DOCUMENT CLUSTERING WITH DUAL SUPERVISION

by

Yeming Hu

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

at

Dalhousie University
Halifax, Nova Scotia
June 2012

DALHOUSIE UNIVERSITY

FACULTY OF COMPUTER SCIENCE

The undersigned hereby certify that they have read and recommend to the Faculty of Graduate Studies for acceptance a thesis entitled "DOCUMENT CLUSTERING WITH DUAL SUPERVISION" by Yeming Hu in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Dated: June 19, 2012

External Examiner: _____

Research Co-Supervisors: _____

_____

Examining Committee: _____

_____

Departmental Representative: _____

DALHOUSIE UNIVERSITY

DATE: June 19, 2012

AUTHOR: Yeming Hu

TITLE: DOCUMENT CLUSTERING WITH DUAL SUPERVISION

DEPARTMENT OR SCHOOL: Faculty of Computer Science

DEGREE: Ph.D. CONVOCATION: October YEAR: 2012

_____

Signature of Author

To my parents,

my family, and

my friends.

Table of Contents

List of Tables

List of Figures

Abstract

Nowadays, academic researchers maintain a personal library of papers, which they would like to organize based on their needs, e.g., research, projects, or courseware. Clustering techniques are often employed to achieve this goal by grouping the document collection into different topics. Unsupervised clustering does not require any user effort but only produces one universal output with which users may not be satisfied. Therefore, document clustering needs user input for guidance to generate personalized clusters for different users. Semi-supervised clustering incorporates prior information and has the potential to produce customized clusters. Traditional semi-supervised clustering is based on user supervision in the form of labeled instances or pairwise instance constraints. However, alternative forms of user supervision exist such as labeling features. For document clustering, document supervision involves labeling documents while feature supervision involves labeling features. Their joint use has been called dual supervision. In this thesis, we first explore and propose a framework to use feature supervision for interactive feature selection by indicating whether a feature is useful for clustering. Second, we enhance the semi-supervised clustering with feature supervision using feature reweighting. Third, we propose a unified framework to combine document supervision and feature supervision through seeding. The newly proposed algorithms are evaluated using oracles and demonstrated to be more helpful in producing better clusters matching a single user's point of view than document clustering without any supervision or with only document supervision. Finally, we conduct a user study to confirm that different users have different understandings of the same document collection and prefer personalized clusters. At the same time, we demonstrate that document clustering with dual supervision is able to produce good personalized clusters even with noisy user input. Dual supervision is also demonstrated to be more effective in personalized clustering than no supervision or any single supervision. We also analyze users' behaviors during the user study and present suggestions for the design of document management software.

List of Abbreviations and Symbols Used

$CACC$	Clustering Accuracy
F	F-Measure of coh and sep
$J(y_1^c, y_2^c)$	Jaccard Coefficient between y_1^c and y_2^c
K	Number of clusters
α_s	Weighting Scheme for individual centers
$\mathcal{D}$	Document collection
$\mathcal{JD}$	Jaccard Distance
$\mathcal{JD}_a$	Jaccard Distance without considering cluster labels of keywords
$\mathcal{JD}_b$	Jaccard Distance considering cluster labels of keywords
$\mathcal{JI}$	Jaccard Index
$\mathcal{RD}$	Rand Distance
$\mathcal{RI}$	Rand Index
$\mathcal{V}$	Vocabulary of the document collection
$\mathcal{W}^{\mathcal{L}}$	Labeled feature set
μ_l^c	Cluster center based on an intermediate cluster
μ_l^d	Cluster center based on $\mathcal{D}_l^L$
coh	Measure for cohesiveness of clusters
p_c	Content fraction a user reads to label a document
p_n	Noise fraction
sep	Measure for separation of clusters
ACM	Association for Computing Machinery
AL	Active learning
Constrained$K$Means	Constrained Seeded$K$Means
COPKMeans	Constrained KMeans

DualSeeded$K$Means	KMeans with dual supervision, i.e., document supervision and feature supervision
EM	Expectation-Maximization algorithm
MPCKMeans	KMeans with unified pairwise constraints and metric learning
NMI	Normalized mutual information
PCKMeans	Pairwise constrained KMeans
RCA	Relevant Component Analysis
SAC	Symposium on Applied Computing
SCOPKMeans	Soft COPKMeans
SCREEN	Spherical KMeans via feather projection
Seeded$K$Means	KMeans initialized with seeds for each cluster
TFIDF	Term Frequency, inverted document frequency
Xing$K$Means	KMeans with pairwise constraints metric learning

Acknowledgements

I would like to express much appreciation to my supervisors Dr. Evangelos E. Milios and Dr. James Blustein. Especially, I thank Dr. James Blustein for getting me into the PhD program and Dr. Evangelos E. Milios for giving me the thesis topic. Without their support and supervision, I could not have completed my PhD. I appreciate very much the time Dr. Mario A. Nascimento, Dr. Vlado Keselj, and Dr. Qigang Gao spent on my thesis. They have given lots of useful advice on my thesis.

I also want to thank my parents, my family and my friends. Their encouragement and love have accompanied me through this long battle. Without them, I could not have survived it.

Chapter 1

Introduction

1.1 Motivation

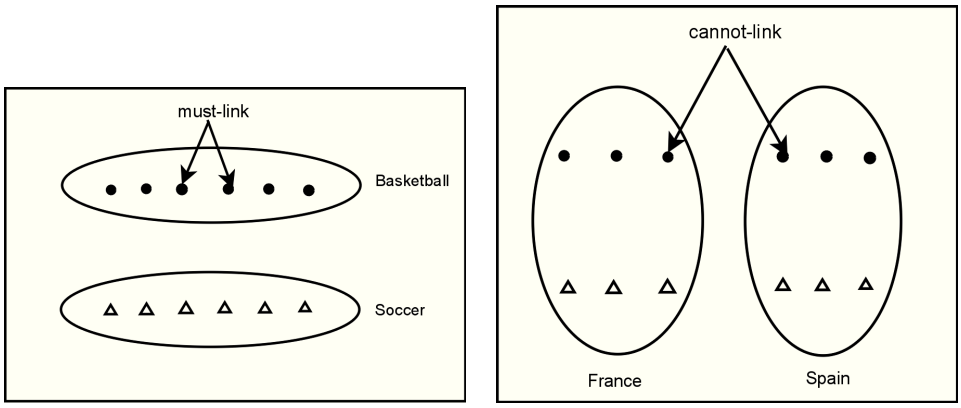
Various document collections are becoming available in many real-life text mining tasks, e.g., billions of web pages in public websites, uncategorized email messages, and personal libraries of research papers. Such document collections are most helpful if organized in a structured way for efficient navigation, browsing and search. There are existing tools with basic management functionality such as Mendeley[1], CiteULike[2] and Bibsonomy[3]. These online personal library tools provide basic browsing aids such as organization by user-defined tags. However, the basic functionality is not enough for efficient browsing. Especially, different users might prefer a personalized organization of the same document collection from their point of view. This task is usually attempted using clustering techniques.

Traditional document clustering is an unsupervised categorization 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 (1) optimizing some loss function over all document assignments to clusters such as KMeans [10], or (2) fitting a probabilistic model onto the document collection such as the multinomial naïve Bayes model [38]. Although an unsupervised method minimizes user effort for document clustering, the output of such a method is a universal set of potential clusters, i.e., all users are presented the same set of clusters given a document collection. However, given the same document collection, different users might want it to be organized in their own structure instead of being given a universal one. Consider a collection of news stories about sports, which talks about basketball and

[1] `http://www.mendeley.com/`

[2] `http://www.citeulike.org/`

[3] `http://www.bibsonomy.org/`



(a) A document pair is labeled "must-link" when document collection is organized by sports

(b) The same document pair as in Fig. 1.1(a) is labeled "cannot-link" when document collection is organized by countries

Figure 1.1: According to different needs to the organization of the same document collection, the same document pair can be given "must-link" in Fig. 1.1(a) but cannot-link in Fig. 1.1(b).

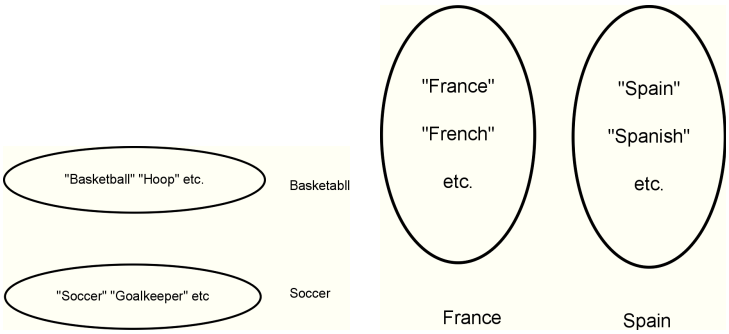
soccer in Spain and France. One user would like to organize the collection by country while another one may want it to be organized by sport. However, unsupervised clustering does not address this demand. Therefore, users are often dissatisfied with the clusters generated by the unsupervised clustering algorithms [23]. This motivates the incorporation of the user supervision in the clustering process.

Consequently, semi-supervised document clustering, which uses both labeled and unlabeled documents, has shown its usefulness in generating clusters that match user expectations [4, 6]. Unlike document classification, it is difficult to give labels to a single document because there is no cluster information before the clusters are generated. Therefore, the user supervision in the form of document pairwise constraints, "must-link" and "cannot-link" [53], is usually employed. In this method, the user labels a document constraint by indicating whether two documents should be placed into the same cluster. In the previous example, suppose there are two documents A and B. Document A talks about basketball in France while document B is about basketball in Spain. Then A and B is "must-link"-ed when the collection needs to be organized by sport but "cannot-link"-ed when it is organized by country as illustrated in Figure 1.1.

Those constraints can be used to either guide the clustering process towards a better user expected partitioning (i.e., constraint-based methods) or to learn a distance metric that the clustering algorithm can employ (i.e., distance-based methods). The constraint-based methods try to satisfy as many constraints as possible by modifying the objective functions for evaluating clusterings [5], enforcing constraints during the clustering process [53], and initializing and constraining the clustering based on the user provided constraints [4]. The distance methods make use of the labeled constraints to learn an appropriate distance measures such as Mahalanobis distance [55]. The constraint-based methods and distance-based methods can be combined in a unified framework [6].

User supervision in the form of document pairwise constraints, called document supervision, improves the performance of the clustering algorithm significantly [4, 5, 6, 55]. However, user effort can also be used to collect alternative forms of user supervision such as labeling terms, which is called feature supervision. In this thesis, we focus on single-word terms for document clustering, so we will use "term", "word", and "feature" interchangeably. Considering our example, instead of labeling documents constraints, the user could label features. If users want to organize the document collection by country, they could provide terms that are related to countries such as "France", "French" and "Spain", "Spanish". Alternatively, if they want to organize the document collection by sport, they could provide terms that are related to sports such as "basketball", "hoop" and "soccer", "goalkeeper" as illustrated in Figure 1.2.

Since both labeling document constraints and labeling terms can help producing user expected organizations, it is natural to combine the two types of user supervision together as illustrated in Figure 1.3. In a hybrid approach, the user can either label terms whilst reading (labeling) document constraints or interleave labeling terms and document constraints. In the former case, the user is allowed to label a document and the terms in the document at the same time while in the latter case, the user is presented either a document (or a document pair) or a list of terms and only the document (or the document pair) or terms from the list can be labeled at one time. Recent research in document classification [17, 37, 46] has demonstrated that feature supervision can greatly reduce the number of documents required to build



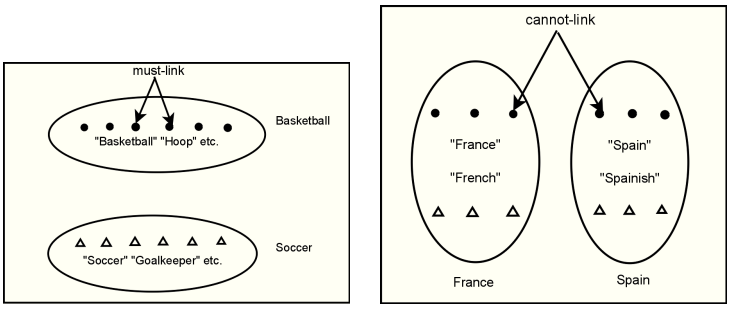
(a) Terms related to sport are labeled when the document collection is organized by sport

(b) Terms related to country are labeled when the document collection is organized by country

Figure 1.2: According to different needs to the organization of the same document collection, different terms could be labeled by users for clustering such as Fig. 1.2(a) and Fig. 1.2(b).

a high-quality classifier. Generally speaking, we conjecture that feature supervision and document supervision can be complementary towards improving cluster quality. Feature supervision is different in document clustering from document classification. In document classification, a term can be assigned into a specific class when it is labeled as a discriminative feature. However, it could be difficult (but possible) for the user to assign it to a cluster because the user needs to explore the cluster to determine its topic. Therefore, it is different and more difficult to incorporate feature supervision in document clustering than document classification.

Not only should the document collection be organized in a way the user expects, but also the user should provide as little supervision as possible because labeling document constraints (documents) and terms is expensive. Therefore, it requires that the most "valuable" information in the form of either document constraints or terms should be selected for user labeling. For document supervision, we might want to present the user the document pairs with the most uncertain relationships instead of those can be easily identified by the algorithm itself. Those easily identified constraints not only waste user supervision, but can also do harm to the clustering performance [20]. However, we might want to present the user the terms (or features) which are the most promising to help the clustering instead of those uncertain ones which are uniformed distributed across clusters and may not contain discriminative



(a) Document constraints and terms labeled when document collection is organized by types of sports

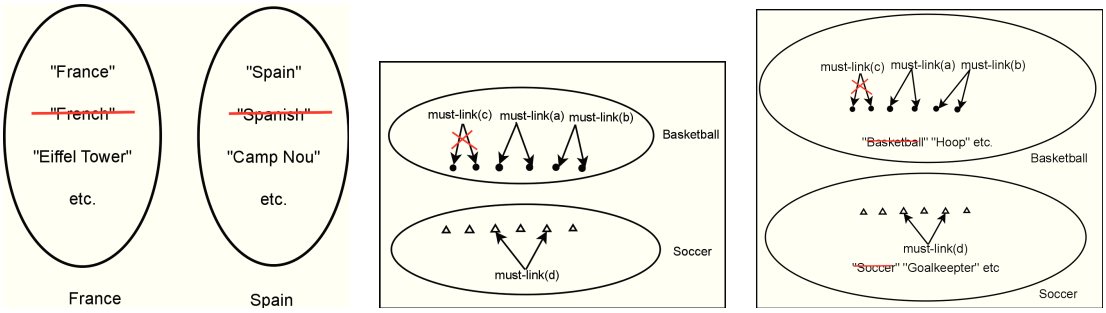
(b) Document constraints and terms labeled when document collection is organized by countries

Figure 1.3: Document constraints and terms can be labeled together to help produce user expected organization such as 1.3(a) and 1.3(b).

information.

Active learning of document constraints and terms can help to avoid labeling redundant information to save user effort. For example, after the terms “France” and “Spain” are selected, the terms “French” and “Spanish” may not be as informative as before. Instead, the terms/phrases “Eiffel Tower” and “Camp Nou” should be selected for user labeling as illustrated in Figure 1.4(a). The same rule is applicable to document constraints as shown in Figure 1.4(b). Labeled document constraints can already implies constraints between some unlabeled document pairs so that those pairs are not much useful even after they are labeled. However, it is not possible for the user to browser the document collection or the whole list of terms to find the most informative ones to label. Therefore, the clustering algorithm should play an active role in this process and select the most informative ones for the user to label.

Although document supervision and feature supervision are complementary, there may also be redundancy between certain document constraints and terms. Some already labeled document constraints may make labeling some terms redundant as illustrated in Figure 1.4(c), vice verse. Since labeling document constraints and terms together are helpful to organize the document collection in user expected way, an ideal active approach should be able to select the most informative document constraint or term for user labeling. This two types of user supervision together lead to the method of active dual supervision, in which document constraints and terms are collected



(a) Labeled terms result in that other terms are not necessary to be labeled

(b) Labeled document constraints result in that other document constraints are not necessary to be labeled

(c) Labeled document constraints result in that some terms are not necessary to be labeled

Figure 1.4: Information redundancy. Labeled document constraints and terms could cause other unlabeled document constraints and terms not necessary to be labeled such as 1.4(a), 1.4(b), and 1.4(c).

simultaneously and the best user expected clusters should be generated with least user effort.

1.2 Our Goals

Our hypothesis is that feature supervision alone or together with document supervision can significantly improve the organization of a document collection better matching user expectation. We would like to incorporate the labeled feature information in two alternative ways: (1) label a feature as useful for clustering but do not associate it with a cluster; (2) label a feature and associate it with a cluster. By incorporating feature supervision into clustering algorithms through these two approaches, we conjecture that the clustering performance should be improved, i.e., the clusters generated can better reflect the user's point of view.

The goal of this thesis is to introduce, formalize and evaluate the feature supervision in document clustering. We first explore feature supervision for interactive feature selection to aid unsupervised document clustering. Second, we augment semi-supervised clustering with feature supervision through feature reweighting. Third, we propose a novel unified framework to combine document supervision and feature supervision through seeding. Finally, we conduct a user study to investigate whether the proposed framework with noisy user input is able to produce personalized clusters

for users. In this thesis, we propose and verify a practical framework to generate personalized clusters from the same document collection for different users. One direct benefit of the personalized clusters might be that the user can use them to find the needed information more efficiently than fully automatic generated clusters.

1.3 Our Contributions

We summarize the contributions of this thesis as follows:

(1) We propose a novel iterative framework which involves users interactively selecting the features used to represent documents and demonstrate that the feature set obtained in this way can produce clusters that better match user expectation.

(2) We present a new method of incorporating the user supervision into unsupervised document clustering through feature supervision and feature reweighting. We demonstrate that feature supervision can significantly improve document clustering accuracy when applied on top of document supervision with various semi-supervised clustering algorithms. In particular, our experiments show that feature supervision is superior to distance metric learning methods based on labeled document constraints since feature supervision requires much less user effort in terms of number of documents labeled to achieve the same performance.

(3) We propose a novel unified framework which combines document supervision and feature supervision. The framework assumes the number of clusters is given and uses both labeled documents and features in terms of seeding clusters and refines this information using intermediate clusters. We demonstrate that the framework successfully combines labeled documents and features through seeding. The framework can also work when only a fraction of clusters are being seeded.

(4) We conduct a user study in which we ask users to group the same document collection into clusters based on their own understandings, which are then used as ground truth to evaluate semi-supervised clustering algorithms for user personalization. The contributions from this user study are:

    (1) By designing and testing useful operations and text visualization methods to help users to group documents, we demonstrate that selecting keywords

during assigning documents takes little time using the designed interface and operations. We suggest that these operation should be included in the supervision interface for document management software.

(2) We observe that different users group the same document collection differently, in terms of the number of clusters, the cluster memberships of documents, and the assigned keywords. In addition, we observe that a user's organization of the document collection changes over time. Therefore, we conclude that clustering algorithms which accommodate personalization should be employed.

(3) We show that semi-supervised clustering algorithms with a small amount of user input can produce personalized clusters and verify that semi-supervised clustering algorithms can produce better quality of clusters with user input than unsupervised clustering, even if user input is noisy.

(4) We demonstrate that assigning keywords for clusters can help clustering algorithms to organize documents better matching user's point of view than labeling only documents. The assigned keywords are used to construct cluster structures through seeding to initialize the clustering algorithms.

With our research, we anticipate to design new algorithms to help the users to build clusters which are best matched with their expectations but with as little effort as possible. Based on the evaluation through both oracles and human users, the frameworks we propose show promise in helping users to better organize their document collection and obtain personalized clusters which match their point of view.

1.4 Algorithms and User Supervision

In this thesis, we use two types of user supervision, i.e., document supervision and feature supervision. *Document supervision* involves document-level supervision: (1) label a document by assignment, i.e., assign a document into a cluster, or (2) label a pair of documents by a constraint, i.e., establish the relationship (constraint) between two documents. We shall assume that the user needs to read a fraction of the documents in order to label a document or a document constraint. *Feature supervision* involves feature-level supervision: (1) label a feature by usefulness, i.e., identify whether a

Algorithm	Chapters	Document supervision
Seeded$K$Means	5, 6	by assignment to a cluster
Constrained$K$Means	5	
COPKMeans	5	by a constraint on a pair of documents
Xing$K$Meanss	5	

Table 1.1: Traditional clustering algorithms with only document supervision, labeling methods the algorithms use in document supervision, and the chapters the algorithms appear.

feature is useful for clustering, or (2) label a feature by association, i.e., associate a feature with a document if the feature describes the topic of the document. A feature can be labeled in three situations: (1) from a list — a feature is labeled as useful or not for clustering when a list of features is presented to the user, or (2) while establishing a document constraint — a feature is labeled as whether useful or not for clustering while the user is labeling a document pair constraint by reading the two documents associated with the constraint, or (3) while assigning a document to a cluster — a feature is labeled and associated with the document while the user is labeling a document by reading the content of the document. In this thesis, we have different clustering algorithms using at least one of those supervision methods. We summarize the clustering algorithms, the labeling (supervision) methods the algorithms use, and the chapters the algorithms appear in Table 1.1 and Table 1.2. We also list the key findings from each algorithm enhanced with feature supervision and the user study conducted in Table 1.3.

1.5 Outline of the Thesis and Copyrights

The rest of the thesis is organized as follows.

Chapter 2 (Related Work) reviews and summarizes the existing semi-supervised document clustering methods, feature supervision methods in document classification. This chapter reiterates the differences of the feature supervision between document clustering and document classification, which explains the necessity of our research. The discussion of the literature review is a significant extension of the ones from our previous conference papers: (1) Paper [22] presented on Track “Information

Algorithm	Chapters	Feature Supervision			Doc. Supervision
		Name	By	From	
Feature selection	4	*usefulness-list*	usefulness for clustering	a list	None
COPFes$K$Means	5	*usefulness-content*	usefulness for clustering	doc. content	pairwise of documents: "must-link" or "cannot-link"
XingFes$K$Means	5				
SeededFes$K$Means	5	*association-content*	association with a document, and with a cluster through the document	doc. content	assignment of a document to a cluster
ConstrainedFes$K$Means	5				
DualSeeded$K$Means	6, 7				

Table 1.2: Novel clustering algorithms presented in this thesis with document supervision and/or feature supervision, labeling methods the algorithms use in document supervision and feature supervision, and the chapters the algorithms appear.

Algorithm/Method	key findings
Feature selection	Feature supervision *usefulness-list* is effective for interactive feature selection, by selecting the useful features for clustering to improve document clustering performance through feature reweighting.
COPFes$K$Means	Feature supervision *usefulness-content* (Table 1.2) incorporated by feature reweighting can improve the clustering performance further in addition to the labeled document constraints which guide the clustering process.
XingFes$K$Means	Metric learning methods do not work well based on labeled document constraints, i.e., clustering performance can be worse than baseline algorithms without metric learning. However, feature supervision *usefulness-content* (Table 1.2) with a small amount of effort can enhance the metric learning method and achieve better performance.
SeededFes$K$Means	Useful features identified from feature supervision *usefulness-content* (Table 1.2) can be used through feature reweighting to enhance document seeding method and improve clustering performance.
ConstrainedFes$K$Means	Useful features identified from feature supervision *usefulness-content* (Table 1.2) can be used through feature reweighting to enhance document seeding and constrained methods and improve clustering performance.
DualSeeded$K$Means	Features associated with clusters from feature supervision *association-content* (Table 1.2) and documents assigned into clusters can be combined together to improve clustering performance through document seeding and feature seeding.
User study	Personalized clusters are desired by users. A user's organization of the same document collection changes over time. Users are able to assigned keywords to clusters during exploration of the document collection. Semi-supervised clustering algorithms with a small amount of user input can produce personalized clusters. Semi-supervised clustering algorithms with dual supervision perform better than those without any supervision or with any single supervision

Table 1.3: Key findings from clustering with dual supervision

Retrieval and Access" at the ACM Symposium on Applied Computing 2011 Conference[4], (2) Paper [24] presented on Track "Information Retrieval and Access" at the ACM Symposium on Applied Computing 2012 Conference[5], (3) Paper [25] presented on Track "Data ming" at the ACM Symposium on Applied Computing 2012 Conference[6], (4) Paper [26] published at the ACM Applied Computing Review[7] and (5) Paper [23] presented at the ACM Symposium on Document Engineering 2012 Conference[8].

Chapter 3 describes the datasets and performance measures to evaluate our proposed algorithms and frameworks. The datasets include different types of documents such as newsgroups messages from the 20 newsgroup dataset, abstracts from the 3-classic dataset, webpages from the webkb and the sector datasets, academic papers with full text from ACM digital library, and news articles from the reuters21578 dataset. Different evaluation measures are used to evaluate algorithms based on classes labels of standard datasets and users' manual organized clusters, and compare two clusterings of the same document collection. The preprocessing procedure of the documents is also introduced in this chapter.

Chapter 4 describes an interactive feature selection framework for document clustering, in which a user can label features by indicating whether a feature is useful for clustering. The discussion of this chapter was adapted from the paper [22] presented at the ACM Symposium on Applied Computing 2011 Conference and the technical

[4] This work is based on an earlier work: Interactive Feature Selection for Document Clustering, in SAC'11 Proceeding of the 2011 ACM Symposium on Applied Computing, Copyright 2011 ACM 978-1-4503-0113-8/11/03. http://dx.doi.org/10.1145/1982185.1982436.

[5] This work is based on an earlier work: Enhancing Semi-supervised Document Clustering with Feature Supervision, in SAC'12 Proceeding of the 2012 ACM Symposium on Applied Computing, Copyright 2012 ACM 978-1-4503-0857-1/12/03. http://dx.doi.org/10.1145/2245276.2245457.

[6] This work is based on an earlier work: Semi-supervised Document Clustering with Dual Supervision through Seeding, in SAC'12 Proceeding of the 2012 ACM Symposium on Applied Computing, Copyright 2012 ACM 978-1-4503-0857-1/12/03. http://dx.doi.org/10.1145/2245276.2245306.

[7] This work is based on an earlier work: A Unified Framework for Document Clustering with Dual Supervision, in the ACM Applied Computing Review, 12(2), ACM 2012. Copyright 2012 ACM 978-1-4503-0857-1/12/03. ACM DOI will be available.

[8] This work is based on an earlier work: Personalized Document Clustering with Dual Supervision, in DocEng'12 Proceeding of the 2012 ACM Symposium on Document Engineering, Copyright 2012 ACM 0-12345-67-8/90/01. ACM DOI will be available.

report [21] at Faculty of Computer Science, Dalhousie University[9] .

Chapter 5 demonstrates the effectiveness of feature supervision with existing semi-supervised clustering algorithms through feature reweighting. The discussion of this chapter was adapted from the paper [24] presented at the ACM Symposium on Applied Computing 2012 Conference.

Chapter 6 proposes a novel unified framework to combine document supervision and feature supervision through seeding. The discussion of this chapter was adapted from the paper [25] at the ACM Symposium on Applied Computing 2012 Conference and the paper [26][10] at the ACM Applied Computing Review.

In Chapters 4, 5, and 6, the proposed frameworks and/or algorithms are evaluated using oracles i.e., simulating human users based on the underlying class labels of the standard datasets. Although oracles allow newly proposed algorithms and frameworks to be evaluated quickly, they have several disadvantages (See details in Chapter 7).

Chapter 7 presents a user study conducted to demonstrate the necessity of cluster personalization and demonstrates that document clustering with dual supervision can produce personalized clusters even with noisy user input from human users.

Chapter 8 summarizes the thesis and presents a discussion of the implications of the thesis. It also reveals the connections between algorithms/frameworks in different chapters and presents the opportunities for further investigation.

[9] `http://www.cs.dal.ca/research/techreports/cs-2010-04`

[10] This work is based on an earlier work: A Unified Framework for Document Clustering with Dual Supervision, in Volume 12, Issue 2 of the ACM Applied Computing Review, Copyright 2012 ACM 0-12345-67-8/90/01. ACM DOI will be available.

Chapter 2

Related Work

Document clustering is a popular research topic because of the huge and ever increasing amount of available electronic documents. However, the clusters of documents generated by the unsupervised clustering algorithms may not reflect the user's point of view. Attempts to produce clusters that match user expectation use various ways of incorporating user supervision during clustering.

On one hand, semi-supervised clustering with instance-level pairwise constraints has been a topic of significant interest. On the other hand, feature supervision in document classification provides us one more approach to involve the user supervision in document clustering. Since we try to incorporate feature supervision in the document clustering setting, we would like to review the work related to both semi-supervised clustering algorithms and feature supervision in document classification. In addition, this thesis also explores personalized document clustering with both document supervision and feature supervision. Therefore, our reviews focus on the following areas: (1) semi-supervised clustering, (2) feature supervision in the document classification setting, and (3) personalized clustering.

We define the following uniform terminology: (1) instance: a data point of the set that is being clustered, e.g. a document, also called an example; (2) feature: an attribute of an instance, e.g. a single-word or multi-word term. In this chapter, we use "instance" uniformly for "document", "data point", and "example". We also use "feature" uniformly for "word", "term", "attribute". However, we might only use "document" and "term" for later chapters as our methods are proposed only for document clustering.

2.1 Semi-supervised Clustering

There has been a significant amount of research on semi-supervised clustering which is generally categorized into the following areas: (1) constraint-based methods, in

which the constraints have to be enforced during the clustering process [53] or modify the optimization of the loss function [31]; (2) labeled cluster seeds or cluster seeds derived from the constraints to initialize the cluster centroids [4]; (3) distance-based methods, in which constraints are employed to learn adaptive distance metric using metric learning techniques [3, 12, 55]; (4) hybrid methods of constraint-based and distance-based methods [6]; (5) feature space transformation method, in which the original high-dimensional feature space can be projected into low-dimensional feature subspaces guided by user labeled constraints [51] and feature selection methods, in which the feature selection algorithm is guided by the pairwise constraints [48, 56]; (6) Besides those five methods which use constraints in a passive manner, there is also some research on actively selecting the most informative constraints for clustering [5, 20, 27].

The current semi-supervised clustering algorithms work with document-level supervision. However, feature-level supervision may further help generate better quality of clusters, i.e., better matching user expectation. More importantly, feature supervision can be done together with document supervision. For example, users can label features through highlighting while they are labeling documents.

We describe briefly the current semi-supervised clustering algorithms and the possible approaches to incorporate feature supervision into them. All those algorithms are general clustering algorithms, which are not designed specifically for document clustering problems. Although all of them are applicable to document clustering, distance-based methods are computation prohibitive when applied to high-dimensional data like documents. A basic premise of this thesis is that features (terms) of documents make sense to humans, and feature selection can be done by humans, which is not true for general clustering problems.

2.1.1 Constraint-Based Methods

COP-KMeans is an algorithm directly enforcing the satisfaction of constraints during the clustering process [53]. Since it requires that all the user provided constraints be satisfied, the algorithm can fail when an instance cannot be placed into any cluster due to constraint violation. Therefore, a soft-constraint version SCOP-KMeans [52] is used to handle soft constraints, i.e. associating a cost with a constraint if it is

violated.

PCKMeans [5] initializes the clusters by selecting instance neighborhoods derived from the pairwise constraints. Although it has the same centroid estimation step as KMeans [10], i.e. the mean of the instances assigned to the cluster, it assigns an instance into the cluster so that it minimizes the sum of the distance and the cost of the constraint violation. Similar to SCOP-KMeans, it can handle the constraints softly.

In a paper by Ji and Xu [31], a set of instances is modeled as a weighted graph with each document as a vertex and each edge connecting a pair of vertices weighted by the similarity between documents. Then the best cuts of the graph under the pairwise constraints are found for the document clustering purpose. In that paper, the normalized cut method is used to demonstrate the idea, but they claim that any spectral clustering method should work.

2.1.2 Seed-Based Methods

Both Seeded-KMeans and Constrained-KMeans [4] initialize the clusters using labeled instances for all clusters or a fraction of the clusters. Pairwise constraints generated from the labeled instances are also used to guide the clustering. By applying the two algorithms to a variety of datasets, Basu et al. [4] demonstrate that clustering by seeding can improve clustering performance significantly even when the seeds are noisy or incomplete.

In Seeded-Kmeans, the labeled instances are only used to initialize the KMeans algorithm instead of random initialization and they can change their cluster memberships during the clustering. In Constrained-KMeans, the labeled instances are used to seed the clusters and cannot change their cluster memberships. Therefore, the constraints between instances have to be satisfied in constrained-KMeans as in COP-KMeans [53]. Experiments indicate that Constraint-KMeans is more sensitive to noisy seeds (i.e., seeds assigned to a cluster which should not be assigned) than Seeded-KMeans is.

Although incomplete cluster seeds, i.e., when not all clusters have labeled data as seeds, provide only partial prior knowledge, they are still helpful to clustering. The transitive closures derived from "must-link" document constraints are usually used

to seed a fraction of the clusters [5, 6].

2.1.3 Distance-Based Methods

Mahalanobis metric[1] is used to scale the features for clustering aiming for improved clustering performance compared with Euclidean distance. A suitable Mahalanobis metric for clustering can be learned from pairwise constraints [55]. In the work of Xing et al. [55], the criterion for finding the Mahalanobis metric is to minimize the squared distance between all pairs of must-link instances under the constraint that the squared distance between all pairs of cannot-link instances should be larger than constant c, where $c > 1$. In the case that the matrix is diagonal, the metric can be learned using an efficient algorithm based on the Newton-Raphson method. In the case that the matrix is full, a different algorithm using gradient descent and the idea of iterative projection is developed.

Bar-Hillel et al. [3] propose a method by which the full Mahalanobis metric matrix can be learned by applying the Relevant Component Analysis (RCA) algorithm [45] to the "chunklets" derived from the "must-link" constraints using convex optimization. In this method, the full matrix is always learned so it may not be applicable for high-dimensional data. In the paper by Bar-Hillel et al. [3], a connection to the method described in Xing et al. [55] is also presented.

An obvious disadvantage of these distance-based methods is that they involve expensive computations. Therefore, they are not suitable for high-dimensional data such as documents, of which the matrix is large. In fact, the computation is prohibitive even for a diagonal matrix.

2.1.4 Hybrid Methods

MPCK-Means [9] provides a new method integrating constraint-based and distance-based learning in a unified framework. In MPCK-Means, the objective function is redefined as the sum of the total squared distances between the points and their cluster centroids, and the cost incurred by violating any pairwise constraint. Therefore, MPCK-KMeans can handle pairwise constraints softly as SCOP-KMeans and

[1] $d(x, y) = d_A(x, y) = \|x - y\|_A = \sqrt{(x - y)^T A(x - y)}$, where x and y are two points and A is a matrix.

PCKMeans, i.e., the pairwise constraints should be satisfied as many as possible but can also be violated. In addition, the Euclidean distance in the objective function is parametrized using a symmetric positive-definite matrix. Similar to the approach Xing et al. [55], the metric matrix may be either diagonal or full. The optimization of this new objective function naturally combines the constraint-guide searching and metric learning for KMeans. Unlike other metric learning techniques, which only learn a single metric using the supervised pairwise constraints, MPCK-Means can use both labeled and unlabeled data to learn multiple metrics, i.e., one metric for each cluster.

The framework proposed in [6] provides a probabilistic model for MPCK-Means using Hidden Markov Random Fields (HMRFs). In addition to Euclidean distance, the HMRFs framework also explores two other popular distortion measures: cosine similarity and Kullback-Leibler divergence.

Since hybrid methods form a combination of constraint-based and distance-based methods, they have the advantages and disadvantages of the individual components. However, the metric learning methods generally require expensive computations so that the hybrid methods might be not suitable for user interaction to refine their input.

2.1.5 Feature Transformation and Selection Methods

Semi-supervised clustering method based on spherical KMeans via feather projection (SCREEN) [51] addresses the problem of constraint-guided feature projection and integrates it with a semi-supervised clustering algorithm. In the initialization stage of the SCREEN method, each transitive closure of "must-link" constraints is replaced with its average instance and is weighted by its size. The average instances are used to represent the "cannot-link"s. Then, a projection matrix is learned by maximizing the sum of distances between "cannot-link" instances. All instances are projected to a low-dimensional space using the learned matrix. After the projection, all instances are clustered by the constrained spherical Kmeans, in which instances that cannot be linked are placed into their closest clusters. Finally, the replaced instances in each transitive closure of "must-link" constraints are recovered and placed into the same cluster as the replacing average instance belongs to. However, after the projection, it

is difficult for users to interpret the document vectors.

Constraint Score [56] makes use of pairwise constraints for feature selection. By incorporating this small amount of supervision in feature selection algorithms, the constraint score method can achieve performance similar to what the Fisher score method can achieve when using full class labels. However, to achieve such performance, the constraint score method relies on a good selection of pairwise constraints. Instead of seeking one single good set of constraints, Bagging Constraint Score [48] incorporates bagging into the constraint score method, where the diversity of the ensemble helps to build a better constraint score for feature selection.

2.1.6 Active Learning with Pairwise Constraints

The active learning scheme for selecting pairwise constraints proposed in Basu et al. [5] is made up of two phases: *Explore* and *Consolidation*. The *Explore* phase uses the farthest-first traversal scheme to find a non-null neighborhood for each underlying cluster as fast as possible. At the end of *Explore* phrase, at least one point has been obtained for each cluster. In *Consolidate* phrase, randomly selected instances can be queried to be paired with one of the neighborhoods of the clusters. This stage expands and consolidates the neighborhoods obtained in the *Explore* phase.

Huang et al. [29] investigates the active learning of constraints for semi-supervised clustering using intermediate clustering results to guide the selection of the document pairs. Two gain models are designed to select the most informative document pair given the current assignments. In the independent model, previously identified constraints do not affect the selection of the next most informative constraints in the current iteration. The dependent model also considers the previously identified constraints to avoid redundant selection and maximize the gain.

Instead of finding a neighborhood [5, 29] for each cluster, Huang and Lam [27] discover neighborhoods only for some clusters so that constraints cannot be consumed too quickly when the datasets are unbalanced, i.e. some clusters are very large while others are very small. In addition, a language model is designed to represent the neighborhoods. Based on this language model, several gain functions are proposed to measure how much information can be gained by having a document pair been

labeled. Three models, namely, uncertainty model, generation error model, and objective function model, are investigated for the gain function.

Constraint Selection by Committee [20] makes use of the ensemble approach to identify informative constraints for semi-supervised clustering. Once a large collection of based clusterings is obtained, a co-association matrix A is built to represent the co-assignments between instances across all clusterings. Unambiguous associations with very large $A_{ij} \approx 1$ (must-link) and small $A_{ij} \approx 0$ (cannot-link) can be identified automatically. Therefore, user supervision can be used to identify the more uncertain and informative constraints with $A_{ij} \approx 0.5$.

Since user supervision is labor-intensive, an active learning scheme should be designed to recommend the most potentially informative documents for the user to label. In this thesis, an adapted version of the *Explore* and *Consolidate* framework is proposed for the user study. The main difference is that the adapted version does not require the number of the clusters K be known. The adapted framework is built into a well-designed interactive user interface with various operations to help users to perform both document supervision and feature supervision. In addition, active learning of labeling features can also be performed [22]. Furthermore, the active learning of features and documents at the same time might be unified into a single framework by adapting the method proposed for document classification [2].

2.2 Feature Supervision

Feature supervision has been mainly used to improve the performance of classification algorithms, such as using the labeled features for each class to constrain the probabilistic model estimation [17], making use of feature feedback with support vector machine [40], and creating pseudo-instances using labeled features for each class [36, 54]. However, classification methods assume there are pre-defined categories to which users can assign documents or keywords. In document clustering setting, users have to form their perception of the document collection during exploration.

2.2.1 Feature Supervision for Document Classification

In this section, we review the main classification algorithms using feature supervision.

A set of representative features for each class is labeled in Liu et al. [36]. 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 [14] is applied iteratively to build new classifiers. In Liu et al. [36], the features are used only for constructing class seeds in but are not used in the following EM process.

Interactive feature selection for actively building an accurate classifier is studied in [40], which uses linear support vector machine as the base classifier. At each iteration of the active learning, users are asked to label both the most uncertain document and a list of features. It uses uncertainty sampling [35] for document-level active learning which requires users to label the document about which the classifier(s) is (are) not certain about. The feature-level active learning ranks the features using information gain based on the currently labeled documents and presents the user with the features that are at top of the ordered list. The labeled features are incorporated into the support vector machine through feature reweighting.

Druck et al. [17] proposes that the labeled features can be used to constrain the probabilistic model estimation on unlabeled instances instead of creating pseudo-instances as in other approaches. These soft constraints are expressed using generalized expectation criterion in a parameter estimation objective function that expresses preferences on values of a model expectation. The complete object function also includes a Gaussian prior on parameters.

Sindhwani and Melville [46] proposes a novel semi-supervised sentiment prediction method which utilizes both labeled instances and features. Their method is based on the joint analysis of instances and features using a bipartite graph representation of the data. They claim that this semi-supervised model performs significantly better than purely supervised and competing semi-supervised classification techniques when applied to a diverse collection of sentiment prediction problems.

Sindhwani et al. [47] studies the problem of active dual supervision using graph models. The classical uncertainty and experimental design based active learning schemes are applied to optimally query the instance oracle and the feature oracle to simultaneously collect two different forms of labels. The transductive experimental design used for optimal feature labeling are based on various measures such as uncertainty, certainty, random, and variance-based measure. The results indicate the

certainty measure is the best of the four measures for active feature supervision. In these schemes, the active learner probabilistically queries the instance oracle or the feature oracle for the most informative example or word based on an interleaving probability.

A unified approach is presented in Attenberg et al. [2] for active dual supervision of labeling both instances and features. This approach determines which feature or document a classifier is most likely to benefit from having been labeled. Unlike previous active dual supervision methods which select documents and features to label separately, this framework uses a holistic approach to active dual supervision based on Expected Utility (estimated risk minimization) framework, in which the most informative documents or features are selected in tandem. Several Expected Utility measures are investigated in the proposed framework. Some measures are supervised utility measures, such as log gain, accuracy and entropy computed on labeled documents while others are unsupervised utility measures, such as Entropy and maximum posterior computed on the pool of unlabeled documents [44].

2.2.2 Feature Supervision for Document Clustering

One of the main research goals in this thesis is to investigate whether feature supervision can improve document clustering performance. As we mentioned before, there are no predefined categories in the document clustering setting and users have to form their perception of the document collection during exploration. Consequently, users do not know the number of clusters and what clusters they have at the beginning. Due to this limitation, users cannot assign features to clusters directly before they create them during exploration of the document collection. Therefore, we propose three methods to label features for document clustering based on how a feature is labeled. In the first method, users identify whether a feature is useful for clustering given a list of features. The confirmed features will be reweighted for the representation of documents. In the second method, users label features whilst they are labeling (by reading) documents. Besides being identified as useful or not for clustering, the features labeled whilst labeling a document can be associated with the document from which they are labeled. After a document is assigned to a cluster, the features associated with the document are automatically assigned to the same cluster. In the

third method, users can assign features directly to already created clusters. For the second and the third methods, features can be used to seed the assigned clusters since they are associated with clusters. At the same time, they can also be used without considering the cluster labels and be reweighted for the representations of the documents.

More specifically, we have different concrete feature supervision methods for the semi-supervised clustering algorithms discussed in Section 2.1. Since constraint-based methods do not create clusters but only identify the relationships between pairs of documents, it is expected that the user can label a feature by indicating whether it is "good" (discriminating) for clustering while labeling a document constraint for the semi-supervised clustering algorithms in this category. Then, the labeled features can be reweighted to represent the documents before the semi-supervised clustering algorithms begin. Note that the labeled features are not associated with any specific clusters in constraint-based methods. In seed-based methods, a user labels documents as seeds for clusters. Therefore, it is expected that a user can label features while labeling a seed (instance) for a cluster. The labeled features are associated with the documents in which they are labeled. After a document is assigned (labeled) into a cluster, the features associated with the document are automatically assigned into the same cluster. In addition, users can come up with their own keywords (features) for a document. They can also assign keywords directly into an already created cluster. Without cluster labels of features being considered, the labeled features can be reweighted for the representations of the documents before the semi-supervised clusterings start. However, the cluster labels of the features might help us to cluster documents better through seeding the clusters to which they are assigned. Therefore, the features with cluster labels can be used to seed the clusters to which they are assigned. Since distance-based methods use document constraints similar to constraint-based methods, feature supervision can be performed similarly by identifying whether a feature is useful for clustering while a document constraint is labeled. The labeled features can be reweighted for the representations of the documents and a new distance metric can be learned based on reweighted document representations. In addition, the labeled features can also be reweighted based on the new distance metric learned from the labeled document constraints if the matrix

used in distance-based methods is diagonal. The feature reweighting always takes place before the semi-supervised clustering algorithms begin. Since hybrid methods form a combination of constraint-based and distance-based methods, the same feature supervision approaches in constraint-based and distance-based methods can be used in hybrid methods. The labeled features can be used to reweight the representations of the documents without cluster labels being considered. In addition, the labeled features can also be reweighted at each iteration of the hybrid methods after the distance metric is learned based on the intermediate clusters and labeled document constraints. In feature transformation methods, the labeled features can be incorporated similarly to constraint-based methods by reweighting the labeled features before the semi-supervised clustering algorithms start. However, after the projection, it is difficult for the user to identify the labeled features any more. In feature selection methods, the labeled features should have the highest constraint score computed based on the labeled document pairs since they are thought of as useful for clustering by users. Therefore, they should always be included in the selected features for clustering.

2.3 Personalized Clustering

There has not been much research on the personalized clustering. Drucker et al. [18] recruited thirty-two participants to group the same document collection and compare the different manual organizations from all users. They found that users had individual styles and create distinct clusters. However, Drucker et al. [18] did not evaluate the personalization of the clusters produced from different algorithms, since the generated clusters are not evaluated by users who provided the supervision. ClusteringWiki [1] incorporates user's intervention and collaboration by editing and changing the cluster memberships through a Wiki interface to personalize the user's search results. Leung et al. [33] introduce a personalized approach to cluster short and ambiguous queries by capturing a user's conceptual preferences. The approach first extracts concepts from the search results returned from a query and then identifies related queries using the extracted concepts. Second, a new two-step agglomerative clustering algorithm is proposed to generate personalized query clusters. Rigou et al. [42] introduces an algorithm to generate personalized clusters of all web documents

satisfying a set of predefined personal user preferences. The personalized algorithm is based on a range tree structure. Since the web documents need to be clustered before being presented to end users, it allows more effective manipulation and supports a better process of decision making. This method uses predefined preferences and is not flexible for dynamic user needs. Feature supervision can be used together with these methods and possibly improve user's personalized clusters further. In these methods, user supervision is applied to search results and queries, which are generally short. In addition, these methods are particularly tuned for web search and might not work on general document clustering with long documents such as academic papers.

Chapter 3

Datasets and Evaluation Measures

In this chapter, we describe the datasets we use to evaluate the newly proposed algorithms and frameworks. At the same time, we present the evaluation measures that are used to compare the performance of the newly proposed algorithms with the baseline algorithms.

3.1 Datasets

We use several datasets to validate our algorithms and frameworks. All datasets including the derived datasets have their own characteristics, such as the types and length of the documents. Those datasets vary by size of the datasets, length of documents, and separability between clusters. All datasets we use in this thesis are summarized in Table 3.1. The details about each dataset can be found in the following sections.

3.1.1 The 20 Newsgroups Collection

The 20-Newsgroups collection[1] is widely used for testing text mining algorithms. There are approximately 20,000 newsgroup messages, which are almost evenly partitioned into 20 different Usenet newsgroups. We derived several sub-datasets from the original 20 Newsgroups dataset to test the robustness of the clustering algorithms.

First, three small datasets with different separability are derived according to the method described in Basu et al. [6], i.e., *news-diff-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. Dataset *news-diff-3* covers topics from 3 quite different newsgroups (alt.atheism, rec.sport.baseball, and sci.space). Dataset *news-related-3* contains 3 related newsgroups (talk.politics.misc, talk.politics.guns, and talk.politics.mideast).

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

Dataset	Description	Categories included	Doc./Cat.	Tot.
news-similar-3	The 20-Newsgroup data set consists of 20 different Usenet newsgroups, each of which has approximately 1000 newsgroup messages. See Details in Section 3.1.1.	3:comp.graphics,comp.os.ms-windows.misc,comp.windows.x	100	300
news-diff-3		3:alt.atheism, rec.sport.baseball, and sci.space	100	300
news-related-3		3:talk.politics.misc, talk.politics.guns, and talk.politics.mideast	100	300
news-multi-7		7:alt.atheism,comp.sys.mac.hardware, misc.forsale,rec.sport.hockey,sci.crypt, talk.politics.guns,soc.religion.christian	100	700
news-multi-10		10:alt.atheism,comp.sys.mac.hardware,misc.forsale, rec.autos,rec.sport.hockey,sci.crypt,sci.med, sci.electronics, sci.space, talk.politics.guns	100	1000
D2-D2&D3-D3	Research papers about Computer Science in full text. See Details in Section 3.1.2.	3:*D2*: Software Engineering, *D2&D3*: Software Engineering and Programming Languages, and *D3*: Programming languages	~ 100	~ 300
D-H-I		3:D: Software, H: Information Systems, and I: Computing Methodologies	~ 200	~ 600
3-classic-abstract	abstracts from CISI, CRAN and MED of the SMART document collection. See Details in Section 3.1.3.	CISI, CRAN, and MED	~ 1000	~ 3000
webkb-sfcp-4	webpages from different universities. See Details in Section 3.1.4.	4:student, faculty, course, project	250	1000
sector-multi-10	webpages from different industrial sectors. See Details in Section 3.1.5.	10:basic.materials,capital.goods,consumer.cyclical, oil.and.gas.integrated, investment.services, biotechnology.and.drugs, hotels.and.motels, communications.equipment, railroad, water.utilities	100 (railroad-95)	995
reuters-multi-10	news articles from Reuters21578. We use the top 10 most frequent categories, documents of which does not have multiple labels. See Details in Section 3.1.6.	10:acq, coffee, crude, earn, gold, interest, money-fx, ship, sugar, trade	100 (gold-90)	990

Table 3.1: All Datasets used in this thesis, derived from the 20-newsgroups (Section 3.1.1), the ACM collection of full text papers about computer science (Section 3.1.2), the 3-classic (Section 3.1.3), the webkb (Section 3.1.4), the industry sectors (Section 3.1.5) and the reuters21578 (Section 3.1.6).

Dataset *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 be clustered. Those three datasets are created for the purpose to study the effect of data separability of the algorithms. Second, we derive two datasets with 7 and 10 clusters, called *news-multi-7* and *news-multi-10* [24, 25] respectively. Each cluster of the datasets contains a random sample of 100 documents from each of the corresponding newsgroup in the 20 newsgroups. *news-multi-7* and *news-multi-10* is generated for the purpose to study the effect of dataset size on the performance of the algorithms.

In summary, we have six datasets from 20 Newsgroups: (1) *original-20*: Original 20 Newsgroups dataset with about 20,000 messages, (2) *news-diff-3*, (3) *news-related-3*, (4) *news-similar-3*, (5) *news-multi-7*, and (6) *news-multi-10*.

3.1.2 The ACM Paper Collection

The second dataset is a collection of 580 papers in full text [21, 22], which were manually collected by the author from the Association for Computing Machinery (ACM) Digital Library[2]. We use the 1998 ACM Computing Classification System to label the categories[3]. In this thesis, we use the categories listed in Table 3.2. Categories H and I are related as they have overlapped areas such as "Data Mining" and "Text Clustering". Two datasets are derived from ACM paper collection. The first derived dataset *D2-D2&D3-D3* contains three clusters, papers of which are randomly sampled from only category $D2$, from both categories $D2$ and $D3$, and from only $D3$ category respectively. Each category has 87 papers in this dataset and is related to each other as they are all from D category. The second derived dataset *D-H-I* consists of three clusters. This dataset has 100 papers randomly sampled from each of D,H,I categories respectively.

Generally speaking, the categories assigned by paper authors are very coarse and cannot reflect the accurate topics of the papers. In addition, it is not uncommon that one paper is related to multiple topics and can be assigned to multiple categories. Therefore, we also use this dataset in our user study besides evaluating our newly

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

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

ACM category code	ACM category name
D	Software
D.2	Software Engineering
D.3	Programming Languages
H	Information Systems
I	Computing Methodologies

Table 3.2: Legend of ACM Categories

proposed algorithms using oracles. Compared to the messages in the 20 Newsgroups, the papers are much longer, and contain many scientific terms. Therefore, these papers should be thought of as a different type of documents from the newsgroup messages.

3.1.3 The 3-classic Dataset

The third 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. 100 documents from each category are sampled to form the reduced 3-classic dataset. This reduced dataset is named *3-classic-abstract*. Similar to *news-diff-3*, the topics of this dataset are quite different across categories.

Since these documents are abstracts, they are often even shorter than the messages in 20 Newsgroups and the document vectors may be more sparse.

3.1.4 The Webkb Dataset

The webkb dataset[5] is a collection of webpages from different universities. The webpages are homepages for students, faculty, projects, or courses. We use 250 documents from each category and name this dataset *webkb-sfcp-4*.

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

[5] `http://www.cs.cmu.edu/~webkb/`

3.1.5 The Sector Dataset

The sector dataset[6] is a collection of corporate webpages classified into a topic hierarchy with about 70 leaves. We use 10 categories, each of which includes approximately 100 documents. Those categories are: (1) basic.materials, (2) capital.goods, (3) consumer.cyclical, (4) oil.and.gas.integrated, (5) investment.services, (6) biotechnology.and.drugs, (7) hotels.and.motels, (8) communications.equipment, (9) railroad, and (10) water.utilities. This dataset is named *sector-multi-10*.

3.1.6 The Reuters21578 Dataset

The Reuters21578 corpus[7] [34] is the most widely used test collection for research on text categorization. It consists of news articles from different news categories. In this thesis, we use the top 10 most frequent categories, in which documents do not belong to multiple categories so that they are suitable for testing hard clustering algorithms. Each category includes about 100 documents. Those categories are: (1) acq, (2) coffee, (3) crude, (4) earn, (5) gold, (6) interest, (7) money-fx, (8) ship, (9) sugar, and (10) trade. This dataset is named *reuters-multi-10*.

3.1.7 Pre-processing of Documents

We pre-process each document by tokenizing the text into bags-of-words[8]. Then, we remove the stop words[9] and stem the remaining words using the Porter stemming algorithm [39]. Since we need to compare the performance of feature sets with different sizes in Chapter 4, the top m features ranked either by mean-TFIDF [50] for unsupervised clustering or the χ^2 method for supervision are selected to represent the documents (See details in Chapter 4). In other chapters (i.e., Chapters 5, 6, 7), the top 2000 features using mutual information between words and documents [15] are chosen to represent the documents using TFIDF values. Feature supervision are performed on these 2000 features (See details in Chapter 5, 6, 7 respectively). For the KMeans based algorithms, a feature vector for each document is constructed

[6] `http://people.cs.umass.edu/~mccallum/data.html`

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

[8] A word is defined as a sequence of alphabetic characters delimited by non-alphabetic characters.

[9] `http://jmlr.csail.mit.edu/papers/volume5/lewis04a/a11-smart-stop-list/english.stop`

with TFIDF weighting and then normalized. For the EM-NB (Naïve Bayes) based algorithms, the term frequency of the selected features is directly used in the related algorithms.

3.2 Evaluation Measures

In this thesis, we introduce user supervision to clustering techniques to improve clustering performance and produce personalized clusters. The algorithms are first proposed and evaluated based on oracles. Oracles are simulated users based on the underlying class labels of the standard datasets. Then, we conduct a user study to ask human users to provide user input to guide the clustering algorithms and evaluate the quality and personalization of the clusters based on clusters users manually create. We use clustering accuracy [8], normalized mutual information (NMI) [16], and Jaccard coefficient [8] to measure clustering performance for oracle-based evaluation as in previous work [11, 4, 9, 21, 22, 24, 25]. In the user study, we need to evaluate personalization of the same collection by different users. Therefore, we compare different clusterings by Rand Distance [41] in terms of cluster memberships of documents and by Jaccard Distance in terms of assigned features to clusters. In addition, we develop measures cohesiveness, separation, and F-Measure to evaluate clusters produced for users based on their manual organizations of the documents. Clustering accuracy and NMI are two external clustering validation metrics that estimate the clustering quality with respect to the underlying document class labels. Clustering accuracy computes the ratio between major chunks of clusters from the same classes and the size of the document collection while NMI measures the similarity between the clusters and underlying classes [30]. They measure how close the reconstructed clusters are to the underlying classes of the documents. Jaccard coefficient and Rand Index measure the similarity between two different clusterings while Jaccard Distance and Rand Distance measure the dissimilarity between two clusterings.

3.2.1 Clustering Accuracy

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

$$CACC(T, C) = \frac{\sum_t n(t)}{\sum_t |t|} = \frac{\sum_t n(t)}{N} \tag{3.1}$$

where N is the size of the document collection. As it is pointed out in Bekkerman et al. [8], 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.

3.2.2 Normalized Mutual Information

Normalized mutual information (NMI) [16] measures the shared information between the cluster assignments S and class labels L of documents. It is defined as:

$$NMI(S, L) = \frac{I(S, L)}{(H(S) + H(L))/2} \tag{3.2}$$

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:

$$H(L) = -\sum_{i=1}^{K} P(l_i) log_2 P(l_i) \tag{3.3}$$

$$H(S) = -\sum_{j=1}^{K} P(s_j) log_2 P(s_j) \tag{3.4}$$

$$I(S, L) = -\sum_{i=1}^{K}\sum_{j=1}^{K} P(l_i, s_j) log_2 \frac{P(l_i, s_j)}{P(l_i)P(s_j)} \tag{3.5}$$

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 $[0, 1]$.

3.2.3 Jaccard Coefficient

Jaccard coefficient [8] is 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:

$$J(y_1^c, y_2^c) = \frac{a}{a+b+c} \tag{3.6}$$

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

3.2.4 Rand Distance

Assume a document collection $\mathcal{D} = \{d_1, d_2, \ldots, d_n\}$ and two clusterings of $\mathcal{D}$, i.e., $\mathcal{X} = \{x_1, x_2, \ldots, x_r\}$ and $\mathcal{Y} = \{y_1, y_2, \ldots, y_s\}$, where x_i and y_j are subsets of $\mathcal{D}$. At the same time, we also have $x_i \cap x_j = \emptyset$ and $\cup_{i=1,\ldots,r} x_i = \mathcal{D}$ where $i, j \in \{1, \ldots, r\}$ and i is not equal to j, and $y_i \cap y_j = \emptyset$ and $\cup_{i=1,\ldots,s} y_i = \mathcal{D}$ where $i, j \in \{1, \ldots, s\}$ and i is not equal to j. We define the following quantities:

- a, the number of pairs of documents that are in the same cluster in $\mathcal{X}$ and $\mathcal{Y}$.
- b, the number of pairs of documents that are in different clusters in $\mathcal{X}$ and $\mathcal{Y}$.
- c, the number of pairs of documents that are in the same cluster in $\mathcal{X}$ but in different clusters in $\mathcal{Y}$.
- d, the number of pairs of documents that are in different clusters in $\mathcal{X}$ but in the same cluster in $\mathcal{Y}$.

The Rand Index, $\mathcal{RI}$, is:

$$\mathcal{RI} = \frac{a+b}{a+b+c+d} \tag{3.7}$$

and the Rand Distance, $\mathcal{RD}$, is:

$$\mathcal{RD} = 1 - \mathcal{RI} = \frac{c+d}{a+b+c+d} \tag{3.8}$$

Rand Index and Rand Distance measure the similarity and the dissimilarity between two clusterings respectively. The values of Rand Index and Rand Distance are in the interval $[0, 1]$.

3.2.5 Cohesiveness, Separation, and F-Measure

In the user study, each user manually groups the same document collection into a set of clusters based on their own perception during the exploration. Therefore, each user might have a very different organization from each other. In this sense, the clusters produced by clustering algorithms for each user should be evaluated against the user's manually organized clusters of the document collection. Instead of computing only the whole similarity of two clusterings, we would like to compute and compare the cohesiveness and separation of the generated clusters based on the user manual organized clusters. Cohesiveness measures how closely related the documents in the same cluster are to each other while separation measures how widely separated the documents from different clusters are from each other. To this end, we developed measures *coh*, *sep*, and F-Measure [23] to evaluate the clusters produced for this user with/without supervision. Those measures treat a user's manual organization as the gold standard. Assuming the gold standard partition $\mathcal{G} = \{g_1, g_2, \ldots, g_k\}$ and a clustering $\mathcal{C}$ produced by a clustering algorithm, we define the following quantities:

- a', the number of pairs of documents that are in the same cluster in $\mathcal{G}$.
- b', the number of pairs of documents that are in the same cluster in $\mathcal{G}$ and $\mathcal{C}$.
- c', the number of pairs of documents that are in different clusters in $\mathcal{G}$.
- d', the number of pairs of documents that are in different clusters in $\mathcal{G}$ and $\mathcal{C}$.

The cohesiveness of $\mathcal{C}$ with respect to $\mathcal{G}$, *coh*, is defined as:

$$coh = \frac{b'}{a'} \tag{3.9}$$

The separation of $\mathcal{C}$ with respect to $\mathcal{G}$, *sep*, is defined as:

$$sep = \frac{d'}{c'} \tag{3.10}$$

and finally F-Measure with respect to $\mathcal{G}$, F, is defined as:

$$F = 2 \times \frac{coh \times sep}{coh + sep} \tag{3.11}$$

where *coh* measures the cohesiveness of $\mathcal{C}$ while *sep* measures the separation of $\mathcal{C}$. Both measures are based on a user's manual organization $\mathcal{G}$. The values of *coh*, *sep*, and F are in the interval $[0, 1]$.

3.2.6 Jaccard Distance

Given two sets $\mathcal{A}$ and $\mathcal{B}$, the Jaccard Index, $\mathcal{JI}$, is:

$$\mathcal{JI} = \frac{|\mathcal{A} \cap \mathcal{B}|}{|\mathcal{A} \cup \mathcal{B}|} \tag{3.12}$$

and the Jaccard Distance, $\mathcal{JD}$, is:

$$\mathcal{JD} = 1 - \mathcal{JI} = \frac{|\mathcal{A} \cup \mathcal{B}| - |\mathcal{A} \cap \mathcal{B}|}{|\mathcal{A} \cup \mathcal{B}|} \tag{3.13}$$

Jaccard Index, $\mathcal{JI}$, measures similarity between two sets while Jaccard Distance, $\mathcal{JD}$, measures dissimilarity between two sets. Given two clusterings $\mathcal{X}$ and $\mathcal{Y}$ of a document collection $\mathcal{D}$ (Section 3.2.4), $\mathcal{X}_w = \{x_{w1}, x_{w2}, \ldots, x_{wr}\}$ and $\mathcal{Y}_w = \{y_{w1}, y_{w2}, \ldots, y_{ws}\}$ are the sets of keywords assigned to each cluster in $\mathcal{X}$ by one user and in $\mathcal{Y}$ by another user, i.e., x_{wi} and y_{wj} are the keywords assigned to cluster x_i and y_j respectively. We define two dissimilarity measures between $\mathcal{X}_w$ and $\mathcal{Y}_w$. One measure $\mathcal{JD}_a$ measures dissimilarity between $\mathcal{X}_w$ and $\mathcal{Y}_w$ without consideration of the cluster labels of the assigned keywords, i.e., $\mathcal{A} = \cup_{i=1,2,\ldots,r} x_{wi}$ and $\mathcal{B} = \cup_{j=1,2,\ldots,s} y_{wj}$ in Eq. 3.13. The other measure $\mathcal{JD}_b$ measures dissimilarity between $\mathcal{X}_w$ and $\mathcal{Y}_w$ with cluster labels of assigned keywords being considered. $\mathcal{JD}_b$ is defined as:

$$\mathcal{JD}_b = \frac{\sum_{i=1}^{r} \min_{j=1,2,\ldots,s} \mathcal{JD}(x_{wi}, y_{wj}) + \sum_{j=1}^{s} \min_{i=1,2,\ldots,r} \mathcal{JD}(y_{wj}, x_{wi})}{r + s} \tag{3.14}$$

In this measure, we compute the average distance between a cluster (e.g. x_{wi} in $\mathcal{X}$) and its closest match (i.e. $\sum_{i=1}^{r} \min_{j=1,2,\ldots,s} \mathcal{JD}(x_{wi}, y_{wj})$) in the other clustering (i.e. $\mathcal{Y}$). The closest match y_{wj} from the other clustering $\mathcal{Y}$ is the cluster that has minimum distance ($\sum_{i=1}^{r} \min_{j=1,2,\ldots,s} \mathcal{JD}(x_{wi}, y_{wj})$) from the original cluster x_{wi}. In this way, cluster

labels are considered when the measure is computed. Note that the closest match relationship is not symmetrical, i.e., with x_{wi}'s closest match in $\mathcal{Y}$ being y_{wj}, the closest match of y_{wj} in $\mathcal{X}$ could be x_{wk}, where k might not be identical to i. The values of $\mathcal{JD}$ and $\mathcal{JD}_b$ are in the interval $[0, 1]$.

Chapter 4

Interactive Feature Selection

This chapter presents how feature supervision in the form of indicating whether a feature is useful for document clustering can be incorporated into unsupervised document clustering [21, 22]. We ask users to confirm features out of a ranked list based on the intermediate clusters and reweight the "accepted" features. An interactive feature selection framework for document clustering is proposed based on this idea. The underlying intuition is that reweighted discriminating features can help guiding clustering algorithms to group similar documents together.

4.1 Introduction

As we mentioned in Chapter 1, traditional unsupervised clustering techniques output potential clusters with minimum user effort but users are often dissatisfied with the generated clusters because they are neither intuitive nor reflect user's point of view. In this chapter, we seek to determine whether clusters better matching user expectation may be generated with some supervision by users. 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 applies 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. Through optimizing the constrained loss function and 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. Users 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, users can also give feedback at the feature level. Therefore,

instead of asking users to label enough documents for an effective feature selection, we ask users to directly label features for clustering.

In this chapter, we explore how user supervision will perform when it is used for feature selection. The work is different from previous semi-supervised clustering approaches as it asks users 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 [40].

An overview of the framework we use in our study 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 users for labeling. Users must label every feature as "accept" or "don't know" according to their understanding of the document collection. The features users label 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, users are 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 users so that users are 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, users label documents in AL but they label features in our framework. Third, uncertain sampling [35] 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 simulated users instead of human users for practicality. Simulated users label features based on document labels (see Section 4.3.4 for details). In addition, both of simulated and human users may make mistakes. More importantly, simulated users can be employed repeatedly. In this chapter, we use KMeans 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. Our interactive clustering framework with interactive feature selection is as follows:

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.

4.2 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.

4.2.1 KMeans

KMeans [10] 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, which is defined as:

$$J = \sum_{i=1}^{N} \sum_{j=1}^{K} r_{ij} \|x_i - \mu_j\|^2 \tag{4.1}$$

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

$$r_{ij} = \begin{cases} 1 & \text{if data point } i \text{ is assigned to cluster } j \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

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.

4.2.2 Multinomial Naïve Bayes Model

Multinomial Naïve Bayes model [38] is a commonly used probabilistic model for text clustering, which assumes a document as a vector of words, with each word generated

Algorithm 1 KMeans [10]

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

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

$$p(c_j) = \frac{\sum_{i=1}^{|\mathcal{D}|} P(c_j|d_i)}{|\mathcal{D}|} \tag{4.3}$$

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

$$p(w_t|c_j) = \frac{1 + \sum_{i=1}^{|\mathcal{D}|} N(w_t, d_i) \cdot P(c_j|d_i)}{|\mathcal{V}| + \sum_{s=1}^{|\mathcal{V}|} \sum_{i=1}^{|\mathcal{D}|} N(w_s, d_i) \cdot P(c_j|d_i)} \tag{4.4}$$

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:

$$P(c_j|d_i) = \frac{P(c_j)P(d_i|c_j)}{\sum_{r=1}^{|C|} P(c_r)P(d_i|c_r)} = \frac{P(c_j)\prod_{k=1}^{|d_i|} P(w_{d_i,k}|c_j)}{\sum_{r=1}^{|C|} P(c_r)\prod_{k=1}^{|d_i|} P(w_{d_i,k}|c_r)} \tag{4.5}$$

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 Expectation-Maximization (EM) algorithm is a widely used iterative algorithm for maximum likelihood estimation for problems involving missing data [14]. Therefore EM algorithm is commonly used to assign cluster labels to the documents during clustering. 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* algorithmname=*EM-NB*,description=Expectation-Maximization with Multinomial Naïve Bayes model, is formed by applying EM algorithm to Naïve Bayes classifier. In *EM-NB* algorithm, Eq. 4.3 and Eq. 4.4 are evaluated in the M step and Eq. 4.5 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 such as 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 another probabilistic model such as *EM-NB* itself, in which case its value is between 0 and 1.

Algorithm 2 *EM-NB*, i.e. Multinomial Naïve Bayes [38]

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:

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. 4.3
- Word conditional probabilities $P(w_t|c_j)$ using Eq. 4.4

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. 4.5
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 # of iterations is reached

4.2.3 Mean-TFIDF Feature Selection Technique

Mean-TFIDF feature selection technique [50] 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 $mean\text{-}TFIDF_j = \frac{\sum_i TFIDF_{(i,j)}}{|\mathcal{D}|}$.

4.2.4 χ^2 Class-Based Feature Selection Technique

The χ^2 value of a feature indicates whether the feature is significantly correlated with a class [43]. 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 follows:

$$\chi^2 = \sum_{c,i} \frac{[k_{c,i} - nPr(C = c) \cdot Pr(I = i)]^2}{nPr(C = c) \cdot Pr(I = i)} \tag{4.6}$$

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.

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 users 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 $\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$

Table 4.1: Definition of Variables

4.3 Methodology

In this section, we introduce the interactive feature selection and clustering frameworks, an approach to investigate the effect of user effort, and cluster evaluation measures. We also give details about the simulated users. In Table 4.1, we define the variables we use in this chapter.

4.3.1 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 users. This is one of the motivations for inviting users to label features during clustering. At each iteration, the features presented to users 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. Users give 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 3. All features accepted by users 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 3 Interactive Feature Selection with User Supervision

Input: m — size of feature set for document clustering, f — # of features presented to users 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 — feature selected for next clustering iteration.

Method:

1: $FS^{t}_{accepted} \leftarrow FS^{t-1}_{accepted}$
2: $FL^{all} \leftarrow$ Rank all 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 users 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: $FS^m \leftarrow FS^m \cup \{(f+i)^{th} feature\}$
15: ***end for***

Algorithm 4 Interactive Document Clustering Framework with Feature Selection

Input: $\{d_1, d_2, \ldots, d_N\}$ — document vectors, s — seed for randomization, f — # of features presented to users 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 EM-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 3
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

4.3.2 Interactive Document Clustering Framework

After a new feature set with user supervision is obtained as described in Algorithm 3, 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 4. At the beginning, clusters are obtained from traditional KMeans with the feature set selected by mean-TFIDF [50]. 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 when the user chooses to stop, or when the generated clusters do not change, or when the maximum number of iterations is reached. The user may choose to stop when either generated clusters or the feature set is satisfactory.

4.3.3 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 [13]. 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 simulate users, 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 can be found in Rigutini and Maggini [43]. The larger the global value is, the better the feature is in discriminating among

clusters.

4.3.4 Simulated Users

In our research, the user supervision is used to identify useful features for clustering, namely, feature selection. Users are asked to select good features in the interactive feature selection framework. Our goal in this chapter 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. Unlike human users, a simulated user (also called oracle) based on the class labels of the data set is fast, costs little and is sufficient for an initial proof-of-concept demonstration.

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 simulated users, 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 users are 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.

In this chapter, we use feature supervision *useful-list* as described in Table 1.2. The algorithms and the corresponding supervision methods are summarized in Section 1.4, Table 1.1 and Table 1.2.

4.3.5 Feature Sets

By using the underlying clustering algorithms, we compare interactive feature selection framework with unsupervised feature selection technique. 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 4.2. All feature sets in Table 4.2 have the same size $m = 600$ except

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

Table 4.2: Definition of Feature Sets

FS_{basic} whose size depends on the number of all extracted features.

4.3.6 Effect of User Effort

In this section, we investigate the effect of user effort on the 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, the size of the feature set f is given as an input parameter of the interactive feature selection and clustering framework. The f value can be thought of as the unit of effort as f features are confirmed by users at each iteration. Therefore, total amount of user input 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 users. Finally, user effort efficiency $eff\text{-}eff$ can be defined as:

$$eff\text{-}eff = \frac{f_{accepted}}{f_{total}} \tag{4.7}$$

The larger $eff\text{-}eff$ is, the larger portion of the presented features is accepted, which may be good for clustering.

Feature Reweighting Since feature reweighting can boost classification performance in active learning [40], feature reweighting is adopted in the interactive clustering framework. Different underlying clustering algorithms have their own method of integrating feature re-weighting. In this chapter, 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

$$P(c_j|d_i) = \frac{P(c_j)P(d_i|c_j)}{\sum_{r=1}^{|C|} P(c_r)P(d_i|c_r)} = \frac{P(c_j)\prod_{k=1}^{|d_i|} P(w_{d_i,k}|c_j)}{\sum_{r=1}^{|C|} P(c_r)\prod_{k=1}^{|d_i|} P(w_{d_i,k}|c_r)} \tag{4.8}$$

for a given document [36]. g affects Eq. 4.8 through the feature term frequency:

$$p(w_t|c_j) = \frac{1 + \sum_{i=1}^{|\mathcal{D}|} \boldsymbol{g}_t \cdot N(w_t, d_i) \cdot P(c_j|d_i)}{|\mathcal{V}| + \sum_{s=1}^{|V|}\sum_{i=1}^{|\mathcal{D}|} \boldsymbol{g}_s \cdot N(w_s, d_i) \cdot P(c_j|d_i)} \tag{4.9}$$

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 :

$$|g_s| = \begin{cases} g & \text{if } w_s \text{ is accepted} \\ 1 & \text{otherwise} \end{cases} \tag{4.10}$$

In our experiments, g is an integer between 1 and 10. Using the above definitions, we study the user effort in the form of feature supervision on clustering performance by answering the following four questions:

1. How does the value of g affect clustering performance?
2. How does clustering performance change in relation to f?
3. How does clustering performance change in relation to f_{total}?
4. How does feature reweighting affect clustering performance?
5. How does feature reweighting affect user effort?

4.4 Experimental Evaluation

In this work, we use datasets: (1) *news-diff-3*, (2) *news-related-3*, (3) *news-similar-3*, (4) *D2-D2&D3-D3*, (5) *D-H-I*, and (6) *3-classic-abstract*. We employ three evaluation

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 4.3: Comparison of KMeans with Different Feature Sets Defined in Table 4.2, namely, FS_{basic}, $FS_{mean\text{-}TFIDF}$, $FS_{iterative}$, $FS_{interactive}$, $FS_{reference}$. The interactive feature selection framework can produce better clusters than unsupervised feature selection methods. The performance of the clustering algorithms improves significantly when it moves from column FS_{basic} to column $FS_{reference}$ except where the performance measures are bold. Accuracy and NMI are calculated based on the underlying class labels while Jaccard Coefficient is computed on based on the clustering produced with the reference feature set. The Jaccard Coefficient of the same two clusterings is always 1.

measures: (1) Clustering Accuracy, (2) NMI, and (3) Jaccard Coefficient. 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 explore how clustering performance depends on user effort. In this chapter, we present a subset of experimental results to illustrate our points. The results for all other datasets have similar patterns as presented here and may be found in Appendix B.

4.4.1 Performance of Different Feature Sets

In this section, we compare and discuss the performance of the same underlying algorithm with different feature sets.

Each pair of one underlying algorithm and one feature set was run 36 times[1] 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 4.3 for KMeans and Table 4.4 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 users, e.g. f is between 100 and 300.

As shown in Table 4.3 and Table 4.4, the interactive feature selection framework can produce better clusters than other unsupervised feature selection methods. In these tables, the performance of the clustering algorithms improves significantly[2] when it moves from column FS_{basic} to column $FS_{reference}$ except where the performance measures are bold. In Table 4.4, the exception is between $FS_{mean\text{-}TFIDF}$ 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 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 interactive feature selection and clustering framework 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 (Eq. 3.6 in Section 3.2.3).

[1] The number of times we ran the algorithms was randomly chosen to be large enough for computing statistical significance.

[2] Two-tailed tests were used for each comparison with $p = 0.05$

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

Table 4.4: Comparison of *EM-NB* with Different Feature Sets Defined in Table 4.2, namely, FS_{basic}, $FS_{mean\text{-}TFIDF}$, $FS_{iterative}$, $FS_{interactive}$, $FS_{reference}$. The interactive feature selection framework can produce better clusters than unsupervised feature selection methods. The performance of the clustering algorithms improves significantly when it moves from FS_{basic} to $FS_{reference}$ except where the measures are bold. Jaccard Coefficient is computed on based on the clustering produced with the reference feature set. The Jaccard Coefficient of the same two clusterings is always 1.

4.4.2 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 Table 4.3 and Table 4.4. 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 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 Figure 4.1 and 4.2. The performance of both KMeans and *EM-NB* increases initially as the size

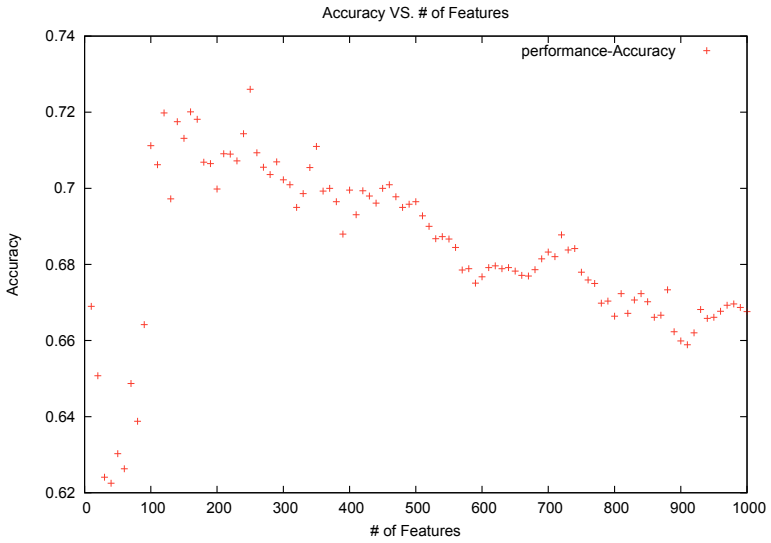


Figure 4.1: Effect of feature set size on *news-related-3* with *KMeans* and *reference feature set*. Reference feature set with various sizes are used, which are selected by the χ^2 based on the underlying document class labels.

of feature set gets larger when applied to all datasets. Maximum performance can be reached with different feature set sizes but $200 \leq m \leq 400$ 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. 4.2) to KMeans (Fig. 4.1) 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.

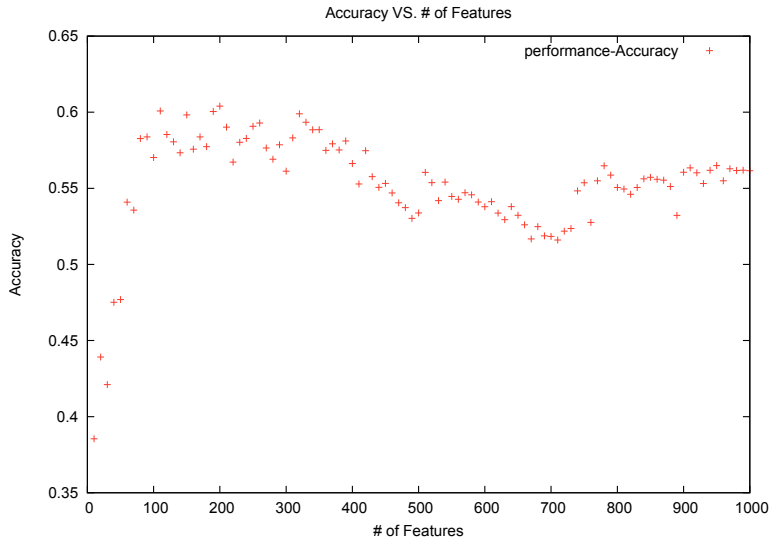


Figure 4.2: Effect of feature set size on *news-similar-3* with EM-NB and *reference feature set*. Reference feature set with various sizes are used, which are selected by the χ^2 based on the underlying document class labels.

4.4.3 Effect of User Effort

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

Effect of user effort on *news-related-3* dataset is shown in Fig. 4.5 for KMeans and Fig. 4.6 for *EM-NB* while effect of user effort on *news-similar-3* dataset is demonstrated in Fig. 4.7 for KMeans and Fig. 4.8 for *EM-NB*. The last four questions brought up in Section 4.3 will be answered one by one as follows.

For all datasets, the user effort spent in terms of f_{total} increases until the algorithm converges increases with f, the number of features presented to users in each iteration as seen in Fig. 4.3. We also note that the effort efficiency declines when more features displayed in each iteration, as seen in Fig. 4.4. This may be due to the fact that the more features are displayed each time, and the higher proportion of features displayed are not in the 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 the χ^2 is not perfect for ranking features (see Section 4.4.2 for more discussions). The observation that the effort efficiency is about 1 when f

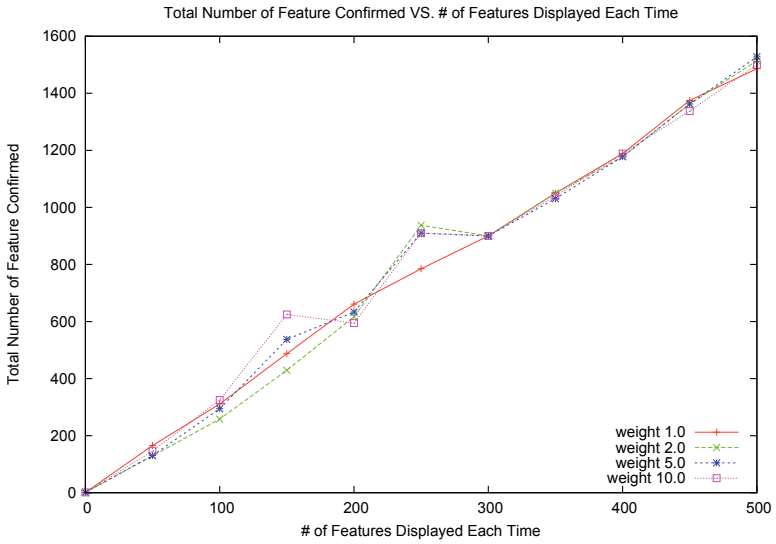


Figure 4.3: f vs. f_{total}, *EM-NB* on *news-diff-3* dataset. f is the number of features presented to users at each iteration while f_{total} is the number of features presented to users after the clustering finishes. Weights are used to reweight user accepted features.

is small (Fig. 4.4) tells that most of the displayed features are accepted as useful for clustering by users. 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 users (Fig. 4.7 and Fig. 4.6). However, when the interactive clustering framework with KMeans works with news-related dataset and ACM (*D-H-I*) dataset, the clustering performance declines after a certain amount of effort is provided. 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-emphasized, which has a negative effect on interactive clustering framework with *EM-NB* (Fig. 4.6 and Fig. 4.8). For

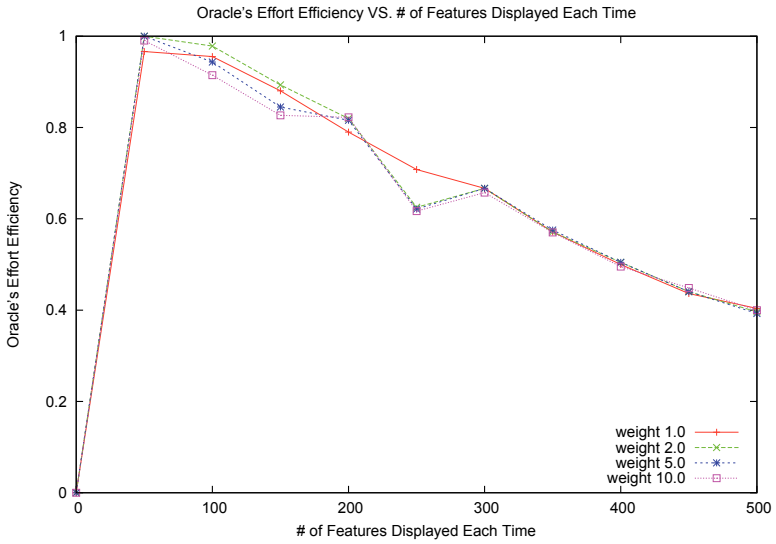


Figure 4.4: Effort-Efficiency, *EM-NB* on *news-similar-3* Dataset. Effort-Efficiency measures the percentage of features which are accepted for clustering out of all features presented to users. The oracle is the simulated user. Weights are used to reweight user accepted features.

the interactive framework with *EM-NB*, probabilities of features in the feature set for clustering are affected through Eq. 4.9 and the performance in terms of NMI declines first and climbs up when more features are accepted by users.

4.4.4 Selection of Weight g

In our experiments, we tried different values of weight g (see details in Section 4.3.6) 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. 4.5) or help reach maximum clustering performance earlier (Fig. 4.7), 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.

4.4.5 KMeans and *EM-NB* on the Same Datasets

We also compare the interactive framework with KMeans versus *EM-NB* as the underlying algorithm on the same datasets (Fig. 4.9). 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 users, 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.

4.4.6 KMeans or *EM-NB* on Different Datasets

We compare the framework with KMeans or *EM-NB* on the three newsgroups subdatasets with respect to the same number of features. When the same user effort in terms of number of features confirmed by users 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. 4.10).

4.5 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 premature algorithm convergence before good clusters are achieved and large f values may require more user effort without performance improvement, allowing users to change the 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 introduced. By storing the history of clustering, the users can roll back to previous clusterings and make new decisions about feature selections from there.

3. Visualization of clusters. In order to assist users in judging 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. In this way, users 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.

4.6 Summary

Users can interact with the document clustering process at either the document-level or the feature-level. Existing semi-supervised clustering algorithms improve performance by exploiting the constraints between documents defined by users. In this chapter, we have focused on user-guided clustering at the feature-level.

We designed and created a new framework that enables users to guide the clustering process by selecting features which are meaningful to them. The framework interleaves interactive feature selection and clustering iteratively until users choose to stop or the underlying algorithm reaches its terminating conditions. At each iteration, users are presented a list of the top f features ranked by the cluster-based feature selection of the most recent clusters. Since the ranking is based on the recent clusters, meaningful features are likely to have higher ranking. Users rate each of those features by selecting one of the two options: "accept" and "don't know". A revised feature set including the features users "accept"-ed and highly ranked features are used to re-cluster the documents.

This novel method was evaluated by comparison with three different unsupervised feature selection techniques over six different document datasets. The novel method performed significantly better in all cases.

We also studied the effect of user effort on clustering performance in the new framework. Our experiments indicate that a certain number of features must be labeled by users 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 these few over-emphasized features.

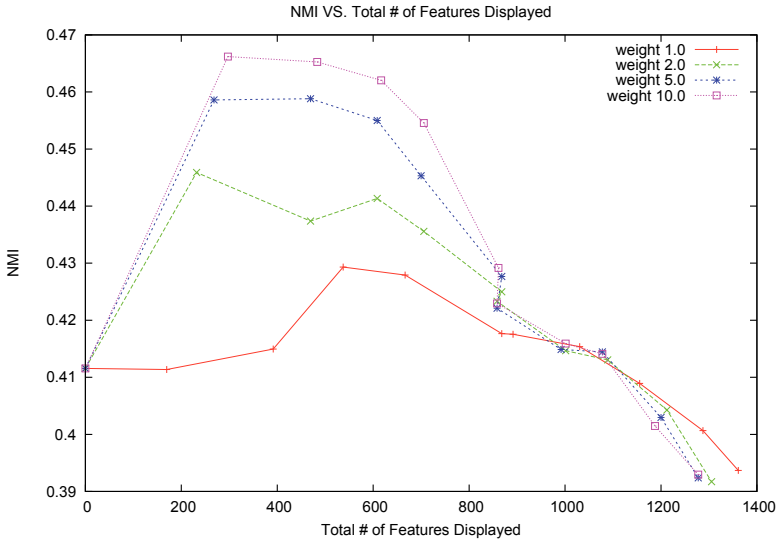


Figure 4.5: Effect of User Effort, KMeans on *news-related-3* Dataset. The clustering performance increases with more features confirmed (user effort), and then decreases after a certain number of features are confirmed. Noisy features are introduced by the simulated users in the later phase, which downgrades the clustering performances. Weights are used to reweight user accepted features.

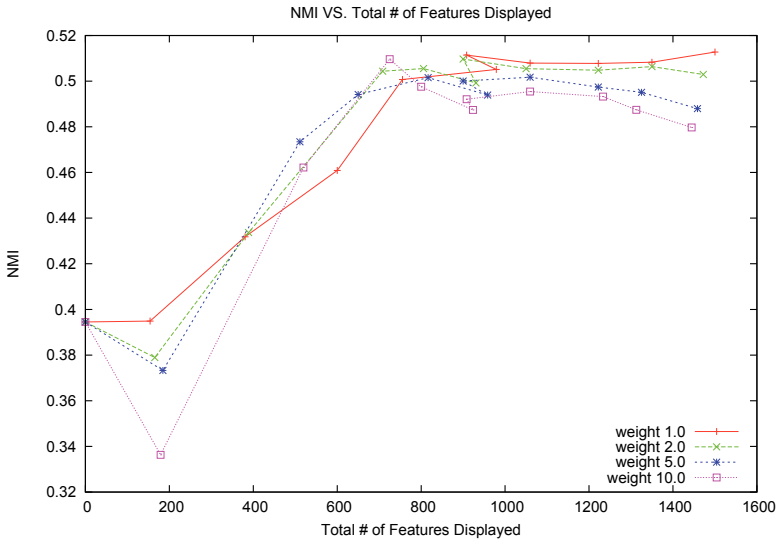


Figure 4.6: Effect of User Effort, *EM-NB* on *news-related-3* Dataset. The more user effort, namely, more features confirmed by users, the better clustering performance. Weights are used to reweight user accepted features.

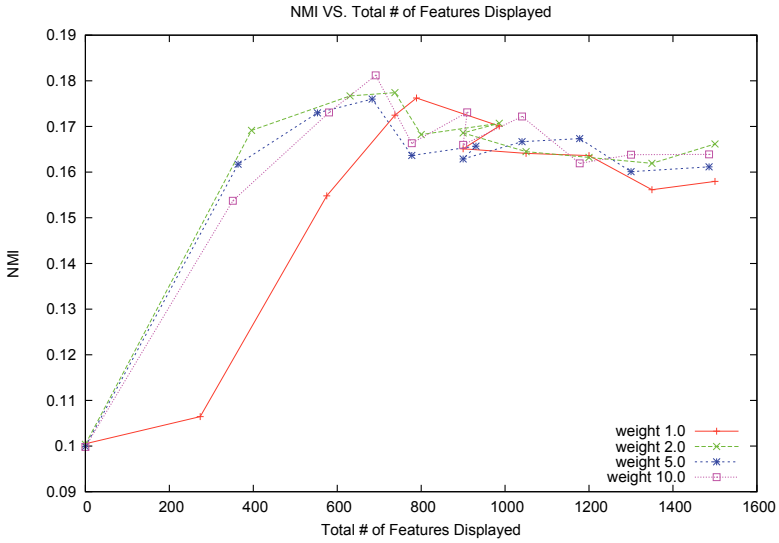


Figure 4.7: Effect of User Effort, KMeans on *news-similar-3* Dataset. The more user effort, namely, more features confirmed by users, the better clustering performance. Weights are used to reweight user accepted features.

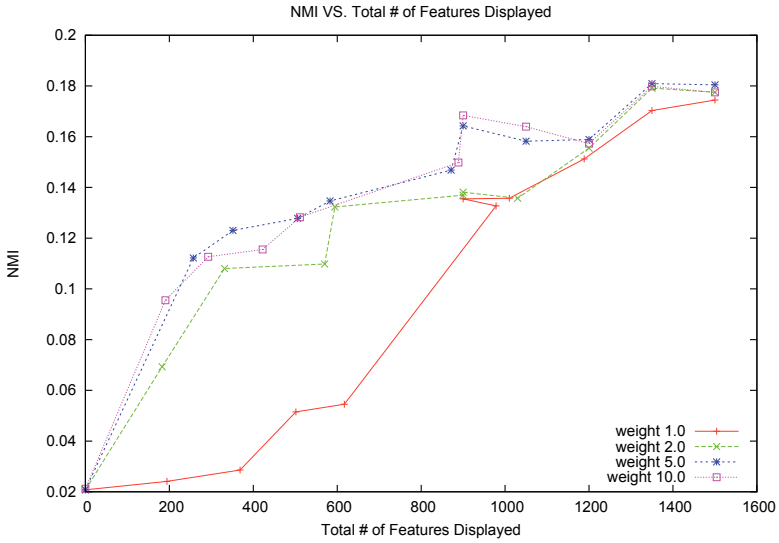
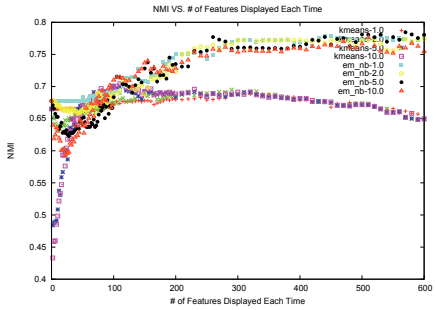
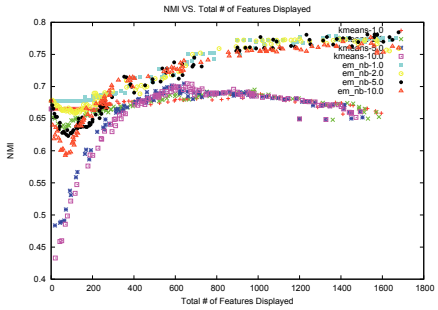


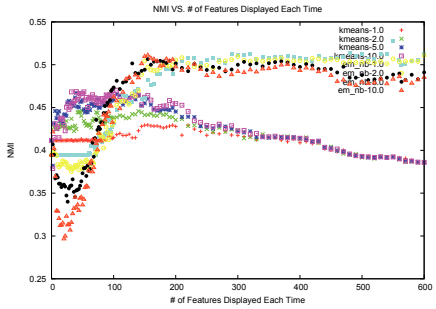
Figure 4.8: Effect of User Effort, *EM-NB* on *news-similar-3* Dataset. The more user effort, namely, more features confirmed by users, the better clustering performance. Weights are used to reweight user accepted features.



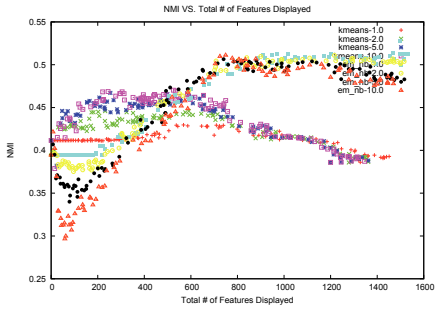
(a) KMeans and *EM-NB* on *news-diff-3* dataset: (x axis-f) vs. (y axis-NMI)



(b) KMeans and *EM-NB* on *news-diff-3* dataset: (x axis-f_{total}) vs. (y axis-NMI)

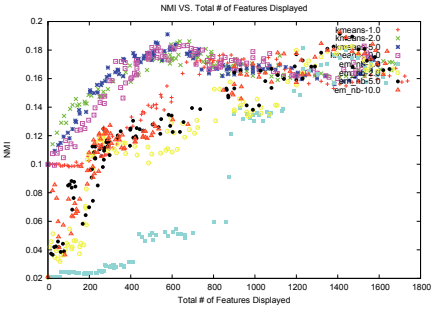


(c) KMeans and *EM-NB* on *news-related-3* dataset: (x axis-f) vs. (y axis-NMI)



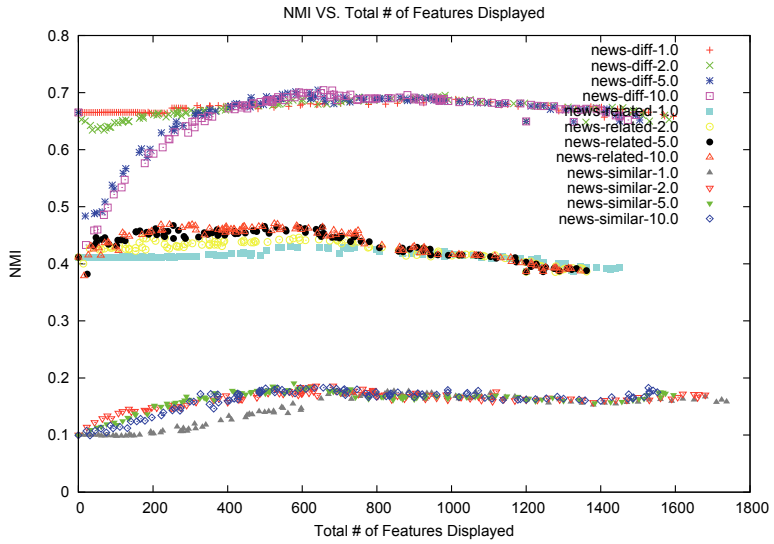
(d) KMeans and *EM-NB* on *news-related-3* dataset: (x axis-f_{total}) vs. (y axis-NMI)

(e) KMeans and *EM-NB* on *news-similar-3* dataset: (x axis-f) vs. (y axis-NMI)

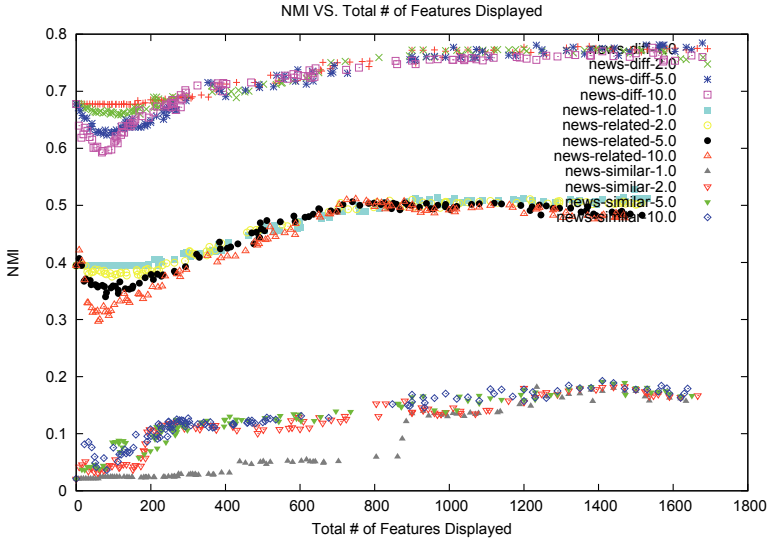


(f) KMeans and *EM-NB* on *news-similar-3* dataset: (x axis-f_{total}) vs. (y axis-NMI)

Figure 4.9: The interactive clustering framework (Algorithm 4) with different underlying algorithms on the same newsgroups datasets. Within the interactive clustering framework, *EM-NB* works better than KMeans on news-diff dataset. When applied to news-related and news-similar datasets, KMeans outperforms *EM-NB* when only a few features are confirmed by users. 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 are confirmed as "good" for clustering. The noisy features have more negative effect on *EM-NB* than on KMeans.



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



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

Figure 4.10: The interactive clustering framework (Algorithm 4) with the same underlying algorithm on *news-diff-3*, *news-related-3*, *news-similar-3* datasets. With interactive feature selection, *news-similar-3* dataset is still the most difficult one to clustered and *news-diff-3* dataset remains the easiest one with the same user effort, namely, the number of features confirmed by users.

Chapter 5

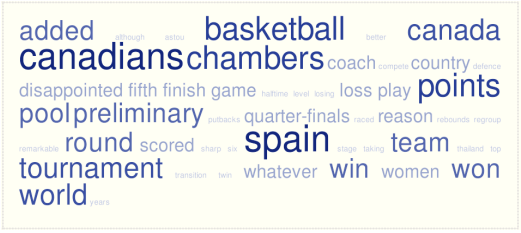
Dual Supervision through Feature Reweighting

In the previous chapter, we explored how feature supervision can affect clustering performance through feature reweighting. In the next two chapters, we will explore how user supervision at both document-level and feature-level helps document clustering. The joint use of user supervision at the document and feature levels is referred to as dual supervision. In this chapter, we incorporate feature supervision through feature reweighting to influence the document representations.

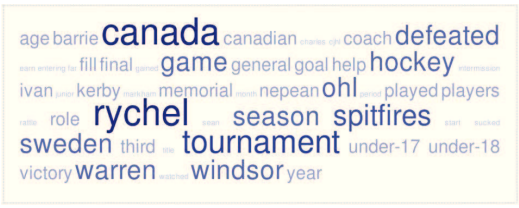
5.1 Introduction

In Section 1.4, we defined two types of user supervision, i.e., document supervision and feature supervision for document clustering. In this chapter, we explore the document supervision and feature supervision in the forms described next. *Document Supervision* involves labeling documents, i.e., assigning a document to a cluster or specifying a "must-link" or "cannot-link" [53] for a pairwise constraint between two documents. *Feature Supervision* involves labeling features, i.e., indicating whether a feature discriminates clusters. Note that a labeled feature is not assigned to a cluster but is known for its usefulness for clustering.

Traditional semi-supervised clustering, which uses both labeled and unlabeled instances, has shown its usefulness in generating clusters matching user expectation.



(a) Text Cloud of Document A about Canadian Basketball



(b) Text Cloud of Document B about Canadian Hockey

Figure 5.1: Text Clouds of Two Documents

User supervision usually takes the form of document supervision. However, the user can also provide alternative forms of user supervision such as feature supervision involving labeling features for document clustering. Since this research focuses on document clustering, we may use *instance* and *document*, *feature* and *word* interchangeably. Labeling documents and words can be performed at the same time, with little additional effort for labeling words, if an appropriate document visualization is used, such as text clouds [32]. While the user assigns a document to a cluster or specifies a pairwise constraint based on the document's text cloud, the words appearing in the text cloud can also be labeled by being clicked or highlighted.

Example 5-1. Documents A and B in Fig. 5.1 can be specified as a "must-link" when clustered by country but a "cannot-link" when clustered by sport. Correspondingly, the user would label the words "Canada" "Canadian" "Spain" in the first case but "basketball" "points" "hockey" "rychel" (last name of a hockey player) in the latter case. □

Different labeled words reflect different organizations and the user forms his point of view based on the perception of the words in the text clouds. It has been argued that document supervision and feature supervision are complementary rather than completely redundant and their joint use has been called *dual supervision* [2].

In this chapter, we assume that the user labels a document or establishes a pairwise constraint by reading a fraction of the documents' contents. At the same time, the user can label a word by indicating (e.g. highlighting) whether it discriminates among clusters. The text cloud could be used to visualize the fraction of the content the user reads and augment the labeling. Since the labeled features are not associated with specific clusters, we incorporate them into the semi-supervised clustering through feature reweighting. Despite its simplicity, our proposed method is demonstrated to be quite robust and effective under different experimental settings. We enhance several semi-supervised clustering algorithms mentioned in Section 2.1. We also compare those algorithms using only labeled documents to our proposed methods using only labeled features. Finally, we did experiments with that the user makes mistakes in labeling features, that the user only reads a fraction of a document content, and that various numbers of documents are labeled per cluster. The experimental results on several text datasets imply that far fewer document pairwise constraints

are required for better clustering performance when feature supervision is performed while labeling document pairs. Therefore clusters of better quality can be produced with less user effort when dual supervision is used for document clustering.

5.2 Background

In this section, we introduce pairwise document constraints and present three semi-supervised clustering algorithms, each of which is from a different category in Section 2.1. COP-KMeans is a constraint-based method, and Seeded-KMeans uses labeled documents as cluster seeds for document clustering, while distance metric learning is from the distance-based methods.

5.2.1 Pairwise Document Constraints

Two types of pairwise constraints are used for traditional semi-supervised clustering:

- *"must-link"* constraints specify that two documents have to be placed in the same cluster.

- *"cannot-link"* constraints specify that two documents cannot be placed in the same cluster.

Consequently, there are usually two sets defined: $\mathcal{M}$ is the set of "must-link" constraints and $\mathcal{C}$ is the set of "cannot-link" constraints. Both "must-link" and "cannot-link" constraints are symmetric. Ideally, the must-link constraints are transitive and transitive closures can be derived.

5.2.2 KMeans

KMeans [10] is a clustering algorithm based on iterative assignments of data points and partitions a dataset into K clusters so that the average squared distance between the data points and the cluster centers are locally minimized. More details about KMeans is available in Section 4.2.1.

5.2.3 COP-KMeans

COP-KMeans [53] is a constraint-based method, where the user supervision is provided in the form of must-link and cannot-link constraints. During the clustering process, all the constraints should be satisfied. Otherwise, COP-KMeans fails, in which no clusters are produced. COP-KMeans is presented in Algorithm 5.

5.2.4 Seeded-KMeans

Given a dataset $\mathcal{X}$, KMeans can partition it into K clusters $\{\mathcal{X}_l\}_{l=1}^{K}$. KMeans is usually initialized with randomly selected cluster centers. It was observed that Seeded-KMeans [4] with cluster centers initialized with centroids derived from small sets of labeled instances could improve clustering performance significantly. To this end, we define the seed set $\mathcal{S} \subseteq \mathcal{X}$ be the subset of data points as follows: for each $x_i \in \mathcal{S}$, the user provides the cluster $\mathcal{X}_l$ to which it belongs. We assume that there is at least one data point x_i for each cluster $\mathcal{X}_l$. Note that there is a K-disjoint-partitioning $\{\mathcal{S}_l\}_{l=1}^{K}$ of the seed set $\mathcal{S}$ such that all $x_i \in \mathcal{S}_l$ belongs to $\mathcal{X}_l$ according to the supervision. In Seeded-KMeans [4], the seed set $\mathcal{S}$ is used to initialize the KMeans algorithm. In this method, each cluster center μ_l is initialized by the centroid of $\mathcal{S}_l$ instead of a randomly picked centroid. Note that the seed set is only used in the initialization step and is not used in the remaining steps of KMeans. Therefore, the seeds can change their cluster memberships during the subsequent clustering steps. In Constrained-KMeans [4], the seed set is also used to initialize the KMeans algorithm. However, unlike Seeded-KMeans, the memberships of the seeds are not re-computed and kept unchanged in the subsequent clustering steps. Compared to Seeded-KMeans, Constrained-KMeans is more appropriate when there are no or very few noisy seeds. In fact, Constrained-KMeans can be thought of as a combination of COP-KMeans and Seeded-KMeans. In this paper, we assume the seed set without noise. Seeded-KMeans and Constrained-KMeans are described in Algorithm 6.

5.2.5 Xing-KMeans

Many clustering algorithms, including K-Means, critically rely on a good metric for the input data. A better metric may be learned from the labeled document pairwise

Algorithm 5 COP-KMeans [53]

Input: Set of data points $\mathcal{X}$, must-link set $\mathcal{M}$, cannot-link set $\mathcal{C}$

Output: K clusters $\{\mathcal{X}_l\}_{l=1}^{K}$

Method:

Randomly initialize the cluster centers with $\{\mu_l^0\}_{l=1}^{K}$
repeat
  Based on $\{\mu_l^t\}$, assign each data point x_i to the closest cluster $\mathcal{X}_l^{t+1}$ for which VIOLATE-CONSTRAINTS(x_i,$\mathcal{X}_l$,$\mathcal{M}$,$\mathcal{C}$) returns false. If no such cluster exists, COP-KMeans fails and returns {}. At the end, get $\{\mathcal{X}_l^{t+1}\}_{l=1}^{K}$.
  Update cluster centers: $u_l^{(t+1)} \leftarrow \frac{1}{|\mathcal{X}_l^{(t+1)}|} \sum_{x \in \mathcal{X}_l^{(t+1)}} x$
  $t \leftarrow t + 1$
until No data point assignments change or maximum # of iterations is reached

VIOLATE-CONSTRAINTS(data point x_i, cluster $\mathcal{X}_l$, must-link set $\mathcal{M}$, cannot-link set $\mathcal{C}$)

VIOLATED $\leftarrow$ FALSE
for all $(x, y) \in \mathcal{M}$ **do**
  if $y \notin \mathcal{X}_l$ and y is reassigned in the current iteration **then**
    VIOLATED $\leftarrow$ TRUE;
    break;
  end if
end for
for all $(x, y) \in \mathcal{C}$ **do**
  if $y \in \mathcal{X}_l$ and y is reassigned in the current iteration **then**
    VIOLATED $\leftarrow$ TRUE;
    break;
  end if
end for
return VIOLATED

Algorithm 6 Seeded-KMeans and Constrained-KMeans [4]

Input: Set of data points $\mathcal{X}$, the seed set $\mathcal{S} = \cup_{l=1}^{K} \mathcal{S}_l$

Output: K clusters $\{\mathcal{X}_l\}_{l=1}^{K}$

Method:

initialize each cluster center μ_l using the seed subset $\mathcal{S}_l$: $\mu_l^0 \leftarrow \frac{1}{|\mathcal{S}_l|} \sum_{x \in \mathcal{S}_l} x$
$t \leftarrow 0$
repeat
  for all $x_i \in \mathcal{X}$ **do**
    if Constrained-KMeans and $x_i \in \mathcal{S}$ **then**
      Assign x_i to l where $x_i \in \mathcal{S}_l$
    else
      Assign x_i to the closest cluster $\mathcal{X}_l^{(t+1)}$ based on $\{\mu_l^t\}$ and get $\{\mathcal{X}_l^{(t+1)}\}_{l=1}^{K}$
    end if
  end for
  Update cluster centers:
  $u_l^{(t+1)} \leftarrow \frac{1}{|\mathcal{X}_l^{(t+1)}|} \sum_{x \in \mathcal{X}_l^{(t+1)}} x$
  $t \leftarrow t + 1$
until No data point assignments change or maximum # of iterations is reached

constraints. Xing et al. [55] provide a method to learn a generalized Euclidean distance metric based on the labeled pairwise constraints. Since the learned Euclidean distance metric can be used as a component of KMeans, we call it Xing-KMeans. Assuming the dataset $\mathcal{X}$, must-link set $\mathcal{M}$ and cannot-link set $\mathcal{C}$, the distance metric $dst(x, y)$ between data point x and y can be written in the form of

$$dst(x, y) = dst_A(x, y) = \|x - y\|_A = \sqrt{(x - y)^T A (x - y)} \tag{5.1}$$

The metric learning algorithm tries to learn a positive semi-definite A so that the following optimization problem is satisfied:

$$min_A \sum_{(d_i, d_j) \in M} \|x_i - x_j\|_A^2 \tag{5.2}$$

$$s.t. \sum_{(x_i, x_j) \in C} \|x_i - x_j\|_A \geq 1 \tag{5.3}$$

$$A \succeq 0 \tag{5.4}$$

The choice of the constant 1 in the right hand side of Eq. 5.3 is arbitrary and it can be any positive constant c so long as A is replaced by c^2A.

Since document vectors $\{x_i\}_{i=1}^N$ have very high dimensions, it is computationally prohibitive to estimate the full matrix A. Therefore, we only consider the case when the matrix A is diagonal. When A is a diagonal matrix, it can be represented as $A = diag(A_{11}, A_{22}, \ldots, A_{nn})$, An efficient algorithm for estimating A can be derived with the Newton-Raphson method by defining

$$\begin{aligned} g(A) &= g(A_{11}, A_{22}, \ldots, A_{nn}) \\ &= \sum_{x_i, x_j \in M} \|x_i, x_j\|_A^2 \\ &- \log(\sum_{(x_i, x_j) \in C} \|x_i - x_j\|_A) \end{aligned} \tag{5.5}$$

It can be shown that minimizing g (subject to $A \succeq 0$) is equivalent, up to a multiplication of A by a positive constant to solve the optimization problem defined by Equations (5.2–5.4) [55].

5.3 Methodology

In this section, we describe the details of the document oracle and feature oracle and present the model for labeling documents and features together. We propose a framework to incorporate feature supervision through feature reweighting into various traditional semi-supervised clustering algorithms introduced in Section 5.2. In the tradition semi-supervised clustering, only labeled documents from document supervision are used to either initialize the clustering algorithms or guide the clustering process. In our framework, we introduce one more supervision dimension, namely, labeled features from features supervision into the traditional semi-supervised clustering algorithms.

5.3.1 Oracles

Designing the document oracle is straightforward as all the documents in our datasets have class labels. Therefore, the underlying document class labels can act as a document oracle [2, 4, 6, 12, 31, 51]. However, this is not the case for feature labeling.

Ideally, we should have a gold standard feature set. In order to simulate how a human responds to queries for feature labels, we construct a feature oracle similarly to previous approaches [2, 17]. The χ^2 value of words with respect to the known labels in the document collection is computed and all the words are ranked by their χ^2 values. Then, the top f words are taken as the feature oracle and will be labeled as useful for clustering when the user is queried about them. We define f as the capacity of the feature oracle. The larger f is, the more features this oracle can label. However, as more noisy features may be included when f is large, we should select a proper f for our experiments. We investigate how the value of f affects clustering performance in our experiments. Since a human users can make mistakes, we can also construct a noisy feature oracle. The bottom half of features in the list of features sorted by the χ^2 values with respect to the known labels are considered as noisy features. A noisy feature oracle can be constructed by replacing a certain number of features in the top f features by the same number of features in the bottom half of the list. The construction of a feature oracle is presented in Algorithm 7. Note that our feature oracle is different from those previous feature oracles [2, 17] in two aspects: (1) Our feature oracle only indicates whether a feature is useful for clustering instead of giving the feature a class/cluster label; (2) Our feature oracle can be noisy by introducing $p_n f$ noisy (mislabelled) features with a given probability $p_n > 0$. In this chapter, we use feature supervision *useful-content* as described in Table 1.2. The algorithms and the corresponding supervision methods are summarized in Section 1.4, Table 1.1 and Table 1.2.

5.3.2 Model for Document Supervision

Our purpose in this chapter is to demonstrate that enhancing traditional semi-supervised document clustering with feature supervision can improve the clustering performance. Therefore, with respect to document supervision in our framework, we adopt the same supervision methods using the document oracle as in the traditional semi-supervised clustering algorithms. In Seeded-KMeans based framework, the seeds are randomly sampled according to the size of the seed set for each cluster. In COP-KMeans based framework, the must-link and cannot-link constraints are derived from the sampled seeds for Seeded-KMeans, i.e. the seeds for the same clusters form the

Algorithm 7 Construction of a Feature Oracle

Data Input: Set of unordered features $\mathcal{F}$, Training set $\mathcal{CL}$ – documents and their class labels in the dataset

Parameter Input: Noise level p_n, the percentage of noise features the feature oracle will mislabel as "accept", Feature Oracle Capacity f – the number of features the oracle labels as "accept", $f \ll |\mathcal{F}|$

Output: List of ordered features $\mathcal{L}$ – the list of features the feature oracle labels as "accept"

Method:

1: Compute χ^2 values of all features in $\mathcal{F}$ based on $\mathcal{CL}$
2: Sort all features in $\mathcal{F}$ according to the computed χ^2 values and obtain ordered list $\mathcal{T}$ of the same size as $\mathcal{F}$
3: **for** $i = 1$ to f **do**
4: Flip a coin with the probability p_n getting the tail and obtain the outcome O
5: **if** O is tail **then**
6: Randomly pick a feature from the bottom half of $\mathcal{T}$, which is considered to be a noisy feature
7: Swap i^{th} feature with the picked noisy feature in $\mathcal{T}$
8: **end if**
9: **end for**
10: Generate $\mathcal{L}$ by taking the top f features of $\mathcal{T}$

"must-link" constraints while the seeds for the different clusters form the "cannot-link" constraints. In distance metric learning based framework, we randomly sample the designated number of constraints.

5.3.3 Model for Feature Supervision

We assume that the document (or document constraint) labeling and feature labeling happen simultaneously, i.e., the user can label words as useful for clustering while he is labeling a document, e.g. through highlighting keywords for a document. With this labeling model, a feature is labeled once the user recognizes it as useful for clustering while reading a document. The advantage of this labeling model is saving user effort,

since the user does not need to label the features separately. The disadvantage is that the user does not need to read the whole document content in order to establish a document constraint so that some useful features might be ignored. In our labeling model, we first assume the user will read the whole document content to label a document constraint. Then, we consider the user only read the first fraction $p\%$ of content to label a document or document pair. Furthermore, we can combine the interactive feature selection (Algorithm 3) with this model in the future. By using the two labeling models together, the user can label a feature either while labeling a document or from a list of features presented to him for labeling.

For example, a document d can be considered as a list of words in the order in which the words occur in the document, i.e., $< w_1, w_2, \ldots, w_{|d|} >$, where $|d|$ is the length of the document in terms of the number of words. Note that w_i might be the same as w_j where i is not equal to j, $1 \leq i \leq |d|$ and $1 \leq j \leq |d|$. To label a document, we assume that the user needs to read at least a fraction of the document content p_c, i.e., $< w_s, w_2, \ldots, w_e >$, where $1 \leq s \leq |d|$, $s \leq e \leq |d|$ and $e - s + 1 = \lceil p_c \cdot |d| \rceil$. When $s = 1$, the user reads a document from the beginning. While reading a document, the user is assumed to be able to label words he encounters. The labeled words are included in the labeled feature set $\mathcal{W}^{\mathcal{L}}$. The fraction of document content could be displayed as a text cloud and the user could label words by highlighting them on the text clouds. The user labels a feature if it is a good description of the topic of a cluster and discriminates the cluster from others. Note that the user does not need to associate a feature with a specific cluster.

Definition 5-1. Assuming Labeled Feature Set $\mathcal{W}^{\mathcal{L}} = \{w | M(w) = \textit{labeled}\}$, where M is the function to produce the label of a feature:

$$M(w) = \begin{cases} \textit{labeled} & \text{if } w \text{ is confirmed as useful for clustering} \\ \textit{unlabeled} & \text{otherwise, i.e., } w \text{ is not presented or not confirmed as useful} \end{cases} \tag{5.6}$$

□

5.3.4 Feature Reweighting

Since feature reweighting employed by [22] is simple and effective, we use it on the labeled features for semi-supervised clustering. Although different semi-supervised

clustering algorithms may have their own method of integrating the feature reweighting, we only have one underlying algorithm KMeans in this chapter.

Feature reweighting for KMeans is performed as follows: the *TFIDF* values of labeled features in $\mathcal{W}^{\mathcal{L}}$ are multiplied by a given weight g (> 1):

$$R_w^{d_i}(tfidf) = \begin{cases} O_w^{d_i}(tfidf) \times g & \text{if } w \in \mathcal{W}^{\mathcal{L}} \\ O_w^{d_i}(tfidf) & \text{otherwise} \end{cases} \tag{5.7}$$

where $O_w^{d_i}(tfidf)$ and $R_w^{d_i}(tfidf)$ are the original and reweighted $tfidf$ values of feature w in document d_i respectively. After being reweighted, the vector of *TFIDF* values is normalized. Since Xing-KMeans learns the feature weights based on the pairwise constraints, we use another heuristic to incorporate the labeled features. We perform Euclidean distance metric learning and obtain the feature weights. The most useful features based on the labeled documents constraints are assigned the highest weight by the metric learning algorithm. Since labeled features are regarded as useful for clustering by the user, it is reasonable to assign the highest weight to all labeled features $\in \mathcal{W}^{\mathcal{L}}$.

5.3.5 Semi-supervised Clustering with Feature Supervision

The procedure of semi-supervised clustering with feature supervision is presented in Algorithm 8. Since traditional semi-supervised clustering methods employ user supervision in the form of pairwise constraints or cluster seeds, adding feature supervision to semi-supervised clustering therefore amounts to dual supervision for clustering, i.e., both document supervision and feature supervision [2]. Dual supervision takes place together and before the clustering algorithms begin. The clustering algorithms will use both labeled documents and features to guide the clustering process and produce clusters better matching user expectation.

5.4 Experimental Results

In this section, we demonstrate the effectiveness of our proposed methods on several real-word datasets. Specifically, we study the performance of different weight values for feature reweighting, the size of the feature oracle vocabulary, the fraction of a document's content the user reads, and the noise level of the user (feature oracle). We

Algorithm 8 Semi-supervised Clustering with Feature Supervision

Input: Set of data points $\mathcal{X}$

Output: K clusters $\{\mathcal{X}_l\}_{l=1}^{K}$

Method:

1: Perform *dual supervision*, i.e., *document supervision* and *feature supervision*
2: Obtain the labeled feature set $\mathcal{W}^{\mathcal{L}}$ and the document seed set $\mathcal{S}$ or must-link set $\mathcal{M}$ and cannot-link set $\mathcal{C}$
3: **if** Xing-KMeans **then**
4: Learn diagonal matrix $\mathcal{A}$ and set weights of labeled features to the maximum value in $\mathcal{A}$
5: Perform basic KMeans Clustering using the learned weights
6: **else**
7: Perform feature reweighting based on labeled feature set $\mathcal{W}^{\mathcal{L}}$.
8: Cluster the documents using semi-supervised clustering algorithm.
9: **end if**

enhance several semi-supervised clustering algorithms with feature supervision and compare algorithms with and without feature supervision. In our experiment, each algorithm was run 10 times with different initializations and the average performance is reported.

5.4.1 Datasets and Evaluation Measures

In this work, we use the six datasets and NMI described in Chapter 3 to compare the proposed methods with the baseline algorithms. The six datasets we use are: (1) *news-similar-3*, (2) *news-multi-7*, (3) *news-multi-10*, (4) *webkb-sfcp-4*, (5) *sector-multi-10*, and (6) *reuters-multi-10*.

5.4.2 Clustering Algorithms

In this chapter, we have several variants of KMeans with document supervision and/or feature supervision for our experiments. The algorithms and the corresponding supervision methods are summarized in Section 1.4, Table 1.1 and Table 1.2. The algorithms we use in this chapter are as follows:

- Random-KMeans, which is the unsupervised KMeans with random initialization of the centroids.
- Fes-KMeans, which performs feature supervision during labeling documents (document constraints) but does not use the labeled documents (or document constraints) for clustering.
- COP-KMeans, which enforces labeled constraints during clustering process.
- COPFes-KMeans, which enforces labeled constraints during clustering process and uses feature supervision during labeling constraints.
- Seeded-KMeans, which uses labeled seeds to initialize the centroids for KMeans.
- SeededFes-KMeans, which uses labeled seeds to initialize the centroids for KMeans and performs feature supervision during labeling seeds.
- Constrained-KMeans, which uses labeled seeds to initialize the centroids for KMeans and constrain the clustering process.
- ConstrainedFes-KMeans, which uses labeled seeds to initialize the centroids for KMeans and constrain the clustering process. At the same time, feature supervision is performed during labeling document seeds.
- Xing-KMeans, which learns distance metric based on the labeled document constraints.
- Xing-Fes-KMeans, which learns distance metric and uses feature supervision.

5.4.3 Analysis of Results

Other than when explicitly stated, we assume the whole content is read to label a document and a noise-free feature oracle is employed to label the words in the documents. In addition, we set the number of seeds for each cluster to 10, feature capacity per cluster f to 30.

In this chapter, we mainly use dataset *sector-multi-10-100* to illustrate our points. The results for all other datasets have similar patterns as presented here and may be found in Appendix C.

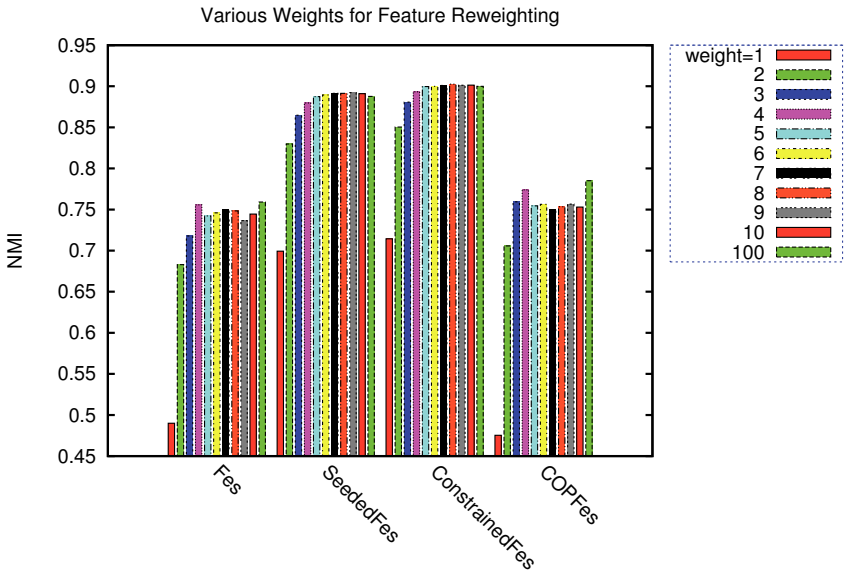
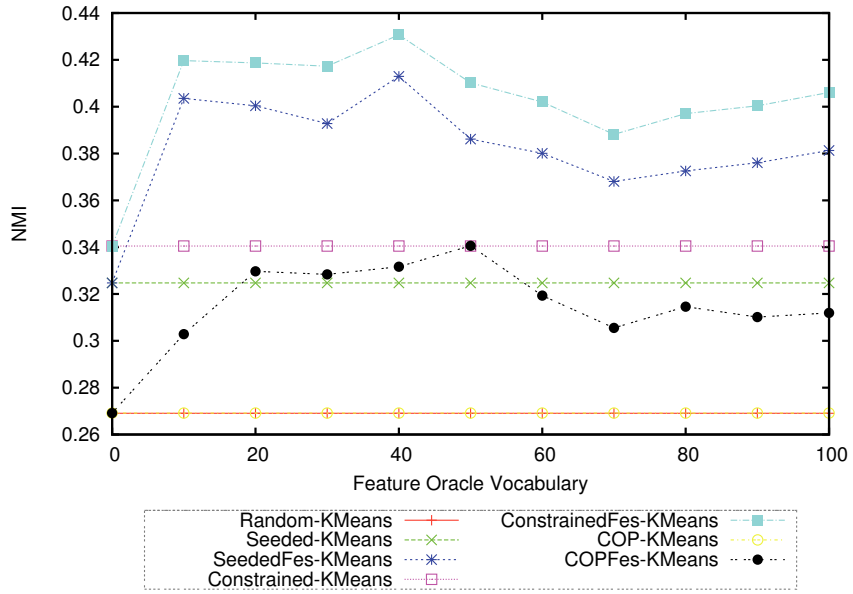


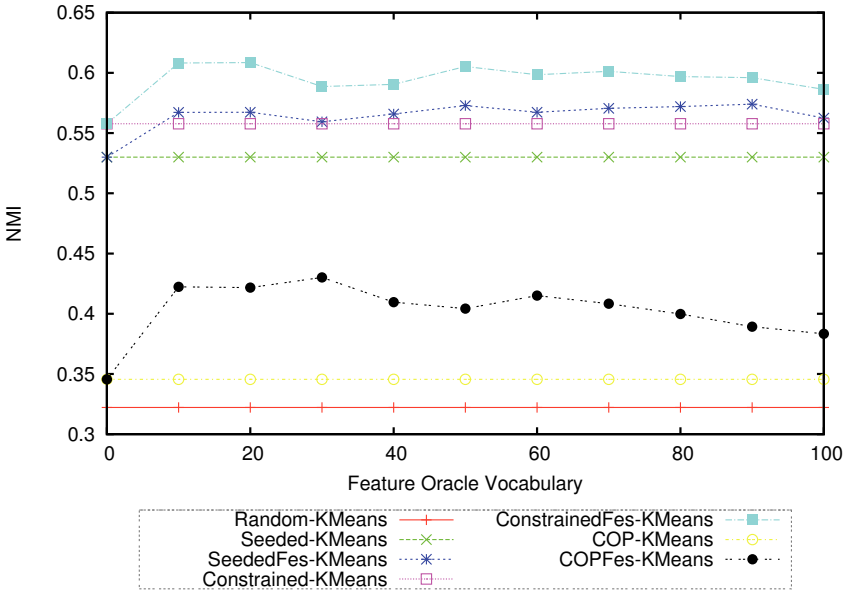
Figure 5.2: Feature reweighting with different weights, on dataset *sector-multi-10*. All weights greater than 1 improve the clustering performance over the corresponding baseline algorithms, i.e., $g = 1$. The baseline algorithms are Random-KMeans, Seeded-KMeans, Constrained-KMeans, and COP-KMeans respectively.

Feature Reweighting g

Different weight values, g (as discussed in Section 5.3.4), might lead to different clustering results. We conducted experiments with different values of g to investigate the robustness of our algorithms. Results show that different datasets and algorithms achieved their best performance with different values of g (Fig. 5.2). However, all weights used (greater than 1) improve the clustering performance over their corresponding baseline algorithms ($g = 1$), namely, Random-KMeans, Seeded-KMeans, Constrained-KMeans, and COP-KMeans. In other words, labeled features are treated the same as other unlabeled features for clustering when g is 1. In this chapter, we select $g = 2$ to report the results on the following experiments. Weight 2 is selected since it is seldom the weight to achieve the best performance for various algorithms on all datasets. Namely, we give the benefit to the baseline algorithms in which g is 1.



(a) *webkb-sfcp-4*



(b) *sector-multi-10*

Figure 5.3: Performance as a function of the size of feature vocabulary, i.e., feature oracle capacity. Semi-supervised clustering with feature supervision shows significantly improved performance over the method without feature supervision. The performance of the clustering algorithms stays relatively stable after the feature oracle vocabulary per cluster reaches a small size 10 to 30.

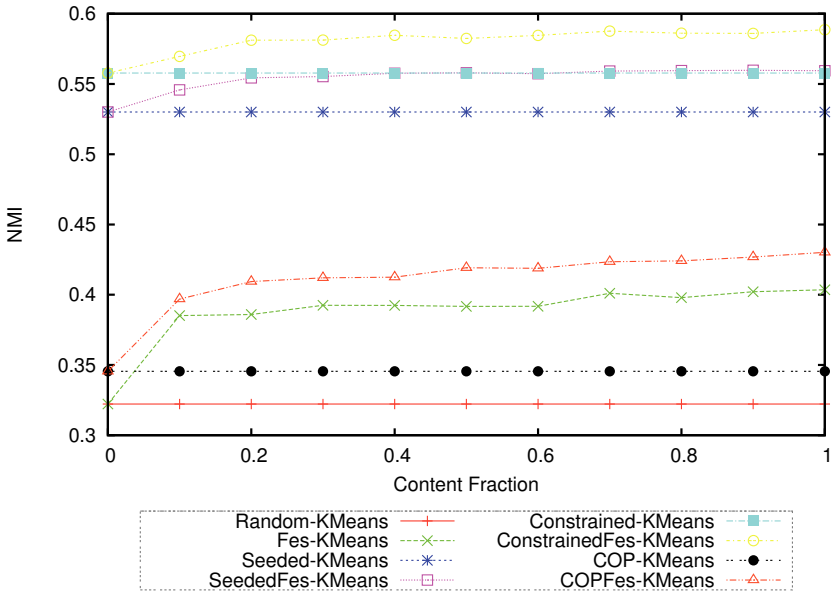


Figure 5.4: Enhanced with feature supervision with varying content being read, on dataset *sector-multi-10-100*. Regardless of the fraction of the content read (at least 10% in our experiments), the performance of semi-supervised clustering with feature supervision is much better over the method with only labeled constraints.

Feature Oracle Capacity f

Feature Oracle Capacity f is the number of features the feature oracle can recognize and label as "accept", namely, the size of the feature oracle vocabulary. If we assume the size of feature oracle vocabulary for each cluster is s, then $f = s \times K$, where is K is the number of clusters. Since we do not know the best value of s for clustering, we conducted experiments with different values of s. We say the f features that the user labels as "accepted" belong to the feature oracle vocabulary. The general hypothesis is that neither too large nor small values of s can produce good clusters.

Assuming the whole content of the labeled documents is read and a noise-free feature oracle, semi-supervised clustering with feature supervision shows significantly[1] improved performance over the method without feature supervision (Fig. 5.3). With feature supervision, Constrained-KMeans and Seeded-KMeans still performs much better than COP-KMeans. It is noticeable that the performance of the clustering algorithms stays relatively stable as a function of f after the feature oracle vocabulary

[1]Two-tailed paired t-test with $p = 0.05$. Also applies to other significance statements.

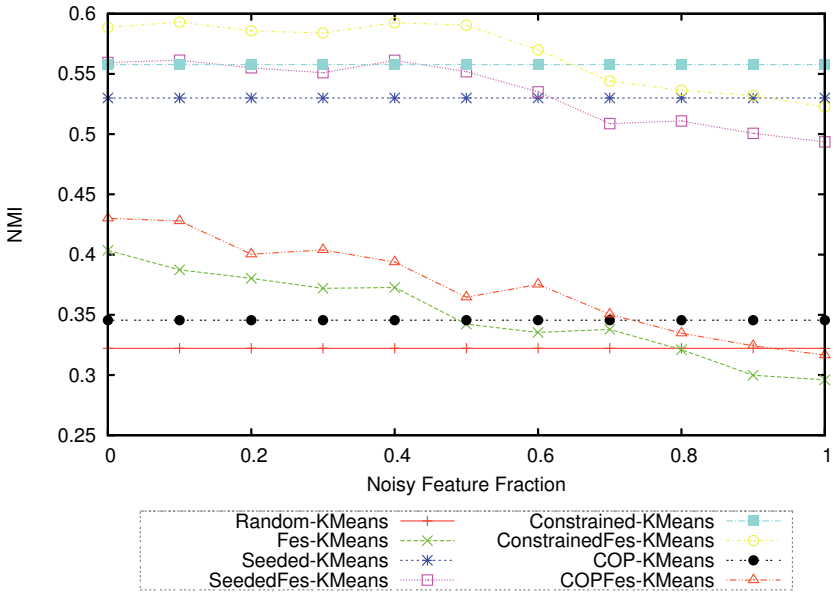


Figure 5.5: Enhanced with feature supervision with varying noise feature fraction, on dataset *sector-multi-10-100*. The clustering performance decreases as more noisy features are introduced by the feature oracle, namely, the more mistakes the feature oracle makes, the worse the performance is. However, even with some incorrect features being labeled as "accepted", the performance of semi-supervised clustering with feature supervision can still improve over the pure document supervision.

per cluster reaches a small size 10 to 30. In practice, it means that the user does not have to know all the discriminative features but only a few of the most discriminative ones. As f grows, clustering performances may decrease, e.g., Fig. 5.3(a). Since the algorithm used to construct the feature oracle is not perfect, it is unavoidable to include some features which are not discriminating for clustering in the feature oracle vocabulary as f grows. We conjecture that clustering performance declines due to the presence of such features introduced by the construction algorithm. The behavior of a noisy feature oracle with explicitly injected poor features is explored later.

Content Fraction p_c

Since the user does not have to read the whole content of a document to label it, we assume that the user reads a fraction p_c of its content start from the beginning. The general hypothesis is that the more content the user reads, the more features the user will label and the better the performance is if the the user can label the features

correctly. However, if the user is not confident with feature labeling, reading more content might not help, but instead it might even harm the clustering performance because more noisy features might be introduced.

Assuming a noise-free feature oracle, the clustering performance with feature supervision is improved with more content of labeled documents being read (Fig. 5.4). At the same time, regardless of the fraction of the content read (as long as it is at least 10% in our experiments), the performance of semi-supervised clustering with feature supervision is much better than the method with only labeled constraints. In fact, the clustering performance only increases moderately with more than 10% of the content of a document being read. Therefore, the user does not need to read the whole content of a document for effective feature supervision, just as the user does not have to read the whole content of the documents to assign a document to a cluster or to establish a document constraint.

Noisy Feature Fraction p_n

To simulate the user making mistakes by accepting poor features for clustering, we construct feature oracles with various fractions of noisy features (See Algorithm 7).

Assuming the whole content of labeled documents is being read, we study the behavior of the noisy feature oracle, which can make mistakes in labeling features. Through the experiments, we find that the clustering performance decreases as more noisy features are introduced by the feature oracle, namely, the more mistakes the feature oracle makes, the worse the performance is (Fig. 5.5). However, even with some incorrect features being labeled as "accepted", the performance of semi-supervised clustering with feature supervision can still improve over the pure document supervision. We observe that our algorithms have high tolerance of mistakes in labeling features (Fig. 5.5). It may be due to the fact the very few labeled features that are highly discriminative dominate the clustering despite the presence of many non-discriminative features.

Number of Seeds or Constraints

We used different numbers of cluster seeds and constraints for the semi-supervised clustering algorithms. The cluster seeds for Seeded-KMeans and Constrained-KMeans

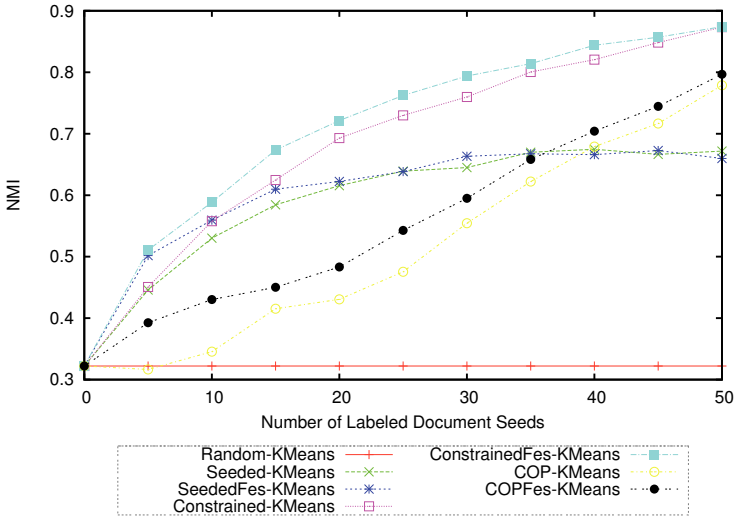


Figure 5.6: Different Number of Document Seeds, on dataset *sector-multi-10* (Constraints for COP-KMeans and COPFes-KMeans are generated from document seeds, see Section 5.4.3 for details). The more documents is labeled, the better clustering performance is. To achieve the same performance without feature supervision, a lot more documents have to be labeled. With more documents labeled, feature supervision becomes less important than when there are only few labeled documents.

are randomly sampled and labeled. Since we compare the COP-KMeans, Seeded-KMeans and Constrained-KMeans, the constraints used for COP-KMeans are constructed from the cluster seeds by establishing "must-link" constraints between the seeds with the same cluster labels and by establishing "cannot-link" constraints between the seeds with different cluster labels. The constraints for Xing-KMeans are randomly sampled.

Feature supervision with a few documents labeled or a few document constraints established can improve the clustering performance significantly compared with the pure document supervision method (Fig. 5.6). To achieve the same performance without feature supervision, a lot more documents have to be labeled. For example, 20 documents per cluster for Constrained-KMeans have to be labeled in order to achieve the same performance as 15 documents per cluster labeled with feature supervision (Fig. 5.6). With more documents labeled, feature supervision becomes less important than when there are only few labeled documents. This implies that feature supervision can help us save user effort from labeling unnecessary documents. Since the user labels

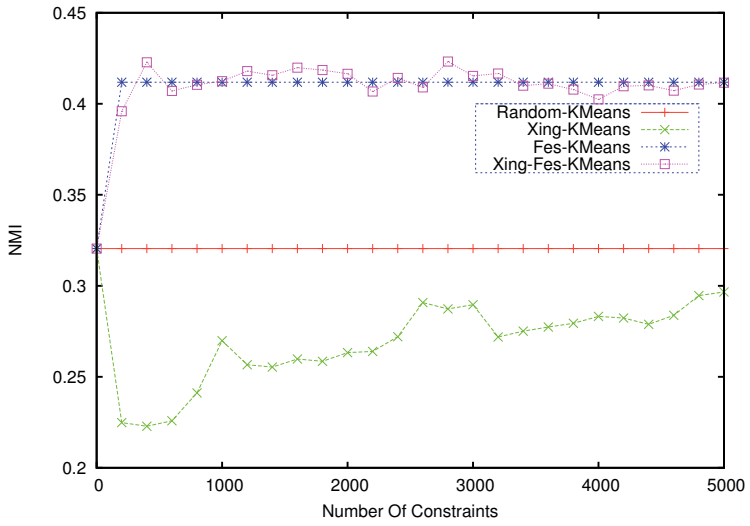


Figure 5.7: Metric learning method and feature supervision method, on dataset *sector-multi-10-100*. Although the distance metric learning method based on labeled document constraints works much worse than Random-KMeans even when quite a large number of constraints is given, Random-KMeans with feature supervision only requires a few constraints and features to be labeled to improve the clustering performance.

features while labeling documents, feature supervision in our proposed methods does not have to involve much extra effort.

Feature Supervision vs. Document Supervision

Besides the semi-supervised clustering with/without feature supervision, we also ran the random$K$Means only with the labeled features, i.e., Fes-KMeans, the algorithm used in Chapter 4 to incorporate labeled features. Random-KMeans with feature supervision performs better than COP-KMeans and comparably with COPFes-KMeans (Fig. 5.5). Although Fes-KMeans performs worse than Seeded-KMeans and Constrained-KMeans, semi-supervised clustering with feature supervision always performs better than without feature supervision on all datasets. The distance metric learning method based on labeled document constraints works much worse than Random-KMeans even when quite a large number of constraints is given (Fig. 5.7). Our explanation is that the high-dimensional and sparse document vectors require too many document constraints to learn a correct distance metric. With only a few

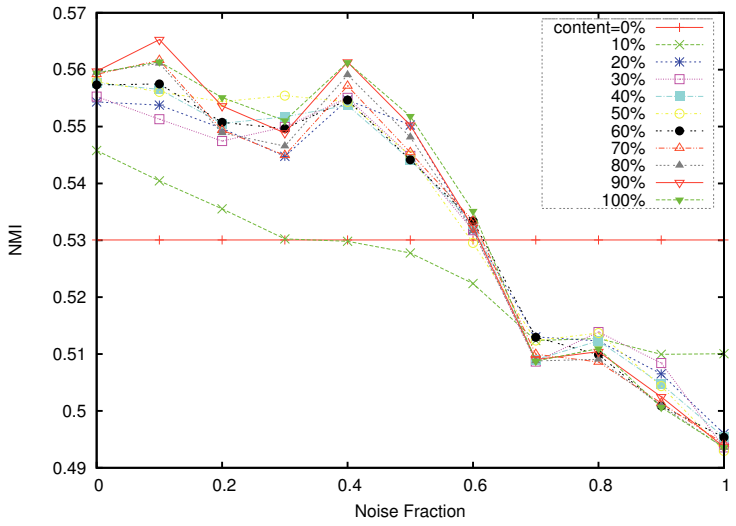


Figure 5.8: SeededFes-KMeans with feature oracle with different noisy feature level for a fraction of content being read. Each curve represents a feature oracle with the corresponding level of noisy features, on dataset *sector-multi-10-100*. A noisy feature oracle still performs very well even when only a small amount of content of a document is read for labeling.

document constraints, some unimportant features are unavoidably over-weighted. However, Random-KMeans with feature supervision only requires a few document constraints and features to be labeled to improve the clustering performance. Note that Xing-Fes-KMeans can still improve the clustering performance further compared to Fes-KMeans. However, the Euclidean distance metric learning algorithm is quite computationally expensive (hours for metric learning versus seconds for feature reweighting for labeled features) even when a diagonal matrix is assumed because of the high-dimensional vector representation of documents.

Noisy Feature Oracle and Content Fraction

Instead of assuming a noise-free feature oracle and that the user reads the whole content of a document to label it, we explore the behavior of the noisy feature oracle while only a fraction of the content is read to label a document. We demonstrate the clustering performance of a fraction of the content being read when feature oracle has different noise levels and the clustering performance of a noisy feature oracle with different noise level when different fractions of the content are read in Fig. 5.8 and

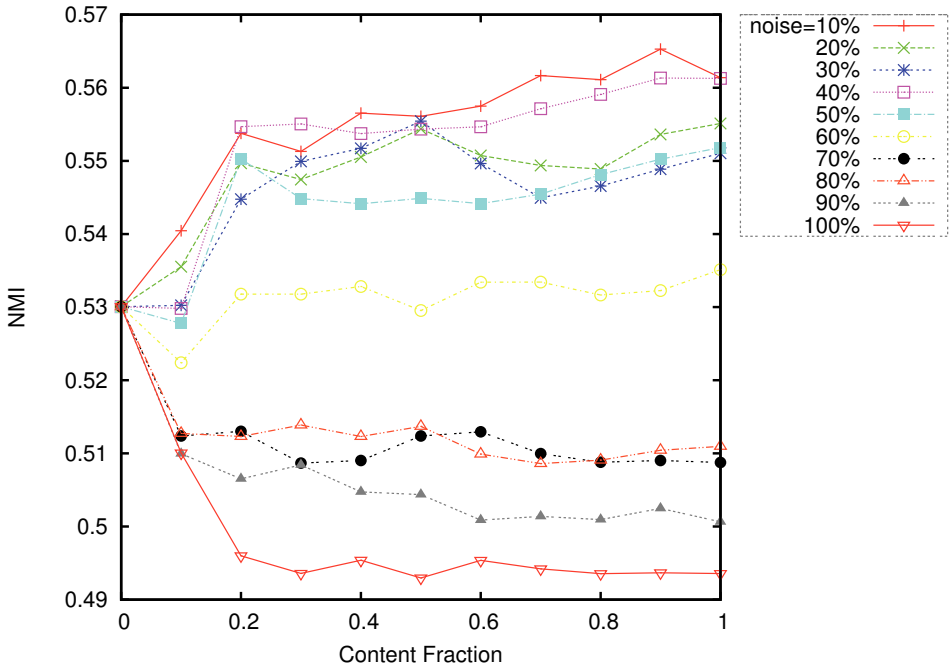


Figure 5.9: SeededFes-KMeans with varying content being read for feature oracle with different noisy feature level. Each curve represents a feature oracle with the corresponding level of content being read, on dataset *sector-multi-10-100*. A noisy feature oracle still performs very well even when only a small amount of content of a document is read for labeling.

Fig. 5.9 respectively. It is suggested that the clustering performance improves as the user reads more content of a labeled document and when the feature oracle is less noisy (Fig. 5.8 and Fig. 5.9). More importantly, those figures demonstrate that a noisy feature oracle still performs very well even when only a small amount of the content of a document is read for labeling. This observation allows human users to make mistakes in feature supervision while reading only a fraction of the content for labeling a document, which supports the practicality of our feature supervision model in which feature supervision during document supervision can improve clustering performance. However, for a very noisy feature oracle such as one with 80% noisy features (Fig. 5.9), the clustering performance decreases when more content of a document is read, since the more content is read, the more noisy features are introduced. In this chapter, the results for Seeded-KMeans are presented. The results for Fes-KMeans, COPFes-KMeans and ConstrainedFes-KMeans have similar patterns in Appendix C.

5.5 Future Work

In this chapter, we discuss how to augment semi-supervised clustering based on document-level supervision with feature-level supervision. We experimented with three different types of traditional semi-supervised clustering algorithms: (1) constraint-based methods, (2) seeding methods, and (3) distance-based methods. To complete this work, we would experiment with one hybrid method [6].

By applying a distance metric learning to text clustering, we found that too many constraints are needed before effective weights are learned. Therefore, we conjecture that it is not suitable to use metric learning based on labeled document constraints when there are not enough constraints for the high-dimensional space vectors. We plan to experiment with more algorithms involving metric learning based on document constraints only [3] or both constraints and intermediate clusters [6]. In this chapter, we assume the feature supervision takes place during document supervision. We can separate those two processes and interleave active document selection [28] and active feature selection [22].

5.6 Summary

In this chapter, we enhance the traditional semi-supervised document clustering with feature supervision, which asks the user to label features by indicating whether they discriminate among clusters. We make the assumption that the user can label features while labeling a document so that the discriminating features are obtained without too much extra work. The labeled features are incorporated into semi-supervised clustering by feature reweighting, which explicitly gives more weight to the features that, according to the user, discriminate among clusters. We explore this enhancement in conjunction with different types of semi-supervised clustering algorithms. Experimental results demonstrate that all types of semi-supervised clustering algorithms enhanced with feature supervision demonstrate significantly improved clustering performance. Specifically, the distance metric learned using feature supervision on top of document constraints performs significantly better than the one learned based only on document constraints. We also find that feature supervision improves clustering performance even when only a small amount of content of the labeled documents is

read and some mistakes are made in labeling features.

Chapter 6

Dual Supervision through Seeding

In the previous chapter, we explored how to incorporate dual supervision into document clustering through feature reweighting, which was used to influence document representations. In this chapter, we will use both document supervision and feature supervision through seeding to influence the clustering algorithms such that clusters better matching user expectation could be produced. We propose two models using labeled features to generate cluster seeds.

6.1 Introduction

As we mentioned previously, traditional document clustering is an unsupervised categorization that partitions a given document collection into clusters so that topically similar documents are placed into the same clusters. However, given the same document collection, different users may want to organize it based on their own points of view instead of a universal one, which is addressed to some extent by incorporating document supervision [4]. In Section 1.4, we defined two types of user supervision, i.e., document supervision and feature supervision for document clustering. In this chapter, we explore document supervision and feature supervision in the forms explained next. *Document Supervision* involves labeling documents, i.e., assigning a document to a cluster. *Feature Supervision* involves labeling features, i.e., associating a feature with a document if that feature describes the topic of that document.

Most prior semi-supervised clustering algorithms use user supervision in the form of document supervision such as labeled instances [4] or instance pairwise constraints [53] for general clustering problems. However, user supervision can also be provided in alternative forms such as labeling features (words) for document clustering in addition to labeling instances (documents). Since this research focuses on document clustering, we may use *instance* and *document*, *feature* and *word* interchangeably. Labeling

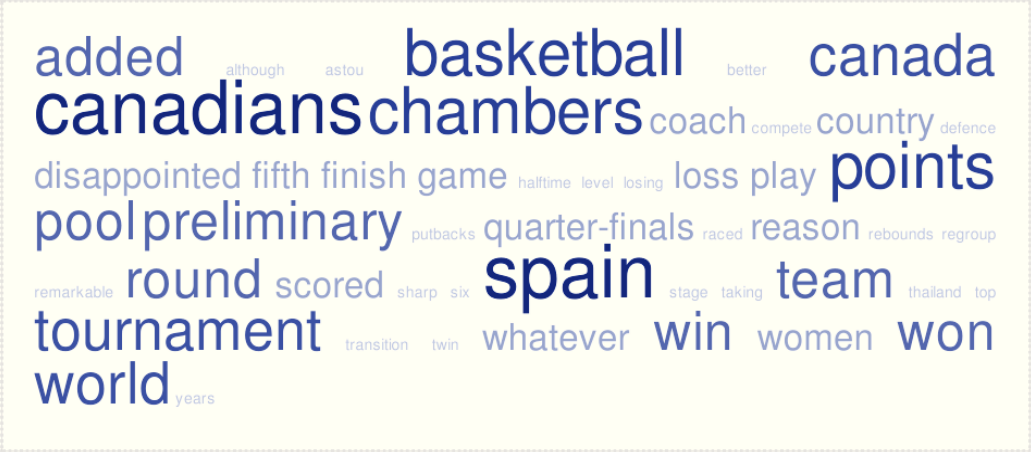


Figure 6.1: Text Cloud of a Document about Canadian Basketball

documents and words can be performed at the same time, with *little additional effort* for labeling words, if an appropriate document visualization is used, such as text clouds [32]. While the user assigns a document to a cluster based on the document's text cloud, the words appearing in the text cloud can also be labeled by being clicked or highlighted.

Example 6-1. Consider a collection of news articles about international sports. While the user labels the document displayed as text cloud (Fig. 6.1) to a cluster, the words associating the document with the specific cluster can also be labeled by being clicked or highlighted. In one scenario, the document (Fig. 6.1) can be labeled to cluster "Canada", in which the words "Canada", "Canadians" should labeled (associated) with the document. In another scenario, the document would be labeled to cluster "Basketball", in which the words "basketball", "points" should be associated with the document. □

Example 6-2. Assume we have two papers and one talks about programming languages while the other is about software debugging. One human user can assign them into the same cluster "software engineering" while another one would like to put them into two clusters, i.e., "languages" and "debugging". Clearly, the keywords (features) assigned for the two cases will be different too. □

As we mentioned before, users might label different words for clusters to reflect their own organizations of the same document collection. A user can form his understanding of the document collection based on the perception of the words in the

text clouds. By using the text cloud for labeling documents, the user can not only label documents to seed the clusters but also label the words discriminating between clusters. Namely, *dual supervision* [2], the joint use of document supervision and features, can be performed using text clouds.

In this chapter, we assume that the user labels a document by reading its content. At the same time, the user can label a word by indicating (e.g. highlighting) whether it is associated with the document or the specific cluster. The text cloud could be used to visualize the document content and enhance the labeling. We extend two methods incorporating the labeled features from document classification to document clustering, namely, feature-vote-model [17] which uses labeled features to vote for cluster label of an unlabeled document, and feature-generative-model [37] which uses labeled features to infer a multinomial generative model. In (semi-supervised) document classification, labeled documents and features are required for each category. However, knowledge of the relevant categories is incomplete in many domains. Semi-supervised document clustering can group documents into partial clusters with labeled documents and features, as well as extend and modify the existing set of clusters to reflect other topical groupings in document collection [4]. In this chapter, we propose a clustering model built from both the labeled documents and the labeled features through seeding. This model can use both labeled documents and features to guide the clustering process. At the same time, the knowledge from the labeled documents and features will be refined by intermediate clusters in an iterative manner. To this end, we present a unified framework which combines knowledge from labeled documents, labeled features, and unlabeled documents by an iterative clustering process. Finally, we demonstrate the effectiveness of the framework on several real-word data sets.

6.2 Methodology

In this section, we first introduce document supervision and feature supervision in the form of document seeding and feature seeding separately. Then, we present two methods to model feature seeding. At the end, we describe a unified framework to incorporate both document seeding and feature seeding into the KMeans algorithm, namely, *DualSeededKMeans*.

6.2.1 KMeans

KMeans is a unsupervised clustering algorithm based on iterative assignments of data points to clusters and partitions a dataset into K clusters so that the average squared distance between the data points and the closest cluster centers are locally minimized. More details about KMeans is available in Section 4.2.1.

6.2.2 Document Seeding

Given a document set $\mathcal{X}$, KMeans partitions it into K clusters $\{\mathcal{X}_l\}_{l=1}^K$. Then, we define the document seed set $\mathcal{D}^L \subseteq \mathcal{X}$ [4] as the following subset of data points: for each document $x_i \in \mathcal{D}^L$, the user provides the cluster $\mathcal{X}_l$ to which it belongs. We assume that there is at least one document data point x_i for each cluster $\mathcal{X}_l$. Note that there is a K-disjoint partitioning $\{\mathcal{D}_l^L\}_{l=1}^K$ of the seed set $\mathcal{D}^L$ such that all $x_i \in \mathcal{D}_l^L$ belong to $\mathcal{X}_l$ according to the supervision. We define the centers of the document seed set $\{\mathcal{D}_l^L\}_{l=1}^K$ as $\{\mu_l^d\}_{l=1}^K$:

$$\mu_l^d = \frac{\sum_{x_i \in \mathcal{D}_l^L} x_i}{|\mathcal{D}_l^L|} \tag{6.1}$$

where x_i is the vector of TFIDF values of the features selected for clustering by an unsupervised feature selection technique (see Section 3.1.7 for details). Those seed centers can be used to both initialize the clustering algorithms and guide the clustering process.

6.2.3 Feature Seeding

Similar to document seed set $\mathcal{D}^L$, we can define the feature seed set $\mathcal{W}^L$ as the following subset of features: for each $w_i \in \mathcal{W}^L$, the user indirectly associates it with the cluster $\mathcal{X}_l$ through document $x_j \in \mathcal{X}_l$ which w_i occurs in and is labeled from. We assume that each cluster has a topic and at least one feature is associated with it. Note that there may not exist a K-disjoint partitioning $\{\mathcal{W}_l^L\}_{l=1}^K$ of the feature seed set because one feature can be associated with multiple clusters. We define the centers of the feature seed set $\{\mathcal{W}_l^L\}_{l=1}^K$ as $\{\mu_l^w\}_{l=1}^K$, which can be derived from either feature-vote-model (see Section 6.2.5 for details) or feature-generative-model (see Section 6.2.6 for details). Then, those seed centers can be used to both initialize the clustering algorithms and guide the clustering process.

6.2.4 Feature Supervision

Feature Supervision is similar to the model described in Section 5.3.3. However, a labeled feature can be associated with a cluster indirectly through the labeled documents from which it is labeled. After a cluster being created, additional features can be associated with a cluster directly by being assigned into the cluster.

6.2.5 Feature-Vote Model

In this method, we use the labeled features in the feature seed set to vote on cluster labels for the unlabeled documents. A similar approach was introduced for document classification [17, 54]. Each labeled feature w in a document x contributes one vote for each of its cluster labels (could be associated with multiple clusters). Then, we normalize the vote totals to get a probabilistic distribution over the cluster labels for each document, i.e., $\{P_{li}\}$ for document x_i and cluster $\mathcal{X}_l$. Assume document x_i contains n_{il} labeled features for cluster $\mathcal{X}_l$, we define:

$$P_{li} = \frac{n_{il}}{\sum_{k=1}^{K} n_{ik}} \tag{6.2}$$

Then, the center of μ_l^w based on the feature seed set can be derived from these soft labeled documents:name=μ_l^w,description=Cluster center based on labeled feature set

$$\mu_l^w = \sum_{x_i \in \mathcal{X}} P_{li} x_i \tag{6.3}$$

where x_i is the vector of TFIDF values of the features selected for clustering by an unsupervised feature selection technique (see Section 3.1.7 for details).

6.2.6 Feature-Generative Model

This model was introduced for sentiment analysis with binary classification [37] and we extend it for document clustering with multiple clusters. In this method, we generate each cluster center from the feature seed set directly. We choose to represent the cluster center as a multinomial distribution which generates documents for the corresponding cluster. Without losing generality, we derive the cluster center for cluster $\mathcal{X}_l$ and *words* and *features* are used interchangeably. We define the following notations to aid our derivations:

$\mathcal{V}$ – set of words used for clustering, including both labeled and unlabeled words

$\mathcal{P}_{\mathcal{X}_l}$ – set of words labeled for cluster $\mathcal{X}_l$

$\mathcal{N}_{\mathcal{X}_l}$ – set of words labeled for the other clusters, i.e., $\{\mathcal{X}_k\}_{k=1}^{K}$, k is not equal to l

$\mathcal{U}$ – set of unlabeled words used for clustering

m – size of vocabulary, i.e. $|\mathcal{V}|$

$p_{\mathcal{X}_l}$ or p – number of words labeled for cluster $\mathcal{X}_l$, i.e. $|\mathcal{P}_{\mathcal{X}_l}|$

$n_{\mathcal{X}_l}$ or n – number of words labeled for the other clusters, i.e. $|\mathcal{N}_{\mathcal{X}_l}|$

In order to derive the multinomial distribution for cluster center of $\mathcal{X}_l$, we assume the following properties about the relationships between words and clusters.

Property 1: All words in $\mathcal{P}_{\mathcal{X}_l}$ are equally likely to occur in a document from cluster $\mathcal{X}_l$.

$$P(w_i|\mathcal{X}_l) = P(w_j|\mathcal{X}_l), \forall w_i, w_j \in \mathcal{P}_{\mathcal{X}_l} \tag{6.4}$$

We refer to the probability of any word in $\mathcal{P}_{\mathcal{X}_l}$ appearing in a document from cluster $\mathcal{X}_l$ simply as $P(w_p|\mathcal{X}_l)$.

Property 2: All words in $\mathcal{N}_{\mathcal{X}_l}$ are equally likely to occur in a document from cluster $\mathcal{X}_l$.

$$P(w_i|\mathcal{X}_l) = P(w_j|\mathcal{X}_l), \forall w_i, w_j \in \mathcal{N}_{\mathcal{X}_l} \tag{6.5}$$

We refer to the probability of any word in $\mathcal{N}_{\mathcal{X}_l}$ appearing in a document from cluster $\mathcal{X}_l$ simply as $P(w_n|\mathcal{X}_l)$.

Property 3: The unlabeled words are treated equally in each cluster.

$$P(w_i|\mathcal{X}_l) = P(w_j|\mathcal{X}_l), \forall w_i, w_j \in \mathcal{U} \tag{6.6}$$

We refer to the probability of any word in $\mathcal{U}$ appearing in a document from cluster $\mathcal{X}_l$ simply as $P(w_u|\mathcal{X}_l)$.

Property 4: A document from cluster $\mathcal{X}_l$ is more likely to contain a word from $\mathcal{P}_{\mathcal{X}_l}$ than a word from $\mathcal{N}_{\mathcal{X}_l}$

$$P(w_p|\mathcal{X}_l) = r \cdot P(w_n|\mathcal{X}_l) \tag{6.7}$$

where r is referred to as polarity level, which measures how much more likely a word in $\mathcal{P}_{\mathcal{X}_l}$ occurs in a document from cluster $\mathcal{X}_l$ compared with a word in $\mathcal{N}_{\mathcal{X}_l}$. Since a word in $\mathcal{P}_{\mathcal{X}_l}$ is more likely to occur in a document from cluster $\mathcal{X}_l$, we have $0 < 1/r \leq 1$.

Property 5: The multinomial probability distribution learned from labeled features

for each cluster is constrained by summing to one.

$$\sum_{i}^{m} P(w_i|\mathcal{X}_l) = 1 \tag{6.8}$$

We use **property 5** as constraints to derive the appropriate probability distribution based on labeled features. By Eq. 6.8 it follows that

$$pP(w_p|\mathcal{P}_{\mathcal{X}_l}) + nP(w_n|\mathcal{P}_{\mathcal{X}_l}) + (m-p-n)P(w_u|\mathcal{P}_{\mathcal{X}_l}) = 1 \tag{6.9}$$

which gives us the following inequality using Eq. 6.7 and given that $m \geq p+n$,

$$\begin{aligned} & pP(w_p|\mathcal{X}_l) + nP(w_n|\mathcal{X}_l) \leq 1 \\ & \Rightarrow pP(w_p|\mathcal{X}_l) + n\frac{P(w_p|\mathcal{X}_l)}{r} \leq 1 \\ & \Rightarrow P(w_p|\mathcal{X}_l) \leq \frac{1}{p+n/r} \end{aligned}$$

Since $0 < 1/r \leq 1$, it follows that,

$$\frac{1}{p+n} \leq \frac{1}{p+n/r} \leq \frac{1}{p}$$

By assigning a large but not too large probability mass to the known words, $P(w_p|\mathcal{X}_l)$ is set to the minimum of the maximum values, i.e.

$$P(w_p|\mathcal{X}_l) = \frac{1}{p+n} \tag{6.10}$$

Now, it follows from Eq. 6.7,

$$P(w_n|\mathcal{X}_l) = \frac{1}{p+n} \cdot \frac{1}{r} \tag{6.11}$$

Now, solving Eq. 6.9, we can have the probabilities for the unlabeled words:

$$P(w_u|\mathcal{X}_l) = \frac{n(1-1/r)}{(p+n)(m-p-n)} \tag{6.12}$$

Finally, we use Equations 6.10, 6.11 and 6.12 to derive the center μ_l^w of cluster $\mathcal{X}_l$. The cluster center μ_l^w is defined as a vector, whose elements are the probabilities of words in $\mathcal{V}$ given the cluster $\mathcal{X}_l$, namely,

$$\mu_l^w = (P(w_1|\mathcal{X}_l), P(w_2|\mathcal{X}_l), \ldots, P(w_m|\mathcal{X}_l)) \tag{6.13}$$

where $w_i \in \mathcal{V}$ and $m = |\mathcal{V}|$ as previously defined.

In our experiments, we set $r = 100$ based on previous experimental results [37].

6.2.7 Combining Multiple Centers

Opinion pool is a general approach to combine information from multiple sources, such as the centers derived from document seed set and feature seed set in our document clustering problem. Particularly, we use *linear opinion pool* approach to aggregate multiple centers. which was used to combine probability distributions for text classification [37]. In this approach, the aggregated (pooling) center is defined as

$$\mu_l = \sum_{s=1}^{S} \alpha_s \mu_l^s \tag{6.14}$$

where S is the number of sources we have.

In addition, we compute the weights α's of individual sources based on their error in labeling the document seed set. In particular, we use the same weighting scheme as [37]:

$$\alpha_s = \log \frac{1 - err_s}{err_s} \tag{6.15}$$

where err_s is the classification error of the source s when the derived centers based on the information provided by the source s are used to classify the documents in the document seed set. All α_s's are normalized to one.

6.2.8 Dual Semi-supervised KMeans

In *DualSeededKMeans*, both the document seeds and feature seeds are used to initialize the KMeans algorithm through derived cluster centers. To this end, the center of the l^{th} cluster is initialized with the pooling center derived from both μ_l^d and μ_l^w (Eq. 6.14) before the clustering starts. During the clustering, the cluster centers are refined using the information contained in the intermediate clusters. This information is expressed in the form of intermediate cluster centers μ_l^c

$$\mu_l^c = \frac{\sum_{x_i \in \mathcal{X}_l^c} x_i}{|\mathcal{X}_l^c|} \tag{6.16}$$

where $\mathcal{X}_l^c$ is the l^{th} intermediate cluster. Then, we can incorporate μ_l^c to the *DualSeededKMeans* algorithm using the *linear opinion pool* technique (Eq. 6.14). The algorithm is described in details in Alg. 9. Note that *DualSeededKMeans* can be specialized to *DocumentSeededKMeans* when feature seed set is empty and *FeatureSeededKMeans* when document seed set is empty.

Algorithm 9 *DualSeededKMeans*

Input: Set of data points $\mathcal{X}$, the document seed set $\mathcal{D}^L = \cup_{l=1}^{K} \mathcal{D}_l^L$, the feature seed set $\mathcal{W}^L = \cup_{l=1}^{K} \mathcal{W}_l^L$

Output: K clusters $\{\mathcal{X}_l\}_{l=1}^{K}$

Method:

Compute $\{\mu_l^d\}$ from $\{\mathcal{D}_l^L\}$ using Eq. 6.1
Compute $\{\mu_l^w\}$ from $\{\mathcal{W}_l^L\}$ using Eq. 6.3 or Eq. 6.13
initialize: $\mu_l^{(0)} = \alpha_d \mu_l^d + \alpha_w \mu_l^w$, for $l = 1, \ldots, K$;$t \leftarrow 0$
repeat
  for all $x_i \in \mathcal{X}$ **do**
    Assign x_i to the closest cluster $\mathcal{X}_l^{(t+1)}$ based on $\{\mu_l^t\}$ and get $\{\mathcal{X}_l^{(t+1)}\}_{l=1}^{K}$
  end for
  Update intermediate cluster centers:
  $\mu_l^c \leftarrow \frac{1}{|\mathcal{X}_l^{(t+1)}|} \sum_{x \in \mathcal{X}_l^{(t+1)}} x$
  Update cluster centers:
  $u_l^{(t+1)} \leftarrow \alpha_d \mu_l^d + \alpha_w \mu_l^w + \alpha_c \mu_l^c$
  $t \leftarrow t + 1$
until *convergence*

6.2.9 Oracles

Most research involving labeling documents simulates human input by a document oracle that uses the underlying class labels of documents in the dataset [2, 4, 6, 12, 31, 51]. However, in the case of features, we do not have a gold-standard set of feature labels. Ideally, we should have a human expert in the loop labeling the selected features. However, such a manual process is not feasible for repetitive large-scale experiments. Therefore, we construct a feature oracle similar to the method described by [17]. Using the document labels, the oracle computes the χ^2 value of each feature with cluster/class label, and accepts a feature if the χ^2 value is above a threshold β. In this chapter, the β value is the mean of the top f most predictive features, where $f = 100K$, namely, 100 times the number of clusters. If accepted, the feature oracle labels a feature with the cluster in which it occurs the most times and any other clusters in which the feature occurs at least half times of the most occurrences. In this

chapter, we use feature supervision *association-content* as described in Table 1.2. The algorithms and the corresponding supervision methods are summarized in Section 1.4, Table 1.1 and Table 1.2.

6.3 Experimental Results

6.3.1 Datasets and Evaluation Measures

We conducted our experiments on six real-word datasets of different sizes and also consisting of different types of text documents. The six datasets we use are: (1) *news-simliar-3* (*D1*), (2) *news-multi-7* (*D2*), (3) *news-multi-10* (*D3*), (4) *webkb-sfcp-4* (*D4*), (5) *sector-multi-10* (*D5*), and (6) *reuters-multi-10* (*D6*). The evaluation measure employed here is NMI. The datasets and NMI are described in Chapter 3. In this chapter, we present a subset of experimental results to illustrate our points. The complete results on all datasets are presented in Appendix D.

6.3.2 Analysis of Results

First of all, we have two sets of comparisons in our experiments. The first set of comparisons is designed to determine whether the user-provided information can be refined by the intermediate clusters, i.e., whether clustering models incorporating unlabeled documents categorize documents better than classification models which only use labeled information. The clustering and classification algorithms are defined as: (1) *DualSeededKMeans*, or its specialized algorithms when one of the seed sets is empty, i.e., *DocumentSeededKMeans* and *FeatureSeeded-KMeans*. Note that Feature-Seeded$K$Means has two variants, namely, Feature-Vote-Model and Feature-Generative-Model to derive cluster centers. (2) Supervised$K$Means, which performs clustering by assigning documents to nearest cluster centers inferred from either the document seed set or the feature seed set or both. It can be achieved by running the *DualSeededKMeans* or its specialized cases, i.e., *DocumentSeededKMeans* and *FeatureSeededKMeans*, with only one iteration. Therefore, we have *DualSupervisedK-Means*, *DocumentSupervisedKMeans*, and *FeatureSupervisedKMeans*. In our experiment, each algorithm was run 10 times with different initializations and the average performance is reported.

We did thorough pairwise comparisons (Table 6.1) to demonstrate that incorporating unlabeled documents can refine the information provided by the user and produce better clusters. Concretely, we compared the following pairs of algorithms:

- DocumentSeeded$K$Means vs. DocumentSupervised$K$Means
- FeatureSeeded$K$Means vs. FeatureSupervised$K$Means using Feature-Vote-Model and Feature-Generative-Model.
- DualSeeded$K$Means vs. DualSupervised$K$Means using Feature-Vote-Model and Feature-Generative-Model.

We observe that all algorithms with refinement by intermediate clusters improve their clustering performance over peer algorithms of Supervised$K$Means (Table 6.1) except when Feature-Vote-Model with only feature supervision performs on dataset *D3* (*news-multi-10*) and DualSeeded$K$Means and DualSupervised$K$Means using Feature-Vote-Model on *D1* (*news-similar-3*) (indicated by * in Table 6.1), which is only 2 out of 30 comparisons. Therefore, intermediate clusters are helpful in improving clustering performance in addition to labeled information.

The second set of comparisons is designed to see whether dual supervision performs better than any single supervision (Table 6.2). Thus, we compare *DualSeededKMeans* with *DocumentSeededKMeans*, and *FeatureSeededKMeans*. Again, we have two variants when feature seed set is involved. We observe that dual supervision with both document labeling and feature labeling generally improves the clustering performance over any single supervision except with feature-generative-model on *D1* (*news-similar-3*) and *D4* (*webkb-sfcp-4*) indicated by * in Table 6.2, which is only 2 out of 24 comparisons. Note that algorithms with dual supervision performs better than document only supervision on all datasets. Therefore, it appears that it is worth labeling features.

Second, we ran experiments with incomplete seeding, namely, only a fraction of categories are seeded by labeled documents, or labeled features, or both (Fig. 6.2 and Fig. 6.3). It can be seen that the performance decreases with increasing number of unseeded clusters. However, the performance does not decrease substantially, suggesting that *DualSeededKMeans* may be able to extend the seeded clusters and

Supervision	Algorithm		*D1*	*D2*	*D3*	*D4*	*D5*	*D6*
None	Basic KMeans		0.069	0.523	0.468	0.341	0.710	0.350
Document	DocumentSeeded$K$Means		0.276	0.692	0.686	0.397	0.815	0.637
	DocumentSupervised$K$Means		0.266	0.625	0.624	0.319	0.786	0.581
Feature	Vote-Model	FeatureSeeded$K$Means	0.551	0.770	0.820*	0.464	0.795	0.649
		FeaureSupervised$K$Means	0.548	0.766	0.820*	0.428	0.791	0.637
	Gen-Model	FeatureSeeded$K$Means	0.515	0.724	0.791	0.470	0.805	0.692
		FeatureSupervised$K$Means	0.512	0.681	0.747	0.413	0.734	0.660
Dual	Vote-Model	DualSeeded$K$Means	0.482*	0.757	0.783	0.421	0.822	0.687
		DualSupervised$K$Means	0.482*	0.745	0.765	0.372	0.815	0.660
	Gen-Model	DualSeeded$K$Means	0.423	0.732	0.738	0.443	0.824	0.684
		DualSupervised$K$Means	0.421	0.703	0.700	0.391	0.812	0.642

Table 6.1: Supervised$K$Means compared to peer algorithms refined by intermediate clusters using measure NMI. 10 documents are labeled for each cluster and features are labeled by feature oracle from the labeled documents. We did two-tailed paired t-test with p = 0.05 for comparing algorithms. In this table, we compare algorithms by pairs, i.e., DocumentSeedsed$K$Means vs. DocumentSupervised$K$Means, FeatureSeeded$K$Means vs. FeatureSupervised$K$Means using Feature-Vote-Model and Feature-Generative-Model. DualSeeded$K$Means vs. DualSupervised$K$Means using Feature-Vote-Model and Feature-Generative-Model. All algorithms refined by intermediate clusters perform significantly better than peer Supervised$K$Means algorithms except FeatureSeeded$K$Means and FeatureSupervised$K$Means using Feature-Vote-Model on *D3* (*news-multi-10*) and DualSeeded$K$Means and DualSupervised$K$Means using Feature-Vote-Model on *D1* (*news-similar-3*) indicated by *.

Supervision	Algorithm		*D1*	*D2*	*D3*	*D4*	*D5*	*D6*
None	Basic KMeans		0.069	0.523	0.468	0.341	0.710	0.350
Document	DocumentSeeded$K$Means		0.416	0.770	0.780	0.466	0.847	0.767
Vote-Model	Feature	FeatureSeeded$K$Means	0.560	0.771	0.819	0.468	0.796	0.679
	Dual	DualSeeded$K$Means	0.561	0.810	0.837	0.484	0.845	0.786
Gen-Model	Feature	FeatureSeeded$K$Means	0.515*	0.746	0.796	0.504*	0.808	0.736
	Dual	DualSeeded$K$Means	0.507*	0.802	0.814	0.502*	0.852	0.797

Table 6.2: Comparison of algorithms with dual supervision to algorithms with any single supervision using measure NMI. 20 documents are labeled for each cluster and features are labeled by feature oracle from those labeled documents. We did two-tailed paired t-test with p = 0.05 for comparing pairs of algorithms. In this table, we compared DualSeeded$K$Means with DocumentSeeded$K$Means, DualSeeded$K$Means with FeatureSeeded$K$Means using Feature-Vote-Model or Feature-Generative-Model. DualSeeded$K$Means works better than DocumentSeeded$K$Means on all datasets. DualSeeded$K$Means works better than FeatureSeeded$K$Means on all datasets except *D1* (*news-similar*) and *D4* (*webkb-sfcp-4*) with Feature-Generative-Model indicated by *.

generate more clusters to fit the regularities in the dataset. Therefore, not all clusters have to be seeded by labeled information.

Finally, we study the behaviors of the *DualSeededKMeans* with different numbers of document seeds. Note that the more document seeds are labeled, the more feature seeds are labeled because the feature seeds are labeled while a document is being labeled. We have the following observations from Fig. 6.4.

- *DualSeededKMeans* always works better than *DocumentSeededKMeans*. However, the performance of the two algorithms is getting close when more document seeds are provided. It suggests that the feature labeling is more useful when there are few documents labeled, i.e., little user effort. One of the possible explanations is that few labeled documents can not represent the cluster structures very well, which can be enhanced by the labeled features at the beginning of the learning curve. However, when enough documents are labeled, the cluster structures can be represented quite well with only documents so that the dual supervision has similar performance to document supervision only.

- When there are only few documents labeled, *FeatureSeededKMeans* (fewer feature seeds) performs better than *DualSeededKMeans* and *DocumentSeededKMeans*. It suggests that feature supervision is more reliable than document supervision when only little supervision can be provided. However, *DualSeededKMeans* and *DocumentSeededKMeans* improve their performance more quickly than *FeatureSeededKMeans* when more document seeds are labeled. When there are enough document seeds labeled, both *DualSeededKMeans* and *DocumentSeededKMeans* perform better than *FeatureSeededKMeans*. Our explanation is that a few labeled features can represent the cluster structures better than a few documents, which also contain other non-discriminating features. Therefore, it is better to label features than documents if only limited user supervision is available.

- Learning curves of *FeatureSeededKMeans* are steep at the beginning but become flat quickly. Our explanation is that enough feature seeds are labeled after a few document seeds get labeled at the beginning. The number of feature seeds labeled does not change much when more document seeds are labeled later.

6.4 Summary

In this chapter, we incorporate document supervision and feature supervision in the form of document and feature seeding. *DualSeededKMeans* is a novel unified framework to combine document supervision, feature supervision and unlabeled documents in the form of seeding. *DocumentSeededKMeans* and *FeatureSeeded-KMeans* are two special cases of *DualSeededKMeans*. Experimental results demonstrate that unlabeled documents can help to refine the information provided by the user and feature supervision is worth the effort to improve the clustering performance further compared to document supervision only and much more helpful when only few documents can be labeled due to manual cost.

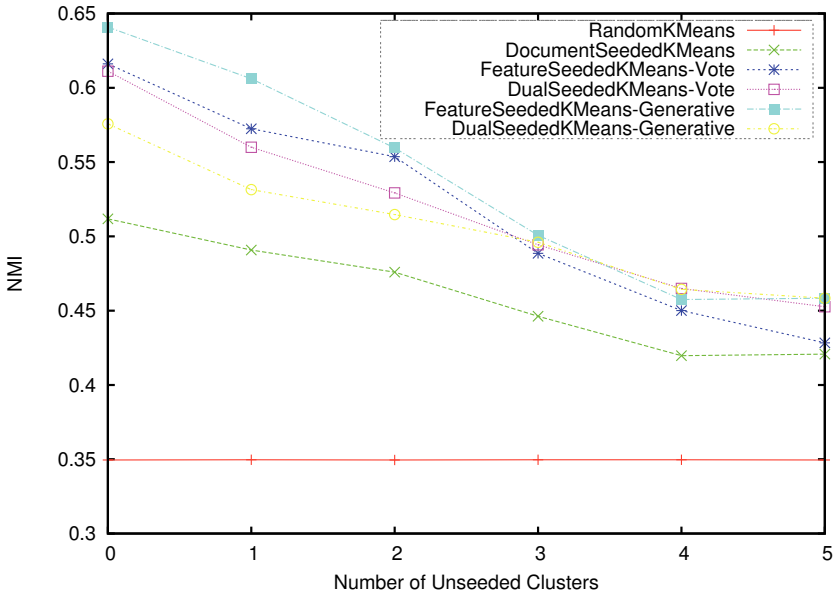


Figure 6.2: Performance as a function of the number of unseeded clusters, on *reuters-multi-10* dataset. 5 documents are labeled for each seeded cluster where FeatureSeeded$K$Means performs better than DocumentSeeded$K$Means and DualSeeded$K$Means.

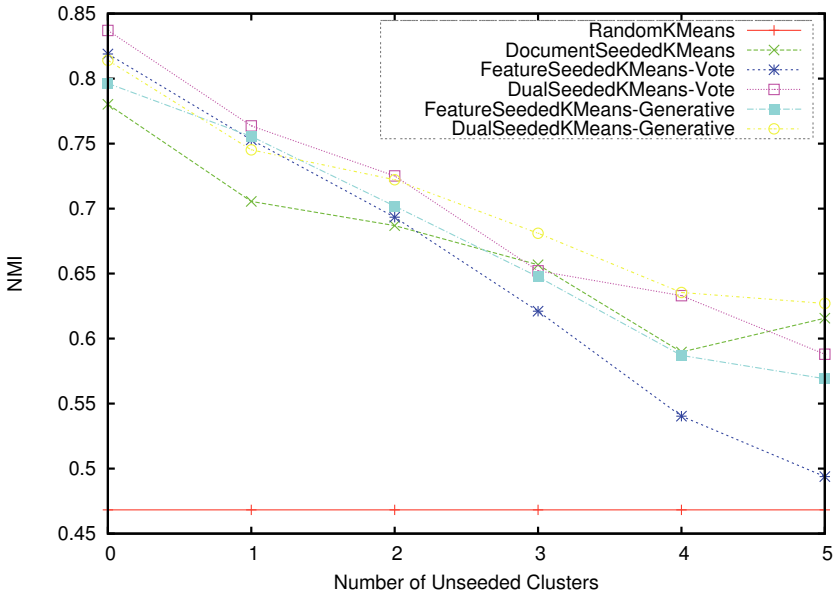
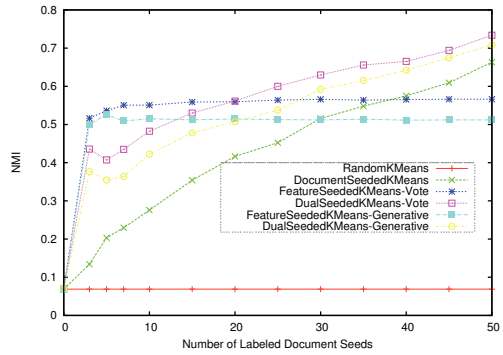
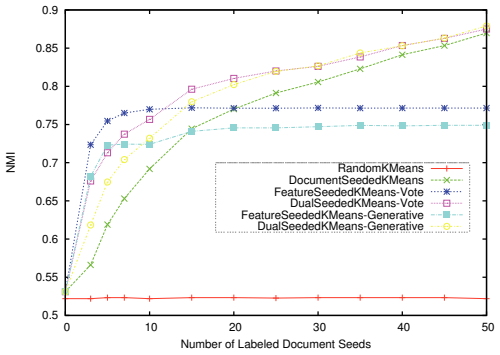


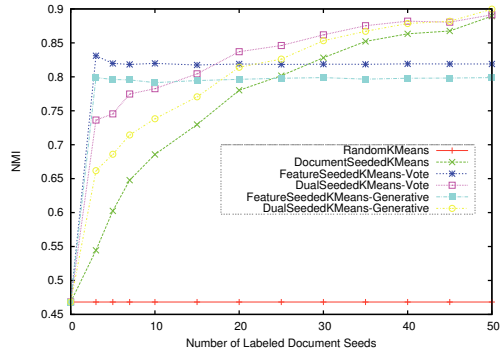
Figure 6.3: Performance as a function of the number of unseeded clusters, on *reuters-multi-10* dataset. 20 documents are labeled for each seeded cluster where DualSeeded$K$Means performs better than DocumentSeeded$K$Means and FeatureSeeded$K$Means.



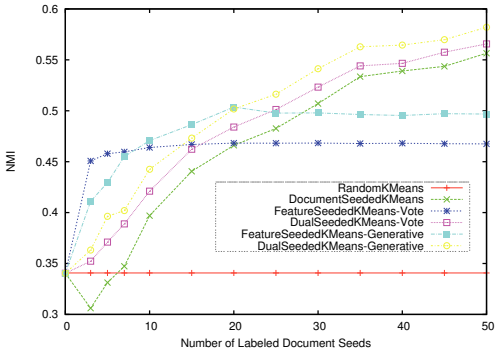
(a) *news-similar-3*



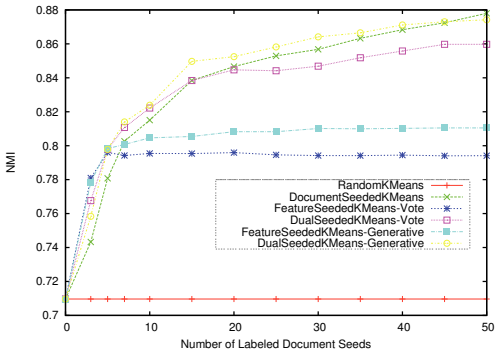
(b) *news-multi-7*



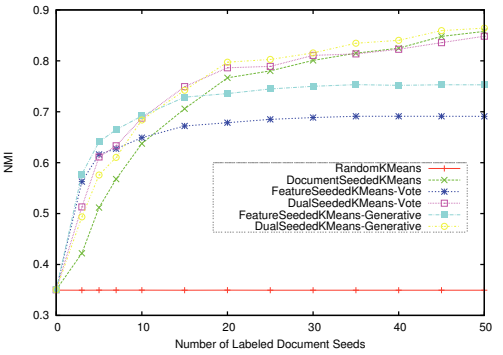
(c) *news-multi-10*



(d) *webkb-sfcp-4*



(e) *sector-multi-10*



(f) *reuters-multi-10*

Figure 6.4: Performance as a function of the number of labeled documents. The more documents are labeled, the more features are labeled and the better is the performance. The usefulness of labeled features is more obvious when there are only a few documents labeled, e.g., < 10. In fact, the feature supervision performs even better than dual supervision at the beginning of the curves, indicating that feature supervision is more reliable when only few documents are labeled.

Chapter 7

Personalized Clustering with Dual Supervision

In Chapters 4, 5, and 6, we proposed algorithms to incorporate feature supervision into clustering techniques through feature selection, feature reweighting, and feature seeding. Similar to previous research, oracles based on underlying class labels of standard datasets are employed to evaluate the newly proposed methods efficiently. However, the potential for semi-supervised techniques to produce personalized clusters cannot be explored in this way since an oracle has the key disadvantage that it always gives the same answer for an assignment of a document or a feature. However, different human users might give different assignments of the same document and/or feature because of different but equally valid points of view. In this chapter, we report on a user study we designed and conducted in which we ask participants (users) to group the same document collection into clusters according to their own understandings. Then we use the resulting clusters as ground truth to evaluate semi-supervised clustering algorithms for user personalization. Through our user study, we observe that different users have their own personalized organizations of the same collection. It also suggests that a user's organization changes over time. We also confirm that semi-supervised algorithms with noisy user input can still produce better organizations matching user expectation (personalization) than traditional unsupervised ones. Finally, we demonstrate that labeling keywords for clusters at the same time as labeling documents can improve clustering performance further compared to labeling only documents with respect to user personalization.

7.1 Introduction

Nowadays, academic researchers maintain a personal library of papers related to their research, projects, and courses, downloaded from digital libraries such as Association for Computing Machinery (ACM) digital library[1]. While those papers might be

[1] `http://dl.acm.org/`

placed into different categories (folders) when they were downloaded, such categories can be too coarse, or inconsistent. For example, a professor may have categories defined by a combination of his graduate students, projects, topics of interest, or by conferences attended. Even worse, papers with different topics might be put into the same folder only for temporary convenience. In fact, even if users categorize the papers appropriately at one time, they might change their mind later on and want to organize the papers in another manner. In addition, researchers might like one organization for their research but another one for preparing their course material. Therefore, the organization of the personal library should be easy to change over time based on user's needs.

Clustering techniques are often employed to group a document collection into different topics. Unsupervised clustering does not require any user effort. However, the users may not be satisfied with the universal output since it does not reflect the individual user's point of view and completely ignores personalization. Semi-supervised clustering incorporates prior information, e.g., user input, into clustering algorithms and normally can produce better quality of clusters. User input is generally provided through user supervision. With respect to document clustering, there are two types of supervision, i.e., document supervision and feature supervision. In document supervision, users provide document-level user input such as labeling a few representative documents for each cluster [4] or identifying relationship between two documents, i.e., "must-link" and "cannot-link" [53]. In feature supervision, users provide feature-level user input such as assigning a few keywords for each cluster [30] or identifying the features (words) which are useful for clustering [22, 24]. The semi-supervised clustering algorithms can also produce personalized clusters if combined with user input from individual users.

The previous semi-supervised clustering algorithms were all experimentally evaluated using oracles. Oracles are based on the underlying class labels of standard datasets. In the case of document supervision, two documents are put into the same cluster or identified as "must-link" by the oracle if they have the same class labels. Otherwise, they are identified as "cannot-link" and must end up in different clusters. With respect to feature supervision, a feature oracle is constructed using feature selection techniques such as the χ^2 or information gain based on the underlying labels

of documents. The constructed feature oracle determines whether a feature is useful for clustering and which cluster the feature should be assigned to. By using oracles, a new semi-supervised clustering algorithm can be evaluated and verified easily and quickly. However, there are two main disadvantages in using oracles to evaluate semi-supervised algorithms. First, oracles always give the correct assignments of documents into clusters or "must-link" and "cannot-link". In real situations, human users can easily make mistakes in assigning documents. Therefore, the semi-supervised algorithms should be tested under noisy supervision, e.g., two documents are placed into the same cluster when they are not meant to. The same problem exists with feature supervision in that a user can pick a useless feature or even assign one cluster's feature to another one especially when there are overlaps between clusters. Although one might claim that noise can be injected into oracle decisions [24], the probability method used to create the feature oracle may not be able to simulate a user's complicated decision process. Second, oracles constructed for one dataset always assign the same label for the same document or the same feature. Assume we have two papers and one talks about programming languages and the other about software debugging. A document oracle based on underlying class labels will always give the same assignments on whether those two papers should be placed into the same cluster. However, one human user can assign them into the same cluster "software engineering" while another one would like to put them into two clusters, i.e., "languages" and "debugging". Clearly, the keywords (features) assigned for the two cases will be different too. Therefore, although semi-supervised algorithms have the potential to produce personalized clusters, they have not been explored for this purpose.

In this chapter, we conduct a user study to determine whether semi-supervised clustering algorithms can produce better quality of clusters when human users are asked to perform document supervision and feature supervision than unsupervised clustering without any supervision. At the same time, we explore the semi-supervised algorithms to produce personalized clusters for individual users when combined with their own user input. We develop an interactive interface to help users to group documents and assign keywords for clusters and documents. The interface helps users to create a new cluster, assign a document to an existing cluster, move a document from one cluster to another, merge two clusters, remove assigned documents and

existing clusters. Thirty-two participants (users) were recruited and asked to label 80 out of the 580 documents (academic papers). The 80 papers are selected by an active recommender described in Section 7.2.3. The papers are generally assigned to three coarse categories selected by their authors, i.e., software, information systems, and computing methodologies. However, the coarse labels are not used at all in this chapter, neither for user supervision nor for the evaluation of the algorithms. The participants do not know the actual number of clusters in the document collection and are asked to group the documents based on their own understanding during exploration. In fact, there are no gold-standard labels for this dataset because each user may create any number of sub-clusters within each coarse category. Therefore, we may obtain different sets of clusters of the same 80 documents from each participant, in terms of the number of clusters, the cluster membership of documents and the keywords assigned to clusters. At the same time, they are asked to select the cluster keywords while they are labeling documents. They can also assign keywords to each cluster directly. In order to demonstrate that semi-supervised clustering works with a small amount of user input, only the first few assigned documents (1 to 6) to each cluster are used as document supervision input (see details in Section 7.3.3). At the same time, only keywords associated with those documents or directly assigned to each cluster are used as feature supervision input. All 580 documents are clustered and the algorithms are evaluated based on the 80 documents manually organized by each participant.

In summary, our contributions in this chapter are: (1) We design and test useful operations and text visualization methods to help users to group documents, which should be included in supervision interface for document management software. We demonstrate that selecting keywords during assigning documents takes little time using the designed interface and operations. (2) We observe that different users group the same document collection differently, i.e., the number of clusters, the cluster memberships of documents, and the assigned keywords. In addition, we observe that a user's organization of a document collection changes over time. Therefore, clustering algorithms which accommodate personalization should be employed to customized clusters for users. (3) We demonstrate that semi-supervised clustering algorithms with a small amount of user input can produce personalized clusters and we verify

that semi-supervised clustering algorithms can still produce better quality of clusters with (noisy) user input than unsupervised clustering. (4) We demonstrate that assigning keywords for clusters can help clustering algorithms to organize documents better matching user's point of view than any single supervision, i.e., labeling only documents or only features.

7.2 Methodology

In this section, we first introduce clustering algorithms we use to demonstrate and verify the usefulness of user input, i.e., the unsupervised clustering algorithm KMeans and semi-supervised clustering algorithm $DualSeeded K$Means. Then, we briefly describe the active learning method we use to recommend documents for user supervision. Finally, we present the interactive user interface we use to collect user input through document supervision and feature supervision.

7.2.1 Unsupervised KMeans

KMeans [10] is a clustering algorithm based on iterative assignments of data points to clusters and partitions a dataset into K clusters so that the average squared distance between the data points and the closest cluster centers are locally minimized. More details about KMeans is available in Section 4.2.1.

7.2.2 Semi-supervised $DualSeeded K$Means

$DualSeeded K$Means [25] is a semi-supervised algorithm which can incorporate user input from both document supervision and feature supervision. It transforms user input from document supervision into document seeding [4, 25] using clusters derived from labeled documents and user input from feature supervision into feature seeding using a Feature-Vote-Model or Feature-Generative-Model [25]. Finally, it combines document seeding and feature seeding using the linear opinion pool [37]. $DualSeeded K$Means is such a general framework that it becomes $DocumentSeeded K$-Means without feature supervision and $FeatureSeeded K$Means without feature supervision. In fact, $DualSeeded K$Means without any supervision is equivalent to

unsupervised KMeans. More details about DualSeeded$K$Means is available in Section 6.2.8. Feature-Generative-Model was used in this chapter.

7.2.3 Active Document Recommendation for User Supervision

Since user supervision is labor-intensive, an active learning scheme is designed to recommend the most potentially informative documents for the user to label, i.e., assigning the documents to a cluster. Our algorithm is an adapted version of the explore-consolidate framework [5] (See Algorithm 10) to the situation when the number of clusters K is not predefined. In the original *Explore* and *Consolidate* framework [5], there are two phases to construct the cluster structure, i.e., *Explore* and *consolidate*. In addition, an oracle is used and the number of clusters K is assumed to be known. In each iteration of the *Explore* phase, a document farthest from the assigned documents is selected using a farthest-first traversal scheme. Then, the document is either assigned to an existing cluster or a new cluster. This step stops after K clusters are created. In each iteration of the *consolidate* phase, a document is randomly selected and assigned to one of the existing K clusters. The purpose of this phase is to consolidate the cluster structure faster because all clusters exist and there is no need to search for the farthest document. However, it is not directly applicable to our work because human users create clusters according to their own understandings of the document collection and different users may create different numbers of clusters (unknown K). Therefore, we do not know when the *Consolidate* phase should start. In the adapted version for the user study, the *Explore* and *Consolidate* phases are interleaved (See Algorithm 11). One iteration of the *Consolidate* phase is performed after every s (4 in this chapter) iterations of the *Explore* phase. However, instead of random selection, a document closest to the smallest cluster is selected in the *Consolidate* phase. The main goal is to have balanced clusters and avoid having too many small clusters.

7.2.4 Interactive User Interface for User Supervision

As we mentioned in Section 7.1, we have two types of user supervision, namely, document supervision and feature supervision. Therefore, we need to provide operations in the user interface to support both types of supervision. We also have to provide

Algorithm 10 *Explore and Consolidate* [5]

Input: Set of document data points $\mathcal{X}$, number of clusters to be created K

Output: The document seed set $\mathcal{D}^L = \cup_{l=1}^{K} \mathcal{D}_l^L$

Phase: *Explore*

initialize: $t \leftarrow 0$ //number of clusters created
$x \leftarrow$ randomly pick a document
$t \leftarrow t + 1$
Create a new a cluster $\mathcal{D}_t^L$ with x
while $t < K$ **do**
    $x \leftarrow$ document farthest from all created clusters $\{\mathcal{D}_l^L\}_{l=1}^{t}$
    if x belongs to an existing created cluster $\mathcal{D}_l^L$, $l <= t$ **then**
        Assign x to cluster $\mathcal{D}_l^L$
    else
        $t \leftarrow t + 1$
        Create a new cluster $\mathcal{D}_t^L$ with x
    end if
end while

Phase: *Consolidate*

while the user still wants to label more documents **do**
    $x \leftarrow$ randomly pick a document not in the created clusters
    Assign x to the cluster $\mathcal{D}_l^L$ ($l <= K$) to which it belongs
end while

Algorithm 11 *Adapted Explore and Consolidate*

Input: Set of document data points $\mathcal{X}$, ratio of *Explore* and *Consolidate* S, i.e., one *Consolidate* step every S *Explore steps*

Output: The document seed set $\mathcal{D}^L$

Method: *Explore* interleaved with *Consolidate*

initialize: $t \leftarrow 0$ //number of clusters created
$s \leftarrow 0$ //number of *Explore* steps
$x \leftarrow$ randomly pick a document
$t \leftarrow t + 1$, $s \leftarrow s + 1$
Create a new a cluster $\mathcal{D}_t^L$ with x
while the users still wants to label more documents **do**
  if $s == S$ **then**
    //**Phase:** *Consolidate*
    $x \leftarrow$ pick a document not in the created clusters and closest to the smallest created cluster, i.e. $\mathcal{D}_{smallest}^L = \arg\min_{l=1,\ldots,t} |\mathcal{D}_l^L|$
    $s \leftarrow 0$
  else
    //**Phase:** *Explore*
    $x \leftarrow$ document farthest from created clusters $\mathcal{D}_t^L$
    $s \leftarrow s + 1$
  end if
  if x belongs to an existing created cluster $\mathcal{D}_l^L$, $l <= t$ **then**
    Assign x to cluster $\mathcal{D}_l^L$
  else
    $t \leftarrow t + 1$
    Create a new cluster $\mathcal{D}_t^L$ with x
  end if
end while

visualizations of clusters and documents to aid user supervision. An interactive user interface[2] is designed for those purposes using dashboard style [19]. The user interface is implemented in Java with G Java 2D Graphics Library[3]. We have four panels in the user supervision interface (Fig 7.1):

(1) **Supervision Panel** {1}[4]: This panel supports document supervision. The sectors of the outer circle denote the clusters and the inner circle represents the document that needs to be labeled (assigned to a cluster) by the user. The (yellow) slices inside a sector denote the documents assigned to the corresponding cluster. The number inside a circle, at the top left corner of a slice or sector is the document or cluster ID. There are always two auxiliary sectors, "New Cluster" and "Trash", which are used to create new clusters and remove clusters or documents respectively. The operations provided by this panel include: (1) Create a new cluster: Drag the inner circle or a slice to the "New Cluster" sector. (2) Move a document: Drag a slice from one sector to another. (3) Merge two clusters: Drag a sector to another. (4) Remove a cluster: Drag a sector to the "Trash" sector. (5) Remove (unlabel or unassign) a document: Drag the inner circle or a slice to the "Trash" sector.

(2) **Document To-Be-Labeled Panel** {2}: This panel displays the information of the document denoted by the inner circle in the **Supervision Panel** and the document ID matches the one in the inner circle. This panel includes two sub-panels to aid users in identifying the topic of the document, i.e., text cloud[5] {5} [32] and the whole content {6} of the document. The user can select a keyword in either sub-panel, i.e., labeling a feature, by double-clicking on the word. After being chosen as a keyword, the word is highlighted in red. If a word is already being highlighted, double-clicking on it removes the highlighting and it is not a keyword any more (un-labeling a feature). The user can also add and delete keywords by entering them in the input field {7} and using the corresponding add/delete buttons {8,9} respectively. All keywords of this document will be

[2]A demo of the interactive interface is available at `http://web.cs.dal.ca/~yeming/demo.htm`
[3]`http://geosoft.no/graphics/`
[4]Corresponding Identification number in Fig. 7.1
[5]Text cloud is computed using OpenCloud: `http://opencloud.mcavallo.org/`

shown in the keyword area of this panel.

(3) **Cluster View Panel** {3}: After the user single-clicks on a sector in the **supervision panel**, the information of the corresponding cluster is displayed in this panel. This panel is similar to the **Document To-Be-Labeled Panel** except that there is no visualization of the whole content simple because a cluster does not have it. The user can assign keywords using the methods introduced previously. Note that keywords assigned to a document become keywords of its cluster automatically while the keywords directly assigned to a cluster are not connected to any document assigned to it. Keywords assigned into a cluster should describe the topic of the cluster and are used by *DualSeededK*Means with Feature-Vote-Model or Feature-Generative-Model in Section 7.2.2.

(4) **Document Labeled Panel** {4}: The layout of this panel is identical to the **Document To-Be-Labeled Panel**. When the user single-clicks on a slice in the **Supervision Panel**, the information about that assigned document is shown in this panel. Besides viewing the topic of the document, the user can also revise the keywords assigned to this document.

7.3 Experimental Results

7.3.1 Datasets

The dataset we use for the user study is a collection of the 580 academic papers in full text from different areas of computer science (See details in Chapter 3). Those papers were manually collected by the author from the ACM Digital Library. Based on the 1998 ACM Computing Classification System, those papers were assigned to one or more of the following areas by their authors: Software including Software Engineering and Programming Languages, Information Systems and Computing Methodologies. Generally speaking, the categories assigned by paper authors are very coarse and cannot reflect the accurate topics of the papers. In addition, it is not uncommon that one paper is related to multiple topics and can be assigned to multiple categories. Therefore, this dataset is well suited for us to determine whether different users have their own point of view of the same document collection. At the same time,

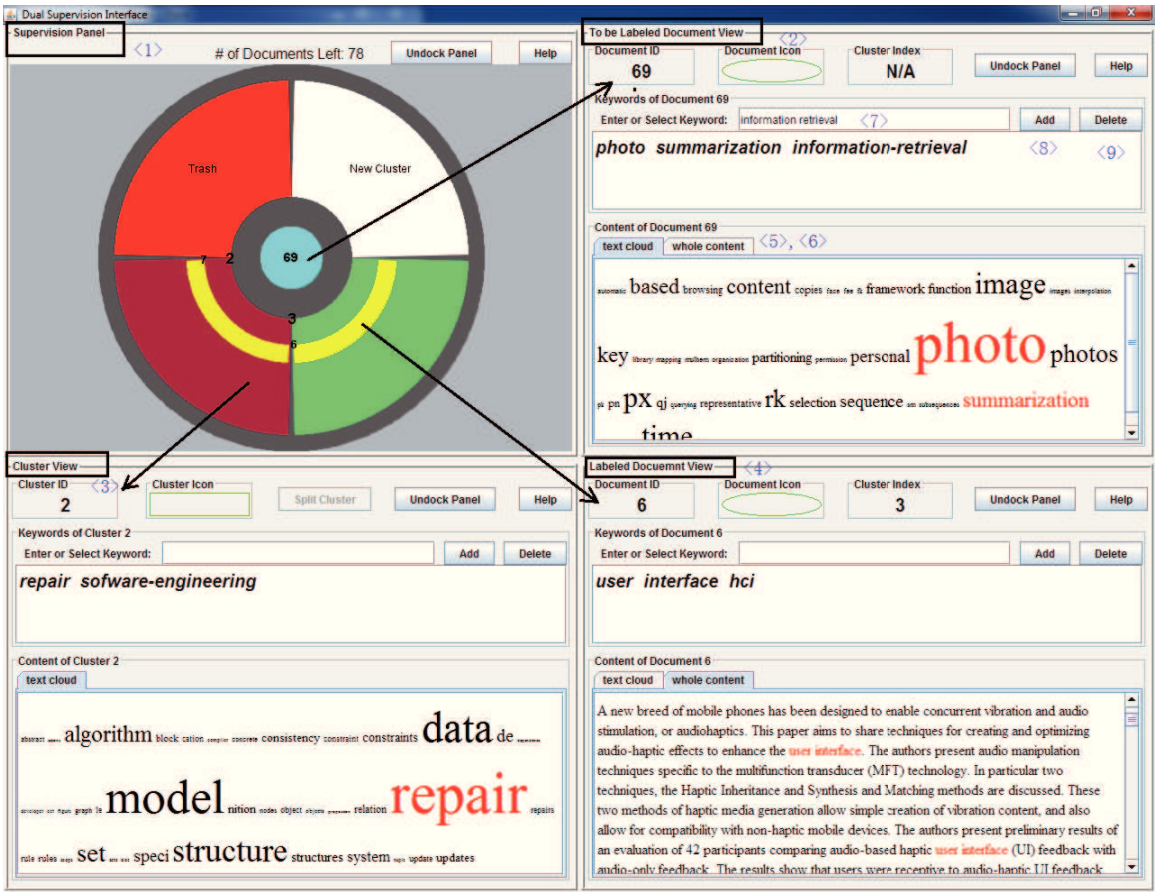


Figure 7.1: Interface for User Supervision. The user interface supports both document and feature supervision with four panels, i.e., top left panel for document supervision, top right panel for the document to be labeled, bottom left panel for a selected cluster, and bottom right Panel for an assigned document. The interface also provides various operations to help users with document supervision and feature supervision in each panel such as creating a new cluster, moving a document, merging two clusters, removing a cluster or a document, labeling and unlabeling a word, etc.

we can demonstrate the usefulness of user supervision for producing personalized organization.

7.3.2 Evaluation Measures

We use Rand Distance based on Rand Index [41] to compare different users' clusterings (groupings) of the same document collection and determine whether different users have their own point of view, thereby motivating the inclusion of user personalization as a requirement for clustering algorithms. Based on Rand Index, we develop measures of cohesiveness and separation to evaluate the clusters produced by clustering algorithms in comparison with users' manual organizations. In addition, we use Jaccard Distance [49] to measure the dissimilarity between the sets of features labeled by different users. The details and definitions of these measures can be in Chapter 3.

7.3.3 Experimental Setup

We recruited thirty-two participants to group 80 of the 580 academic papers in our ACM dataset. Those 80 papers are selected by the active learning method presented in Section 7.2.3 and every participant groups the same 80 papers. The thirty-two participants include 5 female and 27 male graduate students from computer science. At the beginning, the task of the user study is introduced to all participants explaining that there are no given predefined categories and they are asked to group papers based on their own understandings during the exploration of the collection. They are also aware that they need to assign keywords to a document and those keywords will be included in the cluster keywords automatically after the document is assigned to a cluster. They can also assign and remove keywords to and from clusters directly. Then, they are given a demonstration how to group the documents and assign keywords using the software, whose interface is shown in Fig. 7.1 and then they are given 5 minutes to become familiar with the software. Finally, they are asked to use the software to group the 80 documents. Documents are presented to the user one at a time for labeling. The document to be labeled is represented by the inner circle of Fig. 7.1. The order of appearance of the 80 documents depends on the active learning recommendation method. Although all participants group the same 80 documents, each user is presented the documents in a different order.

Name	Definition
Add	Assign a document into a cluster
Move	Move a document from one cluster to another
Delete	Remove a cluster
Merge	Merge two clusters
Label	Add a keyword through double-clicking
Unlabel	Remove a keyword through double-clicking
AddButton	Add a keyword through Add Button
DelButton	Remove a keyword through Delete Button

Table 7.1: Definitions of operations

For all users, we experiment with document supervision consisting of fewer than the full 80 documents a user labels. We only use the first m documents assigned to each cluster, where m ranges from 1 to 6. Documents within each cluster are ordered based on the time they were assigned to the cluster, either when being labeled for the first time or when moved from another cluster. When a cluster B is merged into cluster A, documents in A precede documents in B. At the same time, only keywords associated with those documents selected for document supervision, or directly assigned to each cluster, are used as feature supervision input. Since the order the documents appear for labeling is distinct, the user input from each user for *DualSeededK*Means includes different sets of documents and labeled features. All 580 papers are clustered and the clusterings produced from different algorithms are evaluated based on the 80 user labeled papers using *coh*, *sep*, and F-Measure.

7.3.4 Results

In this section, we present the user feedback and analyze results from our user study. We performed three kinds of analysis on the following aspects: usage of the interface, personalization of the same document collection, and personalized document clustering with dual supervision.

User Satisfaction with the User Interface

Generally speaking, all participants think they know the topics of document collection well and it is easy to identify the topic of a document and to identify the keywords that need to be assigned into a cluster after they manually organize the 80 documents

recommended by the active recommender. They also indicate that the operations to assign and move a document, delete and merge clusters are easy to use. However, only one participant used the operation "split a cluster" because others did not realize its existence. They would like to use the software to organize their personal library of papers if the system came with proper documentation and agree that the system can help them to organize their papers better. A few interesting points we find out are:

- Twenty-nine participants think assigning keywords only takes a little time (less than 10 seconds) while only three of them indicate that it takes some time (more than 10 second but less than 1 minute). No one thinks it takes much time (more than 1 minute).

- All participants except one think that the whole content is more useful than text cloud in identifying the topic of a document. This point can also be verified by Table 7.5, which shows that about 70% keywords are labeled based on the whole content. This is surprising since we expected that text cloud would be more helpful. One of the possible explanations is that we used single words for text cloud and multiple-word phrases could have made the text cloud more useful, especially for scientific documents, which tend to have many multi-word terms.

- All participants review the topic of an existing cluster through keywords instead of reading documents assigned to this cluster. Therefore, it is very important to assign meaningful and correct keywords to a cluster from the beginning.

In addition, many participants suggest that they would like to have more functionality such as searching documents by words. They also suggest that we might add a spell checker for the keywords they enter. More specifically, some participants like to have all assigned documents with a keyword within a cluster highlighted when the keyword of that cluster is selected.

We present a few excerpts from users' feedback as evidence for the above observations:

- "The pie visualization[6] is very easy to use after practicing on it."

[6]The "Supervision Panel"

Operation	MIN	AVG	MED	MAX
Add	80	80.5	80	92
Move	0	4.31	3	22
Delete	0	3.03	2	10
Merge	0	1.84	1	10
Label	5	72.00	54	205
Unlabel	0	4.15	2	18
AddButton	0	12.53	9	80
DelButton	0	9.28	7	36

Table 7.2: Usage of interface operations. It is not uncommon that a user moves a document, deletes an existing cluster and merges two clusters. Many keywords are removed after being assigned (Unlabel and DelButton). Users remove keywords mainly through DelButton because they clean the keywords at the end of the organization. In sum, a user changes his perception of the document collection during exploration.

- "I really liked the drag and drop feature, which has made the system very easy to use."
- " 'Split A Cluster' helped me when I by mistake merged two clusters together."
- "I like the typing keywords feature because it allows me to generalize or be more specific about keywords without being constrained to a predefined list."
- "I found text clouds less useful than I expected."
- "It should be useful to have cluster or document keywords when the mouse hovers it in the supervision panel."
- "When a document or cluster is selected, I would expect this cluster or document was somehow highlighted in the supervision panel. Without it, it is not easy to move a document from one cluster to another."
- "I'd like to have the search (CTRL+F) function and a spell checker."
- ...

User Behaviors

We analyze operations defined in Table 7.1 which users use during grouping documents and assigning keywords so we can identify the most useful ones that should be

NAME	MIN	AVG	MED	MAX
# of Clusters	4	6.34	6	9
Assigned Documents	68	76.47	77	80
Assigned Keywords / Cluster	1	9.09	8	26

Table 7.3: Statistics of # of clusters created, assigned documents and keywords. Different users create different numbers of clusters, assign different numbers of documents, and assign different numbers of keywords. However, it shows that users assign about 10 keywords per clusters, which could be used to help clustering algorithms to produce clusters matching their expectations.

included in future interface design. We also present the analysis of the text visualization methods used in the user interface.

A document is considered as "assigned" after it is placed into an existing or a newly created cluster. Otherwise, it is considered as "unassigned", i.e., the document is placed into the "trash" cluster. The average of assigned documents out of the 80 documents is 76.47 (Table 7.3). Therefore, users know topics of most documents recommended by the active recommender. The most unassigned documents by a user is 12, which is 15% of all documents recommended. However, most users only have less than 4 documents not assigned to any cluster. In fact, some users put a few documents into trash clusters at the beginning. Later on, those documents are retrieved and assigned into an existing cluster. Oracles in previous work could not simulate this behavior, in which not all recommended documents are assigned at the beginning and users can have chance to put some documents on hold and cluster them later. In addition, users are able to assign at least a few keywords for each cluster and generally assign about 9 distinct keywords for each cluster (Table 7.3).

We display the minimum, average, median, and maximum times users use each operation in Table 7.2. The fact that we have 92 (more than 80) add operations indicates that some documents are moved from an existing cluster to create a new cluster. It is also not uncommon that a user move a document from one cluster to another, delete an existing cluster and merge two existing clusters. In addition, users also assign plenty of keywords for clusters. At the same time, many assigned keywords are removed since the users do not consider them as keywords after they learn more about the collection. On one hand, keywords are assigned through double-clicking more often than using add keywords buttons. That is mostly due to the fact users can

Name	Through Document		Directly		Total
Label	68.03	91.13%	3.97	8.87%	72.00
AddButton	9.00		3.53		12.53
Unlabel	3.78	51.90%	0.37	48.10%	4.15
DelButton	3.19		6.09		9.28

Table 7.4: Keywords assigned through documents or directly to clusters. Most keywords are assigned into clusters through documents. Keyword removals equally often happen through documents and clusters (directly). Especially, it is observed that keyword removals through documents take place when a user is labeling a documents through double-clicking while keyword removals through clusters usually happen after the user finishes the manual organization and wants to clean the keywords representing the cluster.

assign the keywords during reading a document. However, some keywords have to be assigned or removed through add or delete keyword buttons because these keywords do not exist or are difficult to find in any document. On the other hand, users remove keywords mainly through the delete button. That is because users normally clean the keywords for a cluster using delete button at the end of the manual organization so that the keywords left can represent the topic of the cluster well. All frequent use of those operations confirm that a user can change his perception of the document collection while exploring the document collection. Therefore, clustering software should enable users to change the existing cluster structures.

Next, we analyze user behaviors on assigning keywords. A user can assign keywords to a document and the keywords associated with the document are assigned to a cluster automatically after the document is assigned into the cluster. A user can also assign keywords to a cluster directly. In addition, we are interested in which visualization method users use most often. We observe that most keywords are assigned through documents while a small percentage is assigned into clusters directly from Table 7.4. Although most participants assigned keywords primarily by double-clicking and rarely by using "AddButton", one participant only used "AddButton" because he said the keywords that he came up with could best reflect his perception of the collection. With regards to keyword removals, users removed keywords equally often through documents and directly from clusters. It is observed that keyword removals through documents mostly take place when users read documents and try to learn its topic while users remove keywords from clusters directly after they finish

Name	Text Cloud		Whole Content		Total
Label	23.03	31.97%	48.97	68.01%	72.00
Unlabel	1.37	33.01%	2.78	66.99%	4.15

Table 7.5: Keywords assigned through text cloud or whole content. It is observed that more than two-thirds of the keywords assigned through double clicking are selected using whole content of document. This fact is consistent with the user feedback that the whole content is more helpful in discerning the topic of the document than the text cloud. However, text cloud is still useful for assigning keywords since it has fewer words than whole content and is easier to find the word to be assigned. As some users indicated in the post-study questionnaires, text cloud is useful to have a general idea about the document but the whole content helps to find the exact topic of the document.

#	4	5	6	7	8	9	Total
Frequency	2	7	11	6	2	4	32
Percentage	6.25	21.88	34.38	18.75	6.25	12.5	100

Table 7.6: Frequency of # of clusters created. Different users create different numbers of clusters based on their own perceptions of the same document collection.

the manual organizations and want to clean the keywords which represent the topics of clusters. We observe that more than two-thirds of the keywords assigned through double clicking are selected using whole content of document from Table 7.5. This fact is consistent with the user feedback that the whole content is more helpful in discerning the topic of the document than the text cloud. However, text cloud is still useful for assigning keywords since it has fewer words than whole content and easier to find the word to be assigned. As some users indicated in the post-study questionnaires, text cloud is useful to have a general idea about the document but the whole content helps to find the exact topic of the document.

Personalization

We compare different groupings of the same 80 papers from all participants in terms of both documents and keywords assigned. We also compare the groupings of the same 80 papers from the same users at different times to see whether the same users have different views of the same collection over time.

First, users create different numbers of clusters based on their own understanding although all numbers are between 4 and 9 (Table 7.3). More specifically, about 80%

Measure	MIN	AVG	MED	MAX
$\mathcal{RD}$	0.1308	0.2483	0.2455	0.3817
$\mathcal{JD}_a$	0.6346	0.8632	0.8677	1.0
$\mathcal{JD}_b$	0.6483	0.9007	0.9082	1.0

Table 7.7: Statistics of user manual organizations. The Rand Distance 0.25 shows that there is substantial disagreement (Rand Distance is 0 for completely agreement) between different users and distinct clusters were created. In addition, the Jaccard Distances in terms of labeled keywords indicate even more disagreement between different users (average distance about 0.90).

of the participants created 5, 6, or 7 clusters while others created 4, 8, 9 clusters (Table 7.6). The frequency of the cluster numbers are close to uniform distribution among 5, 6, 7 (the more frequent cluster numbers, about 74%) and among 4, 8, 9 (the less frequent ones, about 36%) respectively. Therefore, users tend to create different numbers of clusters. Later on, we will observe that different participants have distinct clusters regardless of whether the cluster numbers are the same or not.

We present the minimum, average, median and maximum Rand Distance and Jaccard Distance between organizations of all user pairs of clusterings in Table 7.7. The average Rand Distance between the user pair organizations is about 0.25. If different users create similar partitions of the same document collection, we would expect that the average Rand Distance is close to 0. Therefore, the Rand Distance 0.25 shows that there is substantial disagreement between different users and distinct clusters were created. In addition, the Jaccard Distances in terms of labeled keywords indicate even more disagreement between different users (average distance about 0.90). That is because there is normally a much bigger word vocabulary than the number of documents and many different keywords can be used to identify the same cluster topic, i.e., completely different keyword sets can be used for the same topic. $\mathcal{JD}_b$ shows a higher disagreement than $\mathcal{JD}_a$ because $\mathcal{JD}_b$ considers the cluster label of keywords while $\mathcal{JD}_a$ does not. Therefore, it confirms our conjecture that different users have different point of view of the same document collection.

Finally, we compare manual organizations by the same users but at different times. Generally speaking, the two organizations are still distinct from one another although they are closer when compared to organizations from different users (Table 7.7 and Table 7.8). For example, the average Rand Distance between user pair's manual

Measure	P_1	P_2	P_3	P_4	P_5	AVG
$\mathcal{RD}$	0.2202	0.073	0.1652	0.1323	0.1647	0.1511
$\mathcal{JD}_a$	0.5574	0.4444	0.82	0.8684	0.6667	0.6714
$\mathcal{JD}_b$	0.6072	0.4348	0.8396	0.8269	0.7498	0.6917

Table 7.8: Manual organizations by five same users at different times. Generally speaking, the two organizations are still distinct from one another although they are closer when compared to organizations from different users (Table 7.7). For example, the average Rand Distance between user pair's manual organizations is 0.25, while from the same user is 0.15.

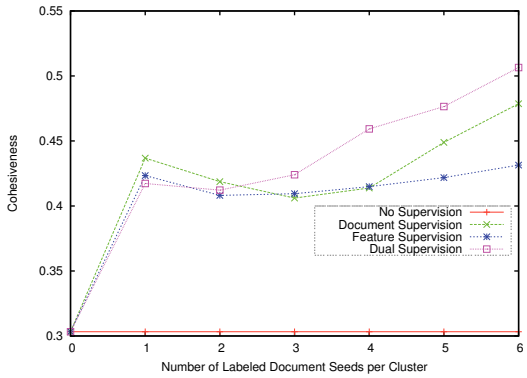
Name	No Supervision	Document	Feature	Dual
$\mathcal{RD}$	0.080	0.2540	0.1711	0.2226

Table 7.9: Rand distances between clusterings produced by each algorithm for different users. The average Rand Distance between clusterings produced with document supervision and dual supervision with 4 documents and associated keywords is about 0.23, which is very close to the average Distance between different users' manual organization (about 0.25). On the other hand, the Rand Distance between clusterings produced without supervision is only 0.08. Therefore, clusterings obtained via semi-supervised clustering with user supervision are highly depended on user input, and therefore they can be viewed as personalized clusterings.

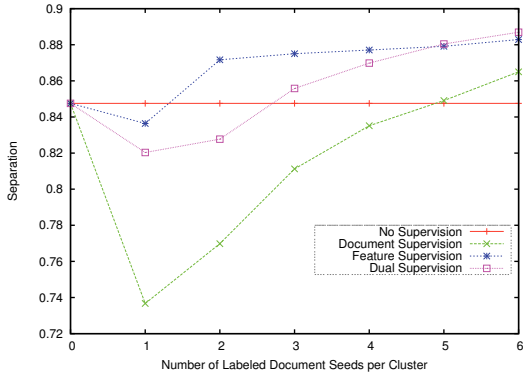
organizations is 0.25, while from the same user is 0.15.

Document Clustering with User Supervision

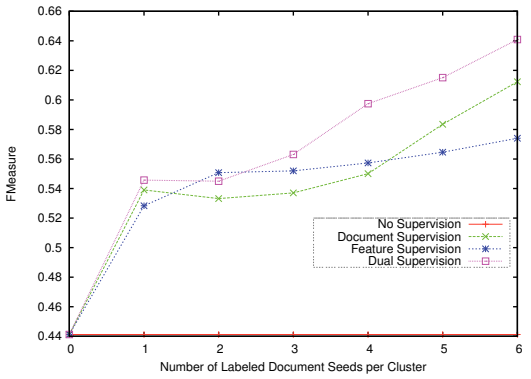
Semi-supervised clustering algorithms have been demonstrated to be able to improve clustering performance over unsupervised peer algorithms using oracles [21, 22, 24, 25]. Since oracles used in previous work are assumed to give "correct" answer all the time, our purpose here is to verify that document clustering with human's noisy supervision can still produce more consistent clusters with user's manual organization than unsupervised clustering techniques. Instead of using a single universal ground truth as in previous work, each user has his own ground truth in our case. Therefore, we also explore whether document clustering with user supervision can produce personalized clusters. We present the results of clustering algorithms initialized by a few (1 to 6) documents and the keywords labeled in those documents or assigned directly into the clusters. We use *coh*, *sep*, and F-Measure to quantify the consistency of the computed clustering with the user's manual organization. We observe that document supervision is able to group similar documents together while feature



(a) Cohesiveness of clusters: measuring similarity of documents in one cluster



(b) Separation of clusters: measuring dissimilarity between clusters



(c) F-Measure of clusters: measuring both cohesiveness and separation

Figure 7.2: Performance of clustering algorithms with no supervision, document supervision, feature supervision and dual supervision. Dual supervision, can generally produce clusters better matching user's expectation than no supervision or any single supervision, i.e., document supervision and feature supervision. Since assigning keywords is efficient for users, it is worth the effort to improve the clustering performance.

supervision is better at separating dissimilar documents (Fig. 7.2(a) and Fig. 7.2(b)). A small number of labeled documents appears to lead to unbalanced clusters in terms of size with high cohesiveness (e.g. *coh* is 1 when all documents are place into the a single cluster), since the labeled documents can not represent the cluster structure well. We observe this behavior in Fig. 7.2(b). A small number of keywords assigned to clusters (through the labeled documents or directly by the user) appears to lead to more balanced clusters with a better representation of cluster structures, as seen in Fig. 7.2(b). Especially, document supervision can not separate dissimilar documents very well into different clusters when less than 4 labeled documents per cluster are provided (Fig. 7.2(b)). Since the keywords (features) assigned by users are representative of the clusters, feature supervision can provide good performance in terms of both cohesiveness and separation. Dual supervision, the combination of document supervision and feature supervision, can generally produce clusters better matching user expectation (Fig. 7.2)[7]. Since assigning keywords is efficient for users, it is worth the effort in order to improve the clustering performance. In addition, the performance of the clustering algorithms improves with more labeled documents (and more assigned keywords) for initialization. This is easily understandable since more documents and/or keywords can represent the cluster structures better.

We also investigate how consistent the clusterings are between different participants. We compute and display the average Rand Distances between clusterings generated by the same algorithm for different users with same type and amount of supervision in Table 7.9. Like the manual organizations from each user (Table 7.7), the clusterings produced with user's input are also distinct from each other (Table 7.9). The average Rand Distance between clusterings produced with document supervision and dual supervision with 4 documents and associated keywords is about 0.23 (Table 7.9), which is very close to the average Rand Distance between different users' manual organization (about 0.25 in Table 7.7). On the other hand, the Rand Distance between clusterings produced without supervision is only 0.08. Therefore, clusterings obtained via semi-supervised clustering with user supervision are highly depended on user input, and therefore they can be viewed as personalized clusterings.

[7]Two-tailed paired t-test with p = 0.05.

7.4 Future Work

Since text cloud with single words is not as helpful as we expected, it is worth to further investigate the utility of the text cloud with multiple-word terms. According to user feedback, functions such as searching documents by words, retrieving documents in a cluster with a specific keyword of that cluster, and a spell checker should be added to the user interface in the future. Since a user changes his perception of the document collection during exploration, the software should be able to interleave user supervision and clustering, i.e., the user should be able to make adjustments of documents and features after intermediate clusters are obtained [7], and then the clustering procedure is repeated with the updated user input. Other future work directions include enabling users to create hierarchical clusters with the user interface and allowing soft clustering, namely, a document to be assigned to multiple clusters.

7.5 Summary

Thirty-two participants were recuited to organize the same document collection. We analyzed users' behaviors during their manual organization. The analysis shows that users can easily find the keywords to assign to a cluster based on the whole content of the documents and it is efficient according to users' feedback. Instead of only assigning keywords existing in the documents, users also like to come up with phrases to describe the topics of clusters. By comparing all groupings from all participants, we find that each user has his own perception of the document collection and a clustering algorithm with user supervision is required to produce personalized clusters, which better reflect his point of view. At the same time, we confirm that previously proposed semi-supervised document clustering algorithms can produce personalized clusters with a small amount of user input even if it is noisy. It is also demonstrated that the same user can change his perception of the documents over time. Therefore, operations such as moving a document between clusters and merging two clusters should be available in software for document clustering. We also find that text cloud with single words is less useful than the full text for users to grasp the topic of a document.

Chapter 8

Conclusions and Future Work

In this thesis, the focus of the research is to produce personalized clusters of the same document collection such as a personal library of papers based on individual user's point of view. This problem can be tackled using clustering techniques. However, traditional unsupervised clustering gives a universal output without any personalization, which users may find unsatisfactory. Therefore, prior knowledge from individual users has to be incorporated into clustering algorithms in order to generate customized clusters for different users. Various semi-supervised clustering algorithms have been designed to work with labeled instances (documents) to improve clustering performance. Previous semi-supervised algorithms are mainly used to improve clustering performance, while the potential for semi-supervised clustering algorithms to produce personalized clusters has not been explored. Besides labeling documents, an alternative form, i.e., labeling features for document clustering, exists for user supervision. In addition, it has been argued that labeling documents is more labor-intensive compared to labeling features [40]. This supervision is called feature supervision. Correspondingly, labeling documents is called document supervision and the joint use of document supervision and feature supervision is called dual supervision. In this thesis, we proposed frameworks and algorithms to incorporate feature supervision into traditional unsupervised and semi-supervised clustering algorithms. We also present a user study, which confirms that different users have their own personalized organizations of the same document collection. In the user study, the framework for document clustering with dual supervision is evaluated and confirmed to perform well even with noisy user input. Generally speaking, the research was done and evaluated in two steps: (1) Oracle-based. The proposed frameworks and algorithms are evaluated by oracles. Oracles are based on the underlying class labels of the standard datasets to simulate human users providing document and feature

supervision. (2) Human Users. The framework for document clustering with dual supervision is evaluated and validated by noisy input provided by human users. It also validates that semi-supervised clustering techniques are able to produce personalized clusters with user input from individual users.

The methodological contributions of the thesis include the following.

First, we designed and created a framework that enable users to label features by indicating whether they are useful for clustering from a ordered list of features ranked by the χ^2 based on intermediate clusters. The framework interleaves interactive feature selection and clustering iteratively until users are satisfied with the generated clusters. The advantage of this framework is that the features to be labeled are recommended to users so that they do not need to read whole documents to find the features. However, it might be difficult for users to judge whether a feature is useful or not without context.

Second, we introduced feature supervision through feature reweighting in combination with document supervision and experimented with various existing semi-supervised clustering algorithms extended to incorporate feature supervision. In pratice, the features could be labeled while labeling documents through text cloud or document content.

Third, we present a novel unified framework to combine document supervision and feature supervision through seeding. Instead of modifying the document representations through feature reweighting, features associated with documents are used directly to initialize the clusters by using feature seeding.

Finally, we conducted a user study to evaluate our newly designed document clustering algorithms with dual supervision. We designed an interactive user interface, which enabled users to group the documents flexibly. Through the user study, we observe that different users have their own personalized organizations of the same collection and a user's organization changes over time. Therefore, we propose that document clustering algorithms should be able to incorporate user input and produce personalized clusterings based on user needs.

Since the frameworks proposed in this thesis are based on efficient clustering algorithms, i.e., KMeans and Multinomial clustering, they are scaled to quite large datasets and can be applied to obtain personalized web. In fact, any clustering

algorithm can be used as the underlying algorithm with the dual supervision according to user needs such as time requirement and performance. One limitation of this work is the quite complicated interface design, which requires the users to spend some time to become familiar with it. In addition, the frameworks do not support hierarchical clustering, which is a popular document management method. However, the concept of dual supervision can be easily extended to accommodate hierarchical clustering.

In this thesis, we explored different visualization methods to present features such as a ranked list in Chapter 4, text cloud and whole content in Chapter 5, 6 and 7. It is natural to combine the visualization methods together so that users can easily find the features they want to label and also have easy access to the context at the same time. We also explored two techniques to combine feature supervision with document supervision, feature reweighting in Chapter 5 and feature seeding in Chapter 6. Since these two methods are complementary to each other, we can use feature reweighting to modify the document representations and feature seeding to influence the clustering algorithm in a unified framework. A future work direction includes introducing feature supervision into hierarchical clustering, in which the features associated with a parent cluster should be more general than those of children clusters. In addition, the concept of dual supervision can also be applied to text streams. Other possible future work specific to chapters is presented in Chapter 5 and 7.

Bibliography

[1] D.C. Anastasiu, B.J. Gao, and D. Buttler. Clusteringwiki: personalized and collaborative clustering of search results. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1263–1264, New York, NY, USA, 2011. ACM.

[2] 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*, pages 40–55. Springer, 2010.

[3] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations. In *International Conference on Machine Learning*, volume 20, page 11, 2003.

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

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

[6] 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.

[7] R. Bekkerman, H. Raghavan, J. Allan, and K. Eguchi. Interactive clustering of text collections according to a user-specified criterion. In *Proceedings of the 2007 International Joint Conference on Artificial Intelligence*, volume 20, 2007.

[8] 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.

[9] M. Bilenko, S. Basu, and R.J. Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *Proceedings of the Twenty-First International Conference on Machine Learning*, pages 81–88, 2004.

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

[11] P. S. Bradley, K. P. Bennett, and A. Demiriz. Constrained k-means clustering. *MSR-TR-2000-65, Microsoft Research*, 2000.

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

[13] F. Debole and F. Sebastiani. Supervised term weighting for automated text categorization. In *Proceedings of the 2003 ACM Symposium on Applied Computing*, pages 784–788. ACM, 2003.

[14] 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.

[15] I.S. Dhillon, S. Mallela, and D.S. Modha. Information-theoretic co-clustering. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 89–98. ACM, 2003. ISBN 1581137370.

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

[17] G. Druck, G. Mann, and A. McCallum. Learning from labeled features using generalized expectation criteria. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 595–602. ACM, 2008.

[18] S.M. Drucker, D. Fisher, and S. Basu. Helping Users Sort Faster with Adaptive Machine Learning Recommendations. In *Proceedings of the 13th International conference on Human-Computer Interaction*, pages 187–203, 2011.

[19] S. Few. *Information Dashboard Design*. O'Reilly Media, 2006.

[20] D. Greene and P. Cunningham. Constraint selection by committee: An ensemble approach to identifying informative constraints for semi-supervised clustering. *Machine Learning: ECML 2007*, pages 140–151, 2007.

[21] 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.

[22] 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"*, pages 1148–1155. ACM Special Interest Group on Applied Computing, 2011.

[23] 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.

[24] Y. Hu, E.E. Milios, and J. Blustein. Enhancing Semi-supervised Document Clustering with Feature Supervision. In *Proceedings of the 27th ACM Symposium Applied Computing, On Track "Information Access and Retrieval"*, pages 950–957. ACM, 2012.

[25] Y. Hu, E.E. Milios, and J. Blustein. Semi-supervised Document Clustering with Dual Supervision through Seeding. In *Proceedings of the 27th ACM Symposium Applied Computing, On Track "Data Mining"*, pages 463–470. ACM, 2012.

[26] Y. Hu, E.E. Milios, and J. Blustein. A unified framework for document clustering with dual supervision. *ACM Applied Computing Review*, 12(2), 2012.

[27] R. Huang and W. Lam. Semi-supervised Document Clustering via Active Learning with Pairwise Constraints. In *Proceedings of the 2007 IEEE International Conference on Data Mining*, pages 517–522. IEEE Computer Society, 2007.

[28] R. Huang and W. Lam. An active learning framework for semi-supervised document clustering with language modeling. *Data & Knowledge Engineering*, 68(1): 49–67, 2009.

[29] R. Huang, W. Lam, and Z. Zhang. Active learning of constraints for semi-supervised text clustering. In *Proceedings of the SIAM International Conference on Data Mining*, pages 113–124, 2007.

[30] 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.

[31] 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.

[32] J. Lamantia. Text Clouds: A New Form of Tag Cloud? http://www.joelamantia.com/tag-clouds/text-clouds-a-new-form-of-tag-cloud, 2007. Accessed on April 12, 2012.

[33] K.W. Leung, W. Ng, and D. Lee. Personalized concept-based clustering of search engine queries. *IEEE Trans. on Knowl. and Data Eng.*, 20(11):1505–1518, 2008.

[34] D.D. Lewis. Evaluating text categorization. In *Proceedings of Speech and Natural Language Workshop*, pages 312–318. Defense Advanced Research Projects Agency, Morgan Kaufmann, Feburary 1991.

[35] 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.

[36] 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, 2004.

[37] P. Melville, W. Gryc, and R.D. Lawrence. Sentiment analysis of blogs by combining lexical knowledge with text classification. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1275–1284, 2009.

[38] 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.

[39] M.F. Porter et al. An algorithm for suffix stripping, 1980.

[40] 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.

[41] W.M. Rand. Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association*, pages 846–850, 1971.

[42] M. Rigou, S. Sirmakessis, and G. Tzimas. A method for personalized clustering in data intensive web applications. In *Proceedings of the Joint International Workshop on Adaptivity, Personalization & the Semantic Web*, pages 35–40, New York, NY, USA, 2006. ACM.

[43] 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.

[44] N. Roy and A. McCallum. Toward optimal active learning through sampling estimation of error reduction. In *International Conference on Machine Learning*, pages 441–448, 2001.

[45] N. Shental, T. Hertz, D. Weinshall, and M. Pavel. Adjustment learning and relevant component analysis. *Computer Vision—ECCV 2002*, pages 181–185, 2006.

[46] V. Sindhwani and P. Melville. Document-word co-regularization for semi-supervised sentiment analysis. In *Proceedings of the Eighth IEEE International Conference on Data Mining*, pages 1025–1030. IEEE, 2009.

[47] V. Sindhwani, P. Melville, and R.D. Lawrence. Uncertainty sampling and transductive experimental design for active dual supervision. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 953–960. ACM, 2009.

[48] D. Sun and D. Zhang. Bagging Constraint Score for feature selection with pairwise constraints. *Pattern Recognition*, 43(6):2106–2118, 2010. ISSN 0031-3203.

[49] P. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Pearson Addison Wesley, 2005.

[50] 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, April 23 2005.

[51] 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.

[52] K. Wagstaff. *Intelligent Clustering with Instance-Level Constraints*. PhD thesis, Cornell University, 2002.

[53] 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*, pages 577–584, 2001.

[54] X. Wu and R. Srihari. Incorporating prior knowledge with weighted margin support vector machines. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 326–333. ACM, 2004. ISBN 1581138881.

[55] E.P. Xing, A.Y. Ng, M.I. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. *Advances in Neural Information Processing Systems*, pages 521–528, 2003.

[56] D. Zhang, S. Chen, and Z.H. Zhou. Constraint Score: A new filter method for feature selection with pairwise constraints. *Pattern Recognition*, 41(5):1440–1451, 2008. ISSN 0031-3203.

Appendix A

Copyright Permission

The following is an excerpt from ACM Copyright Policy, version 7[1]:

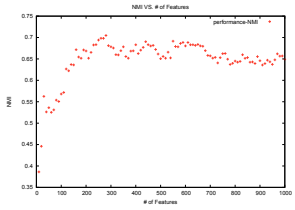
> (...)
> 2.5 Rights Retained by Authors and Original Copyright Holders Under the ACM copyright transfer agreement, the original copyright holder retains:
>
> - all other proprietary rights to the work such as patent
> - the right to reuse any portion of the work, without fee, in future works of the author's own, including books, lectures and presentations in all media, provided that the ACM citation and notice of the Copyright are included
>
> (...)

The policy applies to four articles written by the author of the thesis, which were used in the preparation of this thesis (see Section 1.5 for details).

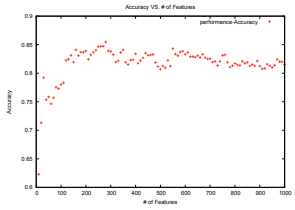
[1]Full policy available at: `http://www.acm.org/publications/policies/copyrightpolicy`

Appendix B

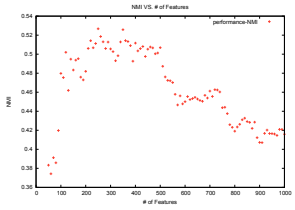
Experimental Results (All Figures) for Chapter 4



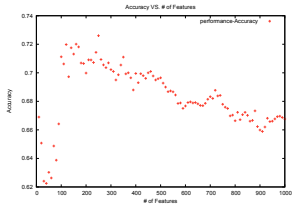
(a) *news-diff-3*,NMI



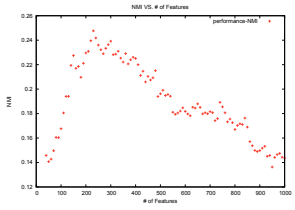
(b) *news-diff-3*,Accuracy



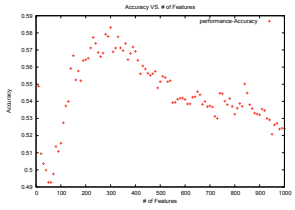
(c) *news-related-3*,NMI



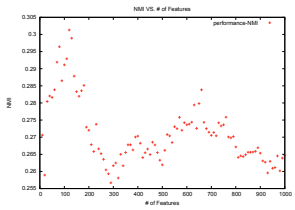
(d) *news-related-3*,Accuracy



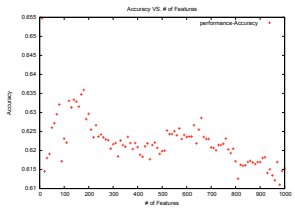
(e) *news-similar-3*,NMI



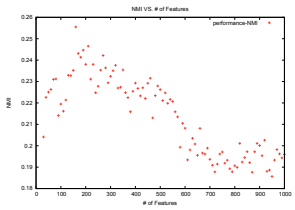
(f) *news-similar-3*,Accruacy



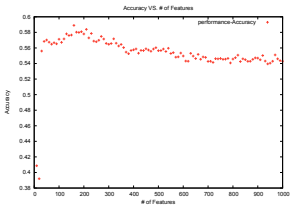
(g) ACM (*D2-D2&D3-D3*),NMI



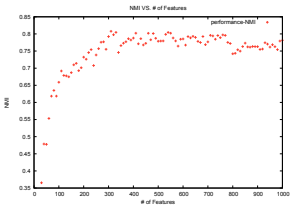
(h) ACM (*D2-D2&D3-D3*),Accuracy



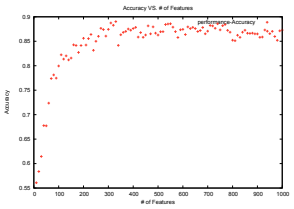
(i) ACM (*D-H-I*),NMI



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

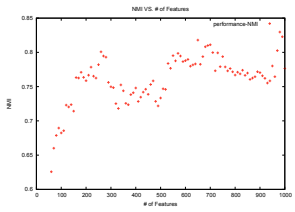


(k) *3-classic-abstract*,NMI

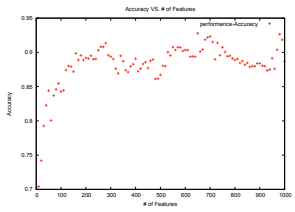


(l) *3-classic-abstract*,Accuracy

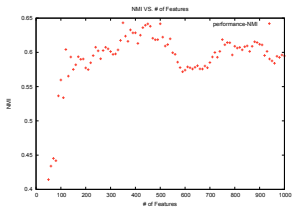
Figure B.1: Effect of feature set sizes on different datasets (a),(c), (e),(g), (i),(k) use NMI as performance measure. (b),(d), (f),(h), (j),(l) use accuracy as performance measure. x axis-feature set size, y axis-the corresponding performance, NMI or Accuracy.



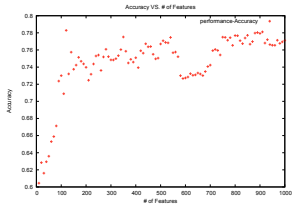
(a) *news-diff-3*,NMI



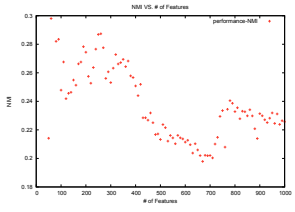
(b) *news-diff-3*,Accuracy



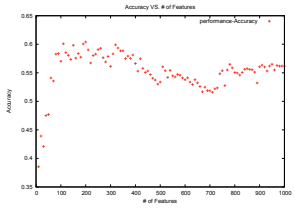
(c) *news-related-3*,NMI



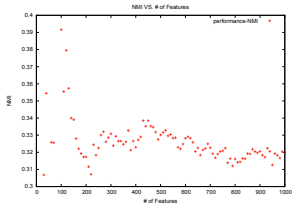
(d) *news-related-3*,Accuracy



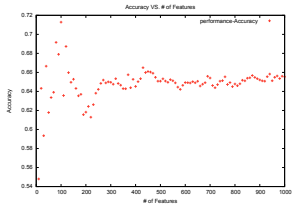
(e) *news-similar-3*,NMI



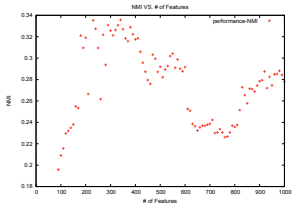
(f) *news-similar-3*,Accuracy



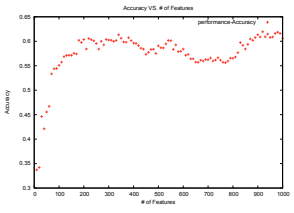
(g) ACM (*D2-D2&D3-D3*),NMI



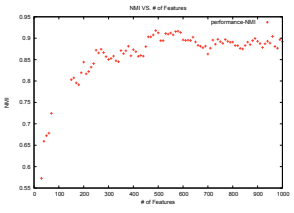
(h) ACM (*D2-D2&D3-D3*),Accuracy



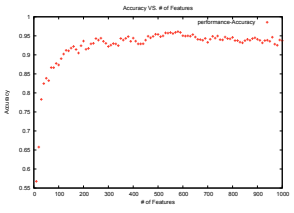
(i) ACM (*D-H-I*),NMI



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

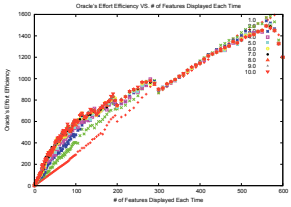


(k) *3-classic-abstract*,NMI

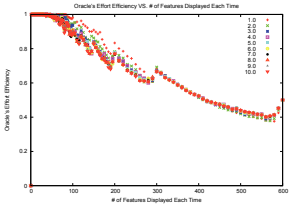


(l) *3-classic-abstract*,Accuracy

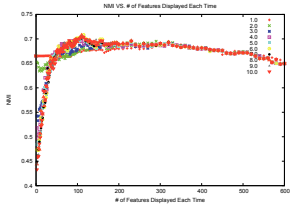
Figure B.2: Effect of feature set sizes on different datasets (a),(c), (e),(g), (i),(k) use NMI as performance measure. (b),(d), (f),(h), (j),(l) use accuracy as performance measure. x axis-feature set size, y axis-the corresponding performance, NMI or Accuracy.



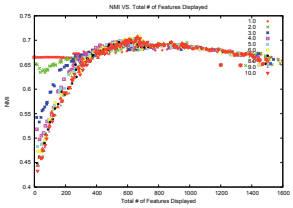
(a) KMeans on *news-diff-3* dataset: (x axis-f) vs. (y axis-f_{total})



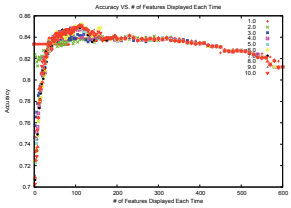
(b) KMeans on *news-diff-3* dataset: (x axis-f) vs. (y axis-eff-eff)



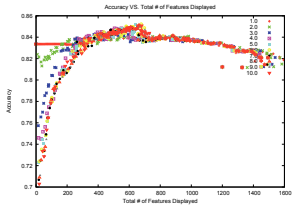
(c) KMeans on *news-diff-3* dataset: (x axis-f) vs. (y axis-NMI)



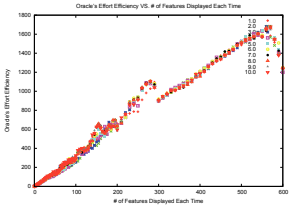
(d) KMeans on *news-diff-3* dataset: (x axis-f_{total}) vs. (y axis-NMI)



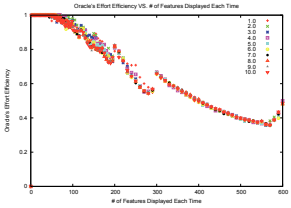
(e) KMeans on *news-diff-3* dataset: (x axis-f) vs. (y axis-$Accuracy$)



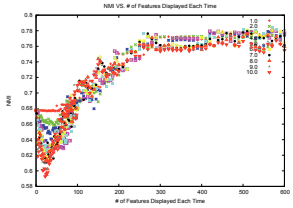
(f) KMeans on *news-diff-3* dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)



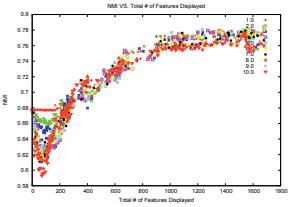
(g) *EM-NB* on *news-diff-3* dataset: (x axis-f) vs. (y axis-f_{total})



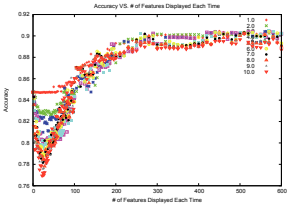
(h) *EM-NB* on *news-diff-3* dataset: (x axis-f) vs. (y axis-eff-eff)



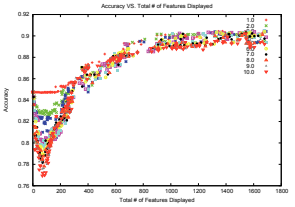
(i) *EM-NB* on *news-diff-3* dataset: (x axis-f) vs. (y axis-NMI)



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

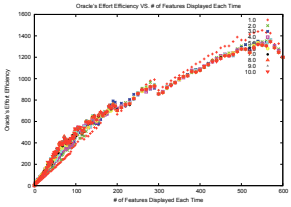


(k) *EM-NB* on *news-diff-3* dataset: (x axis-f) vs. (y axis-$Accuracy$)

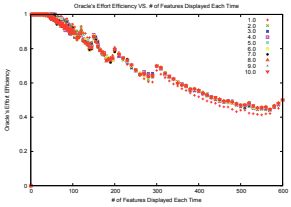


(l) *EM-NB* on *news-diff-3* dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

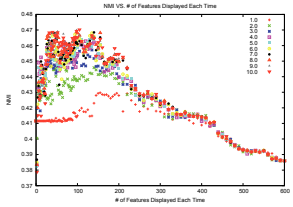
Figure B.3: Effect of user effort on *news-diff-3* dataset. (a), (b), (c), (d), (e) and (f) use KMeans as the underlying algorithm while (g), (h), (i), (j), (k) and (l) use *EM-NB* as the underlying algorithm. Legends are all weights between 1 and 10.



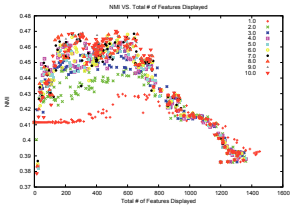
(a) KMeans on *news-related-3* dataset: (x axis-f) vs. (y axis-f_{total})



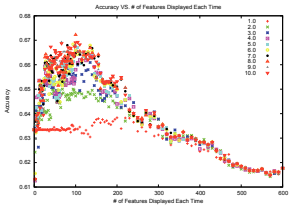
(b) KMeans on *news-related-3* dataset: (x axis-f) vs. (y axis-eff-eff)



(c) KMeans on *news-related-3* dataset: (x axis-f) vs. (y axis-NMI)

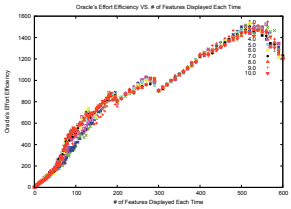


(d) KMeans on *news-related-3* dataset: (x axis-f_{total}) vs. (y axis-NMI)

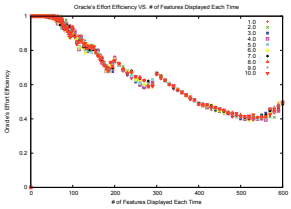


(e) KMeans on *news-related-3* dataset: (x axis-f) vs. (y axis-$Accuracy$)

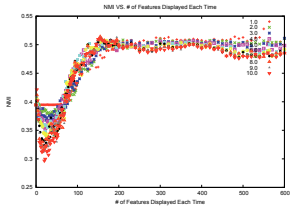
(f) KMeans on *news-related-3* dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)



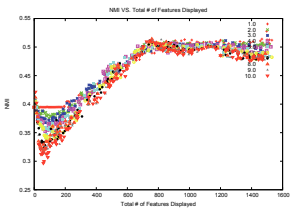
(g) *EM-NB* on *news-related-3* dataset: (x axis-f) vs. (y axis-f_{total})



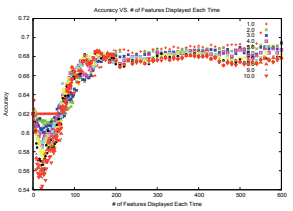
(h) *EM-NB* on *news-related-3* dataset: (x axis-f) vs. (y axis-eff-eff)



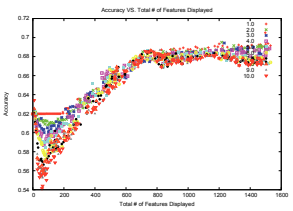
(i) *EM-NB* on *news-related-3* dataset: (x axis-f) vs. (y axis-NMI)



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

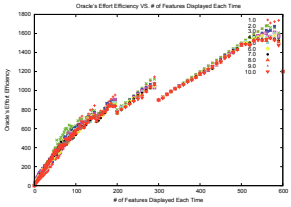


(k) *EM-NB* on *news-related-3* dataset: (x axis-f) vs. (y axis-$Accuracy$)

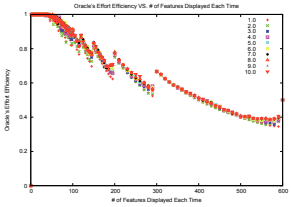


(l) EM on *news-related-3* dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

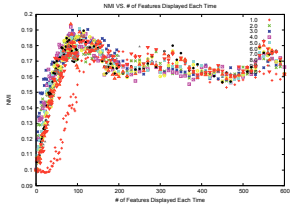
Figure B.4: Effect of user effort on *news-related-3* dataset. (a), (b), (c), (d), (e) and (f) use KMeans as the underlying algorithm while (g), (h), (i), (j), (k) and (l) use *EM-NB* as the underlying algorithm. Legends are all weights between 1 and 10.



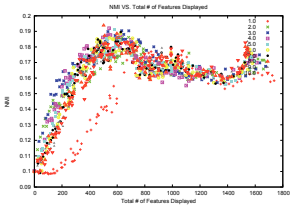
(a) KMeans on *news-similar-3* dataset: (x axis-f) vs. (y axis-f_{total})



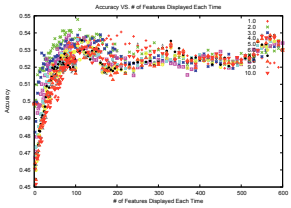
(b) KMeans on *news-similar-3* dataset: (x axis-f) vs. (y axis-eff-eff)



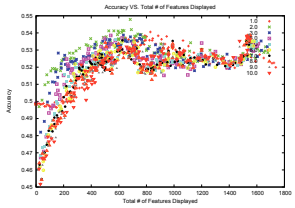
(c) KMeans on *news-similar-3* dataset: (x axis-f) vs. (y axis-NMI)



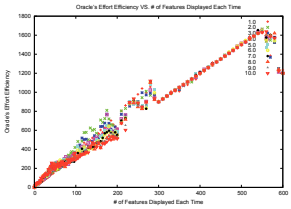
(d) KMeans on *news-similar-3* dataset: (x axis-f_{total}) vs. (y axis-NMI)



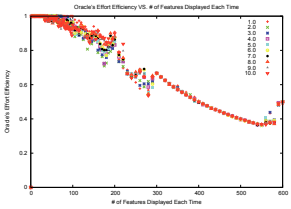
(e) KMeans on *news-similar-3* dataset: (x axis-f) vs. (y axis-$Accuracy$)



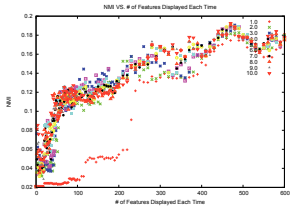
(f) KMeans on *news-similar-3* dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)



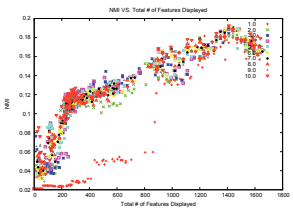
(g) *EM-NB* on *news-similar-3* dataset: (x axis-f) vs. (y axis-f_{total})



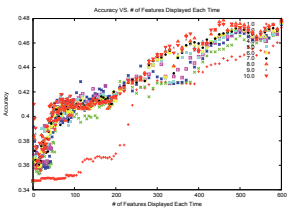
(h) *EM-NB* on *news-similar-3* dataset: (x axis-f) vs. (y axis-eff-eff)



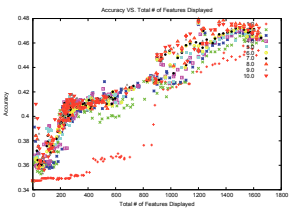
(i) *EM-NB* on *news-similar-3* dataset: (x axis-f) vs. (y axis-NMI)



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

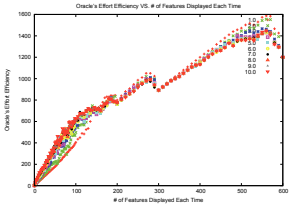


(k) *EM-NB* on *news-similar-3* dataset: (x axis-f) vs. (y axis-$Accuracy$)

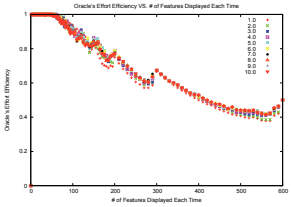


(l) EM on *news-similar-3* dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

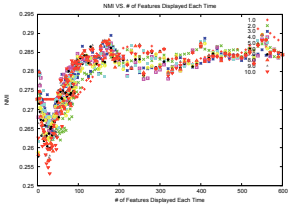
Figure B.5: Effect of user effort on *news-similar-3* dataset. (a), (b), (c), (d), (e) and (f) use KMeans as the underlying algorithm while (g), (a), (b), (c), (k) and (l) use *EM-NB* as the underlying algorithm. Legends are all weights between 1 and 10.



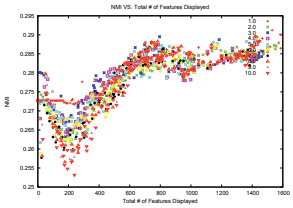
(a) KMeans on *D2-D2&D3-D3* dataset: (x axis-f) vs. (y axis-f_{total})



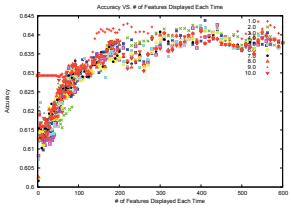
(b) KMeans on *D2-D2&D3-D3* dataset: (x axis-f) vs. (y axis-eff-eff)



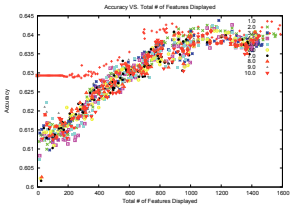
(c) KMeans on *D2-D2&D3-D3* dataset: (x axis-f) vs. (y axis-NMI)



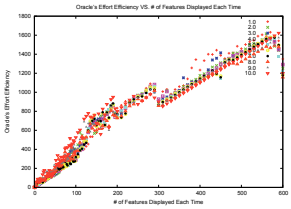
(d) KMeans on *D2-D2&D3-D3* dataset: (x axis-f_{total}) vs. (y axis-NMI)



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

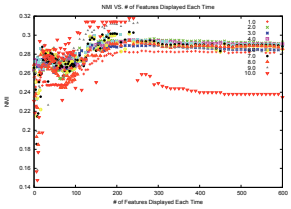


(f) KMeans on *D2-D2&D3-D3* dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

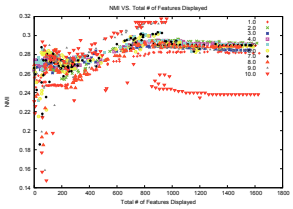


(g) *EM-NB* on *D2-D2&D3-D3* dataset: (x axis-f) vs. (y axis-f_{total})

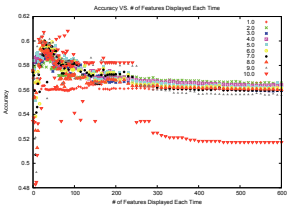
(h) *EM-NB* on *D2-D2&D3-D3* dataset: (x axis-f) vs. (y axis-eff-eff)



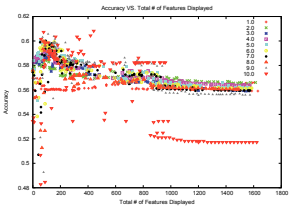
(i) *EM-NB* on *D2-D2&D3-D3* dataset: (x axis-f) vs. (y axis-NMI)



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

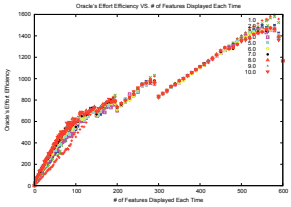


(k) *EM-NB* on *D2-D2&D3-D3* dataset: (x axis-f) vs. (y axis-$Accuracy$)

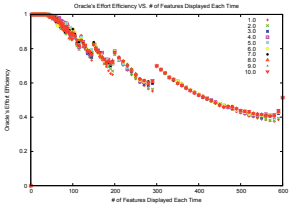


(l) EM on *D2-D2&D3-D3* dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

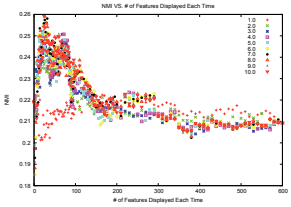
Figure B.6: Effect of user effort on *D2-D2&D3-D3* dataset. (a), (b), (c), (d), (e) and (f) use KMeans as the underlying algorithm while (g), (h), (i), (j), (k) and (l) use *EM-NB* as the underlying algorithm. Legends are all weights between 1 and 10.



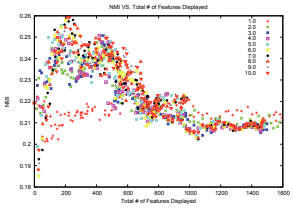
(a) KMeans on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-f_{total})



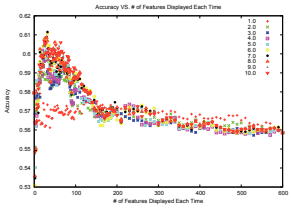
(b) KMeans on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-eff-eff)



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

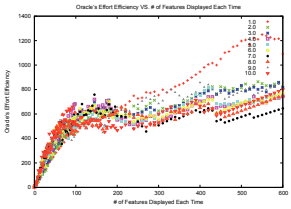


(d) KMeans on ACM (D-H-I) dataset: (x axis-f_{total}) vs. (y axis-NMI)

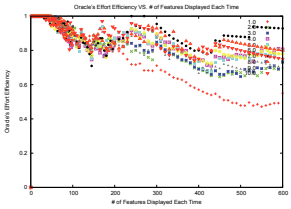


(e) KMeans on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-$Accuracy$)

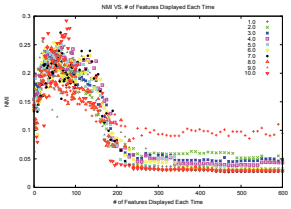
(f) KMeans on ACM (D-H-I) dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)



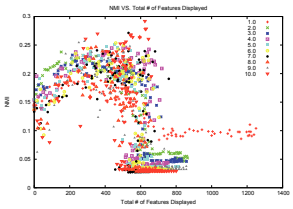
(g) *EM-NB* on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-f_{total})



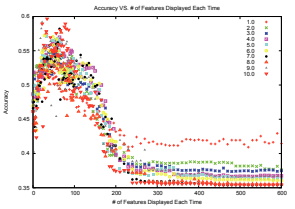
(h) *EM-NB* on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-eff-eff)



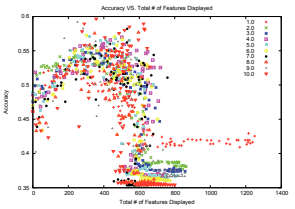
(i) *EM-NB* on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-NMI)



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

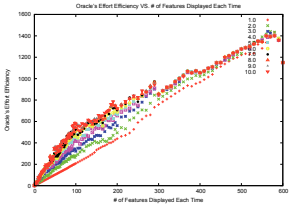


(k) *EM-NB* on ACM (D-H-I) dataset: (x axis-f) vs. (y axis-$Accuracy$)

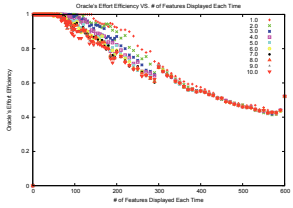


(l) *EM-NB* on ACM (D-H-I) dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

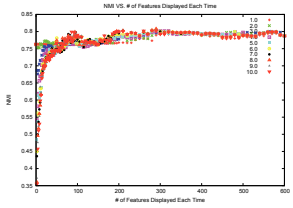
Figure B.7: Effect of user effort on ACM (D-H-I) dataset. (a), (b), (c), (d), (e) and (f) use KMeans as the underlying algorithm while (g), (h), (i), (j), (k) and (l) use *EM-NB* as the underlying algorithm. Legends are all weights between 1 and 10.



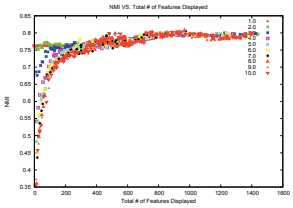
(a) KMeans on *3-classic-abstract* dataset: (x axis-f) vs. (y axis-f_{total})



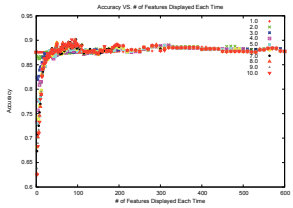
(b) KMeans on *3-classic-abstract* dataset: (x axis-f) vs. (y axis-eff-eff)



(c) KMeans on *3-classic-abstract* dataset: (x axis-f) vs. (y axis-NMI)

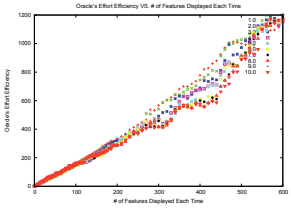


(d) KMeans on *3-classic-abstract* dataset: (x axis-f_{total}) vs. (y axis-NMI)

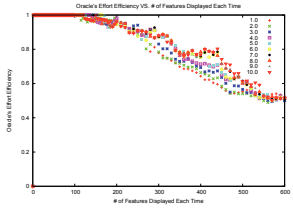


(e) KMeans on *3-classic-abstract* dataset: (x axis-f) vs. (y axis-$Accuracy$)

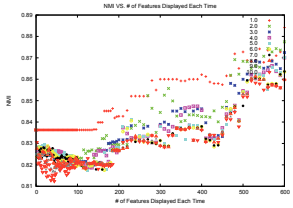
(f) KMeans on *3-classic-abstract* dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)



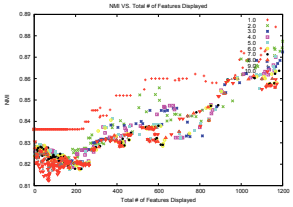
(g) *EM-NB* on *3-classic-abstract* dataset: (x axis-f) vs. (y axis-f_{total})



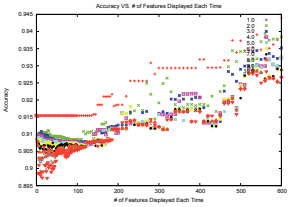
(h) *EM-NB* on *3-classic-abstract* dataset: (x axis-f) vs. (y axis-eff-eff)



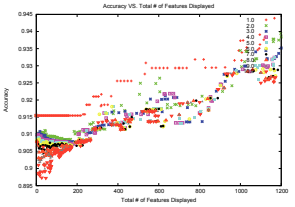
(i) *EM-NB* on *3-classic-abstract* dataset: (x axis-f) vs. (y axis-NMI)



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

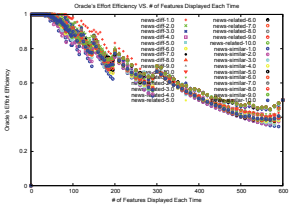


(k) *EM-NB* on *3-classic-abstract* dataset: (x axis-f) vs. (y axis-$Accuracy$)

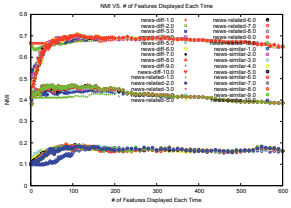


(l) *EM-NB* on *3-classic-abstract* dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

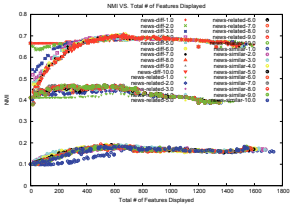
Figure B.8: Effect of user effort on *3-classic-abstract* dataset. (a), (b), (c), (d), (e) and (f) use KMeans as the underlying algorithm while (g), (h), (i), (j), (k) and (l) use *EM-NB* as the underlying algorithm. Legends are all weights between 1 and 10.



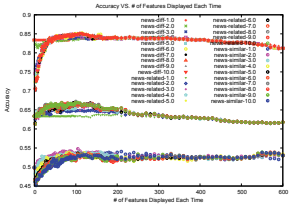
(a) KMeans on newsgroups datasets: (x axis-f) vs. (y axis-eff-eff)



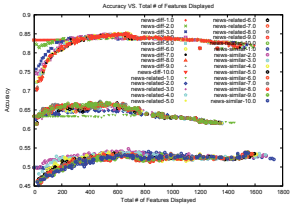
(b) KMeans on newsgroups dataset: (x axis-f) vs. (y axis-NMI)



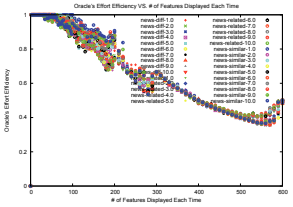
(c) KMeans on newsgroups dataset: (x axis-f_{total}) vs. (y axis-NMI)



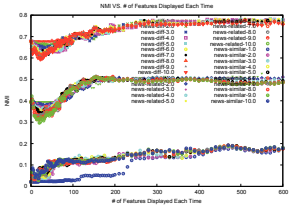
(d) KMeans on newsgroups dataset: (x axis-f) vs. (y axis-$Accuracy$)



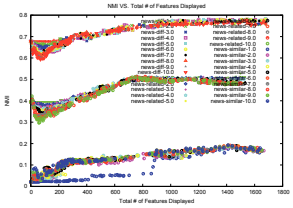
(e) KMeans on newsgroups dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)



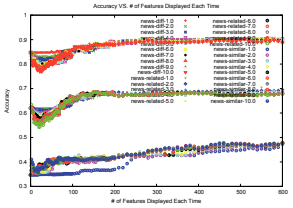
(f) *EM-NB* on newsgroups datasets: (x axis-f) vs. (y axis-eff-eff)



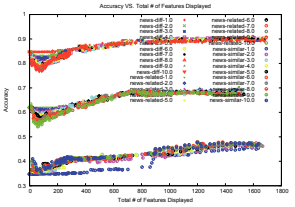
(g) *EM-NB* on newsgroups dataset: (x axis-f) vs. (y axis-NMI)



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

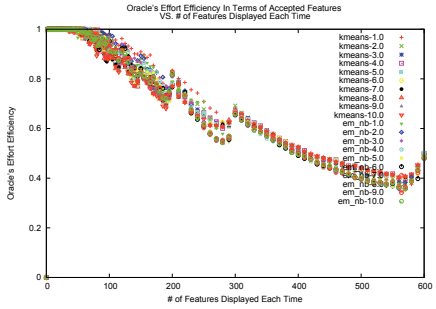


(i) *EM-NB* on newsgroups dataset: (x axis-f) vs. (y axis-$Accuracy$)

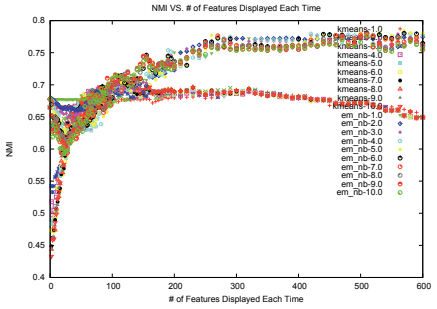


(j) *EM-NB* on newsgroups dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

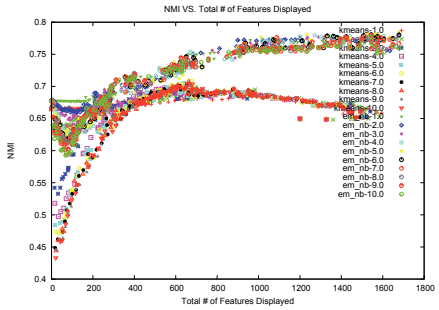
Figure B.9: DCIFS (Algorithm 4) with the same underlying algorithm on *news-diff-3*, *news-related-3*, *news-similar-3* datasets. Legends are all weights between 1 and 10 for all newsgroups datasets.



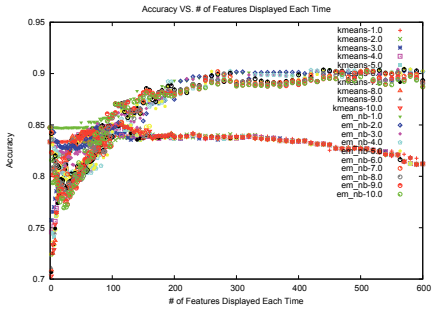
(a) KMeans and *EM-NB* on *news-diff-3* dataset: (x axis-f) vs. (y axis-eff-eff)



(b) KMeans and *EM-NB* on *news-diff-3* dataset: (x axis-f) vs. (y axis-NMI)



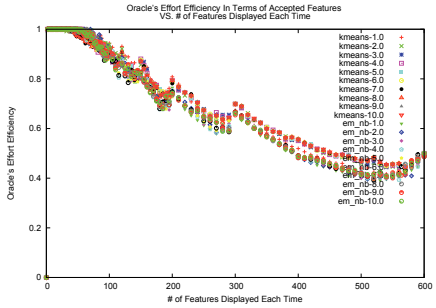
(c) KMeans and *EM-NB* on *news-diff-3* dataset: (x axis-f_{total}) vs. (y axis-NMI)



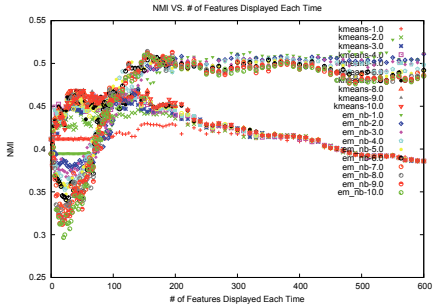
(d) KMeans and *EM-NB* on *news-diff-3* dataset: (x axis-f) vs. (y axis-$Accuracy$)

(e) KMeans and *EM-NB* on *news-diff-3* dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

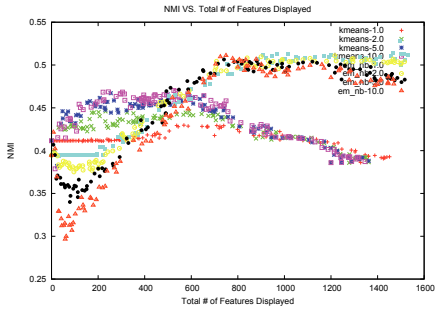
Figure B.10: DCIFS (Algorithm 4) with different underlying algorithms on the same new-diff dataset. Legends are all weights between 1 and 10 for both KMeans and *EM-NB*.



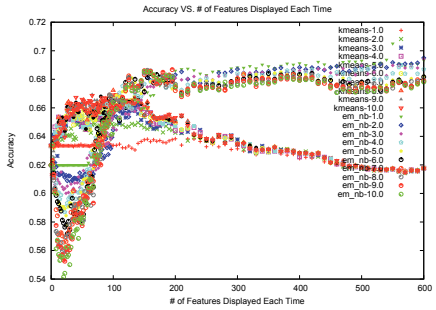
(a) KMeans and *EM-NB* on *news-related-3* dataset: (x axis-f) vs. (y axis-eff-eff)



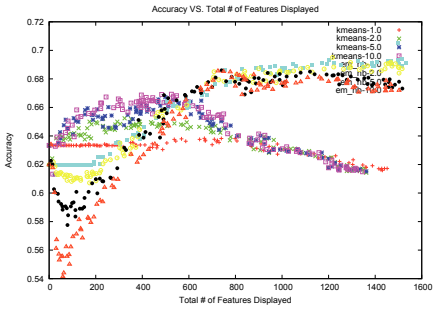
(b) KMeans and *EM-NB* on *news-related-3* dataset: (x axis-f) vs. (y axis-NMI)



(c) KMeans and *EM-NB* on *news-related-3* dataset: (x axis-f_{total}) vs. (y axis-NMI)

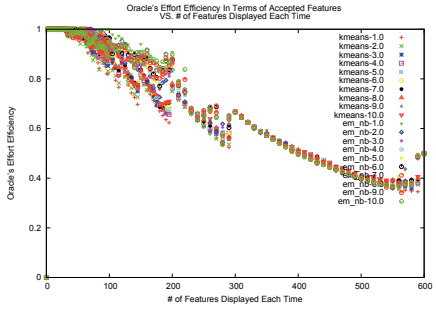


(d) KMeans and *EM-NB* on *news-related-3* dataset: (x axis-f) vs. (y axis-$Accuracy$)

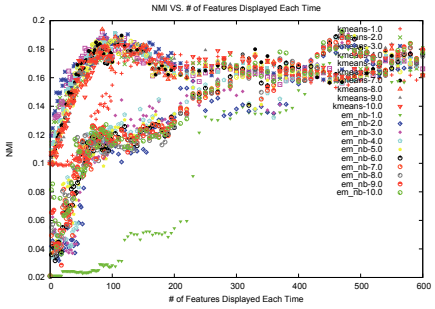


(e) KMeans and *EM-NB* on *news-related-3* dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

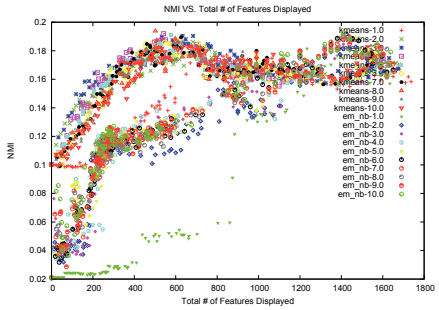
Figure B.11: DCIFS (Algorithm 4) with different underlying algorithms on the same *news-related-3* dataset. Legends are all weights between 1 and 10 for both KMeans and *EM-NB*.



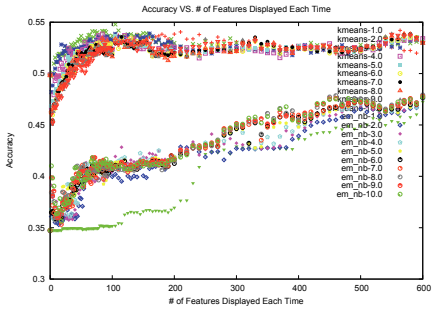
(a) KMeans and *EM-NB* on *news-similar-3* dataset: (x axis-f) vs. (y axis-eff-eff)



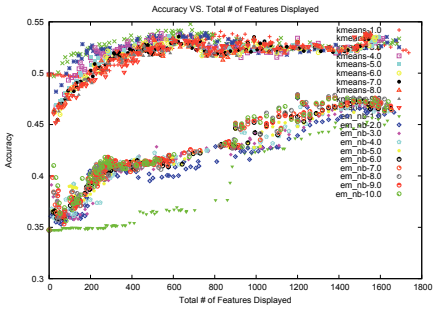
(b) KMeans and *EM-NB* on *news-similar-3* dataset: (x axis-f) vs. (y axis-NMI)



(c) KMeans and *EM-NB* on *news-similar-3* dataset: (x axis-f_{total}) vs. (y axis-NMI)

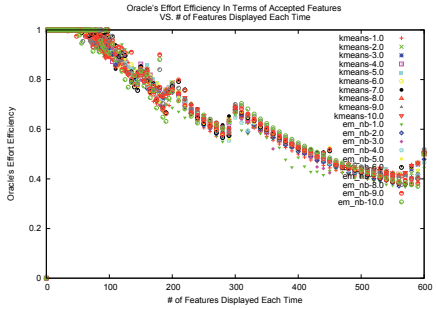


(d) KMeans and *EM-NB* on *news-similar-3* dataset: (x axis-f) vs. (y axis-$Accuracy$)

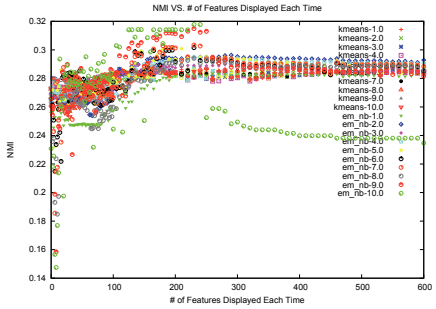


(e) KMeans and *EM-NB* on *news-similar-3* dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

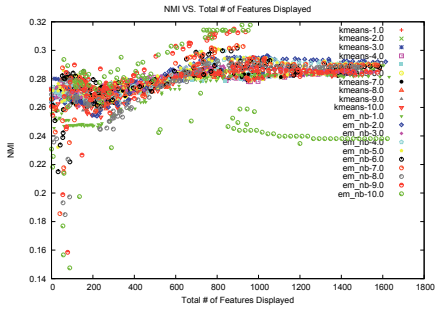
Figure B.12: DCIFS (Algorithm 4) with different underlying algorithms on the same *news-similar-3* dataset. Legends are all weights between 1 and 10 for both KMeans and *EM-NB*.



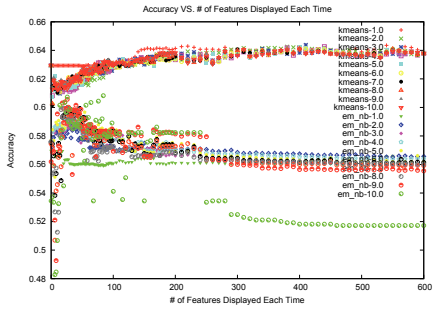
(a) KMeans and *EM-NB* on ACM (*D2-D2&D3-D3*) dataset: (x axis-f) vs. (y axis-eff-eff)



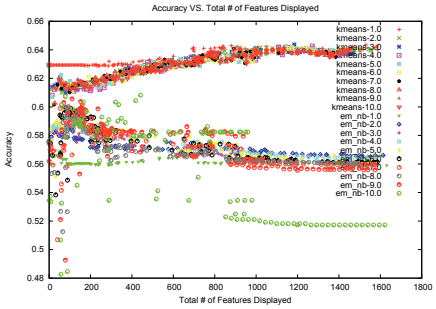
(b) KMeans and *EM-NB* on ACM (*D2-D2&D3-D3*) dataset: (x axis-f) vs. (y axis-NMI)



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

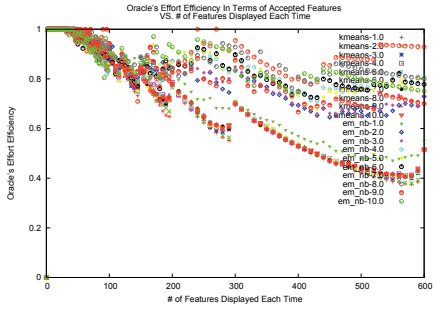


(d) KMeans and *EM-NB* on ACM (*D2-D2&D3-D3*) dataset: (x axis-f) vs. (y axis-$Accuracy$)

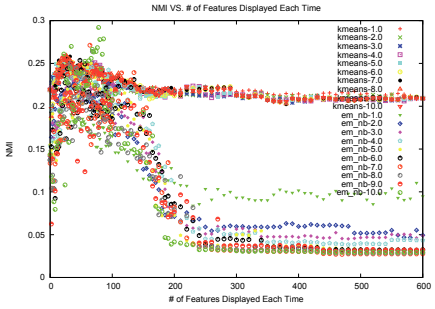


(e) KMeans and *EM-NB* on ACM (*D2-D2&D3-D3*) dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

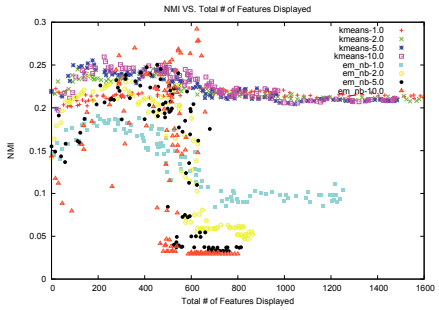
Figure B.13: DCIFS (Algorithm 4) with different underlying algorithms on the same ACM (*D2-D2&D3-D3*) dataset. Legends are all weights between 1 and 10 for both KMeans and *EM-NB*.



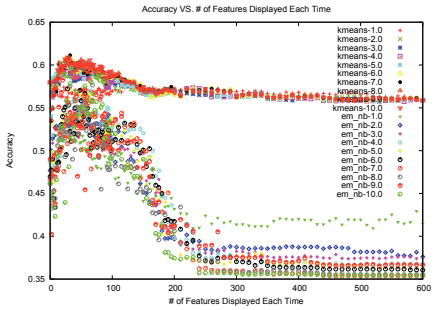
(a) KMeans and *EM-NB* on ACM (*D-H-I*) dataset: (x axis-f) vs. (y axis-eff-eff)



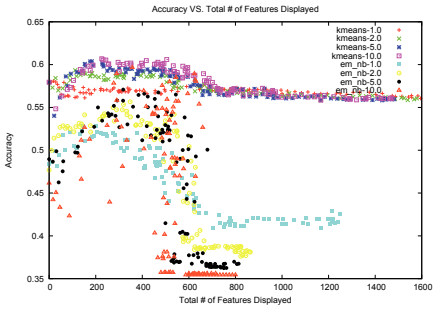
(b) KMeans and *EM-NB* on ACM (*D-H-I*) dataset: (x axis-f) vs. (y axis-NMI)



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

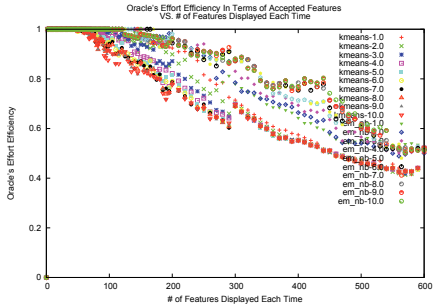


(d) KMeans and *EM-NB* on ACM (*D-H-I*) dataset: (x axis-f) vs. (y axis-$Accuracy$)

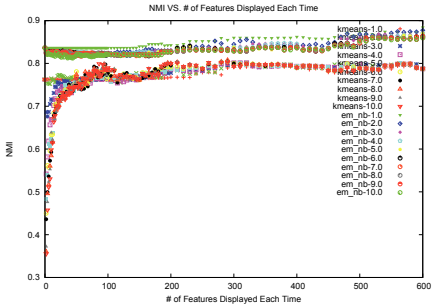


(e) KMeans and *EM-NB* on ACM (*D-H-I*) dataset: (x axis-f_{total}) vs. (y axis-$Accuracy$)

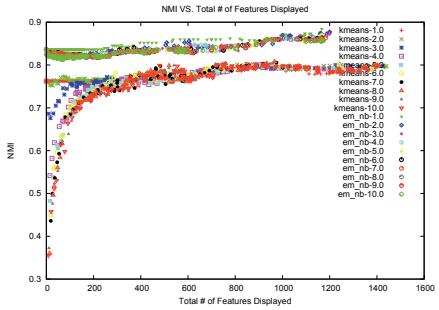
Figure B.14: DCIFS (Algorithm 4) with different underlying algorithms on the same ACM (*D-H-I*) datasets. Legends are all weights between 1 and 10 for both KMeans and *EM-NB*.



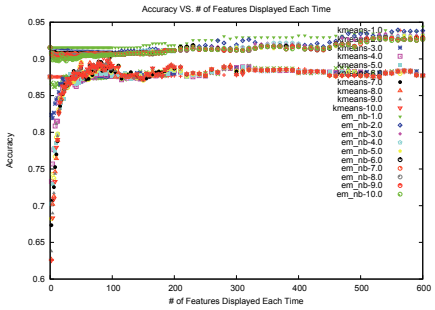
(a) KMeans and *EM-NB* on *3-classic-abstract* dataset: (x axis-f) vs. (y axis-eff-eff)



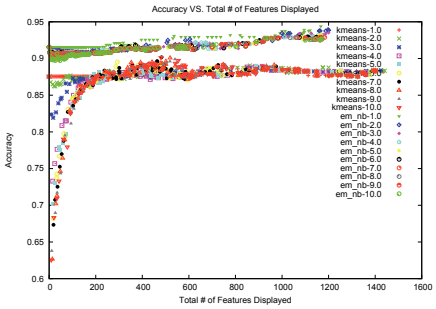
(b) KMeans and *EM-NB* on *3-classic-abstract* dataset: (x axis-f) vs. (y axis-NMI)



(c) KMeans and *EM-NB* on *3-classic-abstract* dataset: (x axis-f_{total}) vs. (y axis-NMI)



(d) KMeans and *EM-NB* on *3-classic-abstract* dataset: (x axis-f) vs. (y axis-*Accuracy*)

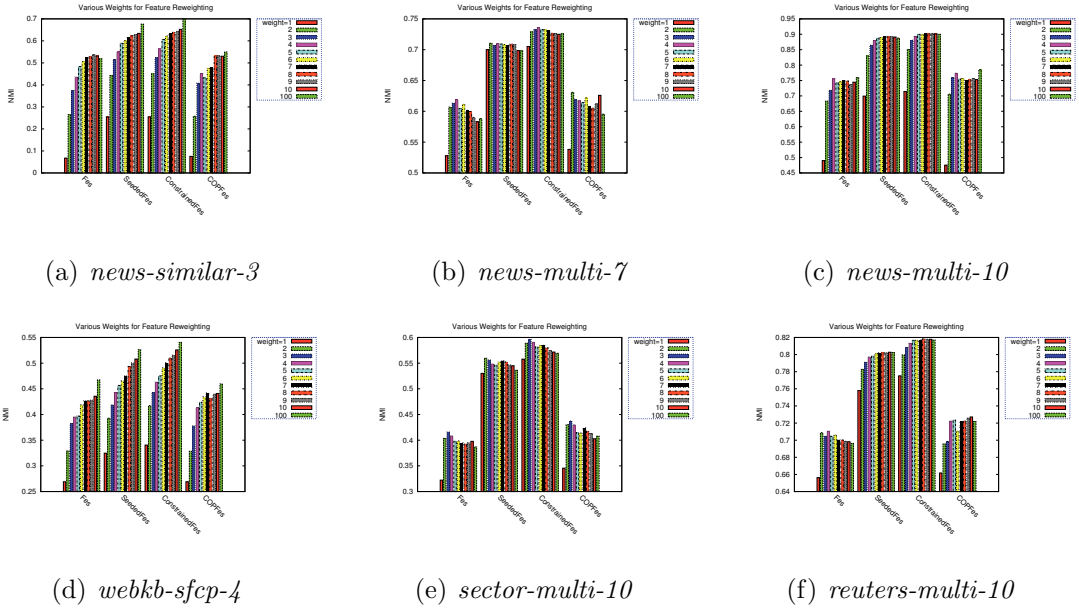


(e) KMeans and *EM-NB* on *3-classic-abstract* dataset: (x axis-f_{total}) vs. (y axis-*Accuracy*)

Figure B.15: DCIFS (Algorithm 4) with different underlying algorithms on the same *3-classic-abstract* dataset. Legends are all weights between 1 and 10 for both KMeans and *EM-NB*.

Appendix C

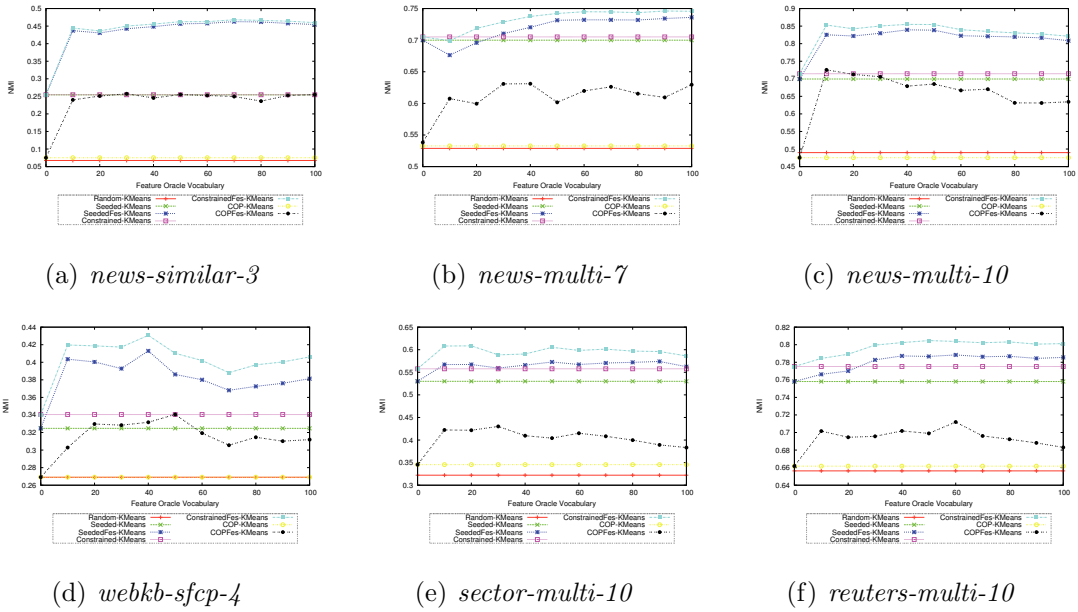
Experimental Results (All Figures) for Chapter 5



(a) *news-similar-3* (b) *news-multi-7* (c) *news-multi-10*

(d) *webkb-sfcp-4* (e) *sector-multi-10* (f) *reuters-multi-10*

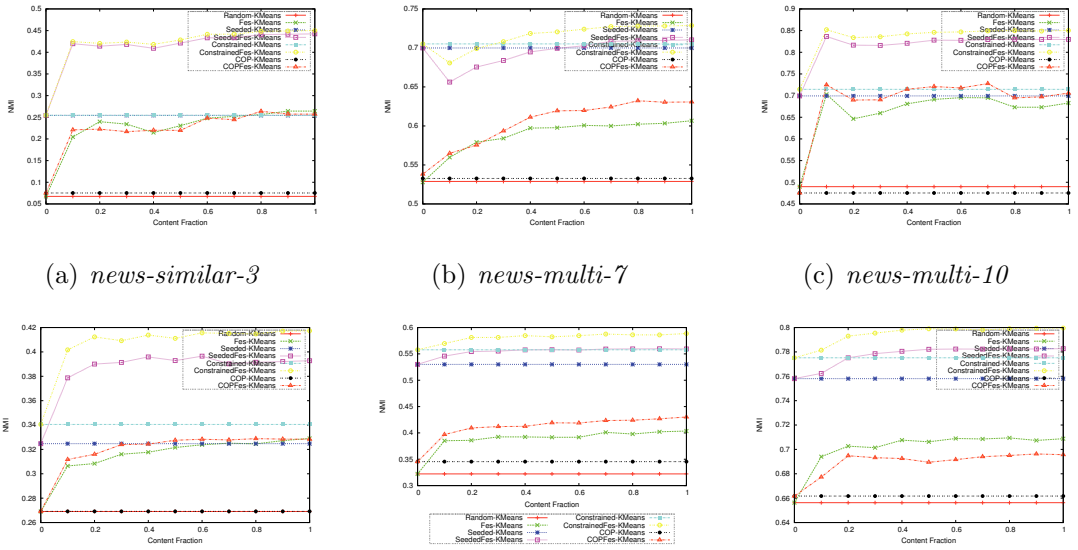
Figure C.1: Feature Reweighting with Different Weights



(a) *news-similar-3* (b) *news-multi-7* (c) *news-multi-10*

(d) *webkb-sfcp-4* (e) *sector-multi-10* (f) *reuters-multi-10*

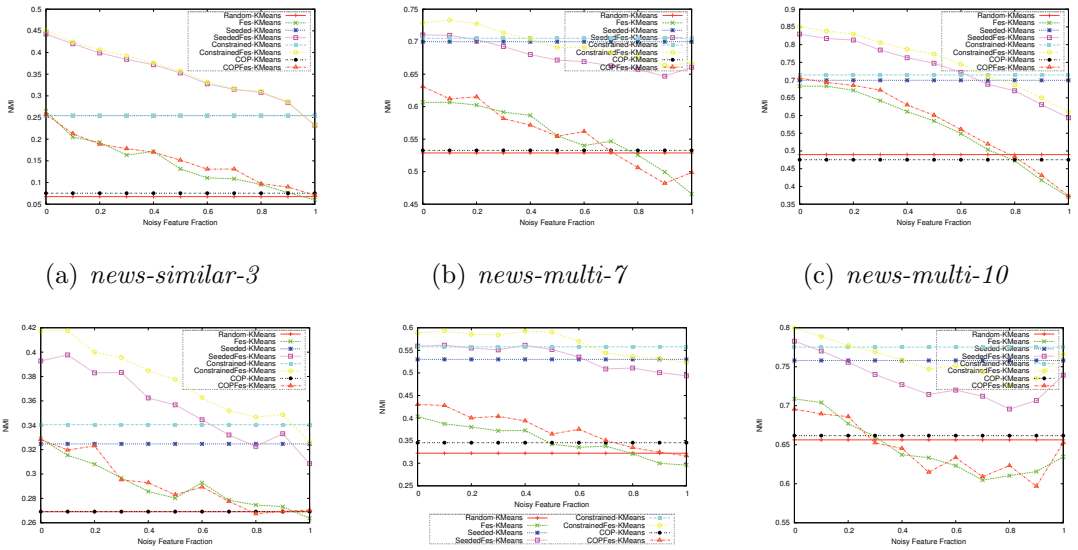
Figure C.2: Performance as a Function of the Size of Feature Vocabulary, i.e., Feature Oracle Capacity



(a) *news-similar-3* (b) *news-multi-7* (c) *news-multi-10*

(d) *webkb-sfcp-4* (e) *sector-multi-10* (f) *reuters-multi-10*

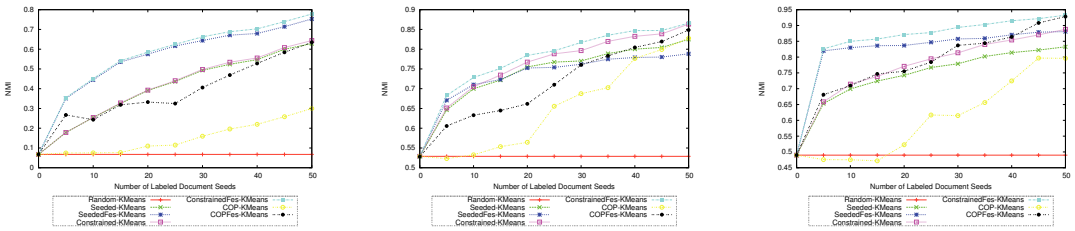
Figure C.3: Different Algorithms Enhanced with Feature Supervision with Varying Content being Read



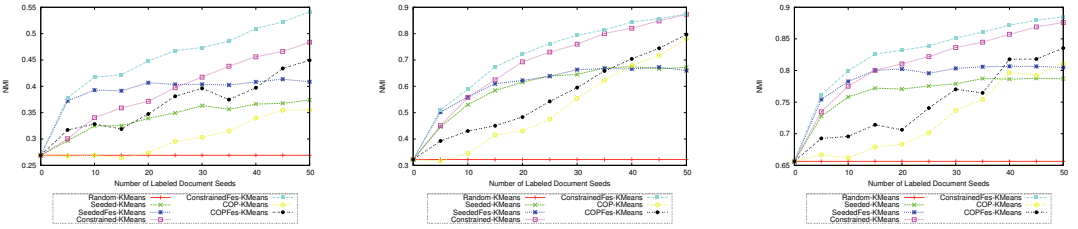
(a) *news-similar-3* (b) *news-multi-7* (c) *news-multi-10*

(d) *webkb-sfcp-4* (e) *sector-multi-10* (f) *reuters-multi-10*

Figure C.4: Different Algorithms Enhanced with Feature Supervision with Varying Noisy Feature Fraction

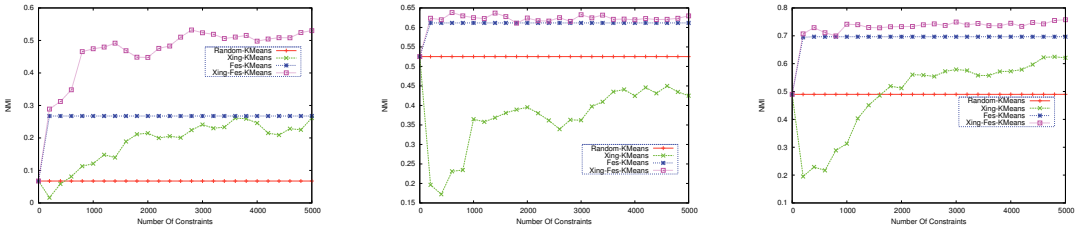


(a) *news-similar-3* (b) *news-multi-7* (c) *news-multi-10*

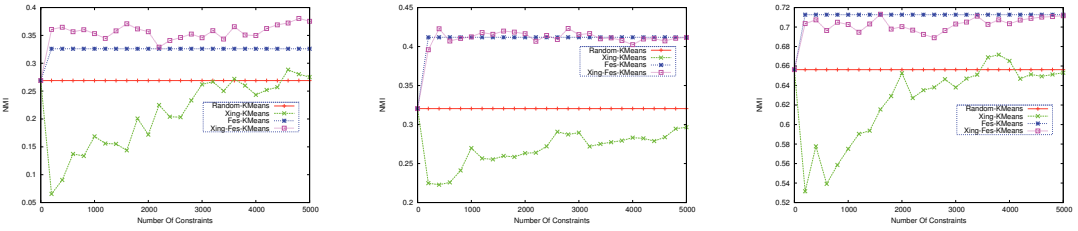


(d) *webkb-sfcp-4* (e) *sector-multi-10* (f) *reuters-multi-10*

Figure C.5: Different Number of Document Seeds (Constraints for COP-KMeans and COPFes-KMeans are Generated from Document Seeds, see Section 5.4.3 for Details)

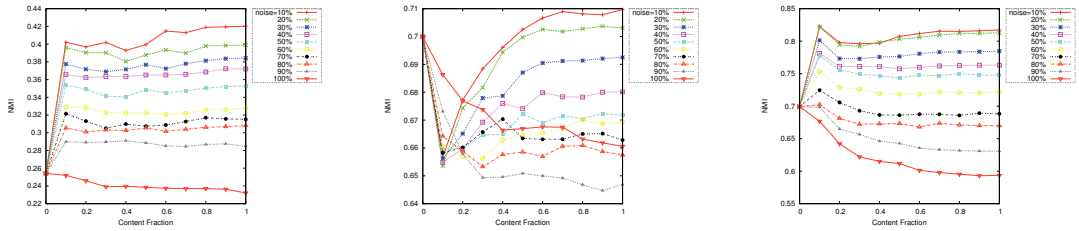


(a) *news-similar-3* (b) *news-multi-7* (c) *news-multi-10*

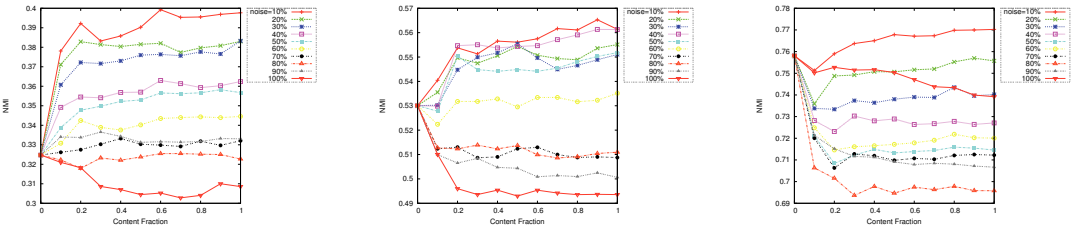


(d) *webkb-sfcp-4* (e) *sector-multi-10* (f) *reuters-multi-10*

Figure C.6: Metric Learning Method and Feature Supervision Method

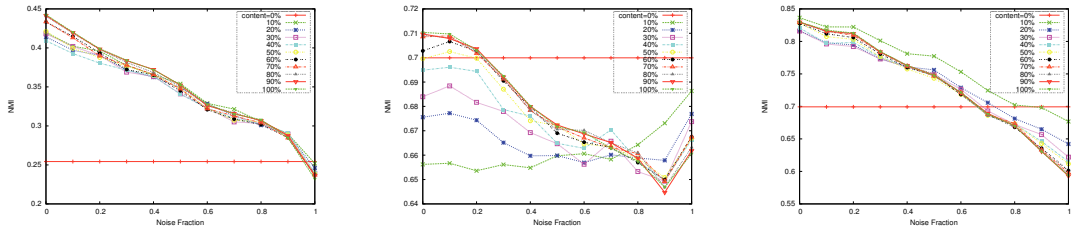


(a) *news-similar-3* (b) *news-multi-7* (c) *news-multi-10*

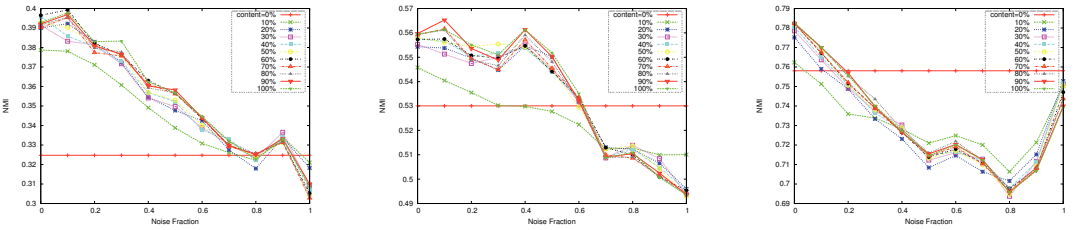


(d) *webkb-sfcp-4* (e) *sector-multi-10* (f) *reuters-multi-10*

Figure C.7: SeededFes-KMeans with Varying Content being Read for Feature Oracle with Different Noisy Feature Level. Each curve represents a Feature Oracle with the Corresponding Level of Noisy Features.



(a) *news-similar-3* (b) *news-multi-7* (c) *news-multi-10*

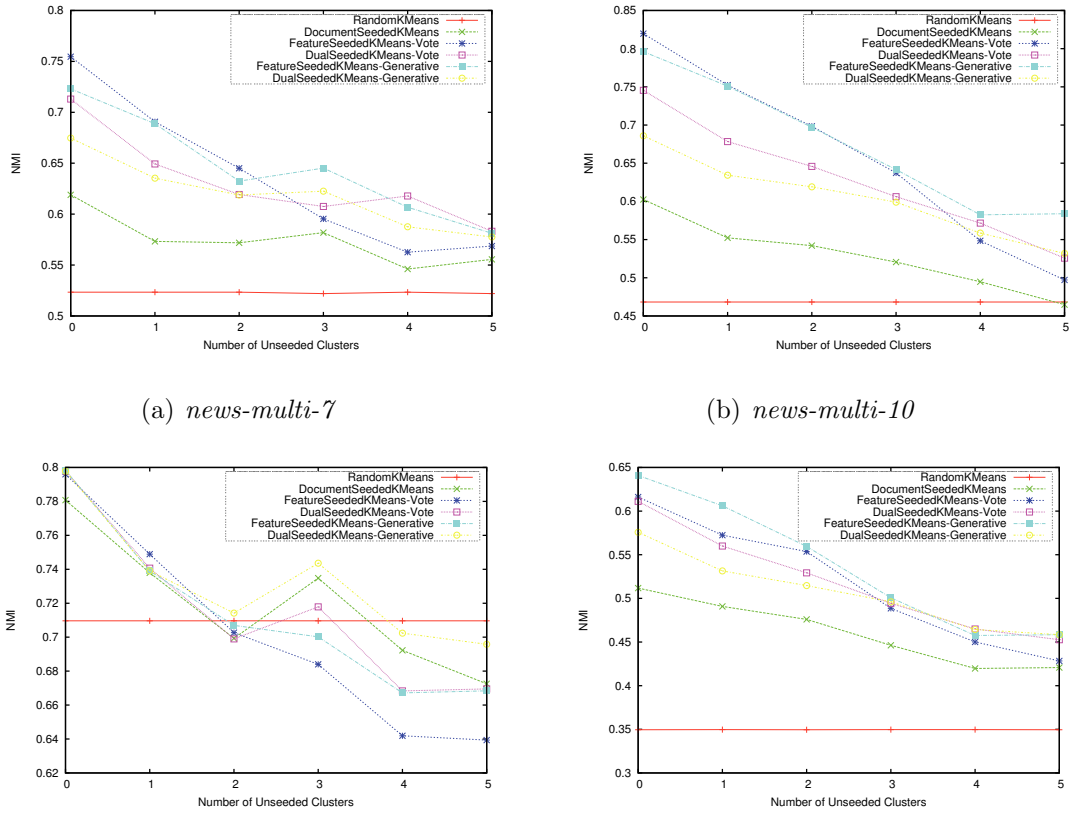


(d) *webkb-sfcp-4* (e) *sector-multi-10* (f) *reuters-multi-10*

Figure C.8: SeededFes-KMeans with Feature Oracle with Different Noisy Feature Level for Varying Content being Read. Each curve represents a certain percentage of content being read.

Appendix D

Experimental Results (All Figures) for Chapter 6



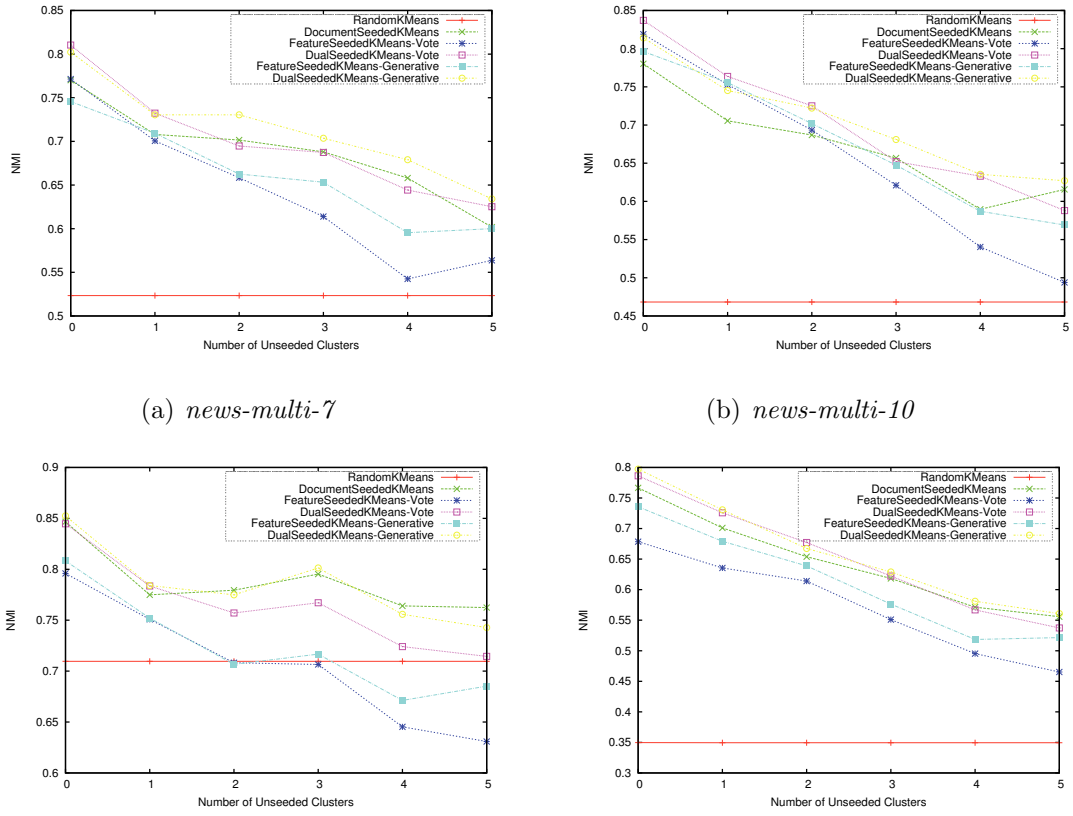
(a) *news-multi-7*

(b) *news-multi-10*

(c) *sector-multi-10*

(d) *reuters-multi-10*

Figure D.1: Performance as a Function of the Number of Unseeded Clusters. 5 Documents Are Labeled for Each Seeded Cluster where FeatureSeeded$K$Means works better than DocumentSeeded$K$Means and DualSeeded$K$Means



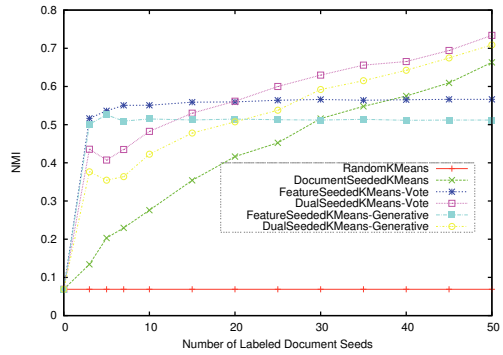
(a) *news-multi-7*

(b) *news-multi-10*

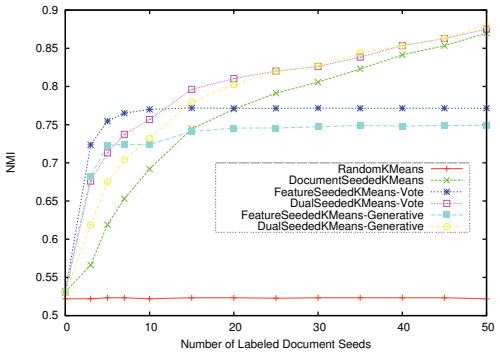
(c) *sector-multi-10*

(d) *reuters-multi-10*

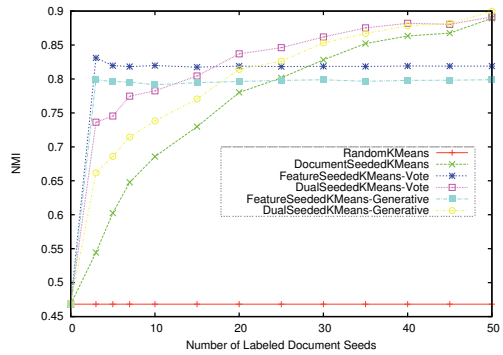
Figure D.2: Performance as a Function of the Number of Unseeded Clusters. 20 Documents Are Labeled for Each Seeded Cluster where DualSeeded$K$Means works better than DocumentSeeded$K$Means and FeatureSeeded$K$Means



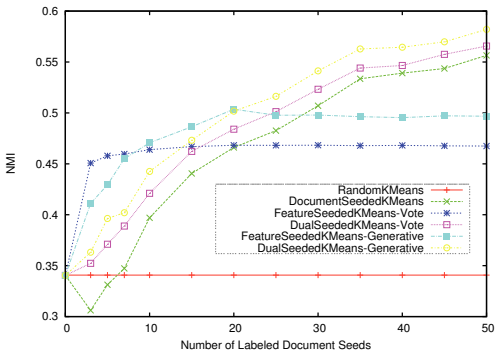
(a) news-similar-3-100



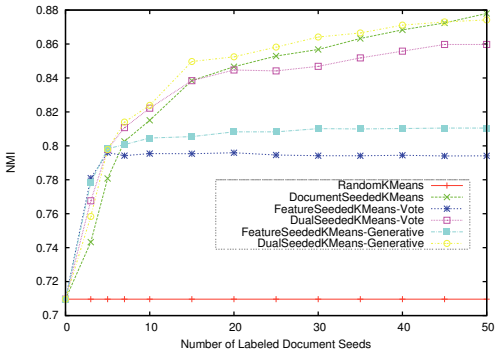
(b) *news-multi-7*



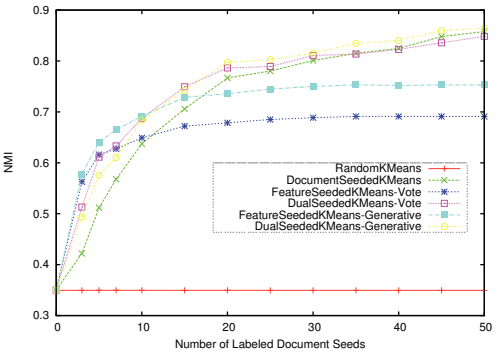
(c) *news-multi-10*



(d) webkb-sfcp-4-250



(e) *sector-multi-10*



(f) *reuters-multi-10*

Figure D.3: Performance as a Function of the Number of Labeled Documents. The more documents labeled, the more features labeled and the better performance. The usefulness of labeled features are more obvious when there are only a few documents labeled, e.g., < 10. In fact, the feature supervision works even better than dual supervision at the beginning of the curves, indicating that feature supervision is more reliable when only few documents are labeled.