Intelligent Data Analysis 18 (2014) 561–581
DOI 10.3233/IDA-140658
IOS Press

Interactive document clustering with feature supervision through reweighting[1]

Yeming Hu[a,*], Evangelos E. Milios[a] and James Blustein[a,b]
[a]*Faculty of Computer Science, Dalhousie University, Halifax, Nova Scotia, Canada*
[b]*School of Management, Dalhousie University, Halifax, Nova Scotia, Canada*

Abstract. Unsupervised document clustering groups documents into clusters without any user effort. However, the clusters produced are often found not in accord with user's perception of the document collection. In this paper we describe a novel framework and explore whether clustering performance can be improved by including user supervision at the feature level. Unlike existing semi-supervised clustering methods, which ask the user to label documents, this framework interactively asks the user to label features. The proposed method ranks all features based on the recent clusters using cluster-based feature selection and presents a list of highly ranked features to the user for labeling. The feature set for the next clustering iteration includes both features accepted by the user and other highly ranked features. The experimental results on several real datasets demonstrate that the feature set obtained using the new interactive framework can produce clusters that better match the user's expectations compared with the unsupervised version of the methods. Moreover, we quantify and evaluate the effect of reweighting previously accepted features and of user effort. Different underlying clustering algorithms such as K Means and Multinomial Naïve Bayes model are demonstrated to perform very well with the newly proposed framework.

Keywords: Interactive clustering, interactive feature selection, user supervision, feature supervision, feature reweighting

1. Introduction

Traditional document clustering is an unsupervised classification of a given document collection into clusters so that the documents within the same cluster are more topically similar than those in different clusters. Such methods work by either (a) optimizing some loss function, such as K-Means [6], over all document assignments or (b) fitting a probabilistic model, such as the multinomial naïve Bayes model [19] onto the document collection. Unsupervised processes minimize user effort during clustering and output a universal set of potential clusters. However, different users usually have their own view of a given document collection and the universal set of potential clusters do not necessarily reflect the user's perception of the document collection [14].

In this paper, we seek to determine whether clusters better matching user expectation may be generated with some supervision by the user. User supervision can be used in the two components of clustering: in the algorithm itself and in the representation of the documents to be clustered. Semi-supervised clustering methods employ user-provided constraints between documents such as "must-link" and "cannot-link" to modify the clustering algorithm by changing either the loss function or the probabilistic model.

*Corresponding author: Yeming Hu, Faculty of Computer Science, Dalhousie University, 6050 University Avenue, Halifax, Nova Scotia, Canada. Tel.: +1 902 494 7111; Fax: +1 902 492 1517; E-mail: yeming@cs.dal.ca.

[1]Short version of this paper appears in [13].

Through optimizing the constrained loss function or forming the probabilistic model with constraints, the user expectation is reflected in the clustering algorithm and finally in the generated clusters. Besides constraining the clustering algorithm, user supervision can also be used to achieve a document representation that is more in accord with the user's view. The user can influence the document representations by selecting the feature set to represent the documents. Document category information, which is not available in document clustering setting, is required for an effective feature selection. However, the user can also give feedback at the feature level. Therefore, instead of asking the user to label enough documents for an effective feature selection, we ask the user to directly label features for clustering, which has been shown to cost less time than labeling documents [20].

In this paper, we explore how user supervision performs when it is used for feature selection. The work is different from previous semi-supervised clustering approaches as it asks the user to label features instead of documents, and the supervision takes the form of selecting features from a list rather than labeling document constraints. Traditional semi-supervised clustering algorithms and our framework perform at different supervision levels, i.e., document-level and feature-level respectively. Their performance is not directly comparable because it is difficult to establish a common quantification of user effort, when the user labels features versus documents. A key benefit of labeling features is that it may take less time than labeling documents as reported in the active learning setting [20].

An overview of our framework is as follows. We first obtain document clusters using the current feature set. Then, cluster-based feature selection is performed based on the obtained clusters serving as the classes, generating a ranked list of features. We present the top f features in the ranked list to the user for labeling. The user must label every feature as "accept" or "don't know" according to their understanding of the document collection. The features the user labels as "accept" and a certain number of highly ranked features are used for the new representations of the documents. The clustering algorithm iterates using the new document representations. In this framework, the user is always presented with a number of features based on the recent clusters. The ranking of the features changes at each iteration. In our framework we try to present the features which are the most promising to be accepted by the user so that the user is asked to label as few features as possible.

Our framework is related to the paradigm of active learning (AL) in the document classification setting. It differs from the interactive feature selection framework proposed in the following ways. First, AL is normally used with document classification algorithms but our framework performs in the document clustering setting. Compared to a document clustering algorithm, a classification algorithm requires labeled documents for training a classifier. Second, the user labels documents in AL but they label features in our framework. Third, uncertain sampling [17] is used in AL to find the most uncertain document for labeling at each iteration. However, cluster-based feature selection is used to locate a list of the most promising features for labeling.

To explore whether user supervision at the feature level can generate clusters better matching user expectation, we propose an interactive framework for feature selection, in which the feature set obtained from the interactive feature selection is used for clustering. This framework includes several components: an underlying clustering algorithm, unsupervised feature selection, cluster-based feature selection, and user supervision. We use this framework to select the features for producing clusters and evaluate whether the generated clusters better conform to user expectation. We also use this framework to evaluate and quantify the effect of feature reweighting and user effort in terms of labeling features. In our study, we use a simulated user instead of the human user for practicality. The simulated user labels features based on document labels (see Section 4.5 for details). In addition, both of a simulated user and a human user may make mistakes. More importantly, the simulated user can be employed repeatedly.

In this paper, we use K Means and Multinomial Naïve Bayes model as the underlying clustering algorithms. However, we believe that other clustering algorithms also work because our interactive feature selection framework does not depend on any specific algorithm. In addition, we use unsupervised mean-*TFIDF* feature selection and χ^2, cluster-based feature selection method. We conducted experiments on some real-word datasets to investigate: (1) the effectiveness of the proposed framework, and (2) the effect of feature reweighting and user effort in terms of labeling features.

The rest of this paper is organized as follows. Section 2 introduces the related work. Section 3 briefly describes the background knowledge for our work. In Section 4, we present the interactive framework for clustering and feature selection. Specially, cluster-based feature selection based on clusters is described in detail. Details of the experimental Evaluation are given in Section 5. In Section 6, we try to give some suggestions for designing an interactive document clustering tool. We conclude with a discussion of the implications of this work and the opportunities for further investigations in Section 7.

2. Related work

Existing semi-supervised clustering makes use of user supervision in the form of document-level constraints. Those methods are generally grouped into four categories. First, constraints are used to modify the optimization of the loss function [16] or estimation of parameters [4]. Second, cluster seeds are derived from the constraints to initialize the cluster centroids [3]. Third, constraints are employed to learn adaptive distance using metric learning techniques [2,7]. Finally, the original high-dimensional feature space can be projected into low-dimensional feature subspaces guided by constraints [23]. In this paper, the user is asked to give feedback at the feature level instead of the document level. Except active learning of document constraints such as [15], most semi-supervised clustering algorithms involve the user supervision outside the clustering process. In this way, all the document constraints are defined before the clustering starts. In our interactive feature selection framework, the user interacts with the clustering process and label the presented features.

Interactive feature selection in the context of active learning is studied in [20], which used linear support vector machine as the base classifier. At each iteration of the active learning, the user is asked to label both the most uncertain document and a list of features. Active learning works in the document classification setting and operates at the document level. It normally uses uncertainty sampling [17] which requires the user to label the document about which the classifier(s) is (are) not certain about. In our framework, we explore the interactive feature selection for document clustering and no document labeling is required.

A set of representative features for each class is labeled in [18]. These features 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 [9] is applied iteratively to build new classifiers. The features are only labeled once for constructing cluster seeds in [18] but the feature set is iteratively updated in our approach for document clustering. Cluster-based feature selection is also performed iteratively in the algorithm proposed in [21]. The main idea of this work is to label a few documents for cluster seeds and for supervised feature selection. It does not involve any user supervision inside the clustering process. There are several class-based feature selection methods such as χ^2, information gain and gain ratio [8], which all use document category information to estimate feature discriminative power. There also exist several unsupervised feature selection methods for document clustering, such as document frequency (DF), mean *TFIDF*, and term frequency variance (TfV) [22]. Despite their simplicity, they are effective in selecting good features. All those techniques are completely unsupervised and employ term (feature)

frequency and/or document frequency to rank features based on their suitability for clustering. Our interactive framework aims to involve the user in the feature selection process.

3. Background

In this section, we present the underlying clustering algorithms and feature selection techniques used in our framework. In our framework, we test one partitioning clustering algorithm and one probabilistic model clustering algorithm, i.e., KMeans 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, we use the χ^2 feature selection technique.

3.1. *KMeans*

KMeans [6] is a very popular clustering algorithm because of its simplicity and efficiency. It clusters data points by locally optimizing a loss function or distortion measure defined as:

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

which represents the sum of the squares of the distances of each data point to its assigned cluster center μ_j. The optimization of J involves finding the assignments $\{r_{ij}\}$ and cluster centroids $\{\mu_j\}$ such that the value of J is minimized. $\{r_{ij}\}$ is defined as

$$r_{ij} = \begin{cases} 1 & \text{if data point } i \text{ is assigned to cluster } j \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

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 KMeans algorithm is illustrated in Algorithm 1. The time complexity of KMeans is $O(IKNM)$, where I is the number of iterations that KMeans runs until convergence, K is the number of clusters, N is the number of data points, and M is the dimensionality of the data points. Since the time complexity is linear to the parameters, i.e., I, K, N, M, KMeans and its variants (KMeans based methods) are very efficient.

3.2. *Multinomial Naïve Bayes model*

Multinomial Naïve Bayes model [19] 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 $\mathcal{D}$ and $|\mathcal{D}|$ is the size of $\mathcal{D}$. 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 $\mathcal{V} = \{w_1, w_2, \ldots, w_{|\mathcal{V}|}\}$. 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, the posterior probability $P(c_j|d_i)$ has to be computed from the prior probability and the word conditional probability. Based on Bayesian probability and the multinomial model, we have the prior probability

$$p(c_j) = \frac{\sum_{i=1}^{|\mathcal{D}|} P(c_j|d_i)}{|\mathcal{D}|} \tag{3}$$

Algorithm 1 KMeans [6]

Input: Data point vectors $\{d_1, d_2, \ldots, d_N\}$, seed s for random initialization of the cluster centroids $\{\mu_1, \mu_2, \ldots, \mu_K\}$
Output: Data point assignments $\{r_{ij}\}$
Method:

1: Randomly initialize the cluster centroids $\{\mu_j\}$ 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 μ_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 $\{\mu_j\}$ based on the new data point assignments $\{r_{ij}\}$
8: **until** No data point assignments change or maximum # of iterations is reached

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

$$p(w_t|c_j) = \frac{1 + \sum_{i=1}^{|\mathcal{D}|} N(w_t, d_i) \cdot P(c_j|d_i)}{|\mathcal{V}| + \sum_{s=1}^{|\mathcal{V}|} \sum_{i=1}^{|\mathcal{D}|} N(w_s, d_i) \cdot P(c_j|d_i)} \tag{4}$$

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:

$$P(c_j|d_i) = \frac{P(c_j)P(d_i|c_j)}{\sum_{r=1}^{|C|} P(c_r)P(d_i|c_r)} = \frac{P(c_j)\prod_{k=1}^{|d_i|} P(w_{d_i,k}|c_j)}{\sum_{r=1}^{|C|} P(c_r)\prod_{k=1}^{|d_i|} P(w_{d_i,k}|c_r)} \tag{5}$$

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 distributions in the clusters.

The Multinomial Naïve Bayes clustering algorithm, also called *EM-NB* algorithm, is formed by applying EM algorithm [9] to Naïve Bayes classifier. In *EM-NB* algorithm, Eqs (3) and (4) are evaluated in the M step and Eq. (5) is evaluated in the E step. The initial $p(c_j|d_i)$ can be derived from clusters obtained from KMeans. In this case, the value of $p(c_j|d_i)$ is 1 when d_i is in cluster c_j. Otherwise, the value is 0. The initial $p(c_j|d_i)$ can also be obtained from *EM-NB* itself, in which case its value is between 0 and 1.

3.3. Mean-TFIDF feature selection technique

Mean-*TFIDF* feature selection technique [22] 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 $j(ft_j)$ is defined as $idf_j = \log \frac{|\mathcal{D}|}{|\{d: ft_j \in d\}|}$ where $\mathcal{D}$ denotes the document collection, $n_{(i,j)}$ denotes occurrences of term i in document j and d denotes a document in $\mathcal{D}$. 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 $mean\text{-}TFIDF_j = \frac{\sum_i TFIDF_{(i,j)}}{|\mathcal{D}|}$.

Algorithm 2 *EM-NB*, i.e. Multinomial Naïve Bayes [19]

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

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

3.4. χ^2 class-based feature selection technique

The χ^2 value of a feature indicates whether the feature is significantly correlated with a class [21]. Larger values indicate higher correlation. Basically, the χ^2 test aggregates the deviations of the measured probabilities from the expected probabilities assuming independence. Assuming random variable $C \in \{0, 1\}$ denotes class and random variable $I \in \{0, 1\}$ denotes existence of feature j, the χ^2 value of the feature j defined as $\chi^2 = \sum_{c,i} \frac{[k_{c,i} - nPr(C=c)\cdot Pr(I=i)]^2}{nPr(C=c)\cdot Pr(I=i)}$ 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 where 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. Methodology

In this section, we first explain the algorithm at a high level. Then we introduce the interactive clustering and feature selection frameworks. At the same time, we present an approach to investigate the effect of user effort, and cluster evaluation measures. We also give details about the simulated user.

Our clustering framework with interactive feature selection is summarized as follows:

Table 1
Definition of variables

Variable	Definition
s	Seed for the randomization of KMeans cluster centroids $\{\mu_j\}$
m	Size of feature set for document clustering
f	Number of features presented to the user at each iteration
y^c	Recently generated clusters
$\{r_{ij}\}$	Assignment of document i to cluster j
FS^m	Feature set selected for next clustering iteration
FS_{basic}	All features extracted
$FS^t_{accepted}$	Set of features accepted until iteration t
g	The weight for accepted features in $FS^t_{accepted}$
$\{d_1, d_2, \ldots, d_N\}$	Document collection $\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$

1. Initialization.
   (a) Perform unsupervised feature selection and take the top m ranked features to represent the documents.
   (b) Perform the underlying clustering algorithm using the feature set obtained in step 1a and obtain clusters of documents.
2. Interactive feature selection and clustering.
   (a) Perform cluster-based feature selection based on the recent clusters.
   (b) Perform interactive feature selection.
   (c) Perform the underlying clustering algorithm using the feature set obtained in step 2b and obtain new clusters of documents.
   (d) Stop if no document membership changes. Otherwise, go to step 2a.

4.1. Notations

Notations used in this paper are summarized in Table 1. Particularly, m denotes the size of feature set used for document clustering while f denotes the number of features presented to a user at each iteration. and FS^m denotes a feature set with size of m, the value of which can be changed by the user.

4.2. Interactive document clustering framework

After a new feature set with user supervision is obtained (as described in Algorithm 4), the documents are re-clustered using this new feature set. During the re-clustering, the accepted features may be given higher weights. The algorithm for interactive document clustering based on interactive feature selection is given in Algorithm 3. At the beginning, clusters are obtained from traditional KMeans with the feature

Algorithm 3 Interactive Document Clustering Framework with Feature Selection (Notations also in Table 1)

Input: $\{d_1, d_2, \ldots, d_N\}$ – document vectors, s – seed for randomization, f – # of features presented to the user at each iteration, g – the weight for accepted features, m – size of feature set for document clustering, FS_{basic} – all features extracted.
Output: $\{r_{ij}\}$ – assignments of document to clusters, i.e. final clusters.
Method:

1: Obtain an initial set of clusters $y^c_{initial}$ using the underlying algorithm (KMeans or *E*M-NB) with given seed s and feature set selected by unsupervised feature selection, e.g., mean-*TFIDF*
2: $y^c \leftarrow y^c_{initial}$
3: $t \leftarrow 0$
4: $FS^0_{accepted} \leftarrow \{\}$
5: **repeat**
6: $t \leftarrow t + 1$
7: Perform feature Selection with User Supervision, Algorithm 4
8: Initialize the underlying clustering algorithm with previous iteration's parameters
9: Cluster documents using the new feature set and the initialized underlying clustering algorithm and obtain new clustering y^c_{new} and data point assignments $\{r_{ij}\}$
10: $y^c \leftarrow y^c_{new}$
11: **until** No data point assignment changes or maximum # of iterations is reached or the user chooses to terminate

set selected by mean-*TFIDF* [22]. There are no user accepted features at the beginning. It is worth noting that the feature set can be constructed automatically without user supervision by setting f to 0. In addition, the clustering process can terminate at any time (1) when the user chooses to stop, or (2) when the generated clusters do not change, or (3) when the maximum number of iterations is reached. The user may choose to stop when either generated clusters or the feature set is satisfactory.

4.3. Interactive feature selection framework

The high dimensionality of the document text reduces the clustering algorithm performance. Feature selection can alleviate this problem and generate a feature set which is easily interpreted by the user. This is one of the motivations for inviting the user to label features during clustering. At each iteration, the features presented to the user for confirmation are the top f features ranked by cluster-based feature selection, e.g. the χ^2, treating the most recent clusters as classes. The user gives one of the "accept" or "don't know" answers when a feature is presented. If the feature is believed to be useful for discriminating among clusters, the user will give answer "accept"; otherwise, an answer "don't know" is given. The algorithm that incorporates feature supervision for feature selection is presented in Algorithm 4. Note that "don't know" features might be displayed again at later iterations since the user might confirm the features after he sees more other features and therefore knows more about the topics. All features accepted by the user will be included in the feature set for next clustering iteration. The remaining features, up to the total number m (a fixed number given by user) of features for clustering, are selected according to the ranking obtained by the cluster-based feature selection based on the most recent clusters.

Algorithm 4 Interactive feature selection with user supervision (notations also defined in Table 1)

Input:
m – size of feature set for document clustering,
f – # of features presented to the user each time,
$FS^{t-1}_{accepted}$ – set of features accepted until $t-1$ iteration,
FS_{basic} – all features extracted,
y^c – intermediate clusters.

Output:
$FS^{t}_{accepted}$ – set of features accepted until t iteration,
FS^m – m features selected for next clustering iteration.

Method:

1: $FS^{t}_{accepted} \leftarrow FS^{t-1}_{accepted}$
2: $FL^{all} \leftarrow$ Ranked list of features in FS_{basic} by cluster-based feature selection, e.g. the χ^2, based on y^c
3: {*//accepted features and "don't know" features are presented to the user only once and multiple times respectively}*
4: $FL = FL^{all} - FS^{t-1}_{accepted}$
5: ***for all*** $i = 1$ *to* f ***do***
6: *Present* i^{th} *feature in* FL *to the user, get* $reply$
7: ***if*** $reply ==$ "$accept$" ***then***
8: *Add* i^{th} *feature into* $FS^{t}_{accepted}$
9: ***end if***
10: ***end for***
11: $FS^m \leftarrow FS^{t}_{accepted}$
12: $size \leftarrow size\ of\ FS^m$
13: ***for*** $i \leftarrow 1$ *to* $m - size$ ***do***
14: $FS^m \leftarrow FS^m \cup \{(f+i)^{\text{th}} feature\}$
15: ***end for***

4.4. Cluster-Based feature selection

When document class labels are available, class-based feature selection can be performed. Such examples are the χ^2, information gain, and gain ratio. In our work, we apply those techniques without human attached labels, by treating clusters as classes. The cluster a document belongs to is treated as the label of the document. We make use of the class-based feature selection and the cluster labels to perform feature selection. To be unambiguous, we call it cluster-based feature selection as there is no user-supervision in the document class labels.

The cluster-based (class-based) feature selection ranks the features according to the corresponding measures [8]. Take the χ^2 as an example and suppose there are K clusters. There is one χ^2 value for each feature t and each cluster c. Therefore, there are K values of the χ^2 for a feature t which we call *local values*. In order to sort the features, we need one global value for each feature. The *global value* can be defined either as the sum of the local values or the maximum of the local values. Since we only use the χ^2 to rank the features and the simulate user, it is not important which definition we choose as long as it is able to rank features based on the underlying document class labels. In our research, we compute a feature's global value as the sum of its local values. More details about the two definitions

Table 2
Definition of feature sets

Feature set	Definition
FS_{basic}	All features extracted, i.e., without doing any feature selection
$FS_{mean\text{-}TFIDF}$	Selected by mean-TFIDF feature selection method
$FS_{iterative}$	Selected by the interactive feature selection framework without user supervision, i.e., f is 0
$FS_{interactive}$	Selected by the interactive feature selection with user supervision
$FS_{reference}$	An optimal feature set selected by the χ^2 test based on true class labels

can be found in [21]. The larger the global value is, the better the feature is in discriminating among clusters.

4.5. The simulated user

In our research, user supervision is used to identify useful features for clustering, namely, feature selection. The user is asked to select good features in the interactive feature selection framework. Our goal in this paper is to compare our interactive framework with unsupervised feature selection. More importantly, we explore whether our interactive framework is significantly better at feature selection than state-of-the-art unsupervised approaches. To test for statistical significance, many runs of the algorithms must be performed, which is very costly in terms of human effort required. Like other text mining research involving users [1,3,4,7,16,23], a simulated user (also called oracle) is employed in this paper. Compared with human users, a simulated user is fast, costs little and is sufficient for an initial proof-of-concept demonstration. At the same time, we realize there are many properties of a human user that an simulated user cannot simulate. For example, different users have different domain expertise, in-depth knowledge of the documents, etc. Therefore, a proper user study should be conducted for the evaluation of the proposed framework after this initial proof-of-concept study.

Based on the document class labels, a ranking of all features is obtained using class-based feature selection and the top m features can be taken to form a reference feature set for user simulation. Then the simulated user works as follows. It gives the answer "accept" if the presented feature is included in the reference feature set. Otherwise, the answer is "don't know". With a simulated user, we can quantify the performance of the clustering algorithm by comparing computed clusters against the underlying class labels, which are though of as the clusters the user is expecting. In the simulated user scenario, the interactive framework terminates when the generated clusters do not change or the maximum number of iterations is reached.

4.6. Feature sets

By using the underlying clustering algorithms, we compare interactive feature selection framework with unsupervised feature selection techniques. Since our framework aims to select a better feature set for clustering, the underlying algorithms with feature sets selected by different methods are compared. The various feature sets are listed in Table 2. All feature sets in Table 2 have the same size $m = 600$ except FS_{basic} whose size depends on the number of all extracted features.

4.7. Effect of user effort

In this section, we investigate the effect of user effort on document clustering. To the best of our knowledge, our work is the first one to do that. A few variables are defined for the analysis. As we know,

f, the number of the features presented to a user at each iteration is given as an input parameter of the interactive feature selection and clustering framework. The f value can be thought of as a measure of effort as f features are confirmed by the user at each iteration. Therefore, total amount of user effort spent in the document clustering depends on the value of f. Suppose r is the number of iterations, then the total number of features inspected is defined as $f_{total} = f \times r$. Out of the f_{total} features inspected, we define $f_{accepted}$ as the number of features accepted by the user. Finally, user effort efficiency *eff* can be defined as:

$$eff = \frac{f_{accepted}}{f_{total}} \tag{6}$$

The larger *eff* is, the larger portion of the presented features is accepted, which may be good for clustering.

4.7.1. Feature reweighting

Since feature reweighting can boost classification performance in active learning [20], feature reweighting is adopted in the interactive clustering framework. Different underlying clustering algorithms have their own method of integrating feature re-weighting. In this paper, we use KMeans and Multinomial Naïve Bayes model or *EM-NB*. For KMeans, the *TFIDF* values of the accepted features is multiplied by the given weight g and then the vector of *TFIDF* values is normalized. In *EM-NB*, the posterior probability of a class is

$$P(c_j|d_i) = \frac{P(c_j)P(d_i|c_j)}{\sum_{r=1}^{|C|} P(c_r)P(d_i|c_r)} = \frac{P(c_j)\prod_{k=1}^{|d_i|} P(w_{d_i,k}|c_j)}{\sum_{r=1}^{|C|} P(c_r)\prod_{k=1}^{|d_i|} P(w_{d_i,k}|c_r)} \tag{7}$$

for a given document [18]. g affects word conditional probability through the feature term frequency:

$$p(w_t|c_j) = \frac{1 + \sum_{i=1}^{|\mathcal{D}|} \boldsymbol{g}_t \cdot N(w_t, d_i) \cdot P(c_j|d_i)}{|\mathcal{V}| + \sum_{s=1}^{|V|}\sum_{i=1}^{|\mathcal{D}|} \boldsymbol{g}_s \cdot N(w_s, d_i) \cdot P(c_j|d_i)} \tag{8}$$

where g_s (the same for g_t) is the weight given to feature w_s in the selected feature set. The weight g_s of a given feature is defined as:

$$|g_s| = \begin{cases} g & \text{if } w_s \text{ is accepted} \\ 1 & \text{otherwise} \end{cases} \tag{9}$$

In our experiments, g is an integer between 1 and 10. Using the above definitions, we investigate how clustering performance and user effort is affected by the values of g, f, f_{total}, and by feature reweighting.

5. Experiments

5.1. Datasets

In this work, we use six datasets to test our newly proposed algorithm: (1) *news-diff-3*, (2) *news-*

Table 3
Legend of ACM categories

ACM category code	ACM category name
D	Software
$D.2$	Software engineering
$D.3$	Programming languages
H	Information systems
I	Computing methodologies

related-3, (3) *news-similar-3*, (4) *D2-D2&D3-D3*, (5) *D-H-I*, and (6) *3-classic-abstract*. The first three datasets are derived from the widely used 20-Newsgroups collection[1] for text classification and clustering. Three reduced datasets, *News-Different-3*, *News-Related-3*, and *News-Similar-3*, are derived according to [4]. *News-Different-3* covers 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 conceptual overlap between groups, it is the most difficult one to cluster.

The fourth and fifth datasets are collections of papers in full text, which were manually collected by the authors from Association for Computing Machinery (ACM) Digital Library.[2] We use the 1998 ACM Computing Classification System to label the categories.[3] In this paper, we use the categories listed in Table 3. H and I are related as they have overlapping areas such as "Data Mining" and "Text Clustering" areas. Two datasets are derived from the ACM paper collection. The first, *D2-D2&D3-D3*, contains papers which are only from category $D2$, from both categories $D2$ and $D3$, and only from the $D3$ category respectively. Each category has 87 papers in this dataset and is related to the others as they are all from D category. The second, *D-H-I*, consists of 100 papers from each of D,H,I categories.

The sixth dataset 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. One hundred 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 bags-of-words.[5] Then, we remove the stop words and stem all other words. The top m features ranked either by mean-*TFIDF* or the χ^2 method are employed for clustering. For the K-Means-based algorithms, a feature vector for each document is constructed with *TFIDF* weighting and then normalized. For EM-NB-based algorithms, the term frequency of the selected features is directly used in the related algorithms.

5.2. Evaluation measures

We use three evaluation measures: (1) Clustering Accuracy, (2) *NMI*, and (3) Jaccard Coefficient.

5.2.1. Clustering accuracy

Assume we have a clustering T and the underlying classes C. To estimate the clustering accuracy, we

[1] http://people.csail.mit.edu/jrennie/20Newsgroups/.
[2] http://portal.acm.org/dl.cfm.
[3] http://www.acm.org/about/class/1998/.
[4] ftp://ftp.cs.cornell.edu/pub/smart.
[5] A word is defined as a sequence of alphabetic characters delimited by non-alphabetic characters.

map each cluster $t \in T$ to one underlying class $c \in C$ if the documents from c dominate t, i.e., the number of documents from c is maximum. 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:

$$\mathit{CACC}(T, C) = \frac{\sum_t n(t)}{\sum_t |t|} = \frac{\sum_t n(t)}{N} \tag{10}$$

where N is the size of the document collection. As it is pointed out in [5], 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. The *CACC* values are in the interval $[0, 1]$. In all our experiments, we set K the same as the number of underlying classes in the datasets.

5.2.2. Normalized mutual information

Normalized mutual information (*NMI*) [10] measures the shared information between the cluster assignments S and class labels L of documents. It is defined as:

$$\mathit{NMI}(S, L) = \frac{I(S, L)}{(H(S) + H(L))/2} \tag{11}$$

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. Assuming 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, we define:

$$H(L) = -\sum_{i=1}^{K} P(l_i) \log_2 P(l_i) \tag{12}$$

$$H(S) = -\sum_{j=1}^{K} P(s_j) \log_2 P(s_j) \tag{13}$$

$$I(S, L) = -\sum_{i=1}^{K} \sum_{j=1}^{K} P(l_i, s_j) \log_2 \frac{P(l_i, s_j)}{P(l_i)P(s_j)} \tag{14}$$

where $P(l_i) = n(l_i)/N$, $P(s_j) = n(s_j)/N$ and $P(l_i, s_j) = n(l_i, s_j)/N$. The *NMI* values are in the interval $[0, 1]$.

5.2.3. Jaccard coefficient

Jaccard coefficient [5] is usually used to measure similarity between two clusterings with no underlying class labels being used. Given two clusterings y_1^c and y_2^c, we define a as the number of document pairs, such that two documents are from the same cluster in both y_1^c and y_2^c, b as 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 as 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 Coefficient between y_1^c and y_2^c is defined as:

$$J(y_1^c, y_2^c) = \frac{a}{a + b + c} \tag{15}$$

Note that the values of Jaccard Coefficient are in the interval $[0, 1]$.

5.3. Experimental setup

In our experiments, we set the number of clusters, i.e. K, to the number of true classes in the datasets. Two underlying algorithms, KMeans and Multinomial Naïve Bayes Model (*EM-NB*) are employed. However, we expect that other clustering algorithms will also work because our interactive framework does not depend on any specific algorithm. We use unsupervised mean-*TFIDF* feature selection and the χ^2 method for the cluster-based feature selection. We first present the results of the underlying algorithms with feature sets selected by different feature selection techniques. Second, we explore the effect of feature set size on document clustering. Third, we study how clustering performance depends on user effort. Fourth, we explore the effect of weight g for feature reweighting. Fifth, we compare KMeans (or EM-NB) with feature supervision on different newsgroup datasets. Finally, we compare KMeans with EM-NB (both with feature supervision) on the same datasets. In this paper, we present a subset of experimental results to illustrate our points. The results for all other datasets have similar patterns.

5.4. Performance on different feature sets

In this section, we compare and discuss the performance of the same underlying algorithm with different feature sets. The feature sets used in Tables 4 and 5 are defined in Table 2. Particularly, "Reference" column gives the performance of the optimal feature set selected based on true class labels while "Iterative" column presents the performance of the feature set obtained using the interactive feature selection framework without user supervision.

Each pair of one underlying algorithm and one feature set was run 36 times[6] with different initializations over all the datasets. In our experiments, we set the size of feature set $m = 600$. The average results are listed in Table 4 for KMeans and Table 5 for *EM-NB*. For the performance of interactive feature set, we take the average performance when the performance stabilizes with the number of features f displayed to the user, e.g. f is between 100 and 300.

As shown in Tables 4 and 5, the interactive feature selection framework can produce better clusters than other unsupervised feature selection methods with some user effort. In these tables, the performance of the clustering algorithms improves significantly[7] in the direction from column FS_{basic} to column $FS_{reference}$ except where the performance measures are bold. In Table 5, the exception is between $FS_{mean-TFIDF}$ and $FS_{iterative}$ including both *NMI* and Clustering Accuracy measures of *news-diff-3* dataset and *news-similar-3* dataset. Although the automatically constructed feature set based on iterations does not always perform better than the unsupervised feature set, the feature set selected with some user supervision does. It is especially true when the automated feature set performs much worse than the unsupervised feature set on *news-similar-3* dataset, user supervision can bring the clustering back to the right track and obtain better performance. Also note that the interactive feature selection and clustering framework with some user effort achieves comparable performance to the underlying algorithm with the reference feature set $FS_{reference}$. The clustering performance of the reference feature set in terms of accuracy and *NMI* might be worse than that of the interactive feature set but is always the best in terms of Jaccard Coefficient. That is because the accuracy and *NMI* are calculated based on the underlying class labels while Jaccard Coefficient is computed based on the clustering produced with the reference feature set. The Jaccard Coefficient of the same two clusterings is 1 according to the definition of Jaccard Coefficient [11].

[6]The number of times we ran the algorithms was randomly chosen to be large enough for computing statistical significance.
[7]Two-tailed T-tests were used for each comparison with $p = 0.05$.

Table 4
Comparison of KMeans with different feature sets defined in Table 2

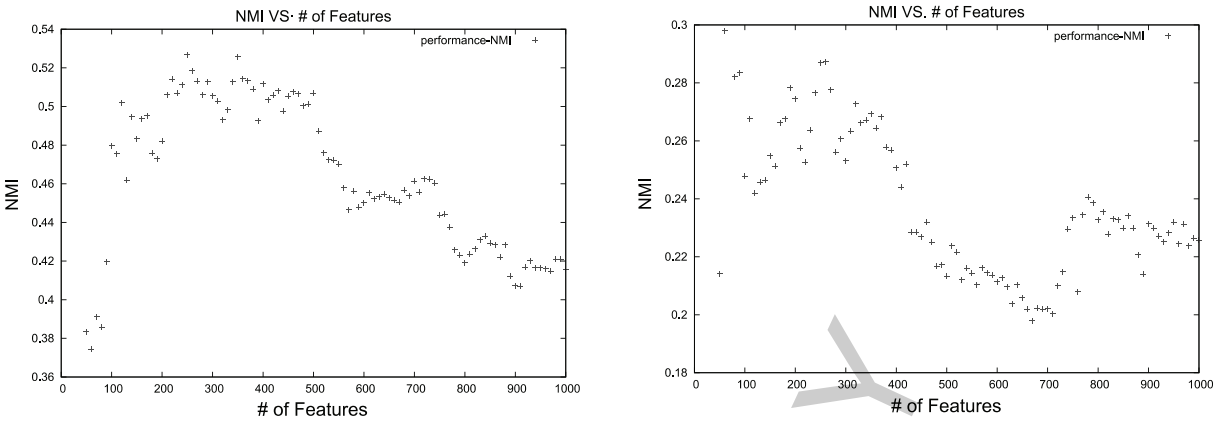
Dataset	Measure	Performance by feature sets				
		Basic	Mean-TFIDF	Iterative	Interactive	Reference
News-diff-3	NMI	0.4051	0.5957	0.6651	0.7084	0.6804
	Accuracy	0.6941	0.7931	0.8335	0.8522	0.8330
	Jaccard coefficient	0.4819	0.6263	0.6476	0.6801	1.0000
News-related-3	NMI	0.1755	0.3341	0.4116	0.4702	0.4501
	Accuracy	0.5285	0.5931	0.6334	0.6722	0.6768
	Jaccard coefficient	0.3748	0.4956	0.5278	0.5570	1.0000
News-similar-3	NMI	0.0380	0.0765	0.1004	0.1938	0.1818
	Accuracy	0.4243	0.4669	0.4988	0.5479	0.5411
	Jaccard coefficient	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 coefficient	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 coefficient	0.4753	0.5673	0.5251	0.6651	1.0000
3-classic-abstract	NMI	0.5779	0.7220	0.7626	0.8079	0.7854
	Accuracy	0.7544	0.8481	0.8755	0.9017	0.8744
	Jaccard coefficient	0.6192	0.7462	0.7801	0.8127	1.0000

Table 5
Comparison of *EM-NB* with different feature sets defined in Table 2

Dataset	Measure	Performance by feature sets				
		Basic	Mean-TFIDF	Iterative	Interactive	Reference
News-diff-3	NMI	0.5267	*0.6742*	*0.6737*	0.7845	0.7879
	Accuracy	0.7622	*0.8474*	*0.8450*	0.9050	0.9034
	Jaccard coefficient	0.5471	0.6867	0.7208	0.8318	1.0000
News-related-3	NMI	0.1966	0.3756	0.3933	0.5227	0.5741
	Accuracy	0.5469	0.6093	0.6150	0.7051	0.7273
	Jaccard coefficient	0.3450	0.5012	0.5257	0.5995	1.0000
News-similar-3	NMI	0.0819	*0.1491*	*0.0259*	0.1925	0.2114
	Accuracy	0.4742	*0.4464*	*0.3481*	0.4793	0.5379
	Jaccard coefficient	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 coefficient	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 coefficient	0.4333	0.5112	0.5419	0.7525	1.0000
3-classic-abstract	NMI	0.6829	0.8182	0.8412	0.8841	0.8960
	Accuracy	0.7946	0.9069	0.9179	0.9439	0.9503
	Jaccard coefficient	0.6683	0.8199	0.8467	0.9069	1.0000

5.5. *Effect of feature set size*

Sometimes, the interactive clustering framework with user interaction obtains better performance than the underlying algorithm with the reference feature set as seen from Tables 4 and 5. The explanation is that the reference feature set is not perfect for clustering, which could be due to several reasons. First, the reference feature set is selected based on the underlying class labels, from which different class-based



(a) *news-related-3* and *K Means* (b) *news-similar-3* and *EM-NB*

Fig. 1. Effect of feature set size with *reference feature set*. Reference feature set with various sizes are used, which are selected by the χ^2 based on the underlying document class labels.

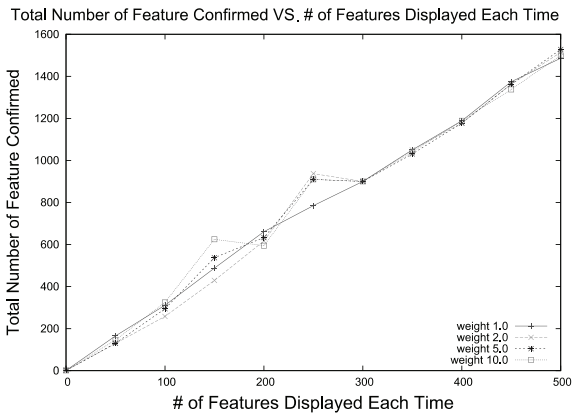
feature selection techniques may produce different rankings of features. Second, the size of feature set m for clustering may affect clustering performance. The above two reasons motivated exploration of the effect of feature set size on document clustering. We do experiments with the reference feature sets with various sizes. As we mentioned before, the reference feature sets are selected by the χ^2 test based on the underlying document class labels. Therefore, we run the base clustering algorithms KMeans and *EM-NB* to cluster documents with the reference feature sets with different sizes. The effect of feature set size in terms of clustering accuracy is illustrated in Figs 1(a) and (b). The performance of both KMeans and *EM-NB* increases initially as the size of feature set gets larger when applied to all datasets. Maximum performance can be reached with different feature set sizes but $200 \leqslant m \leqslant 400$ usually gives the maximum performance. The clustering performance of both KMeans and *EM-NB* on datasets with very different topics across clusters, such as *news-diff-3* dataset and *3-classic-abstract* 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 many more noisy features in the ideal feature set for datasets like *news-similar-3* dataset than others like *news-diff-3*. As more features are added, the "good" features dominate at first but noisy features take over later on. Comparing *EM-NB* (Fig. 1(b)) to KMeans (Fig. 1(a)) on *news-related-3* and *news-similar-3* datasets, we observe that *EM-NB* has smaller performance change than KMeans when noisy features are introduced later on.

5.6. *Effect of user effort*

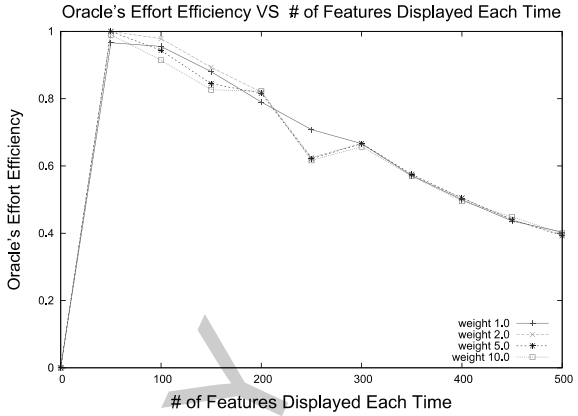
In this section, we study the effect of user effort on clustering performance with feature reweighting.

We only show a portion of the figures we obtained from the experiments here. All the figures can be found in [12]. Effect of user effort on *news-related-3* dataset is shown in Fig. 3(b) for KMeans and Fig. 3(a) for *EM-NB* while effect of user effort on *news-similar-3* dataset is demonstrated in Fig. 4(a) for KMeans and Fig. 4(b) for *EM-NB*.

For all datasets, f_{total}, the user effort spent, increases with f, the number of features presented to the user in each iteration as seen in Fig. 2(a). We also note that the effort efficiency declines when more features displayed in each iteration, as seen in Fig. 2(b). This may be due to the fact that the more features are displayed each time, the higher is the portion of features displayed that are not in the

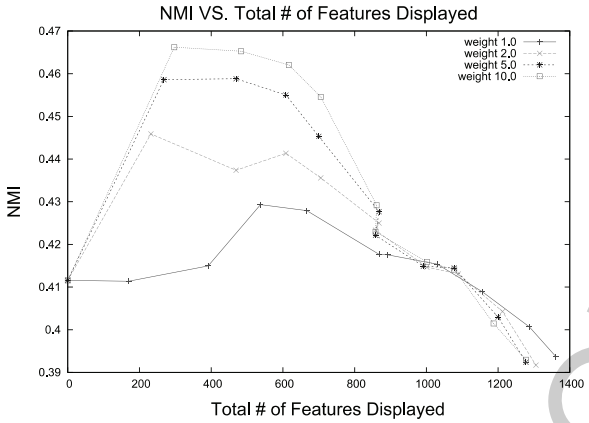


(a) f vs. f_{total}, *EM-NB* on *news-diff-3* Dataset

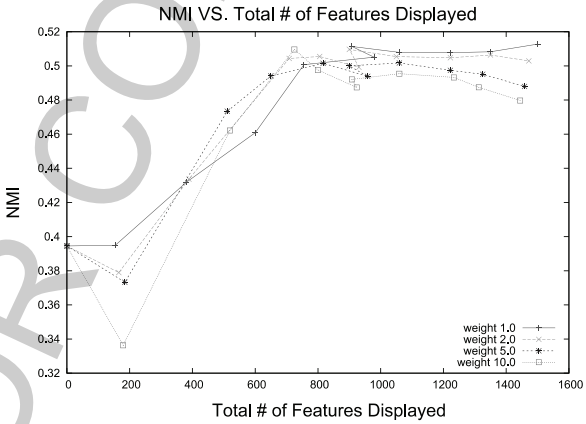


(b) Effort-Efficiency, *EM-NB* on *news-similar-3* Dataset

Fig. 2. f vs. f_{total} and effort efficiency.



(a) KMeans on *news-related-3*



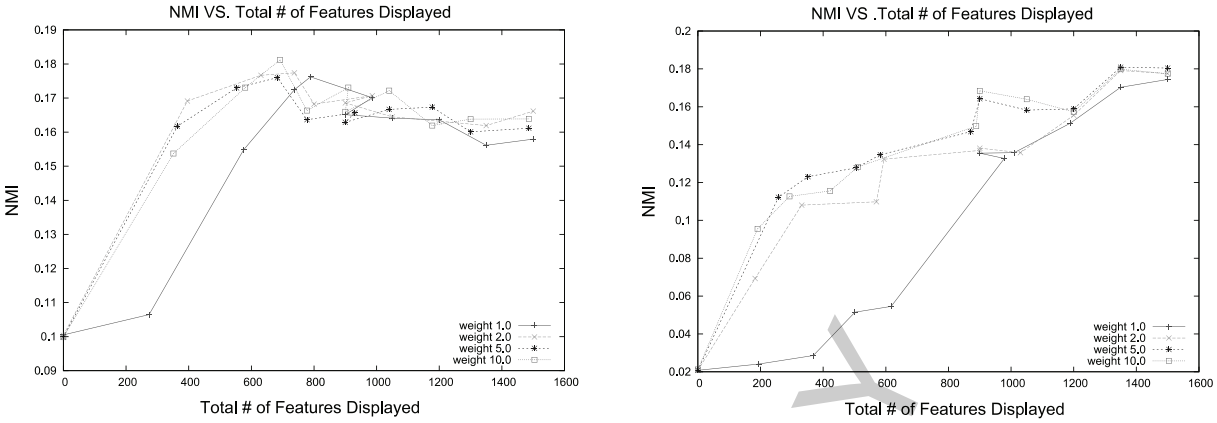
(b) *EM-NB* on *news-related-3*

Fig. 3. Effect of User Effort on *news-related-3*.

reference feature set, i.e., noisy features. The effort efficiency does not always decline due to the fact that the intermediate clusters are noisy and the feature ranking method χ^2 is not perfect for ranking features (see Section 5.5 for more discussion). The observation that the effort efficiency is about 1 when f is small (Fig. 2(b)) implies that most of the displayed features are accepted as useful for clustering by the user. This fact indicates that the most useful features for clustering are ranked very high by the χ^2 based on the intermediate clusters even when the χ^2 is not perfect and the intermediate clusters are noisy.

Generally speaking, the clustering performance improves with more effort provided from the user (Figs 3(a) and 4(a)). However, when the interactive clustering framework with KMeans works with news-related dataset and ACM (*D-H-I*) dataset, the clustering performance declines beyond a certain amount of effort. One possible reason is that the extra effort is used to introduce noisy feature from the reference feature set $FS_{reference}$.

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 of the reference feature set. When weight g is greater than 1 and total accepted features f_{total} is very small, the accepted features could be over-



(a) KMeans on *news-similar-3* (b) *EM-NB* on *news-similar-3*

Fig. 4. Effect of User Effort on *news-similar-3*.

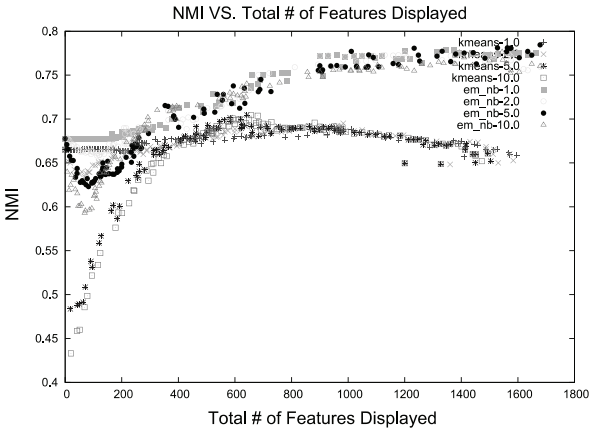
emphasized, which has a negative effect on interactive clustering framework with *EM-NB* (Figs 3(a) and 4(b)). For the interactive framework with *EM-NB*, probabilities of features in the feature set for clustering are affected through Eq. (8) and the performance in terms of *NMI* declines first and climbs up when more features are accepted by the user.

5.7. Selection of weight g

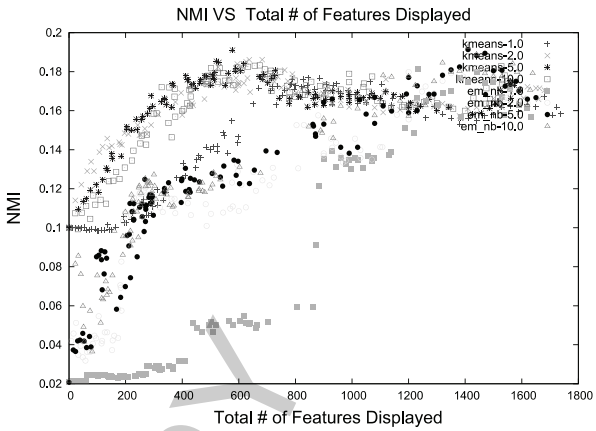
In our experiments, we tried different values of weight g (see details in Section 4.7.1) from 1 to 10 to reweight accepted features. Comparing the effect of different g values on various datasets, it can be found that feature reweighting helps to improve the document clustering performance. It can either improve clustering accuracy (Fig. 3(b)) or help reach maximum clustering performance earlier (Fig. 4(a)), which saves user effort. When the interactive framework with *EM-NB* works with $g > 1$, it improves performance when applied to the *news-similar-3* dataset (which represents the dataset that is the hardest to cluster) although it achieves comparable performance when applied to other datasets. We suggest $g = 5$ to avoid over-emphasis on accepted features.

5.8. KMeans and EM-NB on the same datasets

We also compare the interactive clustering framework with KMeans versus *EM-NB* as the underlying algorithm on the same datasets (Fig. 5). It is found that the framework with *EM-NB* is more stable than with KMeans once maximum performance is reached. In particular, the framework with KMeans declines more strongly after maximum performance is reached when applied to news-related dataset and ACM (*D-H-I*) dataset. Within the interactive clustering framework, *EM-NB* performs better than KMeans on *news-diff-3* dataset. When applied to *news-related-3* and *news-similar-3* datasets, KMeans outperforms *EM-NB* when only a few features are confirmed by the user, e.g., $f_{total} < 100$. With more features confirmed, *EM-NB* can achieve better performance than KMeans. It is mainly due to the fact that there are more overlaps in those two datasets, which causes noisy features to be confirmed as "good" for clustering. The noisy features have more negative effect on *EM-NB* than on KMeans.

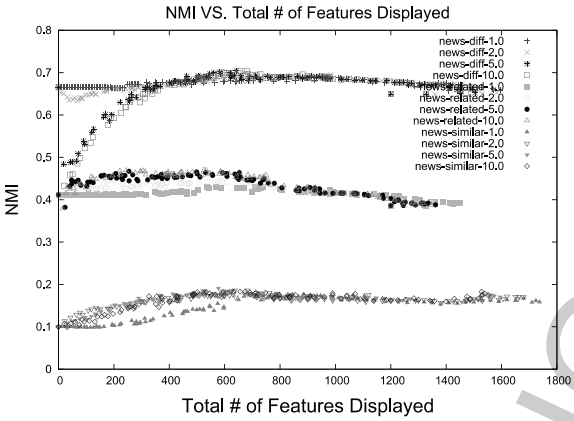


(a) KMeans and $EM\text{-}NB$ on $news\text{-}diff\text{-}3$ dataset: (x axis-f_{total}) vs. (y axis-NMI)

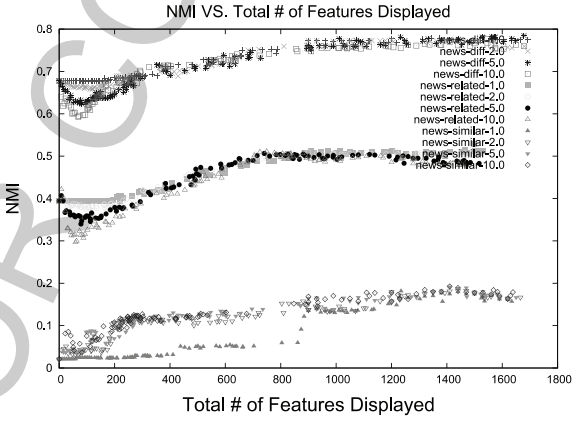


(b) KMeans and $EM\text{-}NB$ on $news\text{-}similar\text{-}3$ dataset: (x axis-f_{total}) vs. (y axis-NMI)

Fig. 5. The interactive clustering framework (Algorithm 3) with different underlying algorithms on the same newsgroups datasets.



(a) KMeans on newsgroups dataset: (x axis-f_{total}) vs. (y axis-NMI)



(b) $EM\text{-}NB$ on newsgroups dataset: (x axis-f_{total}) vs. (y axis-NMI)

Fig. 6. The interactive clustering framework (Algorithm 3) with the same underlying algorithm on *news-diff-3*, *news-related-3*, *news-similar-3* datasets. (a) KMeans on newsgroups dataset, (b) *EM-NB* on newsgroups dataset .

5.9. KMeans or EM-NB on different datasets

We compare the interactive clustering framework with KMeans or *EM-NB* on the three newsgroups sub-datasets with respect to the same number of features. When the same user effort in terms of number of features confirmed by the user is available, we can see that the *news-similar-3* dataset is still the most difficult one to be grouped and the *news-diff-3* dataset remains the easiest one (Fig. 6).

6. Guidelines for designing interactive framework

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

1. *Number of features displayed at each iteration* f. The user should be allowed to change the number of features f during the clustering process. The larger f is, the more effort is required from the user and the better performance might be achieved. Therefore, the user can decide the trade-off between the effort and performance by changing f.
2. *History of intermediate clusterings.* As a human user can make mistakes in identifying features, clustering performance can decrease if some (noisy) features are introduced. By storing the history of clustering, the user can roll back to previous clusterings and make new decisions about feature selections from there.
3. *Visualization of clusters.* In order to assist the user in judging the quality of clusters, visualization techniques such as multi-document summarization should be applied to clusters.
4. *User-modified weights.* The user should be allowed to change weights for accepted features. Our recommendation for the weight value is 5, but the user 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. In this way, the user can control and adjust the effect of individual features based on a feature's usefulness for the document clustering from their point of view. However, assigning individual weights may be time-consuming.

Developing a interactive clustering tool or framework is not trivial task and one needs lots of domain knowledge/expertise. The ontology of the domain should be incorporated into the tool to help users to make decisions.

7. Conclusions

In this paper, we designed and created a new framework that enables the user to guide the clustering process by selecting features which are meaningful to them. The framework interleaves interactive feature selection and clustering iteratively until the user chooses to stop or the underlying algorithm reaches its terminating conditions. This novel method was evaluated by comparison with three different unsupervised feature selection techniques over six different document datasets. Our experiments indicate that a certain number of features must be labeled by the user for clustering performance to be improved and to avoid early convergence of the clustering algorithm at a local optimum. After a certain amount of user input, e.g. enough features are confirmed as useful for clustering, the performance may either stay the same or decline a little. Our results show that reweighting of previously "accepted" features can also improve clustering performance. However, large weights (greater than 10) should be avoided to prevent over-emphasizing the accepted features for some datasets, which might make the clustering algorithms group the documents only based on these few over-emphasized features.

Acknowledgment

We would like to thank the anonymous reviewers for their insightful comments. This research was supported in part by the NSERC (Natural Sciences and Engineering Research Council) Business Intelligence Network, and by the MITACS NCE.

References

[1] J. Attenberg, P. Melville and F. Provost, A unified approach to active dual supervision for labeling features and examples,

in: *ECML PKDD 2010 Part I, LNAI 6321*, Springer, (2010), 40–55.
[2] M.S. Baghshah and S.B. Shouraki, Metric learning for semi-supervised clustering using pairwise constraints and the geometrical structure of data, *Intelligent Data Analysis* **13**(6) (2009), 887–899.
[3] S. Basu, A. Banerjee and R. Mooney, Semi-supervised clustering by seeding, in: *International Conference on Machine Learning* (2002), 19–26.
[4] 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*, ACM, (2004), 59–68.
[5] 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*, ACM, (2009), 99–108.
[6] C.M. Bishop, Pattern recognition and machine learning, Springer New York, 2006.
[7] H. Cheng, K.A. Hua and K. Vu, Constrained locally weighted clustering, *Proceedings of VLDB'08* **1**(1) (2008), 90–101.
[8] F. Debole and F. Sebastiani, Supervised term weighting for automated text categorization, in: *Proceedings of the 2003 ACM Symposium on Applied Computing*, ACM, (2003), 784–788.
[9] 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) (1977), 1–38.
[10] B.E. Dom, An information-theoretic external cluster-validity measure, Technical Report RJ 10219, IBM Research Division, 2001.
[11] Y. Hu, Document clustering with dual supervision, PhD thesis, Dalhousie University, 2012.
[12] Y. Hu, E.E. Milios and J. Blustein, Interactive document clustering using iterative class-based feature selection, *Technical report, CS-2010-04, Faculty of Computer Science*, Dalhousie University, Canada, (2010).
[13] Y. Hu, E.E. Milios and J. Blustein, Interactive feature selection for document clustering, in: *Proceedings of the 26th Symposium on Applied Computing, on Track "Information Access and Retrieval"*, ACM Special Interest Group on Applied Computing, (2011), 1148–1155.
[14] Y. Hu, E.E. Milios and J. Blustein, Personalized document clustering with dual supervision, in: *Proceedings of the 12th ACM Symposium on Document Engineering*, ACM, (2012).
[15] R. Huang and W. Lam, An active learning framework for semi-supervised document clustering with language modeling, *Data & Knowledge Engineering* **68**(1) (2009), 49–67.
[16] 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*, ACM, (2006), 412.
[17] D.D. Lewis and J. Catlett, Heterogeneous uncertainty sampling for supervised learning, in: *Proceedings of the Eleventh International Conference on Machine Learning* (1994), 148–156.
[18] 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* (2004), 425–430.
[19] 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* (1998), 792–799.
[20] H. Raghavan, O. Madani and R. Jones, Interactive feature selection, in: *Proceedings of IJCAI 05: The 19th International Joint Conference on Artificial Intelligence* (2005), 841–846.
[21] L. Rigutini and M. Maggini, A semi-supervised document clustering algorithm based on EM, in: *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI'05)*, (2005).
[22] B. Tang, M. Shepherd, E.E. Milios and M. Heywood, Comparing and combining dimension reduction techniques for efficient text clustering, in: *International Workshop on Feature Selection for Data Mining*, in conjunction with 2005 SIAM International Conference on Data Mining, Newport Beach, California, (23 April 2005).
[23] 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*, ACM, (2007), 707–716.