Proceedings of the SIAM International Conference on Data Mining,
(SDM-2004), pp. 333-344, Lake Buena Vista, FL, April, 2004

Active Semi-Supervision for Pairwise Constrained Clustering

Sugato Basu
Computer Sciences,
Univ. of Texas at Austin,
Austin, TX 78712
sugato@cs.utexas.edu

Arindam Banerjee
Electrical and Computer Eng.,
Univ. of Texas at Austin,
Austin, TX 78712
abanerje@ece.utexas.edu

Raymond J. Mooney
Computer Sciences,
Univ. of Texas at Austin,
Austin, TX 78712
mooney@cs.utexas.edu

Abstract

Semi-supervised clustering uses a small amount of supervised data to aid unsupervised learning. One typical approach specifies a limited number of *must-link* and *cannot-link* constraints between pairs of examples. This paper presents a pairwise constrained clustering framework and a new method for actively selecting informative pairwise constraints to get improved clustering performance. The clustering and active learning methods are both easily scalable to large datasets, and can handle very high dimensional data. Experimental and theoretical results confirm that this active querying of pairwise constraints significantly improves the accuracy of clustering when given a relatively small amount of supervision.

1 Introduction

In many data mining and machine learning tasks, there is a large supply of unlabeled data but limited labeled data, since labeled data 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 [6, 20, 30]. More specifically, *semi-supervised clustering*, the use of class labels or *pairwise constraints* on some examples to aid unsupervised clustering, has been the focus of several recent projects [4, 22, 33, 34].

In a *semi-supervised clustering* setting, the focus is on clustering large amounts of unlabeled data in the presence of a small amount of supervised data. In this setting, we consider a framework that has pairwise *must-link* and *cannot-link* constraints between points in a dataset (with an associated cost of violating each constraint), in addition to having distances between the points. These constraints specify that two examples must be in the same cluster (must-link) or different clusters (cannot-link) [33]. In real-world unsupervised learning tasks, e.g., clustering for speaker identification in a conversation [17], visual correspondence in multi-view image processing [7], clustering multi-spectral information from Mars images [32], etc., considering supervision in the form of constraints is generally more practical than providing class labels, since true labels may be unknown a priori, while it can be easier to specify whether pairs of points belong to the same cluster or different clusters.

We propose a cost function for *pairwise constrained clustering* (PCC) that can be shown to be the configuration energy of a Hidden Markov Random Field (HMRF) over the data with a well-defined potential function and noise model. Then, the pairwise-constrained clustering problem becomes equivalent to finding the HMRF configuration with the highest posterior probability, i.e., minimizing its energy. We present an algorithm for solving this problem.

Further, in order to maximize the utility of the limited supervised data available in a semi-supervised setting, supervised training examples should be *actively* selected as maximally informative ones rather than chosen at random, if possible [27]. In that case, fewer constraints will be required to significantly improve the clustering accuracy. To this end, we present a new method for actively selecting good pairwise constraints for semi-supervised clustering.

Both our active learning and pairwise constrained clustering algorithms are linear in the size of the data, and hence easily scalable to large datasets. Our formulation can also handle very high dimensional data, as our experiments on text datasets demonstrate.

Section 2 outlines the pairwise constrained clustering framework, and Section 3 presents a refinement of `KMeans` clustering [13, 25], called `PCKMeans`, that utilizes pairwise constraints. In Section 4, we present a method for actively picking good constraints by asking queries of the form "Are these two examples in same or different classes?". Experimental results on clustering high-dimensional text data and UCI data demonstrate that (1) `PCKMeans` clustering with constraints performs considerably better than unconstrained `KMeans` clustering, and (2) active `PCKMeans` achieves significantly steeper learning curves compared to `PCKMeans` with random pairwise queries.

2 Pairwise Constrained Clustering

Centroid-based partitional clustering algorithms (e.g., `KMeans`) find a disjoint k partitioning $\{\mathcal{X}_h\}_{h=1}^{k}$ (with each partition having a centroid $\boldsymbol{\mu}_h$) of a dataset $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$ such that the total distance between the points and the cluster centroids is (locally) minimized. We introduce a framework for *pairwise constrained clustering* (PCC) that has pairwise *must-link* and *cannot-link* constraints [33] between a subset of points in the dataset (with a cost of violating each constraint), in addition to distances between points. Since centroid-based clustering cannot handle pairwise constraints explicitly, we formulate the goal of clustering in the PCC framework as minimizing a combined objective function: the sum of the total distance between the points and their cluster centroids and the cost of violating the pairwise constraints.

For the PCC framework with both must-link and cannot-link constraints, let $\mathcal{M}$ be the set of 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 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. Note that we consider the tuples in $\mathcal{M}$ and $\mathcal{C}$ to be order-independent, i.e., $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{C} \Rightarrow (\mathbf{x}_j, \mathbf{x}_i) \in \mathcal{C}$, and so also for $\mathcal{M}$. 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 l_i be the cluster assignment of a point $\mathbf{x}_i$, where $l_i \in \{h\}_{h=1}^k$. The cost of violating must-link and cannot-link constraints is typically quantified by metrics [23]. We restrict our attention to the uniform metric (also known as the generalized Potts metric), for which the cost of violating a must-link $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{M}$ is given by $w_{ij}\mathbb{1}[l_i \neq l_j]$, i.e., if the must-linked points are assigned to two different clusters, the incurred cost is w_{ij}. Similarly, the cost of violating a cannot-link $(\mathbf{x}_i, x_j) \in \mathcal{C}$ is given by $\overline{w}_{ij}\mathbb{1}[l_i = l_j]$, i.e., if the cannot-linked points are assigned to the same cluster, the incurred cost is $\overline{w}_{ij}$. Note that here $\mathbb{1}$ is the indicator function, with $\mathbb{1}[true]$ = 1 and $\mathbb{1}[false]$ = 0. Using this model, the problem of PCC 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 $\boldsymbol{\mu}_{l_i}$:

$$\text{(2.1)} \qquad \mathcal{J}_{\text{pckm}} = \frac{1}{2}\sum_{\mathbf{x}_i \in \mathcal{X}} ||\mathbf{x}_i - \boldsymbol{\mu}_{l_i}||^2 + \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{M}} w_{ij}\mathbb{1}[l_i \neq l_j] + \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{C}} \overline{w}_{ij}\mathbb{1}[l_i = l_j]$$

Minimizing this objective function can be shown to be equivalent to maximizing the posterior configuration probability of a Hidden Markov Random Field, details of which are given in Appendix A.1. The mathematical formulation of this framework was motivated by the *metric labeling* problem and the *generalized Potts* model [7, 23], for which Kleinberg et al. [23] proposed an approximation algorithm. Their formulation only considers the set $\mathcal{M}$ of must-link constraints, which we extended to the PCC framework by adding the set $\mathcal{C}$ of cannot-link constraints. Our proposed pairwise constrained `KMeans` (`PCKMeans`) algorithm greedily optimizes $\mathcal{J}_{\text{pckm}}$ using a `KMeans`-type iteration with a modified cluster-assignment step. For experiments with text documents, we used a variant of `KMeans` called spherical `KMeans` (`SPKMeans`) [11] that has computational advantages for sparse high dimensional text data vectors. We will present our algorithm and its motivation based on `KMeans` in Section 3, but all of it can be easily extended for `SPKMeans`. In the domains that we will be considering, e.g., text clustering, different costs for different pairwise constraints are not available in general, so for simplicity we will be assuming all elements of W and $\overline{W}$ to have the same constant value w in (2.1). We will make a detailed study of the effect of the choice of w in Section 5.

Note that `KMeans` has a running time of $\mathcal{O}(ndk)$, where n is the number of data points, d is the number of dimensions and k is the number of clusters. `SPKMeans` has a running time of $\mathcal{O}(lk)$, where l is the number non-zero entries in the $n \times d$ input data matrix. So they are both linear in the size of the input, making our `PCKMeans` algorithm for the PCC framework quite efficient. `PCKMeans` can also handle sparse high-dimensional data (e.g. text, gene micro-array), since it has the computational advantage of `SPKMeans` in these domains.

3 Clustering Algorithm

Given a set of data points $\mathcal{X}$, a set of must-link constraints $\mathcal{M}$, a set of cannot-link constraints $\mathcal{C}$, the weight of the constraints w and the number of clusters to form k, we propose an algorithm `PCKMeans` that 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 $\mathcal{J}_{\text{pckm}}$ is (locally) minimized.

In our previous work [4], we had observed that proper initialization of centroid-based algorithms like KMeans using the provided semi-supervision in the form of labeled points greatly improves clustering performance. Here, instead of labeled points, we are given supervision in the form of constraints on pairs of points – in this case also, our goal in the initialization step will be to get good estimates of the cluster centroids from the pairwise constraints.

In the initialization step of `PCKMeans`, we take the transitive closure of the must-link constraints [33] and augment the set $\mathcal{M}$ by adding these entailed constraints.[1] Note that our current model considers consistent (non-noisy) constraints, but it can also be extended to handle inconsistent (noisy) constraints, as discussed in Section 7. Let the number of connected components in the augmented set $\mathcal{M}$ be λ, which are used to create λ neighborhood sets $\{N_p\}_{p=1}^{\lambda}$.

[1] A note on complexity: the transitive closure and constraint augmentation takes $\mathcal{O}(|\mathcal{M}| + |\mathcal{C}|)$ operations.

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, again assuming consistency of constraints. From now on, we will 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 $\{N_p\}_{p=1}^{\lambda}$, which are used to initialize the cluster centroids. 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 perturbations of the global centroid of $\mathcal{X}$ [14].

Algorithm: `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, weight of constraints w.
Output: Disjoint k partitioning $\{\mathcal{X}_h\}_{h=1}^{k}$ of $\mathcal{X}$ such that objective function $\mathcal{J}_{\text{pckm}}$ 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 $\mathbf{x}$ to the cluster h^* (i.e. set $\mathcal{X}_{h^*}^{(t+1)}$), for $h^* = \arg\min_h(\frac{1}{2}\|\mathbf{x} - \boldsymbol{\mu}_h^{(t)}\|^2 + w\sum_{(\mathbf{x},\mathbf{x}_j)\in\mathcal{M}} \mathbb{1}[h \neq l_j] + w\sum_{(\mathbf{x},\mathbf{x}_j)\in\mathcal{C}} \mathbb{1}[h = l_j])$
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. $t \leftarrow (t+1)$

Figure 1: `PCKMeans` algorithm

The algorithm `PCKMeans` alternates between the cluster assignment and centroid estimation steps (see Figure 1). In the cluster assignment step of `PCKMeans`, every point $\mathbf{x}$ is assigned to a cluster such that it minimizes the sum of the distance of $\mathbf{x}$ to the cluster centroid and the cost of constraint violations incurred by that cluster assignment (by equivalently satisfying as many must-links and cannot-links as possible by the assignment). Note that the cluster 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. For our experiments, we consider a random ordering of the points in this assignment step. The centroid re-estimation step is the same as `KMeans`, i.e., each cluster centroid is calculated by taking the mean of the points assigned to that cluster.

LEMMA 1. *The algorithm* `PCKMeans` *converges to a local minimum of* $\mathcal{J}_{\text{pckm}}$.

Proof. For analyzing the cluster assignment step, let us consider (2.1). Each point $\mathbf{x}$ moves to a new cluster h only if the component $(\frac{1}{2}\|\mathbf{x} - \boldsymbol{\mu}_h\|^2 + w\sum_{(\mathbf{x},\mathbf{x}_j)\in\mathcal{M}} \mathbb{1}[h \neq l_j] + w\sum_{(\mathbf{x},\mathbf{x}_j)\in\mathcal{C}} \mathbb{1}[h = l_j])$ of $\mathcal{J}_{\text{pckm}}$, contributed by the *point* $\mathbf{x}$, decreases. So when all points are given their new assignment, $\mathcal{J}_{\text{pckm}}$ will decrease or remain the same.

For analyzing the centroid re-estimation step, let us consider an equivalent form of (2.1):

$$\mathcal{J}_{\text{pckm}} = \frac{1}{2}\sum_{h=1}^{k}\sum_{\mathbf{x}_i\in\mathcal{X}_h}\|\mathbf{x}_i - \boldsymbol{\mu}_h\|^2 + \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{M}} w_{ij}\mathbb{1}[l_i \neq l_j] + \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{C}} \overline{w}_{ij}\mathbb{1}[l_i = l_j] \tag{3.2}$$

Each cluster centroid $\boldsymbol{\mu}_h$ is re-estimated by taking the mean of the points in the partition $\mathcal{X}_h$, which minimizes the component $(\frac{1}{2}\sum_{\mathbf{x}_i\in\mathcal{X}_h}\|\mathbf{x}_i - \boldsymbol{\mu}_h\|^2)$ of $\mathcal{J}_{\text{pckm}}$ in (3.2) contributed by the *partition* $\mathcal{X}_h$. The pairwise constraints do not take in part in this step because the constraints are not an explicit function of the centroid. As a result only the first term $(\frac{1}{2}\sum_{h=1}^{k}\sum_{\mathbf{x}_i\in\mathcal{X}_h}\|\mathbf{x}_i - \boldsymbol{\mu}_h\|^2)$ of $\mathcal{J}_{\text{pckm}}$ in (3.2) is minimized in this step.

Hence the objective function decreases after every cluster assignment and centroid re-estimation step till convergence, implying that the `PCKMeans` algorithm will converge to a local minimum of $\mathcal{J}_{\text{pckm}}$. ■

4 Active Learning Algorithm

In the semi-supervised setting, getting labels on data point pairs may be expensive. In this section, we discuss an active learning scheme in the PCC setting in order to improve clustering performance with as few queries as possible. Formally, the scheme has access to a noiseless oracle that can assign a must-link or cannot-link label on a given pair $(\mathbf{x}_i, \mathbf{x}_j)$, and it can pose a constant number of queries to the oracle.[2]

In order to get pairwise constraints that are more informative than random in the PCC model, we have developed an active learning scheme for selecting pairwise constraints using the *farthest-first* traversal scheme. The basic idea of farthest-first traversal of a set of points is to find k points such that they are far from each other. In farthest-first traversal, we first select a starting point at random, choose the next point to be farthest from it and add it to the traversed set, then pick the following point farthest from the traversed set (using the standard notion of distance from a set: $d(\mathbf{x}, S) = \min_{y \in S} d(\mathbf{x}, \mathbf{y})$), and so on. Farthest-first traversal gives an efficient approximation of the *k-center* problem [18], and has also been used to construct hierarchical clusterings with performance guarantees at each level of the hierarchy [10]. For our data model (see Appendix A.2), we prove another interesting property of farthest-first traversal (see Appendix A.4) that justifies its use for active learning.

In [4], it was observed that initializing `KMeans` with centroids estimated from a set of labeled examples for each cluster gives significant performance improvements. Under certain generative model-based assumptions, one can connect the mixture of Gaussians model to the KMeans model [21]. A direct calculation using Chernoff bounds shows that if a particular cluster (with an underlying Gaussian model) with true centroid $\boldsymbol{\mu}$ is seeded with m points (drawn independently at random from the corresponding Gaussian distribution) and the estimated centroid is $\hat{\boldsymbol{\mu}}$, then

$$\mathbf{Pr}(|\hat{\boldsymbol{\mu}} - \boldsymbol{\mu}| \geq \delta) \leq e^{-\delta^2 m/2} \tag{4.3}$$

where $\delta \in \mathbb{R}+$. Thus, the deviation of the centroid estimates falls exponentially with the number of seeds, and hence seeding results in good initial centroids. Since good initial centroids are very critical for the success of greedy algorithms such as `KMeans`, we follow the same principle for the pairwise case: we will try to get as many points (proportional to the actual cluster size) as possible per cluster by asking pairwise queries, so that `PCKMeans` is initialized from a very good set of centroids.

The proposed active learning scheme has two phases. The first phase explores the given data to get k pairwise disjoint non-null neighborhoods, each belonging to a different cluster in the underlying clustering of the data, as fast as possible. Note that even if there is only one point per neighborhood, this neighborhood structure defines a correct skeleton of the underlying cluster. For this phase, we propose an algorithm `Explore` that essentially uses the *farthest-first* scheme to form appropriate queries for getting the required pairwise disjoint neighborhoods.

[2] The oracle can also give a *don't-know* response to a query, in which case that response is ignored (pair not considered as a constraint) and that query is not posed again later.

At the end of `Explore`, at least one point has been obtained per cluster. The remaining queries are used to consolidate this structure. The cluster skeleton obtained from `Explore` is used to initialize k pairwise disjoint non-null neighborhoods $\{N_p\}_{p=1}^k$. Then, given any point $\mathbf{x}$ not in any of the existing neighborhoods, we will have to ask at most $(k-1)$ queries by pairing $\mathbf{x}$ up with a member from each of the disjoint neighborhoods N_p to find out the neighborhood to which $\mathbf{x}$ belongs. This principle forms the second phase of our active learning algorithm, and we call the algorithm for this phase `Consolidate`. In this phase, we are able to get the correct cluster label of $\mathbf{x}$ by asking at most $(k-1)$ queries. So, $(k-1)$ *pairwise labels are equivalent to a single pointwise label* in the worst case.

Now, we present the details of the algorithms for performing the exploration and the consolidation.

Algorithm: `Explore`
Input: Set of data points $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$, access to an oracle that answers pairwise queries, number of clusters k, total number of queries Q.
Output: $\lambda \leq k$ disjoint neighborhoods $\{N_p\}_{p=1}^\lambda$ corresponding to the true clustering of $\mathcal{X}$ with at least one point per neighborhood.
Method:
1. Initialize: set all neighborhoods $\{N_p\}_{p=1}^k$ to null
2. Pick the first point $\mathbf{x}$ at random, add to N_1, $\lambda \leftarrow 1$
3. While queries are allowed and $\lambda < k$
   $\mathbf{x} \leftarrow$point farthest from existing neighborhoods $\{N_p\}_{p=1}^\lambda$
   if, by querying, $\mathbf{x}$ is cannot-linked to all existing neighborhoods
     $\lambda \leftarrow \lambda + 1$, start a new neighborhood N_λ with $\mathbf{x}$
   else
     add $\mathbf{x}$ to the neighborhood with which it is must-linked

Figure 2: Algorithm `Explore`

4.1 Explore In the exploration phase, we use a very interesting property of the farthest-first traversal. Given a set of k disjoint balls of unequal size in a metric space, we show that the farthest-first scheme is sure to get one point from each of the k balls in a reasonably small number of attempts (see Appendix A.4). Hence, our algorithm `Explore` (see Figure 2) uses farthest-first traversal for getting a skeleton structure of the neighborhoods. In `Explore`, while queries are still allowed and k pairwise disjoint neighborhoods have not yet been found, the point $\mathbf{x}$ farthest from all the existing neighborhoods is chosen as a candidate for starting a new neighborhood. Queries are posed by pairing $\mathbf{x}$ with a random point from each of the existing neighborhoods. If $\mathbf{x}$ is cannot-linked to all the existing neighborhoods, a new neighborhood is started with $\mathbf{x}$. If a must-link is obtained for a particular neighborhood, $\mathbf{x}$ is added to that neighborhood. This continues till the algorithm runs out of queries or k pairwise disjoint neighborhoods have been found. In the latter case, the active learning scheme enters the consolidation phase.

Algorithm: Consolidate
Input: Set of data points $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$, access to an oracle that answers pairwise queries, number of clusters k, total number of queries Q, k disjoint neighborhoods corresponding to true clustering of $\mathcal{X}$ with at least one point per neighborhood.
Output: k disjoint neighborhoods corresponding to the true clustering of $\mathcal{X}$ with higher number of points per neighborhood.
Method:
1. Estimate centroids $\{\boldsymbol{\mu}_h\}_{h=1}^k$ of each of the neighborhoods
2. While queries are allowed
2a. randomly pick a point $\mathbf{x}$ not in the existing neighborhoods
2b. sort the indices h with increasing distances $\|\mathbf{x} - \boldsymbol{\mu}_h\|^2$
2c. for $h = 1$ to k
query $\mathbf{x}$ with each of the neighborhoods in sorted order till a must-link is obtained, add $\mathbf{x}$ to that neighborhood

Figure 3: Algorithm `Consolidate`

4.2 Consolidation The consolidation phase starts when at least one point has been obtained from each of the k clusters. The basic idea in the consolidation phase is that since we now have points from all the clusters, the proper neighborhood of any random point $\mathbf{x}$ can be determined within a maximum of $(k-1)$ queries. The queries will be formed by taking a point $\mathbf{y}$ from each of the neighborhoods in turn and asking for the label on the pair $(\mathbf{x}, \mathbf{y})$ till a must-link has been obtained. We will either get a must-link reply in $(k-1)$ queries, else if we get cannot-link replies for the $(k-1)$ queries to the $(k-1)$ neighborhoods, we can infer that the point is must-linked to the remaining neighborhood. Note that it is practical to sort the neighborhoods in increasing order of the distance of their centroids from $\mathbf{x}$ so that the correct must-link neighborhood for $\mathbf{x}$ is encountered sooner in the querying process. The outline of the algorithm `Consolidate` is given in Figure 3.

Both `Explore` and `Consolidate` add points to the clusters at a good rate. It can be shown using the result in Appendix A.4 that the `Explore` phase gets at least one point from each of the k underlying clusters in maximum $k\binom{k}{2}$ queries. When the active scheme has finished running `Explore` and is running `Consolidate`, it can also be shown using a generalization of the coupon collector's problem (Appendix A.4) that with high probability it will get one new point from each cluster in approximately $k^2 \log k$ queries. `Consolidate` therefore adds points to clusters at a faster rate than `Explore` by a factor of $\mathcal{O}(\frac{k}{\log k})$, which is validated by our experiments in Section 5. Note that this analysis is for balanced clusters, but a similar analysis with unbalanced clusters gives the same improvement factor.

Finally, we briefly address the case when the right number of clusters k is not known to the clustering algorithm, so that k is also unknown to the active learning scheme. In this case, only `Explore` is used while queries are allowed. `Explore` will keep discovering new clusters as fast as it can. When it has obtained all the clusters, it will not have any way of knowing this. However, from this point onwards, for every farthest-first $\mathbf{x}$ it draws from the dataset, it will always find a neighborhood that is must-linked to it. Hence, after discovering all the clusters, `Explore` will essentially consolidate the clusters too. However, when k is known, it makes sense to invoke `Consolidate` since (1) it adds points to clusters at a faster rate than `Explore`, and (2) it picks random samples following the underlying data distribution, which have certain nice properties in terms of estimating good centroids (e.g., Chernoff bounds on the centroid estimates, as shown in (4.3)), that the samples obtained using farthest-first traversal need not have.

5 Experiments

In this section, we outline the details of our experiments.

5.1 Datasets In our experiments with high-dimensional text documents, we used datasets created from the `20 Newsgroups` collection.[3] It has messages collected from 20 different Usenet newsgroups, 1000 messages from each newsgroup. From the original dataset, a reduced dataset `News-all20` was created by taking a random subsample of 100 documents from each of the 20 newsgroups – this subsample is a more difficult dataset to cluster than the original `20 Newsgroups`, since each cluster has fewer documents. `News-all20` has 2000 points in 16089 dimensions. By selecting 3 categories from the reduced dataset `News-all20`, two other datasets were created: `News-sim3` that consists of 3 newsgroups on similar topics (comp.graphics, comp.os.ms-windows, comp.windows.x) with significant cluster overlap, and `News-diff3` that consists of 3 newsgroups on different topics (alt.atheism, rec.sport.baseball, sci.space) with well-separated clusters. `News-sim3` has 300 points in 3225 dimensions, while `News-diff3` had 300 points in 3251 dimensions. Another dataset we used in our experiments is a subset of `Classic3` [11] containing 400 documents – 100 *Cranfield* documents from aeronautical system papers, 100 *Medline* documents from medical journals, and 200 *Cisi* documents from information retrieval papers. This `Classic3-subset` dataset was specifically designed to create clusters of unequal size, and has 400 points in 2897 dimensions. Similarities between data points in the text datasets were computed using cosine similarity, following `SPKMeans` [11]. `SPKMeans` maximizes the average cosine similarity between points and cluster centroids, so that the objective function monotonically increases with every iteration till convergence. All the text datasets were preprocessed following the methodology of Dhillon et al. [11].

For experiments on low-dimensional data, we selected the UCI dataset `Iris` [5], which has 150 points in 4 dimen-

[3] http://www.ai.mit.edu/people/jrennie/20Newsgroups

sions. The Euclidean metric was used for computing distances between points in this dataset, following KMeans. In this case, the objective function, which is the average squared Euclidean distance between points and cluster centroids, decreases at every iteration till convergence. The Iris dataset was not pre-processed in any way.

5.2 Clustering evaluation We have used three metrics for cluster evaluation: *normalized mutual information* (NMI), *pairwise F-measure*, and *objective function*.

Normalized mutual information (NMI) determines the amount of statistical information shared by the random variables representing the cluster assignments and the user-labeled class assignments of the data points. We computed NMI following the methodology of Strehl et al. [31]. NMI measures how closely the clustering algorithm could reconstruct the underlying label distribution in the data. If C is the random variable denoting the cluster assignments of the points, and K is the random variable denoting the underlying class labels on the points, then the NMI measure is defined as [2]:

$$NMI = \frac{I(C;K)}{(H(C)+H(K))/2}$$

where $I(X;Y) = H(X) - H(X|Y)$ is the mutual information between the random variables X and Y, $H(X)$ is the Shannon entropy of X, and $H(X|Y)$ is the conditional entropy of X given Y.

Pairwise F-measure is defined as the harmonic mean of pairwise precision and recall, the traditional information retrieval measures adapted for evaluating clustering by considering pairs of points – for every pair of points that do not have explicit constraints between them, the decision to cluster this pair into same or different clusters is considered to be correct if it matches with the underlying class labeling available for the points. Pairwise F-measure is defined as:

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

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

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

Measures like modified Rand Index [22, 33, 34] that are frequently used for evaluation of clustering in the PCC framework are very similar to pairwise F-measure. NMI has also become a popular clustering evaluation metric [2, 12, 15]. We present results using both these evaluation measures and observe from the results that they are strongly correlated. We also show results for the objective function $\mathcal{J}_{\text{pckm}}$.

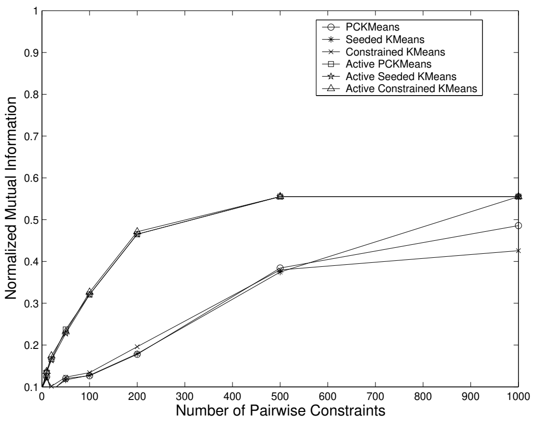

Figure 4: Comparison of NMI values on News-sim3

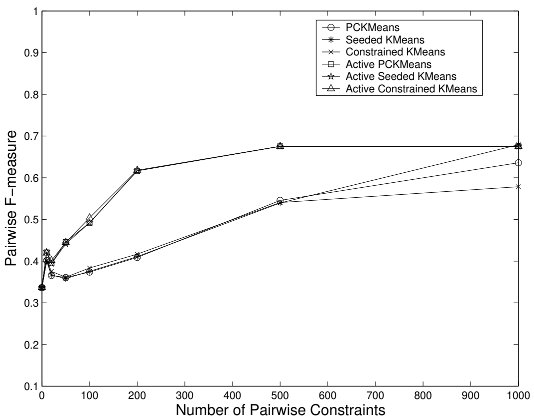

Figure 5: Comparison of pairwise F-measure values on News-sim3

5.3 Experimental Methodology For all the algorithms, on each dataset, we have generated learning curves with 10-fold cross-validation, where the x-axis represents the number of pairwise constraints given as input to the algorithms. For non-active PCKMeans the pairwise constraints are selected at random, while for active PCKMeans the pairwise constraints are selected using our active learning scheme. For studying the effect of pairwise constraints and active learning, 10% of the dataset is set aside as the test set at any particular fold. The training sets at different points of the learning curve are pairwise constraints obtained from the remaining 90% of the data, with increasing number of pairwise constraints being given as input to the clustering along the learning curve. The clustering algorithm is run on the whole dataset, and the corresponding objective function is reported. Following the methodology of Basu et al. [4], NMI and pairwise F-measure are calculated only on the test set, from which no constraints were supplied. The results at each point on the learning curve are obtained by averaging over 10 folds. We did not continue the learning curve beyond 1000 queries (5000 for `News-all20`), since the general nature of the results was evident in this range. Moreover, in practical active learning applications, it is unrealistic to expect the user to answer even 1000 queries.

5.4 Results The results of the experiments are shown in Figures 4-11. Since the standard deviations of NMI, pairwise F-measure and objective function values in the plots were small for all the datasets, they have not been shown in the plots to reduce clutter.

Choice of w: We experimented with different values of the constraint weight parameter w. If w is set to 0, the algorithm is initialized with neighborhoods derived from the given constraints and then normal `KMeans` iterations are run till convergence. This is similar to the `SeededKMeans` algorithm outlined in [4], where the labeled data (seeds) are used to only initialize the KMeans algorithm and are not used in the following steps of the algorithm.

If w is set to a very high value, the algorithm is initialized with neighborhoods derived from the given constraints and the constraints become hard constraints, since the constraint cost violation component of the $\mathcal{J}_{\text{pckm}}$ objective function far supersedes its distance component. This is similar to the `ConstrainedKMeans` algorithm outlined in [4]. In this algorithm, the seeds are also used to initialize the `KMeans` algorithm. However, in the subsequent steps, the cluster labels of the seed data are kept unchanged and only the labels of the non-seed data are re-estimated.

If w is set to an intermediate value, the algorithm gives a tradeoff between minimizing the total distance between points and cluster centroids and the cost of violating the constraints. In the result plots in Figures 4-12, `PCKMeans` refers to running the algorithm with the intermediate value of w. The parameter w can be chosen by the user according to the degree of confidence in the constraints, or chosen to be a constant of the same order as the average similarity (for Spherical `KMeans`) or distance (for Euclidean `KMeans`) between pairs of points in the dataset. We set w to be 0.001 for the text-datasets and 1 for `Iris` dataset.

Thus, the w parameter acts as a tuning knob, giving us the continuum between a `SeededKMeans`-like algorithm on one extreme, where there is no guarantee of the constraint satisfaction in the clustering, and a `ConstrainedKMeans`-like algorithm on the other extreme, where the clustering process is forced to respect all the given constraints. Note that we can selectively guarantee that any particular constraint is satisfied throughout the clustering iterations, by selecting a very high corresponding cost of constraint violation.

The comparative results of active and non-active algo-

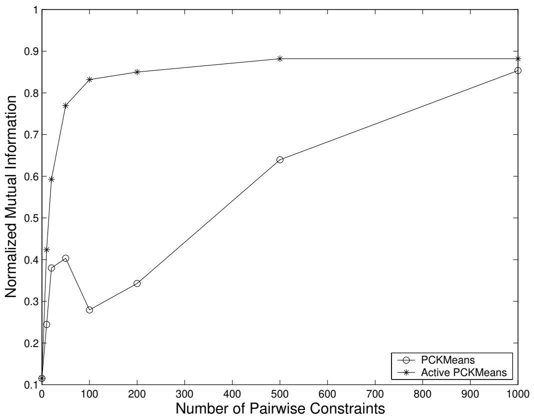

Figure 6: Comparison of NMI values on `News-diff3`

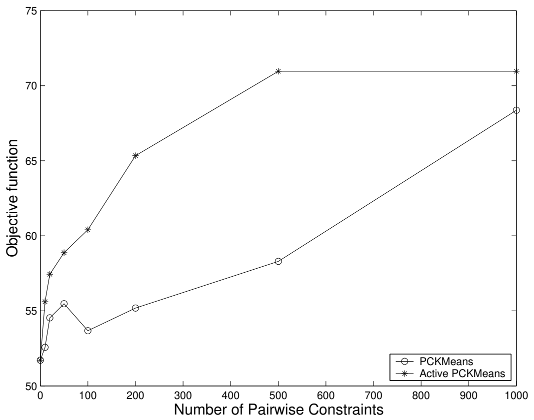

Figure 7: Comparison of objective function on `News-diff3`

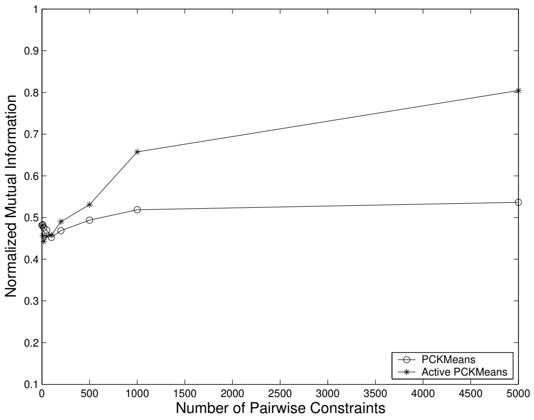

Figure 8: Comparison of NMI values on `News-all20`

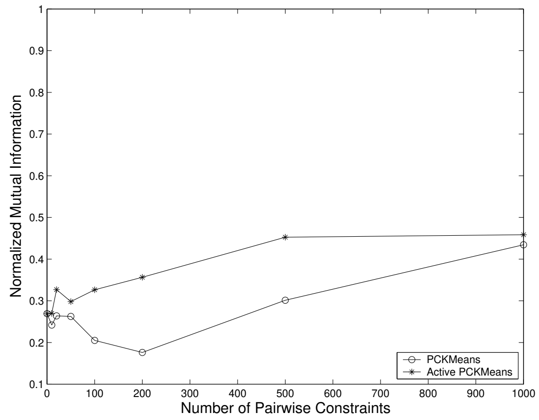

Figure 9: Comparison of NMI values on `Classic400`

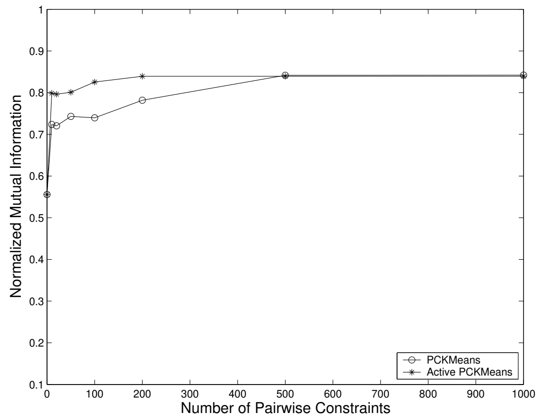

Figure 10: Comparison of NMI values on `Iris`

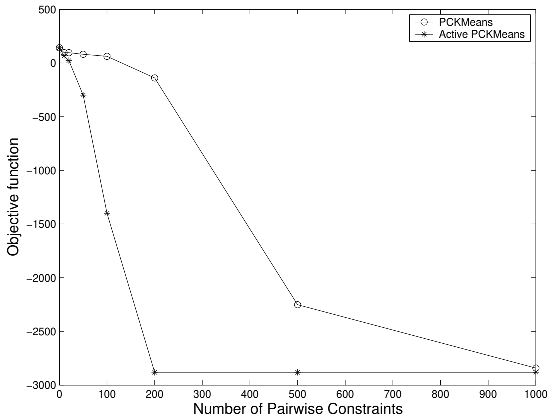

Figure 11: Comparison of objective function on `Iris`

rithms obtained for different values of w were similar for the datasets considered (see Figures 4 and 5). This leads us to conclude that proper initialization (by active learning) using the constraints gives much more benefit than satisfying the constraints during the algorithm, which validates the observation in [4] that proper initialization is crucial for good performance of centroid-based clustering algorithms. This point is explained in more detail in the discussion below. In Figures 6-11, we only present the results for the intermediate value of w for clarity of the plots. The curves for NMI and Pairwise F-measure were very similar for the datasets we considered (see Figures 4 and 5), so we only presenting the NMI results.

Objective function results: We show a representative objective function plot for a text dataset clustered using `SPKMeans` (Figure 7), for which the objective function increases along the learning curve. For Figure 11, the objective function is decreasing along the learning curve since simple `KMeans` with Euclidean distance was used for this dataset.

Note that for each objective function plot, the active and non-active schemes have the same number of constraints in the $\mathcal{M}$ and $\mathcal{C}$ sets at any point on the learning curve, but the actual constraints they have may be different. The active and the non-active schemes are allowed to both choose their own sets of constraints, and the objective function value after running `PCKMeans` clustering depends on this choice. For active `PCKMeans`, the constraints it chooses give it a better initialization (which is discussed in detail below), resulting in better value of the objective function after running the clustering algorithm.

Non-active schemes: As shown in Appendix A.3, if the number of random pairwise constraints is low, the probability that the k largest neighborhoods are in fact from k different clusters is very low. Until this point on the learning curve, some of the neighborhoods used to initialize `PCKMeans` can actually belong to the same cluster, so that we may not get representatives from all the clusters. This gives a poor initialization of `PCKMeans` that may cause the algorithm to converge to bad local minima. Consequently the clustering produced by `PCKMeans` can be unstable, resulting in varying pairwise F-measure and NMI values on the test set. This initial jitter can be observed in all the Figures 4-11. Beyond this point on the learning curve, non-active `PCKMeans` will most likely be initialized with points from each cluster. So after the initial jitter, the performance of non-active `PCKMeans` improves steadily along the learning curve with respect to objective function, NMI and pairwise F-measure, *showing the benefit of incorporating supervised data (constraints) in the clustering process*.

Active schemes: For the active algorithms, we consistently get *significant improvements over the non-active algorithms*, for all datasets we have considered. Firstly, we see the jitter only in the very early part of the learning curve. This is because the `Explore` phase creates only one neighborhood from each cluster and continues until k pairwise disjoint neighborhoods are found, creating all the neighborhoods within a small number of queries (see Appendix A.4). The jitter is so early in the learning curve that it cannot be even observed in the plots. In Figure 8, the jitter disappears after about the first 20 queries. The `Explore` phase of the active selection algorithm guarantees that the pairwise disjoint neighborhoods inferred from the constraints belong to different clusters in the actual underlying clustering, and so these neighborhoods would give us good initializations for the clustering algorithm. The `Consolidate` phase grows the k pairwise disjoint neighborhoods already created, so that when the active learning scheme runs out of queries, `PCKMeans` is initialized using centroids constructed from good neighborhoods. The improvement of the active scheme is more pronounced for the difficult high-dimensional text datasets, e.g., Figure 4-9.

From the above results, we conclude:

- Semi-supervised clustering with constraints performs considerably better than unsupervised clustering for the datasets we have considered (note that unsupervised clustering corresponds to semi-supervised clustering with 0 constraints). For both the active and non-active algorithms, the clustering evaluation measures (NMI and pairwise F-measure) and the objective function improve with increasing number of pairwise constraints provided along the learning curve.

- Active selection of pairwise constraints, using our two-phase active learning algorithm, significantly outperforms random selection of constraints.

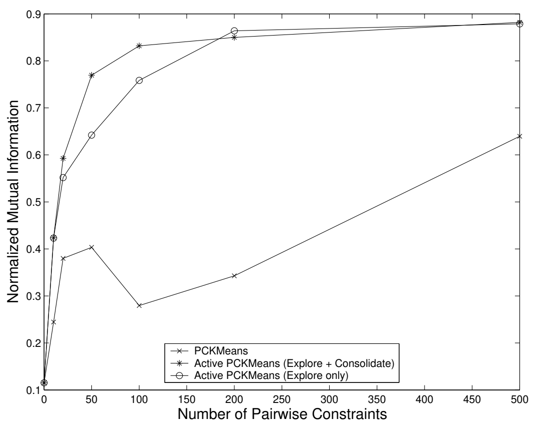

Figure 12: Comparison of Explore and Consolidate phases w.r.t. NMI on `News-diff3`

Explore Vs Consolidate: We also ran some ablation experiments, comparing the performance of the active `PCKMeans` scheme with both `Explore` and `Consolidate` with active `PCKMeans` with `Explore` only. We ran the ablation experiment on the `News-diff3` dataset. From the NMI result shown in Figure 12, we can see that running `Explore` only in the active learning phase gives improvement over random choice of constraints, but running both `Explore` and `Consolidate` gives even better results. So, *both* `Explore` *and* `Consolidate` *are useful phases of the active learning algorithm.*

6 Related Work

`COP-KMeans` is another algorithm in the pairwise constrained clustering model [33], but it does not handle soft-constraints, i.e., constraints that can be violated with an associated violation cost, which `PCKMeans` does. A soft-constrained algorithm `SCOP-KMeans` has been recently proposed [32], whose performance would be interesting to compare with `PCKMeans`. Bansal et al. [3] proposed a theoretical model where they performed clustering using only pairwise constraints, which is different from our model since we consider both constraints and an underlying metric between the points while clustering. Other work with the pairwise constrained clustering model includes learning distance metrics for clustering from pairwise constraints [17, 22, 34]. In this domain, Cohn. et al. [8] have proposed iterative user-feedback to acquire constraints, but it was not an active learning algorithm.

Active learning in the classification framework is a long-studied problem, where different principles of query selection have been studied, e.g., reduction of the version space size [16], reduction of uncertainty in predicted label [24], maximizing the margin on training data [1], finding high variance data points by density-weighted pool-based sampling [27], etc. However, active learning techniques in classification are not applicable in the clustering framework, since the basic underlying concept of reduction of classification error and variance over the distribution of examples [9] is not well-defined for clustering. In the unsupervised setting, Hofmann et al. [19] consider a model of active learning which is different from ours – they have incomplete pairwise similarities between points, and their active learning goal is to select new data, using expected value of information estimated from the existing data, such that the risk of making wrong estimates about the true underlying clustering from the existing incomplete data is minimized. In contrast, our model assumes that we have complete similarity information between all pairs of points, along with pairwise constraints whose violation cost is a component of the objective function (2.1), and the active learning goal is to select pairwise constraints which are most informative about the underlying clustering. Klein et al. [22] also consider active learning in semi-supervised clustering, but instead of making example-level queries they make cluster level queries, i.e., they ask the user whether or not two whole clusters should be merged. Answering example-level queries rather than cluster-level queries is a much easier task for a user, making our model more practical in a real-world active learning setting.

7 Conclusions and Future Work

In this paper, we have presented a pairwise constrained clustering framework and a new theoretically well-motivated method for actively selecting good pairwise constraints for semi-supervised clustering. Experiments on text and UCI data show that (1) `PCKMeans` with constraints performs considerably better than unconstrained `KMeans`, and (2) our active learning scheme performs quite well, giving significantly steeper learning curves compared to random pairwise queries. The active learning and pairwise constrained clustering algorithms are both linear and hence suitable for real-world clustering applications, as they can be easily scaled to large datasets and can handle very high-dimensional data.

The `Explore` stage of the active learning scheme is currently sensitive to outliers in the data. Note however that it is as sensitive to outliers as the baseline algorithm `KMeans`. Outlier sensitivity can be handled by density-weighted point selection in `Explore`, where we could have a modified farthest-first traversal that selects distant points from dense regions of the data space [27]. Such a formulation of active learning would be more robust to outliers, and can be used with more outlier-robust clustering algorithms, e.g., KMedian [28].

Our current clustering model assumes that the constraints are consistent, i.e., there is no noise in the constraints. We are working on incorporating a noise model into our PCC framework, so that we will be able to handle noisy constraints. This would involve some changes to the algorithms, e.g., not adding the inferred constraints between neighborhoods in the initialization step of `PCKMeans`, selectively rejecting points using a noise model in the `Explore` stage of the active learning algorithm, etc. The clustering model also assumes that the right number of clusters is given as an input – in the future, we want to select the number of clusters automatically, by incorporating a model selection criterion into the PCC objective function.

The cluster assignment step in the `PCKMeans` algorithm is an incremental step, dependent on the order in which the points are assigned to the cluster. Currently, we used a random ordering for cluster assignment for our experiments. We plan to investigate other cluster assignment schemes in future, and look into an online version of `PCKMeans`.

On the theoretical side, we want to explore the correlation between NMI and F-measure that we have empirically observed in our experiments, and come up with better guarantees for improvement of PCC with actively selected constraints over PCC with randomly selected constraints.

Finally, we also want to apply our active learning algorithm in the PCC framework to other domains, especially gene micro-array data analysis in bioinformatics.

8 Acknowledgments

This research was supported in part by IBM PhD Fellowship, by Intel and by NSF grant IIS-0117308.

A Appendix

A.1 Hidden Markov Random Field The PCKMeans objective function in (2.1) tries to minimize the total distance between points and their cluster centroids such that the minimum number of specified constraints between the points are violated. This mathematical formulation can be motivated by considering a Markov Random Field (MRF) [7] defined over $\mathcal{X}$ such that the field (or set) $F = \{F_i\}_{i=1}^n$ of random variables over $\mathcal{X}$ can take values $\{l_i\}_{i=1}^n$ with $l_i \in \{h\}_{h=1}^k, \forall i$. Let a configuration l denote the joint event $\{F = l\} = \{F_i = l_i\}_{i=1}^n$. Restricting the model to MRFs whose clique potentials involve pairwise points, the prior probability of a configuration l is $P(l) \propto \exp(-\sum_i \sum_j V_{(i,j)}(l_i, l_j))$, where

$$V_{(i,j)}(l_i, l_j) = \begin{cases} w_{ij}\mathbb{1}[l_i \neq l_j] & \text{if } (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{M} \\ \overline{w}_{ij}\mathbb{1}[l_i = l_j] & \text{if } (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{C} \\ 0 & \text{otherwise} \end{cases}$$

We assume an identity covariance Gaussian noise model for the observed data (von-Mises Fisher distribution [26] was considered as the noise model for high-dimensional text data)[4], and also assume that the observed (noisy) data points have been drawn independently of each other following this model. If $\{\boldsymbol{\mu}_h\}_{h=1}^k$ denote the true representatives corresponding to the labels $\{h\}_{h=1}^k$, the conditional probability of the observation $\mathcal{X}$ for a given configuration l is $P(\mathcal{X}|l) \propto \exp(-\frac{1}{2}\sum_{\mathbf{x}_i} \|\mathbf{x}_i - \boldsymbol{\mu}_{l_i}\|^2)$. Since the MRF is defined over the *hidden* true labels $\{l_i\}_{i=1}^n$ of the observed points $\{\mathbf{x}_i\}_{i=1}^n$, this model is called a Hidden Markov Random Field (HMRF) [35], which is a direct generalization of a Hidden Markov Model.

Since the posterior probability of a configuration l is $P(l|\mathcal{X}) \propto P(l)P(\mathcal{X}|l)$, the PCC objective function is proportional to the negative logarithm of the posterior probability of the specified HMRF. Hence, finding the MAP configuration of the HMRF becomes equivalent to minimizing the objective function in (2.1).

A.2 Model Assumptions First of all, we present the formal model of the dataset based on which all analysis will be done. The data is assumed to be coming from k disjoint uniform density balls of unequal size in a metric space.

[4]The framework can be shown to hold for arbitrary exponential noise models.

The balls are defined in terms of the metric. All data points inside any particular ball are assumed to be in the same cluster, and points from different balls are assumed to be from different clusters. The oracle is assumed to know this model. Let n be the total number of points under consideration. Let $\{\pi_h\}_{h=1}^k$ be the probabilities of drawing a point randomly from the h-th ball B_h. Without loss of generality, we assume $\pi_1 \leq \pi_2 \leq \cdots \leq \pi_k$. Further, let $1/l \leq \pi_1$. Let m_h be the number of points in the dataset from B_h. Then, $\pi_h = m_h/n$ and $\pi_h \propto V_h$, the volume of B_h, $\forall h$. Now, the number of possible *cannot-links* is $\sum_{\{h,l,h<l\}} m_h m_l$ and the number of *must-links* is $\sum_h \binom{m_h}{2}$. Let $\alpha = \sum_{\{h,l,h<l\}} m_h m_l / \sum_h \binom{m_h}{2}$.

A.3 Analysis of random initialization In PCKMeans, initialization is done using the k largest sized neighborhoods. We argue that within a small number of queries, the probability of getting even a 3-point neighborhood from any cluster is very low. Given Q pairs at random, on average there will be one *must-link* in every $(1+\alpha)$ pairs. Hence, there will be a total of $Q/(1+\alpha)$ *must-link* pairs in the expected behavior. Then, for the h-th cluster, there will be $r_h = \pi_h Q/(1+\alpha) \ll m_h$ *must-link* pairs on average. We focus on a particular cluster B_h on which r_h pairs have been selected at random. The size of the cluster is $m_h = n/k$. We will not get a 3-point neighborhood from B_h if none of the points $\mathbf{x} \in B_h$ gets drawn more than once in the random pair sampling. If the sampling of r_h pairs is replaced by the sampling of $2r_h$ vertices, the probability of getting a vertex twice is increased. Hence, the probability p_h of *not* getting a 3-point neighborhood is lower bounded by the probability of not getting a vertex twice in the vertex sampling setting. So,

$$\begin{aligned} p_h &\geq \sum_{\substack{\sum_l \beta_l = 2r_h \\ \beta_l < 2, \forall l}} \binom{2r_h}{\beta_1 \cdots \beta_{m_h}} \left(\frac{1}{m_h}\right)^{2r_h} \\ &= 1 \cdot \left(1 - \frac{1}{m_h}\right) \cdot \left(1 - \frac{2}{m_h}\right) \cdots \left(1 - \frac{2r_h - 1}{m_h}\right) \\ &\geq \left(1 - \frac{2r_h}{m_h}\right)^{2r_h} \approx 1 - \frac{4r_h^2}{m_h} = 1 - \frac{4m_h Q^2}{n^2(1+\alpha)^2} \end{aligned}$$

which is close to 1 for small values of Q. Hence, the probability of getting 3-point neighborhoods is very low. Therefore, the initialization is essentially done by k random draws from a set of approximately $Q/(1+\alpha)$ 2-point neighborhoods. In this setting, the probability of getting exactly one neighborhood from each cluster is

$$k! \prod_{h=1}^k \pi_h \leq \frac{k!}{k^k} = \frac{\sqrt{2\pi k}}{e^k}(1 + \frac{1}{12k} + O(\frac{1}{k^2}))$$

using the AM-GM inequality and the Stirling's formula. Clearly, the probability is quite low. This results in significant variance in the initializing neighborhoods and explains the initial jitter for the non-active algorithms for low values of Q.

A.4 Analysis of Explore We shall refer to points from the same cluster as having the same color. If the probability of drawing points of different colors is given by $1/l \leq \pi_1 \leq \pi_2 \leq \cdots \leq \pi_k$, then, by an extension of the coupon collector's problem [29], one can show that points of all colors will be drawn with high probability within $l \ln k + O(l)$ draws. We claim that the farthest first scheme gets points of all colors within l attempts with probability 1.

In the worst case, if the disjoint balls are placed by an adversary, the adversary will try to place the balls such that getting a point from at least one ball is very difficult. One can show that the optimum strategy for the adversary will be to make the smallest ball the most difficult to reach. Using a packing argument, we show that irrespective of the placement of the balls, the farthest first traversal cannot avoid any particular ball for long. Consider two balls b, B with probabilities π_b, π_B. Let r_b, r_B be the radii of the two balls, and V_b, V_B be the volumes of the two balls. Further, let $\sigma_b(B)$ denote the packing number of B with b balls — the maximum number of disjoint b balls that can be packed inside the ball B. Now, if there are just these two balls in the universe and if farthest-first traversal starts in B, the points obtained from B before entering b must have pairwise distances (between their centers) of at least $2r_b$, because otherwise the traversal would have picked the farthest point from b and got a distance of at least $2r_b$. Hence, the traversal cannot stay in B for more that $\sigma_b(B)$ farthest-first jumps because there are exactly these many points inside B that can be at a distance of at least $2r_b$ from each other. Now, the packing number $\sigma_b(B) \leq V_B/V_b = \pi_B/\pi_b$, the ratio of their probabilities. This argument can be extended to the general case of k balls. In the general case, the number of times the farthest first traversal can continue without entering the ball B_i is

$$n_i \leq \sum_{\substack{h=1 \\ h \neq i}}^k \sigma_{B_i}(B_h) \leq \sum_{\substack{h=1 \\ h \neq i}}^k \pi_h/\pi_i$$

Clearly, this number is largest for the smallest ball B_1. So, the maximum number of farthest-first jumps before reaching B_1 is given by

$$n_1 \leq \sum_{h=2}^k \pi_h/\pi_1 \;=\; (1-\pi_1)/\pi_1 = 1/\pi_1 - 1 \leq l - 1$$

In the next jump, farthest-first gets a point from B_1. Hence, the farthest first traversal will find points of all the colors in $(l-1)+1 = l$ attempts. Note that this is a significant $\ln k$ factor improvement over the random scheme.

References

[1] N. Abe and H. Mamitsuka. Query learning strategies using boosting and bagging. In *Proc. of 15th Intl. Conf. on Machine Learning (ICML-98)*, pages 1–10, 1998.

[2] A. Banerjee, I. Dhillon, J. Ghosh, and S. Sra. Generative model-based clustering of directional data. In *Proc. of 9th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining (KDD-2003)*, 2003.

[3] Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. In *IEEE Symp. on Foundations of Comp. Sci*, 2002.

[4] S. Basu, A. Banerjee, and R. J. Mooney. Semi-supervised clustering by seeding. In *Proc. of 19th Intl. Conf. on Machine Learning (ICML-2002)*, 2002.

[5] C. L. Blake and C. J. Merz. UCI repository of machine learning databases. http://www.ics.uci.edu/˜mlearn/MLRepository.html, 1998.

[6] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proc. of the 11th Annual Conf. on Computational Learning Theory*, 1998.

[7] Yuri Boykov, Olga Veksler, and Ramin Zabih. Markov random fields with efficient approximations. In *IEEE Computer Vision and Pattern Recognition Conf.*, 1998.

[8] David Cohn, Rich Caruana, and Andrew McCallum. Semi-supervised clustering with user feedback. Technical Report TR2003-1892, Cornell University, 2003.

[9] David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

[10] Sanjoy Dasgupta. Performance guarantees for hierarchical clustering. In *Conf. on Computational Learning Theory*, pages 351–363, 2002.

[11] I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42:143–175, 2001.

[12] Byron E. Dom. An information-theoretic external cluster-validity measure. Research Report RJ 10219, IBM, 2001.

[13] Richard O. Duda, David G. Stork, and Peter E. Hart. *Pattern Classification, 2nd ed.* Wiley, New York, NY, 1999.

[14] Usama M. Fayyad, Cory Reina, and P. S. Bradley. Initialization of iterative refinement clustering algorithms. In *Proc. of 4th Intl. Conf. on Knowledge Discovery and Data Mining (KDD-98)*, pages 194–198, 1998.

[15] Xiaoli Fern and Carla Brodley. Random projection for high dimensional data clustering: A cluster ensemble approach. In *Proc. of 20th Intl. Conf. on Machine Learning (ICML-2003)*, 2003.

[16] Yoav Freund, H. Sebastian Seung, Eli Shamir, and Naftali Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28:133–168, 1997.

[17] Aharon Bar Hillel, Tomer Hertz, Noam Shental, and Daphna Weinshall. Learning distance functions using equivalence relations. In *Proc. of 20th Intl. Conf. on Machine Learning (ICML-2003)*, 2003.

[18] D. Hochbaum and D. Shmoys. A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, 10(2):180–184, 1985.

[19] Thomas Hofmann and Joachim M. Buhmann. Active data clustering. In *Advances in Neural Information Processing Systems 10*, 1998.

[20] Thorsten Joachims. Transductive inference for text classification using support vector machines. In *Proc. of 16th Intl. Conf. on Machine Learning (ICML-99)*, 1999.

[21] Michael Kearns, Yishay Mansour, and Andrew Y. Ng. An information-theoretic analysis of hard and soft assignment methods for clustering. In *Proc. of 13th Conf. on Uncertainty in Artificial Intelligence (UAI-97)*, pages 282–293, 1997.

[22] Dan Klein, Sepandar D. Kamvar, and Christopher Manning. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In *Proc. of 19th Intl. Conf. on Machine Learning (ICML-2002)*, 2002.

[23] Jon Kleinberg and Eva Tardos. Approximation algorithms for classification problems with pairwise relationships: Metric labeling and Markov random fields. In *IEEE Symp. on Foundations of Comp. Sci*, 1999.

[24] David Lewis and William Gale. A sequential algorithm for training text classifiers. In *Proc. of 17th Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, 1994.

[25] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. of 5th Berkeley Symp. on Mathematical Statistics and Probability*, pages 281–297, 1967.

[26] K. V. Mardia and P. Jupp. *Directional Statistics*. John Wiley and Sons Ltd., 2nd edition, 2000.

[27] Andrew K. McCallum and Kamal Nigam. Employing EM and pool-based active learning for text classification. In *Proc. of 15th Intl. Conf. on Machine Learning (ICML-98)*, Madison, WI, 1998. Morgan Kaufmann.

[28] P. Mirchandani and R. Francis, editors. *Discrete Location Theory*. Wiley, New York, NY, 1990.

[29] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[30] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39:103–134, 2000.

[31] Alexander Strehl, Joydeep Ghosh, and Raymond Mooney. Impact of similarity measures on web-page clustering. In *Workshop on Artificial Intelligence for Web Search (AAAI 2000)*, pages 58–64, July 2000.

[32] Kiri Wagstaff. *Intelligent Clustering with Instance-Level Constraints*. PhD thesis, Cornell University, 2002.

[33] Kiri Wagstaff, Claire Cardie, Seth Rogers, and Stefan Schroedl. Constrained K-Means clustering with background knowledge. In *Proc. of 18th Intl. Conf. on Machine Learning (ICML-2001)*, 2001.

[34] Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart Russell. Distance metric learning, with application to clustering with side-information. In *Advances in Neural Information Processing Systems 15*. MIT Press, 2003.

[35] Y. Zhang, S. M. Smith, and M. Brady. Hidden Markov random field model and segmentation of brain MR images. Technical Report TR00YZ1, Oxford Center for FMRIB, John Radcliffe Hospital, 2000.