Clustering with a Reject Option: Interactive Clustering as Bayesian Prior Elicitation

Akash Srivastava
Informatics Forum, University of Edinburgh
10, Crichton St
EH8 9AB, Edinburgh, UK
akash.srivastava@ed.ac.uk

James Zou
Microsoft Research and Stanford University
One Memorial Drive,
Cambridge, MA 02142, USA
jamesyzou@gmail.com

Charles Sutton
Informatics Forum, University of Edinburgh
10, Crichton St
EH8 9AB, Edinburgh, UK
csutton@inf.ed.ac.uk

ABSTRACT

A good clustering can help a data analyst to explore and understand a data set, but what constitutes a good clustering may depend on domain-specific and application-specific criteria. These criteria can be difficult to formalize, even when it is easy for an analyst to know a good clustering when she sees one. We present a new approach to interactive clustering for data exploration, called TINDER, based on a particularly simple feedback mechanism, in which an analyst can choose to reject individual clusters and request new ones. The new clusters should be different from previously rejected clusters while still fitting the data well. We formalize this interaction in a novel Bayesian prior elicitation framework. In each iteration, the prior is adapted to account for all the previous feedback, and a new clustering is then produced from the posterior distribution. To achieve the computational efficiency necessary for an interactive setting, we propose an incremental optimization method over data minibatches using Lagrangian relaxation. Experiments demonstrate that TINDER can produce accurate and diverse clusterings.

1. INTRODUCTION

Clustering is a popular tool for exploratory data analysis. A good clustering can help to guide the analyst to better understanding of the data set at hand. An informative clustering captures not only the properties of the data, but also the goals of the analyst. What makes it challenging to identify a good clustering is that it is often difficult to encode the analyst's goals explicitly as machine learning objectives. Moreover, in many settings, the analyst does not have a well-specified objective in mind prior to encountering the data, but rather continuously updates her goals as she learns more through exploratory analysis. Because the clustering problem is ill-posed, many good clusterings of similar quantitative value exist for a given data set. Even if a clustering algorithm succeeds in finding a quantitatively good clustering, it still may not be what the user qualitatively wanted. Nevertheless, the data analyst may not be able to formalize precisely as a quantitative criterion what differentiates a "good" clustering from a "bad" one. Still, it seems reasonable to expect that the analyst will know a good clustering when she sees one.

This gap between formal clustering criteria and the user's exploratory intuition is the motivation for interactive clustering [9, 3, 24, 6] and alternative clustering approaches [7, 8, 18, 10]. Interactive clustering methods focus on allowing the user to specify precisely how the clustering should be improved, such as by splitting or merging clusters [9, 3]. Although this can be useful, there are other situations in which the analyst can tell that a clustering does not meet her exploratory needs, without having a clear idea of how it should be improved. Alternative clustering methods, on the other hand, produce a set of clusterings which are chosen to be as diverse as possible while still fitting the data. This supports a more exploratory type of data analysis, but many such methods do not scale well to an interactive setting, and sometimes the notion of an alternative is too coarse-grained: An analyst may wish to preserve some parts of a clustering while discarding others.

To allow the user to provide fine-grained "non-constructive" feedback on a clustering, we introduce a simple rejection-based approach to interactive clustering, in which the analyst chooses to reject a subset of clusters and replace them with different ones. This framework contains alternative clustering as the special case in which the user rejects all clusters. The system returns another clustering, which is chosen fit the data as well as possible, while avoiding the creation of any cluster that is similar to the rejected ones. To reflect the notion of "rejecting" a cluster, we call this interaction mechanism TINDER (Technique for INteractive Data Exploration via Rejection).

We formalize this process in a Bayesian framework, in which we view the interaction procedure as a mechanism for prior elicitation. After the user rejects a set of clusters, we modify the prior distribution over model parameters to severely downweight regions of the parameter space that would lead to clusters that are similar to those previously rejected. This prior downweighting is achieved through a mutual information criterion, defined in such a way to prevent the rejection feedback from simply resulting in label permutation. In interactive settings, it is important that the response to the user's feedback be produced quickly, which

KDD 2016 Workshop on Interactive Data Exploration and Analytics (IDEA'16) August 14th, 2016, San Francisco, CA, USA.

suggests the use of a stochastic method, but unfortunately our penalty function does not decompose into a simple sum over data points. To surmount this, we propose an optimization method that introduces an auxiliary distribution, similar in spirit to variational methods, but that follows a Langrangian duality type argument rather than Jensen's inequality. The resulting objective function can then be optimized using a stochastic coordinate descent algorithm over minibatches of data points, which we show to be efficient in practice.

2. RELATED WORK

Previous work on interactive clustering methods exploits various types of user feedback. One type is *must-link* and *cannot-link* constraints between pairs of data points [24, 5]. Alternately, a second type of feedback is to request that entire clusters be *split* or *merged* [9, 3, 4]. A third type of feedback is for the analyst to explicitly choose the set of features to use in the clustering procedure [6, 11]. Similarly, interactive methods have been proposed for topic models using must-link / cannot-link [1], split/merge [15, 22], and feature-level feedback [17]. While all three types of feedback improve clustering quality, they require that the analyst have a certain level of knowledge about the data set and her information need, which might not be appropriate for a highly exploratory analysis. They can also be quite demanding in requiring active guidance from the user. We are unaware of previous work that uses cluster-level accept/reject feedback like we do.

In contrast, alternative clustering methods [13, 2, 7, 18, 10, 8] focus on generating a set of high-quality clusterings that are chosen to be different from each other, which the user can select between. Work in this area has generated diverse sets of clusters by randomly reweighting features [7], by exploring the space of possible clusterings using Markov Chain Monte Carlo [8], or by penalizing the objective function to encourage clusterings to be diverse [13, 18, 10]. Our framework for interactive clustering includes alternative clustering as a special case, bridging between interactive and alternative clustering. In particular, the objective function that we propose recovers the CAMI method [10] as a special case in which the user always rejects all clusters, and the objective function is optimized jointly over all clusterings, rather than one clustering at a time in response to user feedback. Additionally, the optimization method that we propose (Section 3) is different from the previous work and necessary for obtaining interactive performance.

Our work is similar in motivation to *diverse subset selection*, which is concerned with selecting subsets of data from a collection such that the inter-set diversity and the intra-set diversity is maximized, for example in summarization [14]. The application of diverse k-best summarization presented in [12] is a related problem to alternative clustering, if one considers a set of cluster centroids to be a summary of a data set. Finally, contrastive learning is aimed at fitting a latent variable model so that the latent variables explain the difference between one data set from another, for example, a data set of Chinese news articles versus a dataset of economic articles [25]. Our current work is somewhat analogous to this, in that to support interactive data exploration, we search for different latent variable explanations of a *single* data set.

3. INTERACTIVE CLUSTERING WITH REJECTIONS

Now we describe our rejection-based framework for interactive clustering. We begin with an overview of the interaction method. The data are first clustered according to a standard clustering algorithm. We present this clustering to the analyst for inspection, for example, by displaying the data points or the features that are most closely associated with each cluster. Then if the clustering does not meet the information need of the analyst, she can provide feedback. For each cluster, the analyst can either: (a) **reject** the cluster if it is not relevant to her information need, (b) **accept** the cluster if it is relevant, or else (c) **do neither**, expressing no opinion about the cluster. Once this feedback is complete, we cluster the data again, modifying the objective function of the clustering algorithm to penalize clusters that are similar to rejected clusters, and to reward clusters that are similar to accepted ones. This modified objective encourages the algorithm to return a new clustering that still fits the data well but that respects the user feedback. The process can be repeated as many times as desired. We call each iteration of this process a *feedback iteration*.

Now we describe the clustering method used in TINDER. We formalize the interaction mechanism as a type of Bayesian prior elicitation [21]. At each feedback iteration t, we perform Bayesian clustering with parameters θ, but with a different prior $\pi_t(\theta)$ that strongly downweights parameter vectors that are associated with rejected clusters, and strongly upweights parameters that are associated with accepted clusters. We perform clustering using a standard Bayesian mixture model. Let x denote a single data item, $h \in \{1, \dots K\}$ be a discrete latent variable that indicates the cluster membership of x, and the vector θ denote all of the model parameters, that is, the parameters of the prior distribution $p(h|\theta)$ over clusters, and the parameters of the conditional distribution $p(x|h, \theta)$ of data items given the cluster label. At feedback iteration t, the data is modelled as

$$p_t(x, \theta) = \sum_h p(x|h, \theta)p(h|\theta)\pi_t(\theta). \quad (1)$$

As the subscripting in (1) suggests, the prior distribution will change after every feedback iteration (in a way that we shall discuss in a moment), but the other parts of the probabilistic model will not.

For computational reasons, we perform maximum a posteriori (MAP) estimation. Let $\mathbf{x} = (x_1 \dots x_N)$ denote the data, where x_i is a single data point, and $\mathbf{h} = (h_1 \dots h_N)$ denote an assignment of cluster labels to all data points. Then, at each feedback iteration, MAP estimation computes the parameter estimate $\theta_t = \max_\theta \log p(\theta|\mathbf{x})$ and a soft cluster assignment $p(\mathbf{h}|\mathbf{x}, \theta_t)$ over cluster labels. (Note that this distribution has the same functional form across all iterations, and the parameter θ_t could be different in iteration t.) After reviewing the clustering, the analyst chooses a set of clusters to accept and reject. Let $A_t \subseteq \{1, \dots K\}$ be the indices of the clusters that the user has accepted and $R_t \subseteq \{1, \dots K\}$ be those the user has rejected. The sets A_t and R_t are disjoint. Cluster indices that do not appear in $A_t \cup R_t$ are those clusters for which the analyst has expressed no opinion.

Now we describe how TINDER produces a revised clustering at feedback iteration t. Following a Bayesian framework, we interpret the user feedback from clusterings $0 \dots t-1$ as an indirect source of information about the analyst's prior

beliefs over θ, that she was unable to encode mathematically into the prior distribution. Therefore we define a revised prior distribution $\pi_t(\theta)$ based on all the previous feedback, which is designed in such a way that the resulting clustering, which we denote $p(\mathbf{h}|\mathbf{x}, \theta_t)$, will respect the feedback. The prior $\pi_t(\theta)$ has the form

$$\pi_t(\theta) \propto \pi_0(\theta) \prod_{s=0}^{t-1} \exp\{-\beta f_s(\theta, \theta_s)\},$$

where f_s is a function that measures how well the parameter vector θ respects the feedback (A_s, R_s) from iteration s (lower is better). The parameter β is a temperature parameter.

For example, consider the case of "reject all" feedback, in which the user has rejected all previous clusters, that is, $R_s = \{1, \ldots, K\}$ for all s. This special case has been studied in the literature under the name of alternative clustering (Section 2). In this context, we want f_s to measure the degree of similarity between the cluster distribution $p(\mathbf{h}|\mathbf{x}, \theta)$ and the cluster distribution $p(\mathbf{h}|\mathbf{x}, \theta_s)$ that the user rejected, so that new parameters θ which produce clusters similar to those from θ_s will have lower probability. A naive choice for $f_s(\theta, \theta_s)$ would be to use the negative Kullback-Leibler divergence between the distributions $p(\mathbf{h}|\mathbf{x}, \theta)$ and $p(\mathbf{h}|\mathbf{x}, \theta_s)$. However, in the context of clustering, this metric suffers from the issue of label switching, i.e., merely permuting the cluster assignments can produce high divergence.

Instead, we begin by defining a joint distribution over the individual cluster labels h and h_s that would be assigned by the current clustering and the previous clustering to the same data point x. This joint distribution is

$$p_{\theta,\theta_s}(h, h_s, x) = p(h|x, \theta) p(h_s|x, \theta_s) \tilde{p}(x), \tag{2}$$

where $\tilde{p}(x) = N^{-1} \sum_i \delta_{x,x_i}$ is the empirical distribution over data points, for the Kronecker delta function δ. This now defines a bivariate marginal distribution

$$p_{\theta,\theta_s}(h, h_s) = \frac{1}{N} \sum_{j=1}^{N} p(h|x_j, \theta) p(h_s|x_j, \theta_s) \tag{3}$$

that measures the overall dependence between the two different clusterings, marginalizing out the data. In other words, p_{θ,θ_s} is the joint distribution over pairs of cluster labels that results from randomly choosing a data item x, and clustering it independently according to the distributions $p(h|x, \theta)$ and $p(h_s|x, \theta_s)$. The distribution p_{θ,θ_s} also yields marginal distributions $p_\theta(h) = \sum_{h_s} p_{\theta,\theta_s}(h, h_s)$ and $p_{\theta_s}(h_s) = \sum_h p_{\theta,\theta_s}(h, h_s)$ for each of the individual clusterings, which are simply the prior probabilities of the cluster labels from each clustering.

Now we can define f_s. We begin with the special case of reject all feedback. The distribution $p_{\theta,\theta_s}(h, h_s)$ measures the joint distribution between the new clustering and the previous one at iteration s, so to ensure that these two clusterings are different, we simply minimize their mutual information. This yields

$$f_s(\theta, \theta_s) = I(H; H_s) = \sum_{h=1}^{K} \sum_{h_s=1}^{K} p_{\theta,\theta_s}(h, h_s) \log \frac{p_{\theta,\theta_s}(h, h_s)}{p_\theta(h) p_{\theta_s}(h_s)}. \tag{4}$$

To handle accept feedback, f_s takes a similar form, but the sign is flipped for the clusters in A_s, so that f_s encourages similarity rather than dissimilarity. More specifically:

$$f_s(\theta, \theta_s) = \sum_{h_s \in R_s} \sum_{h=1}^{K} p_{\theta,\theta_s}(h, h_s) \log \frac{p_{\theta,\theta_s}(h, h_s)}{p_\theta(h) p_{\theta_s}(h_s)} - \sum_{h_s \in A_s} \sum_{h=1}^{K} p_{\theta,\theta_s}(h, h_s) \log \frac{p_{\theta,\theta_s}(h, h_s)}{p_\theta(h) p_{\theta_s}(h_s)} \tag{5}$$

Note that clusters for which the user has said "no opinion" are in neither A_s nor R_s, and therefore such clusters have no effect on π_t, and hence no effect on the clustering in subsequent feedback iterations. This completes the definition of π_t. Now, to compute the revised clustering, we perform MAP estimation on (1), which is equivalent to maximizing

$$L_t(\theta) = \sum_{j=1}^{N} \log p(x_j|\theta) - \beta \sum_{s=1}^{t-1} f_s(\theta, \theta_s) + \log \pi_0(\theta), \tag{6}$$

where β can now be interpreted as a weighting parameter to bring the terms to a common scale. Denote by θ_t the new MAP parameter estimate, i.e., $\theta_t = \max_\theta L_t(\theta)$. Then the new clustering that is displayed to the analyst is based on the soft assignment $p(\mathbf{h}|\mathbf{x}, \theta_t)$.

Examples..

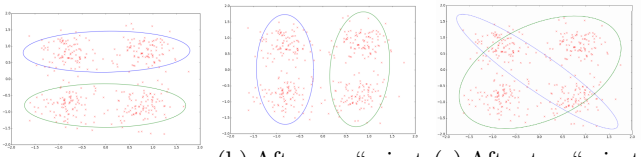
As an illustrative example, consider the 2D dataset shown in Figure 1(a), which is generated from a mixture of four isometric Gaussians. The ellipses in the figure show the clustering resulting from maximizing the likelihood of a mixture of two Gaussians using expectation maximization (EM) in the zeroth feedback iteration of TINDER. Starting from here, suppose the user rejects both clusters. Then Figure 1(b) shows the resulting clustering that TINDER generates in the next feedback iteration. Rejecting both clusters again results in the clustering in Figure 1(c). Therefore using TINDER, an analyst can obtain three quantitatively different explanations of the data in three feedback iterations.

Our per-cluster feedback framework recovers alternative clustering, in which the goal is to as explore as many diverse clusterings as possible, as the special case in which all previous clusters are rejected. By providing more specific feedback, the user can perform a more directed style of exploration, in which the user guides the clustering procedure toward a partitioning that interests her. By incorporating both alternative clustering and the more directed style of per-cluster feedback in the same framework, TINDER allows the analyst to flexibly alternate between more exploratory and more directed navigation through the space of possible clusterings.

Although we have described TINDER as a clustering method, that is, where $p(x, h|\theta)$ is a mixture model, the same logic can be applied to more general graphical models, e.g., ones that contain other latent variables in addition to x and h. All we require is that the model contains a discrete latent variable that we can use in the same way as the cluster labels h, and that MAP estimation of θ be tractable. We leave further exploration of this idea to future work.

Optimization.

In this section we discuss how to perform MAP estimation of θ, i.e., to efficiently optimize L_t. The gradient of L_t is easy to compute, so it is possible to apply standard optimization algorithms like conjugate gradient. However, for



(a) Initial clustering (b) After one "reject all" (c) After two "reject all"

Figure 1: Example of TINDER clusterings produced in three feedback iterations on synthetic data, showing (a) the initial clustering from expectation maximization (EM), and after (b) one round and (c) two rounds of "reject all clusters" feedback from the user.

an interactive algorithm, each feedback iteration needs to be relatively fast, because the computation is run while the user is waiting. To achieve interactive performance on large data sets, we would therefore prefer a stochastic gradient style of algorithm, in which each update to the parameters only depends on a small subset of data points. But the form of $f_s(\theta, \theta_s)$ makes this difficult. Notice that the distribution $p_{\theta,\theta_s}(h, h_s)$ contains a summation over data points within it, and this appears inside a log within f_s. Therefore the gradient of L_t does not decompose into a simple sum over data items.

Alternately, we could optimize L_t using the standard EM algorithm for MAP estimation. Recall that in this algorithm, the E step is unchanged from maximum likelihood, but the M step contains the log prior distribution as part of the objective. Applying this to L_t, the M step becomes

$$\theta_t \leftarrow \max_{\theta} \sum_j \sum_h q_j(h) \log p(h, x_j|\theta) - \beta \sum_s f_s(\theta, \theta_s) + \log \pi_0(\theta),$$

where q_j is the standard EM auxiliary distribution. It is not clear that this objective is any easier to optimize than (6), nor is it clear how to derive a stochastic gradient-style algorithm.

Instead we take a different approach, inspired by Lagrangian relaxation. To simplify the exposition, we will describe the optimization algorithm only for the case of "reject all" feedback, but the extension to the other types of feedback is straightforward. First we introduce an auxiliary random variable H, whose output is a cluster assignment, and whose distribution is given by a variational distribution $q_j(h)$ for each data point x_j. As in (3), we can induce a joint distribution over the random variable H and the random variable H_s whose distribution is given by $p(h_s|x_j, \theta_s)$. This joint distribution is

$$p_{q,\theta_s}(h, h_s) = N^{-1} \sum_j q_j(h) p(h_s|x_j, \theta_s).$$

Notice that this distribution, and therefore the resulting mutual information, which we denote $I_q(H; H_s)$, is a function of the variational distribution q. Then optimizing (6) is equivalent to

$$\max_{\theta,q} \log p_\theta(x) - \beta \sum_s I_q(H; H_s) + \log \pi_0(\theta). \quad (7)$$
$$\text{s.t. } \mathrm{KL}(q_j \,\|\, p(h|\theta, x_j)) = 0 \quad \forall j \in \{1, 2, \ldots N\},$$

where KL indicates the Kullback-Leibler divergence. Incorporating the constraint using a penalty term with parameter α leads to

$$\max_{\theta,q} \log p_\theta(x) - \beta \sum_s I_q(H; H_s) \quad (8)$$
$$- \alpha \sum_j \mathrm{KL}(q_j \,\|\, p(h|\theta, x_j)) + \log \pi_0(\theta).$$

If α is large enough, then the solution of (8) will be the same as for (7). Coordinate descent on (8) yields the EM-like algorithm:

"E"-Step:

$$q \leftarrow \max_q -\beta \sum_s I_q(H; H_s) - \alpha \sum_j \mathrm{KL}(q_j \,\|\, p(h|\theta, x_j)) \quad (9)$$

"M"-Step:

$$\theta \leftarrow \max_\theta E[\log p(\mathbf{x}, \mathbf{h}|\theta)]_q \quad (10)$$

This is not strictly an EM algorithm, because we lose the lower bound property that would have arisen if we had applied Jensen's inequality. However, if at the end of optimization procedure, we have that $q_j(h) = p(h|x_j, \theta)$ for all j (which will happen if α is set high enough, and can be easily checked), then θ is a local maximum of L_t.

Now we can optimize the objective in the "E" step by coordinate descent. The mutual information $I_q(H; H_s)$ still depends on all of the data points via $p_{q,\theta_s}(h, h_s)$, but now if we perform stochastic coordinate descent on each distribution q_j, then the value of $p_{q,\theta_s}(h, h_s)$ can be updated incrementally, so recomputing $I_q(H; H_s)$ does not require iterating through the entire data set. The "M" step is very fast, as it is exactly the same as the M step in the EM algorithm for maximum likelihood.

4. EXPERIMENTS

In this section, we evaluate the diversity and the quality of the clusterings produced by TINDER. Following previous work in alternative and interactive clustering [9, 3, 24, 6, 7, 8, 18, 10], we present an automatic evaluation in which we measure the quality of clusterings by comparing how well the clusters correspond to gold standard labels. An automatic evaluation allows us to compare the output of the learning algorithms directly without dealing with difficult and potentially confounding aspects of user interface design.

We evaluate TINDER for both per-cluster feedback (TINDER: Per Cluster), in which the user feedback is attempting to drive the system toward a given clustering, and in global mode (TINDER: Global), which is the alternative clustering setting in which the user is exploring the data set by rejecting all clusters. To replicate per-cluster feedback within an automatic evaluation, we simulate a user using the following heuristic. At each feedback iteration, the user provides feedback on one cluster at a time. If the cluster purity is below 50% with respect to the gold standard labels, the simulated user rejects the cluster otherwise the cluster is accepted. If none of the clusters are above this threshold, then the entire clustering is rejected. The reasoning here is that we are simulating a user whose information need is to find a clustering similar to that defined by the gold standard labels.

We compare TINDER to the popular *Decorrelated-kMeans* (Dec-kMeans) [18] algorithm for alternative clustering, which uses a penalized k-means objective to encourage the centroids from the previous clustering to be orthogonal to those from

the current clustering. Since this method in the default setting produces just two clusterings, we extended it by adding additional error and penalty terms to produce more than two clusterings at a time. We also compare to running EM using different random initializations, which will produce different clusterings because of its sensitivity to initialization. We call this method *random restarts*.

We use two data sets: a small collection of 640 face images of 20 people in different orientations from the CMU face dataset [20] and a large collection of 10,000 thumbnail images from CIFAR10 [19]. The CIFAR10 data is significantly larger than other data sets that have been used in the alternative clustering literature. The CIFAR10 data set is labeled with 10 classes. In the CMU face dataset, each image has three different types of labels: the identity of the person in the image, their gender, and the pose (orientation of the face). This provides three natural clusterings of the data. To obtain features, for the CIFAR10 dataset we use the the embedding generated by training the VGG network [23] on the CIFAR10 training set. For the CMU face dataset, we apply PCA to the raw pixel values and retain 90% of the variance from the original data.

To evaluate diversity, the Adjusted Rand Score (ARS) is used to measure the distance between two clusterings [16]; an ARS of 0 indicates no association between a pair of clusterings, and a score of 1 indicates a perfect match between the clusterings. To measure the diversity of a set of clusterings, we average the pairwise ARS over all pairs of clusterings in the set. To evaluate the quality of a clustering, we report its purity with respect to a set of ground truth labels. To evaluate the quality of a set of clusterings, we report the maximum purity of any clustering the set, reflecting the idea that, after examining a set of different clusterings, an analyst can choose the single clustering that she finds most useful.

We use a mixture of Gaussians (GMM) for modeling the CMU Face and the CIFAR10 datasets. In both cases, for the zeroth feedback iteration we set $\pi_0(\theta)$ to be one. We reabsorb the relaxed Lagrange multiplier α from Section 3 in β as well. Empirically, we found that TINDER performs well by simply setting β such that the penalty term, $\beta \sum_s f$, and the log-likelihood have the same order of magnitude. Dec-kMeans also has a similar weighing parameter λ, which we set according to the guidelines provided by the original authors [18].

All methods are allowed the same number of feedback iterations, i.e., all methods are evaluated on the same number of clusterings. To ensure this, first we run TINDER in per-cluster mode until the clustering stabilizes, that is, until the simulated user accepts all clusters. Then we run each of the other methods, including TINDER Global, for the same number of feedback iterations that was required by TINDER: Per Cluster. All methods are repeated 20 times from different random initializations, and we report the average maximum quality and the average diversity over the repetitions. For the CMU face data set, we use $K = 8$ clusters, whereas for CIFAR10, we use $K = 10$ clusters.

4.1 Results

Table 1 summarizes the clustering quality of the methods. The three columns for the CMU Face data report quality with respect to each of the three different types of gold standard labels. For TINDER: Per Cluster, the feedback from the simulated user is based on same set of gold standard

(a) Clustering 0 (no feedback)

(b) Clustering 1 (after one feedback iteration). For space, only two of the ten clusters are shown.

(c) Clustering 2 (after two feedback iterations). For space, only one of the ten clusters is shown.

(d) Clustering 5 (after five feedback iterations). For space, only four of the ten clusters are shown.

Figure 2: Example of TINDER clusterings on the CIFAR10 dataset.

labels that is used to evaluate quality; that is, the goal is to measure the effectiveness of per-cluster feedback at reaching a specific clustering that the user discovers during exploration. We find that TINDER: Per Cluster outperforms or matches the other methods, indicating that the more specific guidance provided by per-cluster feedback indeed leads to higher quality clusterings. On average TINDER requires four feedback iterations to stabilize. The quality of the clusters from the other three methods are similar to each other.

We report the clustering diversity in Table 2. For both the datasets, TINDER: Global clearly returns a much more diverse set of clusterings, outperforming both the baseline methods by a significant margin. On CIFAR10, Dec-kMeans oscillates between two similar clusterings and as a result performs worse than random restarts. These results indicate that overall, TINDER: Global returns a more diverse set of clusters of equivalent quality to the other alternative clustering methods. As expected, TINDER: Per Cluster results are not as diverse, because the goal of per-cluster feedback is to drive the method towards a specific target clustering, which necessarily reduces diversity.

To illustrate the effect of the feedback, we display in Figure 2 some of the clusters from TINDER: Global on the CIFAR10 dataset. TINDER: Global clusterings are not just able to find all the original CIFAR10 clusters but other meaningful clusters as well. In the figure, each of the rows represents a cluster and shows the top 6 images from that cluster ordered by their likelihood under the cluster. Figure 2(a) shows the initial clustering for $K = 10$ with no feedback. Clustering 1 (Figure 2(b)) is produced by TINDER after a single iteration of "reject all" feedback. We see that Clusters 2 and 3 from Clustering 0 (which contain deer and horses, respectively) are replaced in Clustering 1 by clusters 2' and 3', which contain

Table 1: Clustering quality, measured by purity to ground truth labels (higher is better).

	CIFAR10	CMU Face Person	CMU Face Gender	CMU Face Pose
Random Restarts	0.89	0.37	0.87	**0.44**
Dec-kMeans	0.90	0.37	0.86	0.42
TINDER: Global	0.89	0.37	0.89	0.40
TINDER: Per Cluster	**0.93**	**0.39**	**0.93**	**0.44**

Table 2: Diversity of returned sets of clusterings, measured by Adjusted Rand Score (lower is better).

	CIFAR10	CMU Face
Random Restarts	0.56	0.55
Dec-kMeans	0.83	0.38
TINDER: Global	**0.15**	**0.27**
TINDER: Per Cluster	0.88	0.59

large animals (Cluster 2') and horses with riders (Cluster 3'). The result of the next feedback iteration is shown in Clustering 2 (Figure 2(c)). We see that Cluster 9 has been replaced by Cluster 9', which contains images of birds and planes, which were scattered over multiple clusters in Clustering 0. Finally, after five feedback iterations, Clustering 5 (Figure 2(d)) includes clusters of ships (Cluster 4'), cats (Cluster 5'), birds (Cluster 6') and planes (Cluster 9"), which did not exist in Clustering 0. These new clusters replace Clusters 4-6 and 9 from Clustering 0, which have low purity.

As for running time, each feedback iteration of TINDER requires a few seconds for the CMU Face data set and under a minute for CIFAR10. Our implementation of Dec-kMeans performs comparably. Both methods take the same amount of time as standard EM without feedback. Finally, we observe that TINDER can be easily applied to any mixture model, not just a mixture of Gaussians. To demonstrate this, we also applied TINDER to a mixture of multinomials model for text data (see supplementary material).

5. CONCLUSION

In this paper we have presented a method for interactive clustering based on a particularly simple feedback mechanism, in which an analyst can reject individual clusters and request new ones. The interaction is formalized as a method of prior elicitation in a Bayesian model of clustering. We showed the efficacy of this method on two real world datasets. An interesting direction of future work would be to extend our approach to other graphical models for data exploration, such as topic models.

6. REFERENCES

[1] D. Andrzejewski, X. Zhu, and M. Craven. Incorporating domain knowledge into topic modeling via Dirichlet forest priors. In *International Conference on Machine Learning (ICML)*, pages 25–32. ACM, 2009.

[2] E. Bae and J. Bailey. COALA: A novel approach for the extraction of an alternate clustering of high quality and high dissimilarity. In *IEEE International Conference on Data Mining (ICDM)*, pages 53–62, 2006.

[3] M.-F. Balcan and A. Blum. Clustering with interactive feedback. In *Algorithmic Learning Theory*, pages 316–328. Springer, 2008.

[4] M.-F. Balcan, A. Blum, and S. Vempala. A discriminative framework for clustering via similarity functions. In *ACM Symposium on Theory of Computing*, pages 671–680. ACM, 2008.

[5] S. Basu, A. Banerjee, and R. J. Mooney. Active semi-supervision for pairwise constrained clustering. In *SIAM International Conference on Data Mining (SDM)*, volume 4, pages 333–344, 2004.

[6] R. Bekkerman, H. Raghavan, J. Allan, and K. Eguchi. Interactive clustering of text collections according to a user-specified criterion. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 684–689, 2007.

[7] R. Caruana, M. Elhawary, N. Nguyen, and C. Smith. Meta clustering. In *Data Mining, 2006. ICDM'06. Sixth International Conference on*, pages 107–118. IEEE, 2006.

[8] Y. Cui, X. Z. Fern, and J. G. Dy. Learning multiple nonredundant clusterings. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 4(3):15, 2010.

[9] D. R. Cutting, D. R. Karger, J. O. Pedersen, and J. W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *ACM SIGIR conference on Research and Development in Information Retrieval*, pages 318–329. ACM, 1992.

[10] X.-H. Dang and J. Bailey. Generation of alternative clusterings using the CAMI approach. In *SIAM International Conference on Data Mining (SDM)*, 2010.

[11] S. Dasgupta and V. Ng. Which clustering do you want? inducing your ideal clustering with minimal feedback. *Journal of Artificial Intelligence Research*, 30:581–632, 2010.

[12] J. A. Gillenwater, R. K. Iyer, B. Lusch, R. Kidambi, and J. A. Bilmes. Submodular hamming metrics. In *Advances in Neural Information Processing Systems*, pages 3123–3131. 2015.

[13] D. Gondek and T. Hofmann. Non-redundant data clustering. In *IEEE International Conference on Data Mining (ICDM)*, 2004.

[14] B. Gong, W.-L. Chao, K. Grauman, and F. Sha. Diverse sequential subset selection for supervised video summarization. In *Advances in Neural Information Processing Systems*, pages 2069–2077, 2014.

[15] Y. Hu, J. Boyd-Graber, and B. Satinoff. Interactive topic modeling. In *Association for Computational Linguistics*, 2011.

[16] L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985.

[17] J. Jagarlamudi, H. Daumé III, and R. Udupa. Incorporating lexical priors into topic models. In *Proceedings of the 13th Conference of the European*

Chapter of the Association for Computational Linguistics, pages 204–213. Association for Computational Linguistics, 2012.

[18] P. Jain, R. Meka, and I. S. Dhillon. Simultaneous unsupervised learning of disparate clusterings. *Statistical Analysis and Data Mining*, 1(3):195–210, 2008.

[19] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images, 2009.

[20] T. M. Mitchell. Machine learning. *Computer Science Series (McGraw-Hill, Burr Ridge, 1997) MATH*, 1997.

[21] A. O'Hagan, C. E. Buck, A. Daneshkhah, J. R. Eiser, P. H. Garthwaite, D. J. Jenkinson, J. E. Oakley, and T. Rakow. *Uncertain judgements: eliciting experts' probabilities*. John Wiley & Sons, 2006.

[22] Q. Pleple. Interactive topic modeling. Master's thesis, University of California, San Diego, 2013.

[23] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[24] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl. Constrained k-means clustering with background knowledge. In *International Conference on Machine Learning (ICML)*, pages 577–584, 2001.

[25] J. Y. Zou, D. J. Hsu, D. C. Parkes, and R. P. Adams. Contrastive learning using spectral methods. In *Advances in Neural Information Processing Systems*, pages 2238–2246, 2013.