao[†], James Kwok[‡], Changshui Zhang[†]
[†]State Key Laboratory of Intelligent Technology and Systems
Tsinghua National Laboratory for Information Science and Technology (TNList)
Department of Automation, Tsinghua University, Beijing, China
[‡]Department of Computer Science and Engineering
Hong Kong University of Science and Technology, Hong Kong
{zhaobinhere@hotmail.com, jamesk@cse.ust.hk, zcs@mail.tsinghua.edu.cn}

Abstract

This paper presents a simple but powerful extension of the maximum margin clustering (MMC) *algorithm that optimizes multivariate performance measure specifically defined for clustering, including* Normalized Mutual Information, Rand Index *and* F-measure*. Different from previous* MMC *algorithms that always employ the error rate as the loss function, our formulation involves a multivariate loss function that is a non-linear combination of the individual clustering results. Computationally, we propose a* cutting plane *algorithm to approximately solve the resulting optimization problem with a guaranteed accuracy. Experimental evaluations show clear improvements in clustering performance of our method over previous* maximum margin clustering *algorithms.*

I. Introduction

Recently, *maximum margin clustering (MMC)* has attracted considerable interests in the data mining and machine learning communities [18], [19], [20], [23], [26], [27], [24]. The key idea of *MMC* is to extend the maximum margin principle of *support vector machines (SVM)* to the unsupervised learning scenario. Given a set of data samples, *MMC* performs clustering by labeling the samples such that the *SVM* margin obtained is maximized over all possible cluster labelings. Recent studies have demonstrated its superior performance over conventional clustering methods.

Existing *maximum margin clustering* algorithms all directly adopt the formulation of *support vector machines*. Specifically, the (regularized) hinge loss is minimized with respect to both the data labels and separating hyperplane. In a classification problem, the classification error is an obvious performance measure, and the hinge loss, which upper bounds the classification error, is thus well justified to be used as the loss function. However, its justification for use in clustering is not so clear. Often, the objective of clustering is not on learning the exact labels. Instead, a good clustering solution should minimize the intra-cluster variance and maximize the inter-cluster variance. The most suitable and well-defined performance measures for clustering include the *Normalized Mutual Information* [28], *Rand Index* [13] and *F-measure* [12]. Empirical studies show that *MMC* optimized for the error rate may have suboptimal performance in terms of the *Rand Index* (or *NMI, F-measure*) [26]. Indeed, despite the battery of clustering algorithms that have been proposed, we are not aware of any clustering algorithm (not necessarily based on *MMC*) that can directly optimize clustering-specific performance measures. So, a natural question is: Why do we minimize a loss function that is defined for classification, instead of directly minimizing those loss functions that are specifically defined for measuring clustering performance?

Recently, a similar observation has also been made in the context of *ranking*. Consequently, researchers in the ranking community have proposed algorithms that directly optimize the ranking performance measures such as the mean average precision (*MAP*) [21] and the normalized discounted cumulative gain (*NDCG*) [1], [10]. In this paper, we reformulate *maximum margin clustering* so that loss functions that are defined for clustering can be explicitly optimized in the objective function. However, different from the error rate, these clustering-specific loss functions cannot be decomposed linearly into a sum of loss functions over the individual examples. Instead, they are non-linear combinations of the individual clustering results.

Besides, as the pattern labels are unknown in a cluster-

1550-4786/09 $26.00 © 2009 IEEE
DOI 10.1109/ICDM.2009.37

ing problem, the resulting optimization problem contains an exponential number of constraints. In this paper, we propose an efficient algorithm for solving the resulting optimization problem based on the *cutting plane* algorithm [8], [17]. As will be shown in the sequel, one can construct a nested sequence of successively tighter relaxations of the original problem, and each optimization problem in this sequence can be efficiently solved as a *quadratic programming problem* by using the *constrained concave-convex procedure* (*CCCP*) [16]. A key issue to the success of this approach is that the most violated constraint in each cutting plane iteration can be found efficiently. Experimental evaluations on toy and real-world data sets show that the proposed method outperforms existing *maximum margin clustering* algorithms in terms of these clustering-specific performance measures.

The rest of this paper is organized as follows. In Section 2, we first review the basics of *maximum margin clustering* and performance measures widely used in clustering. The formulation of *MMC* with clustering-specific loss functions is presented in Section 3. Section 4 presents the experimental results on several toy and real-world data sets, followed by some concluding remarks in Section 5.

II. Review

A. Maximum Margin Clustering

As briefly introduced in Section I, the key idea of *maximum margin clustering* (*MMC*) is to extend the maximum margin principle from supervised learning to unsupervised learning. In the two-cluster case, given a set of examples $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, *MMC* aims at finding the best label combination $\mathbf{y} = \{y_1, \ldots, y_n\} \in \{-1, +1\}^n$ such that an *SVM* trained on $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ will yield the largest margin. Although it is indeed possible to extend our approach to the multi-class case [20], [26], the extension is not simple and for ease of presentation we focus on simple two class clustering in this paper. Binary clustering is still a reasonably well-motivated problem, as binary clustering algorithms can play an important part in hierarchical clustering.

Computationally, *MMC* solves the following optimization problem [26]:

$$\begin{aligned} \min_{\mathbf{y}\in\{\pm 1\}^n} \min_{\mathbf{w},b,\xi_i} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{n}\sum_{i=1}^{n}\xi_i \\ \text{s.t.} \quad & \forall i \in \{1,\ldots,n\}: \\ & y_i[\mathbf{w}^T\Phi(\mathbf{x}_i) + b] \geq 1 - \xi_i,\ \xi_i \geq 0, \\ & -l \leq \sum_{i=1}^{n}[\mathbf{w}^T\Phi(\mathbf{x}_i) + b] \leq l. \end{aligned} \tag{1}$$

where the data samples $\mathcal{X}$ are mapped into a high (possibly infinite) dimensional feature space using a possibly nonlinear function $\phi(\mathbf{x})$, and by using the kernel trick, this mapping could be done implicitly. Specifically, we define the kernel matrix K formed from the inner products of feature vectors, such that $K_{ij} = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$, and transform the problem such that $\phi(\mathbf{x})$ only appears in the inner product. However, in those cases where kernel trick cannot be applied, if we still want to use a nonlinear kernel, it is possible to compute the coordinates of each sample in the *kernel PCA basis* [14] according to the kernel K. More directly, as stated in [2], one can also compute the Cholesky decomposition of the kernel matrix $K = \hat{\mathbf{X}}\hat{\mathbf{X}}^T$, and set $\phi(\mathbf{x}_i) = (\hat{\mathbf{X}}_{i,1}, \ldots, \hat{\mathbf{X}}_{i,n})^T$. For notational simplicity, we will use $\mathbf{x}$ instead of $\Phi(\mathbf{x})$ in the sequel.

The last constraint in (1) is often called the class balance constraint, and $l > 0$ is a constant controlling the class imbalance. More precisely, this constraint is a relaxation of the integer constraint $-l \leq \sum_{i=1}^{n} y_i \leq l$, and is widely used in maximum margin clustering [25], [26] and spectral clustering [15]. Specifically, this constraint prevents the trivially "optimal" solution that assigns all patterns to the same class and thus achieves "infinite" margin. It also avoids the unwanted solution of separating a single outlier or a very small group of samples from the rest of the data.

B. Performance Measures in Clustering

In this section, we review three performance measures that have been widely used in clustering, namely, *Normalized Mutual Information (NMI)*, *Rand Index* and the *F-measure*.

The *NMI* measures the clustering quality from an information-theoretic perspective. In the context of clustering, let $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k\}$ be the set of clusters obtained on the n samples, and $\mathcal{L} = \{\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_m\}$ be the set of true class labels. Moreover, let $P(\mathcal{C}_i)$, $P(\mathcal{L}_j)$, and $P(\mathcal{C}_i \cap \mathcal{L}_j)$ be the probabilities of an example in cluster $\mathcal{C}_i$, class $\mathcal{L}_j$, and the intersection of $\mathcal{C}_i$ and $\mathcal{L}_j$, respectively. Then, *NMI* is defined as [28]:

$$NMI(\mathcal{C}, \mathcal{L}) = \frac{I(\mathcal{C}, \mathcal{L})}{\sqrt{H(\mathcal{C})H(\mathcal{L})}}, \tag{2}$$

where

$$\begin{aligned} I(\mathcal{C}, \mathcal{L}) &= \sum_{i=1}^{k}\sum_{j=1}^{m} P(\mathcal{C}_i \cap \mathcal{L}_j) \log \frac{P(\mathcal{C}_i \cap \mathcal{L}_j)}{P(\mathcal{C}_i)P(\mathcal{L}_j)} \\ &= \frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{m} |\mathcal{C}_i \cap \mathcal{L}_j| \log \frac{n|\mathcal{C}_i \cap \mathcal{L}_j|}{|\mathcal{C}_i||\mathcal{L}_j|} \end{aligned} \tag{3}$$

is the mutual information between $\mathcal{C}$ and $\mathcal{L}$, and

$$H(\mathcal{C}) = -\sum_{i=1}^{k} P(\mathcal{C}_i)\log P(\mathcal{C}_i) = -\sum_{i=1}^{k} \frac{|\mathcal{C}_i|}{n} \log \frac{|\mathcal{C}_i|}{n} \quad (4)$$

is the entropy of $\mathcal{C}$. A higher *NMI* value indicates better clustering quality.

On the other hand, the *Rand Index* views clustering as a series of decisions, one for each of the $\frac{1}{2}n(n-1)$ pairs of examples in the data set $\mathcal{X}$. Ideally, a clustering algorithm should assign two examples to the same cluster if and only if they are similar. Thus, there are four possible scenarios for each assignment. A true positive (TP) decision assigns two similar examples to the same cluster, and a true negative (TN) decision assigns two dissimilar examples to different clusters; while a false positive (FP) decision assigns two dissimilar examples to the same cluster, and a false negative (FN) decision assigns two similar examples to different clusters. The *Rand index (RI)* then measures the percentage of correct decisions, as:

$$RI = \frac{TP + TN}{TP + FP + FN + TN}. \quad (5)$$

As with *NMI*, higher values indicate higher quality.

A potential problem with the *Rand index* is that it gives equal weights to false positives and false negatives. However, in many real-world applications, having dissimilar examples in the same cluster is often less undesirable than separating similar examples into different clusters. To alleviate this problem, the *F measure* introduces a user-defined parameter β to penalize false negatives more strongly than false positives. Specifically, it is defined as

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}, \quad (6)$$

where $\beta \geq 1$, and

$$P = \frac{TP}{TP + FP}, \quad R = \frac{TP}{TP + FN} \quad (7)$$

are the precision and recall, respectively.

III. Maximum Margin Clustering with Multivariate Loss Function

As can be seen from (1), *MMC* (as in the standard SVM) uses the hinge loss, which is an upper bound of the classification error. However, the objective of clustering is often not on learning the exact labels, and therefore the error rate may not be suitable. In this section, we consider a direct optimization of the performance measures for clustering, including the *Normalized Mutual Information*, *Rand Index* and *F-measure*.

However, one advantage of using the hinge loss as the loss function is that it can be decomposed linearly as a combination of loss functions over all the examples. Specifically, let $\Delta(h(\mathcal{X}), \mathbf{y})$ be the loss function that measures the disagreement between hypothesis $h(\mathcal{X})$ and the learned labels $\mathbf{y}$. Then,

$$\Delta_{\text{hinge-loss}}(h(\mathcal{X}), \mathbf{y}) = \sum_{i=1}^{n} \Delta_{\text{hinge-loss}}(h(\mathbf{x}_i), y_i).$$

However, the aforementioned performance measures for clustering cannot be decomposed this way. For example, to compute *NMI*, we need to have complete information of the entire clustering result. In fact, these measures are nonlinear combinations of the individual clustering results. To employ the performance measures defined for clustering as the loss function in *maximum margin clustering*, we need to design algorithms that finds the hypothesis $h(\mathcal{X})$ which minimizes $\Delta(h(\mathcal{X}), \mathbf{y})$ over the entire data set $\mathcal{X}$.

In this paper, we present a new formulation of *maximum margin clustering* that directly optimizes multivariate performance measures defined for clustering (Section III-A). Specifically, we build upon the approach used by [5], [4] for optimizing multivariate performance measures for classification. Unlike the supervised problem, however, *maximum margin clustering* with multivariate loss function is formulated as a non-convex integer programming problem and requires a substantially extended algorithm, which we describe in Section III-B.

A. Multivariate Prediction Rule

Conventionally, *maximum margin clustering* defines hypothesis h as a univariate function from a single example $\mathbf{x}$ to its corresponding label $y \in \{-1, +1\}$: $h : \mathcal{X} \to \mathcal{Y}$. This hypothesis is suitable for loss functions that can be decomposed as linear combinations over the examples, such as the error rate. However, for loss functions involving nonlinear combinations of the individual clustering results, this decomposition is no longer possible. Therefore, to employ those performance measures defined specifically for clustering as the loss function in *maximum margin clustering*, we need to consider a multivariate hypothesis $\bar{h}$ that maps a tuple of examples $\bar{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_n) \in \bar{\mathcal{X}} = \mathcal{X} \times \cdots \times \mathcal{X}$ to a tuple of labels $\bar{y} = (y_1, \ldots, y_n) \in \{-1, +1\}^n$.

To implement this multivariate mapping, a generalized linear model

$$f(\bar{\mathbf{x}}, \bar{\mathbf{y}}) = \mathbf{w}^T \Psi(\bar{\mathbf{x}}, \bar{\mathbf{y}}) \quad (8)$$

is employed to decode the top-scoring output $\bar{h}_{\mathbf{w}}(\bar{\mathbf{x}}) = \arg\max_{\bar{\mathbf{y}} \in \bar{\mathcal{Y}}} f(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ for $\bar{\mathbf{x}}$. Here, $\mathbf{w}$ is a parameter vector as in the conventional formulation of *MMC*, and Ψ is a function that returns a feature vector measuring the match between tuples $\bar{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ and $\bar{\mathbf{y}} = (y_1, \ldots, y_n)$.

For *maximum margin clustering*, we define Ψ as

$$\Psi(\bar{\mathbf{x}}, \bar{\mathbf{y}}) = \sum_{i=1}^{n} y_i \mathbf{x}_i. \tag{9}$$

The multivariate prediction rule $\bar{h}$ allows us to reformulate *maximum margin clustering* to enable the inclusion of a sample-based multivariate loss function $\bar{\Delta}(\bar{h}(\bar{\mathbf{x}}), \bar{\mathbf{y}})$, instead of an example-based loss function $\Delta(h(\mathbf{x}), y)$ that is only suitable for decomposable loss functions. For a non-negative multivariate loss function $\bar{\Delta}$, we can reformulate *maximum margin clustering* as the following optimization problem:

$$\begin{aligned} \min_{\bar{\mathbf{y}}, \mathbf{w}, \xi} \ & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\xi \qquad (10) \\ \text{s.t. } & \forall \bar{\mathbf{y}}' \in \bar{\mathcal{Y}} : \mathbf{w}^T[\Psi(\bar{\mathbf{x}}, \bar{\mathbf{y}}) - \Psi(\bar{\mathbf{x}}, \bar{\mathbf{y}}')] \geq \bar{\Delta}(\bar{\mathbf{y}}', \bar{\mathbf{y}}) - \xi, \\ & \xi \geq 0, \\ & -l \leq \sum_{i=1}^{n} \mathbf{w}^T \mathbf{x}_i \leq l. \end{aligned}$$

Recall that the optimal label vector $\bar{\mathbf{y}}$ is given by

$$\begin{aligned} \bar{\mathbf{y}} &= \arg\max_{\bar{\mathbf{y}}' \in \bar{\mathcal{Y}}} \{\mathbf{w}^T \Psi(\bar{\mathbf{x}}, \bar{\mathbf{y}}')\} \\ &= \arg\max_{\bar{y}_i' \in \{-1, +1\}} \sum_{i=1}^{n} \bar{y}_i' \mathbf{w}^T \mathbf{x}_i, \end{aligned} \tag{11}$$

since each y_i can be optimized individually. Thus, we have $\forall i \in \{1, \ldots, n\} : \bar{y}_i = \mathrm{sgn}(\mathbf{w}^T \mathbf{x}_i)$, and the corresponding value for $\mathbf{w}^T \Psi(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ is $\sum_{i=1}^{n} |\mathbf{w}^T \mathbf{x}_i|$. Inserting $\bar{y}_i = \mathrm{sgn}(\mathbf{w}^T \mathbf{x}_i)$ into problem (10), performing *maximum margin clustering* with the multivariate loss function $\bar{\Delta}$ can therefore be equivalently formulated as

$$\begin{aligned} \min_{\mathbf{w}, \xi} \ & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\xi \qquad (12) \\ \text{s.t. } & \forall \bar{\mathbf{y}}' \in \bar{\mathcal{Y}} : \\ & \sum_{i=1}^{n} |\mathbf{w}^T \mathbf{x}_i| - \sum_{i=1}^{n} \bar{y}_i' \mathbf{w}^T \mathbf{x}_i \geq \sum_{\bar{\mathbf{y}} \in \bar{\mathcal{Y}}} z_{\bar{\mathbf{y}}} \bar{\Delta}(\bar{\mathbf{y}}', \bar{\mathbf{y}}) - \xi, \qquad (13) \\ & \xi \geq 0, \\ & -l \leq \sum_{i=1}^{n} \mathbf{w}^T \mathbf{x}_i \leq l, \end{aligned}$$

where $z_{\widetilde{\mathbf{y}}} = \prod_{i=1}^{n} I([\widetilde{y}_i = \mathrm{sgn}(\mathbf{w}^T \mathbf{x}_i))$ (with $I(\cdot)$ being the indicator function) selects the optimal $\bar{\mathbf{y}}$ out from the summmation.

B. Cutting Plane Algorithm

The major difficulty with problem (12) lies in the fact that there are an exponential number of constraints involved. For each admissible tuple $\bar{\mathbf{y}}' \in \{-1, +1\}^n$, a corresponding constraint (13) must be satisfied. The massive amount of constraints renders solving the problem exactly very time demanding. Therefore, the algorithm we propose in this paper targets to find an approximate solution to problem (12) with user-defined precision ϵ. That is to say, the approximate solution we seek satisfies all the constraints in problem (12) up to precision ϵ, where ϵ can be arbitrarily small. Technically, the algorithm finds a small subset of constraints from the whole set of constraints in problem (12) that ensures a sufficiently accurate solution. Specifically, we employ an adaptation of the *cutting plane* algorithm [8], [6], [17], [5], [26], where we construct a nested sequence of successively tighter relaxations of problem (12). Computationally, the *cutting plane* algorithm keeps a subset Ω of working constraints and computes the optimal solution to problem (12) subject to the constraints in Ω. The algorithm then adds the most violated constraint for problem (12) into Ω. In this way, a successively strengthening approximation of the original *maximum margin clustering* with the multivariate loss function is constructed by a cutting plane that cuts off the current optimal solution from the feasible set [8]. The algorithm stops when no constraint in (12) is violated by more than ϵ.

The *cutting plane* algorithm for solving problem (12) is presented in Algorithm 1. There are two steps involved in the main procedure: solving problem (12) subject to the constraint subset Ω and finding the most violated constraint. We will elaborate on these two steps in the following sections.

Algorithm 1 Cutting plane algorithm for maximum margin clustering with multivariate loss function.

Initialize: Constraint subset $\Omega = \emptyset$.
repeat
  Solve problem (12) for $\mathbf{w}$ under the current working constraint set Ω.
  Select the most violated constraint $\bar{\mathbf{y}}'$ and set $\Omega = \Omega \cup \{\bar{\mathbf{y}}'\}$.
until the newly selected constraint $\bar{\mathbf{y}}'$ is violated by no more than ϵ.

1) Optimization via the CCCP: In each iteration of the *cutting plane* algorithm, we need to solve problem (12) to obtain the optimal hypothesis $\bar{h}$ under the current working constraint subset Ω. Although the objective function in (12) is convex, the constraints are not, and this makes problem (12) difficult to solve. Fortunately, the *constrained concave-convex procedure (CCCP)* is just designed to solve those optimization problems with a concave-convex objective function under concave-convex constraints [22], [16].

Specifically, the *concave-convex procedure* (*CCP*) [22] is a method for solving non-convex optimization problem

whose objective function could be expressed as a difference of convex functions. It can be viewed as a special case of variational bounding [7] and related techniques including lower(upper) bound maximization(minimization) [11], surrogate functions and majorization [9]. While in [22] the authors only considered linear constraints, Smola et al. [16] proposed a generalization, the *constrained concave-convex procedure* (*CCCP*), for problems with a concave-convex objective function under concave-convex constraints.

Assume we are solving the following optimization problem [16]

$$\begin{aligned} \min_{\mathbf{z}} \quad & f_0(\mathbf{z}) - g_0(\mathbf{z}) \\ s.t. \quad & f_i(\mathbf{z}) - g_i(\mathbf{z}) \leq c_i \quad i = 1, \ldots, n \end{aligned} \tag{14}$$

where f_i and g_i are real-valued convex functions on a vector space $\mathcal{Z}$ and $c_i \in \mathcal{R}$ for all $i = 1, \ldots, n$. Denote by $T_1\{f, \mathbf{z}\}(\mathbf{z}')$ the first order *Taylor expansion* of f at location $\mathbf{z}$, that is $T_1\{f, \mathbf{z}\}(\mathbf{z}') = f(\mathbf{z}) + \partial_{\mathbf{z}} f(\mathbf{z})(\mathbf{z}' - \mathbf{z})$, where $\partial_{\mathbf{z}} f(\mathbf{z})$ is the gradient of the function f at $\mathbf{z}$. For non-smooth functions, it can be easily shown that the gradient $\partial_{\mathbf{z}} f(\mathbf{z})$ would be replaced by the subgradient [3]. Given an initial point $\mathbf{z}_0$, the *CCCP* computes $\mathbf{z}_{t+1}$ from $\mathbf{z}_t$ by replacing $g_i(\mathbf{z})$ with its first-order Taylor expansion at $\mathbf{z}_t$, i.e. $T_1\{g_i, \mathbf{z}_t\}(\mathbf{z})$, and setting $\mathbf{z}_{t+1}$ to the solution to the following relaxed optimization problem

$$\begin{aligned} \min_{\mathbf{z}} \quad & f_0(\mathbf{z}) - T_1\{g_0, \mathbf{z}_t\}(\mathbf{z}) \\ s.t. \quad & f_i(\mathbf{z}) - T_1\{g_i, \mathbf{z}_t\}(\mathbf{z}) \leq c_i \quad i = 1, \ldots, n \end{aligned} \tag{15}$$

The above procedure continues until $\mathbf{z}_t$ converges, and *Smola et al.* [16] proved that the *CCCP* is guaranteed to converge.

For problem (12), both the objective function and class balance constraint are convex. Moreover, constraint (13) is, although non-convex, the difference of two convex functions. Notice that while $\sum_{i=1}^{n} |\mathbf{w}^T \mathbf{x}_i| - \sum_{\widetilde{\mathbf{y}} \in \{-1,+1\}^n} z_{\widetilde{\mathbf{y}}} \bar{\Delta}(\bar{\mathbf{y}}', \widetilde{\mathbf{y}})$ is convex, it is a non-smooth function of $\mathbf{w}$. To use the *CCCP*, we need to calculate the *subgradients*:

$$\partial_{\mathbf{w}} \left\{ \sum_{i=1}^{n} |\mathbf{w}^T \mathbf{x}_i| - \sum_{\bar{\mathbf{y}} \in \bar{\mathcal{Y}}} z_{\bar{\mathbf{y}}} \bar{\Delta}(\bar{\mathbf{y}}', \bar{\mathbf{y}}) \right\} \Bigg|_{\mathbf{w}=\mathbf{w}_t} = \sum_{i=1}^{n} \mathrm{sgn}(\mathbf{w}^T \mathbf{x}_i) \mathbf{x}_i$$

Given an initial point $\mathbf{w}_0$, the *CCCP* computes $\mathbf{w}_{t+1}$ from $\mathbf{w}_t$ by replacing $\sum_{i=1}^{n} |\mathbf{w}^T \mathbf{x}_i| - \sum_{\widetilde{\mathbf{y}} \in \{-1,+1\}^n} z_{\widetilde{\mathbf{y}}} \bar{\Delta}(\bar{\mathbf{y}}', \widetilde{\mathbf{y}})$ in constraint (13) with its first-order Taylor expansion at $\mathbf{w}_t$, i.e.,

$$\begin{aligned} & \sum_{i=1}^{n} |\mathbf{w}_t^T \mathbf{x}_i| - \sum_{\bar{\mathbf{y}} \in \bar{\mathcal{Y}}} z_{\bar{\mathbf{y}}_t} \bar{\Delta}(\bar{\mathbf{y}}', \bar{\mathbf{y}}) + \sum_{i=1}^{n} \mathrm{sgn}(\mathbf{w}_t^T \mathbf{x}_i)(\mathbf{w} - \mathbf{w}_t)^T \mathbf{x}_i \\ = & \sum_{i=1}^{n} \mathrm{sgn}(\mathbf{w}_t^T \mathbf{x}_i) \mathbf{w}^T \mathbf{x}_i - \sum_{\bar{\mathbf{y}} \in \bar{\mathcal{Y}}} z_{\bar{\mathbf{y}}_t} \bar{\Delta}(\bar{\mathbf{y}}', \bar{\mathbf{y}}) \end{aligned} \tag{16}$$

By substituting the above first-order Taylor expansion into problem (12), we obtain the following *quadratic programming (QP)* problem:

$$\begin{aligned} \min_{\mathbf{w}, \xi} \quad & \frac{1}{2} \mathbf{w}^T \mathbf{w} + C\xi \\ \text{s.t. } \forall \bar{\mathbf{y}}' \in \bar{\mathcal{Y}} : & \sum_{i=1}^{n} \bar{y}_i' \mathbf{w}^T \mathbf{x}_i - \xi - \sum_{i=1}^{n} \mathrm{sgn}(\mathbf{w}_t^T \mathbf{x}_i) \mathbf{w}^T \mathbf{x}_i \\ & + \sum_{\bar{\mathbf{y}} \in \bar{\mathcal{Y}}} z_{\bar{\mathbf{y}}_t} \bar{\Delta}(\bar{\mathbf{y}}', \bar{\mathbf{y}}) \leq 0, \\ & \xi \geq 0, \\ & -l \leq \sum_{i=1}^{n} \mathbf{w}^T \mathbf{x}_i \leq l. \end{aligned} \tag{17}$$

The above *QP* problem can be solved in polynomial time. Following the *CCCP*, the obtained solution $(\mathbf{w}, \xi)$ from this *QP* problem is then used as $(\mathbf{w}_{t+1}, \xi_{t+1})$, and the iteration continues until convergence. The algorithm for solving problem (12) subject to the constraint subset Ω is summarized in Algorithm 2. As for its termination criterion, we check if the difference in objective values from two successive iterations is less than $\alpha\%$ (which is set to 0.01 in the experiments).

Algorithm 2 Solve problem (12) subject to constraint subset Ω via the CCCP.

Initialize $\mathbf{w}_0$.
repeat
  Obtain $(\mathbf{w}, \xi)$ as the solution to the *quadratic programming* problem (17).
  Set $\mathbf{w}_{t+1} = \mathbf{w}$ and $t = t + 1$.
until the stopping criterion is satisfied.

2) The Most Violated Constraint: In each iteration of the *cutting plane* algorithm, we need to locate the most violated constraint. Here, the feasibility of a constraint is measured by the corresponding value of ξ. Therefore, the most violated constraint is the one that would result in the largest ξ, i.e.,

$$\max_{\bar{\mathbf{y}}' \in \{-1,+1\}^n} \sum_{i=1}^{n} \bar{y}_i' \mathbf{w}^T \mathbf{x}_i + \bar{\Delta}(\bar{\mathbf{y}}', \bar{\mathbf{y}}), \tag{18}$$

where $\bar{\mathbf{y}}$ is the optimal label vector in (11). The tuple of labels $\bar{\mathbf{y}}'$ can take an exponential number of different values, and this renders an exhaustive search over all $\bar{\mathbf{y}}' \in \{-1, +1\}^n$ infeasible. Luckily, as in [5], the performance measures defined for clustering can all be computed from the contingency table (Table I).

In the following, we show how the performance measures for clustering can be calculated from the contingency table.

Table I. Contingency table for binary clustering.

	$y=1$	$y=-1$
$y'=1$	a	b
$y'=-1$	c	d

Normalized Mutual Information: Using the definition of *Normalized Mutual Information* in (2), it can be computed from the contingency table as

$$NMI=\Bigg\{a\log\frac{an}{(a+c)(a+b)}+b\log\frac{bn}{(b+d)(a+b)} +c\log\frac{cn}{(a+c)(c+d)}+d\log\frac{dn}{(b+d)(c+d)}\Bigg\}\Bigg/ \Bigg\{\sqrt{\Bigg[(a+c)\log\frac{a+c}{n}+(b+d)\log\frac{b+d}{n}\Bigg]} \sqrt{\Bigg[(a+b)\log\frac{a+b}{n}+(c+d)\log\frac{c+d}{n}\Bigg]}\Bigg\}.$$

The corresponding loss function is then defined as $\bar{\Delta}_{NMI}=1-NMI$.

Rand Index: Unlike *NMI*, the computation of *Rand index* is more complicated as it considers whether each pair of examples are correctly grouped together or not. Recall that from Table I, there are two clusters obtained, one for $y'=1$ and the other for $y'=-1$. For the cluster corresponding to $y'=1$, a patterns have true label $y=1$ while b patterns have true label $y=-1$. Let ${}_mC_2=\frac{m(m-1)}{2}$. There are thus a total of ${}_aC_2+{}_bC_2+ab$ pattern pairs in this cluster, of which ${}_aC_2+{}_bC_2$ pairs are true positives (i.e., pattern pairs that should be and are in the same cluster), while ab pairs are false positives (i.e., patterns pairs that are in the same cluster but should be in different clusters). Similarly, for the obtained cluster corresponding to $y'=-1$, ${}_cC_2+{}_dC_2$ pairs are true positives, while cd pairs are false positives. Hence,

$$TP={}_aC_2+{}_bC_2+{}_cC_2+{}_dC_2,$$

and

$$FP=ab+cd.$$

Similarly, TN counts the number of pattern pairs that should be and are in different clusters, and it is obvious that

$$TN=ad+bc.$$

Analogously, we also have

$$FN=ac+bd.$$

Hence, the *Rand Index* in (5) can be computed from the contingency table as:

$$RI=\frac{{}_aC_2+{}_bC_2+{}_cC_2+{}_dC_2+ad+bc}{{}_{a+b+c+d}C_2}.$$

Similar to the *NMI*, we define the corresponding loss function as $\bar{\Delta}_{RI}=1-RI$.

F-measure: Similar to the *Rand Index*, we first need to compute the *precision* and *recall*. Using (7) and the equations for TP, FP, TN, FN above, these can be computed from the contingency table as:

$$\begin{aligned}P &= \frac{{}_aC_2+{}_bC_2+{}_cC_2+{}_dC_2}{{}_aC_2+{}_bC_2+{}_cC_2+{}_dC_2+ab+cd},\\ R &= \frac{{}_aC_2+{}_bC_2+{}_cC_2+{}_dC_2}{{}_aC_2+{}_bC_2+{}_cC_2+{}_dC_2+ac+bd}.\end{aligned}$$

The F_β score can therefore be computed using (6), as

$$F_\beta=\frac{(\beta^2+1)s}{\beta^2(s+ac+bd)+(s+ab+cd)}, \qquad (19)$$

where $s={}_aC_2+{}_bC_2+{}_cC_2+{}_dC_2$. In particular, note that (19) is different from the computation of the F_β score in [5], as we are considering example pairs in this clustering context. The corresponding loss function is defined as $\bar{\Delta}_{FMeasure}=1-F_\beta$. Thus, all the performance measures designed for clustering can be directly computed from the contingency table.

The computation of argmax can be achieved by scanning all different contingency tables. Since the elements in the contingency table satisfies the conditions: $a+c=\#(y=1)$ and $b+d=\#(y=-1)$, there are only $O(n^2)$ different contingency tables for a binary clustering problem with n examples. Therefore, the loss function introduced in Section II-B can take at most $O(n^2)$ different values. Specifically, we denote the loss function value that is computed from the contingency table as $\bar{\Delta}(a,b,c,d)$. Our algorithm for locating the most violated constraint iterates over all possible contingency tables. Let $\bar{\mathcal{Y}}^n_{abcd}$ be the set containing all tuples $\bar{\mathbf{y}}\in\{-1,+1\}^n$ that results in contingency table (a,b,c,d) [5]. As the value of $\bar{\Delta}(a,b,c,d)$ is the same for all tuples in $\bar{\mathcal{Y}}^n_{abcd}$, maximization over $\bar{\mathcal{Y}}^n_{abcd}$ can be calculated as

$$\bar{\mathbf{y}}_{abcd}=\operatorname*{arg\,max}_{\bar{\mathbf{y}}'\in\bar{\mathcal{Y}}^n_{abcd}}\sum_{i=1}^n\bar{y}'_i\mathbf{w}^T\mathbf{x}_i.$$

The above objective function is a linear combination over the individual examples. Therefore, the solution $\bar{\mathbf{y}}_{abcd}$ can be computed elementwise. Specifically, for a particular contingency table (a,b,c,d), denote the set of examples with $y=1$ as $\mathcal{X}_+$ and the set of examples with $y=-1$ as $\mathcal{X}_-$. The maximum value of $\sum_{i=1}^n\bar{y}'_i\mathbf{w}^T\mathbf{x}_i$ is achieved when the a examples from $\mathcal{X}_+$ with the largest values of $\mathbf{w}^T\mathbf{x}_i$ are classified as positive, and the d examples from $\mathcal{X}_-$ with the lowest values of $\mathbf{w}^T\mathbf{x}_i$ are classified as

negative. Finally, the overall argmax can be computed by maximizing over the stratified maxima plus their constant loss $\bar{\Delta}(a, b, c, d)$ in $O(n^2)$ time [5].

C. Accuracy of the Cutting Plane Algorithm

The *cutting plane* algorithm selects a subset of constraints from the whole set of constraints in problem (12) and calculates an approximate solution. The following theorem characterizes the accuracy of the solution returned by the proposed algorithm.

Theorem 1: For any data set $\mathcal{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ and any $\epsilon > 0$, the cutting plane algorithm returns a point $(\mathbf{w}, \xi)$ for which $(\mathbf{w}, \xi + \epsilon)$ is feasible in problem (12).

Proof: The *cutting plane* algorithm terminates when the newly selected constraint satisfies the inequality

$$\forall \bar{\mathbf{y}}' \in \bar{\mathcal{Y}} : \mathbf{w}^T[\Psi(\bar{\mathbf{x}}, \bar{\mathbf{y}}) - \Psi(\bar{\mathbf{x}}, \bar{\mathbf{y}}')] \geq \bar{\Delta}(\bar{\mathbf{y}}', \bar{\mathbf{y}}) - (\xi + \epsilon). \quad (20)$$

If this holds, then since the newly selected constraint is the most violated one, all the other constraints will also satisfy inequality (20). Therefore, if $(\mathbf{w}, \xi)$ is the solution returned by the proposed algorithm, then $(\mathbf{w}, \xi + \epsilon)$ is a feasible solution to problem (12). ■

According to Theorem 1, ϵ indicates how close one wants the error rate of the best hypothesis to be, and it can thus be used as a stopping criterion.

D. Links with Conventional MMC

Denote the conventional *maximum margin clustering* with loss function being the error rate as $\mathrm{MMC}_{\mathbf{uni}}$, and the formulation proposed in this paper as $\mathrm{MMC}_{\mathbf{multi}}$. It would be interesting to study the relationship between these two formulations of *maximum margin clustering*. Specifically, we have the following theorem:

Theorem 2: When using the error rate as the loss function, $\mathrm{MMC}_{\mathbf{uni}}$ arises as a special case of $\mathrm{MMC}_{\mathbf{multi}}$.

Proof: (sketch) For $\mathrm{MMC}_{\mathbf{multi}}$ with loss function being error rate, define $\bar{\Delta} = 2error$ [5]. Specifically, we can show that for every $\mathbf{w}$, the smallest feasible ξ and $\sum_{i=1}^{n} \xi_i$ are related by $\xi = 2\sum_{i=1}^{n} \xi_i$. By selecting $C_{multi} = C_{uni}/2$, the two optimization problems have the same objective value and an equivalent set of constraints. ■

IV. Experiments

In this section, we study the clustering performance of the *maximum margin clustering* with multivariate loss function algorithm on several data sets. All the experiments are performed with MATLAB 7.0 on a 1.66GHZ Intel $\mathrm{Core}^{\mathrm{TM}}2$ Duo PC running Windows XP with 1.5GB main memory.

A. Data Sets

We use six data sets which are selected to cover a wide range of properties: **ionosphere, digits, letter** and **satellite** (from the UCI repository), **ringnorm**[1] and **20 newsgroup**[2]. For the **digits** data, we follow the experimental setup of [26] and focus on those pairs (3 vs 8, 1 vs 7, 2 vs 7, and 8 vs 9) that are difficult to differentiate. For the **letter and satellite** data sets, there are multiple classes and we use their first two classes only [26]. For the **20-newsgroup** data set, we choose the topic *rec* which contains *autos*, *motorcycles*, *baseball* and *hockey* from the version 20-news-18828. We preprocess the data in the same manner as in [29] and obtain 3970 document vectors in a 8014-dimensional space. Similar to the **digits** data set, we focus on those pairs (*autos vs. motorcycles (Text-1)*, and *baseball vs. hockey (Text-2)*) that are difficult to differentiate. A summary of all these data sets is in Table 2.

Table II. Descriptions of the data sets.

Data	Size	Dimension
digits1v7	361	64
digits2v7	356	64
digits3v8	357	64
digits8v9	354	64
ionosphere	354	64
ringnorm	1000	20
letterAvB	1555	16
satellite	2236	36
Text-1	1980	8014
Text-2	1989	8014

B. Clustering Performance

Besides our $\mathrm{MMC}_{\mathbf{multi}}$ algorithm, we also implement some other competitive algorithms and present their results for comparison. Specifically, we use **K-Means (KM)** and **Normalized Cut (NC)** [15] as baselines, and also compared with the **Maximum Margin Clustering** algorithm proposed in [26] which achieves state-of-the-art clustering accuracy among all *MMC* algorithms. For notational convenience, we denote this algorithm as $\mathrm{MMC}_{\mathbf{uni}}$. For **k-means**, the cluster centers are initialized randomly. For **NC**, the implementation is the same as in [15], and the width of the Gaussian kernel is set by exhaustive search from the grid $\{0.1\sigma_0, 0.2\sigma_0, \ldots, \sigma_0\}$, where σ_0 is the range of distance between any two data points in the dataset. Moreover, for $\mathrm{MMC}_{\mathbf{uni}}$, the implementation is

[1] http://ida.first.fraunhofer.de/projects/bench/

[2] http://people.csail.mit.edu/jrennie/20Newsgroups/

the same as in [26]. For $\text{MMC}_{\mathbf{uni}}$ and $\text{MMC}_{\mathbf{multi}}$, a linear kernel is employed and parameters are selected via grid search using the same grid. For **k-means**, $\text{MMC}_{\mathbf{uni}}$ and $\text{MMC}_{\mathbf{multi}}$, we present the results averaged over 50 random runs.

Table 3 shows the results for *Normalized Mutual Information* on the various data sets. As can be seen, our algorithm $\text{MMC}_{\mathbf{multi}}$ outperforms other competing algorithms on almost all data sets. Similarly, Table 4

Table III. Normalized Mutual Information comparisons.

Data	KM	NC	$\text{MMC}_{\mathbf{uni}}$	$\text{MMC}_{\mathbf{multi}}$
digits1v7	0.975	0.008	**1.00**	**1.00**
digits2v7	0.801	0.075	**1.00**	**1.00**
digits3v8	0.714	0.069	0.816	**0.861**
digits8v9	0.562	0.002	0.845	**0.860**
ionosphere	0.072	0.164	0.122	**0.197**
ringnorm	0.199	0.220	0.207	**0.422**
letterAvB	0.323	0.219	0.695	**0.703**
satellite	0.771	0.765	0.894	**0.923**
Text-1	0.021	0.722	0.761	**0.830**
Text-2	0.018	0.653	0.808	**0.900**

also demonstrates the advantage of $\text{MMC}_{\mathbf{multi}}$ over other clustering algorithms in terms of the *Rand Index*. For the

Table IV. Rand Index comparisons.

Data	KM	NC	$\text{MMC}_{\mathbf{uni}}$	$\text{MMC}_{\mathbf{multi}}$
digits1v7	0.995	0.504	**1.00**	**1.00**
digits2v7	0.940	0.550	**1.00**	**1.00**
digits3v8	0.904	0.545	0.945	**0.961**
digits8v9	0.835	0.500	0.956	**0.962**
ionosphere	0.564	0.626	0.599	**0.640**
ringnorm	0.635	0.653	0.642	**0.676**
letterAvB	0.706	0.644	0.873	**0.895**
satellite	0.922	0.919	0.971	**0.982**
Text-1	0.499	0.884	0.905	**0.939**
Text-2	0.500	0.842	0.929	**0.969**

F-measure, we set $\beta = 1.5$ and provide the clustering results in Table 5. Again, it is clear that $\text{MMC}_{\mathbf{multi}}$ achieves higher F_β. Clearly, experimental results shown in Table 3,4,5 demonstrate the advantage of $\text{MMC}_{\mathbf{multi}}$ in clustering accuracy over competing algorithms. Moreover, while $\text{MMC}_{\mathbf{uni}}$ may sometimes be inferior to *NC* (say, e.g. on **ionosphere** and **ringnorm** in table 3), $\text{MMC}_{\mathbf{multi}}$ is always the best.

C. CPU Time

One potential concern on $\text{MMC}_{\mathbf{multi}}$ is that its improvement on clustering accuracy might come from sacrificing time efficiency. Here, we compare the CPU time of $\text{MMC}_{\mathbf{multi}}$ (using $\bar{\Delta}_{RI}$) with the *maximum margin clustering* algorithm in [26], which is so far the most efficient *MMC* algorithm. As shown in Figure 1, the speed of $\text{MMC}_{\mathbf{multi}}$ is comparable to that of $\text{MMC}_{\mathbf{uni}}$. Indeed, $\text{MMC}_{\mathbf{multi}}$ even converges faster than $\text{MMC}_{\mathbf{uni}}$ on several data sets. However, unlike $\text{MMC}_{\mathbf{uni}}$ which can only employ the error rate as the loss function, $\text{MMC}_{\mathbf{multi}}$ can handle much more complicated loss functions.

Table V. F-measure comparisons (with $\beta = 1.5$).

Data	KM	NC	$\text{MMC}_{\mathbf{uni}}$	$\text{MMC}_{\mathbf{multi}}$
digits1v7	0.994	0.503	**1.00**	**1.00**
digits2v7	0.940	0.549	**1.00**	**1.00**
digits3v8	0.904	0.544	0.945	**0.961**
digits8v9	0.835	0.498	0.956	**0.962**
ionosphere	0.594	0.652	0.627	**0.672**
ringnorm	0.689	0.685	0.688	**0.724**
letterAvB	0.706	0.644	0.897	**0.900**
satellite	0.944	0.942	0.978	**0.986**
Text-1	0.761	0.887	0.907	**0.940**
Text-2	0.762	0.849	0.931	**0.970**

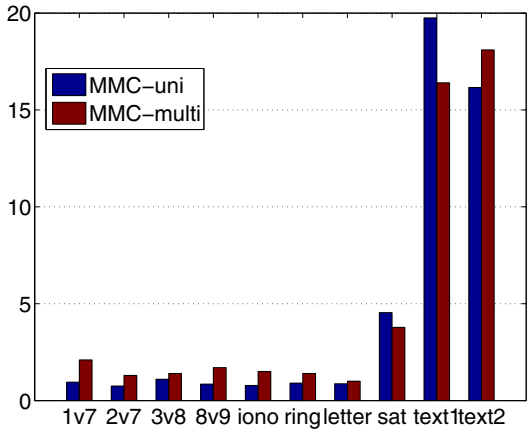

Figure 1. CPU time (in seconds) comparisons of $\text{MMC}_{\mathbf{multi}}$ and $\text{MMC}_{\mathbf{uni}}$.

V. Conclusions

We propose *maximum margin clustering (MMC)* that directly optimizes multivariate performance measure defined for clustering, including *Normalized Mutual Information, Rand Index* and *F-measure*. To tackle the exponential number of constraints involved in the optimization problem, we employ *cutting plane* algorithm to efficiently solve the problem. Preliminary theoretical analysis of the

algorithm is provided, where we show that the formulation for *maximum margin clustering* proposed in this paper generalizes the conventional *MMC*. Experimental evaluations show clear improvements in clustering performance of our method over previous *maximum margin clustering* algorithms.

In the future, we plan to extend the current method to multi-cluster problems. Note that while a direct multi-cluster extension is possible, it is complicated by the fact that it might involve an exponential number of contingency tables while finding the most violated constraint. Therefore, an efficient strategy for searching in this space of contingency tables is required.

Acknowledgments

This work is supported by projects (60835002) and (60675009) of the National Natural Science Foundation of China.

References

[1] C. Burges, R. Ragno, and Q. Le. Learning to rank with nonsmooth cost functions. In *Advances in Neural Information Processing Systems 18*, 2006.

[2] O. Chapelle and A. Zien. Semi-supervised classification by low density separation. In *AISTATS*, 2005.

[3] P. M. Cheung and J. T. Kwok. A regularization framework for multiple-instance learning. In *ICML*, 2006.

[4] T. Finley and T. Joachims. Supervised clustering with support vector machines. In *Proceedings of the 22nd international conference on Machine learning*, 2005.

[5] T. Joachims. A support vector method for multivariate performance measures. In *Proceedings of the Twenty-Second International Conference on Machine Learning*, pages 377–384, Bonn, Germany, August 2005.

[6] T. Joachims. Training linear SVMs in linear time. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pages 217–226, Philadelphia, PA, USA, 2006.

[7] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.

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

[9] K. Lange, D.R. Hunter, and I. Yang. Optimization transfer using surrogate objective functions. *Jounal of Computational and Graphical Statistics*, 9:1–59, 2000.

[10] Q. Le and A. Smola. Direct optimization of ranking measures. Technical report, NICTA, 2008.

[11] S. P. Luttrell. Partitioned mixture distributions: An adaptive bayesian network for low-level image processing. In *IEEE Proceedings on Vision, Image and Signal Processing*, volume 141, pages 251–260, 1994.

[12] C. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[13] W. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66:846–850, 1971.

[14] B. Schölkopf, A. J. Smola, and K. R. Müller. Kernel principal component analysis. *Advances in kernel methods: Support vector learning*, pages 327–352, 1999.

[15] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[16] A. J. Smola, S.V.N. Vishwanathan, and T. Hofmann. Kernel methods for missing variables. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, Barbados, January 2005.

[17] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

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

[19] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. In *Advances in Neural Information Processing Systems 17*. MIT Press, 2005.

[20] L. Xu and D. Schuurmans. Unsupervised and semi-supervised multi-class support vector machines. In *Proceedings of The Twentieth National Conference on Artificial Intelligence*, pages 904–910, Pittsburgh, PA, USA, 2005.

[21] Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *Proceedings of the International Conference on Research and Development in Information Retrieval*, pages 271–278, Amsterdam, The Netherlands, 2007. MIT Press.

[22] A. Yuille and A. Rangarajan. The concave-convex procedure. *Neural Computation*, 15:915–936, 2003.

[23] K. Zhang, I. W. Tsang, and J. T. Kwok. Maximum margin clustering made practical. In *Proceedings of the Twenty-Fourth International Conference on Machine Learning*, Corvallis, Oregon, USA, June 2007.

[24] B. Zhao, J. Kwok, and C. Zhang. Multiple kernel clustering. In *Proceedings of the 9th SIAM International Conference on Data Mining*, 2009.

[25] B. Zhao, F. Wang, and C. Zhang. Efficient maximum margin clustering via cutting plane algorithm. In *Proceedings of SIAM International Conference on Data Mining*, 2008.

[26] B. Zhao, F. Wang, and C. Zhang. Efficient multiclass maximum margin clustering. In *Proceedings of the Twenty-Fifth International Conference on Machine Learning*, pages 1248–1255, Helsinki, Finland, July 2008.

[27] B. Zhao, F. Wang, and C. Zhang. Maximum margin embedding. In *Proceedings of the 8th IEEE International Conference on Data Mining*, 2008.

[28] S. Zhong and J. Ghosh. A unified framework for model-based clustering. *Journal of Machine Learning Research*, 4:1001–1037, 2003.

[29] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *Advances in Neural Information Processing Systems 16*. MIT Press, 2004.