Proceedings of the ICML-2003 Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining Systems, pp.42-49, Washington DC, August, 2003

Comparing and Unifying Search-Based and Similarity-Based Approaches to Semi-Supervised Clustering

Sugato Basu SUGATO@CS.UTEXAS.EDU
Mikhail Bilenko MBILENKO@CS.UTEXAS.EDU
Raymond J. Mooney MOONEY@CS.UTEXAS.EDU
Department of Computer Sciences, University of Texas, Austin, TX 78712

Abstract

Semi-supervised clustering employs a small amount of labeled data to aid unsupervised learning. Previous work in the area has employed one of two approaches: 1) Search-based methods that utilize supervised data to guide the search for the best clustering, and 2) Similarity-based methods that use supervised data to adapt the underlying similarity metric used by the clustering algorithm. This paper presents a unified approach based on the K-Means clustering algorithm that incorporates *both* of these techniques. Experimental results demonstrate that the combined approach generally produces better clusters than either of the individual approaches.

1. Introduction

In many learning tasks, there is a large supply of unlabeled data but limited labeled data since it can be expensive to generate. Consequently, *semi-supervised learning*, learning from a combination of both labeled and unlabeled data, has become a topic of significant recent interest (Blum & Mitchell, 1998; Joachims, 1999; Nigam et al., 2000). More specifically, *semi-supervised clustering*, the use of class labels or constraints[1] on some examples to aid standard unsupervised clustering, has been the focus of several projects in the past few years (Wagstaff et al., 2001; Basu et al., 2002; Klein et al., 2002; Xing et al., 2003).

Existing methods for semi-supervised clustering fall into two general approaches that we call *search-based* and *similarity-based*. In search-based approaches, the clustering algorithm itself is modified so that user-provided labels or constraints are used to bias the search for an appropriate partition. This can be done by modifying the objective function for evaluating clusterings so that it includes satisfying constraints (Demiriz et al., 1999), enforcing constraints during the clustering process (Wagstaff et al., 2001), and/or initializing clusters based on labeled examples (Basu et al., 2002). In similarity-based approaches, an existing clustering algorithm that uses a similarity metric is employed; however, the similarity metric is first trained to satisfy the labels or constraints in the supervised data. Several similarity metrics have been used for similarity-based semi-supervised clustering including string-edit distance trained using EM (Bilenko & Mooney, 2003), KL divergence trained using gradient descent (Cohn et al., 2000), Euclidean distance modified by a shortest-path algorithm (Klein et al., 2002), or Mahalanobis distances trained using convex optimization (Xing et al., 2003). Several clustering algorithms using trained similarity metrics have been employed for semi-supervised clustering, including single-link (Bilenko & Mooney, 2003) and complete-link (Klein et al., 2002) agglomerative clustering, EM (Cohn et al., 2000), and K-Means (Xing et al., 2003).

Unfortunately, similarity-based and search-based approaches to semi-supervised clustering have not been adequately compared experimentally, so their relative strengths and weaknesses are largely unknown. Also, the two approaches are not incompatible, therefore, applying a search-based approach with a trained similarity metric is clearly an additional option which may have advantages over both existing approaches. In this paper, we present a new unified semi-supervised clustering algorithm derived from K-Means that incorporates *both* metric learning and using labeled data as seeds and/or constraints. By ablating the similarity-based and search-based components in this unified method, we present experimental results comparing and combining the two approaches. Both methods for

[1] Constraints typically specify that two examples must be in the same class (*must-link*) or must be in different classes (*cannot-link*).

semi-supervision individually can improve clustering accuracy, although metric learning requires sufficient labeled data to be beneficial. Finally, when metric learning is helpful, combining it with seeding and constraints results in even better performance than either approach alone.

2. Problem Formulation

2.1. Clustering with K-Means

K-Means is a clustering algorithm based on iterative relocation that partitions a dataset into K clusters, locally minimizing the total distance between the data points and the cluster centroids. Let $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^N, \mathbf{x}_i \in \mathbb{R}^m$ be a set of data points, x_{id} be the d-th component of $\mathbf{x}_i$, $\{\boldsymbol{\mu}_h\}_{h=1}^K$ represent the K cluster centroids, and l_i be the cluster assignment of a point $\mathbf{x}_i$, where $l_i \in \mathcal{L}$ and $\mathcal{L} = \{1, \ldots, K\}$. The Euclidean K-Means algorithm creates a K-partitioning[2] $\{\mathcal{X}_l\}_{l=1}^K$ of $\mathcal{X}$ so that the objective function $\sum_{\mathbf{x}_i \in \mathcal{X}} \|\mathbf{x}_i - \boldsymbol{\mu}_{l_i}\|^2$ is locally minimized.

It can be shown that the K-Means algorithm is essentially an EM algorithm on a mixture of K Gaussians under assumptions of identity covariance of the Gaussians, uniform priors of the mixture components and expectation under a particular conditional distribution (Basu et al., 2002). If $\mathcal{X}$ denotes the observed data, Θ denotes the current estimate of the parameter values of the mixture of Gaussians model and $\mathcal{L}$ denotes the missing data, then in the E-step the EM algorithm computes the expected value of the complete-data log-likelihood $\log p(\mathcal{X}, \mathcal{L}|\Theta)$ over the conditional distribution $p(\mathcal{L}|\mathcal{X}, \Theta)$ (Bilmes, 1997). Maximizing the complete data log-likelihood under the assumptions specified above can be shown to be equivalent to minimizing the K-Means objective function. In the Euclidean K-Means formulation, the distance between a point $\mathbf{x}_i$ and its corresponding cluster centroid $\boldsymbol{\mu}_{l_i}$ is calculated using the square of the Euclidean distance $\|\mathbf{x}_i - \boldsymbol{\mu}_{l_i}\|^2 = (\mathbf{x}_i - \boldsymbol{\mu}_{l_i})^T(\mathbf{x}_i - \boldsymbol{\mu}_{l_i})$. This measure of distance is a direct consequence of the identity covariance assumption of the underlying Gaussians.

2.2. Semi-supervised Clustering with Constraints

In a *semi-supervised clustering* setting, a small amount of labeled data is available to aid the unsupervised clustering process. For pairwise constrained clustering, we consider a framework that has pairwise must-link and cannot-link constraints (with an associated cost of violating each constraint) between points in a dataset, in addition to having distances between the points (Wagstaff et al., 2001). Supervision in the form of constraints is generally more practical than providing class labels in the clustering framework, since true labels may be unknown a priori, while a human expert can easily specify whether pairs of points belong to the same cluster or different clusters.

Since K-Means clustering cannot handle pairwise constraints explicitly, we formulate the goal of clustering in the pairwise constrained clustering framework as minimizing a combined objective function, which is defined as the sum of the total square distances between the points and their cluster centroids and the cost of violating the pairwise constraints. The mathematical formulation of this framework is motivated by the *metric labeling* problem and the *generalized Potts* model (Kleinberg & Tardos, 1999; Boykov et al., 1998).

In the pairwise constrained clustering framework, let $\mathcal{M}$ be the set of unordered must-link pairs such that $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{M}$ implies $\mathbf{x}_i$ and $\mathbf{x}_j$ should be assigned to the same cluster, and $\mathcal{C}$ be the set of unordered cannot-link pairs such that $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{C}$ implies $\mathbf{x}_i$ and $\mathbf{x}_j$ should be assigned to different clusters. Let $W = \{w_{ij}\}$ and $\overline{W} = \{\overline{w}_{ij}\}$ be two sets that give the weights corresponding to the must-link constraints in $\mathcal{M}$ and the cannot-link constraints in $\mathcal{C}$ respectively. Let d_M and d_C be two metrics that quantify the cost of violating must-link and cannot-link constraints: $d_M(l_i, l_j) = \mathbb{1}[l_i \neq l_j]$ and $d_C(l_i, l_j) = \mathbb{1}[l_i = l_j]$, where $\mathbb{1}$ is the indicator function ($\mathbb{1}[true] = 1$, $\mathbb{1}[false] = 0$) and l_i are the cluster labels. Using this model, the problem of pairwise constrained clustering under must-link and cannot-link constraints is formulated as minimizing the following objective function, where point $\mathbf{x}_i$ is assigned to the partition $\mathcal{X}_{l_i}$ with centroid $\boldsymbol{\mu}_{l_i}$:

$$\begin{aligned}\mathcal{J}_{\text{pckm}} &= \sum_{\mathbf{x}_i \in \mathcal{X}} \|\mathbf{x}_i - \boldsymbol{\mu}_{l_i}\|^2 + \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{M}} w_{ij} \mathbb{1}[l_i \neq l_j] \\ &\quad + \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{C}} \overline{w}_{ij} \mathbb{1}[l_i = l_j] \qquad (1)\end{aligned}$$

We will refer to this model as the pairwise constrained K-Means (PC-KMeans) model.

2.3. Semi-supervised Clustering via Metric Learning

Another avenue for utilizing labeled data involves adapting the distance metric employed by the clustering algorithm. Intuitively, this allows capturing the user's view of which objects should be considered similar and which dissimilar. Since the original data representation may not be embedded in a space where clusters are sufficiently separated, modifying the distance metric transforms the representation so that dis-

[2] K disjoint subsets of $\mathcal{X}$, whose union is $\mathcal{X}$

tances between same-cluster objects are minimized, while distances between different-cluster objects are maximized. As a result, clusters discovered using the learned distance metrics adhere more closely to the notion of similarity expressed by the labeled data than the clusters obtained using untrained distance metrics.

Following previous work (Xing et al., 2003), we can parameterize Euclidean distance with a symmetric positive-definite weight matrix $\mathbf{A}$ as follows: $\|\mathbf{x}_i - \mathbf{x}_j\|_{\mathbf{A}} = \sqrt{(\mathbf{x}_i - \boldsymbol{\mu}_{l_i})^T \mathbf{A} (\mathbf{x}_i - \boldsymbol{\mu}_{l_i})}$. If $\mathbf{A}$ is restricted to be a diagonal matrix, then it scales each axis by a different weight and corresponds to feature weighting; otherwise new features are created that are linear combinations of the original features. In our clustering formulation, using the matrix $\mathbf{A}$ is equivalent to considering a generalized version of the K-Means model described in Section 2.1, where all the Gaussians have a covariance matrix $\mathbf{A}^{-1}$ (Bilmes, 1997).

It can be easily shown that maximizing the complete data log-likelihood under this generalized K-Means model is equivalent to minimizing the objective function:

$$\mathcal{J}_{\text{mkmeans}} = \sum_{\mathbf{x}_i \in \mathcal{X}} \|\mathbf{x}_i - \boldsymbol{\mu}_{l_i}\|^2_{\mathbf{A}} - \log(\det(\mathbf{A})) \tag{2}$$

where the second term arises due to the normalizing constant of a Gaussian with covariance matrix $\mathbf{A}^{-1}$.

2.4. Unifying Constraints and Metric Learning in Clustering

Previous work on semi-supervised clustering (Cohn et al., 2000; Xing et al., 2003) that used labeled data for learning a metric only utilized the pairwise constraint information to learn weights that minimize constraint violations. We propose to incorporate metric learning directly into the clustering algorithm in a way that allows unlabeled data to influence the metric learning process along with pairwise constraints.

Combining objective functions (1) and (2) leads to the following objective function that attempts to minimize cluster dispersion under a learned metric along with minimizing the number of constraint violations:

$$\begin{aligned}\mathcal{J}_{\text{combined}} &= \sum_{\mathbf{x}_i \in \mathcal{X}} \|\mathbf{x}_i - \boldsymbol{\mu}_{l_i}\|^2_{\mathbf{A}} + \sum_{(\mathbf{x}_i,\mathbf{x}_j) \in \mathcal{M}} w_{ij} \mathbb{1}[l_i \neq l_j] \\ &+ \sum_{(\mathbf{x}_i,\mathbf{x}_j) \in \mathcal{C}} \overline{w}_{ij} \mathbb{1}[l_i = l_j] - \log(\det(\mathbf{A})) \quad (3)\end{aligned}$$

If we assume uniform weights w_{ij} and $\overline{w}_{ij}$, as traditionally done in the generalized Potts model (Boykov et al., 1998), one problem with this objective function would be that all constraint violations are treated equally. However, the cost of violating a must-link constraint between two *close* points should be higher than the cost of violating a must-link constraint between two points that are *far apart*. Such cost assignment reflects the intuition that it is a "worse error" to violate a must-link constraint between similar points, and such an error should have more impact on the metric learning framework. Multiplying the weights w_{ij} with the penalty function $f_M(\mathbf{x}_i, \mathbf{x}_j) = \max(\alpha_{\min}, \alpha_{\max} - \|\mathbf{x}_i - \mathbf{x}_j\|^2_{\mathbf{A}})$ gives us the overall cost of violating a must-link constraint between two points $\mathbf{x}_i$ and $\mathbf{x}_j$, where $\alpha_{\min}$ and $\alpha_{\max}$ are non-negative constants that correspond to minimum and maximum penalties respectively. They can be set as fractions of the square of the maximum must-link distance $\max_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{M}} \|\mathbf{x}_i - \mathbf{x}_j\|^2$, thus guaranteeing that the penalty for violating a constraint is always positive. Overall, this formulation enables the penalty for violating a must-link constraint to be proportional to the "seriousness" of the violation.

Analogously, the cost of violating a cannot-link constraint between two *distant* points should be higher than the cost of violating a cannot-link constraint between points that are *close*, since the former is a "worse error". Multiplying weights $\overline{w}_{ij}$ with $f_C(\mathbf{x}_i, \mathbf{x}_j) = \min(\alpha_{\min} + \|\mathbf{x}_i - \mathbf{x}_j\|^2_{\mathbf{A}}, \alpha_{\max})$ allows us to take the "seriousness" of the constraint violation into account. The combined objective function then becomes:

$$\begin{aligned}\mathcal{J}_{\text{mpckm}} = &\sum_{\mathbf{x}_i \in \mathcal{X}} \|\mathbf{x}_i - \boldsymbol{\mu}_{l_i}\|^2_{\mathbf{A}} - \log(\det(\mathbf{A})) \\ &+ \sum_{(\mathbf{x}_i,\mathbf{x}_j) \in \mathcal{M}} w_{ij} f_M(\mathbf{x}_i, \mathbf{x}_j) \mathbb{1}[l_i \neq l_j] \\ &+ \sum_{(\mathbf{x}_i,\mathbf{x}_j) \in \mathcal{C}} \overline{w}_{ij} f_C(\mathbf{x}_i, \mathbf{x}_j) \mathbb{1}[l_i = l_j] \end{aligned} \tag{4}$$

The weights w_{ij} and $\overline{w}_{ij}$ provide a way to specify the relative importance of the unlabeled versus labeled data while allowing individual constraint weights. This objective function $\mathcal{J}_{\text{mpckm}}$ is greedily optimized by our proposed metric pairwise constrained K-Means (MPC-KMeans) algorithm that uses a K-Means-type iteration.

3. Algorithm

Given a set of data points $\mathcal{X}$, a set of must-link constraints $\mathcal{M}$, a set of cannot-link constraints $\mathcal{C}$, corresponding weight sets W and $\overline{W}$, and the number of clusters to form K, metric pairwise constrained K-Means (MPC-KMeans) finds a disjoint K partitioning $\{\mathcal{X}_h\}_{h=1}^K$ of $\mathcal{X}$ (with each partition having a centroid $\boldsymbol{\mu}_h$) such that $\mathcal{J}_{\text{mpckm}}$ is (locally) minimized.

The algorithm MPC-KMeans has two components. Utilizing constraints during cluster initialization and satisfaction of the constraints during every cluster assignment step constitutes the search-based component of the algorithm. Learning the distance metric by re-estimating the weight matrix $\mathbf{A}$ during each algorithm iteration based on current constraint violations is the similarity-based component.

Intuitively, the search-based technique uses the pairwise constraints to generate seed clusters that initialize the clustering algorithm, and also uses the constraints to guide the clustering process through the iterations. Seeds inferred from the constraints bias the clustering towards a good region of the search space, thereby possibly reducing the chances of it getting stuck in poor local optima, while a clustering that satisfies the user-specified constraints is produced simultaneously.

The similarity-based technique distorts the metric space to minimize the costs of violated constraints, possibly removing the violations in the subsequent iterations. Implicitly, the space where data points are embedded is transformed to respect the user-provided constraints, thus capturing the notion of similarity appropriate for the dataset from the user's perspective.

3.1. Initialization

To generate the seed clusters during the initialization step of MPC-KMeans, we take the transitive closure of the must-link constraints (Wagstaff et al., 2001) and augment the set $\mathcal{M}$ by adding these entailed constraints, assuming consistency of the constraints. Let the number of connected components in the augmented set $\mathcal{M}$ be λ. These λ connected components are used to create λ neighborhood sets $\{N_p\}_{p=1}^{\lambda}$, where each neighborhood set consists of points connected by must-links from the augmented set $\mathcal{M}$. For every pair of neighborhoods N_p and $N_{p'}$ that have at least one cannot-link between them, we add cannot-link constraints between every pair of points in N_p and $N_{p'}$ and augment the cannot-link set $\mathcal{C}$ by these entailed constraints. We will overload notation from this point and refer to the augmented must-link and cannot-link sets as $\mathcal{M}$ and $\mathcal{C}$ respectively.

Note that the neighborhood sets N_p, which contain the neighborhood information inferred from the must-link constraints and are unchanged during the iterations of the algorithm, are different from the partition sets $\mathcal{X}_h$, which contain the cluster partitioning information and get updated at each iteration of the algorithm.

After this preprocessing step, we get λ neighborhood sets $\{N_p\}_{p=1}^{\lambda}$. These neighborhoods provide a good initial starting point for the MPC-KMeans algorithm. If $\lambda \geq K$, where K is the required number of clusters, we select the K neighborhood sets of largest size and initialize the K cluster centers with the centroids of these sets. If $\lambda < K$, we initialize λ cluster centers with the centroids of the λ neighborhood sets. We then look for a point $\mathbf{x}$ that is connected by cannot-links to every neighborhood set. If such a point exists, it is used to initialize the $(\lambda+1)^{th}$ cluster. If there are any more cluster centroids left uninitialized, we initialize them by random points obtained by random perturbations of the global centroid of $\mathcal{X}$.

3.2. E step

MPC-KMeans alternates between cluster assignment in the E-step, and centroid estimation and metric learning in the M-step (see Figure 1).

In the E-step of MPC-KMeans, every point $\mathbf{x}$ is assigned to a cluster so that the sum of the distance of $\mathbf{x}$ to the cluster centroid and the cost of constraint violations possibly incurred by this cluster assignment is minimized. Note that this assignment step is order-dependent, since the subsets of $\mathcal{M}$ and $\mathcal{C}$ associated with each cluster may change with the assignment of a point. In the cluster assignment step, each point moves to a new cluster only if the component of $\mathcal{J}_{\text{mpckm}}$ contributed by this point decreases. So when all points are given their new assignment, $\mathcal{J}_{\text{mpckm}}$ will decrease or remain the same.

3.3. M step

In the M-step, the cluster centroids $\boldsymbol{\mu}_h$ are first re-estimated using the points in $\mathcal{X}_h$. As a result, the contribution of each cluster to $\mathcal{J}_{\text{mpckm}}$ is minimized. The pairwise constraints do not take in part in this centroid re-estimation step because the constraints are not an explicit function of the centroid. Thus, only the first term (the distance component) of $\mathcal{J}_{\text{mpckm}}$ is minimized in this step. The centroid re-estimation step effectively remains the same as K-Means.

The second part of the M-step is metric learning, where the matrix $\mathbf{A}$ is re-estimated to decrease the objective function $\mathcal{J}_{\text{mpckm}}$. The updated matrix $\mathbf{A}$ is obtained by taking the partial derivative $\frac{\partial \mathcal{J}_{\text{mpckm}}}{\partial \mathbf{A}}$ and setting it to zero, resulting in:

$$\begin{aligned}\mathbf{A} = \; & \Bigg(\sum_{\mathbf{x}_i \in \mathcal{X}} (\mathbf{x}_i - \boldsymbol{\mu}_{l_i})(\mathbf{x}_i - \boldsymbol{\mu}_{l_i})^T \\ & - \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{M}^*} w_{ij}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T \mathbb{1}[l_i \neq l_j] \\ & + \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{C}^*} \overline{w}_{ij}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T \mathbb{1}[l_i = l_j] \Bigg)^{-1}\end{aligned}$$

where M^* and C^* are subsets of M and C that exclude the constraint pairs for which the penalty functions f_M and f_C take the threshold values α_{min} and α_{max} respectively. See Appendix A for the details of the derivation.

Since estimating a full matrix $\mathbf{A}$ from limited training data is difficult, we limit ourselves to diagonal $\mathbf{A}$, which is equivalent to learning a metric via feature weighting. In that case, the d-th diagonal element of $\mathbf{A}$, a_{dd}, corresponds to the weight of the d-th feature:

$$a_{dd} = \Bigg(\sum_{\mathbf{x}_i \in \mathcal{X}} (\mathbf{x}_{id} - \boldsymbol{\mu}_{l_i d})^2 - \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{M}^*} w_{ij}(\mathbf{x}_{id} - \mathbf{x}_{jd})^2 \mathbb{1}[l_i \neq l_j] + \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{C}^*} \overline{w}_{ij}(\mathbf{x}_{id} - \mathbf{x}_{jd})^2 \mathbb{1}[l_i = l_j] \Bigg)^{-1}$$

Intuitively, the first term in the sum, $\sum_{\mathbf{x}_i \in \mathcal{X}} (\mathbf{x}_{id} - \boldsymbol{\mu}_{l_i d})^2$, scales the weight of each feature proportionately to the feature's contribution to the overall cluster dispersion, analogously to scaling performed when computing Mahalanobis distance. The second two terms that depend on constraint violations, $-\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{M}^*} w_{ij}(\mathbf{x}_{id} - \mathbf{x}_{jd})^2 \mathbb{1}[l_i \neq l_j]$ and $\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{C}^*} \overline{w}_{ij}(\mathbf{x}_{id} - \mathbf{x}_{jd})^2 \mathbb{1}[l_i = l_j]$, respectively contract and stretch each dimension attempting to mend the current violations. Thus, the metric weights are adjusted at each iteration in such a way that the contribution of different attributes to distance is equalized, while constraint violations are minimized.

The objective function decreases after every cluster assignment, centroid re-estimation and metric learning step till convergence, implying that the MPC-KMeans algorithm will converge to a local minima of $\mathcal{J}_{\text{mpckm}}$.

Algorithm: `m-PCKMeans`
Input: Set of data points $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^N$,
set of *must-link* constraints $\mathcal{M} = \{(\mathbf{x}_i, \mathbf{x}_j)\}$,
set of *cannot-link* constraints $\mathcal{C} = \{(\mathbf{x}_i, \mathbf{x}_j)\}$,
number of clusters K, sets of constraint weights W and $\overline{W}$.
Output: Disjoint K partitioning $\{\mathcal{X}_h\}_{h=1}^K$ of $\mathcal{X}$ such that objective function $\mathcal{J}_{\text{mpckm}}$ is (locally) minimized.
Method:
1. Initialize clusters:
1a. create the λ neighborhoods $\{N_p\}_{p=1}^\lambda$ from $\mathcal{M}$ and $\mathcal{C}$
1b. sort the indices p in decreasing size of N_p
1c. if $\lambda \geq K$
  initialize $\{\boldsymbol{\mu}_h^{(0)}\}_{h=1}^K$ with centroids of $\{N_p\}_{p=1}^K$
  else if $\lambda < K$
  initialize $\{\boldsymbol{\mu}_h^{(0)}\}_{h=1}^\lambda$ with centroids of $\{N_p\}_{p=1}^\lambda$
  if $\exists$ point $\mathbf{x}$ *cannot-linked* to all neighborhoods $\{N_p\}_{p=1}^\lambda$
   initialize $\boldsymbol{\mu}_{\lambda+1}^{(0)}$ with $\mathbf{x}$
  initialize remaining clusters at random
2. Repeat until *convergence*
2a. `assign_cluster`: Assign each data point $\mathbf{x}_i$ to cluster h^* (i.e. set $\mathcal{X}_{h^*}^{(t+1)}$), for $h^* = \arg\min_h(\|\mathbf{x}_i - \boldsymbol{\mu}_h^{(t)}\|_{\mathbf{A}}^2 - \log(\det(\mathbf{A})) + \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{M}} w_{ij} f_M(\mathbf{x}_i,\mathbf{x}_j)\mathbb{1}[h \neq l_j] + \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{C}} \overline{w}_{ij} f_C(\mathbf{x}_i,\mathbf{x}_j)\mathbb{1}[h = l_j])$
2b. `estimate_means`: $\{\boldsymbol{\mu}_h^{(t+1)}\}_{h=1}^K \leftarrow \{\frac{1}{|\mathcal{X}_h^{(t+1)}|}\sum_{\mathbf{x}\in\mathcal{X}_h^{(t+1)}} \mathbf{x}\}_{h=1}^K$
2c. `update_metric`: $\mathbf{A}^{-1} = \sum_{\mathbf{x}_i\in\mathcal{X}} (\mathbf{x}_i - \boldsymbol{\mu}_{l_i})(\mathbf{x}_i - \boldsymbol{\mu}_{l_i})^T - \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{M}^*} w_{ij}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T \mathbb{1}[l_i \neq l_j] + \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{C}^*} \overline{w}_{ij}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T \mathbb{1}[l_i = l_j]$
2d. $t \leftarrow (t+1)$

Figure 1. MPC-KMeans algorithm

4. Experiments

4.1. Methodology and Datasets

Experiments were conducted on several datasets from the UCI repository: *Iris*, *Wine*, and representative randomly sampled subsets from the *Pen-Digits* and *Letter* datasets. For *Pen-Digits* and *Letter*, we chose two sets of three classes: $\{\mathbf{I, J, L}\}$ from *Letter* and $\{\mathbf{3, 8, 9}\}$ from *Pen-Digits*, sampling 20% of the data points from the original datasets randomly. These classes were chosen from the handwriting recognition datasets since they intuitively represent difficult visual discrimination problems.

We have used pairwise F-measure to evaluate the clustering results. It is based on the traditional information retrieval measures, adapted for evaluating clustering by considering same-cluster pairs:

$$Precision = \frac{\#PairsCorrectlyPredictedInSameCluster}{TotalPairsPredictedInSameCluster}$$

$$Recall = \frac{\#PairsCorrectlyPredictedInSameCluster}{TotalPairsInSameCluster}$$

$$F-measure = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

We generated learning curves with 10-fold cross-validation for each dataset to determine the effect of utilizing the pairwise constraints. Each point in the learning curve represents a particular number of pairwise constraints given as input to the algorithm. Unit constraint weights W and $\overline{W}$ were used, since the datasets did not provide individual weights for the constraints. The maximum square distance between must-link constraints was used as value for $\alpha_{\max}$, while $\alpha_{\min}$ was set to 0. The clustering algorithm is run on the whole dataset, but the pairwise F-measure is calculated only on the test set. Results were averaged over 50 runs of 10 folds.

4.2. Results and Discussion

Figs.2-5 show learning curves for the four datasets. For each dataset, we compared four semi-supervised clustering schemes:

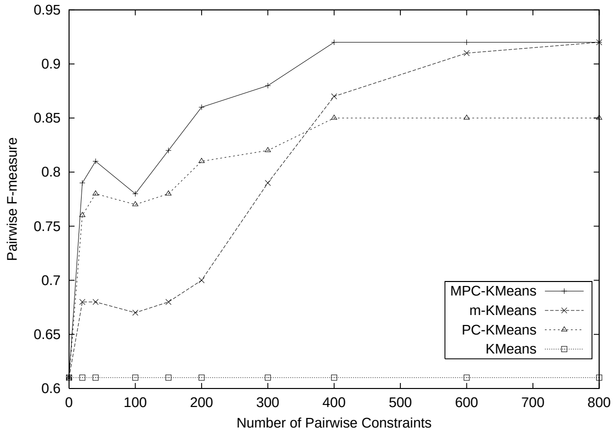

Figure 2. Results on the *Iris* dataset

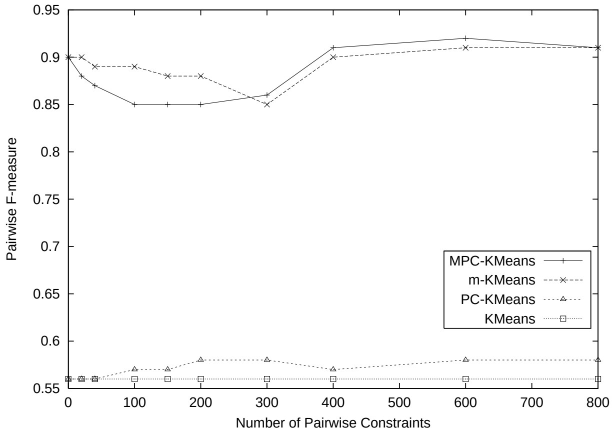

Figure 3. Results on the *Wine* dataset

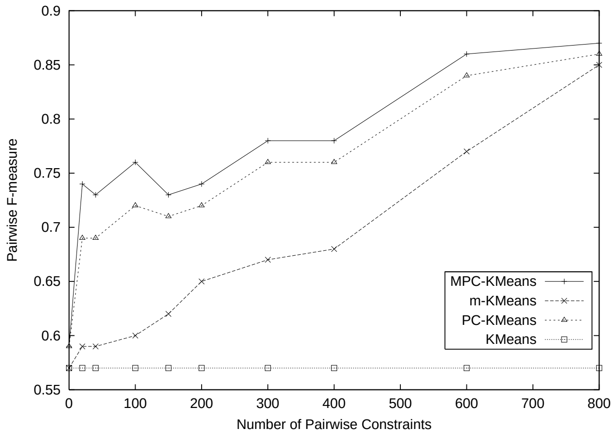

Figure 4. Results on the *Digits-389* dataset

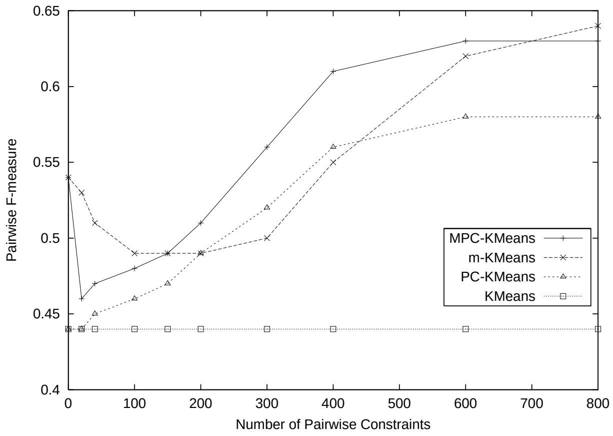

Figure 5. Results on the *Letter-IJL* dataset

- MPC-KMeans clustering, which involves both seeding and metric learning in the unified framework described in Section 2.4;
- m-KMeans, which is K-Means clustering with the metric learning component described in Section 3.3, without utilizing constraints for seeding;
- PC-KMeans clustering, which utilizes constraints for seeding the initial clusters and forces the cluster assignments to respect the constraints without doing any metric learning, as outlined in Section 2.2;
- Unsupervised K-Means clustering.

On the presented datasets, the unified approach (MPC-KMeans) outperforms individual seeding (PC-KMeans) and metric learning (m-KMeans) approaches. The learning curves illustrate that providing pairwise constraints is beneficial to clustering quality.

For the *Wine* and *Letter-IJL* datasets, the difference between methods that utilize metric learning (MPC-KMeans and m-KMeans) and those that do not (PC-KMeans and regular K-Means) with no pairwise constraints indicates that even in the absence of constraints, weighting features by their variance (essentially using Mahalanobis distance) improves clustering accuracy. For the *Wine* dataset, additional constraints provide an insubstantial improvement in cluster quality on this dataset, which shows that meaningful feature weights are obtained from scaling by variance using just the unlabeled data.

Some of the metric learning curves display a characteristic "dip", where clustering accuracy decreases when initial constraints are provided, but after a certain point starts to increase and eventually outperforms the initial point of the learning curve. We conjecture that this phenomenon is due to the fact that feature weights learned from few constraints are unreliable, while increasing the number of constraints provides the metric learning mechanism enough data to estimate good metric parameters.

On the other hand, seeding the clusters with a small number of pairwise constraints has an immediate positive effect on the final cluster quality, while providing more pairwise constraints has diminishing returns, i.e., PC-KMeans learning curves rise slowly. When both seeding and metric learning are utilized, the unified approach benefits from the individual strengths of the two methods, as can be seen from the MPC-KMeans results.

Overall, our results indicate that the unified approach to utilizing pairwise constraints in clustering outperforms using seeding and metric learning individually and leads to improvements in cluster quality.

5. Future Work

Extending our approach to high-dimensional datasets like text, where Euclidean distance performs poorly, is the primary avenue for future research. We are currently working on a formulation that utilizes an objective function similar to $\mathcal{J}_{\mathrm{mpckm}}$ for spherical K-Means (Dhillon & Modha, 2001). The weight matrix $\mathbf{A}$ is likely to be singular for high-dimensional data; such a scenario could be handled by regularization.

Comparing our unified approach to other search-based and similarity-based techniques, e.g., those of (Xing et al., 2003) and (Cohn et al., 2000), is another area for future work. We are also planning to incorporate active selection of pairwise constraints in a framework similar to (Basu et al., 2003) with our proposed approach.

In some situations, obtaining data labels directly instead of pairwise constraints may be possible. While it is possible to infer pairwise constraints in such scenarios from the labels, the number of pairwise constraints grows quadratically with the amount of labeled data, making training on the entire set of pairwise constraints intractable. In such scenarios, sampling mechanisms for selecting a small number of meaningful pairwise constraints are an interesting topic for future work.

6. Conclusions

This paper has presented a new approach to semi-supervised clustering that unifies the previous search-based and similarity-based methods. We have presented a general formalization of the semi-supervised learning problem that has allowed us to develop a variation of the standard K-Means clustering algorithm that uses supervised data both as seeds and constraints during search, as well as for adapting the underlying distance metric. By ablating the individual components of this unified approach, we have experimentally compared the individual approaches to each other and to their combination. When only small amounts of supervised data are used, the search-based approach produces more accurate clusters than the similarity-based approach. By combining the advantages of both techniques, the unified approach generally performs better than either of the approaches individually.

7. Acknowledgments

We would like to thank Arindam Banerjee for insightful comments. This research was supported in part by an IBM PhD Fellowship, a Faculty Fellowship from IBM Corporation, and by NSF grant IIS-0117308.

References

Basu, S., Banerjee, A., & Mooney, R. J. (2002). Semi-supervised clustering by seeding. *Proceedings of 19th International Conference on Machine Learning (ICML-2002)*.

Basu, S., Banerjee, A., & Mooney, R. J. (2003). Active semi-supervision for pairwise constrained clustering. Submitted for publication, available at `http://www.cs.utexas.edu/~sugato/`.

Bilenko, M., & Mooney, R. J. (2003). Adaptive duplicate detection using learnable string similarity measures. *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2003)*. Washington, DC.

Bilmes, J. (1997). *A gentle tutorial on the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models* (Technical Report ICSI-TR-97-021). ICSI.

Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. *Proceedings of the 11th Annual Conference on Computational Learning Theory*. Madison, WI.

Boykov, Y., Veksler, O., & Zabih, R. (1998). Markov random fields with efficient approximations. *IEEE Computer Vision and Pattern Recognition Conf.*

Cohn, D., Caruana, R., & McCallum, A. (2000). Semi-supervised clustering with user feedback. Unpublished manuscript. Available at `http://www-2.cs.cmu.edu/~mccallum/`.

Demiriz, A., Bennett, K. P., & Embrechts, M. J. (1999). Semi-supervised clustering using genetic algorithms. *ANNIE'99 (Artificial Neural Networks in Engineering)*.

Dhillon, I. S., & Modha, D. S. (2001). Concept decompositions for large sparse text data using clustering. *Machine Learning*, *42*, 143–175.

Joachims, T. (1999). Transductive inference for text classification using support vector machines. *Proceedings of the Sixteenth International Conference on Machine Learning (ICML-99)*. Bled, Slovenia.

Klein, D., Kamvar, S. D., & Manning, C. (2002). From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. *Proceedings of the The Nineteenth International Conference on Machine Learning (ICML-2002)*. Sydney, Australia.

Kleinberg, J., & Tardos, E. (1999). Approximation algorithms for classification problems with pairwise relationships: Metric labeling and Markov random fields. *IEEE Symp. on Foundations of Comp. Sci.*

Nigam, K., McCallum, A. K., Thrun, S., & Mitchell, T. (2000). Text classification from labeled and unlabeled documents using EM. *Machine Learning*, *39*, 103–134.

Wagstaff, K., Cardie, C., Rogers, S., & Schroedl, S. (2001). Constrained K-Means clustering with background knowledge. *Proceedings of 18th International Conference on Machine Learning (ICML-2001)*.

Xing, E. P., Ng, A. Y., Jordan, M. I., & Russell, S. (2003). Distance metric learning, with application to clustering with side-information. *Advances in Neural Information Processing Systems 15*. MIT Press.

A. Appendix

Given the objective function:

$$
\begin{aligned}
\mathcal{J}_{\text{mpckm}} = & \sum_{\mathbf{x}_i \in \mathcal{X}} \|\mathbf{x}_i - \boldsymbol{\mu}_{l_i}\|^2_{\mathbf{A}} - \log(\det(\mathbf{A})) \\
& + \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{M}} w_{ij} f_M(\mathbf{x}_i,\mathbf{x}_j) \mathbb{1}[l_i \neq l_j] \\
& + \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{C}} \overline{w}_{ij} f_C(\mathbf{x}_i,\mathbf{x}_j) \mathbb{1}[l_i = l_j]
\end{aligned}
$$

we obtain the M-step updates for cluster centroids $\{\boldsymbol{\mu}_h\}_{h=1}^K$ and metric parameterization matrix $\mathbf{A}$ by taking the partial derivatives of $\mathcal{J}_{\text{mpckm}}$ and setting them to zero. We use the following properties from linear algebra:

1. $\mathbf{x}^T\mathbf{A}\mathbf{x} = \text{tr}(\mathbf{A}\mathbf{x}\mathbf{x}^T)$
2. $\frac{\partial}{\partial \mathbf{A}}\text{tr}(\mathbf{A}\mathbf{B}) = \mathbf{B} + \mathbf{B}^T - \text{diag}(\mathbf{B})$
3. $\frac{\partial}{\partial \mathbf{A}}\log(\det(\mathbf{A})) = 2\mathbf{A}^{-1} - \text{diag}(\mathbf{A}^{-1})$
4. $2\mathbf{A} = \text{diag}(\mathbf{A}) \Rightarrow \mathbf{A} = \mathbf{0}$

Following is the derivation for estimating the cluster centroids $\boldsymbol{\mu}_h$ and the weight matrix $\mathbf{A}$:

$$
\begin{aligned}
& \frac{\partial \mathcal{J}_{\text{mpckm}}}{\partial \boldsymbol{\mu}_h} = 0 \Rightarrow \boldsymbol{\mu}_h = \frac{\sum_{\mathbf{x}_j \in \mathcal{X}_h} \mathbf{x}_j}{|\mathcal{X}_h|} \\
& \frac{\partial \mathcal{J}_{\text{mpckm}}}{\partial \mathbf{A}} = 0 \Rightarrow \\
\Rightarrow \quad & \frac{\partial}{\partial \mathbf{A}}\Bigg[\sum_{\mathbf{x}_i \in \mathcal{X}} \text{tr}(\mathbf{A}(\mathbf{x}_i - \boldsymbol{\mu}_{l_i})(\mathbf{x}_i - \boldsymbol{\mu}_{l_i})^T) \\
& + \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{M}} w_{ij} \max(\alpha_{\min}, \\
& \qquad \alpha_{\max} - \text{tr}(\mathbf{A}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T))\mathbb{1}[l_i \neq l_j] \\
& + \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{C}} \overline{w}_{ij} \min(\alpha_{max}, \\
& \qquad \alpha_{min} + \text{tr}(\mathbf{A}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T))\mathbb{1}[l_i = l_j] \\
& - \log(\det(\mathbf{A}))\Bigg] = 0 \qquad \text{(by Prop.1)}
\end{aligned}
$$

$$
\begin{aligned}
\Rightarrow \quad & 2\Bigg[\sum_{\mathbf{x}_i \in \mathcal{X}} (\mathbf{x}_i - \boldsymbol{\mu}_{l_i})(\mathbf{x}_i - \boldsymbol{\mu}_{l_i})^T \\
& - \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{M}^*} w_{ij}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T \mathbb{1}[l_i \neq l_j] \\
& + \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{C}^*} \overline{w}_{ij}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T \mathbb{1}[l_i = l_j] - \mathbf{A}^{-1}\Bigg] \\
& = \text{diag}\Bigg[\sum_{\mathbf{x}_i \in \mathcal{X}} (\mathbf{x}_i - \boldsymbol{\mu}_{l_i})(\mathbf{x}_i - \boldsymbol{\mu}_{l_i})^T \\
& - \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{M}^*} w_{ij}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T \mathbb{1}[l_i \neq l_j] \\
& + \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{C}^*} \overline{w}_{ij}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T \mathbb{1}[l_i = l_j] \\
& - \mathbf{A}^{-1}\Bigg] \qquad \text{(by Prop.2 and Prop.3)}
\end{aligned}
$$

$$
\begin{aligned}
\Rightarrow \quad \mathbf{A}^{-1} = & \sum_{\mathbf{x}_i \in \mathcal{X}} (\mathbf{x}_i - \boldsymbol{\mu}_{l_i})(\mathbf{x}_i - \boldsymbol{\mu}_{l_i})^T \\
& - \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{M}^*} w_{ij}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T \mathbb{1}[l_i \neq l_j] \\
& + \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{C}^*} \overline{w}_{ij}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T \mathbb{1}[l_i = l_j] \quad \text{(by Prop.4)}
\end{aligned}
$$