Mach Learn (2006) 65:3–29
DOI 10.1007/s10994-006-6540-7

Semi-supervised model-based document clustering: A comparative study

Shi Zhong

Received: January 5, 2004 / Revised: November 23, 2005 / Accepted: November 23, 2005 / Published online: 28 March 2006

Abstract Semi-supervised learning has become an attractive methodology for improving classification models and is often viewed as using unlabeled data to aid supervised learning. However, it can also be viewed as using labeled data to help clustering, namely, semi-supervised clustering. Viewing semi-supervised learning from a clustering angle is useful in practical situations when the set of labels available in labeled data are not complete, i.e., unlabeled data contain new classes that are not present in labeled data. This paper analyzes several multinomial model-based semi-supervised document clustering methods under a principled model-based clustering framework. The framework naturally leads to a deterministic annealing extension of existing semi-supervised clustering approaches. We compare three (slightly) different semi-supervised approaches for clustering documents: *Seeded damnl*, *Constrained damnl*, and *Feedback-based damnl*, where *damnl* stands for multinomial model-based deterministic annealing algorithm. The first two are extensions of the seeded k-means and constrained k-means algorithms studied by Basu et al. (2002); the last one is motivated by Cohn et al. (2003). Through empirical experiments on text datasets, we show that: (a) deterministic annealing can often significantly improve the performance of semi-supervised clustering; (b) the constrained approach is the best when available labels are complete whereas the feedback-based approach excels when available labels are incomplete.

Keywords Semi-supervised clustering · Seeded clustering · Constrained clustering · Clustering with feedback · Model-based clustering · Deterministic annealing

Editor: Andrew Moore

S. Zhong (✉)
Department of Computer Science and Engineering, Florida Atlantic University, Boca Raton, FL 33431
e-mail: zhong@cse.fau.edu

1. Introduction

Learning with both labeled and unlabeled data, also called *semi-supervised* learning or *transductive* learning,[1] has recently been studied by many researchers with great interest (Seeger, 2001), mainly as a way of exploiting information in unlabeled data to enhance the performance of a classification model (traditionally trained using only labeled data). A variety of semi-supervised algorithms have been proposed, including co-training (Blum & Mitchell, 1998), transductive support vector machine (Joachims, 1999), entropy minimization (Guerrero-Curieses & Cid-Sueiro, 2000), semi-supervised EM (Nigam et al., 2000), graph-based approaches (Blum & Chawla, 2001; Zhu et al., 2003), and clustering-based approaches (Zeng et al., 2003). They have been successfully used in diverse applications such as text classification or categorization, terrain classification (Guerrero-Curieses & Cid-Sueiro, 2000), gesture recognition (Wu & Huang, 2000), and content-based image retrieval (Dong & Bhanu, 2003).

Despite the empirical successes, negative results have been reported (Nigam, 2001) and the usefulness of unlabeled data and transductive learning algorithms have not been appreciated by theoretical studies. For example, Castelli and Cover (1996) proved that unlabeled data are exponentially less valuable than labeled data in classification problems, even though the proof comes with strong assumptions that the input distribution is known completely and that all class-conditional distributions can be learned from unlabeled data only. Using Fisher information matrices to measure the asymptotic efficiency of parameter estimation for classification models, Zhang and Oles (2000) theoretically and experimentally questioned the usefulness and reliability of transductive SVMs for semi-supervised learning. Cozman et al. (2003) recently presented an asymptotic bias-variance analysis of semi-supervised learning with mixture models. They showed that unlabeled data will always help if the parametric model used is "correct" or has low bias, meaning that the true model is contained in the parametric model family. If the model is not "correct", additional unlabeled data may hurt classification performance. Their analysis is limited due to the assumptions that all classes are available in the labeled data and the distributions for both labeled and unlabeled data are the same.

These results remind us to be careful in applying semi-supervised learning to real world applications: In a text categorization task, we might not have all categories in the labeled data; in a network intrusion detection system, we will certainly encounter new attack types never seen before. Furthermore, new classes may follow different models than those used to characterize labeled data. Since semi-supervised classification models are not well positioned to detect new classes, we are motivated to look at semi-supervised clustering, which exploits labeled data to enhance clustering results on unlabeled data. Semi-supervised clustering can be used to discover new classes in unlabeled data in addition to assigning appropriate unlabeled data instances to existing categories.

In semi-supervised clustering, labeled data can be used as initial seeds (Basu et al., 2002), constraints (Wagstaff et al., 2001), or feedback (Cohn et al., 2003). All these existing approaches are based on model-based clustering (Zhong & Ghosh, 2003) where each cluster is represented by its "centroid".[2] Seeded approaches use labeled data only to help initialize cluster centroids; Constrained approaches keep the grouping of labeled data unchanged (as fixed constraints) throughout the clustering process; Feedback-based approaches first run a regular clustering process and then adjust resulting clusters based on labeled data. Basu et al.

[1] Some researchers make a subtle distinction between *semi-supervised* learning, where there are two sets of unlabeled data—unlabeled training data and unlabeled test data, and *transductive* learning, where there is only one set of unlabeled data. In this paper, we take the second view.

[2] Here "centroid" is a general concept and can be a set of parameters for a probabilistic model.

(2002) compared the first two approaches on text documents based on spherical k-means (Dhillon & Modha, 2001). They observed that the constrained version fares at least as well as the seeded version.

On the application side, much existing work focused on document classification or clustering, which has become an increasingly important technique for (un)supervised document organization, automatic topic extraction, and fast information retrieval or filtering. For example, a web search engine often returns thousands of pages in response to a broad query, making it difficult for users to browse or to identify relevant information. Clustering methods can be used to automatically group the retrieved documents into a list of meaningful categories. Similarly, a large database of documents can be pre-classified or pre-clustered to facilitate query processing by searching only the clusters that are close to the query (Jardine & van Rijsbergen, 1971). In this paper, we focus on semi-supervised clustering of text documents for improving the organization of document collections based on a set of labeled documents.

This paper presents deterministic annealing (DA) extensions of the three semi-supervised clustering methods—seeded clustering, constrained clustering, and feedback clustering—under a model-based clustering framework (Zhong & Ghosh, 2003) and compares their performance on real text datasets with multinomial models.[3] Our experimental results show that deterministic annealing (Rose, 1998) can significantly boost the performance of semi-supervised clustering.

Our comparative study also shows that, for multinomial model-based clustering of documents, the constrained approach performs better than the other two when the classes in labeled data are complete, whereas the feedback-based approach is superior when the labeled data contain only a partial set of text categories. To our knowledge, the semi-supervised DA algorithms are new and the comparison between feedback-based approaches and the other two (seeded and constrained) has not been done before.

The organization of this paper is as follows. Section 2 briefly describes multinomial model-based document clustering with deterministic annealing. Section 3 analyzes three enhanced semi-supervised clustering algorithms based on multinomial models and deterministic annealing. Section 4 presents a comparative study and discusses clustering results on several text datasets. Section 5 discusses some related work, followed by concluding remarks in Section 6.

2. Multinomial model-based deterministic annealing

In Zhong and Ghosh (2003), we presented a unified analysis of model-based partitional clustering from a DA point of view. In this section we shall first summarize model-based DA clustering and then describe multinomial models used for clustering documents. Some properties and results of multinomial model-based DA clustering will also be discussed.

2.1. Model-based DA clustering

In model-based clustering, one estimates K models ($\Lambda = \{\lambda_1, \ldots, \lambda_K\}$) from N data objects, with each model representing a cluster. Let X be a random variable characterizing the set of data objects, Y a random variable characterizing the set of cluster indices. Let the joint probability between X and Y be $P(x, y)$. Defining a prior entropy $H(Y)$ as

[3] As we shall see in Section 2, the multinomial model-based clustering is closely related to the recent information-theoretic (Kullback-Leibler or KL) clustering (Dhillon & Guan, 2003) and Information Bottleneck (IB) clustering (Tishby et al., 1999) methods.

$H(Y) = -\sum_y P(y)\log P(y)$ and an average posterior entropy $H(Y \mid X)$ as $H(Y \mid X) = -\sum_x P(x)\sum_y P(y \mid x)\log P(y \mid x)$, we aim to maximize the expected log-likelihood with entropy constraints

$$\begin{aligned} L &= E_{P(x,y)}[\log P(x \mid \lambda_y)] + T \cdot H(Y \mid X) - T \cdot H(Y) \\ &= \sum_x P(x) \sum_y P(y \mid x) \log P(x \mid \lambda_y) - T \cdot I(X;Y)\,, \end{aligned} \tag{1}$$

there be a verb between $I(X;Y)$ and the mutual information? The parameter T is a Lagrange multiplier used to trade off between maximizing the expected log-likelihood $E_{P(x,y)}[\log P(x \mid \lambda_y)]$ and minimizing the mutual information between X and Y (i.e., compressing data X into clusters Y as much as possible). If we fix $H(Y)$, minimizing $I(X;Y)$ is equivalent to maximizing the average posterior entropy $H(Y \mid X)$, or maximizing the randomness of the data assignment. The prior $P(x)$ is usually set to be constant $1/N$. As N goes to infinity, the sample average approaches the expected log-likelihood asymptotically.

Maximizing (1) over $P(y \mid x)$ and λ_y leads to a generic model-based partitional clustering algorithm (Zhong & Ghosh, 2003) that iterates between the following two steps:

$$P(y \mid x) = \frac{P(y)P(x \mid \lambda_y)^{\frac{1}{T}}}{\sum_{y'} P(y')P(x|\lambda_{y'})^{\frac{1}{T}}} = \frac{P(y)P(x \mid \lambda_y)^{\beta}}{\sum_{y'} P(y')P(x \mid \lambda_{y'})^{\beta}}\,, \quad \beta = \frac{1}{T}, \tag{2}$$

and

$$\lambda_y = \arg\max_{\lambda} \sum_x P(y \mid x)\log P(x \mid \lambda_y)\,. \tag{3}$$

If necessary, $P(y)$ can be estimated as $P(y) = \sum_x P(x)P(y \mid x)$. It can be seen that $P(y \mid x)$ is actually dependent on model parameters Λ, thus should be written as $P(y \mid x, \Lambda)$. For simplicity, however, we use $P(y \mid x)$ where there is no confusion. This model-based clustering

Algorithm: model-based DA clustering

Input: A set of N data objects $X = \{x_1, ..., x_N\}$, model structure $\Lambda = \{\lambda_1, ..., \lambda_K\}$, temperature decreasing rate $\alpha, 0 < \alpha < 1$, initial inverse temperature β_0 and final β_f.

Output: Trained models Λ and a partition of the data objects given by the cluster identity vector $Y = \{y_1, ..., y_N\},\ y_n \in \{1, ..., K\}$.

Steps:

1. Initialization: initialize the model parameters Λ and set $\beta = \beta_0$;
2. Optimization: optimize $L = E_{P(x,y)}[\log P(x|\lambda_y)] - T \cdot I(X;Y)$ by iterating between the following two steps until convergence.
   - E-step: $P(y|x) = \frac{P(y)P(x|\lambda_y)^{\beta}}{\sum_{y'} P(y')P(x|\lambda_{y'})^{\beta}}$;
   - M-step: $\lambda_y = \arg\max_{\lambda} \sum_x P(y|x)\log P(x|\lambda_y)$.
3. Annealing: lower temperature by setting $\beta^{(new)} = \beta^{(old)}/\alpha$, go to step 4 if $\beta > \beta_f$, otherwise go back to step 2.
4. For each data object x_n, set $y_n = \arg\max_y P(y|x_n)$.

Fig. 1 Deterministic annealing algorithm for model-based clustering

algorithm is parameterized by the parameter T, which has a temperature interpretation in deterministic annealing (Rose, 1998). In practice, an inverse temperature parameter β is often used. A generic model-based DA clustering algorithm (Fig. 1) can be constructed by gradually decreasing temperature T. At each temperature, the EM algorithm (Dempster et al., 1977) is used to maximize (1), with cluster labels Y being the hidden variable and Eqs. (2) and (3) corresponding to E-step and M-step, respectively.

It can be shown (Zhong & Ghosh, 2003) that setting $T = 0$ leads to a generic k-means clustering algorithm and setting $T = 1$ leads to the standard mixture-of-models EM clustering algorithm (Banfield & Raftery, 1993). In other words, k-means and EM clustering correspond to two different stages in the model-based DA clustering process ($T = 0$ and $T = 1$, respectively).

2.2. Multinomial models

In Zhong and Ghosh (2005), we compared three different probabilistic models for clustering text documents—multivariate Bernoulli, multinomial, and von Mises-Fisher (vMF) (Mardia, 1975). The Bernoulli model was found to be the least suitable model due to its limited binary representation of documents. For regular mixture-of-models clustering, vMF models slightly outperform multinomial models. With the addition of deterministic annealing, however, multinomial models perform comparably with vMF models. Here is why we use multinomial models instead of vMF models underlying the spherical k-means algorithm (Banerjee et al., 2003): Even though the spherical k-means algorithm is simple and efficient, the mixture-of-vMFs, a soft version of spherical k-means, involves Bessel function in its parameterization and requires intensive computation even for approximated parameter estimation (Banerjee et al., 2003). A deterministic annealing extension would be computationally even more complicated and mask the original purpose of this paper. A standard description of multinomial models is available in many statistics or probability books (e.g., Stark & Woods 1994); here we briefly discuss it in the context of clustering text documents.

A traditional vector space representation is used for text documents, i.e., each document is represented as a high-dimensional vector of "word" counts in the document. The "word" here is used in a broad sense since it may represent individual words, stemmed words, tokenized words, or short phrases. The dimensionality of document vectors equals the vocabulary size.

Based on the naïve Bayes assumption, a multinomial model for cluster y represents a document x by a multinomial distribution of the words in the vocabulary $P(x \mid \lambda_y) = \prod_i P_y(i)^{x(i)}$, where $x(i)$ is the i-th dimension of document vector x, indicating the number of occurrences of the i-th word in document x. To accommodate documents of different lengths, we use a normalized (log-)likelihood measure

$$\log \tilde{P}(x \mid \lambda_y) = \frac{1}{|x|} \log P(x \mid \lambda_y) , \tag{4}$$

where $|x| = \sum_i x(i)$ is the length of document x. The $P_y(i)$'s are the multinomial model parameters and represent the word distribution in cluster y. They are subject to the constraint $\sum_i P_y(i) = 1$ and can be estimated by counting the number of documents in each cluster and the number of word occurrences in all documents in the cluster y (Nigam, 2001). With maximum *a posteriori* estimation and Dirichlet prior $P(\lambda_y) = C \cdot \prod_i P_y(i)$, the parameter estimation of multinomial models amounts to

$$P_y(i) = \frac{1 + \sum_x P(y \mid x)x(i)}{\sum_j \left(1 + \sum_x P(y \mid x)x(j)\right)} = \frac{1 + \sum_x P(y \mid x)x(i)}{|V| + \sum_j \sum_x P(y \mid x)x(j)} , \tag{5}$$

where $|V|$ is the size of the word vocabulary. The posterior $P(y \mid x)$ can be estimated from (2).

A connection between multinomial model-based clustering and the divisive Kullback-Leibler clustering (Dhillon et al., 2002; Dhillon & Guan, 2003) is worth mentioning here. It is briefly mentioned in Dhillon and Guan (2003) but they did not explicitly stress that the divisive KL clustering is *equivalent* to multinomial model-based k-means, which maximizes the following objective function:

$$\frac{1}{N}\sum_x \frac{1}{|x|}\log P\big(x|\lambda_{y(x)}\big) = \frac{1}{N}\sum_x \log \tilde{P}\big(x \mid \lambda_{y(x)}\big)\,, \tag{6}$$

where the $\tilde{P}$ notation is defined in (4). This provides another good reason for our choice of multinomial models in this paper.

2.3. Multinomial model-based DA clustering (*damnl*)

Substituting the generic M-step in the model-based DA clustering (Fig. 1) with (5) gives a multinomial model-based DA clustering algorithm, abbreviated as *damnl*. The normalized log-likelihood measure (4) is used since it accommodates different document lengths and leads to a stable annealing process in our experiments.

After some algebraic manipulation, the objective function of *damnl* can be written as

$$L = -\sum_x D_{KL}(P_x \mid P_{y(x)}) - T \cdot I(X;Y) + \sum_x H(P_x)\,, \tag{7}$$

where the last term is a constant. This leads to the following connection to the Information Bottleneck method.

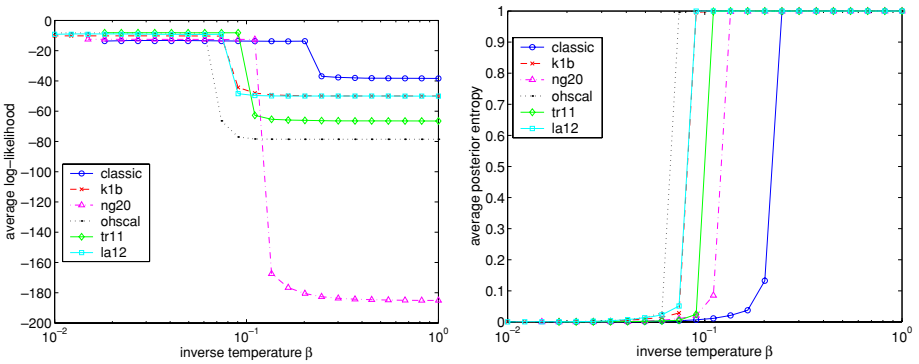
Connection to information bottleneck

In this section we show that, when applied to clustering, the Information Bottleneck method (Tishby et al., 1999) can be seen as a special case of model-based DA clustering with the underlying probabilistic models being multinomial models. This was mentioned by Slonim and Weiss (2003) when they explored the relationship between maximum likelihood formulation and information bottleneck.

The IB method aims to minimize the objective function

$$\begin{aligned} F &= I(X;Y) - \beta I(Z;Y) \\ &= I(X;Y) + \beta(I(Z;X) - I(Z;Y)) - \beta I(Z;X) \\ &= I(X;Y) + \beta E_{P(x,y)}[D_{KL}(P(z \mid x) \mid P(z \mid y))] - \beta I(Z;X) \end{aligned} \tag{8}$$

It trades off between minimizing the mutual information between data X and compressed clusters Y and preserving the mutual information between Y and a third variable Z. Both X and Z are fixed data but Y represents the cluster structure that one tries to find out. The last term in (8) can be treated as a constant w.r.t. to Y and thus the clustering algorithm. It is easy to see that minimizing (8) is equivalent to maximizing (7), with β being the inverse of temperature T and Z being a random variable representing the word dimension. This relationship was also noted in Banerjee et al. (2004).



(a) (b)

Fig. 2 Training curves for, (a) average log-likelihood, and, (b) average (normalized) posterior entropy, for document clustering results using multinomial model-based deterministic annealing on the six datasets described in Table 1

2.4. Annealing results for *damnl*

To see the annealing effect of the *damnl* algorithm, we draw out the training curves (Fig. 2) for both the average (normalized) log-likelihood objective (6) and the average normalized entropy of the posterior distribution

$$\bar{H}_{\text{post}} = \frac{1}{N} \sum_{x} H(P(y \mid x)) / \log K \ .$$

The datasets and experimental setting are explained in Section 4. The X-axis in Fig. 2 shows the inverse temperature parameter β. Each curve is for one text dataset. When β is low, i.e., temperature is high, the average posterior entropy is (close to) 1, which means that the probabilities of a document being assigned to different clusters are (approximately) equal. As β gets large, i.e., temperature decreases to (close to) 0, the average posterior entropy decreases toward 0, which means that the clustering becomes hard and each document goes to only one cluster with high probability. It can be seen that at some stage of the annealing process, the average log-likelihood jumps quickly and the average posterior entropy $\bar{H}_{\text{post}}$ drops to (close to) zero. This stage corresponds to the phase transition point in an annealing process (Rose, 1998). The phase transitions for different datasets are observed to occur at slightly different stages of the annealing process. To achieve good optimization results, one must be careful to choose a temperature schedule that is slow enough not to skip the important phase transition points (Rose, 1998).

3. Semi-supervised model-based DA clustering

In this section, we first present three semi-supervised clustering algorithms under the generic model-based DA clustering framework (Section 2.1) and then discuss the weaknesses and strengths of each approach.

Figure 3 shows the first semi-supervised model-based clustering algorithm—seeded DA clustering. The clustering process differs from regular model-based DA clustering (Fig. 1)

Algorithm: seeded DA clustering

Input: A set of N data objects containing N_l labeled data objects $X^{(l)} = \{x_1, ..., x_{N_l}\}$, N_l labels $Y^{(l)} = \{y_1^{(l)}, ..., y_{N_l}^{(l)}\}$, $y^{(l)} \in S_l \subseteq \{1, ..., K\}$, and N_u unlabeled data objects $X^{(u)} = \{x_{N_l+1}, ..., x_N\}$; model structure $\Lambda = \{\lambda_1, ..., \lambda_K\}$; temperature decreasing rate $\alpha, 0 < \alpha < 1$, and initial inverse temperature β_0 and final β_f.

Output: Trained models Λ and a partition of the data objects given by the cluster identity vector $Y = \{y_1, ..., y_N\}$, $y_n \in \{1, ..., K\}$.

Steps:

1. Initialization: randomly sample N_u labels from $\{1, ..., K\}$ to form $Y^{(u)}$, initialize $Y = Y^{(l)} \cup Y^{(u)}$, train initial model parameters $\lambda_k = \arg\max_{\lambda} \sum_{n:y_n=k} \log P(x_n|\lambda), \forall k \in \{1, ..., K\}$, and set $\beta = \beta_0$;
2. Optimization: optimize (1) by iterating between (2) and (3) until convergence;
3. Annealing: lower temperature by setting $\beta^{(new)} = \beta^{(old)}/\alpha$, go to step 4 if $\beta > \beta_f$, otherwise go back to step 2.
4. For each data object x_n, set $y_n = \arg\max_y \log P(x_n|\lambda_y)$.

Fig. 3 Seeded DA clustering algorithm

Algorithm: constrained DA clustering

Input: A set of N data objects containing N_l labeled data objects $X^{(l)} = \{x_1, ..., x_{N_l}\}$, N_l labels $Y^{(l)} = \{y_1^{(l)}, ..., y_{N_l}^{(l)}\}$, $y^{(l)} \in S_l \subseteq \{1, ..., K\}$, and N_u unlabeled data objects $X^{(u)} = \{x_{N_l+1}, ..., x_N\}$; model structure $\Lambda = \{\lambda_1, ..., \lambda_K\}$; temperature decreasing rate $\alpha, 0 < \alpha < 1$, and initial inverse temperature β_0 and final β_f.

Output: Trained models Λ and a partition of the data objects given by the cluster identity vector $Y = \{y_1, ..., y_N\}$, $y_n \in \{1, ..., K\}$.

Steps:

1. Initialization: randomly sample N_u labels from $\{1, ..., K\}$ to form $Y^{(u)}$, initialize $Y = Y^{(l)} \cup Y^{(u)}$, train initial model parameters $\lambda_k = \arg\max_{\lambda} \sum_{n:y_n=k} \log P(x_n|\lambda), \forall k \in \{1, ..., K\}$, and set $\beta = \beta_0$;
2. Optimization: optimize (1) by iterating between the following two steps until convergence.
   - 2a. For $n \in \{1, ..., N_l\}$, set $P(y|x_n)$ be 1 if $y = y_n^{(l)}$ and 0 otherwise; for $n \in \{N_l + 1, ..., N\}$, update $P(y|x_n)$ according to (2);
   - 2b. Update model parameters according to (3);
3. Annealing: lower temperature by setting $\beta^{(new)} = \beta^{(old)}/\alpha$, go to step 4 if $\beta > \beta_f$, otherwise go back to step 2.
4. For each data object x_n, set $y_n = \arg\max_y \log P(x_n|\lambda_y)$.

Fig. 4 Constrained DA clustering algorithm

only in `Step 1`. The basic idea is to use labeled data to help initialize model parameters: we partition data into a pre-specified number of clusters while keeping data instances with same labels in the labeled data to be in the same cluster.

The second algorithm, constrained DA clustering, is shown in Fig. 4. In addition to using labeled data to help initialization, this algorithm also constrains the assignment of labeled data instances to be hard in the E-step (`Step 2a`)—each labeled instance must stay with its label (initial cluster) with probability 1. This algorithm is basically the semi-supervised EM

Algorithm: feedback DA clustering

Input: A set of N data objects containing N_l labeled data objects $X^{(l)} = \{x_1, ..., x_{N_l}\}$, N_l labels $Y^{(l)} = \{y_1^{(l)}, ..., y_{N_l}^{(l)}\}$, $y^{(l)} \in S_l \subseteq \{1, ..., K\}$, and N_u unlabeled data objects $X^{(u)} = \{x_{N_l+1}, ..., x_N\}$; model structure $\Lambda = \{\lambda_1, ..., \lambda_K\}$; temperature decreasing rate $\alpha, 0 < \alpha < 1$, and initial inverse temperature β_0 and final β_f.

Output: Trained models Λ and a partition of the data objects given by the cluster identity vector $Y = \{y_1, ..., y_N\}$, $y_n \in \{1, ..., K\}$.

Steps:

1. Run the model-based DA clustering algorithm (Figure 1) with random initialization;
2. Match available labels in labeled data with clusters from `Step 1`:
   - 2a. Tag all clusters to be "unoccupied";
   - 2b. For $k \in S_l$, let $X_k^{(l)} = \{x_n^{(l)} | y_n^{(l)} = k\}$. Find cluster j that contains more data instances in $X_k^{(l)}$ than any other cluster;
   - 2c. If cluster j is unoccupied, set $y_n^{(l)} = j, \forall x_n \in X_k^{(l)}$; otherwise increase the number of clusters by 1, $K = K + 1$, and set $y_n^{(l)} = K, \forall x_n \in X_k^{(l)}$;
3. Run the constrained clustering algorithm (e.g., using the algorithm in Figure 4 with a fixed high β) using the Y_l updated in `Step 2`.

Fig. 5 Feedback DA clustering algorithm

algorithm (Nigam et al., 2000; Dong & Bhanu, 2003) wrapped by an annealing process, with the E-step parameterized by a gradually decreasing temperature.

Finally, the feedback DA clustering algorithm is shown in Fig. 5. It is basically the combination of the regular DA clustering algorithm (Fig. 1) and a constrained clustering algorithm, with a feedback step (`Step 2`) in between. The basic idea is to first treat all data as unlabeled and do a regular DA clustering, then take labeled data as feedback to adjust the cluster structure. We caution that there could be several possible heuristic designs. We choose the feedback step used here due to its simplicity since our main purpose is to demonstrate the usefulness of feedback methods, not to find the best feedback strategy. For each class k in the labeled data, we find the corresponding cluster that contains more instances of class k than any other clusters and put all labeled instances of class k into this cluster. In the case when the corresponding cluster is already occupied by another labeled class, we create a new cluster and put all labeled instances of class k into the new cluster. After the adjustment is done, we fix the cluster labels for labeled data instances and run the constrained clustering algorithm (for which we use the algorithm in Fig. 4 with a fixed high β).

The three algorithms presented above are generic and can be used with different models. Plugging multinomial models into the algorithms, we get three semi-supervised multinomial model-based DA clustering algorithms, abbreviated as *Seeded damnl*, *Constrained damnl*, and *Feedback damnl*, respectively. For comparison, we also construct three regular semi-supervised clustering algorithm using mixture-of-multinomials at a fixed low temperature (i.e., high β). While k-means type algorithms (e.g., seeded k-means, etc.) could be constructed, to reuse the code for the above three algorithms, we simply fix β at 100 in the algorithms and get three semi-supervised mixture-of-multinomials algorithms. They are named *Seeded mixmnl*, *Constrained mixmnl*, and *Feedback mixmnl*, respectively. According to the results in Fig. 2, setting $\beta = 100$ makes the posterior data assignment very close to k-means assignment since each data instance goes to its "closest" cluster with high probability.

Table 1 Summary of text datasets (for each dataset, N is the total number of documents, $|V|$ the total number of words, K the number of classes, and N_c the average number of documents per class)

Data	Source	N	$\|V\|$	K	N_c	Balance
NG20	20 Newsgroups	19949	43586	20	997	0.991
classic	CACM/CISI/CRANFIELD/MEDLINE	7094	41681	4	1774	0.323
ohscal	OHSUMED-233445	11162	11465	10	1116	0.437
k1b	WebACE	2340	21839	6	390	0.043
tr11	TREC	414	6429	9	46	0.046
la12	TREC	6279	31472	6	1046	0.289

The *Constrained mixmnl* algorithm is used in `Step 3` of the *Feedback damnl* algorithm (Fig. 5).

Intuitively, the *Seeded damnl* cannot take full advantage of labeled data since the DA process starts the global search at a high temperature where each data instance goes to every cluster with equal probability, which means the initial partition constructed with the help of labeled data may not matter. In fact, one feature of deterministic annealing is its insensitivity to initialization (Rose, 1998). However, starting the seeded approach directly at a low temperature might work since seeding clusters with labeled data can position the local search at a good starting point. The results shown in Basu et al. (2002) and our results in the next section support this analysis. While the DA technique cannot help the seeded approach, we expect that it will benefit the constrained and feedback approaches. Considering the difference between the constrained approach and the feedback approach, we hypothesize that

- The constrained approach should perform better when the available labels are complete and the amount of labeled data is reasonable. Through the DA process, the available labels will guide/bias the search towards "correct" partitioning.
- The feedback approach may be more appropriate when the available labels are incomplete since it starts without the bias of labeled data and is expected to cover all potential labels better than the constrained approach.

4. Experimental results

4.1. Datasets

We used the 20-newsgroups data[4] and several datasets from the CLUTO toolkit[5] (Karypis, 2002). These datasets provide a good representation of different characteristics: the number of documents ranges from 414 to 19949, the number of words from 6429 to 43586, the number of classes from 4 to 20, and balance from 0.046 to 0.998. The *balance* of a dataset is defined as the ratio of the number of documents in the smallest class to the number of documents in the largest class. So a value close to 1 indicates a very balanced dataset and a value close to 0 signifies the opposite. A summary of all the datasets used in this paper is shown in Table 1.

[4] http://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.html.

[5] http://www.cs.umn.edu/~karypis/CLUTO/files/datasets.tar.gz.

The *NG20* dataset is a collection of 20,000 messages, collected from 20 different usenet newsgroups, 1,000 messages from each. We preprocessed the raw dataset using the Bow toolkit (McCallum, 1996), including chopping off headers and removing stop words as well as words that occur in less than three documents. In the resulting dataset, each document is represented by a 43,586-dimensional sparse vector and there are a total of 19,949 documents (after empty documents being removed).

All the datasets associated with the CLUTO toolkit have already been preprocessed (Zhao & Karypis, 2001) in approximately the same way[6] and we further removed those words that appear in two or fewer documents. The *classic* dataset was obtained by combining the CACM, CISI, CRANFIELD, and MEDLINE abstracts that were used in the past to evaluate various information retrieval systems.[7] The *ohscal* dataset was from the OHSUMED collection (Hersh et al., 1994). It contains 11,162 documents from the following ten categories: antibodies, carcinoma, DNA, in-vitro, molecular sequence data, pregnancy, prognosis, receptors, risk factors, and tomography. The *k1b* dataset is from the WebACE project Han et al. (1998). Each document corresponds to a web page listed in the subject hierarchy of Yahoo! (http://www.yahoo.com). The *tr11* dataset is derived from TREC collections (http://trec.nist.gov).

4.2. Evaluation criteria

Clustering evaluation criteria can be based on internal measures or external measures. An internal measure is often the same as the objective function that a clustering algorithm explicitly optimizes, in this paper, the average log-likelihood (6). For document clustering, external measures are more commonly used, since typically the benchmark documents' category labels are known (but of course not used in the clustering process). Examples of external measures include the confusion matrix, classification accuracy, F1 measure, average purity, average entropy, and mutual information (Ghosh, 2003).

In the simplest scenario where the number of clusters equals the number of categories and their one-to-one correspondence can be established, any of these external measures can be fruitfully applied. However, when the number of clusters differs from the number of original classes, the confusion matrix is hard to read and the accuracy difficult or impossible to calculate. It has been argued that the mutual information $I(Y;\hat{Y})$ between a random variable Y, governing the cluster labels, and a random variable $\hat{Y}$, governing the class labels, is a superior measure compared to purity or entropy (Strehl & Ghosh, 2002; Dom, 2001). Moreover, when normalized to lie in the range [0,1], this measure becomes relatively impartial to K. There are several choices for normalization based on the entropies $H(Y)$ and $H(\hat{Y})$. We shall follow the definition of normalized mutual information (NMI) using geometrical mean, $NMI = \frac{I(Y;\hat{Y})}{\sqrt{H(Y)\cdot H(\hat{Y})}}$, as given in Strehl and Ghosh (2002). In practice, we use a sample estimate

$$NMI = \frac{\sum_{h,l} n_{h,l} \log\left(\frac{n \cdot n_{h,l}}{n_h n_l}\right)}{\sqrt{\left(\sum_h n_h \log\frac{n_h}{n}\right)\left(\sum_l n_l \log\frac{n_l}{n}\right)}}, \tag{9}$$

where n_h is the number of documents in class h, n_l the number of documents in cluster l and $n_{h,l}$ the number of documents in class h as well as in cluster l. The NMI value is 1 when

[6] That is, chopping off headers and removing stop words, but with a different software toolkit.

[7] Available from ftp://ftp.cs.cornell.edu/pub/smart.

Table 2 Running time results for four different initialization schemes on six datasets. d is the data dimensionality (i.e., number of terms) and N_{nz} the number of nonzeros in a term-document matrix

	Average running time (seconds)			
	Random	Perturb	Marginal	KKZ
tr11	0	0.018	0.485	0.359
k1b	0	0.047	1.033	0.438
ohscal	0	0.049	0.91	0.998
ng20	0	0.296	6.484	22.25
classic	0	0.048	1.244	0.248
la12	0	0.078	0.482	0.793
Complexity	$O(N)$	$O(N_{nz} + Kd)$		$O(KN_{nz})$

clustering results perfectly match the external category labels and close to 0 for a random partitioning. This evaluation criterion is used in our experiments.

4.3. Cluster initialization

After surveying a range of literature (Katsavounidis et al., 1994; Meila & Heckerman, 2001; He et al., 2004), we experimented with the following four initialization techniques:

- *Random*: we randomly partition data into K groups.
- *Perturb*: We first obtain a (global) multinomial model from all data (as if all data instances were generated from a single model), and then randomly perturb parameters of the model to get K initial models. This methods has been shown to perform well for spherical k-means (Dhillon & Modha, 2001).
- *Marginal*: This scheme was studied in Meila & Heckerman (2001) and showed slight advantage to random initialization. It amounts to sampling from a Dirichlet prior distribution whose parameters are determined by the global multinomial model (same as the one used in the *Perturb* scheme).
- *KKZ*: This scheme is adapted from the initialization method used in Katsavounidis et al. (1994), which has been shown to be one of the best initialization methods for k-means clustering (He et al., 2004). Our modified version works as follows: We initialize the first cluster model using the document that is most distant to the global multinomial model. For each subsequent model, we use the document vector that is most distant to its closest existing cluster. The distance to a cluster is measured by KL-divergence. This method is deterministic compared to the previous three.

We compared the four initialization schemes for *mixmnl* and *damnl* algorithms[8] and found that the best scheme is *KKZ* for *mixmnl* and *Random* for *damnl*. Most of the time, the *Perturb* and *Marginal* schemes are no better than the *Random* approach. Table 2 shows the running time for different initialization schemes. The last row is the theoretical computational complexity.

[8] Results on six datasets are not shown here for simplicity. Another reason is that here we are more interested in the results for semi-supervised algorithms.

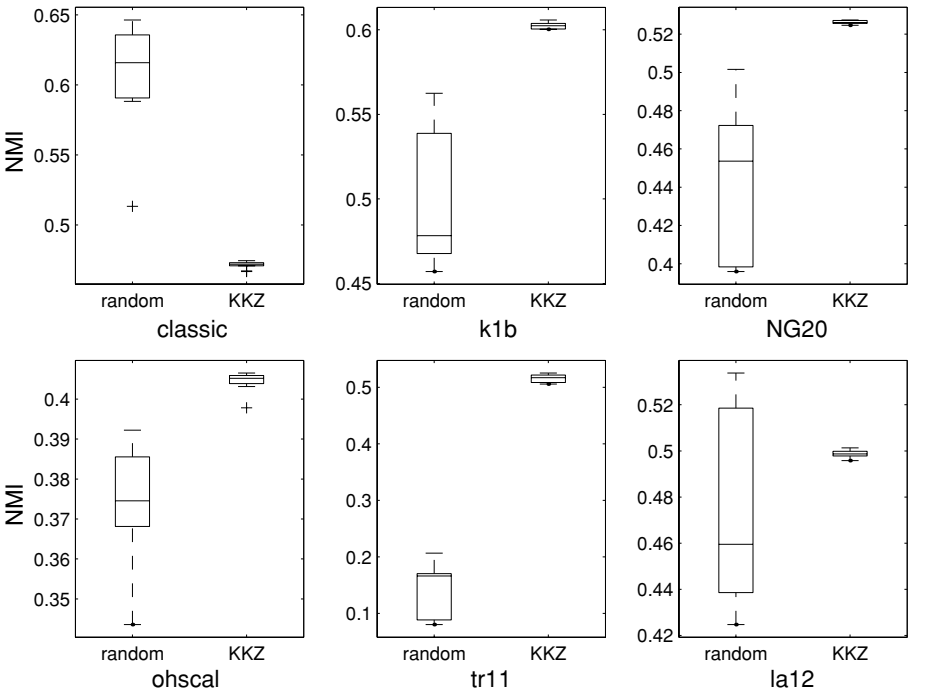


Fig. 6 Comparing NMI results for two initialization schemes (*Random* and *KKZ*) for *feedback mixmnl* on six datasets

Here we are more interested in the effect of initialization on the presented semi-supervised algorithms. Note that the seeded and constrained algorithms do not need any of these four initialization schemes since they use labeled data to initialize clusters. Figure 6 shows the comparison between *Random* and *KKZ* for the *feedback mixmnl* algorithm. The boxplot results are drawn for the distribution of NMI values over 10 runs.[9] Outliers are data with values beyond the ends of the whiskers. The *KKZ* method has a clear advantage over *Random* on five out of six datasets. Figure 7 shows that *Random* is not worse than *KKZ* for the *feedback damnl* algorithm. Based on these results, we decided to use *Random* for all other algorithms but *KKZ* for *feedback mixmnl*.

4.4. Experimental setup

For the *damnl* and semi-supervised *damnl* algorithms, an exponential schedule is used for the inverse temperature parameter β with an initial value of 0.5 and final value of 200. From one iteration to the next, $\beta(m+1) = 1.3\beta(m)$, where m is the iteration number. As seen from the results in Fig. 2, this setting for β is reasonable for the DA process to capture phase transition point(s) on all datasets used in our experiments. For semi-supervised *mixmnl* algorithms, β is set at 100, as discussed in the previous section.

[9] In the plots, the box has lines at the lower quartile (25%), median (50%), and upper quartile (75%) values. The whiskers are lines extending from each end of the box to show the extent of the rest of the data (5% and 95%).

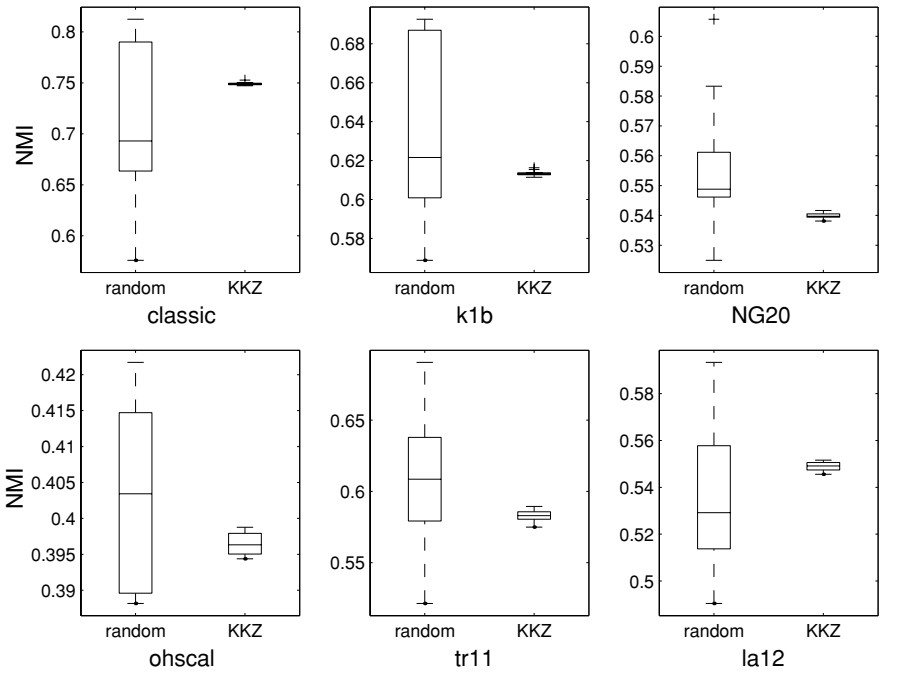


Fig. 7 Comparing NMI results for two initialization schemes (*Random* and *KKZ*) for *feedback damnl* on six datasets

We consider the following two different scenarios when selecting a subset of the documents to be labeled data:

- `Complete labels`—all document classes are randomly sampled. We construct a series of training datasets by randomly sampling (without replacement) 2.5%, 5%, 7.5%, 10%, 20%, ..., and 90% of all documents as the labeled set and the rest as the unlabeled set. The sampling is not stratified, i.e., the percentage of documents in the labeled set may be different for each class.
- `Incomplete labels`—some classes are not available in the labeled set. We pick documents from only half of all classes. We first randomly decide the half of all classes and then sample (not-stratified, without replacement) 2.5%, 5%, 7.5%, 10%, 20%, ..., and 90% of the documents in the selected (half of all) classes as the labeled set.

For each algorithm and each percentage setting, we repeat the random sampling process ten times and report the average and standard deviation of NMI values for clustering results. For better readability, we only show NMI results up to 60% in the figures in the next section.

4.5. Results analysis

`Complete labels` *scenario*

Figures 9–11 shows the NMI results for the `complete labels` scenario. The first three figures compare semi-supervised *mixmnl* and semi-supervised *damnl* algorithms for the

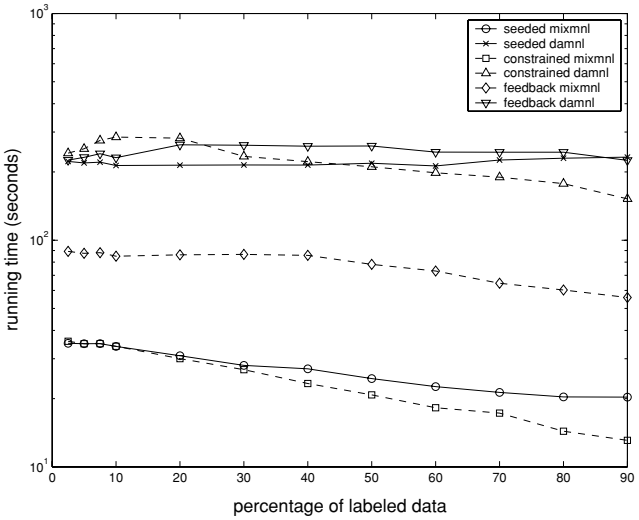


Fig. 8 Running time results for *ng20* dataset

seeded approach, the constrained approach, and the feedback approach, respectively. The results for the *Seeded damnl* algorithm are relatively flat across different amount of labeled data, confirming previous discussion on the weak effect of labeled data on seeded DA clustering. In contrast, a clearer upward trend can be seen for the *Seeded mixmnl* algorithm as the fraction of labeled data increases. The *Seeded mixmnl* algorithm also performs significantly better than *Seeded damnl* in most situations except for the *tr11* dataset and when the fraction of labeled data is low. The reason for the exception on *tr11* dataset may be that the *tr11* dataset is small and unbalanced and thus the amount of labeled data for some classes may be too small or even near empty, causing the *Seeded mixmnl* algorithm to get stuck in a bad local solution quickly. When the fraction of labeled data is small, the initial models are less accurate, causing *Seeded mixmnl* to converge to a worse solution than *Seeded damnl*.

Both constrained approaches and feedback approaches show clear benefits of having more labeled data: the NMI values increase as the percentage of labeled instances grows, as shown in Figs. 10 and 11. It is also evident that the DA versions perform at least as well as the *mixmnl* versions and sometimes significantly better. The only exception is for the constrained approach on the *classic* dataset and when the percentage of labeled data is 10% or lower, where *Constrained damnl* performs significantly worse than *Constrained mixmnl*. Upon more examination, we find that the DA version reaches better average log-likelihoods even as it gives lower NMI values for this case. We suspect that the DA process finds a better local maxima of the objective function but the resulting partition does not conform to the given labels. This will be further investigated in our future work.

We compare the three semi-supervised approaches together in Fig. 12. For the seeded approach, both *Seeded mixmnl* and *Seeded damnl* are selected; For the other two approaches, *Constrained damnl* and *Feedback damnl* are selected since they outperform their *mixmnl* counterparts. The *Constrained damnl* algorithm is obviously the winner, producing significantly higher NMI values in most cases. The only exception is on the *classic* dataset with small

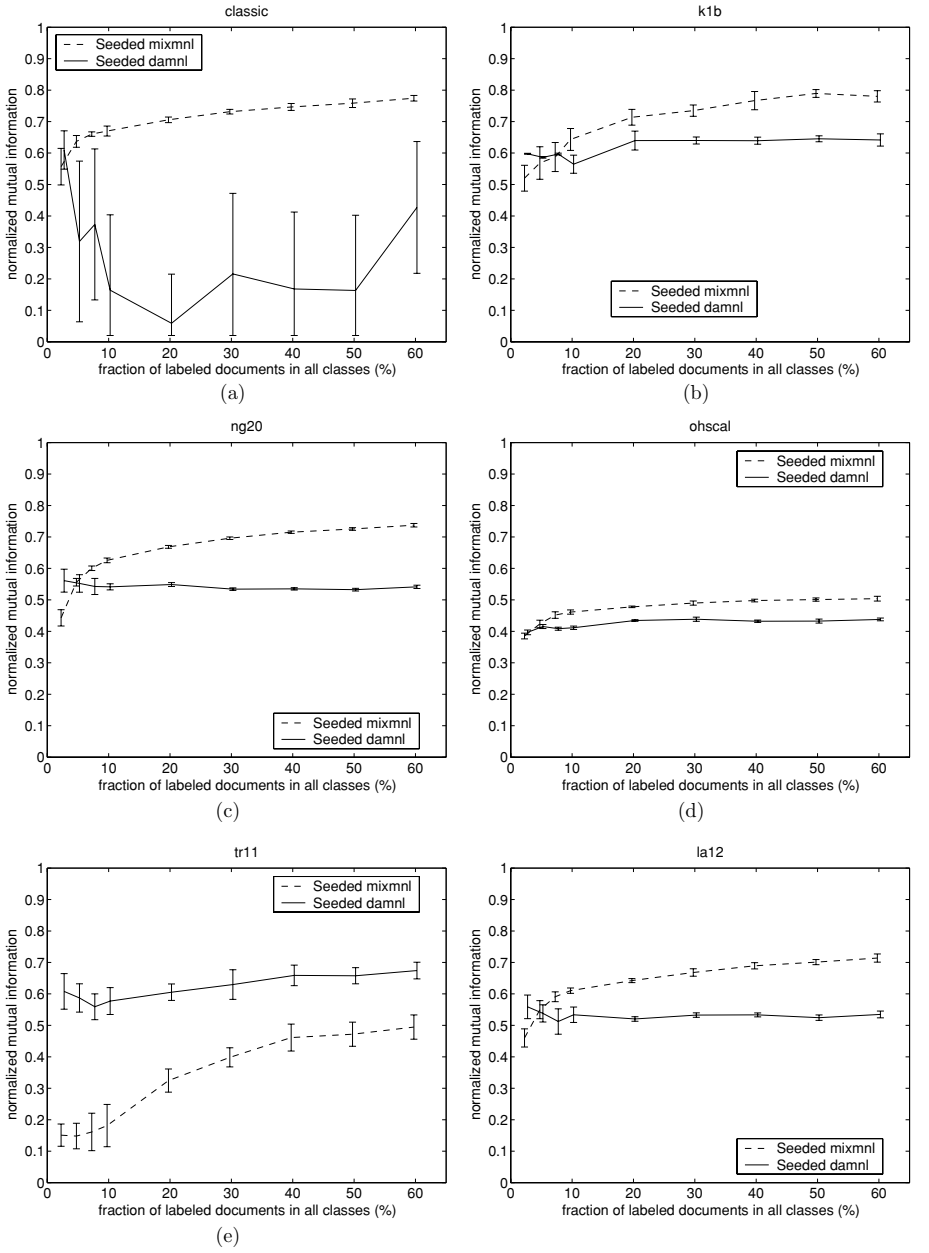


Fig. 9 Comparing NMI results for *Seeded damnl* and Seeded mixmnl algorithms on the six datasets in Table 1

amount of labeled data, as discussed above. *Feedback damnl* is overall better than *Seeded mixmnl*, with superior performance on three datasets (*classic*, *k1b*, and *ng20*) and mixed results on the remaining three datasets (*ohscal*, *tr11*, and *la12*). Except on *tr11* dataset, *Seeded mixmnl* is the second best algorithm when the fraction of labeled data is 5% or higher.

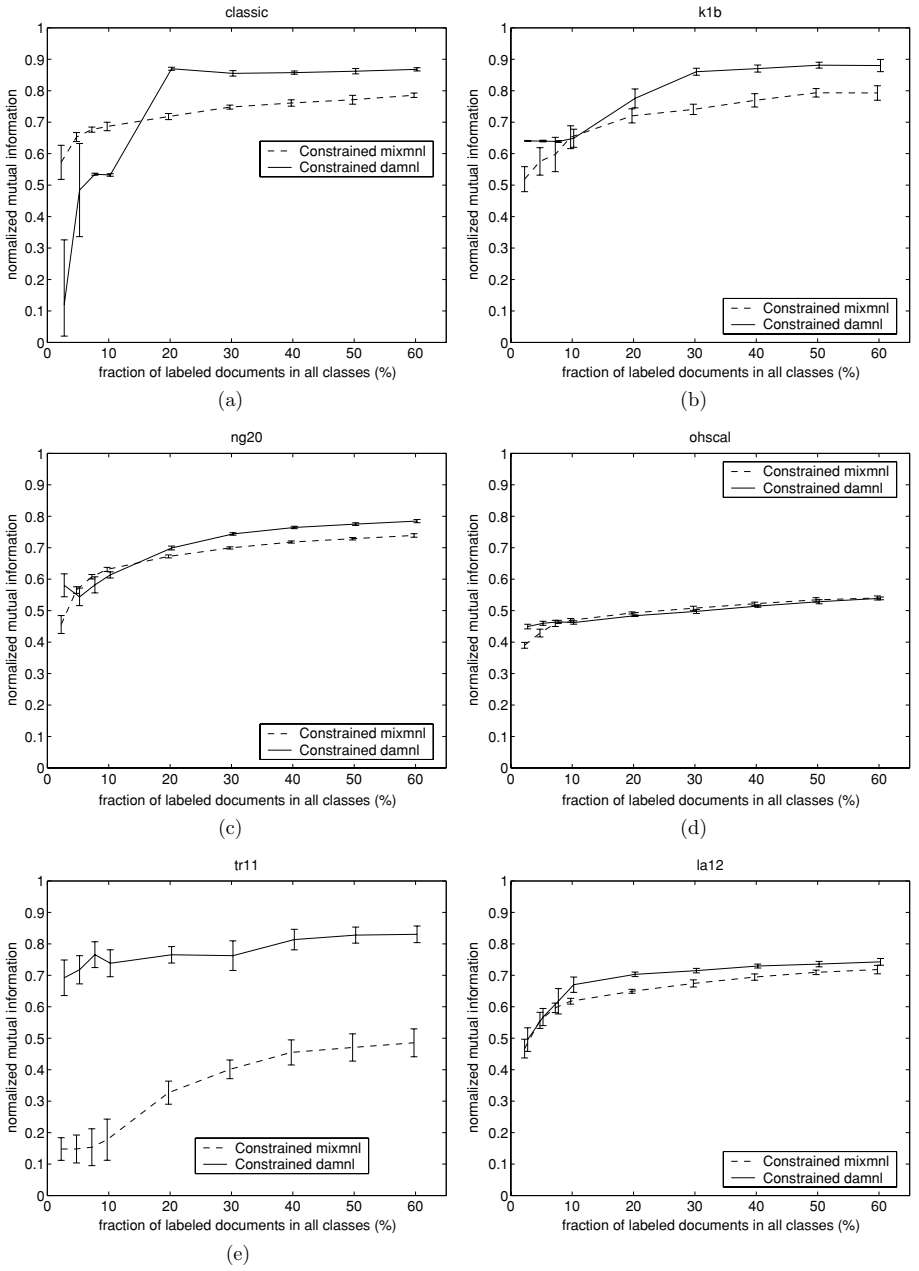


Fig. 10 Comparing NMI results for Constrained damnl and Constrained mixmnl on the six datasets in Table 1

`Incomplete labels` *scenario*

Figures 13–16 show the NMI results for the `incomplete labels` scenario. There is generally no strong upward trend except on *ng20* and *tr11* dataset and mainly for mixmnl

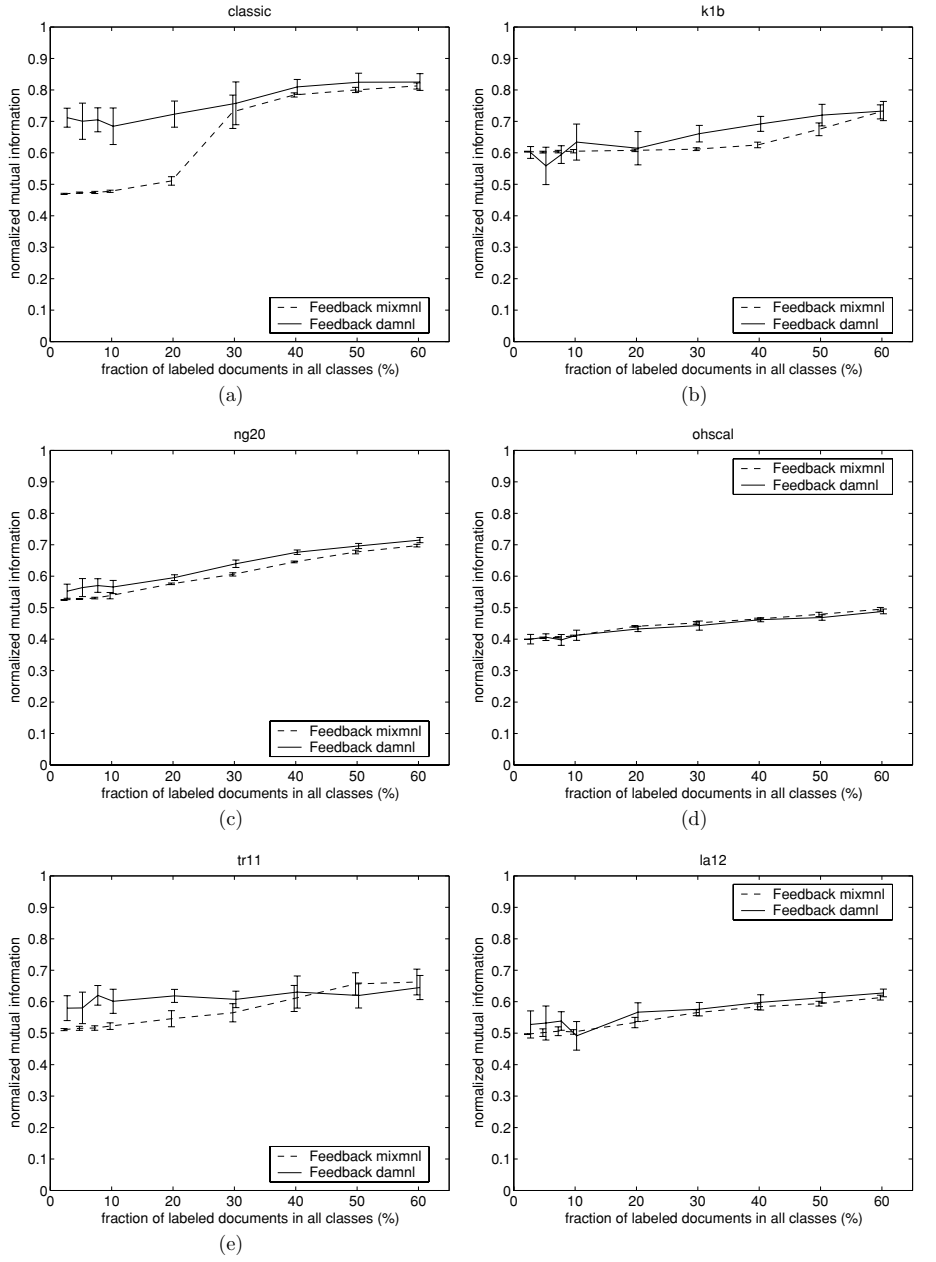


Fig. 11 Comparing NMI results for Feedback damnl and Feedback mixmnl algorithms on the six datasets in Table 1

approaches; Most other curves are relatively flat. This indicate that the benefit of labeled data in the `incomplete labels` scenario is small.

The *Seeded mixmnl* algorithm outperforms *Seeded damnl* only on the *classic* dataset. When the fraction of labeled data is less than 10%, *Seeded damnl* is always comparable to

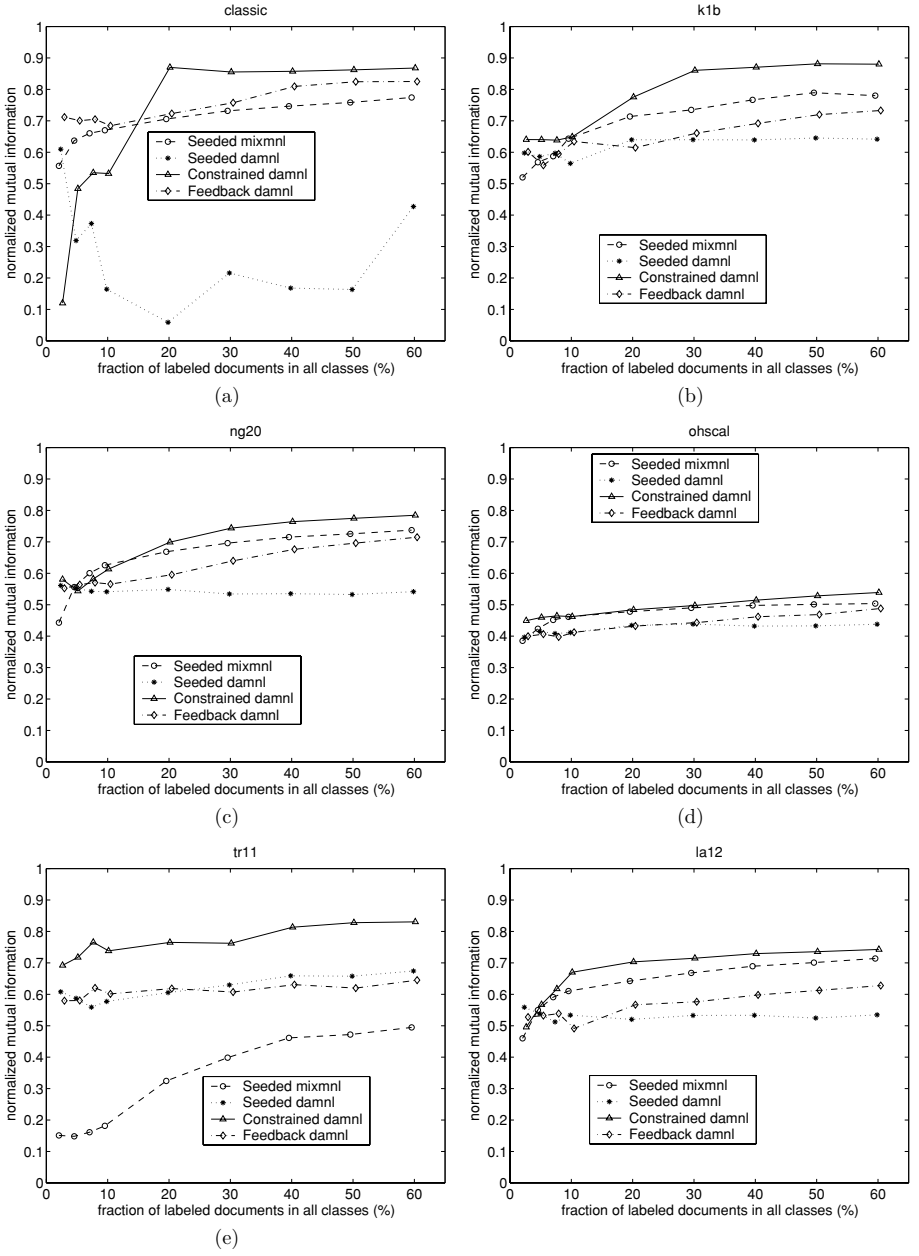


Fig. 12 Comparing NMI results for Seeded mixmnl, Constrained damnl, and Feedback damnl algorithms on the six datasets in Table 1

or significantly better than *Seeded mixmnl*. This observation matches the one seen in the `complete labels` scenario.

Mixed results are observed when comparing *Constrained damnl* with *Constrained mixmnl*—the former wins on *ng20* and *tr11* datasets but loses on the others (Fig. 14). For feed-

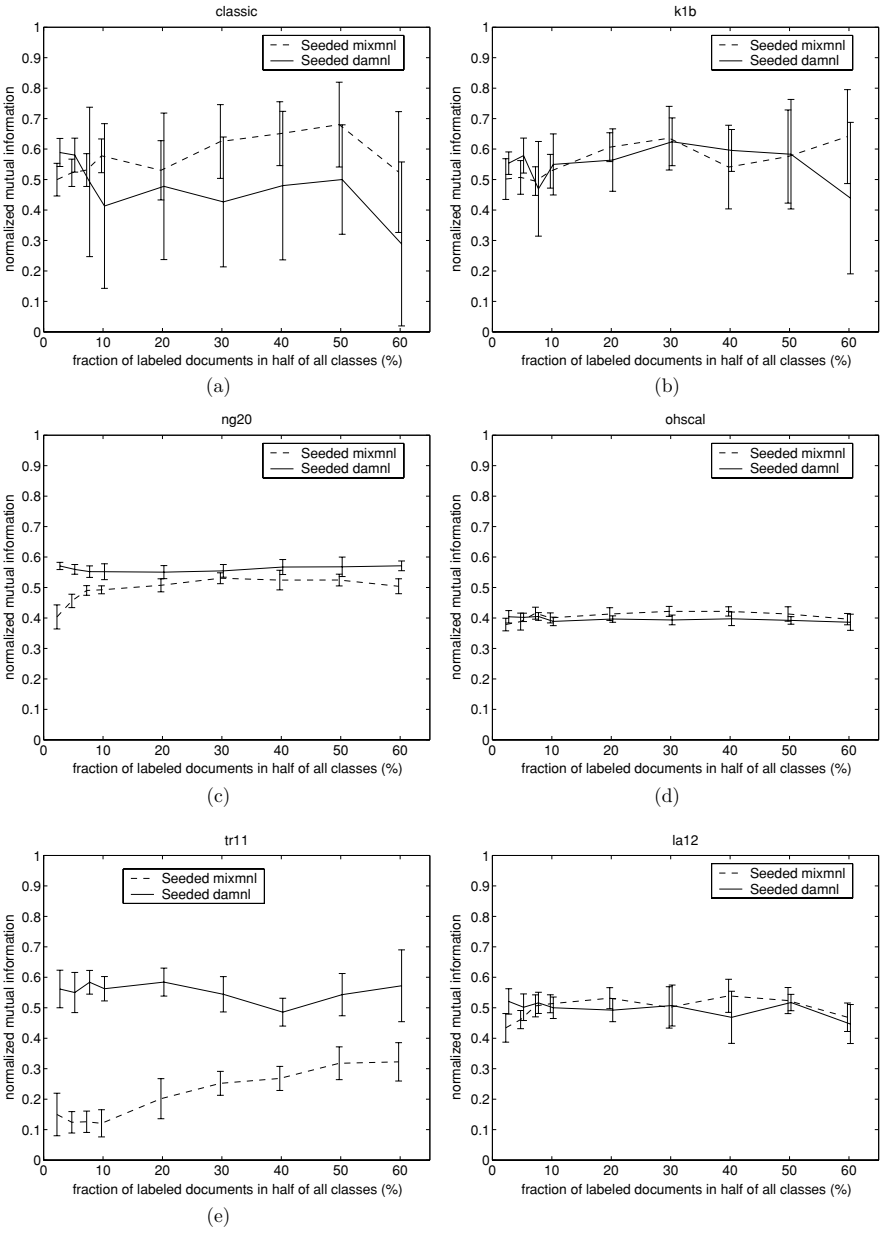


Fig. 13 Comparing NMI results for Seeded damnl and Seeded mixmnl on the six datasets in Table 1 (`incomplete labels` scenario)

back approaches, *Feedback damnl* fares at least as well as *Feedback mixmnl* and significantly better on most datasets (*classic*, *ng20*, *tr11*, and *la12*), as shown in Fig. 15.

The three semi-supervised approaches are compared in Fig. 16. The *Constrained mixmnl* seems to deliver very similar performance to the seeded counterpart. This can be seen in Fig. 16, where the curves for *Seeded mixmnl* and *Constrained mixmnl* almost completely

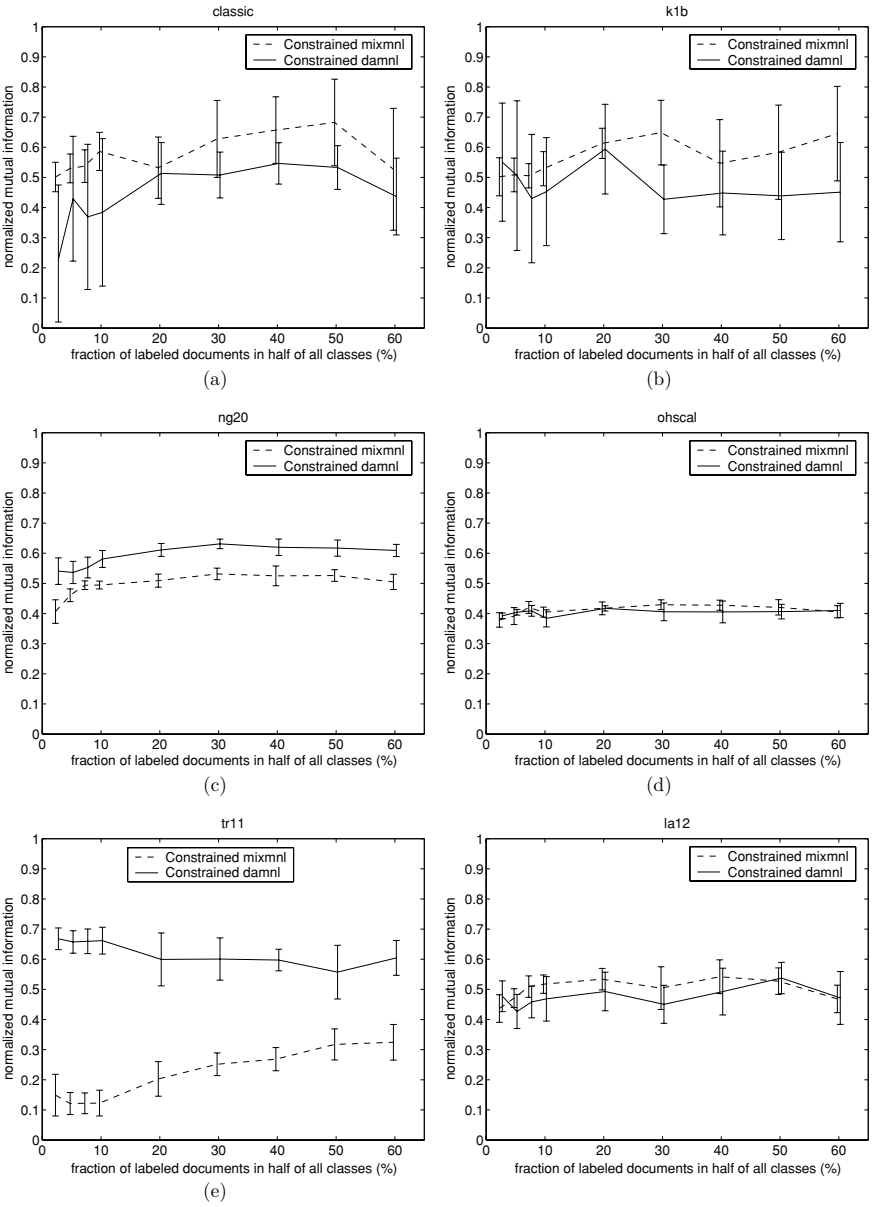


Fig. 14 Comparing NMI results for *Constrained damnl* and *Constrained mixmnl* on the six datasets in Table 1 (`incomplete labels` scenario)

overlap. For the constrained approach, both *Constrained mixmnl* and *Constrained damnl* are selected because of the mixed results in Fig. 14; for the feedback approach, *Feedback damnl* is selected due to better performance than its *mixmnl* counterpart. Overall, *Feedback damnl* is the best algorithm and the only one that has consistent superior performance across all six

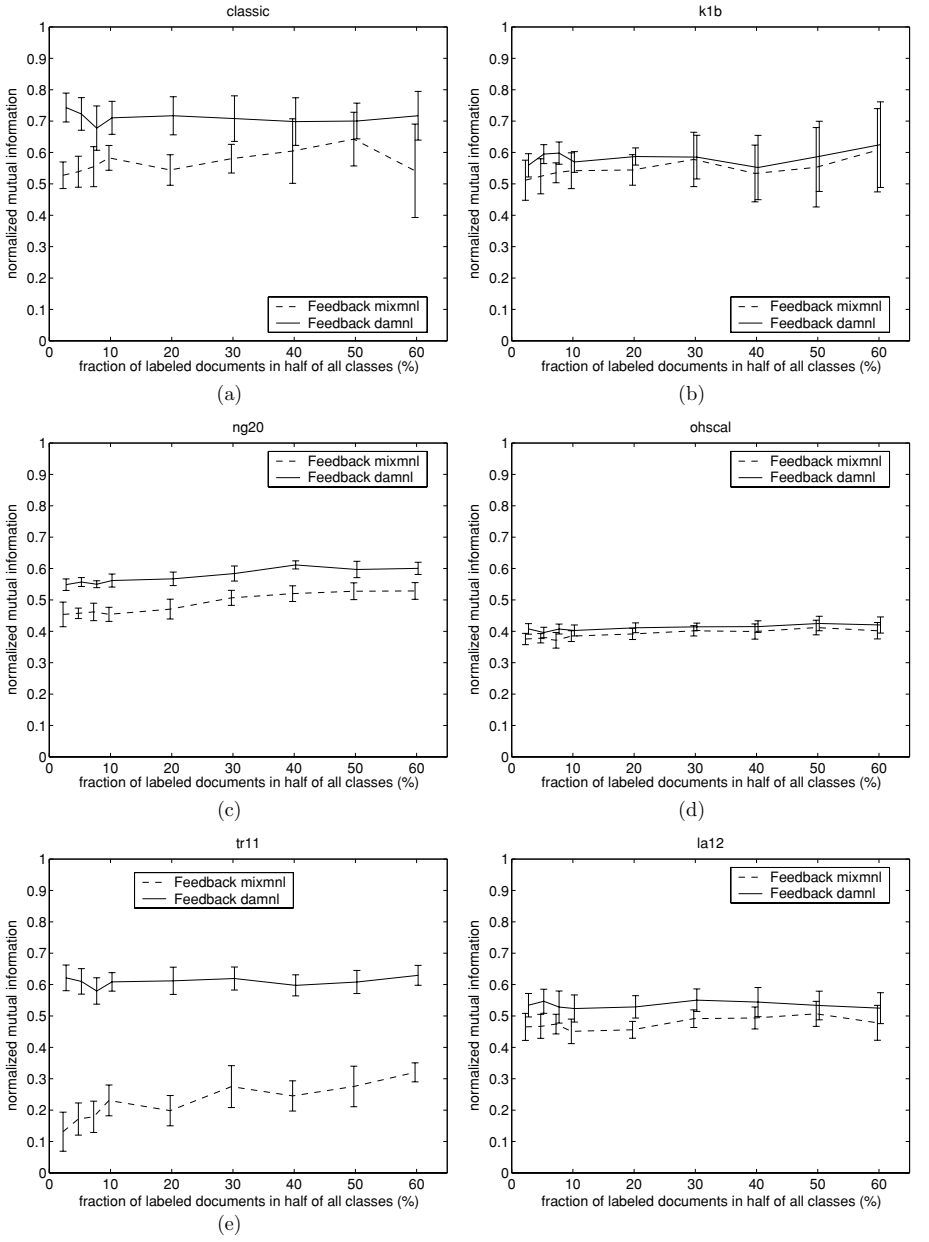


Fig. 15 Comparing NMI results for Feedback damnl and Feedback mixmnl algorithms on the six datasets in Table 1 (`incomplete labels` scenario)

datasets. The other algorithms usually perform comparably with *Feedback damnl* on two or three (out of five) datasets but significantly worse on others.

In summary, the experimental results match favorably with the hypotheses discussed in Section 3 and encourage us to further explore feedback-type approaches in real-world adaptive environments.

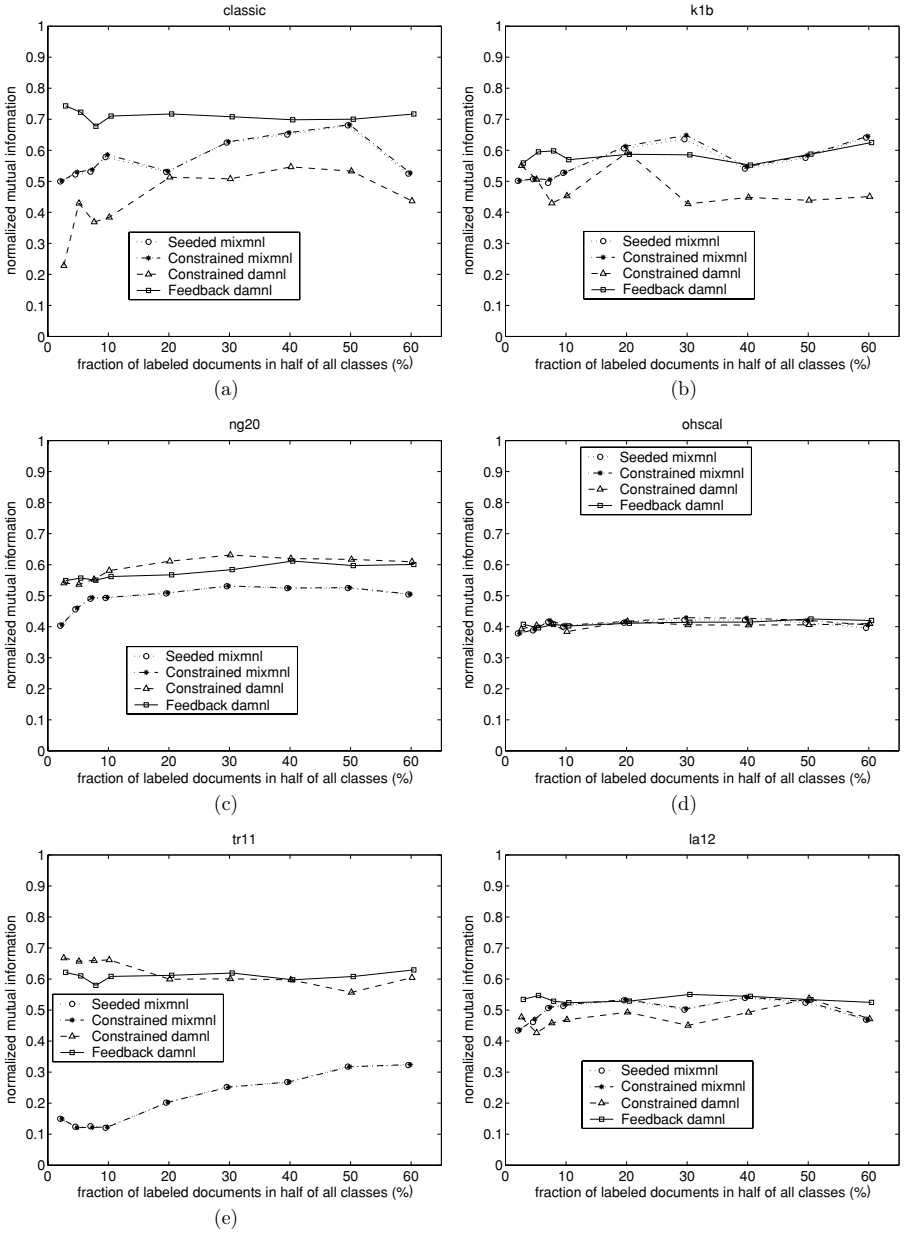


Fig. 16 Comparing NMI results for Seeded mixmnl, Constrained mixmnl, Constrained damnl, and Feedback damnl algorithms on the six datasets in Table 1 (`incomplete labels` scenario)

Time complexity

Some time results are given in Fig. 8 for the largest dataset *ng20*. They are measured on a 2.4 GHz Pentium-4 PC running Windows XP, with the algorithms written in Matlab. The *damnl* algorithms are about four to ten times slower than the *mixmnl* ones. This could be

improved, for example, by using fewer iterations at each temperature. Such improvements will be investigated in the future.

While the time complexities of all *mixmnl* algorithms go down as the fraction of labeled data grows, *Constrained damnl* is the only one in the *damnl* category with this trend. This is not surprising since the *mixmnl* algorithms will likely converge faster for a higher fraction of labeled data. On the other hand, the *Seeded damnl* and *Feedback damnl* use labeled information only at the beginning or the end of a *damnl* process and thus cannot take advantage of faster *mixmnl* algorithms in the whole DA process.

5. Related work

The constrained approach studied in this paper is most related to the semi-supervised EM algorithm presented in Nigam et al. (2000), where the classes of unlabeled data are treated as missing data and labeled data used as E-step constraints in the EM algorithm. Nigam et al. (2000) presented the algorithm, however, in a semi-supervised classification setting and assumed that the class categories in labeled data are complete. In contrast, Dong and Bhanu (2003) employed the semi-supervised EM algorithm in a clustering setting and extended it to include the cannot-be-in-certain-cluster constraints for image retrieval applications.

Wagstaff et al. (2001) proposed a constrained k-means algorithm to integrate background knowledge into the clustering process. They considered two types of constraints: *must-link*—two data instances must be in the same cluster, and *cannot-link*—two data instances cannot be in the same cluster. These constraints are different from labeled data constraints and more similar to the constraints used in Dong and Bhanu (2003). However, the constrained k-means algorithm is heuristic and the EM approach adopted by Dong and Bhanu (2003) seems to be a more principled approach.

The *must-link* and *cannot-link* types of knowledge can also be used as feedback in an interactive clustering process (Cohn et al., 2003). Our feedback-based approach is different in the type of feedback—we use the class labels in labeled data, instead of *must-link* and *cannot-link* pairs, as feedback constraints. Also, unlike our feedback-based approaches, Cohn et al. (2003) employed feedback information to adjust distance metrics used for clustering. Similar work on semi-supervised clustering through learning distance metrics appeared in Chang and Yeung (2004) and in Basu et al. (2004). These methods mainly addressed the pairwise constraint-type feedback, which is sometimes argued to be a more realistic assumption than available cluster labels. Although we focused on labels, our feedback-based algorithm (Fig. 11) can be adapted to take into account pairwise constraints. This extension will be investigated in our future work.

Basu et al. (2002) compared seeded spherical k-means and constrained spherical k-means for clustering documents and showed that the constrained version performs better. Our results supported the same conclusion, but for multinomial models and using deterministic annealing extensions.

Even in a semi-supervised classification setting, clustering has been used to help increase the amount of labeled data. For example, Zeng et al. (2003) showed that the clustering-based strategy excels when the amount of labeled data is very small, according to experimental results on text classification.

6. Conclusion

We have presented deterministic annealing extensions of three semi-supervised clustering approaches based on a model-based DA clustering framework, with multinomial distribu-

tions representing document clusters. Experimental results on several text datasets show that: (a) The annealing process can often significantly improve semi-supervised clustering results; (b) the constrained approach is superior when the available labels are complete while the feedback-based approach should be selected if the labels are incomplete. In real world applications (e.g., dynamic web page clustering, gene expression clustering, and intrusion detection), where new concepts/classes are likely not covered in a small set of labeled data, we expect the feedback-based approach to be a better choice.

We are aware that the specific seeded and feedback-based algorithms used in our experiments are heuristic and just one of many possible designs. They can be improved but we doubt that it will significantly change the conclusion drawn in this paper. In the future, we plan to incorporate pairwise constraints (Wagstaff et al., 2001; Basu et al., 2002) into our feedback-based approaches.

In our experiments, all algorithms are batch methods. That is, model parameters are updated once for each iteration of going through the whole dataset. Online parameter update (on visiting each individual data instance) can be employed to improve performance and is useful in a data stream environment (Zhong, 2005). For example, Nigam and Ghani (2000) reported that incremental approaches work better than iterative (batch) approaches. Developing an incremental version of the algorithms studied in this paper seems to be a viable future direction.

Another interesting (and natural) direction is to automate the finding of new classes in unlabeled data. Two existing methods are likely relevant—multi-clustering (Friedman et al., 2001) and conditional information bottleneck (Gondek & Hofmann, 2003). The former extends the information bottleneck method to generate two or more clusterings of the same data that are as orthogonal as possible. The multi-clustering idea may be used in the semi-supervised setting such that one clustering is supervised by labeled data and others seek for new class labels. The conditional information bottleneck targets directly at finding clusters that have low mutual information with existing labels, thus it may be suitable for discovering new classes.

Acknowledgments We thank David DeMaris and anonymous reviewers for helpful comments.

References

Banerjee, A., Dhillon, I., Ghosh, J., & Merugu, S. (2004). An information theoretic analysis of maximum likelihood mixture estimation for exponential families. In *Proc. 21st Int. Conf. Machine Learning* (pp. 57–64). Banff, Canada.

Banerjee, A., Dhillon, I., Sra, S., & Ghosh, J. (2003). Generative model-based clustering of directional data. In *Proc. 9th ACM SIGKDD Int. Conf. Knowledge Discovery Data Mining* (pp. 19–28).

Banfield, J. D., & Raftery, A. E. (1993). Model-based Gaussian and non-Gaussian clustering. *Biometrics*, *49*, 803–821.

Basu, S., Banerjee, A., & Mooney, R. (2002). Semi-supervised clustering by seeding. In *Proc. 19th Int. Conf. Machine Learning* (pp. 19–26). Sydney, Australia.

Basu, S., Bilenko, M., & Mooney, R. J. (2004). In A probabilistic framework for semi-supervised clustering. *Proc. 10th ACM SIGKDD Int. Conf. Knowledge Discovery Data Mining* pp. 59–68. Seattle, WA.

Blum, A., & Chawla, S. (2001). In Learning from labeled and unlabeled data using graph mincuts. *Proc. 18th Int. Conf. Machine Learning* (pp. 19–26).

Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. *The 11th Annual Conf. Computational Learning Theory* (pp. 92–100).

Castelli, V., & Cover, T. M. (1996). The relative value of labeled and unlabeled samples in pattern recognition with an unknown mixing parameter. *IEEE Trans. Inform. Theory*, *42*, 2102–2117.

Chang, H., & Yeung, D.-Y. (2004). In Locally linear metric adaptation for semi-supervised clustering. *Proc. 21st Int. Conf. Machine Learning* (pp. 153–160). Banff, Canada.

Cohn, D., Caruana, R., & McCallum, A. (2003). *Semi-supervised clustering with user feedback* (Technical Report TR2003-1892). Cornell University.

Cozman, F. G., Cohen, I., & Cirelo, M. C. (2003). Semi-supervised learning of mixture models. In *Proc. 20th Int. Conf. Machine Learning* (pp. 106–113).

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum-likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, *39*, 1–38.

Dhillon, I., & Guan, Y. (2003). In Information theoretic clustering of sparse co-occurrence data. *Proc. IEEE Int. Conf. Data Mining* (pp. 517–520). Melbourne, FL.

Dhillon, I. S., Mallela, S., & Kumar, R. (2002). In Enhanced word clustering for hierarchical text classification. *Proc. 8th ACM SIGKDD Int. Conf. Knowledge Discovery Data Mining* (pp. 446–455).

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

Dom, B. (2001). *An information-theoretic external cluster-validity measure* (Technical Report RJ10219). IBM.

Dong, A., & Bhanu, B. (2003). In A new semi-supervised em algorithm for image retrieval. *Proc. IEEE Int. Conf. Computer Vision Pattern Recognition* (pp. 662–667). Madison, MI.

Friedman, N., Mosenzon, O., Slonim, N., & Tishby, N. (2001). Multivariate information bottleneck. In *Proc. 17th Conf. Uncertainty in Artificial Intelligence*.

Ghosh, J. (2003). Scalable clustering. In N. Ye (Ed.), *Handbook of data mining* (pp. 341–364). Le Erlbaum Assoc.

Gondek, D., & Hofmann, T. (2003). Conditional information bottleneck clustering. *IEEE Int. Conf. Data Mining Workshop on Clustering Large Datasets* (pp. 36–42). Melbourne, FL.

Guerrero-Curieses, A., & Cid-Sueiro, J. (2000). An entropy minimization principle for semi-supervised terrain classification. In *Proc. IEEE Int. Conf. Image Processing* (pp. 312–315).

Han, E. H., Boley, D., Gini, M., Gross, R., Hastings, K., Karypis, G., Kumar, V., Mobasher, B., & Moore, J. (1998). In WebACE: A web agent for document categorization and exploration. *Proc. 2nd Int. Conf. Autonomous Agents* (pp. 408–415).

He, J., Lan, M., Tan, C.-L., Sung, S. -Y., & Low, H.-B. (2004). Initialization of cluster refinement algorithms: A review and comparative study. In *Proc. IEEE Int. Joint Conf. Neural Networks* (pp. 297–302).

Hersh, W., Buckley, C., Leone, T. J., & Hickam, D. (1994). OHSUMED: An interactive retrieval evaluation and new large test collection for research. In *Proc. ACM SIGIR* (pp. 192–201).

Jardine, N., & van Rijsbergen, C. J. (1971). The use of hierarchical clustering in information retrieval. *Information Storage and Retrieval*, *7*, 217–240.

Joachims, T. (1999). Transductive inference for text classification using support vector machines. In *Proc. 16th Int. Conf. Machine Learning* (pp. 200–209).

Karypis, G. (2002). *CLUTO—a clustering toolkit*. Dept. of Computer Science, University of Minnesota. `http://www-users.cs.umn.edu/~karypis/cluto/`.

Katsavounidis, I., Kuo, C., & Zhang, Z. (1994). A new initialization technique for generalized Lloyd iteration. *IEEE Signal Processing Letters*, *1*, 144–146.

Mardia, K. V. (1975). Statistics of directional data. *J. Royal Statistical Society. Series B (Methodological)*, *37*, 349–393.

McCallum, A. K. (1996). Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. Available at `http://www.cs.cmu.edu/~mccallum/bow`.

Meila, M., & Heckerman, D. (2001). An experimental comparison of model-based clustering methods. *Machine Learning*, *42*, 9–29.

Nigam, K. (2001). *Using unlabeled data to improve text classification*. Doctoral dissertation, School of Computer Science, Carnegie Mellon University.

Nigam, K., & Ghani, R. (2000). Understanding the behavior of co-training. *KDD Workshop on Text Mining*.

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

Rose, K. (1998). Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. In *Proc. IEEE*, *86*, 2210–2239.

Seeger, M. (2001). *Learning with labeled and unlabeled data* (Technical Report). Institute for Adaptive and Neural Computation, University of Edinburgh.

Slonim, N., & Weiss, Y. (2003). Maximum likelihood and the information bottleneck. *Advances in Neural Information Processing Systems 15* (pp. 335–342). Cambridge, MA: MIT Press.

Stark, H., & Woods, J. W. (1994). *Probability, random processes, and estimation theory for engineers*. Englewood Cliffs, New Jersey: Prentice Hall.

Strehl, A., & Ghosh, J. (2002). Cluster ensembles—a knowledge reuse framework for combining partitions. *Journal of Machine Learning Research*, *3*, 583–617.

Tishby, N., Pereira, F. C., & Bialek, W. (1999). The information bottleneck method. In *Proc. 37th Annual Allerton Conf. Communication, Control and Computing* (pp. 368–377).

Wagstaff, K., Cardie, C., Rogers, S., & Schroedl, S. (2001). Constrained k-means clustering with background knowledge. In *Proc. 18th Int. Conf. Machine Learning* (pp. 577–584).

Wu, Y., & Huang, T. S. (2000). Self-supervised learning for visual tracking and recognition of human hand. In *Proc. 17th National Conference on Artificial Intelligence* (pp. 243–248).

Zeng, H.-J., Wang, X.-H., Chen, Z., Lu, H., & Ma, W.-Y. (2003). Cbc: Clustering based text classification requiring minimal labeled data. In *Proc. IEEE Int. Conf. Data Mining* (pp. 443–450). Melbourne, FL.

Zhang, T., & Oles, F. (2000). A probabilistic analysis on the value of unlabeled data for classification problems. In *Proc. 17th Int. Conf. Machine Learning* (pp. 1191–1198).

Zhao, Y., & Karypis, G. (2001). *Criterion functions for document clustering: experiments and analysis* (Technical Report #01-40). Department of Computer Science, University of Minnesota.

Zhong, S. (2005). Efficient online spherical k-means clustering. In *Proc. IEEE Int. Joint Conf. Neural Networks*. Montreal, Canada. (pp. 3180–3185).

Zhong, S., & Ghosh, J. (2003). A unified framework for model-based clustering. *Journal of Machine Learning Research*, *4*, 1001–1037.

Zhong, S., & Ghosh, J. (2005). Generative model-based document clustering: A comparative study. *Knowledge and Information Systems: An International Journal*. 8, 374–384.

Zhu, X., Ghahramani, Z., & Lafferty, J. (2003). Semi-supervised learning using gaussian fields and harmonic functions. In *Proc. 20th Int. Conf. Machine Learning* (pp. 912–919). Washington DC.