A Visual Approach for Interactive Keyterm-Based Clustering

SEYEDNASER NOURASHRAFEDDIN and EHSAN SHERKAT, Dalhousie University
ROSANE MINGHIM, Universidade de São Paulo
EVANGELOS E. MILIOS, Dalhousie University

The keyterm-based approach is arguably intuitive for users to direct text-clustering processes and adapt results to various applications in text analysis. Its way of markedly influencing the results, for instance, by expressing important terms in relevance order, requires little knowledge of the algorithm and has predictable effect, speeding up the task. This article first presents a text-clustering algorithm that can easily be extended into an interactive algorithm. We evaluate its performance against state-of-the-art clustering algorithms in unsupervised mode. Next, we propose three interactive versions of the algorithm based on keyterm labeling, document labeling, and hybrid labeling. We then demonstrate that keyterm labeling is more effective than document labeling in text clustering. Finally, we propose a visual approach to support the keyterm-based version of the algorithm. Visualizations are provided for the whole collection as well as for detailed views of document and cluster relationships. We show the effectiveness and flexibility of our framework, *Vis-Kt*, by presenting typical clustering cases on real text document collections. A user study is also reported that reveals overwhelmingly positive acceptance toward keyterm-based clustering.

CCS Concepts: • **Information systems** → **Clustering**; • **Human-centered computing** → *Visual analytics*;

Additional Key Words and Phrases: Document clustering, keyterm-based clustering, visualization, interactive

ACM Reference format:
Seyednaser Nourashrafeddin, Ehsan Sherkat, Rosane Minghim, and Evangelos E. Milios. 2018. A Visual Approach for Interactive Keyterm-Based Clustering. *ACM Trans. Interact. Intell. Syst.* 8, 1, Article 6 (February 2018), 35 pages.
https://doi.org/10.1145/3181669

1 INTRODUCTION

By grouping similar documents, clustering algorithms provide precious information about topics in text collections. Due to the cost of labeling documents, it has many applications in practice, such as analyzing trends in forensics, media posts, health records, and so on. Clustering has been widely studied in the literature and numerous algorithms with various features have been proposed. Traditionally, no user-effort is targeted and the user has minimum interaction with the clustering process.

The research was funded in part by the Natural Sciences and Engineering Research Council of Canada, the Boeing Company, CNPq and FAPESP (Brazil), and the International Development Research Centre, Ottawa, Canada. The reviewing of this article was managed by special issue associate editors Yu- Ru Lin and Nan Cao.
Authors' addresses: S. Nourashrafeddin, E. Sherkat, and E. E. Milios, Faculty of Computer Science, Dalhousie University, 6050 University Avenue, PO BOX 15000, Halifax, NS B3H 4R2, Canada; emails: {nourashraf, ehsansherkat}@dal.ca, eem@cs.dal.ca; R. Minghim, ICMC - University of São Paulo, Av. Trabalhador 400, 13566-590, São Carlos - SP, Brazil; email: rminghim@icmc.usp.br.

https://doi.org/10.1145/3181669

However, topics discovered by an unsupervised algorithm do not always make sense to users [28]. An unsupervised method is not fully aware of individual preferences and would generate similar results for users with different needs. In most application domains and tasks, it is the user who evaluates the quality of clusterings, because no ground truth is available for most collections. These issues require the involvement of users in the clustering process.

Presenting documents and collecting feedback require an interactive clustering algorithm with easily tuned parameters and a well-designed visualization platform. Due to model complexity, many clustering algorithms either require an extensive parameter tuning phase or are not prone to interactiveness. There has been a trend toward user-interactive clustering in the past few years. Some research work is focused on interactive algorithms [2, 26, 28, 47] and the others utilized visual frameworks to ease interaction [12, 35].

Interaction in text clustering can be categorized into three types: document supervision, keyterm supervision, and hybrid keyterm-document supervision:

(1) Document supervision requires labeling a set of training documents as belonging to certain groups, or providing *must-link* and *cannot-link* constraints [5, 50]. Its main challenges are selecting the initial subset of documents to cover all topics of a collection and also tackling users' mistakes in the process of labeling.
(2) Keyterm-based clustering falls in two sub-categories: keyterm selection and keyterm labeling. In the former, users select keyterms and the clustering process is performed in the feature space formed by the selected keyterms [29, 30]. In the latter, user labels keyterms, and clusterings are obtained based on the relevance of keyterms to specific groups of documents [28, 40]. The main tasks of this approach are extracting relevant keyterms from documents and clusters, presenting them to the user, collecting feedback, and incorporating them in the clustering process.
(3) A hybrid supervision approach includes both document and keyterm supervisions [12].

Compared to document-supervised approach, keyterm-supervised clustering is more intuitive and requires less user-effort. This is mainly because the whole process is controlled by adjusting discriminative terms and reading the contents of documents is mostly unnecessary. This form of clustering has progressed considerably in the past few years [29, 30, 40] and the results are highly effective, fast, and likely to reflect individual preferences.

Visual tools have considerably facilitated text analysis [46] by providing topical insights into text collections, minimizing effort required to collect feedback, and presenting the result of algorithms. For textual data, many interfaces utilize keyterms as the main element of interaction [13, 25, 32]. These interfaces can be grouped into three classes:

(1) Interfaces designed only to explore the obtained clusters [13, 16, 51, 52]. The user has no interaction with the clustering process and her preferences are not considered.
(2) Interfaces designed only to collect users' knowledge and no interaction with the clustering process is performed beyond the initial supervision step [15, 23, 32].
(3) Interfaces designed so the user interacts with the clustering process until satisfactory results are obtained [17–19, 35]. Intermediate clusters are displayed iteratively and the user provides feedback to direct the process.

To the best of our knowledge, no other interactive text-clustering visualization has been proposed to support keyterm labeling. The main purpose of this research is to propose a novel interactive clustering method coupled with a visual framework to ease this form of interaction. With a single-screen visualization of various aspects of the collection, the user can easily move from a low-level view of documents and terms to a high-level view of clusters to discover

the appropriate keyterms required to direct the process. We demonstrate the capability of the proposed visual framework in adequately providing support to form desired groups of documents, without relying on external information such as metadata or labels of any kind. We report two use cases and a user study on real document collections for this purpose.

One might argue that single terms are often ambiguous and labeling documents is more intuitive than labeling terms in text clustering. To address this argument, it is necessary to mention that a topic or a document is never represented by a single term in our framework. The representations are always based on a group of terms, and we argue that it is very intuitive for the users to realize whether a term belongs in a group of terms. Additionally, a group of terms helps the users to discover the target sense of an ambiguous term. Our experiments and the results of the user study reported in this article confirm this argument.

The remainder of the article is organized as follows. Section 2 reviews visualization tools proposed for interactive text clustering. Section 3 introduces the underlying clustering algorithm and its interactive versions. Section 4 reports the experiments conducted to evaluate the performance of the proposed algorithms. Section 5 presents our clustering framework and results of two use cases as well as a user study conducted by using the framework. Section 6 includes discussion, conclusions, and future work. In the following sections, we may use terms "term" and "word," interchangeably.

2 RELATED WORK

In this section, we review tools proposed to visually support interactive document clustering. Due to the highly interactive nature of our framework, works on cluster visualization that are not designed to include the user in the loop are not reviewed here. We do refer to References [23, 52] for reviews on visualization tools that are built specifically to visualize clusterings. We also do not review interactive clustering methods proposed without any visualization or interface.

2.1 Visual Text Clustering Using High-Resolution Display

A workspace to spatially organize text documents into clusters is employed in *LightSPIRE* [19] and in *ForceSPIRE* [18]. Text documents are shown on a large, high-resolution display either as full text or filename-only. The user can open and read documents, search for terms, highlight text, and move documents on the display. She finally places similar documents near each other to form clusters.

Bixplorer [20] is another visual interface proposed to support biclustering, simultaneously clustering of terms and documents. It is based on the above-mentioned *LightSPIRE*'s spatial workspace.

We argue that asking users to read documents and relocate them on a display requires much effort, even for small collections. Besides, the equipment needed to show documents' content on a big screen may not be available for everyone. Working based on relevant terms, on the other hand, allows us to express topics of documents in a more abstract form. A few keyterms can represent hundreds of relevant documents. Our case studies also reveal that showing top keyterms will decrease the need for doing expensive reading of documents. The user can easily infer or convey topics by keyterms.

2.2 Visual Text Clustering Using Semi-Supervision

A visual interface to support semi-supervised clustering is proposed in Reference [17]. The underlying clustering algorithm is based on the *must-link* and *cannot-link* constraints. To obtain these constraints, all objects of a collection are displayed as a two-dimensional (2D) scatter plot. The user moves objects to place similar ones next to and dissimilar ones far from each other.

A similar approach, based on the 2D scatter plot, is proposed in Reference [11] to interactively learn distance functions among data objects. An initial scatter plot is shown to the user based on the Euclidean distance. The user moves objects to adjust their locations. The distances are then updated and the system generates a new scatter plot. This process continues until a satisfying plot is obtained.

Another visual clustering tool is proposed in Reference [10] to actively find seed documents of k-means. Based on a 2D projection, the user selects seed documents of clusters and sets their limits. The limit is the maximum distance of an object to the centroid of the cluster it belongs in. An interesting feature of the tool is that the user can select the projection method and decide which dimensions to be kept. This approach may not be appropriate for document clustering, since a single document does not contain enough information to effectively build a cluster around it [1]. Selecting a few seeds for each cluster may increase the chance of generating good clusterings but it increases the amount of user-effort.

Similar approaches to support semi-supervised clustering are proposed in References [15, 23, 32]. Instead of moving objects, the user can select some objects and label them. The labeled objects are then used as seeds in the clustering process. The user can also specify keyterms of each document cluster in Reference [32]. The keyterms are used to form a new feature space. Since the user must read the documents, these approaches are effective in clustering small collections.

Another type of user supervision is in the form of cluster split and merge requests [3]. Starting from an initial clustering, the algorithm interacts with the user in stages. In each stage, the user provides split and merge requests on some clusters and the algorithm makes the changes.

Based on the following reasons, we argue that an interactive visual framework for text clustering needs more functionalities than relocating objects on a 2D scatter plot, being able to select seed documents, or supporting split and merge requests.

- Examining all data objects and moving them on a 2D plot is time-consuming, especially for large collections. It would be more desirable to move a few objects and the system to learn to re-locate the other objects. This issue is addressed in the visualization proposed in Reference [27] by explicitly asking the user about some unmoved objects.
- Some information about documents should be embedded on the plot to ease identifying similar documents. Moreover, a 2D scatter plot, which is usually generated from pairwise document similarity, does not reveal information about document-term and cluster-term relationships.
- The operation of split or merge cluster is fully automatic and the output clusters might not reflect users' preferences. Moreover, the effects of these operations should be reflected in the model, since some objects might be better moved to new clusters.

2.3 Visual Text Clustering Based on Topic Modeling

Topic modeling [8] is widely used for exploring text collections. ConVisIT is a visual interface developed based on an interactive topic modeling algorithm to support exploring online text conversations [26]. Topic modeling is leveraged to cluster the sentences of conversations and to provide insights about the topics being discussed. Interactions are limited to two operations; splitting a cluster and merging two clusters. The user can split a cluster into two sub-clusters if she finds its topic too general. This operation is fully automatic and ConVisIT does not update the model after changes made by these two operations.

ITM is an interactive tree-based topic modeling framework [28] proposed to easily encode user's feedback in topic modeling. Each topic is represented as a tree in *ITM* while each internal node has a weight (probability) and each leaf node is assigned with a term. Each term of the vocabulary

must appear in at least one leaf node. The framework starts by running *Latent Dirichlet Allocation* (*LDA*) to generate initial topics. Each topic is then shown to the user along with its keyterms. The user then provides feedback by selecting important terms. The collected feedback is fed into *ITM* to update the model by changing the structure of the tree and weights of nodes. The user can interact with the framework until a satisfactory clustering is obtained. In terms of interaction, *Vis-Kt* is very similar to *ITM*, since both framework require only some important terms to specify topics. However, *Vis-Kt* provides more functionalities for this purpose:

- The user can change the number of topics by removing, merging, and splitting clusters or adding new ones in *Vis-Kt*. A thorough analysis on changing the number of topics is required for *ITM*.
- The visualization modules in *Vis-Kt* provide more insights into the collection by representing term-document, term-cluster, and document-cluster relationships. It would be more convenient if *ITM* provides more information about the topics rather than showing a list of terms and associated documents. An interactive topic modeling approach requires a well-designed visual interface similar to one proposed in *Vis-Kt*.

iVisclustering is a visualization tool proposed in Reference [35] to support *LDA*-based clustering algorithm. The fundamental difference between *Vis-Kt* and *iVisclustering* is in term supervision. While *LDA* requires, as the main influential parameter, that users work with term weights, our approach requires only term manipulation (adding, removing, or changing relevance order). In terms of precision, both approaches are comparable, but in terms of intuitiveness, term manipulation requires little knowledge of the clustering process to be effective. The differences are mentioned below in detail:

- The vertical positions of terms in term lists of *Vis-Kt* indicate their relevances to the clusters. Each list holds keyterms of one cluster. The user chooses relevant keyterms and adjusts relevances by changing their positions. In other words, she only needs to specify the topics of interest by forming an ordered set of keyterms. In contrast, she needs to increase or decrease floating point weight values in *iVisclustering*. We argue that adjusting weight values is not trivial for users, who may not have any knowledge about the clustering process, but forming ordered lists of keyterms is much more convenient.
- *Vis-Kt* allows the user to add/remove terms to form term lists or assign a term to multiple lists with different levels of relevance. Clusters in our case are created at will, with a few relevant words sufficing to update the model. These features are not presented in *iVisclustering*.
- Merging or splitting clusters are performed using a folder tree in *iVisclustering*, without feedback of these changes to the model. *Vis-Kt* supports these operations by term manipulation. All changes update the model to reflect user actions. *Vis-Kt* also allows the user to specify the topics of new clusters in case she wants to increase the number of clusters. This feature is not provided in *iVisclustering*.
- Visually, both approaches (*iVisclustering* and Vis-Kt) have some similarities. Their main panel shows a graph-based projection, which displays documents' glyphs and their similarities. All documents of each cluster have the same color. In *iVisclustering*, each document is assigned to one cluster (hard clustering). *Vis-Kt* is able to show soft clusterings too. Our graph representations also allow various degrees of interaction such as creating term clouds from the documents and their neighbors. The representations are also coordinated with other views in *Vis-Kt*. For instance, by clicking or searching for a term, all the documents

that contain the term will be highlighted. These features are not provided in graph projections of *iVisclustering*.

UTOPIAN [12] is an interactive user-driven topic modeling system based on semi-supervised Non-negative Matrix Factorization (*NMF*). Different from *Vis-Kt* and *iVisclustering*, it provides hybrid document-keyterm supervision. The user changes weight values of both documents and keyterms in the model to specify her preferences. Similar to *iVisclustering*, this is the only way to direct the underlying clustering process. We believe that for an end-user who has some background knowledge about the collection but not about machine learning, changing weight values is not trivial. The interface of *UTOPIAN* solely consists of a graph representation with limited functionalities to change weight values.

A visual interface to form keyterm clusters and discover patterns of topics over text sequences is proposed in Reference [53]. The interface displays clusters as time-varying curves. The curves are generated based on the frequency of terms in documents over time. The user's judgments on meaning of clusters is based on the keyterms of curves and curve shape similarities. The weak point of the interface is that the shape similarity is not precisely defined and interpretations are quite subjective. The interface is not directly comparable to our proposal, since it handles text streams over time.

In the context of document clustering, a visual framework that offers the possibility to cluster and examine data set and it is easy to use in terms of influencing the clustering process and tuning results, can go a long way towards improving performance for clustering strategies. We believe that our visual framework can accomplish that by providing a novel approach to efficient user-supervised clustering.

3 PROPOSED DOCUMENT CLUSTERING ALGORITHM

This section first presents a novel text document clustering algorithm. Next, three interactive versions of the algorithm based on different types of user supervision are proposed.

3.1 Lexical Double Clusterer

The clustering algorithm proposed in this work is called Lexical Double Clusterer (*LDC*). The idea behind the algorithm is that before finding document clusters it is better to focus on term clusters and the keyterms that represent topics. The proposed algorithm consists of the following phases:

(1) Clustering terms
(2) Distilling term clusters
(3) Extracting lexical seed documents
(4) Clustering documents

The main steps of *LDC* are shown in Algorithm 1. Fuzzy c-means groups terms into k term clusters. Next, term clusters are distilled by removing general terms. From the distilled term clusters, lexical seed documents are identified and finally used as seeds to cluster documents. The input of *LDC* is a document-term matrix. The matrix is obtained from a text corpus using the Bag of Word (*BoW*) model. Term frequency–inverse document frequency (*tfidf* [1]) is used as feature values to indicate the importance of terms in documents.

(1) **Clustering terms:** fuzzy c-means clusters term vectors (columns of the document-term matrix) into k term clusters. For a matrix with M term vectors $\{t_1, t_2, \ldots, t_M\}$, it generates

[1] https://en.wikipedia.org/wiki/Tf-idf.

ALGORITHM 1: Lexical Double Clustering (*LDC*) [40]

Input: *A document-term matrix obtained based on tfidf, the number of clusters k*
Output: *k document clusters*
1: *Fuzzy c-means generates k term clusters*
2: *Non-discriminative terms are removed from the term clusters using Var-tfidf feature selection method*
3: **for** *each distilled term cluster* **do**
4: *Seed documents are extracted using a greedy approach*
5: *A document centroid is computed over the seed documents*
6: **end for**
7: **for** *each document* **do**
8: *The Cosine distances to the document centroids are computed*
9: *The memberships of the document to document clusters are measured as the inverse of the distances*
10: **end for**
11: *Generate a hard partitioning by assigning each document to the closest document centroid.*
12: [*Generate a soft partitioning by thresholding the Cosine distances.*]

k term clusters $\{TC_p\}_{p=1}^{k}$ such that the following objective function is minimized [7]:

$$F_z = \sum_{i=1}^{M} \sum_{p=1}^{k} u_{ip}^{z} dist^2(t_i, \mu_p), \tag{1}$$

where z is a real number larger than 1, u_{ip} is the degree of membership of t_i in TC_p, and μ_p is the centroid of TC_p. At the end of iterations, each term cluster includes only terms whose memberships are greater than $1/k$. This method of defuzzification results in a soft term clustering. The membership matrix, $U = [u_{ip}]_{M \times k}$, is initialized randomly in unsupervised *LDC*. User-supervised versions of *LDC* initialize the matrix using user's feedback.

Even though fuzzy c-means only favors hyperspherical clusters, it fits very well in our experiments. The main reasons are (1) a term can be related to multiple term clusters (topics) with different levels of relevance; this is the main rationale behind fuzzy clustering, (2) initializing its membership matrix lets users guide the term clusterer and the whole process toward individual preferences.

(2) **Distilling term clusters:** only a few terms in each term cluster are discriminative and the other terms do not represent any specific topic [40]. The *Var-tfidf* feature selection method [33] is used in this phase to remove non-discriminative terms from each term cluster using the following steps:

- First, compute each term's score based on the *Var-tfidf* method:

$$\text{Score}(t_j) \equiv \text{Var-tfidf}(t_j) = \frac{1}{N-1} \sum_{i=1}^{N} (tfidf_{ij} - \text{Mean-tfidf}(t_j))^2, \tag{2}$$

  where N is the number of documents and *Mean-tfidf* is the average of feature values over all documents.
- In each term cluster, discard terms whose scores are smaller than the cluster's mean score. The mean score is the average of scores of terms that exist in a term cluster.

The goal of distilling term clusters by feature selection is to find the subspaces of topics of the document collection. The subspaces are then exploited to extract representative (seed)

documents of each topic. Seed documents are those documents that have higher feature values in the subspaces.

(3) **Extracting lexical seed documents:** given the distilled term clusters, we follow a greedy approach to find seed documents. The idea behind this approach is that the discriminative terms of a term cluster have higher *tfidf* values in a subset of documents. These documents have near-zero feature values for the discriminative terms of other term clusters [40]. We perform the following steps to extract seed documents of each distilled term cluster:
   - A term-centroid is computed using the document-term matrix. The term-centroid is the column average of the term vectors corresponding to the terms included in the term cluster. It is a vector with dimensionality equal to the number of documents.
   - The k-means algorithm with $k = 2$ is then applied on the term-centroid (treated as a one-dimensional space) to cluster its elements into two clusters. One cluster corresponds to the documents with near-zero feature values. The other cluster includes seed documents that have higher feature values.

   It is necessary to mention that a seed document might be linked to more than one distilled term cluster, however it has a *tfidf* value in each term-centroid. We consider these values as the weights of seeds in each cluster.

(4) **Clustering documents:** given the seed documents and their weights, document clustering is performed using the Cosine distance:
   - A document centroid is computed over each term cluster's seed documents:

$$\text{DocCentroid}_p = \frac{\sum_{d_i \in \text{Seeds}(TC_p)} w_i * d_i}{|\text{Seeds}(TC_p)|}, \tag{3}$$

     where d_i is the ith row of the document-term matrix, w_i is the weight, of d_i in the term-centroid of TC_p, and $||$ indicates the cardinality of a set.
   - The distance of each document to the document centroids are computed.

   A hard document partitioning is subsequently obtained by assigning each document to the closest centroid. A soft partitioning can also be obtained by thresholding the Cosine distances.

The time complexity of *LDC* is O($NMIk$), where I is the maximum number of iterations of fuzzy c-means.

3.2 User-Supervised LDC

Besides its performance [reported in Section 4], an interesting feature of *LDC* is that it can be easily adapted for interactive use. We propose three user-supervised versions of *LDC* in this section.

(1) Document-supervised *LDC*: a set of documents is randomly selected from the collection and the user is asked to label them, i.e., assign them to a set of classes. Given the labeled documents, keyterms of each class are extracted and used to initialize the term clusterer of *LDC*, fuzzy c-means. Keyterms are extracted using the χ^2 statistic [22]. It is a supervised feature selection method based on the frequencies of terms in document classes. It outputs a probability score for each pair of term and document class.

(2) Term-supervised *LDC*: a document clustering of the collection is first generated by unsupervised *LDC*. The keyterms of each cluster are extracted using the χ^2 statistic by treating clusters as true classes. Next, the user is asked to label the keyterms. The labeled keyterms are used to initialize fuzzy c-means and re-run *LDC*.

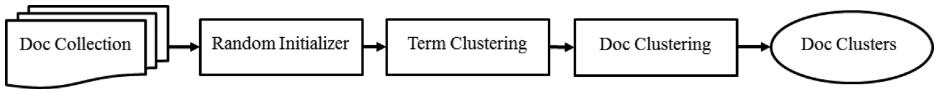


Fig. 1. ***Unsupervised LDC***: the membership matrix of fuzzy c-means is initialized randomly. The obtained term clusters are distilled by removing non-discriminative terms. A greedy approach then identifies seed documents associated with each distilled term cluster. The seed documents are finally used to cluster the documents.

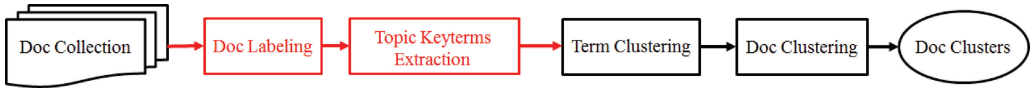


Fig. 2. **Document-supervised *LDC***: a set of documents is selected randomly for labeling. The user assigns them to a set of classes. The keyterms of the classes are then extracted using the χ^2 statistic. Fuzzy c-means is initialized using the keyterms. Term clustering and document clustering are finally performed to generate a document clustering.

(3) Dual-supervised *LDC*: it is the combination of the Document and Term supervised algorithms. Document-supervised *LDC* generates a document clustering. Term-supervised *LDC* is then run given the document clustering.

The block structure of unsupervised *LDC* is depicted in Figure 1. No user supervision is involved and fuzzy c-means is initialized randomly. We use similar block structure to represent the user-supervised versions of *LDC*.

3.2.1 Document-Supervised LDC. Document-supervised *LDC* is quite similar to semi-supervised clustering algorithms [4, 5]. Before starting the clustering process, a set of documents is selected from the collection and the user assigns them to a set of classes. The labeled documents are then used as seeds to initialize the clustering algorithm.

Labeled documents cannot be used directly in *LDC*, since term clustering precedes document clustering as shown in Figure 1. For this purpose, we extract the keyterms of the document classes using the χ^2 statistic. For each class, we compute and sort the χ^2 values in descending order. The corresponding top f terms are then considered as the keyterms of each class. The keyterms are used to initialize fuzzy c-means as shown in Figure 2. The membership of terms in term clusters are stored in the membership matrix of fuzzy c-means. It is initialized randomly in unsupervised *LDC* such that the sum of memberships of each term in term clusters equals to one.

In Document-supervised *LDC*, the f keyterms extracted from the labeled documents are used to initialize the matrix in the following way. For all terms except the keyterms, the matrix is initialized randomly. For the keyterms, their corresponding class entries are set to one. For instance, the membership values of term t_i, considered as a keyterm of the second class out of five classes, is set as $u_i = [0 \quad 1 \quad 0 \quad 0 \quad 0]$. In case of shared keyterms, multiple class entries are set to one.

The user of Document-supervised *LDC* should read at least the title or a few lines of documents. There is no guarantee that the training documents cover all topics of the collection. The interaction is limited to the labeling phase and no interaction with the clustering process is available ever after. The main steps of Document-supervised *LDC* are shown in Algorithm 2.

3.2.2 Term-Supervised LDC. The supervision is in the form of term labeling in Term-supervised *LDC*. Term labeling starts after a hard document partitioning is generated by unsupervised *LDC* (Figure 3). Treating document clusters as true classes, the χ^2 statistic is used to extract keyterms

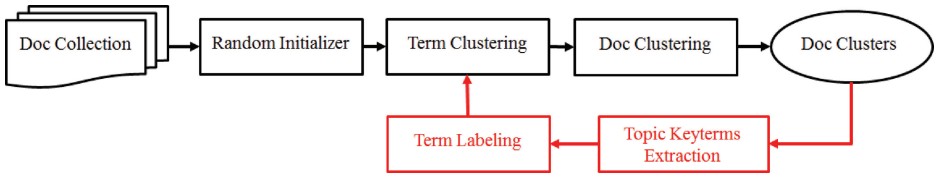


Fig. 3. **Term-supervised *LDC***: unsupervised *LDC*, Algorithm 1, generates a document clustering. The keyterms of the current document clusters are then extracted using the χ^2 statistic. The user performs term labeling on the keyterms. The labeled keyterms are subsequently used to initialize fuzzy c-means for the next round.

ALGORITHM 2: Document-supervised *LDC*

Input: *A document-term matrix obtained based on tfidf, the number of clusters k*
Output: *k document clusters*
1: *Randomly select* $k \times f$ *documents for labeling (training set)*
2: *Ask the user to assign them to classes*
3: *Extract the* f *keyterms of each class using the* χ^2 *statistic*
4: *Initialize the term clusterer, fuzzy c-means, using the extracted keyterms*
5: *Obtain k document clusters using LDC, Algorithm 1.*

ALGORITHM 3: Term-supervised *LDC*

Input: *A document-term matrix obtained based on tfidf, the number of clusters k*
Output: *k document clusters*
1: *Obtain initial k document clusters using unsupervised LDC, Algorithm 1*
2: **repeat**
3: **for** *each document cluster* **do**
4: *Obtain* f *keyterms based on* χ^2
5: **end for**
6: *Perform term labeling*
7: *Use the labeled keyterms to initialize fuzzy c-means and obtain k document clusters*
8: **until** *the user chooses to terminate*

of clusters. The keyterms are then labeled by the user. Fuzzy c-means is initialized by using the labeled keyterms as explained in Section 3.2.1.

The user of Term-supervised *LDC* can continue the clustering process until her desired clustering is obtained. She should have enough background knowledge to place keyterms in meaningful groups to convey topics of interest. The main steps of Term-supervised *LDC* is shown in Algorithm 3.

3.2.3 Dual-Supervised LDC. A combination of document and term supervisions is used in Dual-supervised *LDC*. Document-supervised LDC, Algorithm 2, generates a document clustering based on the labeled documents provided by the user. Term-supervised *LDC*, Algorithm 3, is then run given the current clustering. The block structure of Dual-supervised *LDC* is shown in Figure 4.

The user of this algorithm can continue to interact until satisfactory results are obtained. The user effort includes the user effort of term labeling and document labeling. The main steps of Dual-supervised *LDC* is shown in Algorithm 4.

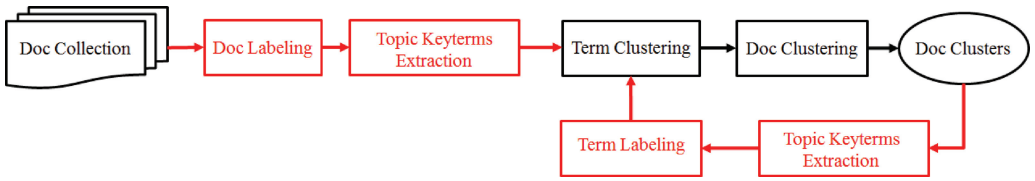


Fig. 4. **Dual-supervised *LDC***: document-supervised *LDC*, Algorithm 2, generates a document clustering. The keyterms of the current document clusters are then extracted. The user performs term labeling on the keyterms. The labeled keyterms are subsequently used to initialize fuzzy *c*-means for the next round.

ALGORITHM 4: Dual-supervised *LDC*

Input: *A document-term matrix obtained based on tfidf, the number of clusters k*
Output: *k document clusters*
1: *Generate a document clustering using Document-supervised LDC, Algorithm 2*
2: **repeat**
3: *Given the current clustering, run Term-supervised LDC, Algorithm 3*
4: **until** *the user chooses to terminate*

4 LDC PERFORMANCE EVALUATION

This section reports the results of experiments conducted to evaluate unsupervised *LDC* and its user-supervised versions. First, unsupervised *LDC* is compared with *LDA* and *NMF* [6] in unsupervised mode. Next, we evaluate and compare the performance of the user-supervised algorithms using simulated users.

4.1 Datasets

To provide a fair evaluation, different types of text collections are used in the experiments of this article (Table 1). Seven collections include news articles, one collection includes web pages, one dataset includes scientific abstracts, and one collection includes text messages:

- The 20newsgroups dataset[2] consists of approximately 20,000 news articles grouped into 20 topics. We employed documents of 9 topics to generate *NewsGroup9*. Each topic contains 80 documents selected randomly. The topics are "comp.os.ms-windows.misc," "comp.sys.mac.hardware," "misc.forsale," "rec.autos," "rec.motorcycles," "talk.politics.mideast," "sci.crypt," "sci.med," and "alt.atheism." *NewsRelated* [31] includes all articles in three related topics of "talk-politics-misc," "talk-politics-guns," and "talk-politics-mideast."
- The documents in the Reuters-21578 dataset[3] appeared on the Reuters newswire in 1987 and labeled manually. *Reuters8* consists of articles in eight categories including "acq," "crude," "earn," "grain," "interest," "money-fx," "ship," and "trade."
- *SMART* data repository[4] contains abstracts of scientific articles about medical, information retrieval, aerodynamics, and computing algorithms. *Classic-4* includes all the abstracts of this repository.
- *WebKB* consists of webpages that are collected from websites of four computer science departments in *Cornell*, *Texas*, *Washington*, and *Wisconsin* universities. The collection is created in the World Wide Knowledge Base (Web-Kb) project in the *CMU* text learning

[2] http://qwone.com/~jason/20Newsgroups/.
[3] http://www.daviddlewis.com/resources/testcollections/reuters21578/.
[4] ftp://ftp.cs.cornell.edu/pub/smart/.

Table 1. Summary of the Text Datasets Used in this Research Work

Dataset Name	No. of Classes	No. of Documents	Description
20newsgroups	20	18,821	All news articles in 20newsgroups after removing duplicates
NewsGroup9	9	720	Subset of news articles of 20newsgroups dataset
NewsRelated	3	2,624	Subset of news articles of 20newsgroups dataset
Reuters8	8	7,674	Articles of Reuters-21578 in eight categories
Classic-4	4	7,095	Scientific abstracts in four categories
WebKB	4	4,168	Webpages of computer science departments
SMS	2	5,479	A public set of text messages
BBC Sport	5	737	BBC Sport news articles collected in 2004-2005
News 2006	—	1,747	News feeds collected in 2006 from Associated Press, CNN, and Reuters
NewsSeparate	13	381	A subset of News 2006 dataset categorized into 13 classes manually

group.[5] The webpages are manually labeled into seven categories: student, faculty, staff, departments, course, project, and other. We selected four categories of student, faculty, course, and project that have more documents than the other categories.

- *SMS Spam Collection* consists of a public set of text messages labeled as *spam* or *non-spam*. The collection is used in the experiments of Reference [14] and is publicly available.[6]
- *BBC Sport* [24] includes news articles collected from the BBC website.[7] It consists of articles about "athletics," "cricket," "football," "rugby," and "tennis."
- *News 2006* [44] is generated by collecting news feeds from three news agencies (*Associated Press*, *CNN*, and *Reuters*) for 48h from April 5 to 7 of 2006. *NewsSeparate* is a subset of *News 2006* and is manually categorized into 13 classes.

4.2 Unsupervised LDC Evaluation

This section reports the experiments conducted to compare *LDC*, *LDA*, and *NMF* in unsupervised mode. Nine datasets of *20newsgroups*, *NewsGroup9*, *NewsSeparate*, *NewsRelated*, *BBC Sport*, *Reuters8*, *Classic4*, *WebKB*, and *SMS* have class labels in this article (Table 1). Given the class labels, the comparison is performed based on the quality of clusterings generated by these algorithms. Two quality measures of *Normalized Mutual Information*(*NMI*) [37] and *F1* score [34] are employed for this purpose.

We followed the preprocessing steps used in [40] for both *LDC* and *NMF*. The preprocessing includes stop-word[8] removal, porter stemming [45], and a dimensionality reduction phase based on an unsupervised feature selection method [33]. Besides, each feature (term) is centered at zero and

[5] http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/.
[6] http://www.dt.fee.unicamp.br/ tiago/smsspamcollection/.
[7] http://www.bbc.com/sport.
[8] http://www.nltk.org/.

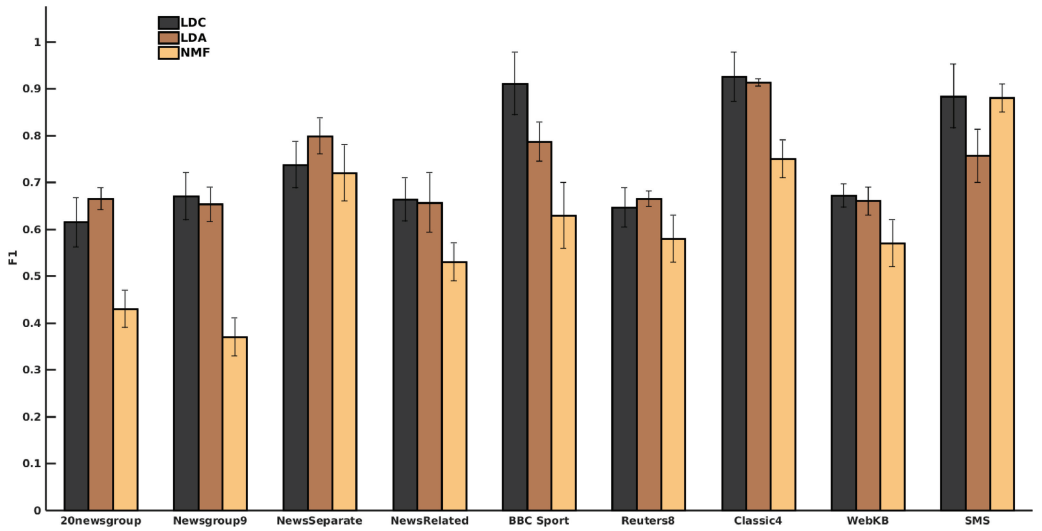


Fig. 5. The average quality of clusterings (based on F1 score) obtained from unsupervised *LDC*, *LDA*, and *NMF* in 50 runs. Based on the paired-sample t-test with $p \leq 0.05$, both *LDC* and *LDA* significantly outperformed *NMF* on *20newsgroups*, *NewsGroup9*, *NewsRelated*, *BBC Sport*, *Classic4*, and *WebKB*. *LDC* significantly outperformed both *LDA* and *NMF* on *BBC Sport*.

normalized with its standard deviation for *NMF*. As a preprocessing step, stop-words are removed for *LDA*.

We ran the algorithms 50 times on each dataset. For *LDA*, we used a C++ implementation[9] and the number of iterations is set to 10,000. For *NMF*, we used the *Matlab* implementation[10] and the number of iterations is set to 10,000. For *LDC*, the maximum number of iterations of fuzzy *c*-means is set to 50. The number of clusters is set to the number of classes in these experiments. The average *NMI*s and *F1*s of these 50 runs are shown in diagrams of Figures 5 and 6, respectively.

From the figures, we note that both *LDC* and *LDA* either significantly outperform *NMF* or the results are similar. *LDC* significantly outperforms *NMF* on six datasets of *20newsgroups*, *NewsGroup9*, *NewsRelated*, *BBC Sport*, *Classic4*, and *WebKB*. On two datasets of *BBC Sport* and *SMS*, *LDC* significantly outperforms *LDA* based on both *F1* and *NMI*.

These experiments demonstrate the effectiveness of *LDC* in generating clusterings similar to or even better than *LDA* and NMF's results. This observation is consistent with the results reported in Reference [40]. *LDC* significantly outperformed *NMF* on the majority of datasets in these experiments.

A version of *LDC* is compared with subspace-based text-clustering algorithms, and double and co-clustering methods. We have demonstrated that the algorithm can generate comparable results or outperform the competitors on real text datasets in References [40, 41]. To save space, we do not report those experiments in this article.

4.3 User-Supervised LDC Evaluation

To simulate user interactions and compare the user-supervised algorithms (proposed in Section 3.2), we introduce two supervision oracles in this section. Although the user profiles in these

[9] http://gibbslda.sourceforge.net/.
[10] http://www.mathworks.com/help/stats/nnmf.html.

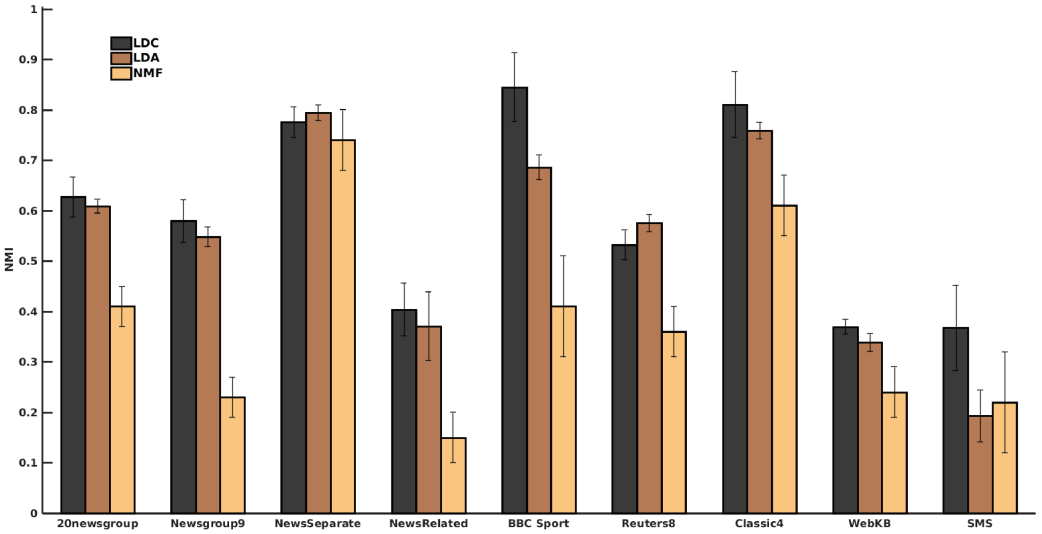


Fig. 6. The average quality of clusterings (based on NMI) obtained from unsupervised *LDC*, *LDA*, and *NMF* in 50 runs. Based on the paired-sample t-test with $p \leq 0.05$, both *LDC* and *LDA* significantly outperformed *NMF* on *20newsgroups*, *NewsGroup9*, *NewsRelated*, *BBC Sport*, *Reuters8*, *Classic4*, and *WebKB*. *LDC* significantly outperformed both *LDA* and *NMF* on *BBC Sport* and *WebKB*. These results are quite consistent with the diagrams of Figure 5.

oracles may be different from the real users' interactions, we can provide a comprehensive comparison among the algorithms with different parameter settings. Both oracles are built from class label of documents:

(1) Document Labeling Oracle: it knows the class label of documents.
(2) Term Labeling Oracle: it knows the class label of terms. Term labels are assigned using the document class labels and χ^2 statistic.

4.3.1 Document Labeling Oracle. The input of this oracle is a set of documents and the output is a set of labeled documents. This is analogous to the case that a real user labels the documents perfectly. She puts documents of the same class in the same group. However, users make mistakes in labeling documents, especially if the topics are similar. To include these mistakes in simulated user interactions, the Document Labeling Oracle has a parameter, P_{exp}, which indicates the degree of user's expertness. $P_{exp} = 1$ corresponds to the expert user who never makes any mistakes, and $P_{exp} = 0$ corresponds to no supervision, unsupervised *LDC*. Any value between zero and one indicates the level of user's expertness.

Based on this parameter, the simulated user either assigns a label to a document correctly or she says, "*I do not know.*" In the latter case, she might remove the document from the training set or assign it to a cluster randomly. The probability of removal is set to 0.5 and the probability of random assignment to a class is set to $0.5/k$. The main steps of the Document Labeling Oracle are shown in Algorithm 5.

4.3.2 Term Labeling Oracle. The inputs of this oracle are k term lists and the outputs are the lists after term labeling. Each list contains the f keyterms of a document cluster extracted by the χ^2 statistic on the current document clustering. This oracle knows the term labels, which are assigned based on the document class labels using the following process. Given the true class of documents,

ALGORITHM 5: Document Labeling Oracle

Input: *a set of (training) documents, the number of clusters k*
Output: *the documents with labels*

for *each document* **do**
  if *Rand(0,1]* $< P_{exp}$ **then**
    Assign the document to the true class
  else if *Rand[0,1]* < 0.5 **then**
    Remove the document from the training set
  else
    Generate a random number between 1 and k
    Assign the document to the random class
  end if
end for

ALGORITHM 6: Assigning Class Labels to Terms

Input: *A document collection, document true class labels*
Output: *Term class labels*

Compute the χ^2 *values for terms and document classes*
for *each term* t_i **do**
  Assign t_i *to the class with the largest* χ^2 *value:*
    $\text{label}(t_i) \leftarrow \text{argmax}_c\ \chi^2(t_i, c)$
end for

ALGORITHM 7: Term Labeling Oracle

Input: *k term lists each includes f keyterms of a document cluster based on* χ^2
Output: *k manipulated term lists after term labeling*

for *each term list* **do**
  for *each term in the term list* **do**
    if *Rand(0,1]* $< P_{exp}$ **then**
      Assign the term to the true list
    else if *Rand[0,1]* < 0.5 **then**
      Remove the term from the term list
    else
      Generate a random number between 1 and k
      Assign the term to the random term list
    end if
  end for
end for

the χ^2 statistic is computed and each term is assigned to the class with the largest χ^2 value. The main steps of this process are shown in Algorithm 6.

Since users may make mistakes in term labeling, the oracle has a parameter, P_{exp}, which indicates the degree of user's expertness. Based on this parameter, the simulated user either assigns a term to a term list correctly or she says, "*I do not know.*" In the latter case, she might remove the term from a list or assign it to a list randomly. The probability of removal is set to 0.5 and the probability of random assignment to a list is set to $0.5/k$. The main steps of this oracle are shown in Algorithm 7.

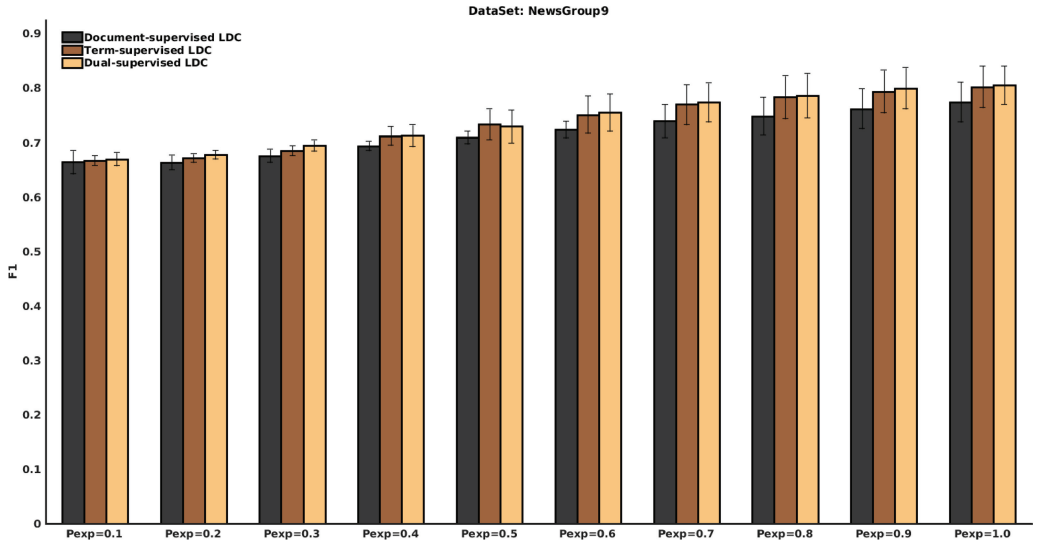


Fig. 7. The average quality of clusterings (*F1* measure) obtained from Term-supervised *LDC*, Document-supervised *LDC*, and Dual-supervised *LDC* based on simulated user interactions. The average is computed over different values of $f = \{1, 2, \ldots, 10\}$ per each level of expertness. Term-supervised *LDC* and Dual-supervised *LDC* outperform Document-supervised *LDC* especially when the user's expertness is $P_{exp} > 0.5$. The clusterings obtained by Term-supervised *LDC* and Dual-supervised *LDC* are quite similar.

4.3.3 Term-Supervised LDC vs. Document-Supervised LDC. To show the benefit of term labeling over document labeling, we compare Term-supervised *LDC*, Algorithm 3, to Document-supervised *LDC*, Algorithm 2. Term-supervised *LDC* calls Term Labeling Oracle, Algorithm 7, to label terms. Similarly, Document-supervised *LDC* calls Document Labeling Oracle, Algorithm 5, for document labeling.

We run these algorithms 50 times for each degree of expertness $P_{exp} = \{0.1, 0.2, \ldots, 1.0\}$ and the number of queries per class submitted to oracles, $f = \{1, 2, \ldots, 10\}$. The total number of queries asked from each oracle is $k \times f$ for each dataset. To provide a fair comparison, one round of interaction is performed for Term-supervised *LDC*.

Given the class labels of documents, the comparison is done based on the quality of clusterings obtained. The average *F1*s and *NMI*s of these 50 runs over different values of f and P_{exp} are depicted in Figures 7 to 10. After analyzing the results of these experiments, we observed that:

- Term-supervised *LDC* and Document-supervised *LDC* can outperform unsupervised *LDC* if the user's expertness is $P_{exp} > 0.5$.
- When the user's expertness is $P_{exp} > 0.5$, term labeling is more effective than document labeling in improving the quality of document clusters.
- The quality of clusters fluctuates when $P_{exp} \leq 0.5$. This implies that the user should have enough background knowledge about the collection or interactions may deteriorate the results.

Considering the above-mentioned observations and also the fact that document labeling requires more time than term labeling, due to the need of reading contents of documents, we can conclude that term labeling is more effective than document labeling in these experiments.

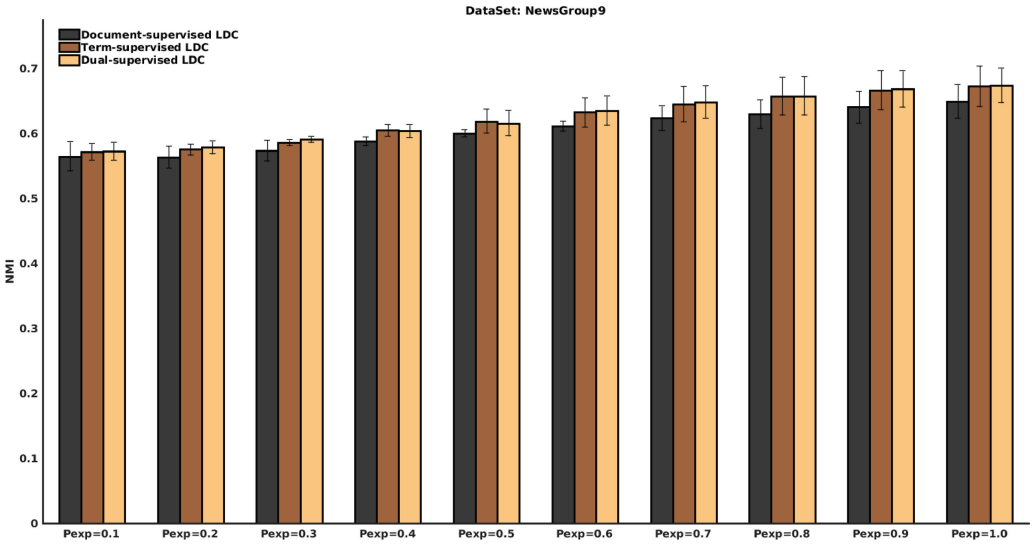


Fig. 8. The average quality of clusterings (*NMI* measure) obtained from Term-supervised *LDC*, Document-supervised *LDC*, and Dual-supervised *LDC* based on simulated user interactions. The average is computed over different values of $f = \{1, 2, \ldots, 10\}$ per each level of expertness. Term-supervised *LDC* and Dual-supervised *LDC* outperform Document-supervised *LDC* especially when the user's expertness is $P_{exp} > 0.5$.

4.3.4 Term-Supervised LDC vs. Dual-Supervised LDC. The goal of experiments of this section is to find out whether adding document labeling to Term-supervised *LDC* would further improve the quality of clusterings. For this aim, we compare Term-supervised *LDC*, Algorithm 3, to Dual-supervised *LDC*, Algorithm 4. Term-supervised *LDC* calls Term Labeling Oracle, Algorithm 7, to label terms. Similarly, Dual-supervised *LDC* employs both oracles for term and document labeling.

We follow the same settings of Section 4.3.3 to conduct the experiments of this section. The outputs are depicted in Figures 7 to 10. After analyzing the results, we observed that:

- The combination of document labeling and term labeling results in clusterings that are quite similar to the outputs of term labeling. This is more evident when the user's expertness is $P_{exp} > 0.5$.
- Dual-supervised *LDC* needs more time than Term-supervised *LDC* due to the extra document labeling.

Based on the experiments of this section, we can conclude that document labeling cannot improve the performance of Term-supervised *LDC* in spite of the initial document labeling phase. So, initial phase of document labeling is unnecessary for *LDC*.

4.3.5 Minimum User-effort to Achieve Significant Improvement. A well-designed clustering framework should be able to elicit individual preferences with minimum effort. On the other hand, one expects that interactions would result in clusterings with similar or better quality than the results of unsupervised algorithms. The goal of experiments of this section is to analyze whether term labeling can improve the quality of clusterings significantly and also how much user-effort is needed for this aim. To answer this concern, we compare Term-supervised *LDC* with unsupervised *LDC* and *LDA* with different parameter settings in terms of the number of queries submitted to oracles, $f = \{1, 2, \ldots, 10\}$, and the user's expertness, $P_{exp} = \{0.1, 0.2, \ldots, 1.0\}$. Each algorithm

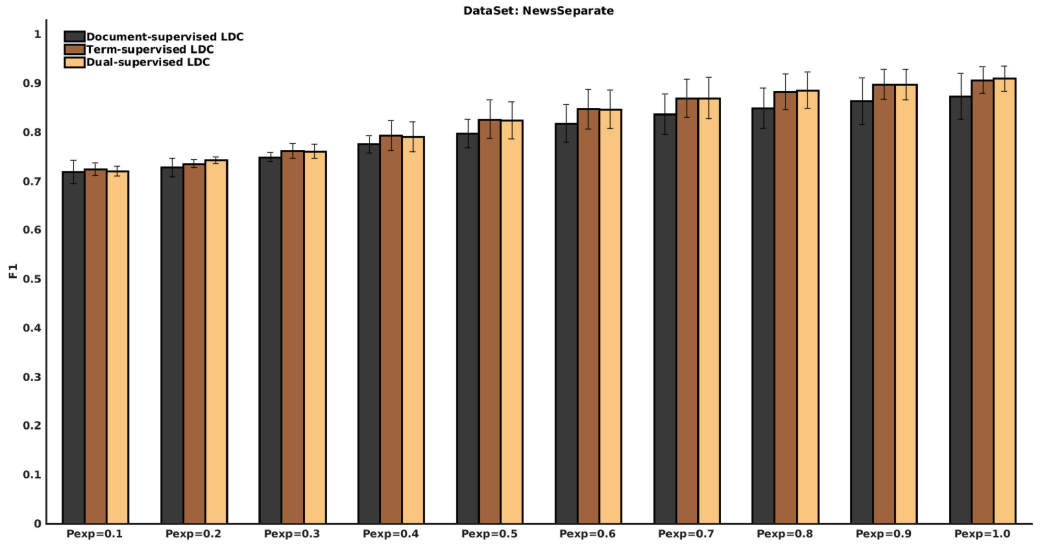


Fig. 9. The average quality of clusterings (*F1* measure) obtained from Term-supervised *LDC*, Document-supervised *LDC*, and Dual-supervised *LDC* based on simulated user interactions. The average is computed over different values of $f = \{1, 2, \ldots, 10\}$ per each level of expertness. Document-supervised *LDC* could not outperform Term-supervised *LDC* and Dual-supervised *LDC*.

is run 50 times. The minimum amounts of interaction required such that Term-supervised *LDC* significantly outperforms unsupervised *LDC* and *LDA* are reported in Table 2. The significant improvements are achieved based on both *F1* and *NMI* measures.

After analyzing the minimum interactions, we note that

- For most datasets, Term-supervised *LDC* can significantly outperform unsupervised *LDC* and *LDA* with a few interactions and high user's expertness. As the expertness decreases, the number of queries increases to achieve significant improvement.
- For two datasets of *SMS* and *BBC Sport*, no significant improvement over unsupervised *LDC* achieved. We believe that unsupervised *LDC* already achieves the highest results on these datasets and no further improvement is possible. Unsupervised *LDC* significantly outperforms *LDA* on these datasets in unsupervised mode.
- For *WebKB*, the minimum amount of effort is quite high in comparison to the other datasets. This is because of the large number of common terms among the topics of this dataset. The web pages are related to more than one topic basically. For instance, it is quite common to have a web page about a computer science course with mentions of faculty, students, and projects.

Based on the experiments conducted on *LDC* and its user-supervised versions, we can conclude that term labeling performs better than document labeling in increasing the quality of clusterings. With a comparable number of simulated user interactions, term labeling was also more effective than term selection (to form a new feature space) in the experiments reported in Reference [43].

In the next section, we propose a visual framework, *Vis-Kt*, to support Term-supervised *LDC* in interaction with real users. We report two use cases on clustering real data to demonstrate the efficacy and flexibility of the framework especially in exploring collections and changing the number and topics of clusters.

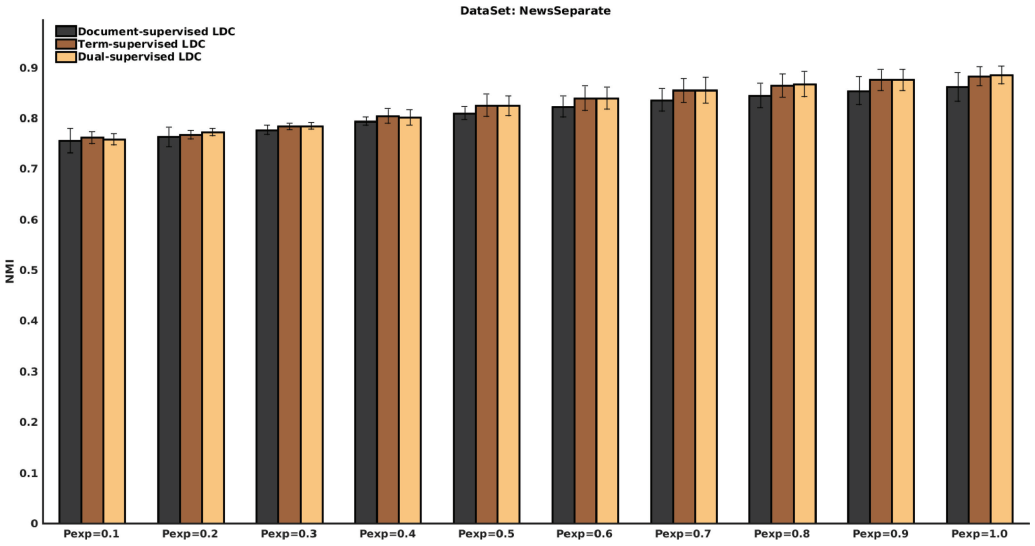


Fig. 10. The average quality of clusterings (*NMI* measure) obtained from Term-supervised *LDC*, Document-supervised *LDC*, and Dual-supervised *LDC* based on simulated user interactions. The average is computed over different values of $f = \{1, 2, \ldots, 10\}$ per each level of expertness. The clusterings of Term-supervised *LDC* and Dual-supervised *LDC* are quite similar.

To allow users to assign a term to multiple clusters with different levels of relevance, we have modified the fuzzy *c*-means initialization of Term-supervised *LDC* in *Vis-Kt*. After term labeling, only the top term of each term list gets a value of one in the membership matrix. The membership of the other terms linearly decreases (based on the number of terms on a list) from 1 to 0.5. We have added this feature, since some topics may share common terms with different levels of relevance. Other than the case of having a term on multiple lists, this setting is completely similar to the experiments reported above and in Reference [40]. Based on our experiments, this setting does not change the performance of Term-supervised *LDC* significantly.

5 VIS-KT - A VISUAL FRAMEWORK TO SUPPORT KEYTERM-BASED CLUSTERING

This section presents the visualization framework designed to support Term-supervised *LDC*. Different sections of the interface are introduced first. Next, two clustering use cases are reported. The results of a user study conducted by using the framework are reported at the end of this section.

The main steps of the framework are depicted in Figure 11. In the first step, the user uploads a document collection, which is then pre-processed as mentioned in Section 4. Based on *tfidf*, the Document-Term matrix of the collection is generated. Next, the user suggests an initial number of clusters, that the system uses to generate the initial document clustering. After clustering, keyterms of the document clusters are extracted and displayed. All other visualizations are also generated. With the help of various visualization modules, the user can change the topics of clusters by adding or removing keyterms, reordering them, and also by changing the number of clusters, and immediately observing the effect of her choices by asking to re-cluster documents. All proposed changes will then be sent to the clustering algorithm, Term-supervised *LDC*. A new document clustering is then generated and displayed to the user and all visualizations are adjusted accordingly. This process repeats until the clustering results are satisfactory to the user.

Table 2. Minimum User-effort and Expertness Required Such that Term-supervised *LDC* Outperforms *LDA* and Unsupervised *LDC* Significantly, Based on the Paired-sample t-test with $p \leq 0.05$

Dataset Name	outperform unsupervised *LDC*	outperform *LDA*
NewsGroup9	$f = 3$ and $P_{exp} = 1.0$ $f = 4$ and $P_{exp} = 0.8$ $f = 5$ and $P_{exp} = 0.8$ $f = 6$ and $P_{exp} = 0.7$ $f = 7$ and $P_{exp} = 0.7$ $f = 8$ and $P_{exp} = 0.7$ $f = 9$ and $P_{exp} = 0.7$ $f = 10$ and $P_{exp} = 0.6$	$f = 2$ and $P_{exp} = 0.8$ $f = 3$ and $P_{exp} = 0.7$ $f = 4$ and $P_{exp} = 0.6$ $f = 5$ and $P_{exp} = 0.6$ $f = 6$ and $P_{exp} = 0.6$ $f = 7$ and $P_{exp} = 0.5$ $f = 8$ and $P_{exp} = 0.5$ $f = 9$ and $P_{exp} = 0.5$ $f = 10$ and $P_{exp} = 0.5$
NewsRelated	$f = 3$ and $P_{exp} = 1.0$ $f = 4$ and $P_{exp} = 1.0$ $f = 5$ and $P_{exp} = 0.9$ $f = 6$ and $P_{exp} = 0.9$ $f = 7$ and $P_{exp} = 1.0$ $f = 8$ and $P_{exp} = 1.0$ $f = 9$ and $P_{exp} = 0.9$ $f = 10$ and $P_{exp} = 0.9$	$f = 3$ and $P_{exp} = 1.0$ $f = 4$ and $P_{exp} = 1.0$ $f = 5$ and $P_{exp} = 0.9$ $f = 6$ and $P_{exp} = 0.8$ $f = 7$ and $P_{exp} = 0.7$ $f = 8$ and $P_{exp} = 0.8$ $f = 9$ and $P_{exp} = 0.8$ $f = 10$ and $P_{exp} = 0.7$
WebKB	$f = 9$ and $P_{exp} = 1.0$ $f = 10$ and $P_{exp} = 0.9$	$f = 9$ and $P_{exp} = 1.0$ $f = 10$ and $P_{exp} = 0.9$
SMS	No significant improvement with $f = 10$ and $P_{exp} = 1.0$	$f = 3$ and $P_{exp} = 0.7$ $f = 4$ and $P_{exp} = 0.7$ $f = 5$ and $P_{exp} = 0.6$ $f = 6$ and $P_{exp} = 0.6$ $f = 7$ and $P_{exp} = 0.6$ $f = 8$ and $P_{exp} = 0.6$ $f = 9$ and $P_{exp} = 0.6$ $f = 10$ and $P_{exp} = 0.6$
BBC Sport	No significant improvement with $f = 10$ and $P_{exp} = 1.0$	$f = 2$ and $P_{exp} = 0.9$ $f = 3$ and $P_{exp} = 0.5$ $f = 4$ and $P_{exp} = 0.5$ $f = 5$ and $P_{exp} = 0.4$ $f = 6$ and $P_{exp} = 0.4$ $f = 7$ and $P_{exp} = 0.4$ $f = 8$ and $P_{exp} = 0.4$ $f = 9$ and $P_{exp} = 0.3$ $f = 10$ and $P_{exp} = 0.3$
NewsSeparate	$f = 1$ and $P_{exp} = 1.0$ $f = 2$ and $P_{exp} = 0.8$ $f = 3$ and $P_{exp} = 0.8$ $f = 4$ and $P_{exp} = 0.7$ $f = 5$ and $P_{exp} = 0.6$ $f = 6$ and $P_{exp} = 0.6$ $f = 7$ and $P_{exp} = 0.5$ $f = 8$ and $P_{exp} = 0.5$ $f = 9$ and $P_{exp} = 0.5$ $f = 10$ and $P_{exp} = 0.5$	$f = 2$ and $P_{exp} = 1.0$ $f = 3$ and $P_{exp} = 0.9$ $f = 4$ and $P_{exp} = 0.8$ $f = 5$ and $P_{exp} = 0.7$ $f = 6$ and $P_{exp} = 0.7$ $f = 7$ and $P_{exp} = 0.7$ $f = 8$ and $P_{exp} = 0.6$ $f = 9$ and $P_{exp} = 0.6$ $f = 10$ and $P_{exp} = 0.6$

Only one round of term labeling is performed for term-supervised *LDC*. Significant improvements are achieved based on both *F1* and *NMI* measures.

5.1 Visual Interface

Vis-Kt's interface consists of different visualization modules called views or panels. They are linked to each other and work as a group. A snapshot of the interface is shown in Figure 12.

5.1.1 Header Panel. This panel is at the top of the interface (Figure 12(A)). On its right side, a search bar is available where the user searches for a term in the vocabulary and adds it to a selected cluster. To select a cluster, she clicks on a rectangle inside *Clusters View* (Figure 12(D)). On its left side, the user can save the current work session, add comments about the clustering,

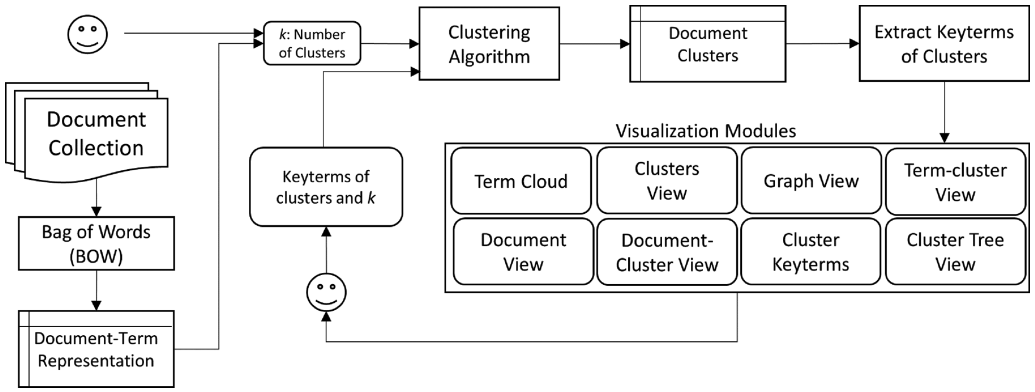


Fig. 11. The main steps of the Visual Keyterm-based (*Vis-Kt*) clustering. A document collection is first pre-processed using *BoW*. An initial document clustering is generated using *LDC* and is displayed to the user. The user manipulates the keyterms of clusters, inspired by various visualization modules, and sends a document re-clustering signal. The modules are adjusted according to the new clustering. This process continues until the clustering becomes satisfactory.

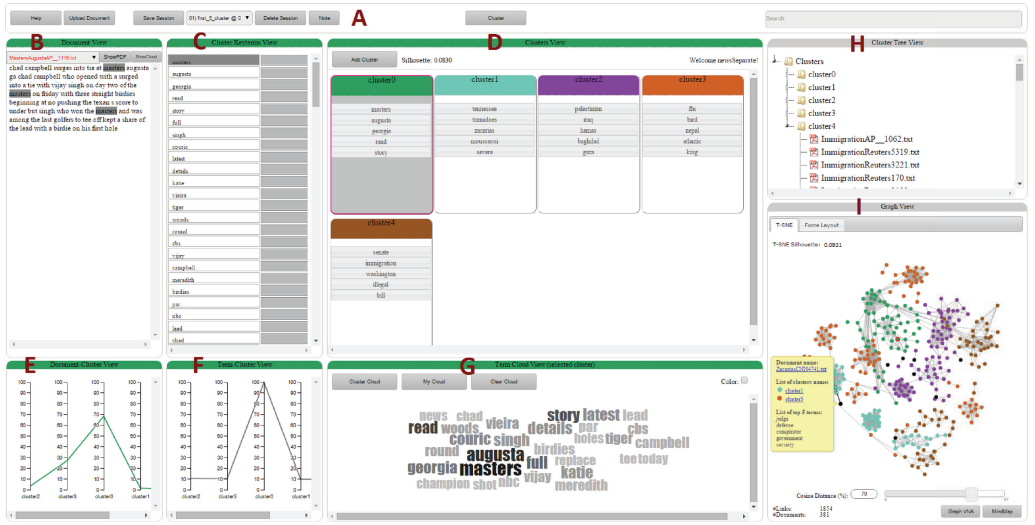


Fig. 12. A snapshot of the proposed visual interface. (A) *Header Panel*: contains buttons to save the current work session, search terms in vocabulary, and send re-clustering signal. (B) *Document View*: shows the textual content of documents. (C) *Cluster Keyterms View*. includes a list of top keyterms of a cluster. (D) *Clusters View*: includes a rectangle for each cluster. Each rectangle contains the top five keyterms of each cluster by default. (E) *Document-Cluster View*: a parallel coordinator diagram demonstrates the relatedness of a document to the clusters. (F) *Term-Cluster View*: a parallel coordinator diagram demonstrates the relatedness of one or multiple terms to the clusters. (G) *Term Cloud View*: displays a term cloud based on either keyterms of a rectangle, keyterms of *Cluster Keyterms View* (C), or keyterms of a document and its neighbors. (H) *Cluster Tree View*: depicts document clusters in the form of a folder tree. (I) *Graph View*: includes two graph representations of documents. Each node is a document and nodes with similar color are in the same cluster.

and move to other pages, such as upload and help. Saving a session helps the user to re-load a particular clustering later. The user sends the re-cluster signal to the clustering algorithm after she has provided feedback by manipulating terms inside the rectangles. The re-cluster button is at the middle of the panel. The clustering algorithm re-clusters the documents based on the terms inside the rectangles in *Clusters View* as explained in Section 3.

5.1.2 Document View. In the left side of the interface (Figure 12(B)), the content of documents is shown. The user selects a cluster and then all documents of the respective cluster are listed as a drop-down menu. By selecting a document, one can view its content, download its original file (currently in *PDF* format), and see its top keyterms. Pressing the *ShowPDF* button downloads the original file. Pressing the *ShowCloud* button creates a term cloud in *Term Cloud View* (Figure 12(G)). The term cloud includes terms with high *tfidf* values in the document. Double clicking on any term inside the content of a document adds it to the term list in the rectangle of the selected cluster. Adding terms can also be done by dragging and dropping.

Selecting a document from the drop-down menu also affects the *Document-Cluster View* (Figure 12(E)) and the *Graph View* (Figure 12(I)). The *Document-Cluster View* shows the membership values of the document in document clusters using a parallel coordinates visualization. In the *Graph View*, the selected document is highlighted along with its adjacent neighbors.

5.1.3 Cluster Keyterms View. The *Cluster Keyterms View* (Figure 12(C)) contains top keyterms of the selected document cluster. Terms are sorted based on their relevances to the cluster. The relevances are computed using χ^2 statistic as explained in Section 3. The bar chart in front of each term also shows the relevances. Double clicking on any term adds it to the rectangle of the selected cluster. This option also works by dragging and dropping. Whenever a term is selected by single click, it will be highlighted in the content of the selected document in *Document View*. The *Graph View* also highlights documents in which the selected term appears. Selecting a term also affects the *Term-cluster View* (Figure 12(F)), the parallel coordinate diagram. The diagram shows the relevances of the term to document clusters. The user can also select multiple terms to compare their relevances to clusters in the parallel coordinate diagram.

5.1.4 Clusters View. The *Clusters View* is the main panel of the interface (Figure 12(D)). Before any interaction, it displays the top five keyterms of each document cluster in a rectangle. The name of each cluster is at the top of its rectangle. The user can assign a title for each cluster or change its color. The titles and colors do not affect the clustering algorithm, but terms inside rectangles do.

The vertical positions of terms in a rectangle indicate their relevance to the respective cluster. The top term has the highest relevance. The minimum number of terms in each rectangle is one.

The *Clusters View* is coordinated with the other panels. Once the user clicks on a term of a rectangle, *Graph View* highlights the documents in which that term appears, *Document View* highlights the term in the content of documents, and *Term-Cluster View* depicts its relevances to clusters in the parallel coordinate diagram. Selecting a rectangle also highlights the documents of the respective cluster in the *Graph View*.

One interesting option of *Clusters View* is adding a new cluster. By clicking on "Add Cluster," an empty rectangle will be added, which then must be populated by words in any of the possible forms to do that (typing, clicking, dragging, moving words). To remove a cluster, users can right-click on its rectangle and choose "Delete Cluster" or "Delete all terms." The framework ignores any empty rectangles. Deleting a cluster does not remove its documents from the collection or clustering process. In fact, none of these panels support adding or removing documents. A particular page is designed for that purpose.

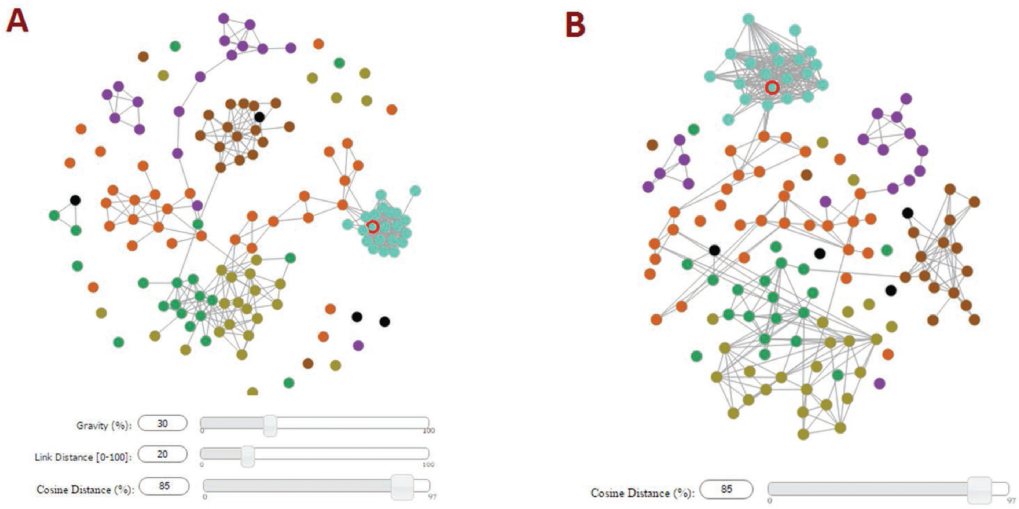


Fig. 13. Two graph layouts in *Graph View*. *Force Layout* (A) tries to minimize the number of crossing edges. The edges have no contribution in the position of nodes in *t-SNE* (B). Based on our experiments, *Force Layout* provides a better projection in case of large collection, due to less clutter. Otherwise, both methods generate nearly similar projections.

The Silhouette index is an intrinsic evaluation measure to validate the consistency within clusters [49]. The Silhouette index of the clustering is shown at the top of this view.

5.1.5 Term Cloud View. The *Term Cloud View* displays term clouds (Figure 12(G)). The user can create them from terms inside a rectangle in *Clusters View*, from top keyterms of a cluster in *Cluster Keyterms View*, from a document and its adjacent document in *Graph View*, or from a single document in *Document View*. Terms with darker and larger font in the term clouds are the ones with larger weights. The user can drag and drop terms from clouds into the rectangles of the *Clusters View*.

5.1.6 Cluster Tree View. Cluster Tree View is a folder tree representation of document clusters (Figure 12 H). Each node corresponds to a cluster and contains its respective documents. By expanding a node, the user can view the original files of documents and download them. An option to download a cluster as a zip file is also available. The folder tree is linked to the other panels such that clicking on a node has the same effects as clicking on a rectangle in *Clusters View*.

5.1.7 Graph View. The *Graph View* (Figure 12(I)) is 2D layout of the document set, with a corresponding graph representation. Each node is a document and each edge represents similar documents based on the Cosine similarity. One can control the connectivity level by changing the similarity threshold via the slider at the bottom of this view. Colors of nodes correspond the the colors of their cluster in the other views. Documents belonging to multiple clusters are colored black. The selected document in the *Document View* has a red stroke around its node (Figure 13). One tooltip is associated with each node, which includes the document's file name, its original file link, a list of the clusters it belongs to, and its top five terms. Clicking on a node highlights a group of adjacent documents including the document itself. Zoom in and zoom out are available for this view.

Table 3. The Silhouette Indices of Clusterings Generated on *NewsSeparate* in Each Iteration

	Average Silhouette			
	Iteration1	Iteration2	Iteration3	Iteration4
Clustering	0.0830	0.1718	0.2011	0.2224
t-SNE	0.0522	0.2305	0.3530	0.3385
Force Layout	0.0474	0.2423	0.3774	0.4681

The silhouette indices of t-SNE and force layout are independent of the user interaction. The silhouette index of the clustering shows the quality of clustering.

Two projection techniques, *t-SNE* [36] and *Force Layout* [21], are employed to map the full set of documents into two dimensional space for this view (Figure 13). For t-SNE, the position of points is determined by the whole similarity relationship among them; in this case, changing the connectivity threshold will not change the positioning of points on the layout but will add and remove edges. Force layout is based on the current status of the graph connectivity, so it will change when users choose new similarity thresholds. The D3 library [9] is used to generate the graphs. Silhouette indices of projections are also shown in this view.

The Vis-Kt framework is implemented as a web-based application using *Javascript*, *D3*, and *jQuery*. The background clustering algorithm is implemented in *Python* and *Perl*. It is freely available as a platform-independent open source application.[11]

5.2 Clustering Use Cases

This section presents two case studies on clustering real text datasets. The authors perform the case studies and the class labels of documents are the only information available about datasets. However, the goal is to obtain insight into the topics but not to necessarily generate clusterings similar to true classes. We evaluate the quality of clusters based on the Silhouette index and also whether the framework is effective in separating topics in the graph projections.

5.2.1 Use case 1: Clustering NewsSeparate. The scenario presented here demonstrates how a user can change the topics and number of clusters to reflect her preferences in the process. Although the framework is very flexible and different users may choose to operate by employing different modules, we follow one possible set of steps to illustrate the process and the support our framework offers. To this end, we have employed *NewsSeparate*, which contains news articles in 13 topics that can be identified as "Trial of Zacarias Moussaoui," "Viondrug dispute," "Tennessee Tornadoes," "Stock Price," "Iraq Suicide," "Nepal Strikes," "Masters Augusta (Golf)," "Katie Courie," "Immigration Bill," "Hamas," "Enron Insurance case," "Bird Flu," and "Da Vinci Code dispute."

Even though we know the number of classes, the initial clustering was generated with five clusters. The rectangles of the clusters, the graph representation of the clustering, and a term cloud of each subgraph are depicted in Figure 14. Based on the keyterms of the rectangles, one can observe that the topics of clusters were "Golf," "Tennessee Tornadoes," "Hamas and Palestine," "Bird Flu," and "Immigration Bill." The average Silhouette indices of this iteration are shown in Table 3.

The term clouds of the subgraphs clearly demonstrate that multiple disjoint groups of documents (based on similarity) were assigned to the same cluster. For instance, the term clouds of two

[11] http://ares.research.cs.dal.ca/IC2/.

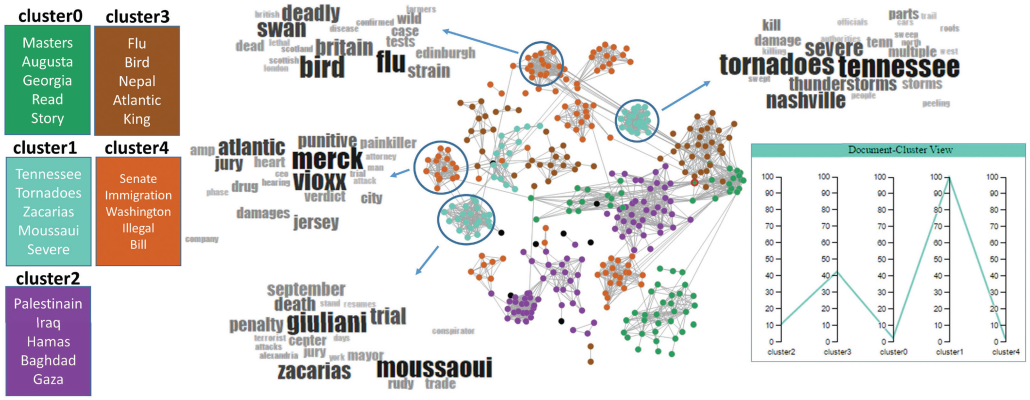


Fig. 14. Use case 1: clustering *NewsSeparate*—Iteration 1: Term clouds of different subgraphs of "cluster1" and "cluster4" show that these clusters must be separated to obtain finer sub-clusters. Top five keyterms of each cluster are shown in the rectangles. Document-Cluster View shows that a single document belongs to "cluster1" and "cluster3" with different degrees of relevance.

subgraphs of "cluster1" correspond to the documents of "Trial of Zacarias Moussaoui" and "Tennessee Tornado." This observation clearly implies that the number of clusters should increase.

We also observed that a few documents were assigned to more than one cluster (black nodes in the graph). A part of these documents contain discussions about the economic aspect of bird flu and were placed into both "Stock Price" and "Bird Flu" clusters. The other documents cover two topics of "Tennessee Tornadoes" and "Bird Flu."

After analyzing the results, we decided to add five new clusters for the next iteration. The topics of the new clusters were specified by using terms in the term clouds, term lists, and document contents. A re-clustering signal along with the term lists were sent to the clustering algorithm.

It is logical that the topics of document clusters alter after each iteration; for instance, "cluster1" in the following iterations will not represent the same topic as in the current iteration.

The results of the second iteration are depicted in Figure 15. Documents were grouped into ten clusters and improvement was achieved based on the Silhouette index (Table 3). The improvement confirms that increasing the number of clusters was successful. Nevertheless, there were still different groups of documents placed in the same cluster. We thus added two new clusters targeting at segregating these topics.

The output of the third iteration is shown in Figure 16. We observed that some documents were mistakenly placed in cluster "Da Vinci Code dispute," while they actually belong to "Stock Price." These topics share keyterms such as "London," "legal," and "expected." We thus removed these common keyterms from the rectangles for the next iteration. The alternative was to keep them in the rectangles with different levels of relevance, but we did not have enough knowledge about the collection to do so. For the fourth iteration, we added two new clusters (Figure 17), and we decided to stop at this point.

This use case shows the efficacy of the framework to generate user-desired clusterings by increasing the number of clusters and specifying the topics of interest. The same scenario is applicable to form the clusters in a top-down approach by merging clusters and reducing the number of clusters.

5.2.2 Use case 2: Clustering BBC Sport. This dataset consists of news articles in five topics, namely "athletics," "cricket," "football," "rugby," and "tennis." An initial clustering was generated

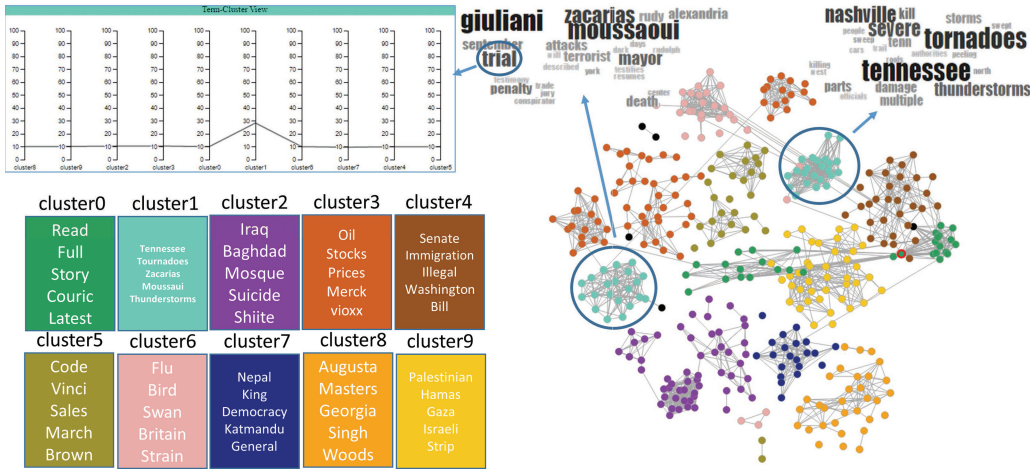


Fig. 15. Use case 1: clustering *NewsSeparate*—Iteration 2: Except for "cluster1," most clusters contain only one group of adjacent documents. There are two groups of documents in "cluster1." The topic of one group is "Trial of Zacarias Moussaoui" and the topic of the other one is "Tennesse Tornadoes." Based on the term clouds, there are some common terms like "death" in both groups. The Term-Cluster View shows the relevance of term "trial" in document clusters.

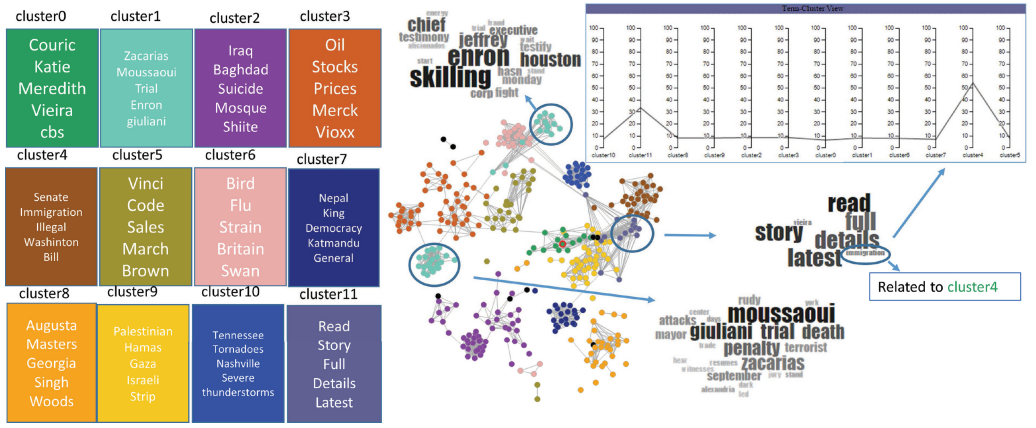


Fig. 16. Use case 1: clustering *NewsSeparate*—Iteration 3: The term cloud of "cluster1" shows that a new cluster is needed to separate topics of "Enron Insurance case" and "Trial of Zacarias Moussaoui." The Term-Cluster View of "immigration" shows that it is more related to "cluster4" than to "cluster11."

with five clusters. The Silhouette index of clustering was 0.0413 and clusters were well-separated (Figure 18(a)). The clusters of "football" and "rugby" had the most multi-clustered documents; clusters of "tennis" and "cricket" had no common documents and were placed far from each other in the graph.

An interesting observation of the first iteration was that one document, which belongs to "cricket," was clustered into "rugby," "football," and "cricket." The term cloud of this document showed that it contains terms "Scotland" and "club," which are the keyterms of "rugby" and "football," respectively. Terms like "manager," "coach," and "league" and the names of countries and

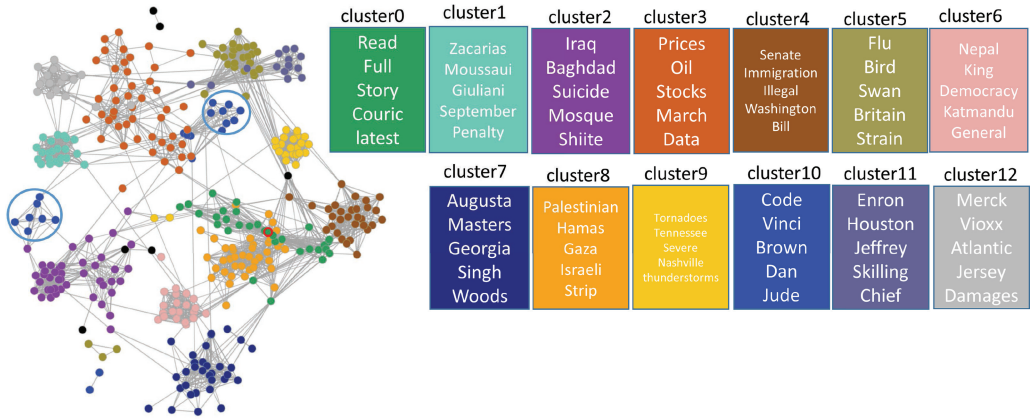


Fig. 17. Use case 1: clustering *NewsSeparate*—Iteration 4: Result of clustering *NewsSeparate* after four iterations. Most groups of adjacent nodes of the graph are in distinct clusters. Some clusters, however, could be distilled further, for instance "cluster10."

cities such as "Australia," "England," and "Athens" are also common among different topics. To clarify the topics of interest, these common terms were removed from rectangles.

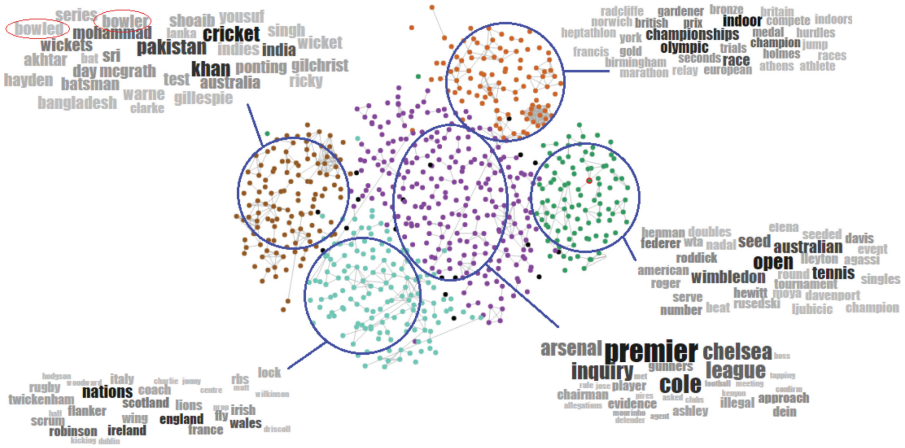
Another observation was that a group of documents in cluster of "athletics" (orange nodes in the graph (Figure 18(a))) were fully connected and projected next to each other. After creating a term cloud of these documents, we could see that the terms "doping," "drugs," and "trial" were common among them. We thus added a new cluster for topic of "doping" with the mentioned terms. The clustering was run for another iteration and a new cluster of "doping" was successfully created (Figure 18(b)). The silhouette index of the clustering was increased to 0.0434 as well.

The other observation in this use case is that terms "bowled" and "bowler" appeared in the top keyterms of cluster "cricket" (brown nodes (Figure 18(a))). Initially, we thought that a cluster of "bowling" was merged into "cricket." A new cluster with terms "bowled," "bowler," and "bowling" was thus created to examine whether the framework would generate a new cluster. The change was successful and a new cluster of "bowling" was created in the third iteration (Figure 18(c)). This cluster contained the keyterms "vaghuan," "boje," "kallis," "giles," "strauss," and "bowling." Most of these keyterms are the names of players who are (or were) famous because of the bowling action in cricket. Bowling is actually the act of throwing the ball toward the wicket in cricket. This cluster is located above "cricket" as shown in Figure 18(c).

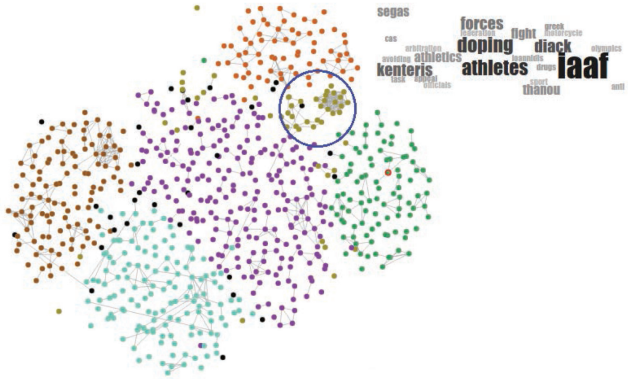
This use case illustrates that even though the documents are categorized into five classes, we could successfully extract some sub-clusters by using the projections and views available in the framework. *Vis-Kt* is able to reflect additional insights into the dataset much beyond the information available as the ground truth.

5.3 User Study

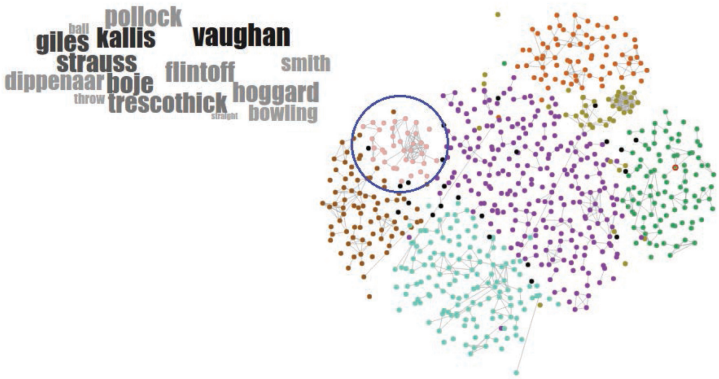
We conducted a user study to evaluate the utility of the framework in interaction with real users. The main targets of the study were to evaluate the individual differences in clustering the same collection and also the utility of the visual aids in performing iteration of keyterm-based clustering. Analysis of the data gathered through the user study revealed that different users have different points of view in clustering the same collection. The users comments also confirmed that presenting the topics of document clusters in the form of term clouds is effective in exploring text



(a)



(b)



(c)

Fig. 18. Use case 2: Clustering *BBC Sport*. (a) The initial clustering, with five clusters, has the Silhouette index of 0.0413. (b) The second clustering is obtained after adding a new cluster about "doping." The Silhouette index of clustering is increased to 0.0434. (c) The third clustering is generated after adding a new cluster about "bowling." The increment in the number of clusters was successful and the Silhouette index is increased to 0.0456.

Table 4. The Participants' Ratings in the Questionnaire

Rate	Label
1	"Strongly Disagree"
2	"Somewhat Disagree"
3	"Neutral"
4	"Somewhat Agree"
5	"Strongly Agree"

corpora. Most participants were also successful in generating their desired number of clusters by using the interface.

Participants of the study were thirty computer science students with passing familiarity with document clustering. They knew the role of keyterms in scientific articles. We introduced the framework to the participants and let them perform as many iterations as they wanted. They could do the study in any place at any time using any computer on their own datasets. Most of the participants datasets consisted of the scientific articles they have read or intended to read for their projects or research work.

We ran the study in two modes. In the first mode, 21 participants clustered by themselves their own document collections, averaging 500 documents each. In the second mode, a group of nine participants individually clustered the same collection of 300 scientific articles. After the user study, each participant filled out a questionnaire and provided comments. The questionnaire had 20 statements and the participants rated each question by a value between one and five (Table 4). In the following sections, we report the results of the study in detail.

5.3.1 Different Points of View. The aim of this analysis is to determine whether the participants of the group mode have different points of view in clustering. We created one collection for them. The collection includes 300 scientific articles in the area of text mining. Each participant clustered the collection without any collaboration with the other participants of the group. The maximum number of clusters generated is 7 and the minimum number is 3 (Figure 19).

Users 1 and 3 have generated three clusters with almost similar topics: "Visualization," "Subspace Projection," and "Semantic Analysis." Users 5, 6, and 7 have generated five clusters. Four clusters are common among them including "Sentiment analysis," "Visualization," "Subspace Projection," and "Semantic analysis." Users 8 and 9 have generated four clusters. User 2 has generated seven clusters. He has generated a cluster about "Topic Modeling," which has not been seen in the others' clusterings.

We also analyzed the amount of time spent by the participants. The analysis shows that there is no correlation between the number of clusters and the interaction time, based on Spearman[12] and Kendall[13] correlation tests. For instance, users 3 and 6 have both interacted with the framework between 1 and 2h, while user 3 has generated three but user 6 has generated six clusters. Or, users 5, 6, 7 have spent different times but generated the same number of clusters.

The number of generated clusters and their topics confirm that different users have different points of view in clustering the same collection. Participants with the same number of clusters

[12] http://en.wikipedia.org/wiki/Spearman's_rank_correlation_coefficient.
[13] http://en.wikipedia.org/wiki/Kendall's_tau.

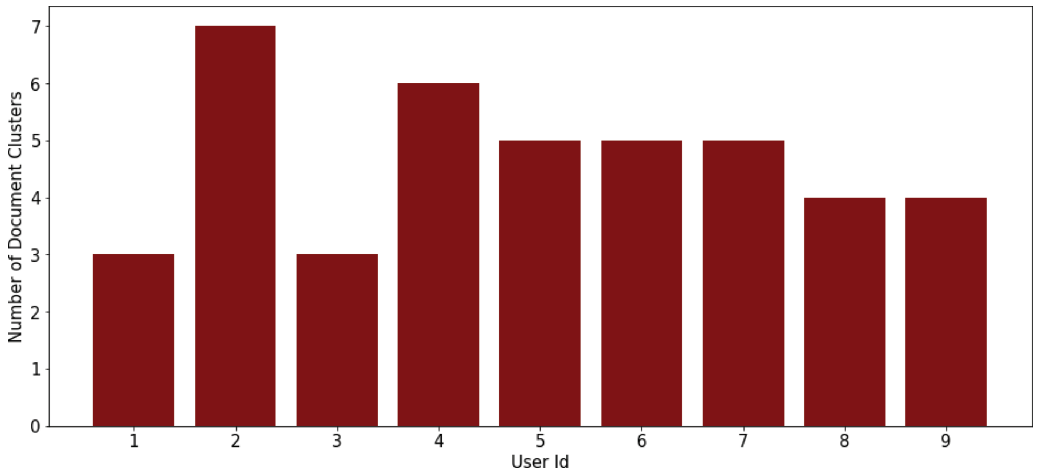


Fig. 19. Different numbers of document clusters are generated by nine participants. The average number of clusters is 4.66 and its standard deviation is 1.32. The collection includes 300 scientific articles in the area of text mining.

have also targeted different topics in our study. This observation raises the following questions regarding the ground truth of textual document sets:

- In building ground truth of a document set, should we merge different points of view or provide multiple labels for each document? The process of document labeling is usually performed manually and multiple experts may provide multiple labels for each document. The number of clusters may also be different. We believe that, after analyzing the labels, each collection should come with multiple sets of ground truth, each representing a candidate clustering. This is mainly because merging different points of view would hide much useful information about topics of text datasets.
- Can we rely on existing gold standard datasets where only single label set for each document collection is provided? There are many clustering algorithms in the literature but only a few of them would generate acceptable results in practice. One reason is that the benchmark text collections are mostly single-labeled and by incorporating and tuning some parameters the proposed clustering algorithms may be optimized to provide the best results. Having multiple label sets would better reveal the performance of a text clusterer or the amount of effort needed to tune its parameters. We believe that comparisons based on gold standard datasets provide a general view of the performance of an algorithm, but interaction with real users reveals its acceptance truly.

5.3.2 User-Supervised Clustering. We analyze the participants' opinions about being able to change the number of clusters and their topics in this section. Three statements were considered in the questionnaire for this purpose:

(1) The automatically generated clusters are far from what you desired.
(2) It is necessary to be able to change the number of clusters.
(3) It is necessary to be able to change the topics of document clusters generated automatically.

Dependency tests based on Spearman and Kendall correlations reveal that:

- Statements 2 and 3 are independent of statement 1. This means that the participants like to be involved in the clustering process regardless of the quality of the clusters generated automatically. Regarding the first statement, 4 participants selected "Somewhat Agree," 8 selected "Neutral," 15 selected "Somewhat Disagree," and 3 participants selected "Strongly Disagree." Therefore, 13% of participants did not like the automatically generated clusters and for the other 87%, the clusters generated by unsupervised *LDC* were close to what they expected.
- Statements 2 and 3 are dependent on each other. This means that if a participant likes to change the number of clusters, she also likes to change the topics of clusters and vice versa. Hence, tuning the clustering algorithms with different numbers of clusters is not sufficient and the participants like to be able to change the topics of document clusters as well. Only two participants disagree with the second statement and three participants disagree with the third one.
- The participants' ratings to these three statements are independent of the amount of time they spent on clustering.

The main conclusion of this analysis is that even though *LDC* can generate clusterings close to what participants like to generate, they still like to interact with the clustering process to obtain their desired topics of interest or explore the collection.

5.3.3 Number of Document Clusters. To evaluate the role of the framework in generating the user-desired number of clusters, we considered two statements in the questionnaire:

(1) Term-based visualization is a useful way to find a desired number of clusters.
(2) It is easy to determine a desired number of clusters.

We analyzed the participants' responses to these questions and obtained the following results:

- Dependency tests based on Spearman and Kendall correlations reveal that the first statement is independent of the interaction time, but the second one has an inverse correlation with time. This means that some participants who had difficulties to determine a desired number of clusters spent more time with the framework and vice versa.
- Twenty participants agreed that the visualization is useful to find a desired number of clusters.
- Regarding the second statement, only two participants disagreed.
  - —Participant *A* also believes that the term-based visualization is not a useful way to find a desired number of clusters. She also somewhat disagreed with the statement, "the interface is fast enough as an interactive system." However, she is either neutral or agreed with the other statements of the questionnaire; for instance, she somewhat disagreed with the statement, "The automatically generated clusters are far from what you desired."
  - —Participant *B* never agreed with any statement of the questionnaire. Her opinion about the statement, "The automatically generated clusters are far from what you desired," is also neutral. She also believes that the interface is not fast enough to be used interactively.

  After analyzing the responses of these two participants, we realized that they both believe the framework is not fast enough to be used interactively. To determine whether the framework is fast enough or not, we evaluate all participants' rating to the statement, "The user interface is fast enough as an interactive system."

  The dependency's tests reveal that the statement is independent of the interaction time. Out of 30 participants, 27 participants agreed that the system is fast and only 3 participants disagreed. The running time of the interface was acceptable to 90% of participants.

One possible answer to the case of Participant *A* is that our server may was busy during her user study, because she gave us the comment, "*If it runs on a local machine, then I imagine it'd be faster, but running the clustering algorithm on a server seemed to be slow more often than I wanted.*"

However, we could not realize why participant *B* had difficulties in clustering her collection. Her comment in the questionnaire is that, "*From the point of the view of aesthetics, the user interface is not very pretty. Maybe a new design can be conducted to the layout and the colors combination.*"

5.3.4 Document Content View. A document content view is considered for each document cluster in the interface. This view helps the user in exploring the content of documents and getting better insight into the clusters. After analyzing the responses, we found that only three participants did not find the document view useful and seven participants were neutral. Around 67% of participants mentioned that this view was useful. This result reveals that the participants may like to view the contents even though the underlying clustering algorithm, Term-supervised *LDC*, is based on term labeling.

5.3.5 User Study Conclusion. A summary of observations obtained after analyzing the responses is mentioned below:

(1) In the group mode, different numbers of clusters were generated. This mode of the study revealed that different points of view existed in clustering the same collection.
(2) Around 13% of participants did not agree that the automatically generated clusters were close to what they expected. In 83% of studies, if a participant liked to change the number of clusters, she also liked to change their topics. Around 93% of participants wanted to change the number of document clusters and 90% of them intended to change the topics.
(3) Around 90% of participants agreed that term-based visualization is a useful approach in exploring the topics of a collection and around 83% of them mentioned that it was easy to realize the topics of clusters from keyterms.
(4) Around 93% of participants agreed that term labeling was effective in generating topics of interest.
(5) Around 83% of participant found it easy to change the number of clusters in the framework.
(6) An interesting observation was that 67% of participants believed that showing the content of documents was useful.
(7) Out of 30 participants, 27 participants agreed that the system was fast enough to be used interactively. However, it should be mentioned that no participant had more than a few hundred documents in our study. The size of dataset can affect the running time of the framework.
(8) Participants took three iterations in average (maximum five) to finalize the process.

From the free comments given by users, we highlighted all around enthusiastic support for the interactive framework and for the use of keyterms as a supervision strategy. There were also frequent compliments to impressive results. Some users suggested visual changes (such as in color), which are currently implemented in the system. A few users suggested the extension of the method to phrases, rather than just single-word terms. We intend to carry out some of the extensions for the next versions of the framework.

6 CONCLUSION AND FUTURE WORK

The proposed framework has demonstrated to have effective clustering results and intuitive user appeal. Case and user studies have shown the power and flexibility of the visual approach in inserting user knowledge into the clustering process. Previous work on guided text clustering based on topical content have used *LDA* (e.g., Reference [35]). Although effective in performance for interactive purposes, the *LDA*-based interactive clustering has the following drawbacks: the main influence in the clustering results occurs by explicitly changing term-topic weights, which is not intuitive for the users; cluster merging may be available at the interface level, but it has to be done literally by feeding back into the model and make a final result consistent with the model; cluster splitting runs the model locally, but the results are not as predictable as in keyterm-based clustering. Our approach, on the other hand, has tightly coupled model and the final result is consistent between the interface and the model and yields predictable outcome.

The results of the user study have shown that the majority of the users reached their target clusterings by using the framework's functionality. Many were impressed with how fast they met their clustering goals. This is partly a reflection of the visual modules and partly a reflection of the flexibility and predictability of the keyterm-based framework as a whole. The acceptance of Vis-Kt was widely positive.

The applications for this type of frameworks range widely, from personal organization of individual document collections to pre-processing larger collections destined to feed other algorithms in a user-centered environment. Vis-Kt application, code, and data are made freely available to general users. Extensions planned include phrase-based clustering and creating an infra-structure for large data sets; leveraging Wikipedia concepts along with terms in the clustering process based on the algorithm proposed in Reference [42]; adjusting the threshold used to generate soft clustering on the graph representations directly; allowing the users to adjust the label of mis-clustered documents besides term labeling; extending Vis-Kt to a clusterer of stream text data. Keyterms of new documents will be extracted [38, 39, 48] and shown to the user and then she decides whether new clusters are needed or do cuments can fit in the current clusters.

REFERENCES

[1] C. C. Aggarwal and Ch. Zhai. 2012. A survey of text-clustering algorithms. In *Mining Text Data*. Springer, 77–128.

[2] D. Andrzejewski, X. Zhu, and M. Craven. 2009. Incorporating domain knowledge into topic modeling via dirichlet forest priors. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML'09)*. ACM, New York, NY, 25–32.

[3] P. Awasthi, M. F. Balcan, and K. Voevodski. 2017. Local algorithms for interactive clustering. *J. Mach. Learn. Res.* 18, 3 (2017), 1–35.

[4] S. Basu, A. Banerjee, and R. J. Mooney. 2002. Semi-supervised clustering by seeding. In *Proceedings of the 19th International Conference on Machine Learning (ICML'02)*. Morgan Kaufmann, San Francisco, CA, 27–34.

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

[6] M. W. Berry, Mu. Browne, A. N. Langville, V. P. Pauca, and R. J. Plemmons. 2007. Algorithms and applications for approximate nonnegative matrix factorization. *Comput. Stat. Data Anal.* 52, 1 (2007), 155–173.

[7] J. Bezdek. 1981. *Pattern Recognition with Fuzzy Objective Functions*. Kluwer Academic Publishers, Norwell, MA.

[8] D. M. Blei, A. Y. Ng, and M. I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.* 3 (2003), 993–1022.

[9] M. Bostock, V. Ogievetsky, and J. Heer. 2011. D^3 data-driven documents. *IEEE Trans. Visual. Comput. Graph.* 17, 12 (2011), 2301–2309.

[10] L. Boudjeloud-Assala, Ph. Pinheiro, A. Blansché, Th. Tamisier, and B. Otjacques. 2015. Interactive and iterative visual clustering. *Information Visualization* 15, 3 (2015), 181–197.

[11] E. T. Brown, J. Liu, C. E. Brodley, and R. Chang. 2012. Dis-function: Learning distance functions interactively. In *Proceedings of the IEEE Conference on Visual Analytics Science and Technology (VAST'12)*. 83–92.

[12] J. Choo, C. Lee, C. K. Reddy, and H. Park. 2013. UTOPIAN: User-driven topic modeling based on interactive nonnegative matrix factorization. *IEEE Trans. Visual. Comput. Graph.* 19, 12 (Dec 2013), 1992–2001.

[13] J. Chuang, C. D. Manning, and J. Heer. 2012. Termite: Visualization techniques for assessing textual topic models. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*. ACM, New York, 74–77.

[14] G. V. Cormack, J. M. G. Hidalgo, and E. P. Sánz. 2007. Feature engineering for mobile SMS spam filtering. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'07)*. ACM, New York, NY, 871–872.

[15] D. Costa and G. Venturini. 2007. A visual and interactive data exploration method for large data sets and clustering. In *Proceedings of the 3rd International Conference on Advanced Data Mining and Applications*. Springer-Verlag, 553–561.

[16] W. Cui, Sh. Liu, L. Tan, C. Shi, Y. Song, Z. J. Gao, H. Qu, and X. Tong. 2011. Textflow: Towards better understanding of evolving topics in text. *IEEE Trans. Visual. Comput. Graph.* 17, 12 (2011), 2412–2421.

[17] M. des Jardins, J. MacGlashan, and J. Ferraioli. 2007. Interactive visual clustering. In *Proceedings of the 12th International Conference on Intelligent User Interfaces*. ACM, New York, 361–364.

[18] A. Endert, P. Fiaux, and C. North. 2012. Semantic interaction for visual text analytics. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'12)*. ACM, New York, 473–482.

[19] A. Endert, S. Fox, D. Maiti, S. Leman, and C. North. 2012. The semantics of clustering: Analysis of user-generated spatializations of text documents. In *Proceedings of the International Working Conference on Advanced Visual Interfaces (AVI'12)*. ACM, New York, 555–562.

[20] P. Fiaux, M. Sun, L. Bradel, C. North, N. Ramakrishnan, and A. Endert. 2013. Bixplorer: Visual analytics with biclusters. *Computer* 46, 8 (2013), 90–94.

[21] Th. M. J. Fruchterman and E. M. Reingold. 1991. Graph drawing by force-directed placement. *Softw. Pract. Exper.* 21, 11 (Nov. 1991), 1129–1164.

[22] L. Galavotti, F. Sebastiani, and M. Simi. 2000. Experiments on the use of feature selection and negative evidence in automated text categorization. In *Proceedings of the 4th European Conference on Research and Advanced Technology for Digital Libraries*. Springer-Verlag, London, UK, 59–68.

[23] C. Gorg, L. Zhicheng, K. Jaeyeon, Ch. Jaegul, P. Haesun, and J. Stasko. 2013. Combining computational analyses and interactive visualization for document exploration and sensemaking in jigsaw. *IEEE Trans. Visual. Comput. Graph.* 19, 10 (Oct 2013), 1646–1663.

[24] D. Greene and P. Cunningham. 2006. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proceedings of the 23rd International Conference on Machine Learning*. ACM Press, 377–384.

[25] F. Heimerl, S. Lohmann, S. Lange, and T. Ertl. 2014. Word cloud explorer: Text analytics based on word clouds. In *Proceedings of the 47th Hawaii International Conference on System Sciences (HICSS'14)*. 1833–1842.

[26] E. Hoque and G. Carenini. 2016. Interactive topic modeling for exploring asynchronous online conversations: Design and evaluation of *ConVisIT*. *ACM Trans. Interact. Intell. Syst.* 6, 1, Article 7 (Feb. 2016), 24 pages.

[27] X. Hu, L. Bradel, D. Maiti, L. House, C. North, and S. Leman. 2013. Semantics of directly manipulating spatializations. *IEEE Trans. Visual. Comput. Graph.* 19, 12 (Dec 2013), 2052–2059.

[28] Y. Hu, B. Boyd-Graber, and J. Satinoff, and A. Smith. 2014. Interactive topic modeling. *Mach. Learn.* 95, 3 (2014), 423–469.

[29] Y. Hu, E. Milios, and J. Blustein. 2011. Interactive feature selection for document clustering. In *Proceedings of the 2011 ACM Symposium on Applied Computing (SAC'11)*. ACM, New York, 1143–1150.

[30] Y. Hu, E. Milios, and J. Blustein. 2012. Enhancing semi-supervised document clustering with feature supervision. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing (SAC'12)*. ACM, New York, 929–936.

[31] Y. Hu, E. Milios, and J. Blustein. 2012. Semi-supervised document clustering with dual supervision through seeding. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing (SAC'12)*. ACM, New York, NY, 144–151.

[32] Y. Hu, E. Milios, J. Blustein, and S. Liu. 2012. Personalized document clustering with dual supervision. In *Proceedings of the 2012 ACM Symposium on Document Engineering (DocEng'12)*. ACM, New York, 161–170.

[33] J. Kogan, C. Nicholas, and V. Volkovich. 2003. Text mining with information-theoretic clustering. *Comput. Sci. Eng.* 5, 6 (Nov. 2003), 52–59.

[34] B. Larsen and Ch. Aone. 1999. Fast and effective text mining using linear-time document clustering. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'99)*. ACM, New York, NY, 16–22.

[35] H. Lee, J. Kihm, J. Choo, J. Stasko, and H. Park. 2012. iVisClustering: An interactive visual document clustering via topic modeling. *Comput. Graph. Forum* 31, 3pt3 (2012), 1155–1164.

[36] L. V. D. Maaten and G. E. Hinton. 2008. Visualizing high-dimensional data using t-SNE. *J. Mach. Learn. Res.* 9 (2008), 2579–2605.

[37] C. D. Manning, P. Raghavan, and H. Schütze. 2008. Flat clustering. In *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, 253–287.

[38] Y. Matsuo and M. Ishizuka. 2004. Keyword extraction from a single document using word co-occurrence statistical information. *Int. J. Artific. Intell. Tools* 13, 01 (2004), 157–169.

[39] J. L. Neto, A. D. Santos, C. A. A. Kaestner, and A. A. Freitas. 2000. Document clustering and text summarization. In *Proceedings of the 4th International Conference Practical Applications of Knowledge Discovery and Data Mining (PADD'00)*.

[40] S. Nourashrafeddin. 2014. *Interactive Term Supervised Text Document Clustering*. Ph.D. Dissertation. Faculty of Computer Science, Dalhousie University, Retrieved from https://dalspace.library.dal.ca/handle/10222/55965.

[41] S. Nourashrafeddin, E. Milios, and D. V. Arnold. 2013. An evolutionary algorithm for feature selective double clustering of text documents. In *Proceedings of the 2013 IEEE Congress on Evolutionary Computation (CEC'13)*. IEEE, 446–453.

[42] S. Nourashrafeddin, E. Milios, and D. V. Arnold. 2014. An ensemble approach for text document clustering using wikipedia concepts. In *Proceedings of the 2014 ACM Symposium on Document Engineering (DocEng'14)*. ACM, New York, 107–116.

[43] S. N. Nourashrafeddin, E. Milios, and D. V. Arnold. 2013. Interactive text document clustering using feature labeling. In *Proceedings of the 2013 ACM Symposium on Document Engineering (DocEng'13)*. ACM, New York, 61–70.

[44] F. V. Paulovich, L. G. Nonato, R. Minghim, and H. Levkowitz. 2008. Least square projection: A fast high precision multidimensional projection technique and its application to document mapping. *IEEE Trans. Visual. Comput. Graph.* 14, 3 (2008), 564–575.

[45] M. F. Porter. 1980. An algorithm for suffix stripping. *Progr.: Electron. Libr. Info. Syst.* 14, 3 (1980), 130–137.

[46] A. A. Puretskiy, G. L. Shutt, and M. W. Berry. 2010. Survey of text visualization techniques. *Text Mining: Applications and Theory* (2010), 105–127.

[47] D. Ramage, D. Hall, R. Nallapati, and Ch. D. Manning. 2009. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 (EMNLP'09)*. Association for Computational Linguistics, Stroudsburg, PA, 248–256.

[48] R. G. Rossi, R. M. Maracini, and S. O. Rezende. 2014. Analysis of domain independent statistical keyword extraction methods for incremental clustering. *Learn. Nonlin. Models* 12, 1 (2014), 17–37.

[49] P. J. Rousseeuw. 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.* 20 (1987), 53–65.

[50] W. Tang, H. Xiong, Sh. Zhong, and J. Wu. 2007. Enhancing semi-supervised clustering: A feature projection perspective. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'07)*. ACM, New York, 707–716.

[51] O. Turetken and R. Sharda. 2005. Clustering-based visual interfaces for presentation of web search results: An empirical investigation. *Info. Syst. Front.* 7, 3 (2005), 273–297.

[52] H. Xu, Zh. Li, Sh. Guo, and K. Chen. 2012. CloudVista: Interactive and economical visual cluster analysis for big data in the cloud. *Proc. VLDB Endow.* 5, 12 (2012), 1886–1889.

[53] Q. You, Sh. Fang, and P. Ebright. 2010. Iterative visual clustering for unstructured text mining. In *Proceedings of the International Symposium on Biocomputing*. ACM, New York, 26.

Received October 2016; revised June 2017; accepted December 2017