Interactive Document Clustering Using Iterative Class-Based Feature Selection

Yeming Hu, Evangelos Milios, James Blustein
Faculty of Computer Science,Dalhousie University
Halifax, NS, Canada
{yeming,eem,jamie}@cs.dal.ca

April 29, 2010

Abstract

Semi-supervised clustering has been found to improve clustering performance by using constraints between documents. Recent research in active learning indicates that feature identification, which takes much less user effort than document labeling, can improve classification performance. We aim to use this new finding to improve document clustering. We first propose an unsupervised clustering framework which involves an iterative updating of the feature set. Users are then invited to help update the feature set by identifying good features. Experiments on various datasets indicate that the performance of document clustering may improve significantly with some user input. In addition, the clustering performance increases initially and then stabilizes with more user effort. Feature reweighting, which gives higher weights to features confirmed by the users, can achieve better clustering performance with less user effort. Based on our experiments, several guidelines are suggested for applying the interactive framework.

1 Introduction

Traditionally, document clustering is an unsupervised classification of a given document collection into clusters so that the documents in the same cluster are more topically similar than those from different clusters. It achieves this goal by either optimizing some loss function over all document assignments such as K-Means [5], or fitting a probabilistic model onto the data set such as Multinomial Naïve Bayes model [13]. The unsupervised process minimizes user effort and generates potential clusters to users. However, users often find that these groupings are not intuitive or do not reflect their point of view. Semi-supervised clustering taking into account user prior knowledge has become a increasingly interesting topic.

Existing semi-supervised clustering methods normally make use of user prior knowledge in the form of document-level pairwise constraints, such as "must-link" or "cannot-link" [19]. Those methods are generally grouped into four categories. First, constraints are used to modify the optimization of the loss function [10] or estimation of parameters [2]. Second, cluster seeds are derived from the constraints to initialize the cluster centroids [1]. Third, constraints are employed to learn adaptive distance using metric learning techniques [6, 16]. Finally, the original high-dimensional feature space can be projected into low-dimensional feature subspaces guided by constraints [17].

However, all existing semi-supervised methods work on document-level constraints, most of which are not designed for high-dimensional data. "Curse of dimensionality" is a well-known problem that the traditional Euclidean notion is not meaningful in high-dimensional datasets [4]. Feature selection is the process of selecting a subset of the features for clustering, which not only alleviates the "curse of dimensionality" problem but also eliminates noisy features. Therefore, feature selection is an effective method for potentially increasing clustering accuracy. Although document constraints are used to project high-dimensional feature space to low-dimensional feature subspaces [17], there is no research, to our knowledge, that involves in direct feature selection during clustering process.

In this paper, we ask users to label features for feature selection instead of labeling pairwise constraints as the existing methods did. This is motivated by the recent finding in active learning [14] that labeling features, which requires much less user effort, can accelerate active learning. In order to allow user interaction with feature selection, we first propose an unsupervised framework, which updates the feature set based on the most recent clusters and clusters the documents iteratively. Second, users are invited to interact with clustering by confirming features in each iteration. Accepted features will be always included in the feature subset for clustering. Within this framework, we also investigate the effect of user effort and feature reweighting on the performance of document clustering.

The remainder of the paper is organized as follows. We review previous related work in Section 2. Our framework for iteratively updating the feature set for clustering and involving users is introduced in Section 4. We discuss the experimental results in Section 5 and conclude in Section 7. In Section 8 we discuss opportunities for future work that arises from our results.

2 Related Work

Our work is related to a number of areas including semi-supervised clustering, feature selection and active learning.

In semi-supervised clustering, instance-level constraints indicate whether the two instances should be placed into the same clusters. As introduced in Section 1, there are four categories of methods to utilize the constraints and the one projecting high-dimensional spaces into low-dimensional subspaces is more related to our work. A technique of constraint-guided feature projection based on instance-level constraints is proposed in [17]. The problem of the constraint-guided feature projection is formulated in the form of an objective function with "must link" and "cannot link" constraints. Then the original dataset is projected into a low-dimensional space such that the distance between any pair of instances involved in "must-link" is minimized and the distance between any pair of instances involved in "cannot-link" is maximized. Finally, a constrained spherical K-Means algorithm was used on the low-dimensional projected data for clustering. Semi-Supervised Projected Clustering [20] utilized limited supervision in the form of both labeled instances and labeled dimensions in subspace clustering. However, the dimensions are labeled before clustering and remains unchanged during the clustering in their method. Also, experiments were performed only on synthetic datasets resembling real gene expression dataset where the correct dimensions were known. In document clustering, we have to decide what features (dimensions) should be presented to users for confirmation. At the same time, the features (words or multi-word terms) in document clustering are easily understood by normal users.

Unlike semi-supervised clustering, active learning works with classification algorithms to achieve maximum document categorization performance with limited user supervision. In the standard active learning setting, learning proceeds iteratively with document classification followed by asking users to label the most uncertain document. Active learning involves document classification [15] and uncertainty sampling [11], in which the user is queried on the unlabeled document the classifier is most uncertain about. Support vector machine (SVM) is usually employed to decide the most uncertain document to query [18]. In each iteration, the document which is closest to the margin is chosen for query. Instead of labeling documents only, users are asked to label features as well as documents in [14]. The experiments in [14] on various text categorization tasks indicate that interleaving labeling features and documents can significantly accelerate active learning. It also indicates that feature reweighting for labeled features can also improve classifier performance over what is achieved via selective sampling alone. Furthermore, experiments show that labeling features takes much less time (about 1/5) than labeling documents. We explore how labeling features and feature reweighting work in the document clustering setting. In addition, the effect of user effort in term of feature labeling is investigated.

Instead of labeling a set of documents, a set of representative words for each class are labeled instead [12]. These words are then used to extract a set of documents for each class, which are used to form the training set. Then, the Expectation-Maximization (EM) algorithm [7] is applied iteratively to build new classifiers. The features are only labeled once for constructing cluster seeds. The SpeClustering Model [9] is a new probabilistic method for text clustering, which assumes that only some of the words in the documents are conditioned on the document's cluster, and that other words follow a more general word distribution independent of which cluster the document belongs to. Then four distinct types of user feedback, including keyword identification for a cluster, are incorporated into SpeClustering model. The features labeled in SpeClustering model are used to modify the estimation of parameters instead of feature selection.

3 Underlying Algorithms and Feature Selection Techniques

In this section, we present the underlying clustering algorithms and feature selection techniques of our framework. In our framework, we test one partitioning algorithm and one probabilistic model, K-Means and Multinomial Naïve Bayes respectively. For traditional document clustering, we employ mean-TFIDF feature selection technique to select feature subset for clustering. For class-based feature selection in our framework we use the χ^2 feature selection technique.

3.1 Clustering Algorithms

Two clustering algorithms, K-Means and Multinomial Naïve Bayes are described in this section. We will assume there are N documents in the collection and K clusters or classes into which the documents should be categorized.

3.1.1 K-Means Clustering Algorithm

K-Means is a very popular clustering algorithm because of its simplicity and efficiency. K-Means clusters data points by optimizing a loss function or distortion measure, given by

$$J = \sum_{i=1}^{N} \sum_{j=1}^{K} r_{ij} \|x_i - \mu_j\|^2 \tag{1}$$

which represents the sum of the squares of the distances of each data point to its assigned vector μ_j [5]. The optimization of J involves finding the assignments $\{r_{ij}\}$ and cluster centroids $\{\mu_j\}$ such that the value of J is minimized. This is usually achieved by an iterative procedure in which each iteration has two alternating steps corresponding to optimizing $\{r_{ij}\}$ and $\{\mu_j\}$. The K-Means algorithm is illustrated in Algorithm 1.

3.1.2 Multinomial Naïve Bayes Model

Multinomial Naïve Bayes model [13] is a commonly used probabilistic model for text clustering, which assumes a document as a vector of words, with each word generated independently by a multinomial probability distribution of the document's class or cluster.

Now suppose we have a labeled training set D and $|D|$ is the size of the collection. In the Naïve Bayes classifier model formulation, $w_{d_i,k}$ denotes the word in position k of document d_i, where each word is from the vocabulary $V = \{w_1, w_2, \ldots, w_{|v|}\}$. The vocabulary is the feature set selected for clustering. There is also a set of predefined classes, $C = \{c_1, c_2, ..., c_n\}$. In order to perform classification, posterior probability $P(c_j|d_i)$ has to be computed from

Algorithm 1 K-Means Clustering Algorithm

- Input: data point vectors $\{d_1, d_2, \ldots, d_N\}$, seed s for cluster centroids $\{u_1, u_2, \ldots, u_K\}$ random initialization,
- Output: data point assignments $\{r_{ij}\}$

```
1: Randomly initialize the cluster centroids $\{u_j\}$ based on the given seed $s$
2: repeat
3:   for all $i = 1$ to $N$ do
4:     Compute all distances $dist_{ij}$ between data point $d_i$ and each cluster centroid $u_j$
5:     Assign data point $d_i$ to the cluster $c_j$ when $dist_{ij}$ is the smallest, namely, $r_{ij} = 1$ when $j = \arg min_k \|d_i - \mu_k\|^2$, otherwise $r_{ij} = 0$
6:   end for
7:   Update cluster centroids $\{u_j\}$ based on the new data point assignments $\{r_{ij}\}$
8: until No data point assignments change or maximum number of iterations is reached
```

prior probability and word conditional probability. Based on Bayesian probability and the multinomial model, we have prior probability

$$p(c_j) = \frac{\sum_{i=1}^{|D|} P(c_j|d_i)}{|D|} \tag{2}$$

and with Laplacian smoothing, we have word conditional probability for each class,

$$p(w_t|c_j) = \frac{1 + \sum_{i=1}^{|D|} N(w_t, d_i) \cdot P(c_j|d_i)}{|V| + \sum_{s=1}^{|V|} \sum_{i=1}^{|D|} N(w_s, d_i) \cdot P(c_j|d_i)} \tag{3}$$

where $N(w_t, d_i)$ is the number of times the word w_t occurs in document d_i. Finally, given the assumption that the probabilities of words given class are independent, we obtain the posterior probability used to classify documents:

$$P(c_j|d_i) = \frac{P(c_j)P(d_i|c_j)}{\sum_{r=1}^{|C|} P(c_r)P(d_i|c_j)} = \frac{P(c_j) \prod_{k=1}^{|d_i|} P(w_{d_i,k}|c_j)}{\sum_{r=1}^{|C|} P(c_r) \prod_{k=1}^{|d_i|} P(w_{d_i,k}|c_j)} \tag{4}$$

In the iterative Multinomial Naïve Bayes Model clustering, the clusters of documents are treated as the predefined classes in each iteration. The prior probability and the word conditional probability of each cluster are computed based on the most recent document distribution in the clusters.

The Expectation-Maximization (EM) algorithm is a widely used iterative algorithm for maximum likelihood estimation for problems involving missing data [7]. Therefore EM algorithm is commonly used to assign the document label (cluster) in the clustering algorithm. There are two steps in each iteration of the EM algorithm, namely, E step and M step. The E step assigns the missing values (cluster labels) and the M step estimates parameters based on the most recent assignments of cluster labels. The Multinomial Naïve Bayes clustering algorithm, also called EM-NB algorithm, is formed by applying EM algorithm to Naïve Bayes classifier. In EM-NB algorithm, Eq. 2 and Eq. 3 are evaluated in the M step and Eq. 4 is evaluated in the E step.

The EM-NB algorithm is given in Algorithm 2. The initial $p(c_j|d_i)$ can be obtained in two ways. It can be derived from clusters obtained from another clustering algorithm like K-Means. In this case, the value of $p(c_j|d_i)$ is 1 when d_i is in cluster c_j. Otherwise, the values is 0. The initial $p(c_j|d_i)$ can also be obtained from another probabilistic model like EM-NB itself, in which case its value is between 0 and 1.

Algorithm 2 EM-NB Clustering Algorithm

- input : data point vectors $\{d_1, d_2, \ldots, d_N\}$ and $P_{initial}(c_j|d_i)$, initial probability that a document belonging to a class (cluster)
- output : $P_{new}(c_j|d_i)$ and *data point assignments* $\{r_{ij}\}$

```
1: repeat
2:   for all $j = 1$ to $|C|$ do
3:     Based on current $P(c_j|d_i)$, compute
         • Prior probability $P(c_j)$ using Eq. 2
         • Word conditional probabilities $P(w_t|c_j)$ using Eq. 3
4:   end for
5:   for all $i = 1$ to $N$ do
6:     for all $j = 1$ to $K$ do
7:       Compute $P_{new}(c_j|d_i)$ given the document using Eq. 4
8:     end for
9:     Assign $d_i$ to cluster $j$, for which $P_{new}(c_j|d_i)$ is maximum, obtain data
       point assignments $\{r_{ij}\}$
10:  end for
11: until No data point assignments change or maximum number of iterations
    is reached
```

3.2 Feature Selection Techniques

Mean-TFIDF, a feature selection technique for traditional document clustering and the χ^2, a class-based feature selection technique are presented in this section.

3.2.1 Mean-TFIDF Feature Selection Technique

Mean-TFIDF feature selection technique is based on the principle that a good feature has high term frequency but low document frequency. It ranks all features by their mean-TFIDF values which are defined as follows. Term frequency tf of a feature j in a document d_i is defined as $tf_{(i,j)} = \frac{n_{(i,j)}}{\sum_k n_{(k,j)}}$ while inverse document frequency idf of a feature j is defined as $idf_j = \log \frac{|D|}{|\{d:ft_j \in d\}|}$ where D denotes the document collection. Then $TFIDF_{(i,j)}$ is the product of tf and idf, namely, $TFIDF_{(i,j)} = tf_{(i,j)} * idf_i$. The mean-TFIDF value of a feature j is the average value of $TFIDF$s over the documents in the collection defined as $mean\text{-}TFIDF_j = \frac{\sum_i TFIDF_{(i,j)}}{|D|}$.

3.2.2 χ^2 Class-Based Feature Selection Technique

The χ^2 value of a feature indicates whether the feature is significantly correlated with a class. Larger values indicate higher correlation. Basically, the χ^2 test aggregates the deviations of the measured probabilities from the expected probabilities assuming independence. Assume random variable $C \in \{0,1\}$ denotes class and random variable $I \in \{0,1\}$ denotes existence of feature j, then χ^2 value of the feature j defined as follows

$$\chi^2 = \sum_{c,i} \frac{[k_{c,i} - nPr(C=c) \cdot Pr(I=i)]^2}{nPr(C=c) \cdot Pr(I=i)} \tag{5}$$

where $k_{c,i}$ is the number of documents in cluster c and with/without feature j indicating by value of i. $Pr(C=c)$ and $Pr(I=i)$ are maximum likelihood probability estimations. Assume there are N documents in the collection. If there are N_c documents in class c, then $Pr(C=c) = N_c/N$. If there are N_i documents with/without feature j indicated by the value of i, then $Pr(I=i) = N_i/N$. In the case of there are more than two classes, the χ^2 value of a feature j is the average of all χ^2 values between feature j and all classes. After obtaining the average χ^2 values, all features are ranked and the top m ones can be used for classification.

When the χ^2 is used for feature selection of document clustering, we treat clusters as classes.

4 A framework of Interactive Document Clustering with Iterative Class-Based Feature Selection

As we discussed in Section 1, the high dimensionality of the document text reduces the clustering algorithm performance. Both feature selection and feature transformation can be used to alleviate this problem. Feature selection has the advantage that it results in a feature subset, which is easier for users to interpret. This motivated asking users to help in the feature selection. In order to involve users in the feature selection, we first introduce an automated framework which uses class-based feature selection to iteratively update the feature set for clustering. Then, instead of updating the feature set automatically, users are brought in to confirm features selected by the feature selection technique in each iteration, which forms an interactive framework. By interacting with the clustering process, users can have a better idea about what clusters should formed when more features are confirmed.

4.1 Document Clustering Using Iterative Class-Based Feature Selection (DCIFS)

In this section, we introduce a fully automated framework which iteratively updates the feature set for clustering based on the most recent formed clusters. This framework involves an iterative procedure, which has two steps in each iteration. In the first step, documents are clustered with the current feature set using the underlying algorithm, e.g. K-Means or EM-NB. In the second step, a new feature set is selected using class-based feature selection, e.g. the χ^2 test, with treating clusters as classes. At the beginning of the framework, the algorithm is initialized by an initial set of clusters obtained using K-Means with feature set selected by mean-TFIDF. The detailed algorithm is illustrated in Algorithm 3.

4.2 Interactive Document Clustering Using Iterative Class-Based Feature Selection (IDCIFS)

We next introduce user interaction to the automated DCIFS framework. Instead of asking users to label documents, we ask users to label or confirm features [12, 14]. In each iteration, the features presented to users for confirmation are the top f features ranked by class-based feature selection, e.g. the χ^2, treating the most recent clusters as classes. Users will give one of three answers when they are presented a feature. If they think the feature is a useful for discriminating among clusters, they will give answer "accept". They will give answer "reject" if they believe the feature does not distinguish between clusters. An answer "don't know" can be given when they are not sure about the feature. How the users interact with document clustering, namely, feature selection, is presented in Algorithm 4. All features accepted by users will be included in the final

Algorithm 3 Document Clustering Using Iterative Class-Based Feature Selection (DCIFS)

- Input: data point vectors $\{d_1, d_2, \ldots, d_N\}$, seed s for K-Means cluster centroids $\{u_j\}$ random initialization, size of selected feature set m
- Output: data point assignments $\{r_{ij}\}$

```
1: Obtain an initial set of clusters $y^c_{initial}$ using K-Means with feature set selected by mean-TFIDF and given seed $s$
2: $y^c \leftarrow y^c_{initial}$
3: **repeat**
4:     Obtain new feature set $FS^m$ based on $y^c$ using class-based feature selection technique, e.g. $\chi^2$
5:     Initialize underlying clustering algorithm with last iteration's parameters
         • K-Means : initialize the cluster centroids based on current clusters $y^c$ and current feature set $FS^m$
         • EM-NB : initialize the $P(c_j|d_i)$ using $P(c_j|d_i)$ obtained from EM-NB of last iteration
6:     Cluster documents using new feature set and initialized underlying clustering algorithm and obtain new clustering $y^c_{new}$ or data point assignments $\{r_{ij}\}$
         • Represent the documents using new feature set $FS^m$
         • Use the initialized underlying clustering algorithm cluster the new represented documents
7:     $y^c \leftarrow y^c_{new}$
8: **until** No data point assignments change or maximum number of iterations is reached
```

feature set for clustering and all features rejected by users are excluded from that. The remaining features, up to the total number of features for clustering (m, an input parameter), are selected according to the ranking obtained by the class-based feature selection based on the most recent clusters.

After obtaining a new feature set with user input, the documents can be reclustered using this new feature set. During the reclustering, the accepted features will be given higher weights. The algorithm for interactive document clustering based on iterative feature selection is given in DCIFS (Algorithm 5).

Algorithm 4 Feature Selection with User Interaction

- Input: m, size of feature set for clustering; f, number of features displayed to users in each iteration; Accepted feature set $FS^{t-1}_{accepted}$ and Rejected feature set $FS^{t-1}_{rejected}$ in iteration $t-1$; Basic feature set FS_{basic} including all features; Recent clustering y^c.
- Output:Accepted feature set $FS^{t}_{accepted}$; Rejected feature set $FS^{t}_{rejected}$.

```
1: $FS^{t}_{accepted} \leftarrow FS^{t-1}_{accepted}$
2: $FS^{t}_{rejected} \leftarrow FS^{t-1}_{rejected}$
3: $FL^{all} \leftarrow$ Rank all features in $FS_{basic}$ by class-based feature selection, e.g. $\chi^2$, based on clustering $y^c$
4: {features accepted or reject will not displayed to users again}
5: $FL = FL^{all} - FS^{t-1}_{accepted} - FS^{t-1}_{rejected}$
6: for all $i = 1$ to $f$ do
7:   Display $i^{th}$ feature in $FL$ to the user, get answer $\$reply$
8:   if $\$reply ==$ "accept" then
9:     Add $i^th$ feature into $FS^{t}_{accepted}$
10:  else if $\$reply ==$ "reject" then
11:    Add $i^th$ feature into $FS^{t}_{rejected}$
12:  else
13:    Do nothing as user gives "don't know" answer
14:  end if
15: end for
```

5 Experiments

5.1 Datasets

We use three datasets to test our proposed framework and explore how clustering performance depends on user effort.

The first data set is widely used 20-Newsgroups collection [1] for text classification and clustering. There are approximately 20,000 newsgroup documents, which are almost evenly partitioned into 20 different newsgroups. **Three** reduced data sets are derived from it [2], which are *News-Different-3*, *News-Related-3*, and *News-Similar-3*. Each derived data set consists of 300 messages, with 100 messages from each of the 3 categories. *News-Different-3* cover topics from 3 quite different newsgroups (alt.atheism, rec.sport.baseball, and sci.space). *News-Related-3* contains 3 related newsgroups (talk.politics.misc, talk.politics.guns, and talk.politics.mideast). *News-Similar-3* consists of messages from 3 similar newsgroups (comp.graphics, comp.os.ms-windows, comp.windows.x). Since *News-Similar-3* has significant overlap between groups, it is the most difficult one to cluster.

The second data set is a collection of papers in full text, which are manually

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

Algorithm 5 Interactive Document Clustering Using Iterative Class-Based Feature Selection (IDCIFS)

```
• Input: data point vectors $\{d_1, d_2, \ldots, d_N\}$; seed $s$ for K-Means cluster centroids
  $\{u_j\}$ random initialization; $f$, the number of features displayed to users each
  time; $g$, the weight of accepted features in $FS^t_{accepted}$; size of selected feature
  set $m$; Basic feature set $FS_{basic}$.
• Output: data point assignments $\{r_{ij}\}$

1: Obtain an initial set of clusters $y^c_{initial}$ using K-Means with feature set selected by mean-TFIDF and given seed $s$
2: $y^c \leftarrow y^c_{initial}$
3: $t \leftarrow 0$
4: $FS^0_{accepted} \leftarrow \{\}$
5: $FS^0_{rejected} \leftarrow \{\}$
6: repeat
7:     $t \leftarrow t + 1$
8:     Feature Selection with User's Interaction, Algorithm 4
           • Obtain $F^t_{accept}$ and $F^t_{rejected}$
           • Obtain new feature set for clustering $FS^m$
9:     Initialize underlying clustering algorithm with last iteration's parameters
           • K-Means : initialize the cluster centroids based on current clusters $y^c$ and current feature set $FS^m$
           • EM-NB : initialize the $P(c_j|d_i)$ using $P(c_j|d_i)$ obtained from EM-NB of last iteration
10:    Cluster documents using new feature set and initialized underlying clustering algorithm and obtain new clustering $y^c_{new}$ or data point assignments $\{r_{ij}\}$
           • Represent the documents using new feature set $FS^m$
           • Use the initialized underlying clustering algorithm cluster the new represented documents
11:    $y^c \leftarrow y^c_{new}$
12: until No data point assignments change or maximum number of iterations is reached
```

collected by the authors from ACM Digital Library[2]. We use the 1998 ACM Computing Classification System to label the categories [3]. In this paper, we

[2]`http://portal.acm.org/dl.cfm`

[3]`http://www.acm.org/about/class/1998/`

use the following categories. D denotes category *Sofware* with two of its subcategories $D.2$ and $D.3$ denoting "Software Engineering" and "Programming Languages" respectively. H is the category "Information Systems" and I denotes "Computing Methodologies". H and I are related as they have overlaps such as "Data Mining" and "Text Clustering" areas. **Two** datasets are derived from ACM paper collection. The first, *D2-D2&D3-D3*, contains papers those are only from $D2$ category, from both $D2$ and $D3$ category, and only from $D3$ category respectively. Each category has 87 papers in this data set and is related to each other as they are all from D category. The second, *D-H-I*, consists of 100 papers from each of D,H,I categories.

The third data set 3-*classic* is made by combining the CISI, CRAN, and MED from the SMART document collection [4]. MED is a collection of 1033 medical abstracts from the Medlars collection. CISI is a collection of 1460 information science abstracts. CRAN is a collection of 1398 aerodynamics abstracts from the Cranfield collection. 100 documents from each category are sampled to form the reduced 3-*classic* dataset. The topics are quite different across categories, like *News-Different-3*.

We pre-process each document by tokenizing the text into bag-of-words. A word is defined as a consecutive English letters delimited by spaces, newlines, punctuations, and so on. Then, we remove the stop words and stem all other words. The top m features ranked either by mean-TFIDF or the χ^2 test are employed for clustering. For K-Means-based algorithm, a feature vector for each document is constructed with TFIDF weighting and then normalized. For EM-NB-based algorithm, the term frequency of the selected features is directly used in the related algorithm.

5.2 Cluster Evaluation Measures

We use three measures to evaluate the cluster quality: clustering accuracy [3], normalized mutual information (NMI) [8], and Jaccard Index [3]. Clustering accuracy and NMI are two external clustering validation metrics that estimate the clustering quality with respect to a given collection of labeled documents. They measure how close the reconstructed clusters are to the underlying classes of the documents. Jaccard index measures the similarity between two different clusterings. One of the two clusterings could be either computed clusters or underlying classes while the other one is computed clusters.

Assume we have a clustering T and underlying classes C. To estimate the clustering accuracy, we map each cluster $t \in T$ to one underlying class $c \in C$ when the documents from c dominate t. Then we define $n(t)$ as the number of dominating documents in t from c. The clustering accuracy $CACC$ of T with respect to C is defined as:

$$CACC(T,C) = \frac{\sum_t n(t)}{\sum_t |t|} = \frac{\sum_t n(t)}{N} \tag{6}$$

[4]`ftp://ftp.cs.cornell.edu/pub/smart`

where N is the size of the document collection. As [3] points out, it is meaningless when the number of clusters K is very large. For example, $CACC$ is 1 when K equals N, the number of documents in the collection. In all our experiments, we set K the same as the number of underlying classes.

NMI measures the share information between the cluster assignments S and class labels L of documents. It is defined as:

$$NMI(S,L) = \frac{I(S,L)}{(H(S)+H(L))/2} \tag{7}$$

where $I(S,L)$, $H(S)$, and $H(L)$ denote the mutual information between S and L, the entropy of S, and the entropy of L respectively. Assume there are K classes and K clusters, N documents, $n(l_i)$ denotes the number of documents in class l_i, $n(s_j)$ denotes the number of documents in cluster s_j, $n(l_i,s_j)$ denotes the number of documents in both class l_i and cluster s_j. Then we have $H(L) = -\sum_{i=1}^{K} P(l_i)log_2 P(l_i)$, $H(S) = -\sum_{j=1}^{K} P(s_j)log_2 P(s_j)$, $I(S,L) = -\sum_{i=1}^{K}\sum_{j=1}^{K} P(l_i,s_j)log_2 \frac{P(l_i,s_j)}{P(l_i)P(s_j)}$, where $P(l_i) = n(l_i)/N$, $P(s_j) = n(s_j)/N$ and $P(l_i,s_j) = n(l_i,s_j)/N$.

We use Jaccard Index to measure similarities between two clusterings when no underlying class labels are used. Jaccard index is defined as in [3]. Given two clusterings y_1^c and y_2^c, we define a the number of document pairs, such that two documents are from the same cluster in both y_1^c and y_2^c, b the number of document pairs, such that two documents are from the same cluster in y_1^c but not in y_2^c, c the number of document pairs, such that two documents are from the same cluster in y_2^c but not in y_1^c. Then the Jaccard Index between y_1^c and y_2^c is defined as:

$$J(y_1^c, y_2^c) = \frac{a}{a+b+c} \tag{8}$$

It is noted that the value of Jaccard Index is between 0 and 1.

5.3 Results and Discussion

We first present the results of the underlying algorithms with feature sets selected by different feature selection techniques. Then we explore how clustering performance depends on user effort. Our experiments find that clustering performance does not always increase in proportion to the user effort and feature reweighting helps improve clustering performance over treating all features equally.

5.3.1 Performance of Different Feature Sets

We compare the two proposed frameworks DCIFS and IDCIFS with the corresponding baseline underlying algorithms K-Means and EM-NB. Since our framework aims to select better feature set for clustering, the underlying algorithms with feature sets selected by different methods are compared. The various feature sets are listed as follows:

- FS_{basic} : feature set including all features extracted, namely, without doing any feature selection.
- $FS_{mean\text{-}TFIDF}$: feature set selected by mean-TFIDF feature selection technique.
- $FS_{iterative}$: feature set used in the proposed framework DCIFS.
- $FS_{interactive}$: feature set used in the proposed framework IDCIFS.
- FS_{ideal} : ideal feature set, selected by χ^2 feature selection technique based on the class labels of the documents.

Therefore, we have the following pairs of algorithm and feature set in Table 1:

Table 1: Combinations of underlying algorithms and feature sets

Feature Set	Underlying Algorithm	
	K-Means	**EM-NB**
basic	*K-Means*+FS_{basic}	*EM-NB*+FS_{basic}
mean-TFIDF	*K-Means*+$FS_{mean\text{-}TFIDF}$	*EM-NB*+$FS_{mean\text{-}TFIDF}$
Iterative	*K-Means*+$FS_{iterative}$	*EM-NB*+$FS_{iterative}$
Interactive	*K-Means*+$FS_{interactive}$	*EM-NB*+$FS_{interactive}$
Ideal	*K-Means*+FS_{ideal}	*EM-NB*+FS_{ideal}

Each method of the pairs was run 36 times with different initializations over all the datasets. In our experiments, we take the top 600 features ranked by the feature selection technique as the feature set for clustering. The average results are listed in Table 2 for K-Means as the underlying algorithm and Table 3 for EM-NB as the underlying algorithm. For the performance of interactive feature set, we take the average performance when the performance stabilizes with the number of feature f displayed to users, e.g. f is between 100 and 300. In Table 2 and Table 3, the performance of the feature set improves significantly when it moves left to right except those in bold. The only exception in Table 2 where K-Means is used as the underlying algorithm is Jaccard Index between $FS_{mean\text{-}TFIDF}$ and $FS_{iterative}$ of news-similar dataset. In Table 3 where EM-NB is used as the underlying algorithm, the exception is still between $FS_{mean\text{-}TFIDF}$ and $FS_{iterative}$ but it includes NMI and Accuracy measures of news-diff dataset and all measures of news-similar dataset. Although the proposed automated DCIFS (Algorithm 3) does not always perform better than the baseline, the DCIFS (Algorithm 5) with user interaction does. Especially, when the automated DCIFS (Algorithm 3) works much worse than the baseline algorithm for news-similar dataset, user interaction can bring the clustering back to the right track and obtain better performance than the baseline algorithms. It is also noted that our DCIFS (Algorithm 5) with user interaction achieves comparable performance as the underlying algorithm with the ideal feature set FS_{ideal}.

Table 2: Comparison of Performances Of K-Means as the Underlying Algorithm with Different Feature Sets for Clustering, namely, FS_{basic}, $FS_{mean\text{-}TFIDF}$, $FS_{iterative}$, $FS_{interactive}$, FS_{ideal}

Dataset	Measure	Performance by Feature Sets				
		basic	**mean-TFIDF**	**Iterative**	**Interactive**	**Ideal**
news-diff	NMI	0.4051	0.5957	0.6651	0.7084	0.6804
	Accuracy	0.6941	0.7931	0.8335	0.8522	0.8330
	Jaccard Index	0.4819	0.6263	0.6476	0.6801	1.0000
news-related	NMI	0.1755	0.3341	0.4116	0.4702	0.4501
	Accuracy	0.5285	0.5931	0.6334	0.6722	0.6768
	Jaccard Index	0.3748	0.4956	0.5278	0.5570	1.0000
news-similar	NMI	0.0380	0.0765	0.1004	0.1938	0.1818
	Accuracy	0.4243	0.4669	0.4988	0.5479	0.5411
	Jaccard Index	0.3561	**0.3833**	**0.3819**	0.5344	1.0000
D2-D2&D3-D3	NMI	0.1609	0.2315	0.2727	0.2912	0.2736
	Accuracy	0.5404	0.5971	0.6293	0.6438	0.6235
	Jaccard Index	0.4105	0.5618	0.6292	0.6702	1.0000
D-H-I	NMI	0.1051	0.1786	0.2193	0.2594	0.2082
	Accuracy	0.4699	0.5335	0.5794	0.6115	0.5496
	Jaccard Index	0.4753	0.5673	0.5251	0.6651	1.0000
3-Classic	NMI	0.5779	0.7220	0.7626	0.8079	0.7854
	Accuracy	0.7544	0.8481	0.8755	0.9017	0.8744
	Jaccard Index	0.6192	0.7462	0.7801	0.8127	1.0000

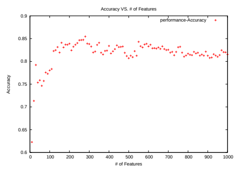
Table 3: Comparison of Performances Of EM-NB as the Underlying Algorithm with Different Feature Sets for Clustering, namely, FS_{basic}, $FS_{mean\text{-}TFIDF}$, $FS_{iterative}$, $FS_{interactive}$, FS_{ideal}

Dataset	Measure	Performance by Feature Sets				
		basic	**meanTFIDF**	**Iterative**	**Interactive**	**Ideal**
news-diff	NMI	0.5267	**0.6742**	**0.6737**	0.7845	0.7879
	Accuracy	0.7622	**0.8474**	**0.8450**	0.9050	0.9034
	Jaccard Index	0.5471	0.6867	0.7208	0.8318	1.0000
news-related	NMI	0.1966	0.3756	0.3933	0.5227	0.5741
	Accuracy	0.5469	0.6093	0.6150	0.7051	0.7273
	Jaccard Index	0.3450	0.5012	0.5257	0.5995	1.0000
news-similar	NMI	0.0819	**0.1491**	**0.0259**	0.1925	0.2114
	Accuracy	0.4742	**0.4464**	**0.3481**	0.4793	0.5379
	Jaccard Index	0.3722	**0.6354**	**0.5925**	0.6765	1.0000
D2-D2&D3-D3	NMI	0.1834	0.2435	0.2486	0.3178	0.3281
	Accuracy	0.5582	0.5596	0.5653	0.6082	0.6493
	Jaccard Index	0.4077	0.5513	0.6086	0.6875	1.0000
D-H-I	NMI	0.1051	0.1786	0.2193	0.2920	0.2082
	Accuracy	0.4881	0.3678	0.4796	0.5967	0.5840
	Jaccard Index	0.4333	0.5112	0.5419	0.7525	1.0000
3-Classic	NMI	0.6829	0.8182	0.8412	0.8841	0.8960
	Accuracy	0.7946	0.9069	0.9179	0.9439	0.9503
	Jaccard Index	0.6683	0.8199	0.8467	0.9069	1.0000

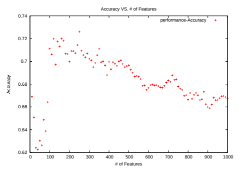
5.3.2 Effect of Feature Set Size

Sometimes, DCIFS (Algorithm 5) with user interaction obtains better performance than the underlying algorithm with ideal feature set Table 2 and 3. The possible reason could be that the so-called ideal feature set is not really ideal. This may be due to several reasons. First, the ideal feature set is selected based

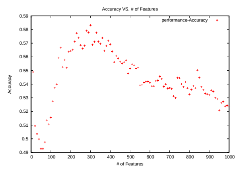
Figure 1: Effect of feature set sizes on the accuracy of clustering of different datasets.(a),(b),(c), (d),(e),(f) with *K-Means* as the underlying algorithm and *ideal feature set.*



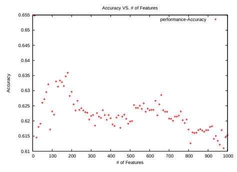
(a) news-diff



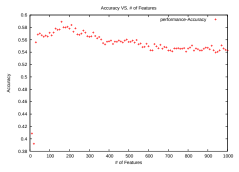
(b) news-related



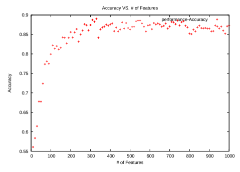
(c) news-similar



(d) ACM (D2-D2&D3-D3)



(e) ACM (D-H-I)



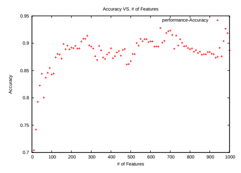
(f) 3-classic

on the labeled documents, from which different class-based feature selection techniques may have different rankings of features. Second, the size of feature set m for clustering may affect clustering performance. The effect of feature set size in terms of clustering accuracy is illustrated in Fig. 1 and 2. More figures illustrating the effect of feature set size are available in Appendix A. The performance of both K-Means and EM-NB increases initially when applied to all datasets. Maximum performance can be reached with different feature set sizes normally between 200 and 400. The clustering performance of both K-Means and EM-NB on datasets with classes of documents on different topics, such as news-diff dataset and 3-classic dataset, remains stable as a function of feature set size after the maximum performance is reached, while the performance on other datasets goes down a little. Our explanation is that there are much more noisy features in the ideal feature set for datasets like news-similar dataset than others. As more features are added, the "good" features dominate at first but noisy features take over later on. Comparing Fig. 2 of EM-NB to Fig. 1 of K-Means on news-related, news-similar, ACM (D-H-I) and ACM (D2-D2&D3-D3) datasets, it is found that EM-NB algorithm has less variations when noisy features added in the later on.

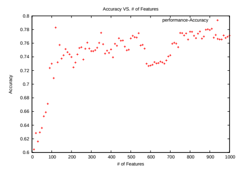
5.3.3 Effect of User Effort

In this section, we study the effect of user effort on clustering performance with feature reweighting. In the proposed framework IDCIFS 5, the number of

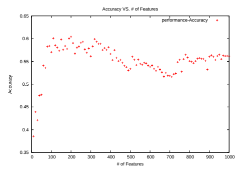
Figure 2: Effect of feature set sizes on different datasets.(a),(b), (c),(d), (e),(f) with *EM-NB* as the underlying algorithm and *ideal feature set*.



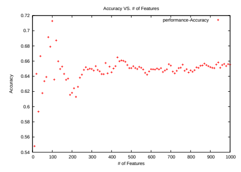
(a) news-diff



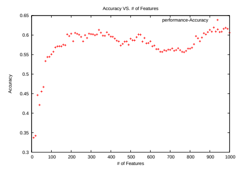
(b) news-related



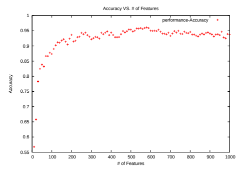
(c) news-similar



(d) ACM (D2-D2&D3-D3)



(e) ACM (D-H-I)



(f) 3-classic

features presented to users in each iteration f, is given as an input parameter. During clustering, different f values can result in different user effort in terms of $f_{total} = f * r$ where r is the number of iterations, total number of features presented to users when the algorithm converges. Out of the f_{total} features presented to users, $f_{accepted}$ features are accepted by users. We define user effort efficiency $eff\text{-}eff$ in terms of f_{total} and $f_{accepted}$ as :

$$eff\text{-}eff = \frac{f_{accepted}}{f_{total}} \tag{9}$$

Since Raghavan et al. [14] find that feature reweighting can boost classification performance in active learning, we incorporated the feature reweighting by giving higher weight to the accepted features. Feature reweighting is incorporated in the following method. When K-Means is used as the underlying algorithm, the *TFIDF* values of accepted features is multiplied by the given weight g and then the vector of meanTFIDF values is normalized. Consider an example: a data point with *TFIDF* vector $X = (2, 3)$ and the feature of the first dimension is accepted by users but the second one is not. Without feature reweighting, the vector X would be normalized to $X^{'} = 2/\sqrt{2^2 + 3^2}, 3/\sqrt{2^2 + 3^2}$. However, after the weight g is incorporated, X becomes $X^{rew} = (2g, 3)$ and the normalized vector is $X^{'} = 2g/\sqrt{(2g)^2 + 3^2}, 3/\sqrt{(2g)^2 + 3^2}$. For EM-NB, g affects Eq. 3 in terms of the feature term frequency:

$$p(w_t|c_j) = \frac{1 + \sum_{i=1}^{|D|} \boldsymbol{g}_t \cdot N(w_t, d_i) \cdot P(c_j|d_i)}{|V| + \sum_{s=1}^{|V|} \sum_{i=1}^{|D|} \boldsymbol{g}_s \cdot N(w_s, d_i) \cdot P(c_j|d_i)} \tag{10}$$

where g_s is the weight given to word (feature) w_s. The weight g_s of a given word is defined as :

$$|g_s| = \begin{cases} g & \text{if } w_s \text{ is accepted} \\ 1 & \text{otherwise} \end{cases} \tag{11}$$

In our experiments, g is an integer between 1 and 10.

Using the above definitions, the effect of user effort on clustering performance is divided into four questions:

1. How does f_{total} change with f?
2. How does clustering performance change with f and f_{total}?
3. How does feature reweighting affect clustering performance?
4. How does feature reweighting affect user effort?

The effect of user effort on news-diff, news-related, news-similar datasets is shown in Fig. 3 and Fig. 4 for K-Means and EM-NB as the underlying algorithm respectively [5]. The four questions are answered one by one as follows.

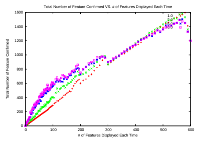
For all datasets, the user effort spent in terms of f_{total} until algorithm 5 converges increases with f, the number of features presented to users in each iteration, e.g. 5(a). We also note that the effort efficiency declines when more features displayed in each iteration, e.g. 5(b). An explanation for this result is that this may be that the more features are displayed each time, the higher proportion of features displayed are not in the "ideal feature set", which are rejected.

Generally speaking, clustering performance increases with more effort provided from users such as 2(c) 3(b). However, when IDCIFS (Algorithm 5) with K-Means as underlying algorithm works with news-related dataset and ACM (D-H-I) dataset, the clustering performance declines after a certain amount of effort is provided, such as in Fig. 10 and Fig. 13. One possible reason is that the extra effort later is used to introduce noisy feature in the so-called "ideal feature set".

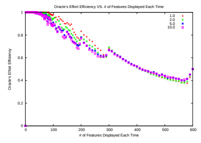
One important finding is that the algorithm converges very quickly when f is very small so that the total number of features accepted is only a small portion. When weight g is greater than 1 and total accepted features f_{total} is

[5] Only NMI as performance measure and values of weight $\{1, 2, 5, 10\}$ are used in those figures. Figures for effect of user effort on other datasets, accuracy as performance measure and other weight values can be found in Appendix A

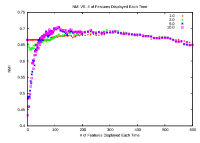
Figure 3: Effect of user effort on newsgroups datasets. (a), (b), show the effect of user effort on news-diff dataset with kmeans as the underlying algorithm. (e), (f), show the effect of user effort on news-related dataset with kmeans as the underlying algorithm. (i), (j), show the effect of user effort on news-similar dataset with kmeans as the underlying algorithm.



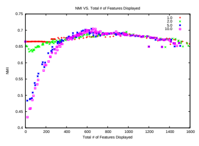
(a) K-Means on news-diff dataset: (x axis-f) vs. (y axis-f_{total})



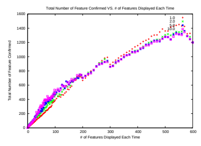
(b) K-Means on news-diff dataset: (x axis-f) vs. (y axis-eff-eff)



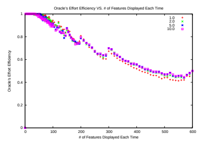
(c) K-Means on news-diff dataset: (x axis-f) vs. (y axis-NMI)



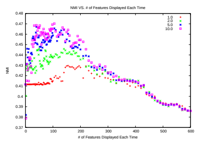
(d) K-Means on news-diff dataset: (x axis-f_{total}) vs. (y axis-NMI)



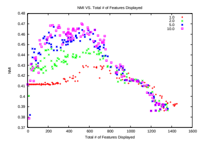
(e) K-Means on news-related dataset: (x axis-f) vs. (y axis-f_{total})



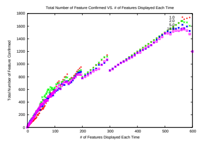
(f) K-Means on news-related dataset: (x axis-f) vs. (y axis-eff-eff)



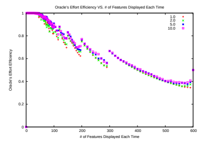
(g) K-Means on news-related dataset: (x axis-f) vs. (y axis-NMI)



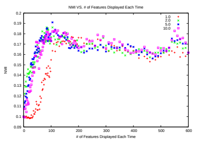
(h) K-Means on news-related dataset: (x axis-f_{total}) vs. (y axis-NMI)



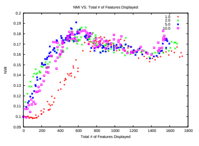
(i) K-Means on news-similar dataset: (x axis-f) vs. (y axis-f_{total})



(j) K-Means on news-similar dataset: (x axis-f) vs. (y axis-eff-eff)

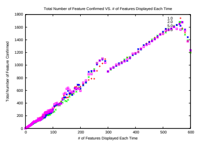


(k) K-Means on news-similar dataset: (x axis-f) vs. (y axis-NMI)

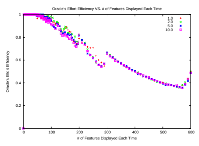


(l) K-Means on news-similar dataset: (x axis-f_{total}) vs. (y axis-NMI)

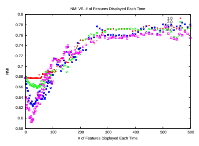
Figure 4: Effect of user effort on newsgroups datasets. (a), (b), show the effect of user effort on news-diff dataset with EM-NB as the underlying algorithm. (e), (f), show the effect of user effort on news-related dataset with EM-NB as the underlying algorithm. (i), (g), show the effect of user effort on news-similar dataset with EM-NB as the underlying algorithm.



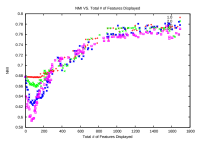
(a) EM-NB on news-diff dataset: (x axis-f) vs. (y axis-f_{total})



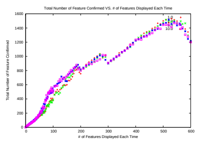
(b) EM-NB on news-diff dataset: (x axis-f) vs. (y axis-eff-eff)



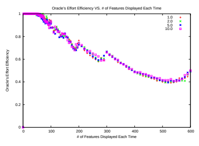
(c) EM-NB on news-diff dataset: (x axis-f) vs. (y axis-NMI)



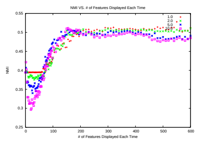
(d) EM-NB on news-diff dataset: (x axis-f_{total}) vs. (y axis-NMI)



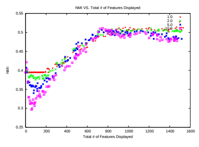
(e) EM-NB on news-related dataset: (x axis-f) vs. (y axis-f_{total})



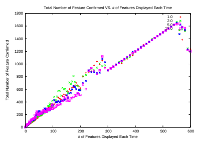
(f) EM-NB on news-related dataset: (x axis-f) vs. (y axis-eff-eff)



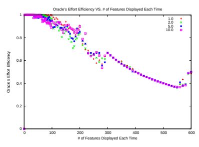
(g) EM-NB on news-related dataset: (x axis-f) vs. (y axis-NMI)



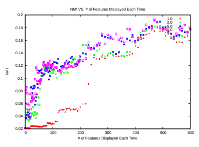
(h) EM-NB on news-related dataset: (x axis-f_{total}) vs. (y axis-NMI)



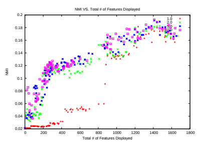
(i) EM-NB on news-similar dataset: (x axis-f) vs. (y axis-f_{total})



(j) EM-NB on news-similar dataset: (x axis-f) vs. (y axis-eff-eff)



(k) EM-NB on news-similar dataset: (x axis-f) vs. (y axis-NMI)



(l) EM-NB on news-similar dataset: (x axis-f_{total}) vs. (y axis-NMI)

very small, the accepted features could be over-emphasized and have negative effect on IDCIFS (Algorithm 5) with EM-NB as the underlying algorithm. For IDCIFS with EM-NB, probabilities of unaccepted features in the feature set for clustering are affected through Eq. 10 and the performance in terms of NMI declines first and climbs back when more features are accepted by users.

In our experiments, we tried different g values from 1 to 10 for accepted features. Comparing the effect of different g values on various datasets, it can be found that feature reweighting helps document clustering process. It can either improve clustering performance Fig. 2(b) or help reach maximum clustering performance earlier Fig. 2(c), which saves user effort. When IDCIFS with EM-NB works with $g > 1$, it improves performance when applied to news-similar dataset(which represents the dataset that is the hardest to cluster) although it achieves comparable performance when applied to other datasets. We suggest 5 should be enough for g value of feature reweighting, which can also avoid over-emphasis on accepted features Fig. 12.

We also compare IDCIFS with K-Means versus EM-NB as the underlying algorithm on the same datasets, e.g. Fig. 6. It is found that IDCIFS with EM-NB are more stable than with K-Means once maximum performance is reached. Particularly, IDCIFS with K-Means declines more strongly after maximum performance is reached when applied to news-related dataset and ACM (D-H-I) dataset. When applied to newsgroups sub-datasets, we find that IDCIFS with K-Means works generally better even when there is only little effort from users, e.g. $f_{total} < 100$.

We compare IDCIFS with K-Means and IDCIFS with EM-NB on the three newsgroups sub-datasets. From Fig. 5, We can tell that the news-similar dataset is still the most difficult one to be grouped and the news-diff dataset is the easiest one when user effort is available.

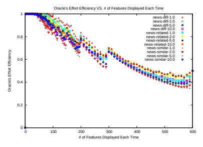
6 Guidelines for Designing Interactive Framework

Based on our experiments on different datasets, guidelines for applying interactive framework can be derived.

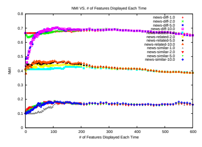
1. Users should be allowed to change the number of features f during the clustering process. Since small f values may cause early algorithm convergence before good clusters are achieved and large f values may require more user effort for the same performance, allowing users to change f value gives users more opportunities to interact with the clustering process and obtain better clusters with less effort. We suggest starting with 100 for f.

2. Keep snapshots of clusters. As human users can also make mistakes in identifying features, clustering performance can decrease if some (noisy) features are confirmed. By storing the history of clustering, the users

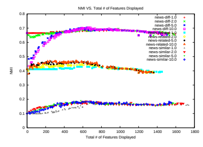
Figure 5: DCIFS (Algorithm 5) with the same underlying algorithm on news-diff, news-related, news-similar datasets



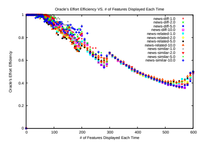
(a) K-Means on newsgroups datasets: (x axis-f) vs. (y axis-eff-eff)



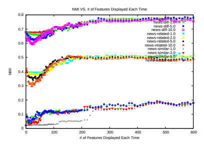
(b) K-Means on newsgroups dataset: (x axis-f) vs. (y axis-NMI)



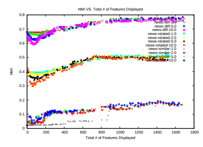
(c) K-Means on newsgroups dataset: (x axis-f_{total}) vs. (y axis-NMI)



(d) EM-NB on newsgroups datasets: (x axis-f) vs. (y axis-eff-eff)



(e) EM-NB on newsgroups dataset: (x axis-f) vs. (y axis-NMI)



(f) EM-NB on newsgroups dataset: (x axis-f_{total}) vs. (y axis-NMI)

can roll back to previous clusters and make new decisions about feature selections from there.

3. Visualization of clusters. In order for users to judge the quality of clusters, visualization techniques such as multi-document summarization should be applied to clusters.

4. Users should be allowed to change weights for accepted features. Our recommendation for the weight value is 5, but users should have the choice to increase or decrease the weight for the accepted features or even assign weights for individual features according to their confidence in the feature. However, assigning individual weights may be time-consuming.

7 Conclusion

In this paper, we first propose an automated framework **D**ocument **C**lustering Using **I**terative Class-Based **F**eature **S**election (DCIFS), which iteratively updates the feature set for clustering and improves the clustering performance over baselines in most cases. In order to eliminate noisy features selected in the automated framework, **I**nteractive **D**ocument **C**lustering Using **I**terative Class-Based **F**eature **S**election (IDCIFS) is proposed, in which users are invited to confirm whether a feature is good or not. Experiments show that the framework IDCIFS improves clustering performance over pure iterative framework DCIFS.

Based on the interactive framework, we study the effect of user effort on clustering performance. Our experiments indicate that a certain amount of features has to be confirmed by users in order to improve the clustering performance and avoid early convergence. After a certain number of features are confirmed by users, the performance either remains stable or declines slightly with more user effort. We give higher weights to accepted features by feature reweighting, which could help clustering by improving performance with less user effort. However, a large weight should be avoided because it may over-emphasize the accepted features for some datasets.

We also studied the effect of size of ideal feature set on the document clustering. Generally speaking, the top 200-400 features ranked by χ^2 are good for all datasets but one dataset, for which the top 100 features works the best. The size of the feature set has a stronger effect on datasets with classes of similar topics than datasets with classes of different topics, which means datasets with classes of similar topics are more sensitive to noisy features. Our framework with user interaction can reduce the effect of noisy features as feature reweighting gives higher weights to the user accepted features.

At the end, we give guidelines for designing interactive framework with input from users.

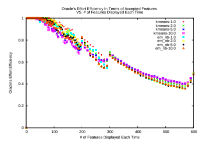
8 Future Work

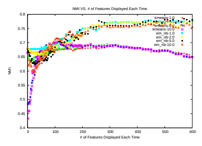
This paper uses an ideal feature set to simulate user interaction. Therefore, we plan to explore whether the results extend to the case when humans are employed to identify features in our future work. There are two approaches to the involvement of human users.

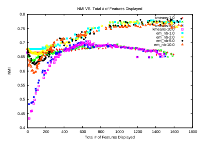
In the first approach, we will ask users to create an "ideal feature set". First, an ordered list of features is created by class-based feature selection based on classes of documents. Second, human users will look through the list and select valid features. This human created "ideal feature set" will be used as the "ideal feature set" in our interactive framework.

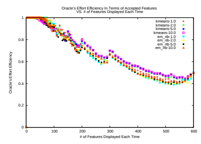
In the second approach, no "ideal feature set" is created, users are asked to identify the good features during each iteration based on their current understanding of the document collection. The guidelines summarized in Section 6 will be applied so that users can have more flexibility when they interact with the document clustering process. With the ability to roll back to previous clustering and change both g and f, users can avoid being trapped in local optima. Although we propose that users can give one of three answers for a given feature, only the "accepted" answer was explored in our experiments. In the future user studies, all three options will be provided to humans and we will investigate whether the "rejected" option could save user effort.

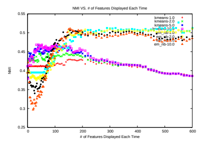
Figure 6: DCIFS (Algorithm 5) with different underlying algorithms on the same newsgroups datasets

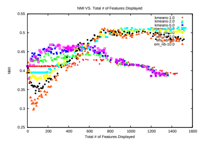

(a) K-Means and EM-NB on news-diff dataset: (x axis-f) vs. (y axis-eff-eff)

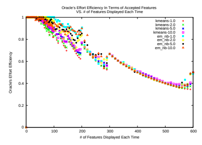

(b) K-Means and EM-NB on news-diff dataset: (x axis-f) vs. (y axis-NMI)

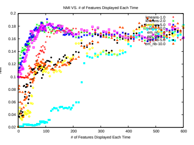

(c) K-Means and EM-NB on news-diff dataset: (x axis-f_{total}) vs. (y axis-NMI)

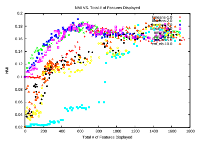

(d) K-Means and EM-NB on news-related dataset: (x axis-f) vs. (y axis-eff-eff)


(e) K-Means and EM-NB on news-related dataset: (x axis-f) vs. (y axis-NMI)


(f) K-Means and EM-NB on news-related dataset: (x axis-f_{total}) vs. (y axis-NMI)


(g) K-Means and EM-NB on news-similar dataset: (x axis-f) vs. (y axis-eff-eff)


(h) K-Means and EM-NB on news-similar dataset: (x axis-f) vs. (y axis-NMI)


(i) K-Means and EM-NB on news-similar dataset: (x axis-f_{total}) vs. (y axis-NMI)

9 References

[1] S. Basu, A. Banerjee, and R. Mooney. Semi-supervised clustering by seeding. In *International Conference on Machine Learning*, pages 19–26, 2002.

[2] S. Basu, M. Bilenko, and R.J. Mooney. A probabilistic framework for semi-supervised clustering. In *Proceedings of the tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 59–68. ACM, 2004.

[3] R. Bekkerman, M. Scholz, and K. Viswanathan. Improving clustering stability with combinatorial MRFs. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 99–108. ACM, 2009.

[4] K. BEYER, J. GOLDSTEIN, R. RAMAKRISHNAN, and U. SHAFT. When is nearest neighbor meaningful? In *3rd International Conference on Database Theory*, pages 217–235, 1999.

[5] C.M. Bishop. *Pattern Recognition and Machine Learning*. Springer New York, 2006.

[6] H. Cheng, K.A. Hua, and K. Vu. Constrained locally weighted clustering. *Proceedings of the PVLDB'08*, 1(1):90–101, 2008.

[7] A.P. Dempster, N.M. Laird, D.B. Rubin, et al. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[8] B.E. Dom. An information-theoretic external cluster-validity measure. Technical Report RJ 10219, IBM Research Division, 2001.

[9] Y. Huang and T.M. Mitchell. Text clustering with extended user feedback. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 420. ACM, 2006.

[10] X. Ji and W. Xu. Document clustering with prior knowledge. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 412. ACM, 2006.

[11] D.D. Lewis and J. Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 148–156, 1994.

[12] B. Liu, X. Li, W.S. Lee, and P.S. Yu. Text classification by labeling words. In *Proceedings of the National Conference on Artificial Intelligence*, pages 425–430. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2004.

[13] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Learning to classify text from labeled and unlabeled documents. In *Proceedings of the National Conference on Artificial Intelligence*, pages 792–799, 1998.

[14] H. Raghavan, O. Madani, and R. Jones. Interactive feature selection. In *Proceedings of IJCAI 05: The 19th International Joint Conference on Artificial Intelligence*, pages 841–846, 2005.

[15] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys (CSUR)*, 34(1):1–47, 2002.

[16] N. Tang and V.R. Vemuri. User-interest-based document filtering via semi-supervised clustering. *Foundations of Intelligent Systems*, pages 573–582, 2005.

[17] W. Tang, H. Xiong, S. Zhong, and J. Wu. Enhancing semi-supervised clustering: a feature projection perspective. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 707–716. ACM, 2007.

[18] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *The Journal of Machine Learning Research*, 2:45–66, 2002.

[19] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl. Constrained k-means clustering with background knowledge. In *Proceedings of the Eighteenth International Conference on Machine Learning*, volume 577, page 584, 2001.

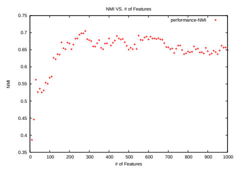
[20] KP Yip, DW Cheung, and MK Ng. On discovery of extremely low-dimensional clusters using semi-supervised projected clustering. In *Proceedings of the 21st International Conference on Data Engineering(ICED-05)*, pages 329–340, 2005.

A Experiment Figures for All Datasets

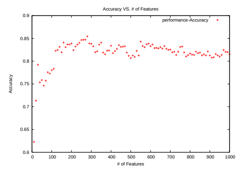
In this appendix, we put related figures for effect of user effort for all datasets with all weights $\{1, 2, \ldots, 10\}$ and both NMI and accuracy as performance. Figures for effect of feature size in term of both NMI and accuracy are also listed.

A.1 Effect of Feature Size on Different Datasets, K-Means Algorithm

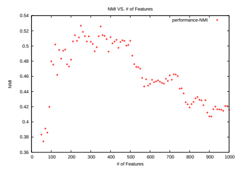
Figure 7: Effect of feature set sizes on different datasets (a),(c), (e),(g), (i),(k) use NMI as performance measure. (b),(d), (f),(h), (j),(l) use accuracy as performance measure. x axis-feature set size, y axis-the corresponding performance, NMI or Accuracy.



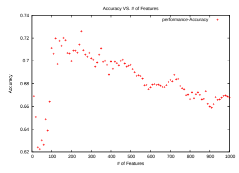
(a) news-diff,NMI



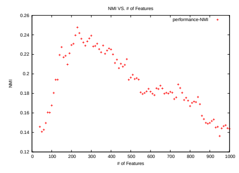
(b) news-diff,Accuracy



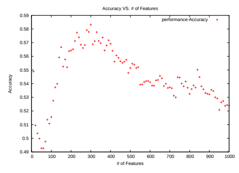
(c) news-related,NMI



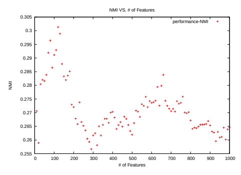
(d) news-related,Accuracy



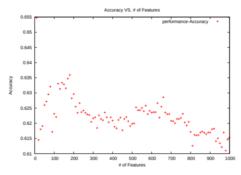
(e) news-similar,NMI



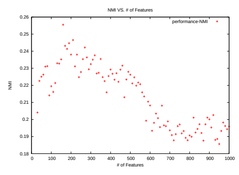
(f) news-similar,Accruacy



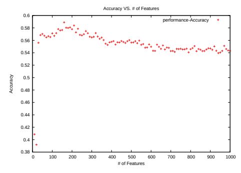
(g) ACM (D2-D2&D3-D3),NMI



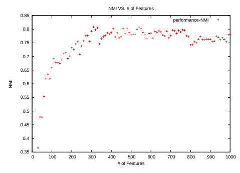
(h) ACM (D2-D2&D3-D3),Accuracy



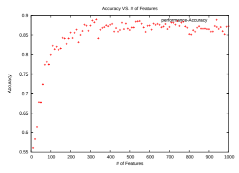
(i) ACM (D-H-I),NMI



(j) ACM (D-H-I),Accuracy



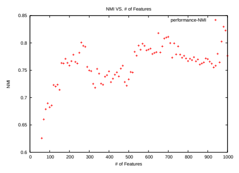
(k) 3-classic,NMI



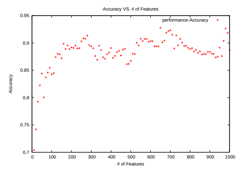
(l) 3-classic,Accuracy

A.2 Effect of Feature Size on Different Datasets, EM-NB Algorithm

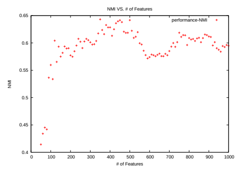
Figure 8: Effect of feature set sizes on different datasets (a),(c), (e),(g), (i),(k) use NMI as performance measure. (b),(d), (f),(h), (j),(l) use accuracy as performance measure. x axis-feature set size, y axis-the corresponding performance, NMI or Accuracy.



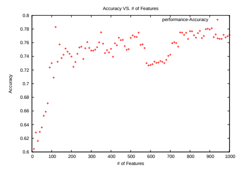
(a) news-diff,NMI



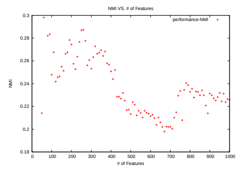
(b) news-diff,Accuracy



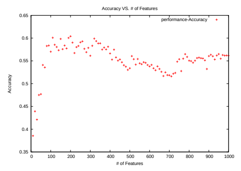
(c) news-related,NMI



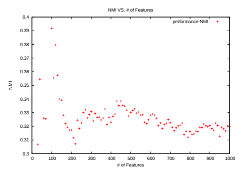
(d) news-related,Accuracy



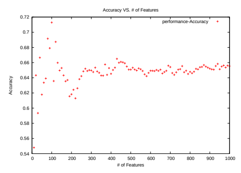
(e) news-similar,NMI



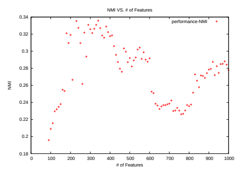
(f) news-similar,Accuracy



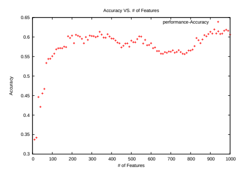
(g) ACM (D2-D2&D3-D3),NMI



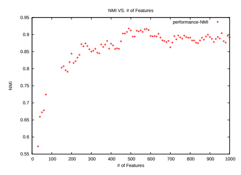
(h) ACM (D2-D2&D3-D3),Accuracy



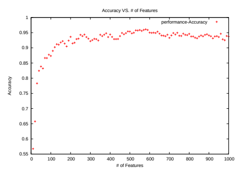
(i) ACM (D-H-I),NMI



(j) ACM (D-H-I),Accuracy



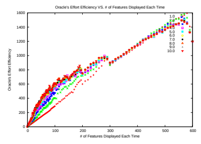
(k) 3-classic,NMI

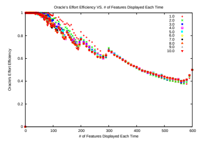


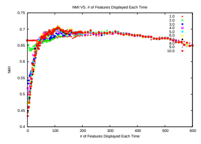
(l) 3-classic,Accuracy

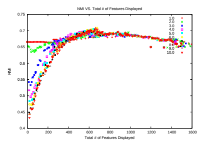
A.3 Effect of User Effort on News-Diff Dataset

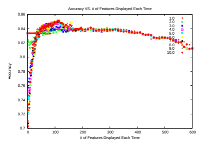
Figure 9: Effect of user effort on news-diff dataset. (a), (b), (c), (d), (e) and (f) use K-Means as the underlying algorithm while (g), (h), (i), (j), (k) and (l) use EM-NB as the underlying algorithm. Legends are all weights between 1 and 10.

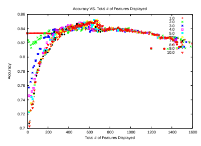

(a) K-Means on news-diff dataset: (x axis-f) vs. (y axis-f_{total})

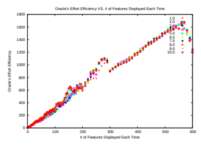

(b) K-Means on news-diff dataset: (x axis-f) vs. (y axis-eff-eff)

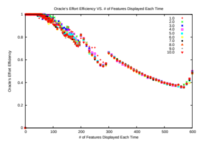

(c) K-Means on news-diff dataset: (x axis-f) vs. (y axis-NMI)

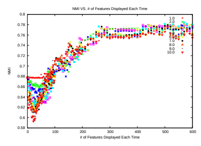

(d) K-Means on news-diff dataset: (x axis-f_{total}) vs. (y axis-NMI)

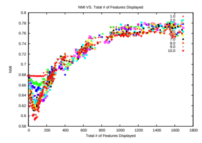

(e) K-Means on news-diff dataset: (x axis-f) vs. (y axis-$Accuracy$)

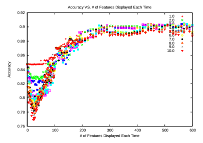

(f) K-Means on news-diff dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

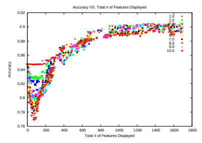

(g) EM-NB on news-diff dataset: (x axis-f) vs. (y axis-f_{total})


(h) EM-NB on news-diff dataset: (x axis-f) vs. (y axis-eff-eff)


(i) EM-NB on news-diff dataset: (x axis-f) vs. (y axis-NMI)

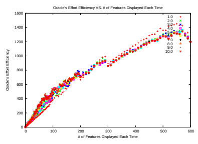

(j) EM-NB on news-diff dataset: (x axis-f_{total}) vs. (y axis-NMI)

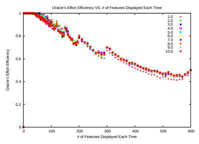

(k) EM-NB on news-diff dataset: (x axis-f) vs. (y axis-$Accuracy$)

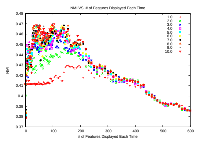

(l) EM-NB on news-diff dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

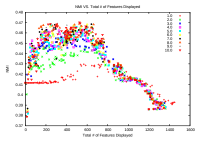
A.4 Effect of User Effort on News-Related Dataset

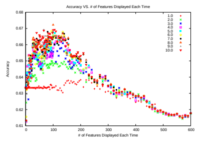
Figure 10: Effect of user effort on news-related dataset. (a), (b), (c), (d), (e) and (f) use K-Means as the underlying algorithm while (g), (h), (i), (j), (k) and (l) use EM-NB as the underlying algorithm. Legends are all weights between 1 and 10.

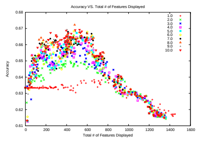

(a) K-Means on news-related dataset: (x axis-f) vs. (y axis-f_{total})

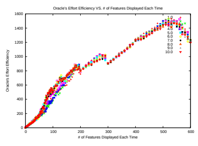

(b) K-Means on news-related dataset: (x axis-f) vs. (y axis-eff-eff)

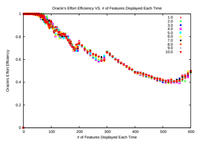

(c) K-Means on news-related dataset: (x axis-f) vs. (y axis-NMI)

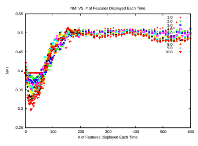

(d) K-Means on news-related dataset: (x axis-f_{total}) vs. (y axis-NMI)

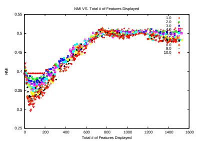

(e) K-Means on news-related dataset: (x axis-f) vs. (y axis-$Accuracy$)

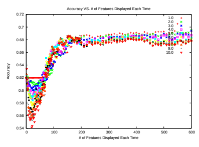

(f) K-Means on news-related dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

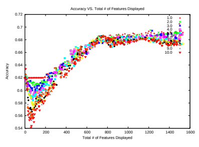

(g) EM-NB on news-related dataset: (x axis-f) vs. (y axis-f_{total})


(h) EM-NB on news-related dataset: (x axis-f) vs. (y axis-eff-eff)


(i) EM-NB on news-related dataset: (x axis-f) vs. (y axis-NMI)

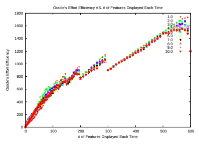

(j) EM-NB on news-related dataset: (x axis-f_{total}) vs. (y axis-NMI)


(k) EM-NB on news-related dataset: (x axis-f) vs. (y axis-$Accuracy$)

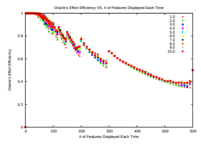

(l) EM on news-related dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

A.5 Effect of User Effort on News-Similar Dataset

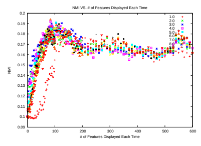
Figure 11: Effect of user effort on news-similar dataset. (a), (b), (c), (d), (e) and (f) use K-Means as the underlying algorithm while (g), (k), (l), (m), (k) and (l) use EM-NB as the underlying algorithm. Legends are all weights between 1 and 10.



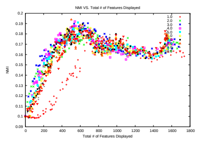
(a) K-Means on news-similar dataset: (x axis-f) vs. (y axis-f_{total})



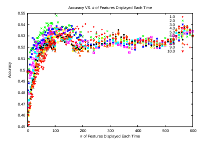
(b) K-Means on news-similar dataset: (x axis-f) vs. (y axis-eff-eff)



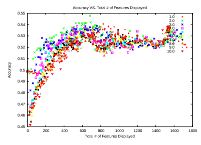
(c) K-Means on news-similar dataset: (x axis-f) vs. (y axis-NMI)



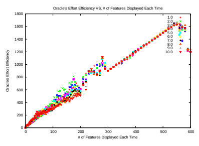
(d) K-Means on news-similar dataset: (x axis-f_{total}) vs. (y axis-NMI)



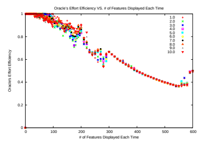
(e) K-Means on news-similar dataset: (x axis-f) vs. (y axis-$Accuracy$)



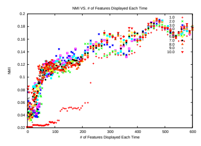
(f) K-Means on news-similar dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)



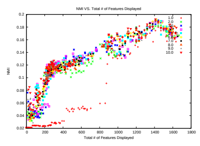
(g) EM-NB on news-similar dataset: (x axis-f) vs. (y axis-f_{total})



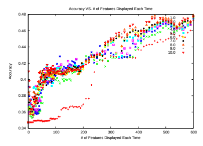
(h) EM-NB on news-similar dataset: (x axis-f) vs. (y axis-eff-eff)



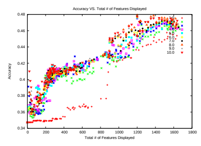
(i) EM-NB on news-similar dataset: (x axis-f) vs. (y axis-NMI)



(j) EM-NB on news-similar dataset: (x axis-f_{total}) vs. (y axis-NMI)



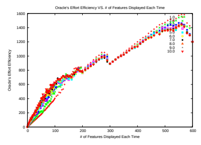
(k) EM-NB on news-similar dataset: (x axis-f) vs. (y axis-$Accuracy$)



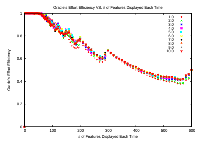
(l) EM on news-similar dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

A.6 Effect of User Effort on ACM (D2-D2&D3-D3) Dataset

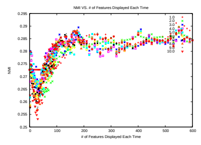
Figure 12: Effect of user effort on D2-D2&D3-D3 dataset. (a), (b), (c), (d), (e) and (f) use K-Means as the underlying algorithm while (g), (h), (i), (j), (k) and (l) use EM-NB as the underlying algorithm. Legends are all weights between 1 and 10.



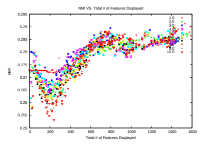
(a) K-Means on D2-D2&D3-D3 dataset: (x axis-f) vs. (y axis-f_{total})



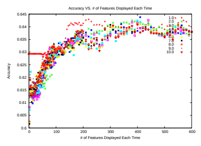
(b) K-Means on D2-D2&D3-D3 dataset: (x axis-f) vs. (y axis-eff-eff)



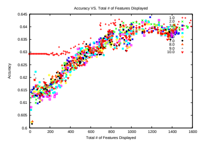
(c) K-Means on D2-D2&D3-D3 dataset: (x axis-f) vs. (y axis-NMI)



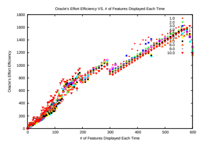
(d) K-Means on D2-D2&D3-D3 dataset: (x axis-f_{total}) vs. (y axis-NMI)



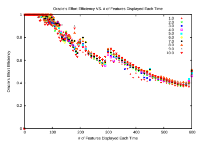
(e) K-Means on D2-D2&D3-D3 dataset: (x axis-f) vs. (y axis-$Accuracy$)



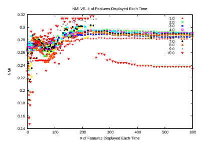
(f) K-Means on D2-D2&D3-D3 dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)



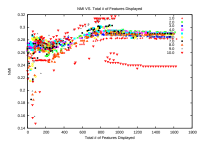
(g) EM-NB on D2-D2&D3-D3 dataset: (x axis-f) vs. (y axis-f_{total})



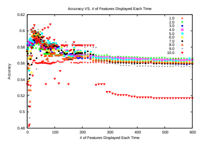
(h) EM-NB on D2-D2&D3-D3 dataset: (x axis-f) vs. (y axis-eff-eff)



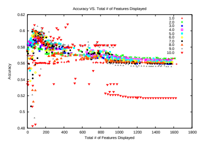
(i) EM-NB on D2-D2&D3-D3 dataset: (x axis-f) vs. (y axis-NMI)



(j) EM-NB on D2-D2&D3-D3 dataset: (x axis-f_{total}) vs. (y axis-NMI)



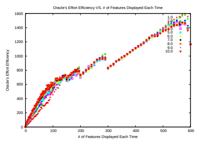
(k) EM-NB on D2-D2&D3-D3 dataset: (x axis-f) vs. (y axis-$Accuracy$)

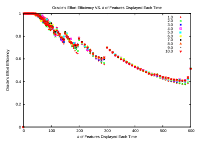


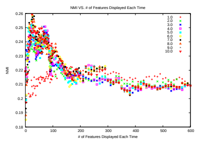
(l) EM on D2-D2&D3-D3 dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

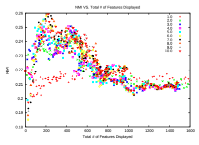
A.7 Effect of User Effort on Acm (D-H-I) Dataset

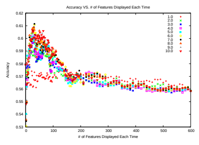
Figure 13: Effect of user effort on ACM (D-H-I) dataset. (a), (b), (c), (d), (e) and (f) use K-Means as the underlying algorithm while (g), (h), (i), (j), (k) and (l) use EM-NB as the underlying algorithm. Legends are all weights between 1 and 10.

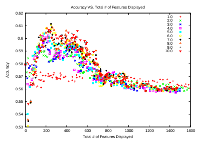

(a) K-Means on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-f_{total})

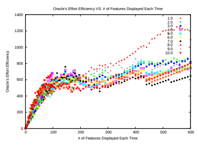

(b) K-Means on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-eff-eff)

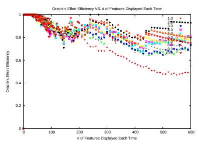

(c) K-Means on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-NMI)

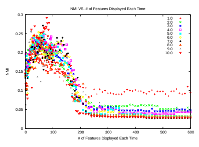

(d) K-Means on ACM (D-H-I) dataset: (x axis-f_{total}) vs. (y axis-NMI)

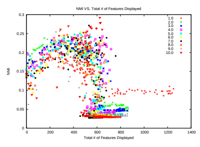

(e) K-Means on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-$Accuracy$)

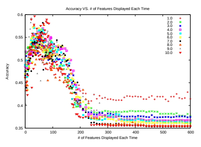

(f) K-Means on ACM (D-H-I) dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

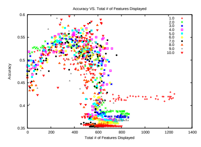

(g) EM-NB on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-f_{total})


(h) EM-NB on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-eff-eff)


(i) EM-NB on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-NMI)

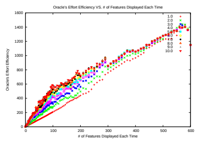

(j) EM-NB on ACM (D-H-I) dataset: (x axis-f_{total}) vs. (y axis-NMI)


(k) EM-NB on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-$Accuracy$)

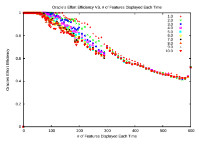

(l) EM-NB on ACM (D-H-I) dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

A.8 Effect of User Effort on 3-classic Dataset

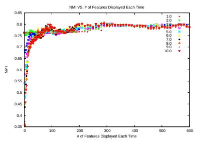
Figure 14: Effect of user effort on 3-classic dataset. (a), (b), (c), (d), (e) and (f) use K-Means as the underlying algorithm while (g), (h), (i), (j), (k) and (l) use EM-NB as the underlying algorithm. Legends are all weights between 1 and 10.



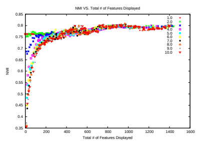
(a) K-Means on 3-classic dataset: (x axis-f) vs. (y axis-f_{total})



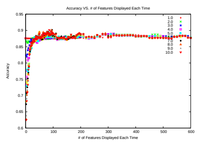
(b) K-Means on 3-classic dataset: (x axis-f) vs. (y axis-eff-eff)



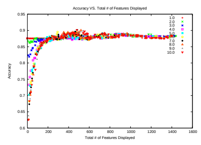
(c) K-Means on 3-classic dataset: (x axis-f) vs. (y axis-NMI)



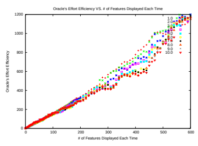
(d) K-Means on 3-classic dataset: (x axis-f_{total}) vs. (y axis-NMI)



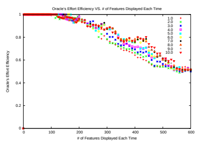
(e) K-Means on 3-classic dataset: (x axis-f) vs. (y axis-$Accuracy$)



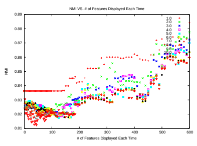
(f) K-Means on 3-classic dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)



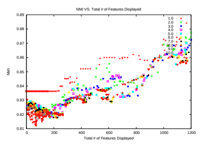
(g) EM-NB on 3-classic dataset: (x axis-f) vs. (y axis-f_{total})



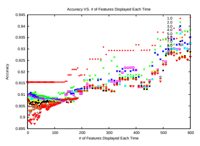
(h) EM-NB on 3-classic dataset: (x axis-f) vs. (y axis-eff-eff)



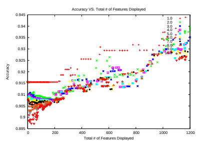
(i) EM-NB on 3-classic dataset: (x axis-f) vs. (y axis-NMI)



(j) EM-NB on 3-classic dataset: (x axis-f_{total}) vs. (y axis-NMI)



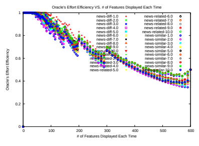
(k) EM-NB on 3-classic dataset: (x axis-f) vs. (y axis-$Accuracy$)



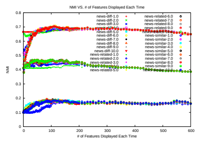
(l) EM-NB on 3-classic dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

A.9 Comparison the Same Algorithm on Different Newsgroups Datasets

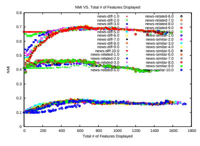
Figure 15: DCIFS (Algorithm 5) with the same underlying algorithm on news-diff, news-related, news-similar datasets. Legends are all weights between 1 and 10 for all newsgroups datasets.



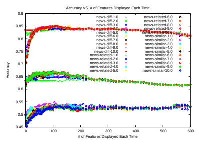
(a) K-Means on newsgroups datasets: (x axis-f) vs. (y axis-eff-eff)



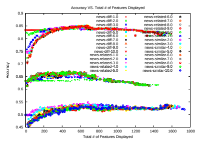
(b) K-Means on newsgroups dataset: (x axis-f) vs. (y axis-NMI)



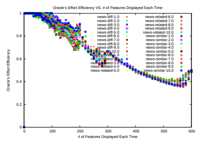
(c) K-Means on newsgroups dataset: (x axis-f_{total}) vs. (y axis-NMI)



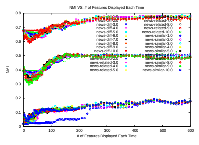
(d) K-Means on newsgroups dataset: (x axis-f) vs. (y axis-$Accuracy$)



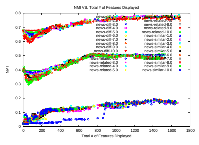
(e) K-Means on newsgroups dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)



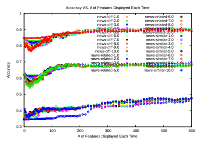
(f) EM-NB on newsgroups datasets: (x axis-f) vs. (y axis-eff-eff)



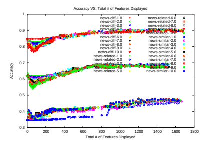
(g) EM-NB on newsgroups dataset: (x axis-f) vs. (y axis-NMI)



(h) EM-NB on newsgroups dataset: (x axis-f_{total}) vs. (y axis-NMI)



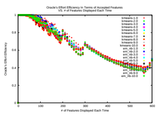
(i) EM-NB on newsgroups dataset: (x axis-f) vs. (y axis-$Accuracy$)

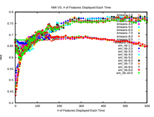


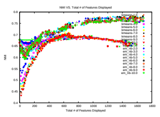
(j) EM-NB on newsgroups dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

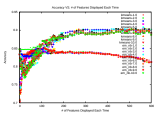
A.10 Comparison Between K-Means and EM-NB on the Same Newsgroups Datasets

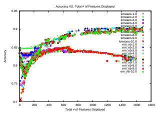
Figure 16: DCIFS (Algorithm 5) with different underlying algorithms on the same newsgroups datasets. Legends are all weights between 1 and 10 for both K-Means and EM-NB.

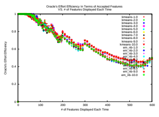

(a) K-Means and EM-NB on news-diff dataset: (x axis-f) vs. (y axis-eff-eff)

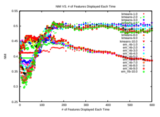

(b) K-Means and EM-NB on news-diff dataset: (x axis-f) vs. (y axis-NMI)

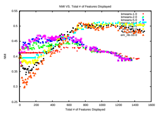

(c) K-Means and EM-NB on news-diff dataset: (x axis-f_{total}) vs. (y axis-NMI)

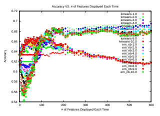

(d) K-Means and EM-NB on news-diff dataset: (x axis-f) vs. (y axis-$Accuracy$)

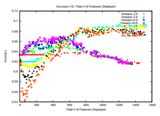

(e) K-Means and EM-NB on news-diff dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

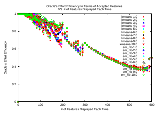

(f) K-Means and EM-NB on news-related dataset: (x axis-f) vs. (y axis-eff-eff)

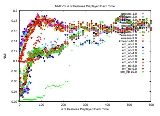

(g) K-Means and EM-NB on news-related dataset: (x axis-f) vs. (y axis-NMI)

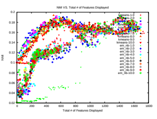

(h) K-Means and EM-NB on news-related dataset: (x axis-f_{total}) vs. (y axis-NMI)

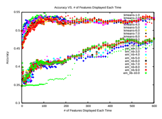

(i) K-Means and EM-NB on news-related dataset: (x axis-f) vs. (y axis-$Accuracy$)

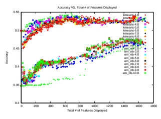

(j) K-Means and EM-NB on news-related dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)


(k) K-Means and EM-NB on news-similar dataset: (x axis-f) vs. (y axis-eff-eff)


(l) K-Means and EM-NB on news-similar dataset: (x axis-f) vs. (y axis-NMI)

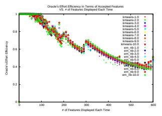

(m) K-Means and EM-NB on news-similar dataset: (x axis-f_{total}) vs. (y axis-NMI)

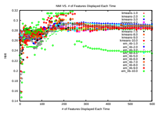

(n) K-Means and EM-NB on news-similar dataset: (x axis-f) vs. (y axis-$Accuracy$)

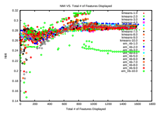

(o) K-Means and EM-NB on news-similar dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

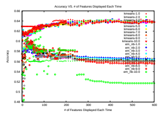
A.11 Comparison Between K-Means and EM-NB on the Same ACM or 3-classic Datasets

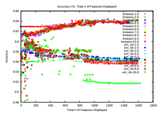
Figure 17: DCIFS (Algorithm 5) with different underlying algorithms on the same newsgroups datasets. Legends are all weights between 1 and 10 for both K-Means and EM-NB.

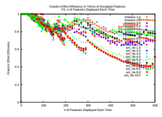

(a) K-Means and EM-NB on ACM (D2-D2&D3-D3) dataset: (x axis-f) vs. (y axis-eff-eff)

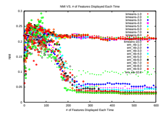

(b) K-Means and EM-NB on ACM (D2-D2&D3-D3) dataset: (x axis-f) vs. (y axis-NMI)

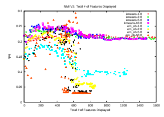

(c) K-Means and EM-NB on ACM (D2-D2&D3-D3) dataset: (x axis-f_{total}) vs. (y axis-NMI)

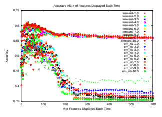

(d) K-Means and EM-NB on ACM (D2-D2&D3-D3) dataset: (x axis-f) vs. (y axis-$Accuracy$)

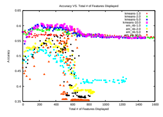

(e) K-Means and EM-NB on ACM (D2-D2&D3-D3) dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

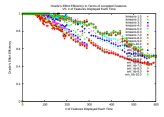

(f) K-Means and EM-NB on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-eff-eff)

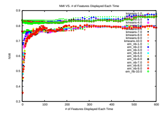

(g) K-Means and EM-NB on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-NMI)

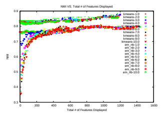

(h) K-Means and EM-NB on ACM (D-H-I) dataset: (x axis-f_{total}) vs. (y axis-NMI)

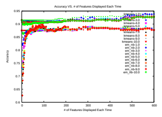

(i) K-Means and EM-NB on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-$Accuracy$)

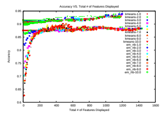

(j) K-Means and EM-NB on ACM (D-H-I) dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)


(k) K-Means and EM-NB on 3-classic dataset: (x axis-f) vs. (y axis-eff-eff)


(l) K-Means and EM-NB on 3-classic dataset: (x axis-f) vs. (y axis-NMI)


(m) K-Means and EM-NB on 3-classic dataset: (x axis-f_{total}) vs. (y axis-NMI)


(n) K-Means and EM-NB on 3-classic dataset: (x axis-f) vs. (y axis-$Accuracy$)


(o) K-Means and EM-NB on 3-classic dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)