A Cluster-Level Semi-supervision Model for Interactive Clustering

Avinava Dubey[1], Indrajit Bhattacharya[2,⋆], and Shantanu Godbole[1]

[1] IBM Research - India
{kumdubey,shantanugodbole}@in.ibm.com
[2] Indian Institute of Science
indrajit@csa.iisc.ernet.in

Abstract. Semi-supervised clustering models, that incorporate user provided constraints to yield meaningful clusters, have recently become a popular area of research. In this paper, we propose a cluster-level semi-supervision model for inter-active clustering. Prototype based clustering algorithms typically alternate between updating cluster descriptions and assignment of data items to clusters. In our model, the user provides semi-supervision directly for these two steps. Assignment feedback re-assigns data items among existing clusters, while cluster description feedback helps to position existing cluster centers more meaningfully. We argue that providing such supervision is more natural for exploratory data mining, where the user discovers and interprets clusters as the algorithm progresses, in comparison to the pair-wise instance level supervision model, particularly for high dimensional data such as document collection. We show how such feedback can be interpreted as constraints and incorporated within the k-means clustering framework. Using experimental results on multiple real-world datasets, we show that this framework improves clustering performance significantly beyond traditional k-means. Interestingly, when given the same number of feedbacks from the user, the proposed framework significantly outperforms the pair-wise supervision model.

1 Introduction

While clustering has been one of the most effective tools for exploratory data mining for decades, it is widely accepted that the clusters generated without any supervision often do not lead to meaningful insights for the user. Accordingly, there has been a lot of interest in recent years in developing semi-supervised clustering models that can accommodate supervision from the user to guide the clustering process[4]. In the most popular model for semi-supervised clustering, the user provides must-link and cannot-link constraints over pairs of data instances [17]. It has been shown that such constraints can significantly improve clustering performance beyond that of unsupervised models.

An interesting feature of this instance-level model is that the constraints are independent of each other, and also of all other instances in the dataset. However, it is straight forward to imagine example datasets, as in Figure 1, where the same pair of instances may be 'must-linked' or 'cannot-linked' depending on the other instances present in the

⋆ Work done while at IBM Research - India.

J.L. Balcázar et al. (Eds.): ECML PKDD 2010, Part I, LNAI 6321, pp. 409–424, 2010.

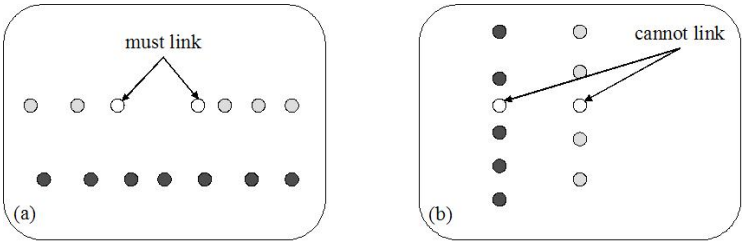


Fig. 1. Cluster examples involving the same pair of instances, shown in white. Based on the other instances in the dataset, in (a) they are must-linked to belong to the same cluster, while in (b) they are cannot-linked.

dataset. Therefore, such independent constraints required by the instance-level model can only be provided when the human supervisor can visualize the space of all the data instances and then decide on the desired shape of the clusters, as in most illustrative examples for instance-level supervision [4]. Such visualization of data items is difficult, if not impossible, in high dimensions, for example when clustering a collection of text documents. A pair of documents, one on the English soccer league and the other on the Spanish soccer league, would belong to the same cluster if the document collection is on different types of sports, but not in a different collection that discusses different European soccer leagues. Thus providing constraints on this pair of documents is not possible using independent pair-wise constraints without visualizing or understanding the document collection in its entirety.

In this paper, as an alternative we propose an interactive cluster-level semi-supervision framework for clustering, where such conditional constraints can be provided by the human supervisor. Prototype or model based clustering algorithms typically iterate over two steps — assignment of data points to clusters, and adjustment of clusters to minimize distortion. In our model, the user provides two different types of feedback, aimed directly at supervising these two different steps, while the algorithm executes. Using *assignment feedback*, the user moves a data point from one of the current clusters to another. Using *cluster description feedback*, the user modifies the feature vector of any current cluster to make it more meaningful. Such an interactive framework is particularly useful for exploring large high-dimensional data sets when the clusters are not known in advance. The current set of clusters provides the user with a summarized, aggregated view of the entire dataset. Conditioned on this current set of clusters, and also enabled by the summary that it provides, he then re-assigns and re-adjusts this clustering as he thinks appropriate. The algorithm learns from this feedback, and from other feedbacks provided in earlier stages, to re-cluster the dataset, which the user can again inspect and critique. The iterative process continues until the user is satisfied with the clustering. The basic idea of interaction for clustering [6,12], interpreted as pair-wise constraints, was proposed as early as 1999, but to the best of our knowledge, there has not been any follow-up work around cluster-level supervision to address large dimensionality of the data.

We show how both these types of feedback can be interpreted as constraints and incorporated within the k-means formulation, and the assignment and update steps can be modified to minimize constraint violation while maintaining low distortion error. Though we focus on the k-means objective function for this paper, we believe that a similar semi-supervision framework can be built around other model-based clustering objective functions as well.

The rest of the paper is organized as follows. In Section 2, we illustrate and motivate the two types of feedback using examples, followed by a review of related work in Section 3. We formalize the problem in Section 4, followed by the inter-active clustering algorithm in Section 5. Possible models for the supervisor and the issue of convergence are discussed in Section 6, and then our experimental evaluation is described in Section 7. We end with concluding remarks and possible future directions in Section 8.

2 Cluster-Level Supervision: An Example

In this section, we present an illustrative example of cluster-level supervision. Though the example is for document collections, we believe that the framework also offers similar advantages for other high dimensional domains.

Consider a very large collection of postings on a vehicle related mailing list. An analyst wishes to understand the issues being discussed, and, in order to do so, decides to partition the posts into k clusters, using the first stage of the algorithm, which is completely unsupervised. Note that he does not have any idea of possible issues ahead of time, apart from that they all relate to vehicles. On inspecting the cluster descriptions — the top words in decreasing order of weight (or importance) for the cluster — he finds that one cluster (c_1) is about $\{$Yamaha, Honda, car, bike, GM$\}$, while another (c_2) is about $\{$parts, power-steering, door, power-windows$\}$. Using his domain knowledge, he understands that the c_1 is about *Bikes & Cars*, while c_2 is about *Car Parts*.

In the first scenario, he likes these cluster descriptions, and goes on to inspect some of the individual posts contained in them. He notices that some posts 'truly' relating to c_1 (*Bikes & Cars*) — for example a few mentioning 'the best part about Honda' — have been assigned incorrectly by the algorithm to the *Car Parts* cluster c_2. He corrects this by appropriately re-assigning these posts to c_1. We call this an *assignment feedback* from the user, where he moves a data instance from one existing cluster to another. The clustering algorithm learns from this feedback — to add 'part' to the description of the first cluster c_1 as well, possibly with a small weight — so that other similar posts get correctly assigned.

In the second scenario, the user decides that he would rather prefer cluster c_1 to be about *Bikes* and cluster c_2 to be about *Cars & Car Parts*. To achieve this, he adjusts the description of the clusters, by changing c_1's description to $\{$Yamaha, Honda, bike$\}$ and c_2's description to $\{$car, GM, power-steering, door, power-windows, Honda$\}$. We call this a *cluster description feedback*, where the user directly modifies the features in the cluster description according to his preference. Observe that it would not have been possible for the user to create these new cluster descriptions, without knowing the summaries provided by the existing cluster descriptions. Again, the algorithm learns from this feedback to correctly reassign the posts to appropriate clusters. We will assume in

our formulation that the user provides a new weight vector for the cluster description, but in practice the user need not specify weights explicitly. For example, in a working system, he may simply rank order the top few features, or click and drag a weight curve over them. However, the details of the user interface is outside the scope of this paper, and we intend to investigate this in future work.

In general, the user is expected to provide both these types of feedback to the algorithm inter-actively, as new clusters emerge and documents get assigned to them. At any stage, the algorithm considers all feedback provided so far, even those at earlier stages, to recluster the documents.

We note that it is possible to provide cluster description feedback indirectly through assignment feedback, and vice versa. Re-assignment of points to centroids will automatically result from cluster description changes and similarly, many assignment constraints will lead to movement of the centroid in the next iteration. However we will see how using them directly to achieve the intended effect saves the user considerable time and effort.

3 Related Work

There has been a lot of research over the last decade on clustering with constraints [4]. The most popular approach provides pair-wise instance-level supervision, which is either used to learn distance metrics [19,16,2] or to provide additional constraints for the clustering algorithm [17,18,5,9,10,11]. Other work on cluster-level constraints looks to control size and balancing of clusters [9,1], or to find alternative clusters [14]. In contrast, we look to provide on-line supervision on descriptions and data assignments for existing clusters.

The idea of assignment feedback is closely related to active learning [7,8]. However, active learning assumes the classes in the data to be known apriori, while in our framework the user aims to simultaneously discover the clusters and the assignments to them in the spirit of exploratory data mining.

The concept of active supervision for clustering has been explored [15,6,3,12] but mostly for pair-wise constraints. Cohn et. al.[6] proposed the idea of inter-active clustering as early as 1999 in an unpublished manuscript. Though they suggest the possible use of cluster-item assignment feedback, the proposed model only incorporates pair-wise constraints. Similarly, desJardins et. al.[12] explore how user interaction with clusters, visualized in a two-dimensional space, can be interpreted as pair-wise constraints for clustering. To the best of our knowledge, the idea of using assignment feedback in conjunction with description feedback on current clusters to address high-dimensionality of the data is novel in the literature.

Many clustering objective functions more sophisticated than k-means distortion have been proposed. For example, co-clustering[13] looks to cluster the features and the items simultaneously. Note that this paper does not propose a new clustering objective function. Using k-means as an illustrative example, we have shown how cluster-level inter-action can be used to improve unsupervised clustering. Such interaction can be built on top of co-clustering and other proto-type based clustering models as well.

4 Problem Formulation

In the traditional k-means problem, we have a set of n data points $\{x_1, \ldots, x_n\}$, each drawn from domain $\mathcal{X}$. A clustering of these data points is defined by k clusters $\{c_1, \ldots, c_k\}$ with corresponding centers $\{\mu_1, \ldots, \mu_k\}$, and an assignment $\delta(c_i, x_j)$ of data points to clusters. We will consider each data point and the cluster center to be defined as weight vectors over the features of $\mathcal{X}$. Given such a clustering C, we can measure the distortion error for C as the summed distance of data points from their corresponding cluster centers:

$$E^x(C, \delta) = \sum_i \sum_j (\mu_i - x_j)^2 \delta(\mu_i, x_j) \tag{1}$$

Typically, given the set of data points, the goal of k-means clustering is to find the k cluster centers and an assignment of data points to clusters such that the total distortion error is minimized. In the rest of the formulation, we will not distinguish between clusters and centers. For example, we will use the notation μ to refer to both a center and its corresponding cluster.

In our version of the problem, we additionally have two different types of feedback provided by the user.

We have a set F^a of l assignment feedbacks, provided by the user possibly over different stages of the inter-active procedure. The i^{th} assignment feedback f_i^a can be represented as $\{x_i^a, \boldsymbol{\mu}_i^a, \mu_i^a\}$, indicating that data point x_i^a is assigned by the user to a specific cluster μ_i^a from the set of current clusters $\boldsymbol{\mu}_i^a$.

We also have a set F^d of m cluster description feedbacks obtained from the user, again over various stages of the inter-active process. For the i^{th} such feedback f_i^d, we assume that the user observes the top t features of a cluster ordered by weight (o_i^d) and provides his preferred feature vector (p_i^d) for it as feedback. We call t the observed description length for cluster description feedback. Accordingly, we represent f_i^d as $\{o_i^d, p_i^d\}$, where o_i^d is an ordered set of features and p_i^d is a weight vector over all features.

Though each feedback is provided at a specific stage of the interaction for a specific set of clusters C, it is desirable to make use of it at later stages when the current set of clusters is C', depending on how different C' is from C. Therefore, at any stage of the clustering, we consider *all* feedbacks that have been provided up to that stage.

In the presence of these two sets of feedbacks from the user, our reformulated clustering goal is to conform with these feedbacks as much as possible, while still maintaining low distortion error. In order to capture this in our objective function, we associate one constraint for each feedback and a penalty that the clustering algorithm has to pay for violating that constraint.

Let us first consider an assignment feedback f_i^a. The most specific constraint that arises from it is that every time the current set of clusters exactly matches $\boldsymbol{\mu}_i^a$, the data point x_i^a always has to be assigned to the specific cluster μ_i^a from among them. The penalty for violating this constraint is the distance between μ_i^a and the cluster $\delta(x_i^a)$ to which x_i^a is assigned instead. (Note that, without ambiguity, we have overloaded the symbol δ to use $\delta(x_i^a)$ as a function that returns a specific cluster.) However, this very specific interpretation would render this constraint irrelevant at later stages of the

clustering when the current set of clusters is even slightly different from $\boldsymbol{\mu}_i^a$. To get around this, we define the relevance $R_i^a(C)$ of an assignment constraint f_i^a, given a current set of clusters C, as the 'similarity' of C with the set of clusters $\boldsymbol{\mu}_i^a$ specified in the feedback. For a 'similar' set of clusters C, there is no penalty when x_i^a is assigned to the cluster $N_i^a(C)$ which is 'nearest' to μ_i^a in the current cluster set C. If, however, x_i^a is assigned to some cluster $\delta(x_i^a)$ which is different from $N_i^a(C)$, then the clustering algorithm has to pay a penalty equal to the distance between $\delta(x_i^a)$ and $N_i^a(C)$. The total assignment error takes into account both the penalty and the current relevance of the constraint. The higher the relevance, the higher is the assignment error for violating the constraint.

In summary, the clustering error associated with an assignment feedback f_i^a is captured as

$$E_i^a(C, \delta) = (\delta(x_i^a) - N_i^a(C))^2 \times R_i^a(C) \tag{2}$$

The total error for the entire set of assignment feedbacks F^a is obtained by summing over the errors for the individual feedbacks: $E^a(C, \delta) = \sum_{i=1}^{l} E_i^a(C, \delta)$.

The relevance of the current set of clusters C to the feedback clusters $\boldsymbol{\mu}_i^a$ is measured using the best mapping $M_i^a(C)$ between the two sets of clusters. The weakness of such a mapping can be measured by the summed distances between mapped clusters from the two sets. The relevance is then defined using an exponential function as

$$R_i^a(C) = \exp(-(\sum_{\mu \in C} (\mu - M_i^a(\mu))^2) \tag{3}$$

Let us now consider a cluster description feedback $f_i^d \in F^d$. The most specific constraint that can be associated with f_i^d is that for any cluster μ from the current set of clusters C, if the observed description of μ is the same as the description o_i^d in the feedback, then the center μ of the cluster should match the user preferred weight vector p_i^d. Recall that the observed description is the ordered set $\text{top}(\mu, t)$ of top t features of μ. In case the current and preferred weight vectors (μ and p_i^d) over features do not match, the clustering algorithm has to pay a penalty equal to the distance between the two weight vectors.

As for the assignment feedback, the specificity of this interpretation would make f_i^d irrelevant for most clusters at later stages, where the top feature sequence $\text{top}(\mu, t)$ does not exactly match o_i^d. We deal with this, as before, by introducing a relevance measure $R_i^d(\mu)$ for each cluster description feedback f_i^d and any cluster μ. The higher the relevance of the feedback for any cluster, the higher is the description error for not conforming with it. The relevance $R_i^d(\mu)$ of a description feedback may be measured using various similarity measures defined for ordered sets:

$$R_i^d(\mu) = \text{RankSim}(\text{top}(\mu, t), o_i^d) \tag{4}$$

For simplicity, we presently define $\text{RankSim}(s_1, s_2)$ to be 1 if the *unordered* sets corresponding to s_1 and s_2 match, and 0 otherwise. We are investigating better measures than reward subset matches, such as Jaccard Similarity.

In summary, the clustering error associated with each cluster description feedback f_i^d is given as:

$$E_i^d(C,\delta) = \lambda_i \sum_{\mu \in C} (\mu - p_i^d)^2 \times R_i^d(\mu) \tag{5}$$

where λ_i is the strength of the i^{th} cluster description feedback. To appreciate the use of λ_i, observe that for assignment feedback, the algorithm can typically satisfy the user by assigning the feedback point to the user specified cluster. However, for description feedback, the points currently assigned to a cluster also affect the position of its new center in conjunction with the user specified description. λ_i is used to specify the relative importance of the user's feedback and the assigned points. We further elaborate on the role of λ_i in Section 5.

The total error for the entire set of cluster description feedbacks F^d is obtained by summing over the errors for the individual feedbacks: $E^d(C,\delta) = \sum_{i=1}^{m} E_i^d(C,\delta)$.

Finally, the total clustering error is the sum of the errors due to distortion, assignment constraints and cluster description constraints:

$$E(C,\delta) = E^x(C,\delta) + E^a(C,\delta) + E^d(C,\delta) \tag{6}$$

Our goal is to find the optimal combination of clusters and assignments that minimize this total error.

5 Interactive Clustering Algorithm

In this section, we look at an algorithm that iteratively alternates between interacting with the user to acquire feedback and minimizing the total error in Equation (6) considering all the feedback obtained from the user so far over all stages of the algorithm. Algorithms for proto-type or model based clustering typically follow an iterative alternating optimization style, where each step consists of two sub-steps - prototype update and re-assignment. In the following subsections, we describe how these two steps can be modified to handle user feedback, and how the user interacts with the algorithm to provide feedback.

Cluster Update: In the cluster update sub-step, the existing clusters C are updated based on the current assignment δ of data points to clusters, and the current relevance $R^a(C)$ and $R^d(C)$ of the feedbacks F^a and F^d. Unfortunately, the different clusters cannot be updated independently as for the traditional k-means algorithm. This is because the feedbacks introduce dependencies across clusters. As a result, we cyclically update each of the k clusters keeping the other $k-1$ clusters fixed. This procedure is repeated until all the clusters stabilize. When the other $k-1$ clusters are held fixed, along with the assignments and the feedback relevances, updating cluster μ_i to minimize total error becomes a quadratic optimization problem. Solving it leads to the following update step:

$$\mu = \frac{1}{Z} \times \sum_x x\delta(x,\mu) + \sum_{i=1}^{l} R_i^a(C)[\delta(x_i^a,\mu)N_i^a(C) +$$

$$\sum_{\mu'} I(\mu', N_i^a(C))\delta(x_i^a, \mu')\mu'] + \sum_{i=1}^{m} \lambda_i R_i^u(\mu) p_i^d$$

where $I()$ is the indicator function, and Z is an appropriate normalization term.

The update rule may be interpreted as follows.

The first term shows the traditional movement of the cluster towards the centroid of the data points currently assigned to it. The second and third terms demonstrate the dependence on the other current centers brought about by the assignment constraints. An assignment feedback f_i^a is relevant for cluster μ either if the feedback data point x_i^a is currently assigned to this cluster, or if μ is the currently preferred cluster $N_i^a(C)$ for feedback f_i^a. In the first case, the cluster μ moves towards that current cluster $N_i^a(C)$ which is the currently preferred cluster for the feedback f_i^a. This is reflected by the second term. In the other case, cluster μ tries to move closer to that cluster μ' to which the feedback point x_i^a is currently assigned. This is reflected by the third term. Both of these movements are influenced by the current relevance $R_i^a(C)$ of the assignment feedback in question.

The effect of the cluster description feedbacks is captured by the last term. For any description feedback f_i^d that is relevant for this cluster, the cluster moves closer to the preferred description p_i^d in the feedback. As before, this movement is also tempered by the relevance $R_i^u(\mu)$ of the feedback for this cluster.

Finally, the updated position of the cluster is the net effect of the influence of all the relevant assignment and description constraints, as well as all of the data points currently assigned to this cluster. Observe that in the update rule, the user preferred description p_i^d for a description feedback behaves similarly to any other data point assigned to the cluster, and would have minimal effect in determining its new position without the weighting term λ_i.

Once all of the k clusters have been iteratively updated and have stabilized, the relevance R^a and R^d of the assignment and description constraints is recalculated based on the updated cluster positions, according to Equation (3) and Equation (4) respectively.

Point Reassignment: In the re-assignment step, the assignment δ of the data points is recalculated based on the updated cluster positions and the current relevance of the constraints. The contribution to clustering error by assigning a data point x to an existing cluster μ can be calculated by considering the distance from the cluster, and, for any assignment feedback f_i^a specified on the point x, the distance of μ from the currently preferred cluster $N_i^a(C)$ for the feedback, and its current relevance $R_i^a(C)$:

$$(\mu - x)^2 + \sum_{i=1}^{l} R_i^a(C) I(x, x_i^a)(\mu - N_i^a(C))^2$$

where $I()$ is again the indicator function. The point is then assigned to that cluster μ among the k current clusters for which this assignment error is minimized. Observe that cluster description feedbacks do not influence the assignment of data points. Also observe that, unlike cluster updates, the reassignment of each data point can still be done independently of the other data points, as in the traditional k-means algorithm.

```
     Algo InteractiveCluster
     Params: Set of data points X, Int k

1.   Initialize F^a and F^d to empty set

     % Initialize clusters
2.   Initialize k clusters in C
3.   Iterate n times or until convergence
4.       Assign each data point in X to nearest cluster
5.       Recompute k clusters from assigned data points

     % Start inter-active k-means
6.   Iterate until user is satisfied with C
7.       Acquire new feedback and add to F^a and F^d
8.       Iterate n times or until convergence
9.           Iteratively update each of k clusters in C
                 based on relevance R^a, R^d and assignment δ
10.          Re-calculate relevance R^a, R^d, based on updated clusters C
11.          Re-calcluate assignment of data points in X
                 based on updated clusters C and relevance R^a

12.  Return k clusters C and assignment δ
```

Fig. 2. High level pseudo-code describing the cluster-level inter-active k-means (CLIKM) algorithm

User Interaction: At each stage, after minimizing total error in Equation (6) considering all the feedback obtained so far, the algorithm returns the new set of clusters for inspection. The user browses over the new cluster descriptions and assignments and provides fresh feedback on them. While it may be possible for the user to inspect all cluster descriptions, or at least the ones that have changed significantly since his previous inspection, it is extremely unlikely that he can inspect cluster assignments of all data items. We will assume that he can provide provide only $n_f = n_a + n_d$ feedbacks at each interaction stage, where n_a is the number of assignment feedbacks and n_d is the number of description feedbacks. We assume $n_d = k$, which means he inspects all clusters, but $n_a \ll n$, where n is the number of data points. Currently, we assume the the user randomly selects n_a data points for inspecting and providing feedback. However, it is possible to do better than this, as in the case of active learning [7]. While we have done initial work on actively selecting the data points for presenting to the user for feedback, this is largely a subject of future research.

The overall cluster-level interactive k-means algorithm (**CLIKM**) is shown in Figure 2. The algorithm starts by creating an initial set of clusters in steps 2-5, based only on distortion error. Then at every step, it updates the clusters (step 9), recalculates the relevance of all feedbacks (step 10) and then re-assigns the data points based on the updated clusters and the relevance of the all constraints acquired so far (step 11). These steps are repeated until convergence based on the current set of feedbacks. The

algorithm terminates when the user is satisfied with the current clustering. Otherwise, the algorithm acquires more feedback from the user (step 7) and reclusters the data points based on the feedback set, which now additionally includes the most recently received feedbacks.

6 A Supervisor Model

Large-scale evaluations with interactive algorithms with human supervisors require a lot of time and effort. In this section, as a substitute, we describe a parameterized supervisor model for our framework, based on gold-standard cluster labels on data points.

The challenge is that cluster-level supervision is conditioned on the *current* set of clusters, which may be different from the true clusters in the gold-standard. We assume that the supervisor is able to construct a correspondence between the true clusters T in the gold standard and the current clusters C available at any stage of the interactive process. This correspondence is found using a maximum weighted matching between the true clusters and the current clusters in bipartite graph, where the edge weight between a true cluster t and a current cluster c is the number of data points from t in c.

As the first supervisor parameter, we control the supervisor's knowledge about the exact description of a true cluster t using a parameter $p \in [0, 1]$. When averaging over documents in a true cluster t to construct its description, any specific document is included in the average computation with probability p, so that the user only has partial knowledge of t's description for $p < 1.0$.

The second supervisor parameter is a recognition threshold r for true clusters from computed clusters. For exploratory data mining, the supervisor often becomes aware of clusters existing in the data as they gradually emerge during the clustering process. We assume that the supervisor recognizes a true cluster t from the current cluster c only if c has ambiguity (measured as entropy over true clusters) below threshold r, and if t is the majority true cluster within c. At any stage of the clustering algorithm, the supervisor has a set $T_r \subseteq T$ of recognized true clusters, and is able map current classes only to these true clusters.

Now, when asked to provide assignment feedback for a data point x given current clusters C and true clusters T, the supervisor first retrieves the true cluster t for x, and then returns the corresponding current cluster. On the other hand, when asked for description feedback on a current cluster $c \in C$, the supervisor first retrieves the corresponding true cluster t, and then returns its inexact description based on his knowledge level p. Note that the supervisor can provide feedback only if the relevant true cluster t belongs in his recognized set of clusters T_r.

In the experimental evaluation of our interactive clustering algorithm in the next section, we consider supervisors with different recognition levels, as well as different levels of knowledge.

Convergence. An important issue that naturally arises for any interactive data mining task is that of convergence. While emphasizing that a detailed investigation of supervisor behavior and convergence is beyond the scope of this paper, here we briefly discuss conditions under which convergence can be guaranteed in the context of our supervisor model.

At a high level, the interactive clustering process converges if and only if the supervisor provides a *consistent sequence of feedbacks*. Here we provide a very strict definition of consistency. A sequence of feedbacks is consistent if all of the following conditions hold. First, the resulting sequence of recognized cluster sets is monotonically non-decreasing, i.e., a cluster once recognized by the supervisor cannot be decided as unrecognizable later. Secondly, for two cluster description feedbacks $f_i^d \equiv \{o_i^d, p_i^d\}$ and $f_j^d \equiv \{o_j^d, p_j^d\}$ provided at two different stages of CLIKM, if they are provided on the same recognized cluster ($o_i^d = o_j^d$), then the preferred descriptions also are the same ($p_i^d = p_j^d$). Thirdly, for two assignment feedbacks $f_i^a \equiv \{x_i^a, \boldsymbol{\mu}_i^a, \mu_i^a\}$ and $f_j^a \equiv \{x_j^a, \boldsymbol{\mu}_j^a, \mu_j^a\}$ provided at different stages, if they are provided for the same data point ($x_i^a = x_j^a$) given the same current set of clusters ($\boldsymbol{\mu}_i^a = \boldsymbol{\mu}_j^a$), then the preferred cluster also has to be the same ($\mu_i^a = \mu_j^a$). Under the above conditions, the interactive process is guaranteed to convergence.

We think that it is possible to relax the first two conditions, but note that third is an absolute requirement. Also, the number of iterations to convergence for a consistent feedback sequence depends on multiple factors, such as the rate of growth of the recognized cluster set, and for multiple assignment feedbacks on the same data point, the similarity between their set of clusters.

7 Experiments

In this section, we experimentally evaluate the effectiveness of our proposed interactive clustering framework. We considered two benchmark real-world text categorization datasets from two different domains, Twenty Newsgroups[1] and Reuters-21578[2]. The goal of the clustering task is to produce clusters that correspond to the gold-standard categories. Generally, this is not expected of unsupervised clustering. We investigate how accurately and efficiently the gold standard categories can be discovered when provided with semi-supervision. We also investigate the relative impact of assignment and cluster description feedback, and the effect of different supervisor parameters on our cluster-level interactive framework. We next describe our datasets, baselines and evaluation metrics before discussing experimental results.

Datasets: For our first dataset (8NG), we selected eight 20 Newsgroup classes (all of $rec.*$ and $sci.*$) having 1000 documents each. Our second dataset (R10) was created by selecting the top 10 categories from the Reuters-21578 corpus, and including all train/test documents, resulting in a collection of 9118 documents. We pre-processed all documents in a standard way using word stemming, and pruning stop-words and infrequent words (occurring less than 5 times in the dataset). We have made the actual processed datasets and their descriptions available online[3].

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

[2] http://www.daviddlewis.com/resources/testcollections/reuters21578/

[3] http://www.godbole.net/shantanu/work/ecml10iclust.html

Evaluation Metric: To evaluate a set of clusters against a gold-standard, we check the correctness of clustering decisions over all pairs of data points. We report the standard F1 measure and Adjusted Rand Index (ARI) over the pairwise clustering decisions. The F1 measure is the harmonic mean of precision and recall over pairwise decisions. The Adjusted Rand Index is defined as $2(ab-cd)/((a+d)(d+b)+(a+c)(c+b))$ where a is the number of true positive pairs, b is the number of true negative pairs, c is the number of false positive pairs and d is the number of false negative pairs. We also evaluated using Normalized Mutual Information and found the trends to be similar to that with F1 and ARI.

Baselines: As the first baseline for our cluster-level interactive k-means algorithm (**CLIKM**), we consider completely unsupervised k-means (**KM**). We also compare against pair-wise constrained clustering with instance-level must-link and cannot-link constraints (**PCC**) [3]. For this comparison, we incrementally provide pair-wise constraints to PCC and cluster-level constraints to CLIKM and compare their performance for the same number of provided constraints. Since PCC infers additional constraints from the provided ones using transitivity of must-link constraints, the actual number of constraints considered by PCC is much larger than the provided number. In contrast, the actual and provided number of constraints is the same for CLIKM. While doing this comparison, it needs to be borne in mind that the nature of the two constraints are quite different from each other. First, since CLIKM constraints are conditioned on the current set of clusters, they cannot be converted to an equivalent set of independent pair-wise constraints over data points. Secondly, the supervisor effort required to provide these two types of constraints will also be quite different, and can only be measured using extensive user studies. Keeping in mind these differences, we study the relative performance of PCC and CLIKM when given an equal — though not necessarily equivalent — number of constraints.

Since all of these algorithms only find local optima, when comparing any two, we provide them with the same initialization. All the reported plots are averaged over 10 runs.

Parameter Settings: As default parameters, we set observed cluster description length $t = 10$, supervisor recognition threshold $r = 0.95$ and knowledge level $p = 0.25$. The strength of description feedback λ is set to be the average cluster size (n/k), so that the importance of the supervisor's feedback is roughly the same as that of the points assigned to the cluster. We set the number of feedbacks at each step $n_f = 200$ for both CLIKM and PCC. (Trends are similar with $n_f = 100$ and 200) For CLIKM, description feedback is provided for all current clusters ($n_d = k$), so that assignment feedback is provided for $n_a = 200 - k$ random data points. For PCC, must-link or cannot-link feedback is provided for n_f random pairs of data points. Recall that the actual number of constraints considered by PCC is much larger than n_f, since PCC infers more constraints using transitivity of must-link constraints.

Experiment 1 - CLIKM vs Baselines: In our first experiment, we compare CLIKM with point assignment and cluster description feedback, PCC with pair-wise item-level feedback, and KM on two datasets. The results are shown in Figure 3, where clustering performance is plotted against the cumulative number feedbacks provided to PCC and CLIKM. The trends are similar for F1 and ARI as the evaluation measure. Expectedly,

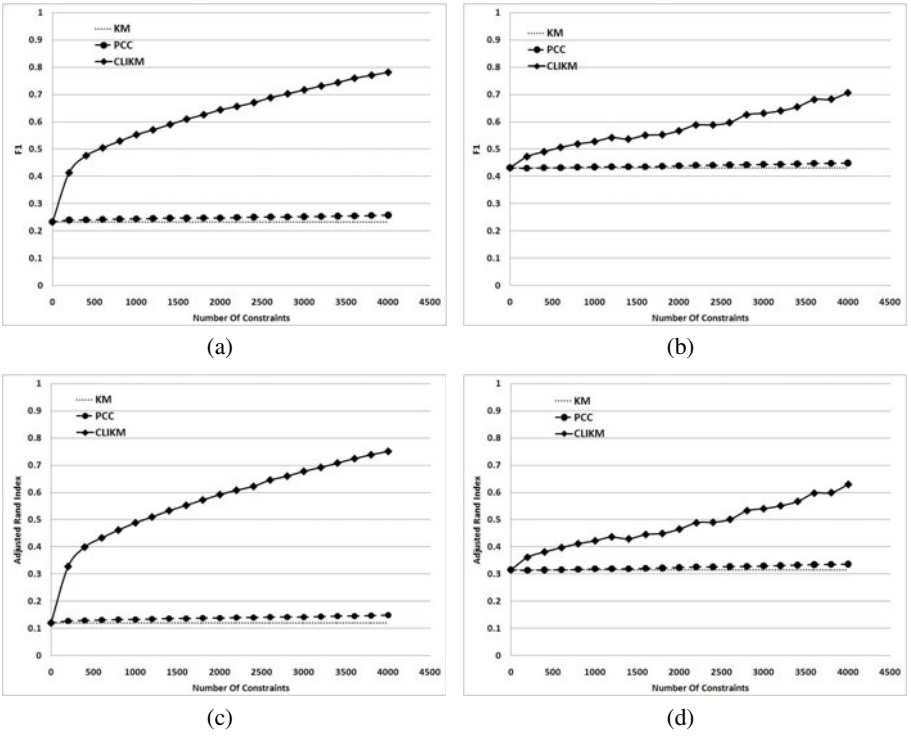
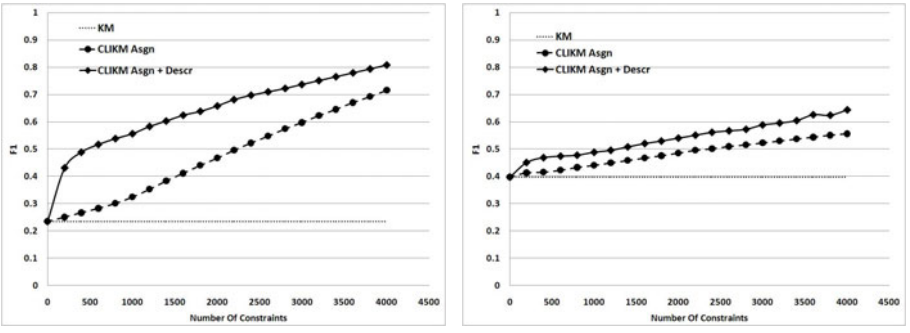


Fig. 3. Performance (F1 and ARI) of CLIKM and PCC vs number of feedbacks from the user against KM as baseline on 8NG (a,c) and R10 (b,d)

with increasing number of feedbacks, the performance CLIKM improves significantly over unsupervised KM. Performance improves most sharply at the beginning, so that after a few hundred feedbacks F1 increases from 0.22 to 0.4 for 8NG and from 0.4 to 0.47 for R10, and increases steadily, but at a slower rate, after that. This is because the user is able to recognize all true clusters during the very first interaction with the default recognition threshold $r = 0.95$. Also observe that performance of CLIKM drops slightly at a couple of places for R10 in Figure 3(b,d). This is due to the supervisor's inexact knowledge when providing description feedback ($p = 0.25$). We investigate the effect of the user's knowledge and recognition ability in greater detail later in the section.

Interestingly, the rate of performance improvement for CLIKM is significantly higher than PCC for both datasets. One potential reason for this is that the space of constraints is quadratic in the number of data items for PCC. In comparison, CLIKM needs at most a linear number of constraints for each clustering iteration, and the number of iterations is usually a constant. As a result, the user can drive the clustering towards his desired state with significantly fewer feedbacks using CLIKM.

In the rest of our experiments, we report only F1 numbers. All trends are similar with ARI.



(a) (b)

Fig. 4. Effect of assignment and cluster description feedback on CLIKM for (a) 8NG and (b) R10

Experiment 2 - Ablation Study: In our second experiment, we perform an ablation study to evaluate the impact of assignment and cluster description feedback separately. The results are shown in Figure 4. The plots clearly shows that improvement brought about by cluster description feedback on top of that from assignment feedback. It demonstrates that cluster description feedback enables the user to guide the clustering much faster than when empowered only with assignment feedback.

Experiment 3 - Varying Supervisor Parameters: The success of interactive clustering depends a lot on the user's ability to recognize desired clusters, his knowledge about the correct description of these clusters and the strength of description feedback that he sets. We evaluate this over the next two experiments.

First, in Table 1, we record the effect of providing description feedback to CLIKM with different combinations of user knowledge p and strength of description feedback λ, after it has already received 3000 assignment feedbacks. The first trend is that performance improves with supervisor knowledge when λ is fixed, over all 3 columns. Recall that $\lambda = n/k$ corresponds to equally weighting user's cluster description feedback and the data points currently assigned to a cluster, so that the first two columns correspond to weighting the feedback 10 times and 2 times lower than the data respectively. The rows provide a more interesting insight. For reasonable supervisor knowledge, performance improves with higher λ. However, when supervisor knowledge is weak (first two rows), increasing λ hurts performance. This suggests that when the supervisor is not confident

Table 1. Clustering performance (F1) after 3000 feedbacks for varying user knowledge (p) and user confidence (λ) on R10

	$\lambda = n/10k$	$\lambda = n/2k$	$\lambda = n/k$
p=0.001	0.486	0.471	0.410
p=0.01	0.487	0.497	0.487
p=0.10	0.489	0.502	0.522
p=0.25	0.490	0.510	0.511

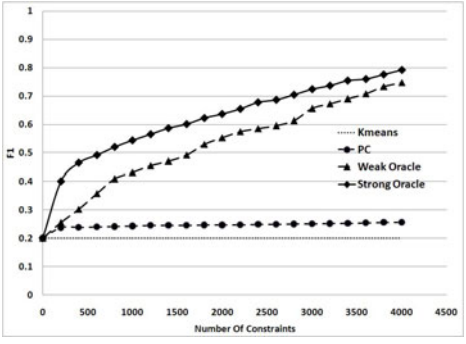


Fig. 5. F1 using strong ($r = 0.95$) and weak supervisor ($r = 0.9$) for 8NG

about his knowledge of the clusters, he should allow the data to influence the clustering more than his supervision.

Finally, we explore the impact of recognition threshold r of the supervisor. In Figure 5, we compare CLIKM performance with the default strong supervisor ($r = 0.95$) against that with a weak supervisor ($r = 0.9$). For the strong supervisor, performance improves significantly right at the beginning when all the true clusters are recognized and description feedback is provided for them. For the weak supervisor, only 50% of the true clusters are recognized at the first stage, and the rest at various later stages of the inter-active process, as marked by the jumps in performance. The gap between the two curves closes steadily as the inter-active process progresses.

Summary of Experiments: In summary, our experiments demonstrate that cluster-level semi-supervision leads to significant and steady improvements in clustering accuracy in comparison with unsupervised k-means. Improvements persist over varying levels of supervisor knowledge and cluster recognition ability. The rate of improvement is several times faster compared to that with an equal number pair-wise instance-level constraints.

8 Conclusions

In this paper we have proposed a novel semi-supervised model for interactive clustering, where the user provides two different types of feedback that align naturally with the update and assignment steps of prototype based clustering. Taking k-means as an example, we have shown how such feedback can be incorporated within prototype based objective functions as additional constraints to be satisfied. We have demonstrated the effectiveness of our model for clustering two real-life benchmark text datasets. Interestingly, our experiments show that performance increases significantly faster using this cluster-level semi-supervision compared to pair-wise instance-level supervision.

The proposed model can be further improved, for example by considering the source cluster in addition to the destination cluster for assignment feedback, and generalizing cluster description feedback with better measures of rank similarity. We also intend to

improve on our supervisor model for more detailed investigation of convergence, and also better understand the cognitive load on the user for instance-level and cluster-level semi-supervision.

We believe that cluster-level supervision can emerge as very promising alternative to pair-wise instance-level supervision for high dimensional domains such as document collections that cannot be easily visualized.

References

1. Banerjee, A., Ghosh, J.: Scalable clustering algorithms with balancing constraints. Data Mining and Knowledge Discovery 13(3) (2006)
2. Bar-Hillel, A., Hertz, T., Shental, N., Weinshall, D.: Learning distance functions using equivalence relations. In: Proc. of ICML (2003)
3. Basu, S., Banjeree, A., Mooney, E.: Active semi-supervision for pairwise constrained clustering. In: Proc. of SDM (2004)
4. Basu, S., Davidson, I., Wagstaff, K.: Constrained clustering: Advances in algorithms, theory, and applications. Chapman and Hall/CRC Data Mining and Knowledge Discovery Series (2008)
5. Basu, S., Banerjee, A., Mooney, R.: Semi-supervised clustering by seeding. In: Proc. of ICML (2002)
6. Cohn, D., Caruana, R., McCallum, A.: Semi-supervised clustering with user feedback. Tech. rep., TR2003-1892, Cornell University (2003)
7. Cohn, D., Atlas, L., Ladner, R.: Improving generalization with active learning. Machine Learning 15(2) (1994)
8. Cohn, D., Ghahramani, Z., Jordan, M.: Active learning with statistical models. Journal of Artificial Intelligence Research 4(1) (1996)
9. Davidson, I., Ravi, S.: Clustering with constraints: Feasibility issues and the k-means algorithm. In: Proc. of SDM (2005)
10. Davidson, I., Ravi, S.: Identifying and generating easy sets of constraints for clustering. In: Proc. of AAAI (2006)
11. Davidson, I., Ravi, S.: Intractability and clustering with constraints. In: Proc. of ICML (2007)
12. desJardins, M., MacGlashan, J., Ferraioli, J.: Interactive visual clustering. In: Proc. of IUI (2007)
13. Dhillon, I., Mallela, S., Modha, D.: Information-theoretic co-clustering. In: Proc. of SIGKDD (2003)
14. Gondek, D., Hofmann, T.: Non-redundant data clustering. In: Proc. of ICDM (2004)
15. Hofmann, T., Buhmann, J.: Active data clustering. In: Proc. of NIPS (1998)
16. Klein, D., Kamvar, S., Manning, C.: From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In: Proc. of ICML (2002)
17. Wagstaff, K., Cardie, C.: Clustering with instance-level constraints. In: Proc. of ICML (2000)
18. Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S.: Constrained k-means clustering with background knowledge. In: Proc. of ICML (2001)
19. Xing, E., Ng, A., Jordan, M., Russell, S.: Distance metric learning, with application to clustering with side-information. In: Proc. of NIPS (2002)