Text Document Topical Recursive Clustering and Automatic Labeling of a Hierarchy of Document Clusters

Xiaoxiao Li[1], Jiyang Chen[2], and Osmar Zaiane[1]

[1] Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada
{xl16, zaiane}@cs.ualberta.ca
[2] Google Canada, Kitchener, Ontario, Canada
jiyang@google.com

Abstract. The overwhelming amount of textual documents available nowadays highlights the need for information organization and discovery. Effectively organizing documents into a hierarchy of topics and subtopics makes it easier for users to browse the documents. This paper borrows community mining from social network analysis to generate a hierarchy of topically coherent document clusters. It focuses on giving the document clusters descriptive labels. We propose to use betweenness centrality measure in networks of co-occurring terms to label the clusters. We also incorporate keyphrase extraction and automatic titling in cluster labeling. The results show that the cluster labeling method utilizing KEA to extract keyphrases from the documents generates the best labels overall comparing to other methods and baselines.

Keywords: Text Mining and Web Mining; Cluster Labeling; Document Clustering

1 Introduction

In this information-explosion era, the retrieval and representation of information is vital for people′s information needs. For textual documents, two main types of information needs are: (1) finding a specific piece of information and (2) browsing the topics and structure of a given document collection [6].

Search engines are effective information retrieval tools for finding specific information. Most search engines return a long list of ranked results to users in response to a query. This presentation works well when the query is non-ambiguous and straight-forward. However, about 16% of user queries are estimated to be Ambiguous Queries, that is to say, they have multiple meanings [10]. For example, the query "jaguar" could mean "jaguar the car", "jaguar the animal" or "jaguar Mac OS" etc. There are even more queries that are Broad Queries that have multiple aspects [10]. In these situations, documents on different aspects of the query, and even irrelevant documents are mixed together. Even an experienced user would waste time and energy in sifting through the long list of

results to locate the ones that they need. The second kind of information need is to browse a document collection without a well-defined goal of searching [6]. A user may just want to browse the structure and topics of a certain document collection [5]. For example, a reader may want to know what topics a blog web site covers to see whether it is of interest; an executive may want to monitor the company emails to have an overview of the subjects discussed. For this need, the long list of documents is not effective either.

One of the solutions to the above problems is document clustering and labeling. This procedure aims at clustering a document collection into smaller groups where each group is on a different topic. It can be done recursively until the topics are specific enough. This will generate a hierarchy of document clusters with labels. This representation allows users to effectively zoom in and locate the documents of interest. It has been proved to facilitate the searching and browsing process [1]. This paper borrows Community Mining from Social Network Analysis to discover different topical coherent document groups and gives each document cluster descriptive labels. Our experiments have shown that our method have strong disambiguation ability and the labeling method utilizing a keyphrase extraction tool KEA gives overall good labels for document clusters.

2 Related Work

2.1 Attempts in improving the ranked list

Three major methods to help users to focus on the partition of documents that they may be interested in are: query refinement recommendation, pre-retrieval classification and post-retrieval clustering.

Popular search engines such as Google, Yahoo! and Bing give query refinement recommendations in the form of "Related Searches" besides the search results. Shortcomings of this method are: 1. it utilizes user query logs which may not be available on all document collections; 2. the recommendations do not have a hierarchical structure; 3. it does not group similar topics; and 4. query senses that are not as popular are left out. Classification can also bring documents into order. It classifies each document into one of the pre-defined classes. The categories are well-defined and distinctive in an ontology. However, due to its manual nature, such an ontology covers only a limited number of topics and it is expensive to build and maintain [23].

Another way to solve this problem is by document clustering and labeling, also known as Automatic Taxonomy Generation (ATG) [23]. Researchers have used document clustering to re-organize and represent retrieved documents and observed superior results than ranked lists [5, 23]. Document clustering attempts to group documents of the same topic together. The clusters are then labeled with labels that indicate their topics. A user can browse the clusters and select the topic of interest and be led to relevant documents. ATG can generate the taxonomy fully automatically with no external knowledge. Some commercial systems that use ATG to represent web search results are yippy.com and car-

rotsearch.com. In this paper, we focus on using ATG to generate a hierarchy of topics to improve the presentation of a ranked list of documents.

2.2 A review of ATG approaches

In this section we briefly review three major ATG categories including document-based, word-based, and co-clustering based methods.

Document-based ATG methods represent each document as an N-dimensional vector of features with the Vector Space Model (VSM). A feature is a wordphrase called a term and the value can be the document frequency. Conventional clustering methods can be used to cluster documents[23]. Some examples are Scatter/Gather [5], STC (Suffix Tree Clustering) [26], and SnakeT [8]. These methods use snippets that are short parts of the documents to cut down running time but snippets do not contain all the information and it is hard to get good labels from them. Our method can work on full texts with reasonable running time.

Word-based ATG methods such as the subsumption algorithm [20], DisCover [14] and J-Walker that uses a concept ontology WordNet [4] aim at organizing words by theasural relationships [12]. They first generate a concept hierarchy where each concept is a single feature, and then assign documents to the concepts. Labels generated by these methods may not be meaningful by general users and one feature is not enough to conclude the topics in the clusters.

Co-clustering based ATG methods select terms from the documents as keywords, cluster the keywords, and at the same time generate document clusters. FCoDoK [13] and FSKWIC [9] represent keywords as M-dimensional vectors and group keywords with similar document distribution together. Dhillon developed an co-clustering ATG algorithms based on bipartite graph partitioning [7]. Chen et al. proposed a method that builds a keyword graph based on the document co-occurrences of the keywords [2]. They use the K Nearest Neighbor (KNN) algorithm to find keyword clusters and then form document clusters by their similarity with each keyword cluster but they do not have statistical analysis on cluster labeling. Scaiella et al. use a Wikipedia annotator TAGME to find the Wikipedia page titles associated with each document snippet [21]. In their keyword graph, a node is a Wikipedia page title (*topic*), the edge weights are the topic-to-topic similarities computed based on the Wikipedia linked-structure. Then they bi-section the keyword graph into clusters. This method only works on snippets and TAGME can introduce errors to the graph.

3 Methodology

The basic idea of our approach is to reformulate the document clustering problem into a topical community mining problem. Rather than clustering the documents, we extract keywords from them to build a graph indicating the sentence co-occurrences of the keywords. We do community mining on this graph to get communities of highly co-occurring keywords. Then, we map the documents back to the keyword communities to form document clusters. Our approach belongs to

the category of co-clustering. Our method maintains the important information while avoiding problems caused by the high-dimensionality in document-based ATG methods. By choosing labels from a group of keywords we are able to describe a cluster with multiple aspects while word-based ATG methods only generate one feature for each cluster.

The process of our approach is shown in Fig 1. The user sends a query to a search engine and sees a taxonomy of the query senses and subtopics along with the documents. Our method applies to any document collection with mixed topics besides search results. We discuss the major phases below.

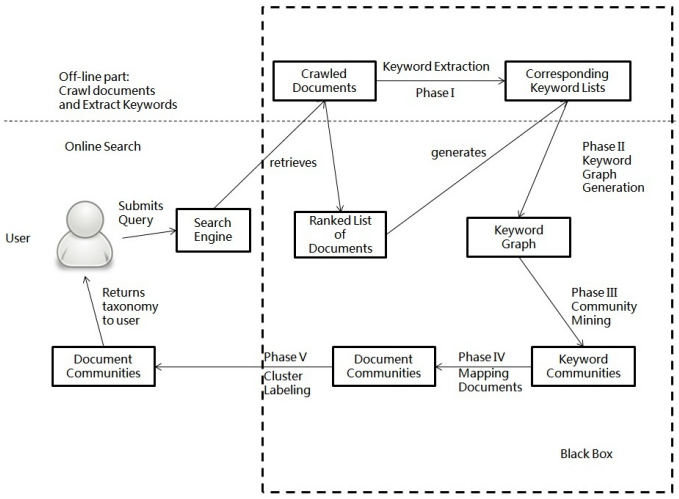

Fig. 1. General procedure of our approach

3.1 Phase I: Keyword Extraction

We first extract keywords from each document. This step is usually time-consuming and can be done off-line along with crawling. We choose Noun Phrases as keywords because they are grammatically consistent and meaningful to users [18]. We first do Part of Speech (POS) Tagging to tag each single word from the document with its part-of-speech, then we lemmatize all the words to reduce the inflectional forms. The next step is pruning where we convert the first word in a sentence to lower case, and remove stop words and words that contain non-alphabetic characters. Finally, we extract Noun Phrases based on a lexical heuristic (Adjective).*(Noun).+. We consider a word or phrase with zero or more Adjectives with one or more Nouns following them as a Noun Phrase [15].

3.2 Phase II: Keyword Graph Generation

We use the keyword pair lists corresponding to the documents to generate a keyword graph. A node in this graph is a keyword. An edge is formed when two nodes have co-occurred in at least one sentence. The edge weight is the sentence co-occurrences. The key assumption in forming edges using co-occurrences is that words describing the same topic are often used together. Co-occurrences have been shown to carry useful correlation information and be able to identify different topical groups [2]. We select nodes based on Document Frequency (DF) to reduce noise. Only keywords with DF higher than a threshold t_{df} remain nodes in the keyword graph. Moreover, terms that are exactly the same, or are contained in the query are removed from the nodes set.

3.3 Phase III: Community Mining

We do community mining on the keyword graph to detect different topical communities. Community mining is the grouping of nodes such that nodes in the same community are more connected with each other than with nodes outside the community. We use the Fast Modularity clustering algorithm ($O(nlog^2n)$) which is based on one of the most well-known community mining metrics: Modularity Q [3]. Modularity Q measures the quality of a graph partitioning. The Fast Modularity algorithm greedily optimizes the modularity score in the graph partitioning in an agglomerative manner [3]. This algorithm automatically detects the number of communities and generates compact taxonomies. It has an advantage over existing commercial systems such as carrotsearch.com and Yippy, and also some most recent works since these methods partition the document collection to about 10 clusters which is not always the real number of topics [2][21]. While many state-of-the-art search result clustering algorithms are flat, our method applies the Fast Modularity algorithm recursively in a top-down manner until certain conditions are reached. We use a Modularity threshold t_Q to determine the need to further split the communities.We further refine the communities by deleting noisy communities and merging similar communities.

3.4 Phase IV: Mapping Documents to Keyword Communities

In this phase we assign the documents to the keyword communities to generate document clusters as illustrated in Fig 2 on some examples from the query "jaguar". A dashed line represents the connection between a document and a keyword. The solid lines are the edges in the keyword graph. For each document on the left, we calculate its overall TFIDF score in each of the keyword communities and assign it to the keyword communities accordingly. Given a document d and a keyword community c, d's overall TFIDF score in c is the sum of the TFIDF scores of all the keywords that are both in d and in c. We assign d to the community that has the highest overall TFIDF score s. Besides, since a document may have multiple topics, we assign d to another community c' as well if its overall TFIDF score in c' is higher than $0.9 * s$.

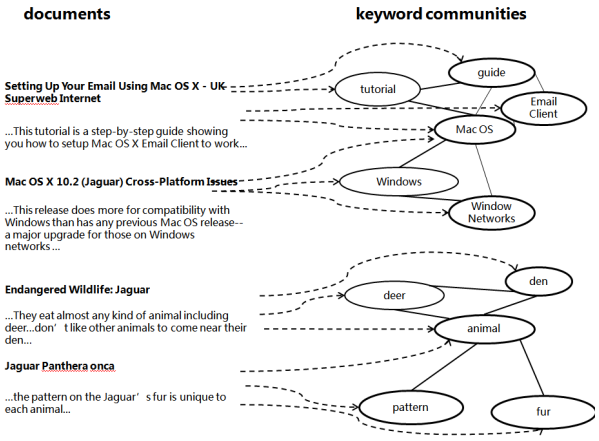

Fig. 2. Illustration of Phase IV

3.5 Phase V: Cluster Labeling

We treat document cluster labeling as a ranking problem of the keywords in each community. The most common cluster labeling method is to use the most frequent or central phrases in a document cluster as labels [17]. In our experiments we use the Degree Centrality labeling method as a baseline BL1. We use the "Frequent and Predict Words" method that detects terms that are more likely to appear in a cluster than in other clusters as labels as another baseline BL2 [19]. We come up with four labeling methods (LM1, LM2, LM3, LM4) that select labels based on the keyword cluster, the document cluster, and the connection between the co-clusters.

LM1 finds important terms from the keyword communities as labels based on the betweenness centrality that reflects a node′s influence on the communications between other nodes in the community. Betweenness centrality measures the number of shortest paths between other nodes that goes through a certain node. The intuition of using betweenness centrality for labeling is that sometimes terms of the same topic may not directly co-occur in a sentence and may be connected by terms that play a vital role in connecting terms.

We also try to select labels based on the document clusters. Keyphrases and titles both introduce the topics of the documents. We propose to incorporate a famous and effective keyphrase extraction algorithm KEA (LM2) [24] and an automatic titling method (LM3) [16] to identify important terms from the documents. We use an updated KEA tool[3] to extract keyphrases from the documents. LM3 extracts title words from each document by an automatic titling method. We take the Noun Phrases (NP) from the first two sentences and the titles from the documents as title words. The terms are ranked by the number of documents where they serve as important terms. The top 20 are cluster labels.

[3] http://www.nzdl.org/Kea/description.html

The last labeling method, LM4, looks at the connections between a keyword community and its corresponding document community (the dashes lines in Fig 2). It ranks the terms by the sum of their TF-IDF scores in each document cluster. The top 20 forms a label list.

After getting the label lists, we do post-processing based on lemmas, abbreviations, synonyms and hypernyms to make them more readable. The top five labels in the post-processed list is the final label list for each cluster.

4 Experiments and Discussions

4.1 Data collection and pre-processing

We constructed our data sets using Google. For each query, we searched for some of its senses in Google and gathered the top results. We merged these pages together as the document collection under the query. We experimented with a multitude of queries with ambiguous meanings. For illustration purposes, we show some examples here. A list of queries, their query senses and the subtopics along with the size of the document collections is shown in Table 1. For example, the *tiger* data set is merged by the search results of *tiger aircraft*, *tiger woods*, *tiger animal*, and *tiger hash*. For the query sense *jaguar car* in the *jaguar* data set, we selected two subtopics: *jaguar car history* and *jaguar car dealer*. We feed them to Google search engine and merged their results as the documents of the query sense *jaguar car*.

Table 1. List of queries, query senses, subtopics of query senses (shown in parentheses) with the number of documents

Query	Query Senses and subtopics	Document Set Size
jaguar	animal (animal facts, animal rescue), car (car history, car dealer), Mac OS, guitar	180
penguin	Pittsburgh hockey team, publisher, kids club, algorithm	150
avp	Volleyball, antivirus software, Avon, movie, airport	150
tiger	Aircraft, Woods, animal, hash	120
michael jordan	basketball player (career, quotes), Berkeley researcher	90

4.2 Evaluation Metrics

We compare our results with the ground truth gained from Google to evaluate the clustering performance. We adapt two evaluation metrics: ARI [25] and Cluster Contamination (CC) [6]. ARI measures how close the clusters generated by our system (P) matches the the ground truth (R). CC measures the purity of the clusters. The clustering is good if it has a high ARI and a low CC. We conducted a user survey with 11 volunteers to obtain the labeling ground truth. The label with most votes is the ground truth label S of a document cluster. Given S and its parent label P, a system label L is a correct label if L is identical with S, S P,

or $P\ S$ [22]. Four evaluation metrics are used, namely match@N, P@N, MRR@N and MTRR@N [14, 22]. N is the number of labels presented to the users. Good labels have high metric scores and small N.

4.3 Experimental results

In our experiments, we set $t_{df} = 0.04$, and $t_Q = 0.3$ (details are omitted due to lack of space). We evaluate the top levels and the lower level clusters in the taxonomy separately. The top levels contain clusters that are directly under the root. They reflect the disambiguation ability of our method. The lower levels show how well our method discovers different aspects (subtopics) of the same topic. While all the levels are important in the browsing process, the top levels carry more responsibility because they are the ones that users see first.

Document Clustering Performances We compare our method with an effective variation of K-Means [11] to examine the document clustering quality. We use the number of clusters found by our method as the parameter k, the keywords extracted by our method as the features and the TFIDF scores as the feature values for K-means. The clustering performance on the top levels is listed in Table 2. Our method gets higher ARI score on all queries and less contamination on all but one queries than K-Means. We found that K-Means tend to generate one big and highly contaminated cluster with several small and pure clusters. Our method does not generate highly polluted clusters thus is more desirable for browsing. Besides, K-Means requires the number of clusters k in advance whereas our method automatically detects k. Overall, our clustering method outperforms K-Means on the top levels. Table 3 shows the clustering performance on the lower levels. Our method has higher ARI scores on 2 out of 3 query senses but it generates more contaminated clusters on lower levels. It is maybe due to the fact that there is no clear separation between the vocabularies used by different subtopics of the same topic.

Table 2. ARI and average CC score of our method and K-means on the top levels

Query	ARI score		average CC score	
	our method	K-Means	our method	K-means
jaguar	0.968	0.521	0.053	0.169
penguin	0.842	0.319	0.152	0.352
avp	0.802	0.615	0.239	0.178
tiger	0.771	0.325	0.193	0.261
michael jordan	1	0.022	0	0.439

Cluster Labeling Performances The cluster labeling performances on the four evaluation metrics of our four methods and two baselines on the top levels are presented in Fig 3 with N ranging from 1 to 5. We can see that LM2 which

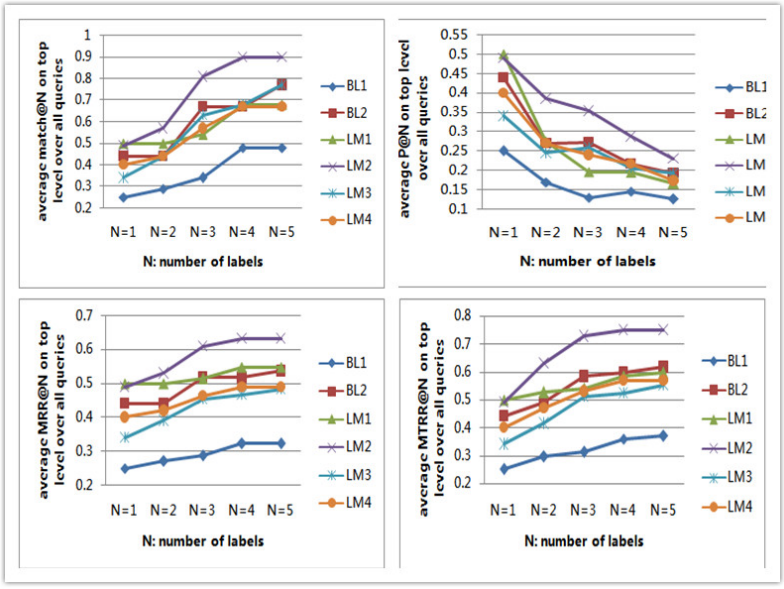

Fig. 3. Average match@N, P@N, MRR@N and MTRR@N on the top level over all queries of different labeling methods

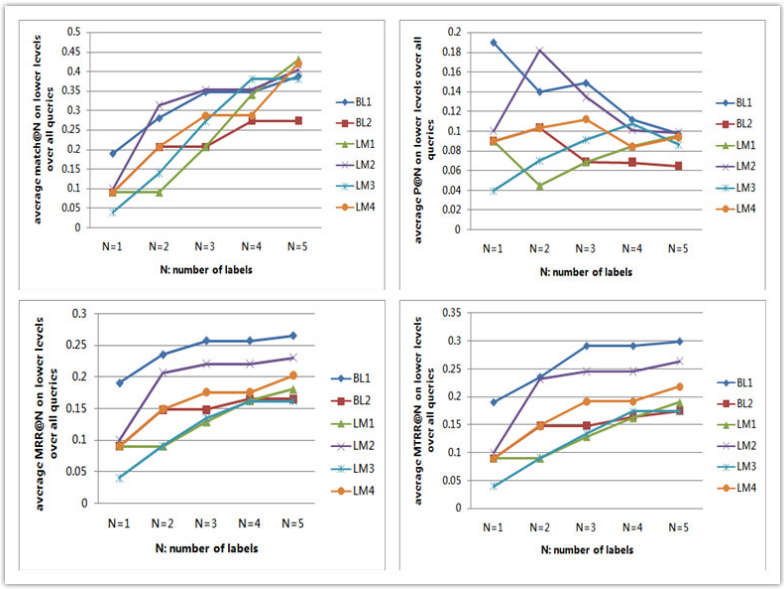

Fig. 4. Average match@N, P@N, MRR@N and MTRR@N on lower levels over all queries of different labeling methods

Table 3. ARI and average CC score of our method and K-means on lower levels

Query sense	subtopics	ARI score		CC score	
		our method	K-Means	our method	K-means
jaguar/animal	facts, rescue	0.428	0.321	0.288	0.261
jaguar/car	history, dealer	0.122	-0.0003	0.512	0.400
Michael Jordan/basketball player	career, quotes	0.107	0.328	0.658	0.550

is the labeling method utilizing KEA achieves the highest average score over all queries on all the metrics. In Table 4 we show the top 5 labels picked by LM2 for each query sense. Beside each label is the number of users who chose it as the cluster label in the user survey. We also show the ground truth labels, each with the number of users who have picked it. The labeling performances on the lower levels are shown in Fig 4. We can see that BL1, which is the worst method on the top levels, is among the best methods on the lower levels. LM2 which is the best on the top levels is the second best on the lower levels. Overall, LM2 is the best labeling method both on the top and the lower levels. Note that the metric scores on the lower levels are less than those of the top levels. In our user survey we have found that even for humans it is harder to agree on the labels of the lower levels than of the top levels. One reason is that the subtopics are difficult to differentiate. Another reason might be that similar terms are used when covering subtopics.

5 Conclusions and Future Work

In this work we use a co-clustering ATG method based on the co-occurrences of frequent keywords to generate a taxonomy for a document collection that performs well in disambiguating topics but not as well in separating subtopics of the same topic. We propose four different labeling methods and found that the labeling method utilizing KEA generates the best overall cluster labels. Future works include ways to improve the performance on the lower levels. One possibility is to explore other ways to build the keyword graph so that it is sensitive to different subtopics. Another future work is to combine different labeling methods to improve the labeling performance by reflecting the strength of each method.

References

1. R. Berendsen, B. Kovachev, E. Nastou, M. de Rijke, and W. Weerkamp. Result disambiguation in web people search. In *ECIR 2012: 34th European Conference on Information Retrieval*, Barcelona, 2012.
2. S.Y. Chen, C.N. Chang, Y.H. Nien, and H.R. Ke. Concept extraction and clustering for search result organization and virtual community construction. *Computer Science and Information Systems*, 9(1):323–355, 2012.
3. A. Clauset, M.E.J. Newman, and C. Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.
4. H. Cui and OR Zaiane. Hierarchical structural approach to improving the browsability of web search engine results. In *Database and Expert Systems Applications, 2001. Proceedings. 12th International Workshop on*, pages 956–960. IEEE, 2001.

Table 4. Top cluster labels by the users, and by our KEA method for each query sense with the number of users who picked each label

Query	Query Sense	Ground Truth Labels	Our labels by LM2
jaguar	animal	animal-7	animals (7), cat (1), habitats (1), species (1), leopard (0)
	car	jaguar car-7	cars (3), jaguar car (7), dealer (1), history (1), sport car (3)
	Mac OS	mac os-7	OS (5), mac (3), mac OS (7), apples (2), window (0)
	guitar	guitar-9	guitars (9), fenders (2), pickup (0), neck (0), bridges (0)
penguin	Pittsburgh hockey team	pittsburgh penguin-7	pittsburgh (3), pittsburgh penguin (7) , teams (1), hockey (5), league (1)
	publisher	book-6 publisher-6	books (6), publishers (6), penguin book (3), imprints (1), penguin group(2)
	kids club	club penguin-7	clubs (1), club penguin (7), kid (1), games (0), online (0)
	algorithm	google penguin algorithm-6	google (3), algorithms (3), updates (2), google penguin (3), algorithm update (2)
AVP	volleyball	volleyball-8	volleyball (8), beaches (0), tours (1), beach volleyball (6), sport (2)
	antivirus software	antivirus-8	kaspersky (2), viruses (2), software (5), antivirus (8), kaspersky lab (1)
	Avon	avon product-7	avon (5), avon product (7), product (2), market (0), stock (3)
	movie	movie-6	predators (2), movies (6), alien (2), weyland (0), predator movie (3)
	airport	international airport-4	airports (1), scranton (1), international airport (4), avoca (2), hotel (1)
tiger	aircraft	tiger airway-8	aircraft (7), pilot (0), tiger moth (2), tiger airway (8), markets (0)
	Woods	golf-6	woods (1), tiger wood (1), golf (6), opens (1), tournament (1)
	animal	animal-8	animal (8), cubs (0), habitat (0), species (2), cat (0)
	hash	tiger hash algorithm-7	hash (4), tiger hash (6), hash function (6), function (1), file (0)
Michael Jordan	basketball player	basketball-7	basketball (7), player (1), NBA[national basketball association] (5), basketball player (5), games (0)
	Berkeley researcher	machine learning-11	research (3), university (3), machine learning (11), learning (1), berkeley (1)

5. Douglass R. Cutting, David R. Karger, Jan O. Pedersen, and John W. Tukey. Scatter/gather: a cluster-based approach to browsing large document collections. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '92, pages 318–329, New York, NY, USA, 1992. ACM.
6. W. Dawid. *Descriptive Clustering as a Method for Exploring Text Collections*. PhD thesis, Poznan University of Technology, Poznań, Poland, 2006.
7. I.S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 269–274. ACM, 2001.
8. P. Ferragina and A. Gulli. A personalized search engine based on web-snippet hierarchical clustering. *Software: Practice and Experience*, 38(2):189–225, 2008.
9. H. Frigui and O. Nasraoui. Simultaneous categorization of text documents and identification of cluster-dependent keywords. In *Fuzzy Systems, 2002. FUZZ-IEEE'02. Proceedings of the 2002 IEEE International Conference on*, volume 2, pages 1108–1113. IEEE, 2002.

10. Bernard J. Jansen, Danielle L. Booth, and Amanda Spink. Determining the user intent of web search engine queries. In *Proceedings of the 16th international conference on World Wide Web*, pages 1149–1150, New York, NY, USA, 2007. ACM.
11. T. Kanungo, D.M. Mount, N.S. Netanyahu, C.D. Piatko, R. Silverman, and A.Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *Pattern Analysis and Machine Intelligence*, 24(7):881–892, 2002.
12. R. Krishnapuram and K. Kummamuru. Automatic taxonomy generation: Issues and possibilities. *Fuzzy Sets and SystemsIFSA 2003*, pages 184–184, 2003.
13. K. Kummamuru, A. Dhawale, and R. Krishnapuram. Fuzzy co-clustering of documents and keywords. In *Fuzzy Systems, The 12th IEEE International Conference on*, volume 2, pages 772–777. IEEE, 2003.
14. K. Kummamuru, R. Lotlikar, S. Roy, K. Singal, and R. Krishnapuram. A hierarchical monothetic document clustering algorithm for summarization and browsing search results. In *Proceedings of the 13th international conference on World Wide Web*, pages 658–665. ACM, 2004.
15. Z. Liu, W. Huang, Y. Zheng, and M. Sun. Automatic keyphrase extraction via topic decomposition. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 366–376. Association for Computational Linguistics, 2010.
16. C. Lopez, V. Prince, and M. Roche. Automatic titling of electronic documents with noun phrase extraction. In *Soft Computing and Pattern Recognition (SoCPaR)*, pages 168–171. IEEE, 2010.
17. C.D. Manning, P. Raghavan, and H. Schutze. *Introduction to information retrieval*, volume 1. Cambridge University Press Cambridge, 2008.
18. Q. Mei, X. Shen, and C.X. Zhai. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 490–499. ACM, 2007.
19. A. Popescul and L.H. Ungar. Automatic labeling of document clusters. *Unpublished manuscript*, 2000.
20. M. Sanderson and B. Croft. Deriving concept hierarchies from text. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 206–213. ACM, 1999.
21. U. Scaiella, P. Ferragina, A. Marino, and M. Ciaramita. Topical clustering of search results. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 223–232. ACM, 2012.
22. P. Treeratpituk and J. Callan. Automatically labeling hierarchical clusters. In *Proceedings of the 2006 international conference on Digital government research*, pages 167–176. ACM, 2006.
23. X. Wang and M. Bramer. Exploring web search results clustering. *Research and Development in Intelligent Systems XXIII*, pages 393–397, 2007.
24. I.H. Witten, G.W. Paynter, E. Frank, C. Gutwin, and C.G. Nevill-Manning. Kea: Practical automatic keyphrase extraction. In *Proceedings of the fourth ACM conference on Digital libraries*, pages 254–255. ACM, 1999.
25. K.Y. Yip, D.W. Cheung, and M.K. Ng. Harp: A practical projected clustering algorithm. *Knowledge and Data Engineering, IEEE Transactions on*, 16(11):1387–1397, 2004.
26. Oren Zamir and Oren Etzioni. Grouper: a dynamic clustering interface to Web search results. In *Proceedings of the eighth international conference on World Wide Web*, WWW '99, pages 1361–1374, New York, NY, USA, 1999. Elsevier North-Holland, Inc.