Università
Ca'Foscari
Venezia

Scuola Dottorale di Ateneo
Graduate School

Dottorato di ricerca
in Informatica
Ciclo XXIX
Anno di discussione 2016

Improving the quality of text clustering and cluster labeling

SETTORE SCIENTIFICO DISCIPLINARE DI AFFERENZA: INF/01
Tesi di Dottorato di Mohsen Pourvali, matricola 956115

Coordinatore del Dottorato	**Tutore del Dottorando**
Prof. Riccardo Focardi	**Prof. Salvatore Orlando**

Università Ca' Foscari di Venezia

Dipartimento di Scienze Ambientali, Informatica e Statistica
Dottorato di Ricerca in Informatica

Ph.D. Thesis: Cycle 29

Improving the Quality of Text Clustering and Cluster Labeling

Mohsen Pourvali

Supervisor

Salvatore Orlando

PhD Coordinator

Riccardo Focardi

September 2016

Author's Web Page: www.unive.it/persone/mohsen.pourvali

Author's e-mail: mohsen.pourvali@unive.it

Author's address:

Dipartimento di Informatica
Università Ca' Foscari di Venezia
Via Torino, 155
30172 Venezia Mestre - Italia
tel. +39 041 2348411
fax. +39 041 2348419
web: http://www.dsi.unive.it

To my beloved wife, Hosna

Abstract

The abundance of available electronic information is rapidly increasing with the advancements in digital processing. Furthermore, huge amounts of textual data have given rise to the need for efficient techniques that can organize the data in manageable forms. In order to tackle this challenge, clustering algorithms try to group automatically similar documents. While clustering plays a significant role that helps to categorize documents, it owes intrinsic limits when it comes to allowing human users to understand the content of documents at a deeper level. This is where cluster labeling techniques come into the scene. The goal of cluster labeling is to label - i.e., describe in an informative way - clusters of documents according to their content. Document clustering and cluster labeling are two vital problems in the information retrieval domain because of their ability to organize increasing amount of texts and describe such the huge amount in a concise way. In this thesis, we have addressed these problems in four parts.

In the first part, we investigate how we can improve the effectiveness of text clustering by summarizing some documents in a corpus, specifically the ones that are much significantly longer than the mean. The contribution in this part is twofold. First, we show that text summarization can improve the performance of classical text clustering algorithms, in particular, by reducing noise coming from long documents that can negatively affect clustering results. Moreover, we show that the clustering quality can be used to quantitatively evaluate different summarization methods.

In the second part, we explore a multi-strategy technique that aims at enriching documents for improving clustering quality. Specifically, we use a combination of entity linking and document summarization, to determine the identity of the most salient entities mentioned in texts. We further investigate ensemble clustering in order to combine multiple clustering results, generated based on the combination of the specific set of features, into a single result of better quality.

In the third part, we investigate the problem of cluster labeling whose quality obviously depends on the quality of document clustering. To this end, we first explore and categorize cluster labeling techniques, providing a thorough discussion of the relevant state-of-the-art literature.

In the fourth part, we then present a fusion-based topic modeling approach to enrich documents' vectors of corpus with the aim of improving the quality of text clustering. We further exploit such vectors through a fusion method for cluster labeling.

Finally, we experimentally prove the effectiveness of our solutions, explained in four parts, in the clustering and cluster labeling problems with various datasets.

Acknowledgments

First and foremost I wish to express my sincere gratitude to my supervisor, Salvatore Orlando, for the continuous support of my Ph.D study, for his patience and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

Besides my supervisor, I would like to thank my reviewers, Nicola Ferro and Alfredo Cuzzocrea, for their deep analysis of my thesis and providing useful comments and suggestions.

Thanks to the Department of Computer Science of the University CaFoscari of Venice for providing a scientific environment and financially supporting my study with a three years grant. Thanks to my colleagues who helped me in any way. I'm also grateful to Venice, because of her beauty that made my mind beautiful.

Special thanks to my beloved wife, Hosna, for the support and care she had provided me during these three long years. Last but not the least, I would like to thank my parents and to my brothers and sister for their never ending love, and affection to me.

Contents

List of Figures

List of Tables

List of Algorithms

1

Introduction

1.1 Motivation

The clustering problem is defined as the process of grouping similar objects into the same cluster, where the similarity between the objects is measured using a similarity function. While we are overwhelming by ever increasing electronic information, the applicative studies on the clustering problem can be very useful in the text domain.

The abundance of available electronic information is rapidly increasing with the advancements in digital processing. Furthermore, huge amounts of textual data have given rise to the need for efficient techniques that can organize the data in manageable forms. To this end, clustering algorithms are a common method to organize huge corpora of textual digital documents, like the ones nowadays available.

The importance of the Text clustering problem is due to the fact of its applicability in the various text domains [2], expanded from Document Organization and Browsing, to Corpus Summarization. Despite of inherent unsupervised learning property of text clustering, it further can be leveraged in order to improve the quality of the results in its supervised variant.

While clustering algorithms represent significant tools that help to categorize documents, adding another step to help users apprehend content of clusters would be more efficient. To this end, cluster labeling techniques aim to describe content of the clusters of documents in an understandable way for users. To achieve this goal, they provide some kind of labels, i.e., textual entities summing up the properties of a cluster.

This thesis aims at improving the performance of classical text clustering algorithms, particularly, by investigating the effectiveness of various state-of-the-art techniques which are applied through novel approaches.

1.2 Content of the Thesis

The content of the thesis that comprises different proposed approaches, dealing with the problem of improving the quality of clustering and cluster labeling, is addressed in four parts.

In the first part, we show the effectiveness of text summarization on improving the quality of text clustering, which is obtained by reducing noise coming from long documents. In text clustering, a text or document is always represented as a bag of words. This representation raises one severe problem: the high dimensionality of the feature space and the inherent data sparsity. Obviously, a single document has a sparse vector over the set of all terms [40]. The performance of clustering algorithms will decline dramatically due to the problems of high dimensionality and data sparseness [1]. Therefore, it is highly desirable to reduce the feature space dimensionality. To this end, we propose a novel method in which n-tsets (i.e., non-contiguous sets of n terms that co-occur in a sentence) are extracted through a graph-based approach. Indeed, the proposed summarization method is a keyphrase extraction-based summarization method in which the goal is to select individual words or phrases to tag a document.

In the second part, we explore the effect of Linked Entities in improving the quality of the text clustering problem. In traditional text clustering, the vector-based representations of texts are purely based on terms occurring in documents. Other information, in particular, latent ones, should be included in the document representation to make more significant document similarities. In this part, we consider that latent information could be defined in two ways: information hidden in some important words of the text document, in particular, in the text fragments mentioning the *most salient entities* [65] linked to a *knowledge base* like Wikipedia, as well as in additional information coming from common semantic concepts based on a lexical database like WordNet.

We have investigated the utility of common latent information hidden among those words that represent linked entities, but not all the linked entities. Indeed, we have applied the Graph-based Ranking summarization algorithm represented in the first part to create a summary based on the main topic of each document, and then we extract only those linked entities of the document that appear in the summary. Such the linked entities which are more relevant to the main topic of the document are known as *salient entities*. Moreover, we utilize WordNet to expand salient entities with ontology-based latent information. We also utilize graph partitioning approach with two aims: discarding irrelevant terms to not be expanded, and applying a clustering ensemble approach. Our experiments show that using only salient entities can significantly improve the quality of classical text clustering algorithms rather than using all the entities.

In the third part, we first investigate the problem of cluster labeling which quality of its results is closely related to the document clustering problem. We then present a fusion- and topic-based approach to enrich documents' vectors of corpus with the aim of improving the quality of text clustering and cluster labeling, consequently. While clustering techniques represent an important tool to categorize documents to give them a better characterization, they possess intrinsic limits when it comes to give a deeper understanding of the documents content to human users. This is where cluster labeling techniques come into the scene.

Cluster labeling techniques aim to better characterize groups (clusters) of documents according to their specific content, and they try to achieve this goal by assigning some kind of labels, i.e., textual entities summing up the properties of a cluster. Different kinds of cluster labeling methods exist, depending on the approach used to infer cluster labels. Mainly, we can speak about two classes of methods: *direct* methods and *indirect* methods. Direct methods try to extract labels directly from the content of documents making up a cluster, and indirect methods use external resources (e.g., Wikipedia) to assign labels to clusters. Our proposed approach for labeling clusters is classified in the direct methods.

In the fourth part, we investigate the effectiveness of topic modeling in the quality of clustering and cluster labeling. Topic modeling algorithms are statistical methods, which are able to find the themes (topics) running through the text documents by analyzing their words. Using topic models in machine learning and text mining is popular due to its applicability. In this part of my thesis, we present a novel approach to improve the quality of clustering using topic models [5] and fusion methods [77]. The core idea of our approach is to enrich the vectors of the documents to be exposed as the representation vector in clustering. To this end, we apply a statistical approach to discover and annotate a corpus with thematic information represented in form of different proportions over different topics for each document. Furthermore, for each cluster, we use such the vectorial representation of documents through a fusion method to label the cluster.

1.3 Related Work

Since we have proposed different approaches coming from different domains, we explore related work of each approach separately. Considering *entity linking* (EL), which we have used for document enriching, there is another method different from EL, used in other works to enrich text document corpora to be clustered, namely *named entity recognition* (NER). NER is to identify the mention of a named entity in text and its type, but without identifying the specific entity. For example, NED identifies that “Rome”, “Tehran”, and “Paris” are mentions of capitals city, but does not disambiguate and link to the specific identifier of the entity. As another example, “U.K”, “United Kingdom”, and “Britain” are countries, but NED cannot unify them by linking the spots to the same unique entity.

Specifically, in text clustering domain, the exploiting of common latent information - which are stated above - during the process of clustering is different from one work to another. In some contributions, they take the latent information of documents into account by considering only the attributes of the named entities [8, 54, 18, 53, 22]. The authors in [8] propose an entity-keyword multi-vector space model that represent a document by a vector on keywords -which are the words of original documents using in traditional VSM model- and four vectors on named entity features (i.e. entity names, types, name-type pairs, and identifiers). The main idea in this work is to generate a trade-off between named entity features and traditional vector space model depending on the importance of entities and keywords among the collection. Besides, there are contributions in which the authors propose to exploit an ontology of common concepts like WordNet rather than on named entities [29, 81, 64, 61].

The proposed clustering approach in [61] is based on two aims; first reducing the high dimensionality of vector space represented by each document, and second taking into account the relationships between terms like synonyms antonyms. For this purpose, the approach uses WordNet lexical categories to map each document words to lexical categories, and then use WordNet ontology *hypernym* (or *hyponym*) to create new classes of similar concepts with the aim of finally reducing documents-words matrix to documents-classes matrix.

The common idea behind the above approaches is that they try to expand the latent information which is hidden among the terms of a document in order to improve somewhat in quality of text clustering, and the obtained results by these approaches indicate such improvement. Intuitively, if we want to cluster a collection of documents based on their *contents*, the aim of clustering may be defined to group those documents which their *main topics*, being discussed in each one, are in common. However, each document contains several topics, for each of which there are relevant terms in documents [5]. Therefore, not all the terms appearing in a document have the same relevance and utility in understanding the main topic being discussed. Indeed, expanding latent information exploiting the terms which are relevant to the main topic of document is more efficient in finding similar documents rather than expanding all terms of included topics, which may contrariwise cause increasing noises coming from irrelevant information.

In case of labeling a cluster which is another part of my thesis, we have above stated two main classes of cluster labeler (direct and indirect). Some of the related work in direct cluster labeling using different feature selection methods [43], picking up the most frequent terms occurring in a cluster, or using top weighted cluster centroid's terms [16] to extract candidate labels. The main drawback of these methods consists in that they may not produce an optimal solution whenever meaningful labels cannot be extracted from the documents making up a cluster. For example, let us consider a cluster of documents discussing about *printmaking*: by looking just at the content of individual documents, it is possible that the set of labels extracted do not contain the topic to which the documents belong; For example, for a cluster with the set of candidate labels *engraving, etching, lithography, steel engraving* selecting *printmaking*, which can be extracted from an external ontology, as the label of the cluster would be more meaningful rather than selecting one of the represented candidates.

In order to tackle this issue, indirect cluster labeling methods consider the usage of external resources (e.g., Wikipedia) to assign labels to clusters [9, 69]. Indeed , the hypothesis behind these approaches is that the describing labels for a cluster could be provided through an external resources. These resources may be an encyclopedia like Wikipedia or a lexical ontology like WordNet. In the above example, *printmaking* can be extracted by using the semantical relationships, which are exist in the lexical ontology of WordNet. But, indirect approaches indeed require a direct approach to provide candidate labels to be expanded by an external resources as well. We discuss in detail different cluster labeling approaches in both classes as a unique survey in Chapter 5.

To investigate related work of using topic models in document clustering, there is another approach in which a topic model could be directly used to map the original high-dimensional representation of documents (word features) to a low dimensional representation (topic features) and then applies a standard clustering algorithm like k-means in the new feature space [41]. It is also possible to consider each topic as a cluster and documents with highest proportion of same topic are located in the same cluster. Lu et al. in [41] investigated performance of two probabilistic topic models PLSA and LDA in document clustering. Authors used the topic models to generate specific topics which are treated each one as a cluster. Therefore, for clustering, the documents are clustered into the topic with the highest probability. In similar way, [78] aims to elaborate on the ability of further other topic modeling algorithms CTM, Hierarchical LDA, and HDP to cluster documents.

We highlight two main problems here: first, we do not know the exact number of topics running through the corpus, besides, because of frequency-based nature of topic models, we cannot claim the topic with the highest probability for a document is the main topic by which the documents must be clustered. Therefore, these two problems are considered as our hypothesis in dealing with topics running through the corpus in this thesis.

2

Preliminary

In this chapter we introduce some definitions and notations, as well as the measures and benchmarks used for evaluating proposed approach.

2.1 Text Summarization

To define text document summarization, we quote a definition introduced by Radev et al. in [60] as follows: “a text that is produced from one or more texts, that conveys important information in the original text(s), and that is no longer than half of the original text(s) and usually significantly less than that”. Specifically, the aim of text summarization is to create a condense version of a text document or a collection of text documents in which most important topics of original document(s) should remain in it.

Generally, we can categorize text summarization techniques into two groups (types); *extractive* and *abstractive*. In extractive summarization, a summary consist of information unites extracted from the original text, producing them verbatim. On the other hand, in abstractive summarization, a summary may contain synthesized information units that may not necessarily occur in the text document, aiming to produce a grammatical summary, which usually requires advanced language generation techniques. Most existing summarization methods are extractive. There are two general areas of research in text summarization, including *single document summarization* and *multi document summarization*.

2.1.1 Single document summarization

In single document summarization, we deal with a single document, containing some important parts and some less important parts. The problem here is to distinguish these parts from each other.

2.1.2 Multi document summarization

Multi document summarization is to extract a single summary from multiple documents. This departs from single-document summarization since the problem involves multiple sources of information that overlap and supplement each other, being contradictory at occasions. So the key tasks are not only identifying and coping with redundancy across documents, but also recognizing novelty and ensuring that the final summary is both coherent and complete [17].

2.2 WordNet

WordNet is a large lexical database designed for use under program control. English nouns, verbs, adjectives, and adverbs are organized into sets of synonyms (synsets), each representing a lexicalized concept. Semantic relations link the synonym sets [51]. Simply, WordNet is a thesaurus in which it groups words together based on their meanings. In WordNet, words that denote the same concept (synonyms) and are interchangeable in many contexts are grouped into unordered sets (*synsets*). Therefore, a word related to n synsets in WordNet has n possible senses. These senses may cover multiple part of speech; for example, if a word appears in 8 synsets, it might have 5 noun senses, 3 verb senses, and an adjective sense. Additionally, for each synset in

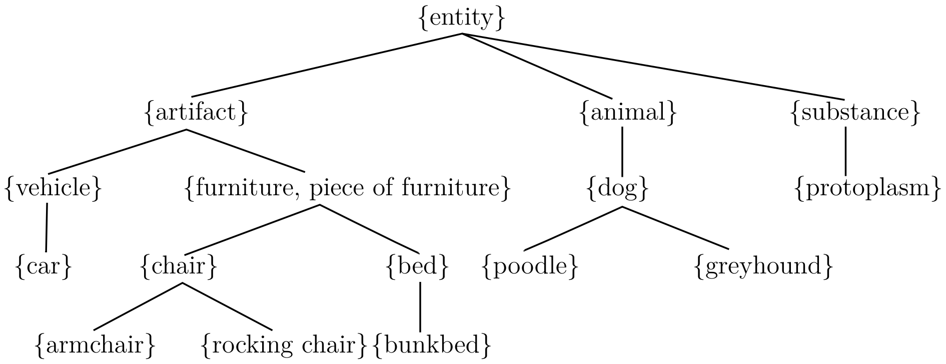


Figure 2-1: A “kind-of” noun hierarchy in WordNet (excerpt) obtained from [19].

WordNet, there is a brief definition (*gloss*), in which the use of the synset members is illustrated by one or more short sentences.

Generally, WordNet includes several semantic relations, i.e. *Synonymy*, *Antonymy*, *Hyponymy*, *Meronymy*, *Troponymy*, *Entailment*, in which the most important relation among synsets is the super-subordinate relation (also called *hyperonymy*, *hyponymy*, or *ISA* relation). Another important relation is *meronymy*, also called the part-whole, or part-name relation. hyponym relation links more general synsets like *furniture* to increasingly specific ones like *bed*. Therefore, it makes hierarchical predecessor/successor concepts that users can navigate within and across the resulting hierarchies in either direction. These hierarchical structures sometimes called “trees” [19], and sometimes because there may be more than one path to reach a node in this structure, they called rooted DAGs (Direct Acyclic Graphs). In both structures, considering only noun word forms, all nodes ultimately go up to the root node which is *entity*. Figure 1 shows a part of hyponymy relation (*is a kind of*) in WordNet. It also indicates transitive property of this relation, for example, a *poodle* is a kind of *dog*, a dog is a kind of *canid*, a canid is a kind of *carnivore*, and after three other nodes we have *chordate* is a kind of animal.

We used a prolog format of WordNet provided by the Princeton University, in that each WordNet relation is represented in a separate file by operator name. Some operators are reflexive (i.e. the "reverse" relation is implicit). So, for example, if x is a hypernym of y, y is necessarily a hyponym of x. In the prolog database, reflected pointers are usually implied for semantic relations. Semantic relations are represented by a pair of synset_ids, in which the first synset_id is generally the source of the relation and the second is the target. If two pairs synset_id,w_num are present, the operator represents a lexical relation between word forms.

More specifically, in this work we only use noun senses of words.

2.3 Topic Models

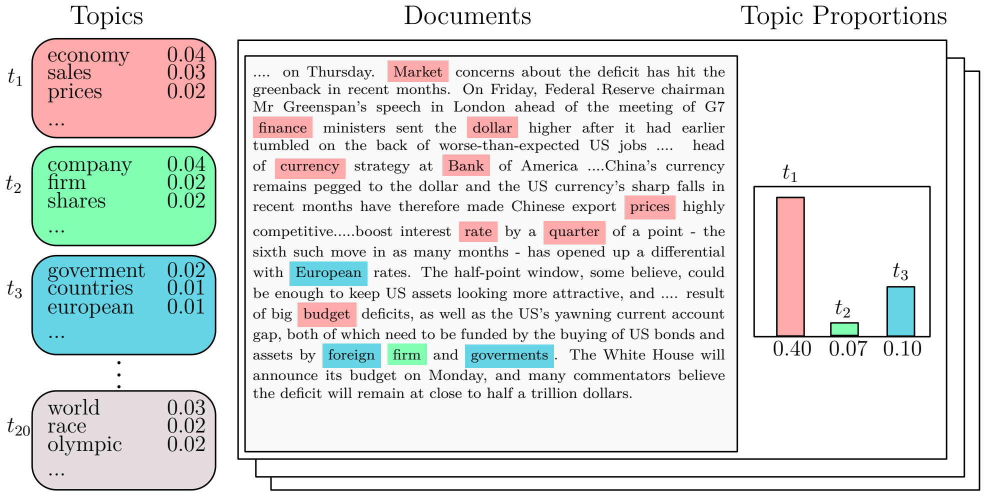
Topic models are based on the idea that documents are created by a mixture of topic, where a topic is a probability distribution over words. Indeed, a topic model is a statistical model by which we can create all the documents of a collection. Topic modelling is a popular framework in machine learning and text mining [28, 80, 67, 67, 4, 45, 47, 37, 76] due to its applicability; extraction of scientific research topics [26, 4], opinion extraction [45], multi document summarization [27], sentiment analysis [71], image labeling [20], predicting visit location [34], and social network analysis [12].

Assume that we want to fill up every document of a corpus with the words, topic model says each document contains multiple topics and exhibits the topics in different proportion. Thus, for each document, there is a distribution over topics, which according to this distribution a topic is chosen for every word of that document, and then from that topic (i.e. distribution over vocabulary) a word is drawn [5]. Figure 2-2 shows three topics and their proportions of 20 topics which is derived from BBC news articles by running topic modeling MALLET[1](i.e. natural language processing toolkit). At right of the figure, proportions of the three topics (with highest probability) within the specified document are shown, and at left of the figure three words of the topics that have the highest probability are shown.

2.3.1 Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) is a topic model widely used in the information retrieval field. Specifically, LDA is a probabilistic model that says each document of a corpus is generated by a distribution over topics, and each topic is characterized by a distribution over words. Figure 2-3 shows a graphical model of LDA. The process

[1] http://mallet.cs.umass.edu/



.... on Thursday. Market concerns about the deficit has hit the greenback in recent months. On Friday, Federal Reserve chairman Mr Greenspan's speech in London ahead of the meeting of G7 finance ministers sent the dollar higher after it had earlier tumbled on the back of worse-than-expected US jobs .... head of currency strategy at Bank of America ....China's currency remains pegged to the dollar and the US currency's sharp falls in recent months have therefore made Chinese export prices highly competitive.....boost interest rate by a quarter of a point - the sixth such move in as many months - has opened up a differential with European rates. The half-point window, some believe, could be enough to keep US assets looking more attractive, and .... result of big budget deficits, as well as the US's yawning current account gap, both of which need to be funded by the buying of US bonds and assets by foreign firm and goverments. The White House will announce its budget on Monday, and many commentators believe the deficit will remain at close to half a trillion dollars.

Figure 2-2: An example of topic modeling, running by MALLET toolkit with 20 topics. The shown document is a portion of the second document from cluster *business* in the BBC news articles.

of generating a document defines a *joint probability distribution* over two observed (i.e. words of corpus) and hidden (i.e. topics) random variables. The data analysis is perform by using that joint distribution to compute the conditional distribution of the hidden variables given the observed variables. Formally, LDA is described as follows:

$$\begin{aligned} &p(\beta_{1:K}, \theta_{1:D}, z_{1:D}, w_{1:D}) \\ &\qquad = \prod_{i=1}^{K} p(\beta_i) \prod_{d=1}^{D} p(\theta_d) \left(\textstyle\prod_{n=1}^{N} p(z_{d,n} \mid \theta_d) p(w_{d,n} \mid \beta_{1:K}, z_{d,n}) \right) \end{aligned} \tag{2.1}$$

Where $\beta_{1:K}$ are topics where each β_k is a distribution over words of the corpus (i.e. vocabulary), θ_d are topic proportions for the dth document, z_d are the topic assignments for the dth document where $z_{d,n}$ is the topic assignment for the nth word in document d, which specifies the topic that nth word in d belongs to, and w_d are the observed words for document d where $w_{d,n}$ is the nth word in document d.

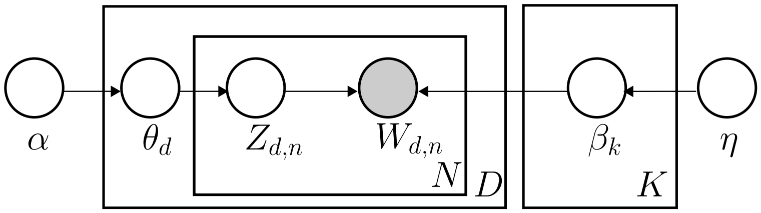


Figure 2-3: The graphical model for latent Dirichlet allocation. Adopted from [5]. Each node is a random variable and is labeled according to its role in the generative process. The hidden nodes are unshaded and the observed nodes are shaded. The rectangles are "plate" notation, which denotes replication. The N plate denotes the collection of words within documents and the D plate denotes the collection of documents within the collection. The K plate denotes the collection of topics.

2.4 Fusion methods

We now introduce two baseline state-of-the-art data fusion methods, frequently used for various information retrieval tasks, namely the CombSUM and CombMNZ fusion methods [77].

Suppose there are n ranked lits which are created by n different systems over a collection of items D. Each system S_i provides a ranked list of items $L_i =< d_{i1}, d_{i2}, ..., d_{im} >$, and a relevance score $s_i(d_{ij})$ is assigned to each of the items in the list. Data fusion technique is to use some algorithms to merge these n ranked lists into one [77].

CombSum uses the following equation:

$$g(d) = \sum_{i=1}^{n} s_i(d) \tag{2.2}$$

If d does not appear in any L_i, a default score (e.g., 0) is assigned to it. According to the global score $g(d)$ all the items can be ranked as a new list.

Another method CombMNZ uses the equation:

$$g(d) = m \times \sum_{i=1}^{n} s_i(d) \tag{2.3}$$

Where m is the number of lists in which item d appears.

The linear combination (i.e. general form of CombSum) uses the equation:

$$g(d) = \sum_{i=1}^{n} w_i \times s_i(d) \tag{2.4}$$

Where w_i is the weight assigned to system S_i.

2.5 HITS

HITS (Hyperlinked Induced Topic Search) [32] is an iterative algorithm that was designed for ranking Web pages. HITS makes a distinction between "authorities" (pages with a large number of incoming links) and "hubs" (pages with a large number of outgoing links). Hence, for each vertex V_i, HITS produces an "authority" and a "hub" score:

$$HITS_A(V_i) = \sum_{V_j \in In(V_i)} HITS_H(V_j) \tag{2.5}$$

$$HITS_H(V_i) = \sum_{V_j \in Out(V_i)} HITS_A(V_j) \tag{2.6}$$

2.6 Experimental Setups

The principal idea of the experiments in this thesis is to show the efficacy of proposed approaches on clustering and cluster labeling results through a manually predefined categorization of the corpus. In addition, in order to evaluate the absolute quality of our proposed methods, we further need a standard dataset to compare our methods with the baselines. We used four following well known benchmarks to test the benefits of the proposed methods.

2.6.1 Benchmarks

The three corpora used in the experiments are described in the following:

Classic4: This dataset is often used as a benchmark for clustering and co-clustering [2]. It consists of 7095 documents classified into four classes denoted MED, CISI, CRAN and CACM.

BBC NEWS: This dataset consists of 2225 documents from the BBC news website corresponding to stories in five topical areas, which are named Business, Entertainment, Politics, Sport and Tech, from 2004-2005 [25]. BBC News is a dataset full of linking entities, and specially is convenient in splitting its documents into their paragraphs.

DUC2002: DUC 2002 contains 567 document-summary pairs which are clustered into 59 topics, and each topic contains about 10 documents. For each topic there are 7 summaries namely *10*, *50*, *100*, *200*, *200e*, *400e*, and *perdocs* which are written by an expert or more. The summary 10 for example is a 10-word summary of all the documents included in a topic which is written by a human, similarly summaries 50, 100 and 200 are created by different sizes. The summaries 200e and 400e are created by extracting important sentences from the documents of each topic. Unlike other summaries, sentences of 200e and 400e summaries grammatically are as the same as the sentences of original documents. The last summary for each topic is perdoc which is a document contains of 100-words summaries for each document of a topic separately. For our evaluation we only used summaries 10, 50, 100, 200, and perdocs.

20NG: 20 News Group (20NG) is a collection of documents manually classified into 20 different categories that each one contains about 1000 documents.

2.6.2 Preprocessing

Preprocessing is an essential step in text mining. In this section we introduce three commonly used preprocessing in text mining domain:

stop words removal

Stop words are words which are commonly used in a language (such as “the”, “at”, etc.). These words do not convey an important information about text, and often ignored by search engines and other natural language processing tools in order to save both space and time. There is no single universal list of stop words used by all

[2] http://www.dataminingresearch.com/index.php/2010/09/classic3-classic4-datasets/

natural language processing tools, and indeed any group of words can be chosen as the stop words for a given purpose. In our experiments, we use those words as stop words which are common in most of natural language processing tools.

stemming

Stemming is the process of reducing derived words to their word *stem* form. A stem could be morphological root of the word, or a stem by which all the related words are identified as the same words. For example, a stemming algorithm should reduce words "working", "worked", and "worker" to the root word,"work". On the other hand, it should reduce words "argue", "argued", "argues", and "arguing" to the stem, "argu". Using such the algorithm benefit us to map all the words which are different because of grammatical reasons, or are derivationally related together to a single stem. We use Porter[3] in our experiments which is the most common algorithm for stemming English.

lower case conversion

Lower case conversion is simply to convert keywords with capital letters to keywords with small letters, which indeed maps all the words that appeared in different shapes, for example because of their position in sentence, to a single word.

In this thesis, we take advantage of the above-mentioned preprocessing firstly to save space and time, and further to consider keywords which are purely related to the main topic of a text document.

2.6.3 Evaluation Measures

For our experiments, we use three well known measures in order to evaluating results of clustering, summarization, and cluster labeling.

Purity

For evaluating the clustering results, we used *Purity* measure. The purity is a simple and transparent evaluation measure which is related to the entropy concept [66]. To

[3] http://tartarus.org/martin/PorterStemmer/

compute the purity criterion, each cluster P is assigned to its majority class. Then we consider the percentage of correctly assigned documents, given the set of documents L_i in the majority class:

$$Precision(P, L_i) = \frac{|P \bigcap L_i|}{|P|} \tag{2.7}$$

The final purity of the overall clustering is defined as follows:

$$Purity(\mathbb{P}, \mathbb{L}) = \sum_{P_j \in \mathbb{P}} \frac{|P_j|}{N} \arg\max_{L_i \in \mathbb{L}} Precision(P_j, L_i) \tag{2.8}$$

where N is the number of all documents, $\mathbb{P} = \{P_1, P_2, ..., P_k\}$ is the set of clusters and $\mathbb{L} = \{L_1, L_2, ..., L_c\}$ is the set of classes.

ROUGE

For evaluating the summarization results, we used ROUGE measure. ROUGE is a method based on N-gram statistics, found to be highly correlated with human evaluations [38]. The ROUGE-N is based on n-grams and generates three scores *Recall*, *Precision*, and the usual *F-measure* for each evaluation.

$$R_n = \frac{\sum\limits_{S \in \{Ref\}} \sum\limits_{\text{n-gram} \in S} Count_{match}(\text{n-gram})}{\sum\limits_{S \in \{Ref\}} \sum\limits_{\text{n-gram} \in S} Count(\text{n-gram})} \tag{2.9}$$

$$P_n = \frac{\sum\limits_{S \in \{Cand\}} \sum\limits_{\text{n-gram} \in S} Count_{clip}(\text{n-gram})}{\sum\limits_{S \in \{Cand\}} \sum\limits_{\text{n-gram} \in S} Count(\text{n-gram})} \tag{2.10}$$

$$F = \frac{2 \times P_n \times R_n}{P_n + R_n} \tag{2.11}$$

R_n (recall) counts the number of overlapping n-gram pairs between the candidate summary to be evaluated and the reference summary created by humans (See [38] for more details). P_n (precision) measures how well a candidate summary overlaps with multiple human summaries using n-gram co-occurrence statistics (See [58] for more details). We used two of the ROUGE metrics in the experimental results, ROUGE-1 (unigram) and ROUGE-2 (bigram).

Match@N and MRR@N

For evaluating the quality of cluster labeling, we use the frameworks represented in [73]. Therefore, for each given cluster, its ground truth labels where obtained by manual (human) labeling are used for the evaluation.

We use **Match@N** (Match at top N results) and **MRR@N** (Mean Reciprocal Rank) measures proposed in [73] to evaluate the quality of the labels. They consider the categories of ODP as the correct labels and then evaluate a ranked list of proposed labels by using following criteria:

- Match@N: It is a binary indicator, and returns 1 if the top N proposed labels contain at least one correct label. Otherwise it returns zero.
- MRR@N: It returns the inverse of the rank of the first correct label in the top-N list. Otherwise it returns zero.

A proposed label for a given cluster is considered correct if it is identical, an inflection, or a Wordnet synonym of the cluster's correct label [9].

3

Improving clustering quality by automatic text summarization

Automatic text summarization is the process of reducing the size of a text document, to create a summary that retains the most important points of the original document. It can thus be applied to summarize the original document by decreasing the importance or removing part of the content. In this chapter we show that text summarization can improve the performance of classical text clustering algorithms, in particular by reducing noise coming from long documents that can negatively affect clustering results. Moreover, we show that the clustering quality can be used to quantitatively evaluate different summarization methods. In this regards, we propose a new graph-based summarization technique for keyphrase extraction, and use various datasets to evaluate the improvement in clustering quality obtained using text summarization.

Our method could be considered as one for unsupervised feature selection, because it chooses a subset from the original feature set, and consequently reduces vector space for each document. In particular, as mentioned above, it is particular effective when applied to longer documents, since these documents reduce purity of clustering. To this end, we propose a novel method in which *n-tsets* (i.e., non-contiguous sets of n terms that co-occur in a sentence) are extracted through a graph-based approach. Indeed, the proposed summarization method is a keyphrase extraction-based summarization method in which the goal is to select individual words or phrases to tag a document. We have utilized HITS algorithm [32], which is designed for web page ranking, in order to boost the chance of a node to be selected as a keyphrase of the document, although other graph-based algorithms have been proposed to summarize texts, For example, we can mention [48] in which sentences, instead of key-phrases, are extracted through undirected graphs.

3.1 Baseline Graph-Based Keyphrase Extraction (HITS)

In this section we discuss the *baseline* used for testing our proposed text summarization method (Chapter 3). We start from this method because it is a simple form of graph-based ranking approach. In addition, we exploit it to boost the score of keyphrases to include in a text summary.

This graph-based method relies on HITS to rank terms. A similar idea can be applied to lexical or semantic graphs which are extracted from text documents in order to identify the most significant blocks (words, phrases, sentences, etc.) for building a summary [50, 39]. Specifically, we applied HITS to directed graphs whose vertexes are terms, and edges represent co-occurrences of terms in a sentence. Before generating the graph, stopword removal and stemming are applied. Once computed the $HITS_A(V_i)$ and $HITS_H(V_i)$ scores for each vertex V_i of the graph, we can rank the graph nodes by five simple functions of the two scores:

$$F_\Gamma(V_i) = \Gamma(HITS_A(V_i), HITS_H(V_i))$$

where Γ corresponds to different ways of combining the two HITS scores. Namely $avg/max/min/sum/prod$ (average/maximum/minimum/sum/product of the Hub and Authority scores). After the scoring of the nodes by F_Γ, we can rank them, and finally return the K-top ranked ones.

3.2 Our Summarization Technique

To create a keyphrase-based summary of a document, we devised a unsupervised technique, called N-tset Graph-based Ranking (NG-Rank) for which n-tset is a set of one or more terms co-occurring in a sentence.

In a document, the discussed subjects are presented in a specific order. For each document paragraph, the first sentence represents a general view of the discussed subject, which is examined in depth in the rest of the sentences. The rest sentences might be ended by a conclusion sentence, which is the final close of the discussed subject. In general, the first and last sentences likely include the main concepts of the document. Therefore, let D be a document of the collection, denoted by $D = (P_1, P_2, ..., P_n)$, where P_i is a paragraph of D. The sentences of P_i are thus partitioned as follows:

- First Sentences (FS): which are the first f consecutive sentences occurring of P_i.
- Middle Sentences (MS): which are the middle sentences of P_i.
- Last Sentences (LS): which are the last l consecutive sentences of P_i.

Once denoted the sentences of each paragraph, our algorithm preprocesses these sentences by removing stop words and applying the Porter stemmer. Suppose that after these processing step, the number of stemmed terms in a document is m. The next step of our algorithm builds an $m \times m$ (normalized) co-occurrence matrix $A_0 = (a_1, a_2, ..., a_m)$ of the terms. Specifically, each entry of matrix A_0 is given by $\frac{t_{ij}}{t_i}$, where t_{ij} indicates the number of times term i and term j co-occur within the various sentences of the documents, and t_i is the number of times term i occurs in the document. We can have:

$$a_{ij} = \begin{cases} 1 & \text{if } t_i = t_{ij} \quad \text{(I)} \\ < 1 & \text{otherwise} \quad \text{(II)} \end{cases}$$

In case $a_{ij} = a_{ji} = 1$ and $t_{ij} > 1$, then the terms i and j always co-occur for the same number of times within the various sentences of the documents. Then we merge them as a new *n-tset* term, and rebuild the matrix, by merging the i^{th} and j^{th} rows (columns). This process is iterated, namely $A_{h+1} = merge(A_h)$, till $\nexists i, j$ such that $a_{ij} = a_{ji} = 1$ and $t_{ij} > 1$. The number of iteration is $I = N - 1$, where N is the biggest *n-tset* found in the document.

For example, consider a document with one paragraph, consisting of 5 sentences, partitioned into the sets FS, LS, and MS (First, Last, and Middle Sentences)[1], where the stemmed terms are represented as capital letters:

$$FS = \{(AB)\} \quad LS = \{(MSR)\} \quad MS = \{(ACDFG), (ACNDG), (MSN)\}$$

In the first iteration, terms C and D are merged as a new term C-D. In addition, also terms M and S are merged as a new term M-S. In the second iteration, terms C-D and G are merged as a new term C-D-G. Note that at the end of this iterative process, each row/column will correspond to n-tsets, $n \geq 1$. Without loss of generality, hereinafter we call "n-tset" both single and multiple terms (identified by our algorithm). The final sentences after the merging is thus:

$$FS = \{(A\ B)\} \quad LS = \{(M\text{-}S\ R)\} \quad MS = \{(A\ C\text{-}D\text{-}G\ F), (A\ C\text{-}D\text{-}G\ N), (M\text{-}S\ N)\}$$

Finally, the *primary score* for each n-tset (single or multiple terms), corresponding to a row a_i of the final matrix A_{last}, is defined as follows:

$$PScore(a_i) = \frac{1}{\sum_{j=0}^{m} a_{ij}} \tag{3.1}$$

If an n-tset appears in long sentences or appears multiple times in short sentences, its row a_i in the matrix is not so sparse, in comparison with n-tsets occurring in a few short sentences. If this property holds, this decreases the value of $PScore$.

In the next step, we use A_{last} as the adjacency matrix to generate a graph of relationships between n-tsets. Each node corresponds to an n-tset occurring in the document, and each edge models the co-occurrence of a pair of n-tsets in a sentence.

[1] $f = l = 1$, where f and l are the number of sentences in FS and LS, respectively.

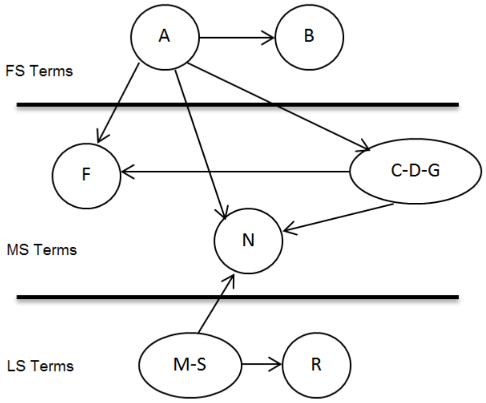


Figure 3-1: Structure of graph, the nodes are n-tsets of the document, in turn partitioned into three sets. The direction of the edges corresponds to the order in which the n-tsets appear in each sentence.

Indeed, the graph is directed. If $a_i \rightarrow a_j$, then a_i occurs before a_j in one or more sentences. The graph of n-tsets for our running example is shown in Figure 3-1. Note that the nodes of the graph are subdivided into three partitions: FS, LS, and MS. This means that each node associated with an n-tset must be univocally assigned to one partition. When the same n-tset occurs in more than one set of sentences – i.e., first, last, or middle sets of sentences – we must choose only one of the three partitions FS, LS, or and MS. Specifically, we assign the n-tset to a partition according to a priority order: we choose FS if the n-tset appears in some of the first sentences, then LS if the n-tset appears in some of the last sentences, MS otherwise.

We exploit this graph to boost the primary score assigned to some n-tsets. Since n-tsets in FS and LS are considered more discriminative than the others, we increase the primary scores of n-tsets whose associated nodes are in the FS or LS partition. In addition, we also boost the primary scores of nodes in MS that are connected to nodes in FS or LS, i.e., there exist a path that connects these nodes in MS to nodes in FS or LS partitions. Specifically, we use two boosting methods that exploit graph properties The first one simply exploits the in/out degree of each node:

$$Score(a_i) = PScore(a_i) + \log(\ \max(v_{in}(a_i), v_{out}(a_i))\ ) \qquad (3.2)$$

The second boosting method exploits function Γ = max among the HITS functions discussed in Section 3.1:

$$Score(a_i) = PScore(a_i) * (1 + \ \max(HITS_A(a_i), HITS_H(a_i)) \ ) \tag{3.3}$$

We obtained better results with $\Gamma = \max$ than with any other alternative functions Γ. It shows that the words occurred at the beginning or at the end of a paragraph are much more important candidates as keywords of the document.

It is worth noting that the nodes in MS that are not boosted maintain, however, the old primary score, i.e., $Score(a_i) = PScore(a_i)$. These nodes are still considered in the following phase.

Specifically, once all the nodes in the graph are scored by $Score(a_i)$, we rank them, and finally return the n-tsets associated with the K-top ranked ones, where the value of K depends on the length of document to be summarized. Indeed, we sort in decreasing order of $Score$ the nodes within each partition of the graph (FS, MS, or LS). After this separated reordering of each partition, we return a summary that contains the *same fraction* α, $0 < \alpha < 1$, of the top-scored n-tsets for each of the three partitions. Specifically, we return $\alpha \cdot |FS|$, $\alpha \cdot |LS|$, and $\alpha \cdot |MS|$ nodes (n-tsets) from the sets FS, MS, and LS.

Finally, the order of the terms in the generated summary is the same as the one in the original document. This step is important to evaluate the quality of the extracted summaries with respect to human-generated ones (using the DUC 2002 dataset).

Hereinafter, we call the summarization algorithm that exploits the boosting methods of Equation (3.2) NG-Rank$_\mathsf{M}$, whereas we call the ones adopting the alterative boosting method of Equation (3.3) NG-Rank$_\mathsf{H}$.

Figure 3-2 shows the created graph for following single paragraph obtained from BBC articles:

> *Breaking news about president Barack Obama. News obtained from BBC. Barack Obama heads into the home stretch of his presidency faced with a republican controlled congress.*

Figures 3-3, 3-4, 3-5, and 3-6 show created graph for following document obtained from DUC2002 dataset[2]:

[2] The graph is splitted into 4 parts shown through four figures.

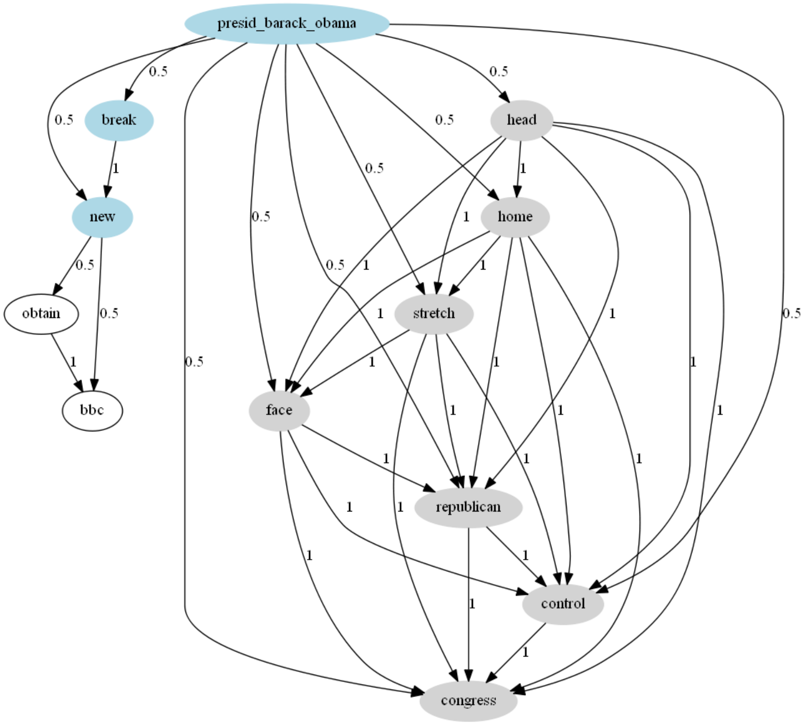


Figure 3-2: Created graph for a single paragraph of a document in BBC.

Hurricane Gilbert swept toward the Dominican Republic Sunday, and the Civil Defense alerted its heavily populated south coast to prepare for high winds, heavy rains and high seas.

The storm was approaching from the southeast with sustained winds of 75 mph gusting to 92 mph.

"There is no need for alarm," Civil Defense Director Eugenio Cabral said in a television alert shortly before midnight Saturday.

Cabral said residents of the province of Barahona should closely follow Gilbert's movement.

An estimated 100,000 people live in the province, including 70,000 in the city of Barahona, about 125 miles west of Santo Domingo.

Tropical Storm Gilbert formed in the eastern Caribbean and strengthened into a hurricane Saturday night. The National Hurricane Center in Miami reported its position at 2 a.m. Sunday at latitude 16.1 north, longitude 67.5 west, about 140 miles south of Ponce, Puerto Rico, and 200 miles southeast of Santo Domingo.

The National Weather Service in San Juan, Puerto Rico, said Gilbert was moving westward at 15 mph with a "broad area of cloudiness and heavy weather" rotating around the center of the storm.

The weather service issued a flash flood watch for Puerto Rico and the Virgin Islands until at least 6 p.m. Sunday.

Strong winds associated with the Gilbert brought coastal flooding, strong southeast winds and up to 12 feet feet to Puerto Rico's south coast. There were no reports of casualties.

San Juan, on the north coast, had heavy rains and gusts Saturday, but they subsided during the night.

On Saturday, Hurricane Florence was downgraded to a tropical storm and its remnants pushed inland from the U.S. Gulf Coast. Residents returned home, happy to find little damage from 80 mph winds and sheets of rain.

Florence, the sixth named storm of the 1988 Atlantic storm season, was the second hurricane.

The first, Debby, reached minimal hurricane strength briefly before hitting the Mexican coast last month.

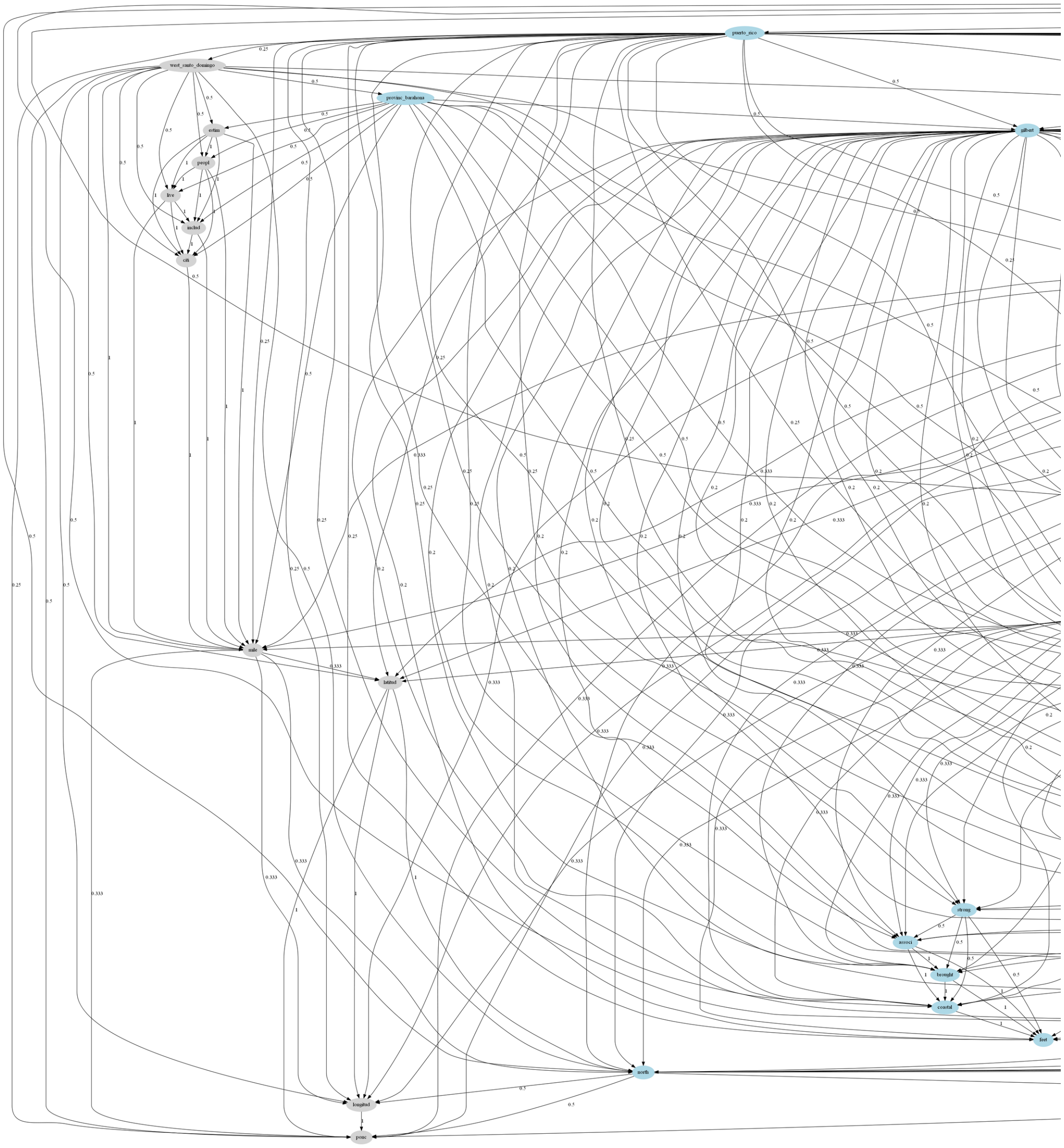


Figure 3-3: Created graph by NG-Rank method for a document in DUC2002 (part 1).

3.3 Experimental Setup

In order to evaluate the absolute quality of our summarization method, we further need a standard dataset to compare our method with the baseline. We used “Classic4” and “BBC NEWS” to test the benefits of summarization on clustering quality , and “DUC 2002” for testing the quality of our summarization method.

We have used four classes of BBC article news in our experiments. Unlike Classic4, the BBC NEWS corpus is full of names of athletes, politicians, etc. These proper names are challenging, because they could be important to be extracted as keyphrase of text. On the other hand, they could reduce the similarity between two related texts. Furthermore, We have used the 100-words summary of DUC2002 provided for each document.

Preprocessing.

In addition to preprocessing explained in Section 2.6.2, we preprocess the corpora Classic4 and BBC to generate from them two new datasets, identifying sentences and paragraphs. Specifically, since our aim is to evaluate the efficacy of summarizing longer documents to improve clustering, for each original dataset we generated a sub-collection of documents of different sizes: a large part of them approximatively contains the same number of terms sz, while the others are much longer than sz. Specifically, longer documents contain a number of terms not less than $3 \cdot sz$.

In more details, we stratified the sampling of each *original labeled dataset* as follows. Let $\mathbb{L} = \{L_1, L_2, ..., L_c\}$ be the original dataset, where L_i is the set of documents labeled with the i^{th} class. From each L_i we thus extract a subset $\mathcal{L}_i$, thus generating the sub-collection $\mathbb{D} = \{\mathcal{L}_1, \mathcal{L}_2, ..., \mathcal{L}_c\}$. Specifically, we have:

$$\mathcal{L}_i = R_i \cup E_i \tag{3.4}$$

where $R_i = \{d \in L_i \mid a \leq size(d) \leq b\}$ and $E_i = \{d \in L_i \mid size(d) \geq 3 \cdot \mathcal{M}\}$, while $\mathcal{M}$ is the average size of the documents in R_i, i.e. $\mathcal{M} = \frac{\sum_{d \in R_i} size(d)}{|R_i|}$.

The constants a and b limit the size of documents in R_i. We tested our method on different sampled sub-collections $\mathbb{D}$, using diverse a and b. The results obtained are similar.

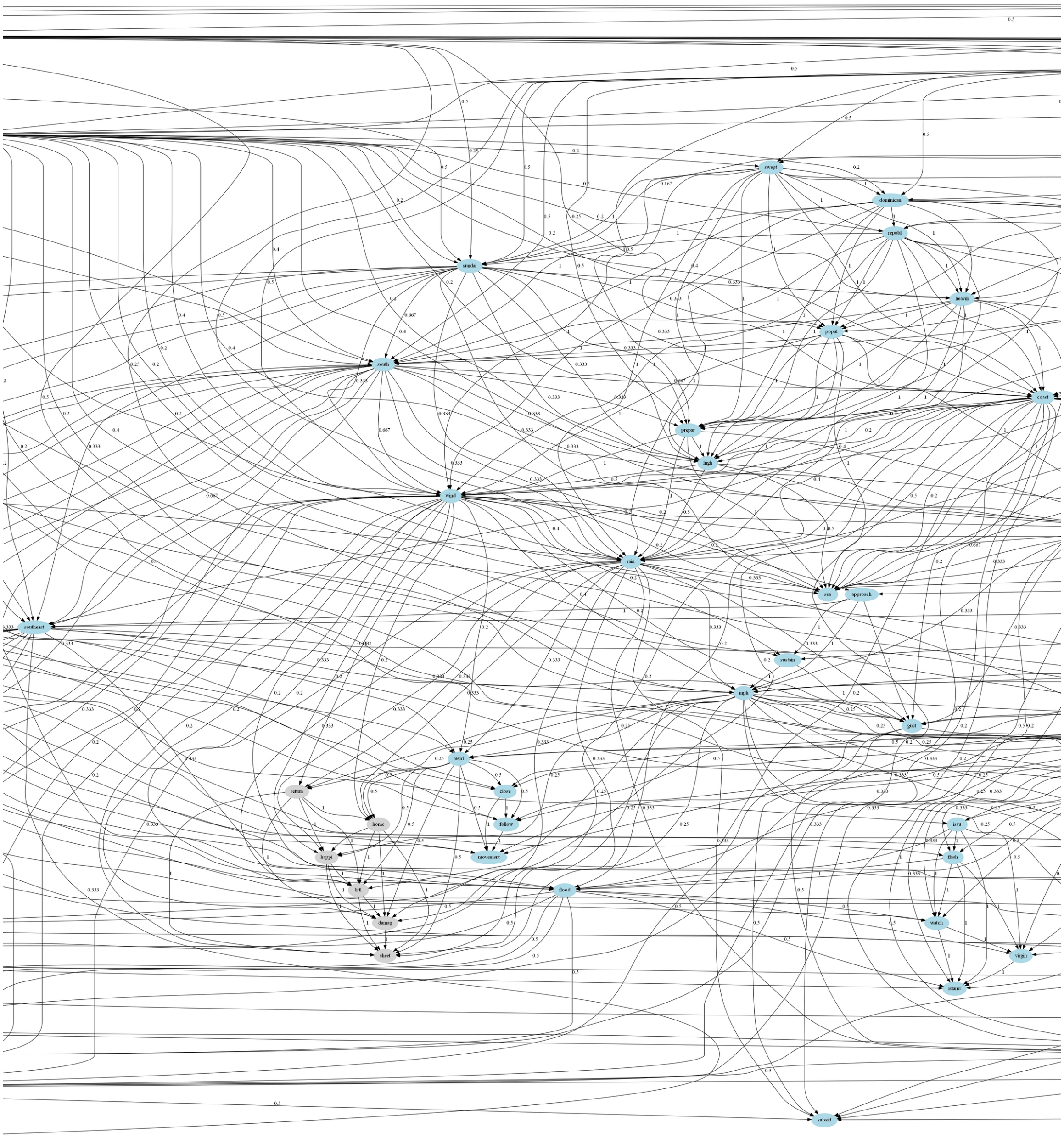


Figure 3-4: Created graph by NG-Rank method for a document in DUC2002 (part 2).

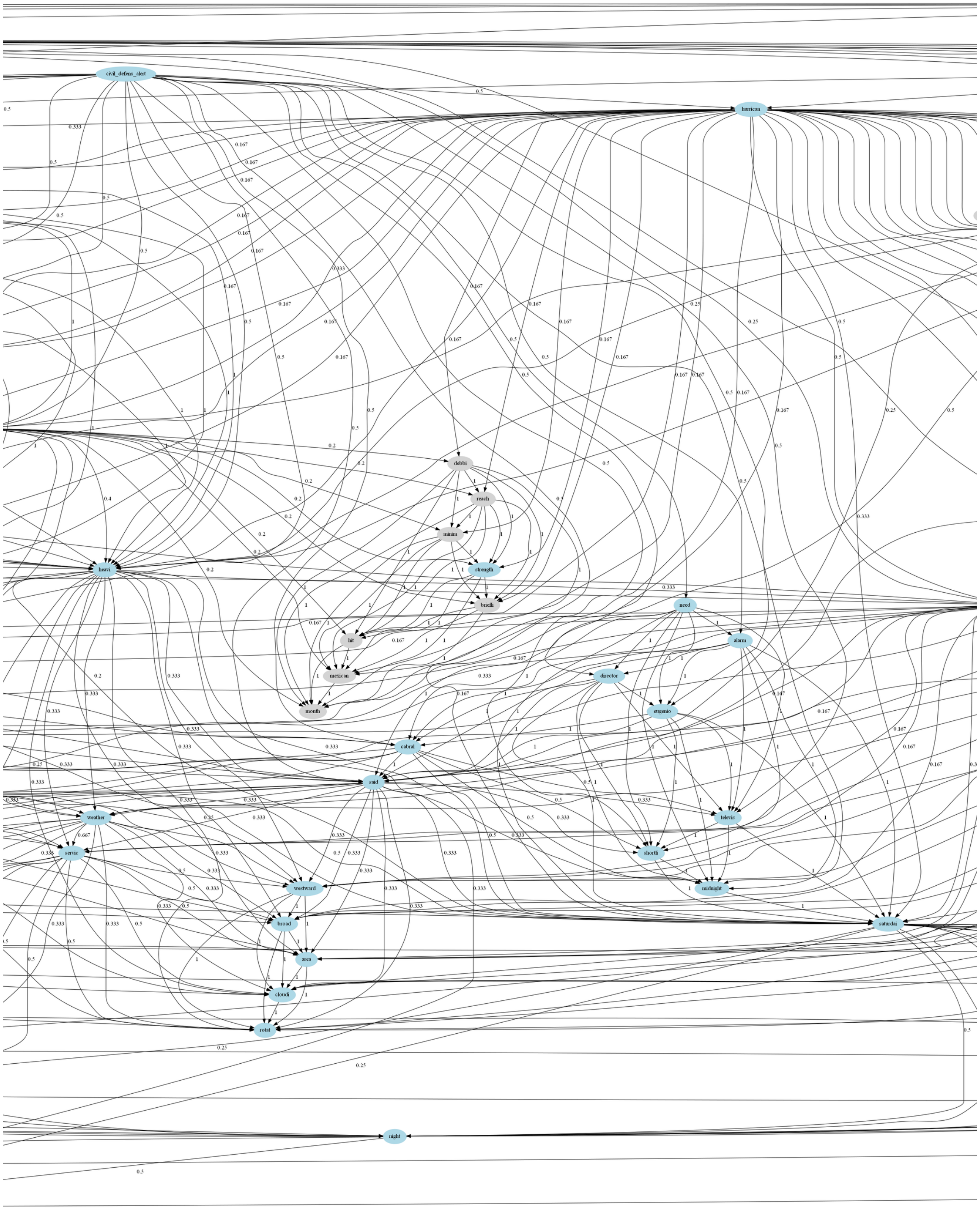


Figure 3-5: Created graph by NG-Rank method for a document in DUC2002 (part 3).

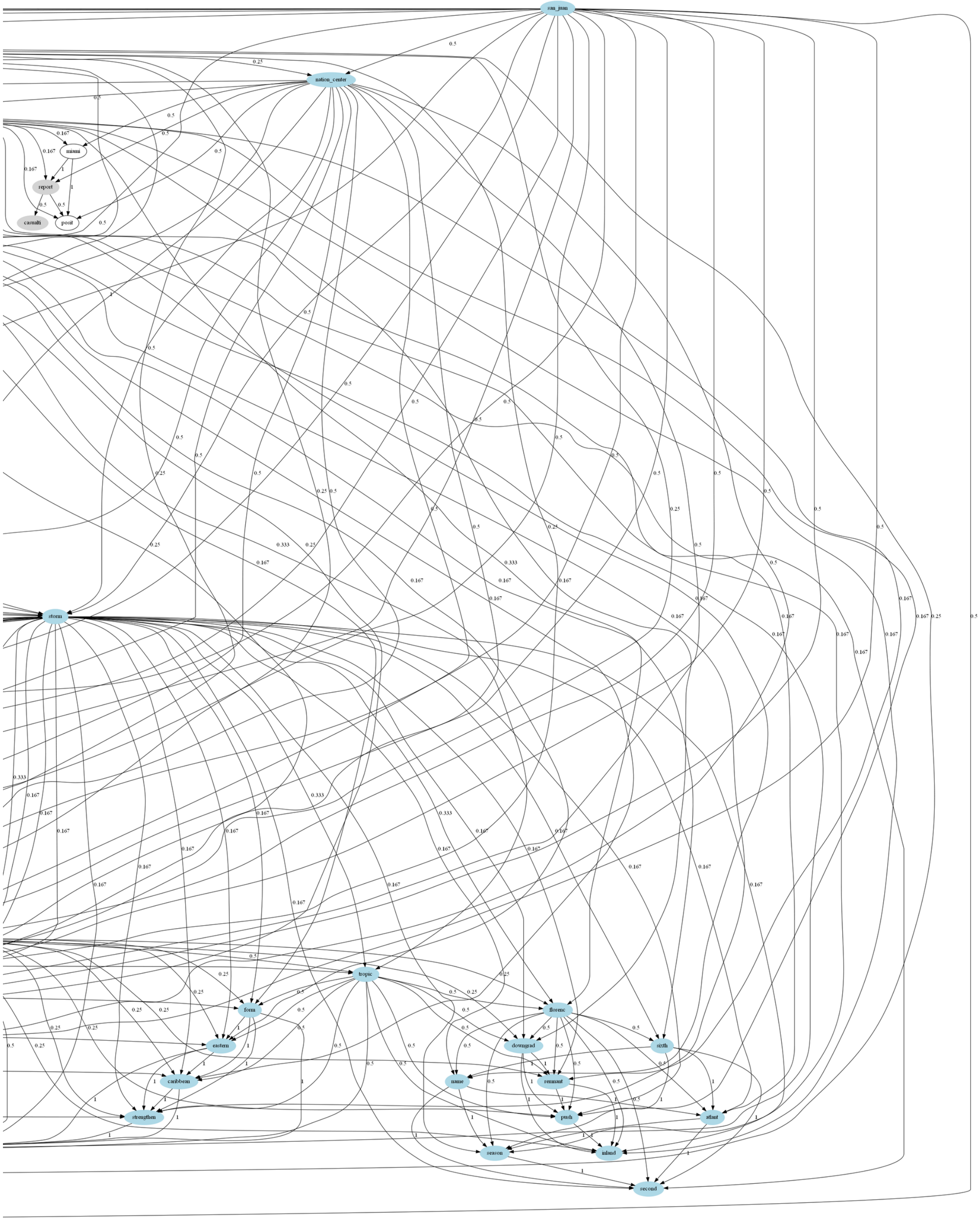


Figure 3-6: Created graph by NG-Rank method for a document in DUC2002 (part 4).

Table 3.1: NG-Rank vs. the baseline DUC 2002

	ROUGE-1			ROUGE-2		
	Avg-R_1	Avg-P_1	Avg-F	Avg-R_2	Avg-P_2	Avg-F
NG-Rank$_\mathsf{H}$	**0.364**	**0.431**	**0.395**	**0.0346**	**0.0567**	**0.043**
NG-Rank$_\mathsf{M}$	0.342	0.396	0.367	0.0341	0.0535	0.0417
Baseline	0.282	0.305	0.293	0.0084	0.0085	0.0085

3.3.1 Evaluation Measures

For evaluating quality of summaries produced by NG-Rank, we used the ROUGE-v.1.5.5 [3] evaluation toolkit.

3.4 Experimental Results

As previously stated, we first evaluate NG-Rank as a keyphrase extraction-based summarization method, by comparing the automatically generated summaries with human-generated ones. Then we indirectly assess the quality of the summaries, automatically extracted by our algorithm, by evaluating the clustering improvement after applying NG-Rank.

3.4.1 Assessing the Quality of the Summarization

For the former tests, we thus utilize DUC 2002, and adopt the ROUGE evaluation toolkit to measure the quality of summaries. DUC 2002 provides reference summaries of 100-words (manually produced) to be used in the evaluation process. We stemmed tokens and removed stop words from reference and extracted summaries. In our experiments, we tested both NG-Rank$_\mathsf{M}$ and NG-Rank$_\mathsf{H}$[4] for extracting keyphrases from documents. To compare our method with the HITS-based algorithm (our baseline), we considered the best results obtained for the possible Γ functions presented in Section 3.1. The size of the summary we have to extract for each documents should

[3] http://www.berouge.com/

[4] For the convergence of HITS, we stop iterating when for any vertex i in the graph the difference between the scores computed at two successive iterations fall below a given threshold: $\frac{|x_i^{k+1}-x_i^k|}{x_i^k} < 10^{-3}$ [50]

be equals to the manually produced reference summary, Since we also remove from them stop words, thus making the reference summaries smaller than the original 100-words ones, we had to choose a suitable parameter α for NG-Rank. Recall that α determines the percentage of top-scored graph nodes in each partition FS, LS, or MS that NG-Rank returns (see Section 3.2).

We used two of the ROUGE metrics in the our comparison, ROUGE-1 (unigram) and ROUGE-2 (bigram). The obtained results are showed in Table 3.1. The convergence time of HITS algorithm increases the execution time of NG-Rank$_\mathsf{H}$, but it is negligible considering the significant results obtained by NG-Rank$_\mathsf{H}$. Due to this encouraging result, we always applied NG-Rank$_\mathsf{H}$ to summarize long documents in our experiments on clustering.

3.4.2 Assessing the Clustering Improvement due to Summarization

In previous experiments, we applied NG-Rank$_\mathsf{H}$ to summarize longer documents in our corpus, before applying a text clustering algorithm. The algorithm adopted for clustering documents was K-Means, while the vectorial representation of documents was based on a classical *tf-idf* weighting of terms, and the measure of similarity between two vector was Cosine similarity. Specifically, we utilized *RapidMiner*[5], which is an integrated environment for analytics, also providing tools for text mining.

Indeed, we tested and evaluated clustering with/without applying NG-Rank$_\mathsf{H}$, to show the improvements in clustering purity due to summarization. Before reporting and examining the various results, we have first to discuss the features of the sampled corpora, which contain some longer documents. These longer documents are exactly our candidates for summarizations. As stated in Section 3.3, for each sampled corpus $\mathbb{D} = \{\mathcal{L}_1, \mathcal{L}_2, ..., \mathcal{L}_c\}$, we have $\mathcal{L}_i = R_i \cup E_i$, where E_i denotes the set of documents of the i^{th} class that are significantly longer than the average length $\mathcal{M}$. More specifically, the documents in E_i have a size that is at least 3 times $\mathcal{M}$. In our test we used five sampled datasets $\mathbb{D}$, with different sizes of $|E_i| = \{7, 12, 18, 25\}$.

Another important remark concerns the size of the summaries extracted by NG-Rank$_\mathsf{H}$ from each longer document in E_i. This size is determined by the parameter α of the

[5] https://rapidminer.com/products/studio/

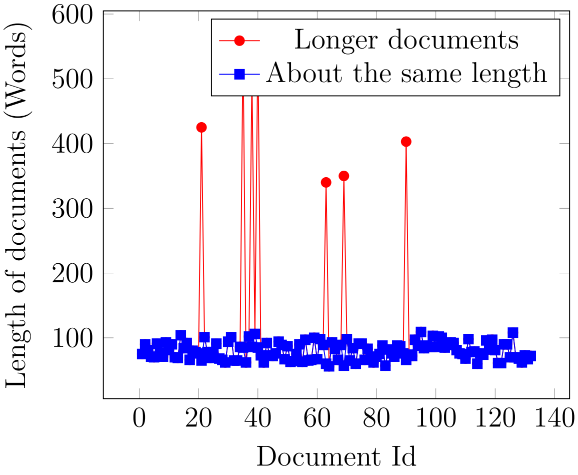


Figure 3-7: Length reduction of documents in the class *Sport* (consists of 7 long documents) of a corpus sampled from BBC. After summarization, the lengths of longer documents are reduced, and all documents become of about the same length len (in the range $50 \leq len \leq 120$).

algorithm (see Section 3.2). For each $d \in E_i$, we chose $\alpha = \lceil \frac{\mathcal{M}}{|d|} \rceil$, where $|d|$ and $\mathcal{M}$ denote, respectively, the length of d and the average length of the shorter documents in the sampled class. Figure 3-7 shows the size of the documents belonging to a given class, namely the class *Sport* in a dataset sampled from the BBC corpus, before and after summarizing larger documents.

Figure 3-8 shows the average purity obtained by clustering documents in each sampled corpus $\mathbb{D}$, with/without summarizing longer documents. The best improvements in the average purity, due to summarization of longer documents, were about 10%.

Table 3.2.(a) shows the clustering results without summarizing the longer documents. The dataset used in the test were obtained from the BBC NEWS corpus, where the longer documents were added to the classes *Polit* and *Sport* only. Specifically, we have $|E_{Polit}| = 25$ and $|E_{Sport}| = 7$, while $|E_{Bus}| = 0$ and $|E_{Enter}| = 0$. The size of each class before adding these longer documents was: $|R_{Bus}| = 116$, $|R_{Enter}| = 117$, $|R_{Polit}| = 75$, and $|R_{Sport}| = 125$. Table 3.2.(b) reports the results obtained by first applying $\mathsf{NG\text{-}Rank_H}$ to summarize the longer documents, and by then clustering all the document collection. We obtained an improvement in the average purity of about

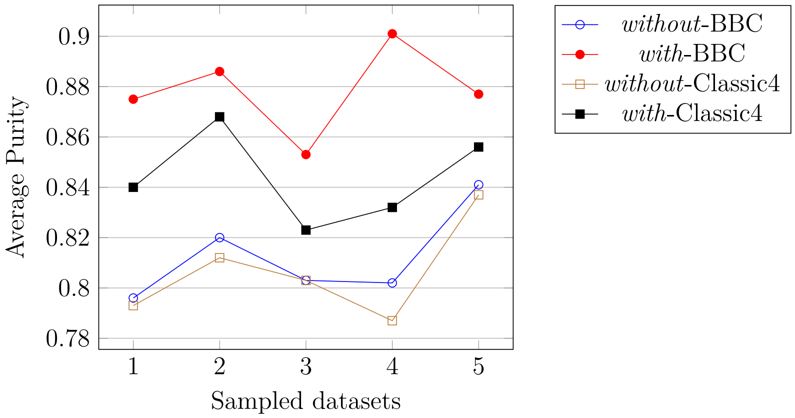


Figure 3-8: Average purity of clustering, *with/without* applying $\mathsf{NG\text{-}Rank_H}$, for five datasets sampled from the BBC and Classic4 corpora. The five sampled datasets, each corresponding to a distinct $j \in \{1, \ldots, 5\}$ on the x-axis, are characterized by different numbers of longer documents $|E_i|$, for each class i.

Table 3.2: Clustering results: (a) original documents without any summarization; (b) after replacing longer documents with their summaries extracted by $\mathsf{NG\text{-}Rank_H}$ (**BBC Dataset**)

Cluster	Bus	Enter	Polit	Sport	Purity
Cluster 0	7	4	83	5	0.838
Cluster 1	1	61	0	0	0.984
Cluster 2	105	50	10	3	0.625
Cluster 3	3	2	7	124	0.912
Total Purity					**0.802**

(a)

Cluster	Bus	Enter	Polit	Sport	Purity
Cluster 0	10	1	93	6	0.845
Cluster 1	0	101	0	2	0.980
Cluster 2	105	15	7	2	0.814
Cluster 3	1	0	0	122	0.992
Total Purity					**0.905**

(b)

10%.

Table 3.3 reports a similar experiment conducted on a dataset sampled from Classic4. Specifically, we have $|E_{Cisi}| = 18$, $|E_{Cran}| = 12$,$|E_{Med}| = 0$, and $|E_{Cacm}| = 7$. The size of each class before adding these longer documents was: $|R_{Cisi}| = 82$, $|R_{Cran}| = 88$,$|R_{Med}| = 100$, and $|R_{Cacm}| = 93$. In this case the improvement in average purity was smaller than for the BBC dataset. However, we registered a similar behaviour, and thus summarizing longer documents by using our algorithm is always valuable.

We conclude with some final remarks about our methodology based on document

Table 3.3: Clustering results: (a) original documents without any summarization; (b) after replacing longer documents with their summaries extracted by $\mathsf{NG\text{-}Rank_H}$ (**Classic4 Dataset**)

Cluster	Cisi	Cran	Med	Cacm	Purity
Cluster 0	0	88	0	1	0.989
Cluster 1	3	1	0	47	0.921
Cluster 2	6	10	100	40	0.641
Cluster 3	91	1	0	12	0.875
Total Purity					**0.815**

(a)

Cluster	Cisi	Cran	Med	Cacm	Purity
Cluster 0	0	88	0	0	1
Cluster 1	4	10	7	60	0.740
Cluster 2	1	2	93	0	0.969
Cluster 3	95	0	0	40	0.703
Total Purity					**0.840**

(b)

summarization. When we add longer documents to a class, we likely increase the frequency of terms that are not relevant to the main topic of the class. Indeed, each document contains several topics, for each of which there are relevant terms in documents [5]. Therefore, when we increase the length of a document, we may cause the number of topics to get larger. We can think of $\mathsf{NG\text{-}Rank_H}$ as a method to remove some of these less important/relevant topics, by retaining the main topics only, hopefully those topics that are common to all the documents in a given class.

4

Enriching Text Documents by Linking Salient Entities and Lexical-Semantic Expansion

In this chapter, we explore a multi-strategy technique that aims at enriching text documents for improving clustering quality. To effectively enrich documents without introducing noise, we limit ourselves to the text fragments mentioning the salient entities, in turn belonging to a *knowledge base* like Wikipedia, while the actual enrichment of text fragments is carried out by using Wordnet.

To feed clustering algorithms, we investigate different document representations, in turn obtained by using several combination of document enrichment and feature extraction. Indeed, this allows us to exploit ensemble clustering, by combining multiple clustering results obtained by using different document representations.

Our experiments indicate that our novel enriching strategies, combined with ensemble clustering, can significantly improve the quality of classical text clustering.

4.1 Introduction

In traditional text clustering, the vector-based representations of texts are purely based on terms occurring in documents. Other information, in particular latent ones, should be included in the document representation to improve the quality of document similarity metrics. In this chapter, we investigate a combination of two techniques to make manifest such latent information. First, we select some important words in a text document, by identifying the *text fragments* mentioning the *most salient entities* linked to a *knowledge base* (indeed, articles of Wikipedia). Second, we enrich such subset of important text fragments using common semantic concepts based on a lexical-semantic database (indeed, WordNet).

We focus on a simple *Entity Linking* (EL) technique [49, 52, 21, 11] aimed at identifying entities from their *mentions* or *spot* (i.e., small fragments of text referring to any entity in a knowledge base) occurring in a large corpus. More precisely, we use Wikipedia as the referring knowledge base of entities and associated mentions. The method exploited returns, for each spot selected, the *entity*, namely a Wikipedia page with an unique URL, its title, and a set of semantic categories (or types) of the page as defined in Wikipedia.

Similarly to other proposals, we aim at enriching the vectorial representation of text documents in order to improve clustering results. EL techniques that exploit Wikipedia as a knowledge base already identify a limited sets of topics being discussed in the text, since they count on the so-called *link probability* property. The link probability of a spot m, denoted by $LP(m)$, is in fact defined as the number of times m occurs as an anchor text in Wikipedia divided by its total number of occurrences in all the Wikipedia pages [49]. This property permits the EL techniques to discriminate between mentions that refers with a high probability to an entity from those referring to an entity only occasionally.

Indeed, we combine such EL technique, which already limits the number of entities identified, with text summarization: the final goal is to identify the most salient entities/topics discussed in a document, and we concentrate our efforts on them to enrich the final document vector representation. Specifically, we exploit a Graph-based Ranking summarization algorithm [59] to create a summary and finally identify the most salient entities. Moreover, we utilize WordNet to expand salient entities with ontology-based latent information. We take advantage of predecessor/successor concept within four semantical relations in WordNet to expand latent information of the salient entities exposed by summarization method. To this end, we use a prolog-readable format of WordNet database to create corresponding rooted DAG (Directed Acyclic Graph) of each relation which ables us to identify and index all the paths from roots to leaves, considering predecessor/successor concept.

Finally, since semantic enrichment allows us to produce different vector representations of documents, and thus different similarity measures between them, we exploit a clustering ensemble approach applied to BBC NEWS articles to validate our technique, indeed the improvement in clustering quality obtained.

4.2 Document Enriching and Ensemble Clustering

We propose an unsupervised approach, called Salient Entities for Enriching Documents (SEED), to enrich documents before clustering. The aim of SEED is to identify the fragments of text to enrich concerning the main topics discussed in each document, overcoming the issues of using term/document frequency to identify such fragments.

The brief description of our approach is as follows: first we extract all the entities from a document. In order to extract such entities from text we use Dandelion Entity Extraction[1], which benefits from the research results of TAGME [21]. For summarizing text and finally identifying the salient entities, we exploit the NG-Rank algorithm [59]. Indeed, the entities that are in common between the summary and the original text are selected as the most salient entities. We then utilize the semantic relations in the WordNet ontology to expand such salient entities, by carefully disambiguating the sense of terms, namely, the spots identified by salient entities in the text (Section 4.2.2). Finally, different representations for documents are provided by combining expanded sets of features. We then exploit ensemble clustering to combine multiple clustering results, obtained by using diverse document enrichment strategies.

4.2.1 Document Enrichiment

In the following, we sketch the various steps for document enrichments:

[1] https://dandelion.eu/

Entity extraction.

We use the Dandelion Entity Extraction API to obtain, given an input text, the Wikipedia entities (titles and URIs) occurring within the text, along with their spots/mentions and some other relevant information. The spot (or mention) of an entity indicates the fragment of text that is identified as a reference to the detected entity, like the anchor text of a hyperlink.

More formally, let $\mathcal{D} = \{D_1, D_2, \ldots, D_m\}$ be a collection of documents, and let $Ent(D_i) = \{(e_1, m_1), (e_2, m_2), \ldots, (e_n, m_n)\}$ be the set of all pairs of *entities* and *associated spots/mentions* (e_i, m_i) occurring in D_i. While each e_i is identified by a URI and/or a unique title, a spot/mention m_i is indeed an *n-gram*, i.e., a contiguous sequence of n terms referring to e_i.

Salient Entity selection.

To select the most salient entities, we exploit the NG-Rank summarization algorithm [59] to create a summary S_i for each document $D_i \in \mathcal{D}$. In principle, only the entities appearing in both S_i and D_i are selected for further semantic expansions. However, since each S_i is a keyword-based summary, an *n-gram* m that is recognized as an entity spot in the original document D_i can appear only partially in S_i, or the terms of m can be scattered over the text of S_i. Obviously, if all the terms of the n-gram m are completely discarded during the summarization and thus do not appear in S_i, the associated entity is not considered salient, but what if the terms of m appear partially or are spread over the summary?

To illustrate our simple method, we consider each summary S_i, and each spot m as a multiset (bag) of words. So, the salient entities $\widehat{Ent}(D_i)$, where $\widehat{Ent}(D_i) \subset Ent(D_i)$, are identified as follows:

$$\widehat{Ent}(D_i) = \{(e, m) \in Ent(D_i) \mid m \cap S_i \neq \emptyset\}$$

where for each $(e, m) \in Ent(D_i)$, we have by definition that $\forall x \in m$, $x \in D_i$. We argue that this method allows us to enrich a document by only expanding important portions of the document, without introducing noise, which could come from a method that semantically enriches terms of irrelevant phrases too, namely salient and not salient ones.

Finally, the set of mentions to salient entities occurring in a document D_i is denoted by $\widehat{M}(D_i)$ and defined as follows:

$$\widehat{M}(D_i) = \bigcup_{(e,m)\in\widehat{Ent}(D_i)} m$$

For example, consider that given a document D, we have:

$$\widehat{Ent}(D) = \{(Profit\ (accounting),\ profit),\ (Market\ (economics),\ market),\\ (United\ States\ dollar,\ dollar),\ (Telecommunication,\ telecoms)\},$$

where the former element of each pair, e.g., *"Profit (accounting)"*, is the *title* of Wikipedia articles, while the latter one, e.g., *"profit"*, is the corresponding n-gram spot. In this example, all the spots are simple 1-grams. Finally we have that $\widehat{M}(D) = \{profit,\ market,\ dollar,\ telecoms\}$.

A word may have some different senses (meanings), and a sense is selected as the best one for a given word depending on the context in which it occurs. To disambiguate the senses of the words in $\widehat{M}(D)$, we propose a Word Sense Disambiguation (WSD) algorithm which is illustrated in Section 4.2.2. The WSD receives $\widehat{M}(D)$, which is also used as word context for disambiguation, and assigns a sense to each word in $\widehat{M}(D)$, according to their semantic relations with the senses of other words of $\widehat{M}(D)$. Indeed, WSD first identifies a set of candidate sense for each word in $\widehat{M}(D)$, then it assigns a vote to each candidate. Finally, these votes are used to rank the possible senses for each word, thus selecting the sense with the highest vote.

Our experiments showed that using a word context that only contains the most relevant words related to the most salient entities (topics) discussed in the document, yields better results in sense disambiguation. Moreover, the words senses that have been disambiguated by using the word context $\widehat{M}(D)$ could further be used in whole document, with the aim of disambiguating other word senses of the whole document.

In addition, WSD is also used to create a weighted graph G_i for each $\widehat{M}(D_i)$, which is exploited for discarding some noisy words from $\widehat{M}(D_i)$ (see Section 4.2.3). At the end of this process, we obtain $\overline{M}(D_i)$, where $\overline{M}(D_i) \subseteq \widehat{M}(D_i)$, which contains the most relevant words that are finally expanded to obtain a richer vector representation of each document $D_i \in \mathcal{D}$.

Expanding Salient Entities.

This step regards the final enrichment, given a document corpus $\mathcal{D} = \{D_i\}_{i=1,\dots,m}$, using a lexical-semantic database (*WordNet*) applied to salient entities, we expand the elements of $\overline{M}(D_i)$.

To disambiguate the sense of words which is stated above, and finally to enrich documents, we exploit four semantic relations in WordNet:

- *hypernym* (kind-of or is-a): Y is a hypernym of X if every X is a (kind of) Y (e.g., *motor vehicle* is a hypernym of *car*).
- *member meronym* (member of): Y is a member meronym of X if Y is a member of X (e.g., *professor* is a member meronym of *faculty*).
- *part meronym* (part of): Y is a part meronym of X if Y is a part of X (e.g., *camshaft is part meronym of engine*).
- *substance meronym* (contains, used in): Y is a substance of X if Y contains (used in) X (e.g., *Water* is a substance meronym of *oxygen*).

Each of these relations can be used to extract a graph from WordNet, in particular *rooted directed acyclic graph* (DAG), where nodes are synsets, and directed edges model one of the above semantic relations. We use such DAGs for several steps, namely to disambiguate sense of words, and finally to enrich the document vectorial representation of documents to be clustered.

Indeed, in this section we discuss how we extract features and prepare the vector representation of documents, once the WSD algorithm has detected the sense of each word in $\overline{M}(D_i)$, and defer the detail of WSD to Section 4.2.2.

Specifically, for each word in $\overline{M}(D_i)$, we exploit the synsets identified by WSD, i.e., the senses of words in $\overline{M}(D_i)$, along with further synsets in WordNet that are related to the first ones through semantic relations of type *hypernym*, *part meronym*, *member meronym*, and *substance meronym*, respectively. Let $Syns(D_i)$ denote the senses (synsets) of the words in $\overline{M}(D_i)$, as identified by WSD. For a given $s_i \in Syns(D_i)$, and for each type of semantic relation, e.g., for *hypernym*, we can distinguish between synsets that are *direct predecessor* and *direct successor* in the hypernym DAG extracted from WordNet. If an edge (s_i, s_{succ}) exists in the DAG, s_{succ} is a direct successors, whereas an edge (s_{pred}, s_i) identifies s_{pred} as a direct predecessor.

At the end of this process, by considering all the words in $\overline{M}(D_i)$ and the four types of relations, we can associate three sets of synsets with each D_i: $Syns(D_i)$, $PredSyns(D_i)$, and $SuccSyns(D_i)$. While $Syns(D_i)$ includes the sense synsets of the words in $\overline{M}(D_i)$, the other two sets contain, respectively, the direct predecessor and direct successor synsets according to all the four types of WordNet relations. The pseudocode in Algorithm 1 illustrates the part of extracting features for preparing the vector representation of documents.

Algorithm 1 DocumentEnriching

```
$\mathcal{D} = \{D_1, D_2, \ldots, D_m\}$
$DAGs = \{DAG(SY), DAG(SP), DAG(SM), DAG(SS)\}$
foreach $D_i \in \mathcal{D}$ do
    $Sum_i \leftarrow NG\_Rank(D_i)$
    $Ent_i \leftarrow PairsOfEntity\_Spot(D_i)$
    foreach $ent \in Ent_i$ do
        if $Sum_i.Contains(ent.m)$ then
            $\widehat{Ent}_i.Add((ent.e, ent.m))$
            $\widehat{M}_i.Add(ent.m)$
        end if
    end for
end for
$WoN \leftarrow Words\_Of\_Spots(\widehat{M}(D_i))$
$BestSenses \leftarrow WordSenseDisam(WoN)$
$\overline{M}(D_i) \leftarrow RelevantTermsByMETIS(BestSenses)$
$SuccSyns(D_i) \leftarrow \{\}$
$PredSyns(D_i) \leftarrow \{\}$
foreach $bs \in BestSenses$ do
    $SuccSyns(D_i).AddSuccessorsOf(bs, DAGs)$
    $PredSyns(D_i).AddPredecessorsOf(bs, DAGs)$
end for
```

Finally, for each document $D_i \in \mathcal{D}$, we pick from a large set of sources to extract the features of the vector representing D_i. In particular, we can exploit:

Or_i: $OrigDoc(D_i)$, denoted in short by Or_i, is the multiset of words associated with the original document D_i;

Sum_i: $SummDoc(D_i)$, denoted in short by Sum_i, is the multiset of words occurring in the summary extracted from D_i by NG-Rank [59];

Na_i: $NamesEN(D_i)$, denoted in short by Na_i, is the multiset of words containing the titles of the salient entities in $\widehat{Ent}(D)$, formally defined as follows:

$$Na_i = \bigcup_{(e,m) \in \widehat{Ent}(D_i)} e$$

Sp_i: $SpotsEN(D_i)$, denoted in short by Sp_i, is the multiset of words containing the spots of the salient entities in $\widehat{Ent}(D)$, formally defined as follows:

$$Sp_i = \widehat{M}(D_i) = \bigcup_{(e,m) \in \widehat{Ent}(D_i)} m$$

Sy_i: $Syns(D_i)$, denoted in short by Sy_i, is the multiset of words occurring in the senses (synsets) of the words in Sp_i, i.e., in the spots of the most salient entities in D_i;

Pre_i: $PredSyns(D_i)$, denoted in short by Pre_i, is the multiset of words occurring in all the synsets that directly precede the ones in Sy_i, according to any of the four types of WordNet relations *hypernym*, *part meronym*, *member meronym*, and *substance meronym*;

Suc_i: $SuccSyns(D_i)$, denoted in short by Suc_i, is the multiset of words including all the synsets that are the direct successors of the ones in Sy_i, according to any of the four types of WordNet semantic relations above.

In all the word multisets listed above, we remove stop words and stem the rest of the words.

Feature selection and ensemble clustering:

We utilize a *clustering ensemble* method, which combines different clustering results to finally partition documents. Even we adopt the same clustering algorithm to partition the input document corpus, since we adopt different enrichments and associated vector representations of documents, the final clustering results may differ. The rationale of using ensemble clustering is that each single enrichment strategy may generally work for the whole corpus, but may introduce noise in the representations of a few documents that are eventually clustered badly. Ensemble clustering permits us to exploit many possible document enrichments, and finally remove possible noisy results through a consensus method.

A cluster ensemble method consists of two steps: *Generation*, which creates a set of possible partitions of the input objects (in our case, a document corpus), and *Consensus*, which computes a new partition by integrating all the partitions obtained in the generation step [75].

In our experiments, for the generation step we adopt a hybrid function $\mathcal{F}$, indeed many different instances $\{\mathcal{F}^h\}_{h=1,\ldots,n}$ of this function, that entails different *feature selection* methods, thus generating different subsets of features and vectorial representation of documents. Specifically, we consider the above multisets of words for each document D_i, denoted by $SES(D_i) = \{Or_i, Sum_i, Na_i, Sp_i, Sy_i, Pre_i, Suc_i\}$, and combine them by using different instances of function $\mathcal{F}$.

Let $\mathbb{C} = \{\mathcal{C}^1, \mathcal{C}^2, \ldots, \mathcal{C}^n\}$ be the different clusterings of the document corpus $\mathcal{D}$, where each clustering $\mathcal{C}^h$ is obtained by first applying the instance $\mathcal{F}^h$ of the feature selection function over the corpus's documents, and then by running over them a given clustering algorithm. In our case, we exploit k-means, a well-known algorithm that takes the input document corpus and produces k disjoint clusters. Specifically, each $\mathcal{C}^i$ is thus a partition od $\mathcal{D}$. Formally, the enriched bag-of-words representation of D_i, obtained by $\mathcal{F}^h$, is denoted by D_i^h, while the instance $\mathcal{F}^h$ of the combining function is due to the different setting of six integer parameters, namely $\alpha, \beta, \gamma, \varepsilon, \delta$, and η:

$$D_i^h = \mathcal{F}^h(D_i|\alpha^h, \beta^h, \gamma^h, \varepsilon^h, \delta^h, \eta^h) =$$
$$\{Or_i\} \ \cup \ (\alpha^h \cdot Sum_i) \cup (\beta^h \cdot Na_i) \cup (\gamma^h \cdot Sp_i) \cup (\varepsilon^h \cdot Sy_i) \cup (\delta^h \cdot Pre_i) \cup (\eta^h \cdot Suc_i)$$

where $\alpha, \beta, \gamma, \varepsilon, \delta, \eta \in \{0, 1, 2, ..., t\}$ indicate the number of times we replicate the elements of $SES(D_i)$ to generate a new bag-of-words document representation D_i^h. More formally, $\alpha^h \cdot Sum_i = \bigcup_{j=1}^{\alpha^h} Sum_i$, and thus D_i^h will contain α^h replicas of the document summary Sum_i. In case of a parameter is zero, for example $\alpha^h = 0$, then $\alpha^h \cdot Sum_i$ is equal to $\emptyset$. In our experiments, we varied these parameters, and used different maximum value t for every parameter.

It is worth remarking that by varying the parameter setting to generate a different $\mathcal{F}^h$ we may change the vocabulary used to identify the dimensions of document vectors, but we also modify the term frequency and thus the *tf.idf* weights used in the vectors. As a consequence, if we enrich and represent a corpus $\mathcal{D}$ according to different $\mathcal{F}^h$, we produce different partitions of the corpus even if we run the same clustering algorithms.

As an example of this behavior, Figure 4-1 shows a corpus of two documents D_i and D_j, which are semantically similar. Whereas the first pair of boxes labeled by 1 represents the original two documents, and thus the multisets Or_i and Or_j, the other pairs of boxes, numbered from 2 to 4, correspond to different enrichments obtained by varying $\mathcal{F}^h$. For the sake of keeping the example simple, the synsets are composed of single words, and distinct words are represented as distinct capital letters. Figure 4-1.(b) indicates parts of predecessor/successor relations in DAGs, corresponding to the words used in (a).

The upper part of each box is intended to contain the synsets included in Pre and Suc (possibly replicated, according to parameters δ and η), the middle part indicates the original documents Or, and finally the bottom part indicates the spots of the most salient entities Sp (possibly replicated, according to parameter γ). In case (a).1, the similarity between D_i and D_j is 0. This case corresponds to the original documents, or equivalently, according to our feature selection function, it corresponds to setting to 0 all the parameters of $\mathcal{F}^h$.

When we select different subsets of features, as case in (a).2 their similarity becomes 0.0769. If we replicate the predecessor/successor synsets, by setting $\delta = \eta = 3$ or $\delta = \eta = 5$ as in (a).3 and (a).4, the similarity of documents keeps rising to 0.228 and 0.289, respectively. Note that replicating the elements of these sets does not mean that the cosine similarity of the two documents increases so much that these documents will certainly be placed in

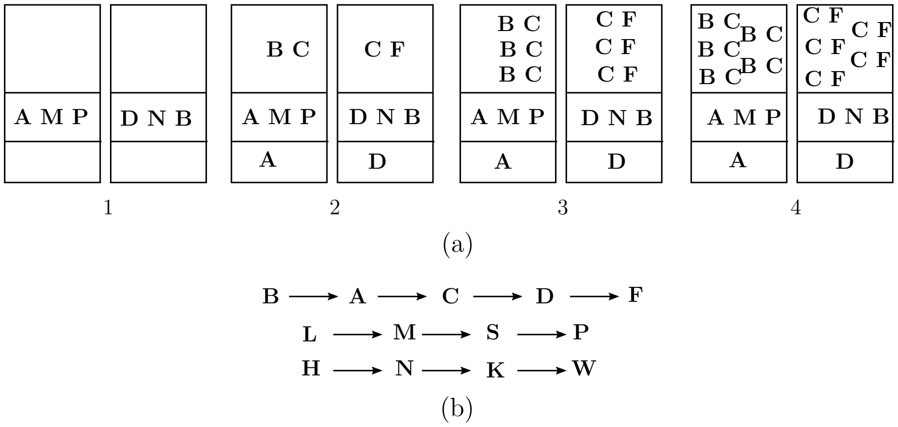


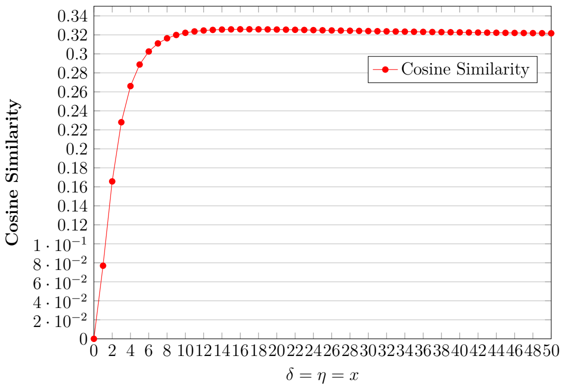
Figure 4-1: (a) two documents with different subsets of features and different values for parameter *tf.idf* of their features (b) semantical relations between the words.

the same cluster by the clustering algorithm. First, because of the nature of the radical functions, the similarity function for two vectors that a few elements of them are increasing with the same factor would be a bounded function similar to $f(x) = \frac{a+bx+cx^2}{\sqrt{d+mx^2}\times\sqrt{f+mx^2}}$. For example, in Figure 4-1, tf.idf values are increasing only for three words B, C, and F in the representation vectors, thus rest of the words' tf.idf values generate the constant values in $f(x)$, and the created equation (i.e., $tf.idf(B) = tf.idf(C) = \frac{1}{2} \times tf.idf(F)$) between the tf.idf of the three words B, C, and F generate the variable of $f(x)$. As shown in Figure 4-2, the maximum similarity that we can have for d_1 and d_2 is obtained by $\delta = \eta = 17$. Second, by replicating elements in the bag-of-words document representation, we may further increase the noises coming from some irrelevant elements (for example, the synset **F** in Figure 4-1.(a)), which may consequently entail a wrong cluster assignment of the two documents.

For the consensus step, we apply the *objects co-occurrence* approach, which is based on the computation of how many times two objects are assigned to the same cluster by the various clustering instances of the ensemble. Like in the *Cluster-based Similarity Partitioning Algorithm* (CSPA)[68], we thus build $m \times m$ similarity matrix (the co-association matrix), which can be viewed as the adjacency matrix of a weighted graph, where the nodes are the elements of the document corpus $\mathcal{D}$, and each edge between two object (documents) is weighted with the number of times the objects appear in the same cluster, for each instance of the clustering ensemble. Then, the graph partitioning algorithm METIS is used for generating the final consensus partition.

$\mathcal{VR}$		A	M	P	D	N	B	C	F	$\delta = \eta$	CS
d_1	→	1	1	1	0	0	0	-	-	0	0.0
d_2	→	0	0	0	1	1	1	-	-	0	
d_1	→	2	1	1	0	0	0.5	0.5	0	1	0.076
d_2	→	0	0	0	2	1	0.5	0.5	1	1	
d_1	→	2	1	1	0	0	2.5	2.5	0	5	0.288
d_2	→	0	0	0	2	1	0.5	2.5	5	5	
...											
d_1	→	2	1	1	0	0	8.5	8.5	0	17	0.325
d_2	→	0	0	0	2	1	0.5	8.5	17	17	
...											
d_1	→	2	1	1	0	0	25	25	0	50	0.321
d_2	→	0	0	0	2	1	0.5	25	50	50	

(a)



(b)

Figure 4-2: (a) The document Vectorial Representation ($\mathcal{VR}$) generated, using $tf.idf$, for documents d_1 and d_2, and Cosine Similarity (CS) calculated for them, according to parameters δ and η; (b) graph of cosine similarity function $f(x) = \frac{0.25x+0.25x^2}{\sqrt{6+0.5x^2} \times \sqrt{5.25+1.25x^2}}$ in Figure 4-1.

4.2.2 Words Sense Disambiguation

In this section, we propose an unsupervised *Words Sense Disambiguation* (WSD) method, using WordNet as a knowledge base. Given a *target word* to be disambiguated, we utilize the four above-mentioned semantic relations of WordNet to identify its best sense (meaning) among all the possible senses in WordNet. The disambiguation WSD strategy takes advantage from the *word context*, i.e., a portion of document that surrounds each word. The size of word contexts may be different (e.g., Unigram, Bigrams, Trigrams, Sentence, Paragraph, or different size of a window) [55]. Determining such word context for a target word is crucially important, because wrong relations between the target word and other words in the context may affect the best sense selection.

The novel idea of our approach is to build a set of words to be disambiguated by including words occurring in the spots of the salient entities of each document $D_i \in \mathcal{D}$. This set is indeed obtained by combining an entity linking toolkit along with a text summarization NG-Rank method [59]. Therefore, the target words are the ones included in $\widehat{M}(D_i)$, and $\widehat{M}(D_i)$ is also used as the context of each target word. Since our word context $\widehat{M}(D_i)$ is extracted by a summary including terms closely related to the *main topic* of documents, semantically related to each other, this should hopefully favor a fair selection of the appropriate senses for each target word.

Our approach proceeds as follows: given $\widehat{M}(D_i) = \{w_1, w_2, ..., w_{n_i}\}$ as the word context,

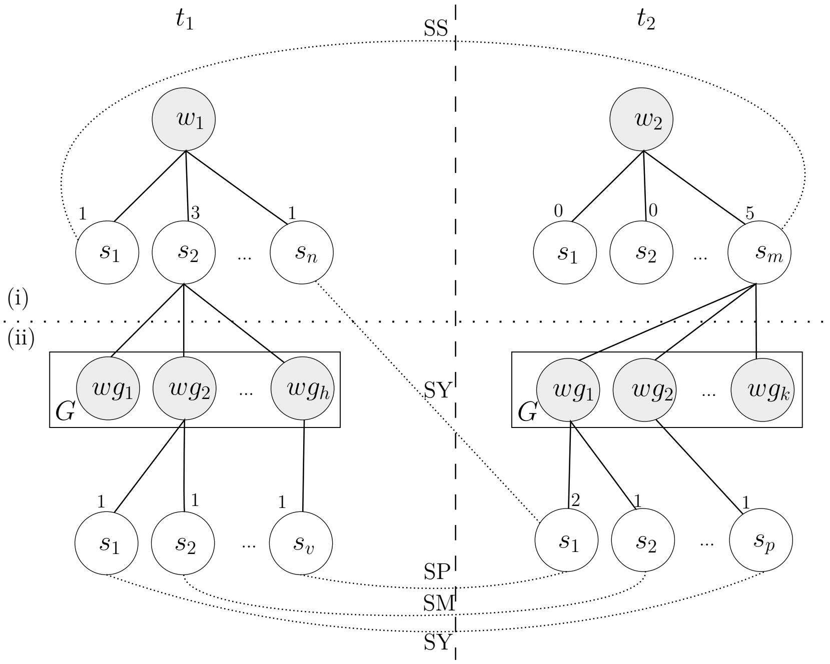


Figure 4-3: Two trees t_1 and t_2 created for two words w_1 and w_2. The gray nodes are words that may be a word of the context, or a word included in the gloss of a synset. The white nodes are synset, i.e., the possible senses of each words. Here, the best senses selected for w_1 and w_2 are s_2 and s_m, respectively, since the votes (3 and 5) are the highest among the candidate senses.

where w_i is a word of a spot, we create n_i semantic trees, one for each $w_k \in \widehat{M}(D_i)$, as illustrated in Figure 4-3. The method works as follows:

1. first, for each $w_k \in \widehat{M}(D_i)$, associated with the root of a tree, we identify $S(w_k) = \{s_1, s_2, ..., s_{m_k}\}$, which includes all the possible senses (*synsets*) of w_k in WordNet. According to the WordNet framework, a sense of a word is one of these possible synsets that can associate with the word, where a synset is an unordered set of synonyms that can be interchangeable used in many contexts. From the root w_k, the tree is thus grown by adding $m_k = |S(w_k)|$ children, where each child corresponds to a distinct synset s_j;

2. for each sense s_j in $S(w_i)$, currently a leaf of the tree, we denote by $G(s_j) = \{wg_1, wg_2, ..., wg_{h_j}\}$ all the terms within the associated *gloss*, where a gloss in WordNet consists of one or more short sentences illustrating the use of the synset members. From each leaf s_j, we further grows the tree, by adding $h_j = |G(s_j)|$ children, where

each child corresponds to a distinct word appearing in the gloss $G(s_j)$;

3. for each $wg_t \in G(s_j)$, and for all $s_j \in S(w_k)$, we repeat step 1, and add another level to the tree. The new leaves are the possible synsets associated with $wg_t \in G(s_j)$.

The pseudocode in Algorithm 2 illustrates the process of creating semantic trees.

Algorithm 2 CreateSemanticTrees $\mathcal{T}(D_i)$

```
$\mathcal{T}(D_i) \leftarrow \{\}$
**foreach** $w_k \in \widehat{M}(D_i)$ **do**
    $t_k \leftarrow null$
    $S(w_k) \leftarrow AllSynsetsOf(w_k)$
    $t_k.Grow(S(w_k))$
    **foreach** $s_j \in S(w_k)$ **do**
        $G(s_j) \leftarrow AllTermsGlossOf(s_j)$
        $t_j.Grow(G(s_j))$
        **foreach** $w_g \in G(s_j)$ **do**
            $S(w_g) \leftarrow AllSynsetsOf(w_g)$
            $t_g.Grow(S(w_g))$
        **end for**
    **end for**
    $\mathcal{T}(D_i).Add(t_k)$
**end for**
```

Finally, our technique creates a *forest of* n_i *indirected trees* $\mathcal{T}(D_i) = \{t_1, t_2, ..., t_{n_i}\}$, each of 3 levels and each associated with a distinct word of context $\widehat{M}(D_i)$.

As explained above, we exploit four semantic relations of the WordNet ontology, namely Hypernym (SY), Member Meronym (SM), Part Meronym (SP), and Substance Meronym (SS). For each of these four relations we can extract a directed graph (rooted DAG), where the nodes are synsets and the edges are the semantic relations.

Returning to consider the forest $\mathcal{T}(D_i)$, for each pair of synsets s_i and s_j occurring in two distinct trees of $\mathcal{T}(D_i)$, if a directed edge between them exists in one the four semantic DAGs, we add an *undirected inter-tree edge* between s_i and s_j, labelled as either SY, SP, SM, or SS. In our example of Figure 4-3, these new undirected inter-tree edges are represented as *dotted (labeled) links* between pairs of synsets. These edges indicate the the two connected synsets are semantic related.

Indeed, to extract the best sense for each word in $\widehat{M}(D_i)$, we proceed through a voting process, using these semantic relations between pairs of synsets as a sort of "mutual vote" between them. The final goal is to rank, for each $w_k \in \widehat{M}(D_i)$, the synsets $S(w_k) = \{s_1, s_2, ..., s_{m_k}\}$ occurring at depth 1 of each tree, for finally selecting the synset that obtains the highest vote. The voting mechanism works as follows:

1. First, we assign an initial vote to each synset s_i occurring in the forest of trees. This initial vote is simply the *degree* of the corresponding node, by only considering the dotted edges, labelled by either SY, SP, SM, or SS. The intuition is that a synset is important if it is related to others synsets occurring in other trees, in turn modelling the context $\widehat{M}(D_i)$.

2. Second, we assign the final vote to each synset in $S(w_k)$ by summing up the votes of all the synsets that belong to the subtree rooted at s_i.

The pseudocode in 3 illustrates the voting strategy of using in word sense disambiguation.

Algorithm 3 Voting

```
$DAGs \leftarrow \{DAG(SY), DAG(SP), DAG(SM), DAG(SS)\}$
for $j = 0$ to $j < \mathcal{T}(D_i).Length$ do
    foreach $s_l \in S(w_j)$ do
        for $k = j + 1$ to $k < \mathcal{T}(D_i).Length$ do
            foreach $s_f \in S(w_k)$ do
                $NumOfRelations \leftarrow RelationBetween(s_l.Subtree, s_f.Subtree, DAGs)$
                if $NumOfRelations > 0$ then
                    $s_l.Vot \mathrel{+}= NumOfRelations$
                    $s_f.Vot+ = NumOfRelations$
                end if
            end for
        end for
    end for
end for
```

If the voting strategy discussed so far is not able to select the best sense for $w_k \in \widehat{M}(D_i)$, for example because all the votes assigned to the synsets in $S(w_k)$ are zero, then we select the sense that was tagged for the highest number of times in the Semantic Concordances[2].

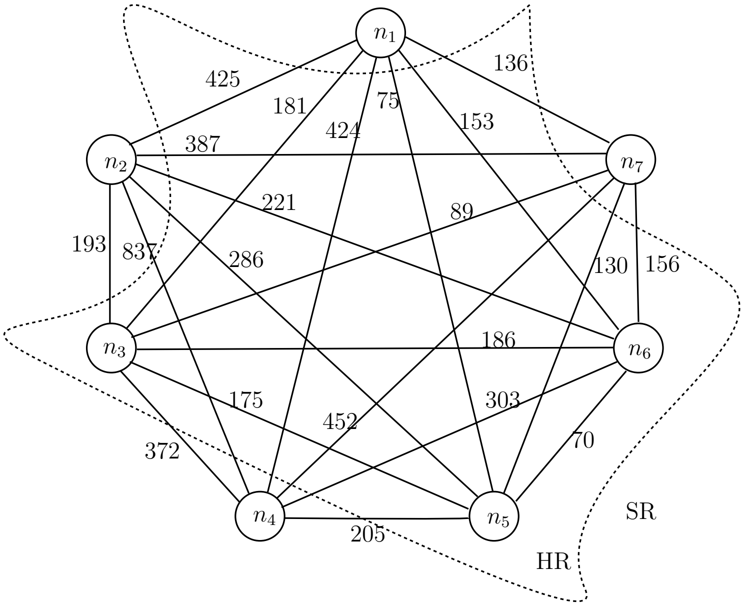
Looking at the example of Figure 4-3, the vote of s_m in tree t_2 should be equal to 1 only considering the *inter-tree relations*, since the *degree* of s_m is equal to 1 if we only consider

[2] Actually, it corresponds to the most common sense of w_k

Entities:
Property, Real estate economics, Real estate appraisal, England and Wales, Market (economics), Financial transaction, Sales, Fiscal yea

Spots:
properti, hous, market, land, transact, sale, quarter

(a)



(b)

Figure 4-4: a) The Entities and Spots (duplication in spots are removed) extracted from set EET, b) The graph G created based on the WSD output for the spot of EET. The nodes are labeled corresponding to the order of spots in (a). The dotted curve is to distinguish HR words from SR words, using METIS algorithm.

its dotted edges. The final vote of s_m becomes 5, by also considering the contribution of the synsets in the subtree rooted at s_m – namely s_1, s_2, and s_p – which contribute to the final vote by the quantity $2 + 1 + 1 = 4$.

4.2.3 Removal of noisy terms

Using only the spots of salient entities for expanding is a efficient way to reduce significant portion of noises created by irrelevant terms. However, there are noises which may be transfered by the spots of the salient entities in $\widehat{M}(D_i)$. In order to discard the transfered noises, we take advantage of the output of WSD algorithm, so that for set $\widehat{M}(D_i)$, the weighted graph G_i is created in which the nodes are associated to the included words of the set. Two nodes n_1 and n_2 are connected if at least there is a mutual vote between two senses of their corresponding words in WSD tree (Figure 4-3). The weight of the connection is computed by summing all the mutual votes between two corresponding words. Figure 4-4 shows corresponding graph for the $\widehat{M}(D_i)$ that created for a document belongs to class *Business* of dataset BBC.

Since the words in $\widehat{M}(D_i)$ are those which were passed through a filter (i.e., NG-Rank method) that attempted to pass only relevant words (i.e., the spots of the salient entities), generally, the included words in $\widehat{M}(D_i)$ are two kinds: *Hard Relevant* (HR) and *Soft Relevant* (SR) words. The HR words are those which are closely related to the main topic, but SR words are those which are related but not as much as HR words. For example, in Figure 4-4, $HR = \{market, transact, sale\}$ and $SR = \{properti, hous, land, quarter\}$. We utilized the graph partitioning algorithm METIS [31] to split graph G produced for $\widehat{M}(D_i)$ into two partitions HR and SR. The aim of using a graph partitioning algorithm is to generate a bisection graph including two partitions of words in which the sum of all edge weights, yielding by summing semantical votes received from semantical relations, of edges connecting two partitions is minimized. Hence, we can capture the noises passed across first filtering. Discarding SR words is important due to extra noises that expanding such words may cause, and consequently affect quality of clustering results.

To distinguish HR words from SR words, we utilized the scores which are assigned to the keywords of the summary created for a document (See [59] for more details). Given set $NG(D) = \{(key_1, g_1), (key_2, g_2), ..., (key_m, g_m)\}$ for document D where g_i is the score assigned to keywords key_i of the summary, we formally distinguish HR words for document D as follows:

$$HR = \begin{cases} P_1 & \text{if } \frac{\sum\limits_{P_1 \in G} g_{i|key_i \in P_1}}{|P_1|} > \frac{\sum\limits_{P_2 \in G} g_{i|key_i \in P_2}}{|P_2|} \\ P_2 & \text{otherwise} \end{cases}$$

Where P_1 and P_2 comprise words of two specified partitions of graph G using METIS. Finally, given document D_i, $\overline{M}(D_i)$ which indicates HR words of $\widehat{M}(D_i)$, is used to expand latent semantic information by exploiting semantic relations in WordNet.

4.3 Experimental Setup

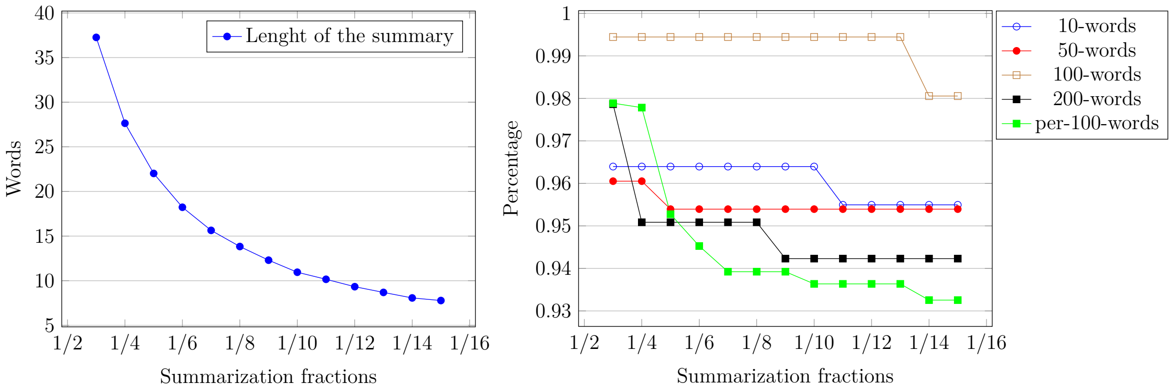
We used “DUC 2002” for testing the quality of the summarization method (NG-Rank) in order to extract salient entities from text, and “BBC NEWS” to test the benefits of our document enriching method on clustering quality. We have used BBC news articles to create two subsets $\mathcal{S}_1$ and $\mathcal{S}_2$ (contain more than 1000 documents) for which we obtained different results in clustering using original documents; one almost with the good and another with the bad results, respectively.

Preprocessing.

Another preprocessing that we have exerted is in utilizing the relations included in WordNet. We used WordNet 3.1, which all the relations are originally represented in a prolog-readable format [3]. We used four semantic WordNet relations *hypernym*, *member meronym*, *substance meronym*, and *part meronym* and two lexical relations with the operator names s and g representing all the *synsets* (synonyms) of different senses of words and specific gloss for each synset respectively. Indeed, in the preprocessing part for WordNet ontology, we created a new representing format for each of the mentioned semantic relations in which a semantic relation is represented in form of a matrix–stored in a text file–instead of prolog format. Intuitively, each matrix is created by tracing all the predecessor/successor (source/target) concepts for each synset. Simply, the represented matrix for a semantic relation shows the extended scheme of all the super-subordinate structures of which relation. For example, $s_i[1045, 7]$ and $s_j[1045, 5]$ in *hypernym* relation matrix refer to two synsets i and j which are appeared in same row 1045, but synset j with the length of two is a superior for synset i. By using such the matrix for each relation, we further establish an indexing for every synset in which synset $Syn(HypM, y_i)$ returns rows' numbers of matrix $HypM$ (i.e. Matrix created for *hypernym* relations) that synste y_i is occurred in. For example, $Syn(HypM, 103736809) = \{1050516\}$ and $Syn(HypM, 103736970) = \{1052740\ 1052874\}$ indicate rows' numbers of two synsets that first one occurred only in row 50516[4] and second one occurred in rows from 52740 to 52874.

[3] http://wordnet.princeton.edu

[4] To avoid overlapping of the same rows from different relations, four different discriminant numbers have been added to the original rows' numbers.



(a)

(b)

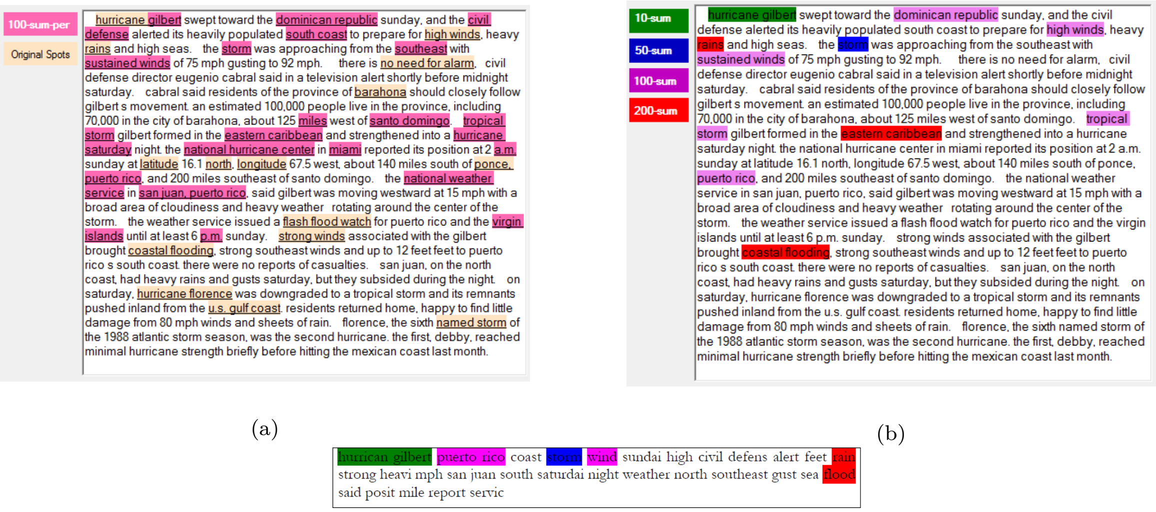
Figure 4-5: (a) Average number of words in summaries created by NG-Rank method with different size of summarization (b) The results of R based on the size of the summaries in DUC2002 and summaries created by NG-Rank method.

4.4 Experimental Results

As previously stated, we first evaluate NG-Rank as a method for extracting salient entities. To this end, we test the quality of extracted salience entities by using DUC2002, which contains a bunch of documents and their summaries manually creating by human. Then we assess the quality of results obtained from the document clustering, after applying our document enriching approach.

4.4.1 Assessing the quality of Salient Entities extracted by NG-Rank method

For a given document d within DUC2002, we produce a set includes words of the spots of the entities appeared in d, namely, $DP(d)$. Moreover, for 5 summaries $SU^{Expert}_{i=1,..,5}(d)$ corresponding to document d, i.e, summaries created by experts, we produce 5 corresponding sets $SP_{i=1,..,5}(d)$ each one includes those words of SU_i which are in common with at least one word included in a spot within $DP(d)$. Simply, we can use elements of SP, considering that in which summary SU^{Expert} they appear, to rank the saliency of their assigned entities. Obviously, an entity which its spot occurs in a 10-words summary, which is a summary over the all the documents with a same topic, significantly, it is more salient than one appear in a 50-words summary. Finally, in order to evaluate NG-Rank method, we created 13



hurricane gilbert swept toward the dominican republic sunday, and the civil defense alerted its heavily populated south coast to prepare for high winds, heavy rains and high seas. the storm was approaching from the southeast with sustained winds of 75 mph gusting to 92 mph. there is no need for alarm, civil defense director eugenio cabral said in a television alert shortly before midnight saturday. cabral said residents of the province of barahona should closely follow gilbert s movement. an estimated 100,000 people live in the province, including 70,000 in the city of barahona, about 125 miles west of santo domingo. tropical storm gilbert formed in the eastern caribbean and strengthened into a hurricane saturday night. the national hurricane center in miami reported its position at 2 a.m. sunday at latitude 16.1 north, longitude 67.5 west, about 140 miles south of ponce, puerto rico, and 200 miles southeast of santo domingo. the national weather service in san juan, puerto rico, said gilbert was moving westward at 15 mph with a broad area of cloudiness and heavy weather rotating around the center of the storm. the weather service issued a flash flood watch for puerto rico and the virgin islands until at least 6 p.m. sunday. strong winds associated with the gilbert brought coastal flooding, strong southeast winds and up to 12 feet feet to puerto rico s south coast. there were no reports of casualties. san juan, on the north coast, had heavy rains and gusts saturday, but they subsided during the night. on saturday, hurricane florence was downgraded to a tropical storm and its remnants pushed inland from the u.s. gulf coast. residents returned home, happy to find little damage from 80 mph winds and sheets of rain. florence, the sixth named storm of the 1988 atlantic storm season, was the second hurricane. the first, debby, reached minimal hurricane strength briefly before hitting the mexican coast last month.

(a)

hurricane gilbert swept toward the dominican republic sunday, and the civil defense alerted its heavily populated south coast to prepare for high winds, heavy rains and high seas. the storm was approaching from the southeast with sustained winds of 75 mph gusting to 92 mph. there is no need for alarm, civil defense director eugenio cabral said in a television alert shortly before midnight saturday. cabral said residents of the province of barahona should closely follow gilbert s movement. an estimated 100,000 people live in the province, including 70,000 in the city of barahona, about 125 miles west of santo domingo. tropical storm gilbert formed in the eastern caribbean and strengthened into a hurricane saturday night. the national hurricane center in miami reported its position at 2 a.m. sunday at latitude 16.1 north, longitude 67.5 west, about 140 miles south of ponce, puerto rico, and 200 miles southeast of santo domingo. the national weather service in san juan, puerto rico, said gilbert was moving westward at 15 mph with a broad area of cloudiness and heavy weather rotating around the center of the storm. the weather service issued a flash flood watch for puerto rico and the virgin islands until at least 6 p.m. sunday. strong winds associated with the gilbert brought coastal flooding, strong southeast winds and up to 12 feet feet to puerto rico s south coast. there were no reports of casualties. san juan, on the north coast, had heavy rains and gusts saturday, but they subsided during the night. on saturday, hurricane florence was downgraded to a tropical storm and its remnants pushed inland from the u.s. gulf coast. residents returned home, happy to find little damage from 80 mph winds and sheets of rain. florence, the sixth named storm of the 1988 atlantic storm season, was the second hurricane. the first, debby, reached minimal hurricane strength briefly before hitting the mexican coast last month.

(b)

hurrican gilbert puerto rico coast storm wind sundai high civil defens alert feet rain strong heavi mph san juan south saturdai night weather north southeast gust sea flood said posit mile report servic

(c)

Figure 4-6: Snapshots of running example that show a document in DUC with (a) all its spots and those appear in per-100-words expert summary; and (b) those spots appeared in 10, 50, 100, and 200-words expert summaries; and (c) its summary created by NG-Rank method.

summaries $SU^{NG-Rank}_{i=1,..,13}(d)$ with different sizes by applying NG-Rank method. Afterwards, for document d, we compare elements of $SU^{NG-Rank}$ and SP depending on the size of the summaries. Figure 4-5.(b) show the obtained results.

In Figure 4-5.(b), vector X indicates the fraction of summarization in each experiment which means how much the original documents are reduced in size, and values on vector Y are computed as follows:

$$R = \frac{|SU^{NG-Rank} \cap SP|}{|SP|} \tag{4.1}$$

Where R could be considered as Recall to evaluate summaries created by NG-Rank method in covering salient entities of documents. As can be seen in Figure 1 the words extracted by NG-Rank algorithm can cover important spots more than 93 percent in the worst case. For example, the words which are extracted by summarizing a document into a summary with one-tenth of its previous size can cover more than 99 percent of important spots which are appeared in 100-words, 93 percent of which appeared in per-100-words, 94 percent of which appeared in 200-words, 95 percent of which appeared in 50-words, and 96 percent of which appeared in 10-words summary created by an expert. The average number of words in summaries generated by NG-Rank method is shown in Figure 4-5.(a). The already determined

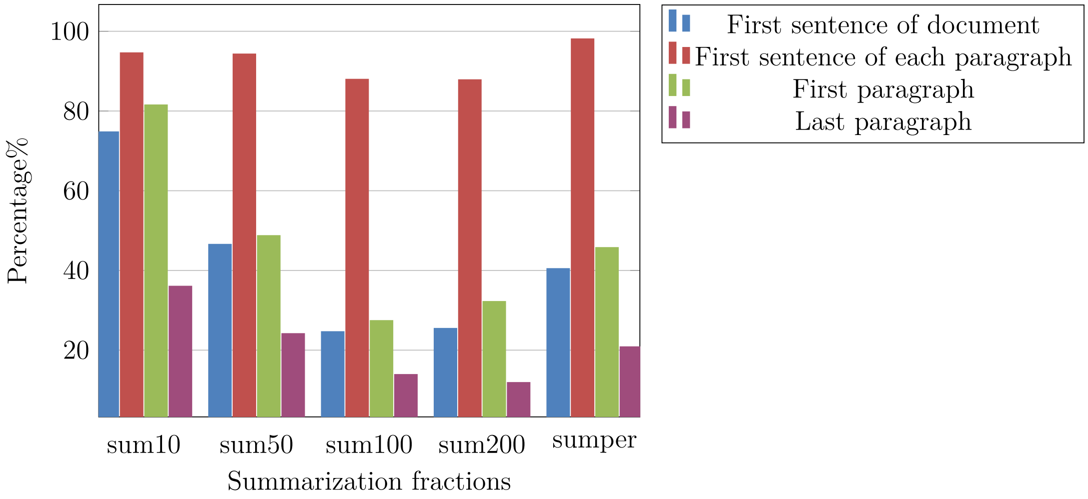


Figure 4-7: The average percentage of occurring important spots in the different sections of DUC documents.

size of summary for the documents can cause low amount in computing the precision of the extracted spots. The average number of extracted spots in expert summaries; 10-words, 50-words, 100-words, 200-words, and 100-per-words are 1.88, 2.12, 1.19, 2.74, and 11.28 respectively.

Since, the summarization method that we used emphasizes *first* and *last* sentences of each paragraph to summarize a document, we also analyzed documents of DUC as a ground truth to indicate capability of extracting most salient linked entities using NG-Rank method, in which phrases appearing in first and last sentences of each paragraph are received higher probability to be extracted. The obtained results are shown in Figure 4-7. We investigated the average percentage of occurring important spots in the different sections: first sentence of document, first sentences of each paragraph, first paragraph of document, and last paragraph of document. As it is indicated in Figure 4-7 the first sentences of each paragraph are remarkable sections of a document in terms of comprising the important spots of the document.

Figure 4-6 shows snapshots of a document in DUC and its summary created by NG-Rank method. The different colors green, blue, violet, red, and pink indicate 10-word, 50-words, 100-words, 200-words, and per-100-words summaries, respectively. In (a), all the spots of original document are in bisque, and spots appeared in 100-words summary (by experts) are in pink. In (b), spots of entities appeared in 10, 50, 100, and 200-words summaries (by experts) are in green, blue, violet, and red, respectively. As it can be seen in (c), the created summary by NG-Rank method covers significant part of the important spots.

For the document in Figure 4-6, different sizes of summaries written by experts are as follows:

- 10-words summary:

  Record Intensity Hurricane Gilbert Causes Havoc In The Caribbean.

- 50-words summary:

  Hurricane Gilbert, a category 5 storm, caused death, massive flooding and damage as it moved through the Caribbean Islands and on to the Yucatan Peninsula. After skirting several island nations, it caused major death and destruction in Jamaica. It then pummeled the Yucatan Peninsula before moving out to sea.

- 100-words summary:

  Tropical Storm Gilbert strengthened into a hurricane on Saturday night, September 10th in the eastern Caribbean. It tracked westerly at about 15mph while building in intensity. After skirting southern Puerto Rico, Haiti and the Dominican Republic, it hit Jamaica with high winds and torrential rains, destroying 100,000 of the countries 500,000 homes and taking nineteen lives. It then passed over the Cayman Islands before slamming into the Yucatan Peninsula causing heavy damage in the Cancun and Cozumel regions. Gilbert is the most intense hurricane ever recorded with a record low barometric pressure of 26.31 inches and sustained winds of 179mph, gusting to 218mph.

- 200-words summary:

  Tropical Storm Gilbert strengthened into an eastern Caribbean hurricane on Saturday night, September 10, 1988. Government officials of the island nations in its westerly path issued alerts, warnings and orders to evacuate their southerly-exposed coastal areas. Puerto Rico, Haiti and the Dominican escaped with only some coastal flooding and a few deaths before Gilbert continued on to strike Jamaica with 110mph winds and torrential rains. The storm, now one of the largest systems seen in the Caribbean for a long time, slowly traversed the length of the island causing massive damage and nineteen deaths. 100,000 of Jamaica's 500,000 homes were destroyed leaving 500,000 people homeless. The storm then passed over the Cayman Islands located south of Cuba. Cuba and our American military bases there both avoided the brunt of the storm before it slammed full force into the Yucatan Peninsula. By now Gilbert had strengthened into the most severe hurricane ever recorded with barometric pressure of 26.31 inches and sustained winds of 179mph, gusting to 218mph. Relief efforts for the heavily damaged Cancun and Cozumel areas were hindered due to massive flooding and power and phone outages. Scientists do not fully understand why some minor tropical storms strengthen into hurricanes, while others do not.

- 100-sum-per summary:

Table 4.1: Top-8 clustering results on sub-set $\mathcal{S}_2$ with different parameters for function $\mathcal{F}$, using topic modeling to generate the RWs to be expanded.

	Purity of Cluster					
Representation	Cluster 0	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Total Purity
P^2TO^*	0.958	1.0	0.678	0.624	0.949	**0.768**
PTO	1.0	0.578	0.542	0.726	1.0	0.683
$\{PY\}^2TO$	0.961	0.961	0.988	0.405	0.836	0.681
$PYCT^2UO$	0.988	0.642	0.735	0.437	0.958	0.656
$PYCT^2O$	0.987	0.658	0.739	0.426	0.958	0.653
$PYCTUO$	0.987	0.712	0.730	0.415	0.958	0.652
$\{PYC\}^2TUO$	0.983	0.5	0.726	0.439	0.958	0.610
TO	0.574	0.542	1.0	0.430	0.923	0.566

$^*P = ParentSyns,\ Y = SynsetSyns,\ C = ChildSyns,$
$T = RWs\ generated\ by\ Topic\ modeling,\ U = Summary,\ O = OriginalDocs$

Tropical Storm Gilbert in the eastern Caribbean strengthened into a hurricane Saturday night. The National Hurricane Center in Miami reported its position at 2 a.m. Sunday to be about 140 miles south of Puerto Rico and 200 miles southeast of Santo Domingo. It is moving westward at 15mph with a broad area of cloudiness and heavy weather with sustained winds of 75mph gusting to 92mph. The Dominican Republic's Civil Defense alerted that country's heavily populated south coast and the National Weather Service in San Juan, Puerto Rico issued a flood watch for Puerto Rico and the Virgin Islands until at least 6 p.m. Sunday.

Table 4.2: The best result of clustering on $\mathcal{S}_2$ obtained with the representation function $\mathcal{F} = P^2SO$ using topic model for generating RWs.

Cluster	Bus	Enter	Polit	Sport	Tech	Purity
Cluster 0	1	2	0	0	69	0.958
Cluster 1	0	0	0	25	0	1.0
Cluster 2	0	114	1	52	1	0.678
Cluster 3	19	14	98	19	7	0.624
Cluster 4	94	1	0	0	4	0.949
Total Purity						**0.768**

To evaluate the effectiveness of NG-Rank method–along with linking entities–for generating the Required Words (RWs) to be expanded, we investigate the obtained result of clustering in which RWs are generated by using *topic modeling* [5]. To this end, we run a topic model, namely, Latent Dirichlet Allocation (LDA) on the collection in which the

Table 4.3: Clustering results on sub-set $\mathcal{S}_1$: (a) using original documents plus all the entities and their categories (NEKW method); (b) using original documents plus salient entities and their categories; (c) using original documents plus all the entities and their spots (d) using original documents plus salient entities and their spots; (e) using original documents plus all the spots of the entities; (f) using original documents plus the spots of salient entities.

Cluster	Bus	Enter	Polit	Sport	Tech	Purity
Cluster 0	0	0	78	0	0	1.0
Cluster 1	87	10	4	1	4	0.821
Cluster 2	2	88	5	1	10	0.830
Cluster 3	10	2	7	0	85	0.817
Cluster 4	1	0	6	94	0	0.931
Total Purity						**0.873**

(a)

Cluster	Bus	Enter	Polit	Sport	Tech	Purity
CCluster 0	0	0	77	0	0	1.0
Cluster 1	92	4	4	0	10	0.836
Cluster 2	1	96	13	2	9	0.793
Cluster 3	7	0	3	0	79	0.888
Cluster 4	0	0	3	94	1	0.959
Total Purity						**0.885**

(b)

Cluster	Bus	Enter	Polit	Sport	Tech	Purity
Cluster 0	0	0	74	0	0	1.0
Cluster 1	97	2	16	0	9	0.782
Cluster 2	0	96	5	16	3	0.780
Cluster 3	3	2	4	0	87	0.906
Cluster 4	0	0	1	80	0	0.987
Total Purity						**0.877**

(c)

Cluster	Bus	Enter	Polit	Sport	Tech	Purity
Cluster 0	0	0	80	0	0	1.0
Cluster 1	97	1	12	0	10	0.808
Cluster 2	0	96	3	13	3	0.835
Cluster 3	3	3	4	0	86	0.896
Cluster 4	0	0	1	83	0	0.988
Total Purity						**0.893**

(d)

Cluster	Bus	Enter	Polit	Sport	Tech	Purity
Cluster 0	0	0	79	0	0	1.0
Cluster 1	87	8	3	0	3	0.821
Cluster 2	2	89	6	1	10	0.830
Cluster 3	10	3	7	0	86	0.817
Cluster 4	1	0	5	95	0	0.931
Total Purity						**0.881**

(e)

Cluster	Bus	Enter	Polit	Sport	Tech	Purity
Cluster 0	0	0	84	0	0	1.0
Cluster 1	93	5	3	0	10	0.838
Cluster 2	1	94	8	3	5	0.847
Cluster 3	6	1	3	0	84	0.894
Cluster 4	0	0	2	93	0	0.979
Total Purity						**0.905**

(f)

number of topics equals the number of clusters. Afterward, for each document, Top-N words of the topic with highest proportion, which are common in both the topic and document, are extracted as the RWs in expansion phase. The value of N is the average number of spots specified by NG-Rank method in documents. The obtained results of clustering with different representations are shown in Table 4.1 and the details of the best result are shown in Table 4.2.

4.4.2 Assessing the Clustering Improvement due to Document Enriching

In previous experiments, we assessed the capability of NG-Rank summarization method in order to extracting salient entities. In this section we evaluate our algorithm (SEED) in improving the quality of clustering. The algorithm adopted for clustering documents is K-Means, while the vectorial representation of documents is based on a classical *tf-idf* weighting of terms, and the measure of similarity between two vector is Cosine similarity. Specifically, we utilized *RapidMiner*[5], which is an integrated environment for analytics, also providing tools for text mining. We evaluate our approach in two parts:

In the first part, we first evaluate the effectiveness of using only the salient entities–along with their categories and spots–in improving the quality of clustering. We then compare its results with the state-of-the-art approach NEKW [8] which represents a trade-off between keywords of a document and entities features by taking into account all the entities of a document. In this evaluation, WordNet is not used for expanding latent information, and in the same way with NEKW, only features of salient entities are used. Table 4.3 and Table show the results.

In the second part, we evaluate the quality of clustering after using WordNet in order to expanding the spots of salient entities. Indeed, we test and evaluate clustering with/without applying SEED, to show the improvements in clustering purity due to document enriching based on salient entities. The results are shown in Table 4.6 and Table 4.7. As stated in section 4.3, for testing clustering quality, we used BBC NEWS articles. From this dataset, we randomly selected two sub-sets $\mathcal{S}_1$ and $\mathcal{S}_2$, where each one contains about 500 typical documents. In more detail, from every labeled class of original dataset, we randomly selected about 100 documents for each sub-set sampling.

Table 4.4 shows 14 partitions (instances) $\{\mathcal{F}^h\}_{h=1,2,..,14}$ of function $\mathcal{F}$. Indeed, every partition is the result of clustering on collection $\mathcal{D}^h = \{D_1^h, D_1^h, .., D_m^h\}_{h=1,2..,14}$. Specifically, first column indicates the representation of documents and other columns are the results of clustering using this representation. The range of parameters $\alpha, \beta, \gamma, \varepsilon, \delta, \eta$ in $\mathcal{F}^h$ is limited by empirical value v, which is computed for every parameter as follows:

$$v_{ms_i} = \lfloor \frac{\sum_{D_i \in \mathcal{D}} LWS(D_i)}{\mu \times \sum_{ms_i \in SES(D_i)} LWS(ms_i)} \rfloor$$

[5] https://rapidminer.com/products/studio/

Table 4.4: A set of partitions with different parameters for function $\mathcal{F}$, which is used in consensus part of cluster ensemble on subs-set $\mathcal{S}_1$.

	Purity of Cluster					
Representation	Cluster 0	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Total Purity
$PYCE^{*}S^{2}O$	1.0	0.925	0.827	0.843	0.913	0.897
$PYCUES^{3}O$	0.947	0.855	0.894	0.943	0.898	0.905
$\{PYC\}^{2}ES^{5}UO$	0.989	0.868	0.902	0.898	0.940	0.917
$P^{4}Y^{2}C^{4}ES^{11}O$	1.0	0.690	0.955	0.836	0.881	0.850
$\{PYC\}^{5}ES^{11}U^{2}O$	0.931	0.756	0.924	0.914	0.920	0.880
$\{PYC\}^{4}EP^{7}U^{2}O$	0.934	0.840	0.917	0.898	0.923	0.901
$\{PYC\}^{3}E^{2}S^{7}UO$	0.937	0.861	0.935	0.917	0.941	0.917
$P^{2}\{YC\}^{3}E^{2}S^{6}O$	0.957	0.908	0.835	0.905	0.950	0.909
$PYCESUO$	1.0	0.978	0.874	0.817	0.979	0.921
$PYCES^{2}O$	1.0	0.978	0.883	0.803	0.979	0.919
$\{PYC\}^{2}ESUO$	1.0	0.927	0.941	0.814	0.979	0.927
$\{PYC\}^{2}ES^{2}UO$	1.0	0.927	0.950	0.816	0.970	0.927
$\{PYC\}^{3}ES^{2}UO$	0.979	1.0	0.896	0.773	0.975	0.911
EUO	1.0	0.937	0.9	0.768	0.979	0.907

$^{*}E = LinkedEn,\ S = Spots$

Where ms_i is the corresponding multiset for the parameter, μ is average of the minimum number of words' occurrence in documents, and $LWS(x)$ returns the length of text x after stop words removal. In our experiments on dataset BBC, the values of v corresponding to the elements of set SES is $\{5, 5, 5, 2, 12, 2\}$, and the size of summaries produced by NG-Rank method is a third of original documents.

In order to performing ensemble clustering, considering the values of v, several representations of documents are generated by combining the elements of the SES. Top-N results of clustering (with highest *total purity*) obtained by using these representations are selected to be used in ensemble clustering. Table 4.4 shows top-14 results of clustering selected among the 50 clustering results obtained by using 50 different document representations. The result of ensemble clustering is shown in Table 4.6.(b). It can be observed in Tabel 4.6 that we have a improvement in result of each cluster, for example, the result of cluster 2 is improved by %16. Furthermore, in Table 4.7, we can see a significant improvement in result of cluster 4 (%25) and cluster 3 (%18). The improvement percentage of their total purity on $\mathcal{S}_1$ and $\mathcal{S}_2$ is about %8 and %15. In addition, we can see more improvements by comparing the result of Table 4.5, in which documents are represented by original TF.IDF

Table 4.5: Clustering results: (a) original documents of sub-set $\mathcal{S}_1$; (b) original documents of sub-set $\mathcal{S}_2$

Cluster	Bus	Enter	Polit	Sport	Tech	Purity
Cluster 0	0	0	73	0	0	1.0
Cluster 1	84	8	1	0	5	0.857
Cluster 2	13	91	19	0	12	0.674
Cluster 3	3	1	4	1	82	0.901
Cluster 4	0	0	3	95	0	0.969
Total Purity						**0.858**

(a)

Cluster	Bus	Enter	Polit	Sport	Tech	Purity
Cluster 0	2	37	14	19	63	0.466
Cluster 1	0	0	0	66	0	1.0
Cluster 2	0	83	0	10	0	0.892
Cluster 3	66	10	84	1	13	0.483
Cluster 4	46	1	1	0	5	0.868
Total Purity						**0.656**

(b)

Table 4.6: Clustering results of dataset $\mathcal{S}_1$: (a) using *NEKW* method; (b) using SEED method.

Cluster	Bus	Enter	Polit	Sport	Tech	Purity
Cluster 0	0	0	78	0	0	1.0
Cluster 1	87	10	4	1	4	0.821
Cluster 2	2	88	5	1	10	0.830
Cluster 3	10	2	7	0	85	0.817
Cluster 4	1	0	6	94	0	0.931
Total Purity						**0.873**

(a)

Cluster	Bus	Enter	Polit	Sport	Tech	Purity
Cluster 0	0	0	86	0	0	1.0
Cluster 1	92	0	5	0	0	0.948
Cluster 2	0	97	0	0	4	0.960
Cluster 3	8	2	7	0	95	0.848
Cluster 4	0	1	2	96	0	0.970
Total Purity						**0.941**

(b)

Table 4.7: Clustering results of dataset $\mathcal{S}_2$: (a) using *NEKW* method; (b) using SEED method.

Cluster	Bus	Enter	Polit	Sport	Tech	Purity
Cluster 0	0	0	0	13	47	0.783
Cluster 1	0	1	0	49	1	0.961
Cluster 2	3	117	1	0	13	0.873
Cluster 3	20	4	51	15	2	0.554
Cluster 4	91	9	47	19	18	0.494
Total Purity						**0.681**

(a)

Cluster	Bus	Enter	Polit	Sport	Tech	Purity
Cluster 0	9	1	1	0	45	0.804
Cluster 1	0	0	0	77	0	1.0
Cluster 2	0	107	0	0	0	1.0
Cluster 3	6	12	78	19	6	0.645
Cluster 4	99	11	20	0	30	0.619
Total Purity						**0.779**

(b)

values.

5

Cluster Labeling

In this chapter, we explore and categorize cluster labeling techniques, providing a thorough discussion of the relevant state-of-the-art literature. Cluster labeling techniques aim to better characterize groups (clusters) of documents according to their specific content, and they try to achieve this goal by assigning some kind of labels, i.e., textual entities describing content of a cluster.

Clustering algorithms have been introduced to automatically group similar documents into subsets (clusters), thus allowing to give them a better characterization; in turn, this allows users to have a better understanding of the documents content. Nonetheless, while clustering techniques represent an important tool to categorize documents, they possess intrinsic limits when it comes to give a deeper understanding of the documents content to human users. This is where cluster labeling techniques come into the scene.

Different kinds of cluster labeling methods exist, depending on the approach used to infer cluster labels. Mainly, we can speak about two classes of methods: *direct* methods and *indirect* methods.

Direct methods try to *directly* extract labels from the content of documents making up a cluster. For example, cluster labels can be extracted using different feature selection methods [43], picking up the most frequent terms occurring in a cluster, or using top weighted cluster centroid's terms [16]. The main drawback of these methods consists in that they may not produce an optimal solution whenever meaningful labels cannot be extracted from the documents making up a cluster. For example, let us consider a cluster of documents discussing about *printmaking*: by looking just at the content of individual documents, it is possible that the set of labels extracted do not contain the topic to which the documents belong; For example, for a cluster with the set of candidate labels *engraving, etching, lithography, steel engraving* selecting *printmaking*, which can be extracted from an external ontology, as the label of the cluster would be more meaningful rather than selecting one of the represented candidates. A direct method may extract the set of candidate labels {*engraving, etching, lithography, steel engraving*}, which obviously do not contain the *common denominator* connecting the documents of the cluster, *printmaking*.

In order to tackle this issue, another group of cluster labeling methods consider the usage of external resources (e.g., Wikipedia) to assign labels to clusters [9, 69]. Indeed , the hypothesis behind these approaches is that the describing labels for a cluster could be provided through an external resources. These resources may be an encyclopedia like Wikipedia or a lexical ontology like WordNet. In the above example, *printmaking* can be extracted by using the semantical relationships, which are exist in the lexical ontology of WordNet. This class of methods which apply external resources for cluster labeling are called *indirect* methods.

We also devote a part of this chapter to review relevant, state-of-the-art literature related to *topic labeling*. The aim of topic labeling is to label topics, by means of some topic modeling algorithm, in order to provide brief topic summaries that can be quickly and easily understood from human users.

Since a cluster of documents are often summarized as a collection of the most significant words they contain which have to be labeled, and in topic labeling, a topic model represented by a set of words which have to be labeled, we therefore consider the methods in topic labeling in this chapter as well.

5.1 Preliminaries

In this section, we introduce some preliminary notions which are used in cluster labeling techniques.

5.1.1 Type of Labels

In the context of cluster labeling, several kinds of labels can be used to characterize clusters of documents. In the following, we detail the three main categories of labels we can find in the literature, namely, the *flat labels*, the *hierarchical labels* and the *graph based labels*.

Flat Labels

A flat label represents a list of terms (labels) extracted from a set of documents or an external ontology, depending on the specific approach used.

Often, flat labels are derived using statistical techniques borrowed from the domains of feature selection and reduction.

Indeed, users to understand and to interpret the content of the documents should only rely on an unstructured keyword-based model of labels.

Most of cluster labeler represents flat labels to describe a cluster [42, 24, 72, 73, 46].

Hierarchical Labels

The hierarchical model is a popular model used to represent documents at different levels of detail, while allowing to navigate through them as well.

Informally, we can describe a hierarchy as a tree, where the root node represents the document in its whole, while internal nodes and leaves represent specific parts of the document. As one descends within the hierarchy, nodes provide details about smaller and smaller parts of the document.

Compared with flat labels to describe a collection, hierarchies represent a summary of collection instead of a few labels to convey content of collection to user. It is useful in topic models, and in related work is used in a question answering system [36].

When applying hierarchical models in the context of cluster labeling, labels can be naturally arranged in hierarchies thanks to the order imposed by hierarchical models.

For instance, [73] proposes a cluster labeling approach based on hierarchical labels. More precisely, the authors consider frequent phrases in the text for cluster labeling and is based on the hypothesis that a good descriptor should occur relatively frequent in the parent cluster, but occur very frequent in the self cluster. Which means in hierarchical cluster scheme, a good descriptor for the cluster as well as helping user to understand content of the cluster, should also differentiate the cluster from its siblings and its parent cluster. To measure the descriptiveness of a phrase in a cluster, the approach for every phrase first computes three features; a) phrase length which is the number of terms in the phrase, b) document frequency in both self cluster S and parent cluster P (DF_s, DF_p), and c) term frequency inverse document frequency in both self cluster S and parent cluster P ($TFIDF_s$, $TFIDF_p$). Further, four *rankings* are computed for every phrase based on the features $\frac{DF_s}{|S|}$, $\frac{DFp}{|p|}$, $TFIDF_s$, $TFIDF_p$. Finally, to touch the basic hypothesis for a good descriptor, authors boost in ranking by computing difference of ranks $(\log(r(\frac{DF_s}{|S|})) - \log(r(\frac{DFp}{|p|})))$ and $(\log(r(TFIDF_p)) - \log(r(TFIDF_s)))$. Using log-scale is to emphasis significant rank changes from parent to self cluster. Combination of all explained features generates final linear model for each phrase ($DScore_P$). Each feature in $DScore_P$ has a weight which is estimated using linear regression and training data. Each label candidate is sorted by its descriptive score. Table 5.1 categorize some references based on the type of labels they applied.

Graph Based Labels

An important problem that arises when arranging documents in clusters is that each cluster should be associated with some kind of information that can be naturally - and clearly - interpreted by humans. As we mentioned previously, the goal of cluster labeling is exactly to provide such kind of information. One of the limitations of approaches based on flat labels is that they cannot convey meaningful relationships between terms, since they just label clusters according to a list of characterizing terms. Approaches based on graph-based labels try to fill this gap, indeed, such approaches try to represent a graphical structure for labels of a cluster. The relationship between the terms in such structure is indicated semantic relevance, co-occurrence, or any concept of relevance that could be investigated between two terms. Furthermore, the included terms in a relationship could be both from candidate labels, or one from the candidate labels and another from either any terms of cluster or an external ontology.

One of the most notable works using graph-based labels is [63]; in this work, the authors propose a graph-based model to capture relationships between terms.

By generating the graph based labels, the algorithm allows users to explore the relationships between the terms in the clusters and thus better interpret the content of cluster. The algorithm first builds a document-term matrix by reading through documents of cluster, and then clusters the document-term matrix by using a K-means type algorithm and extracts top-10 terms of each cluster centroid (based on $tf - idf$). In the next step, the algorithm constructs a term-term matrix, in which each cell is the number of times that two specific terms co-occurs within a certain window of text (a sentence, a paragraph, etc.).

The algorithm construct a similarity matrix by multiplying term-term matrix with its transpose; indeed, each cell of similarity matrix presents cosine similarity between two different terms of cluster. Finally, by using similarity matrix and top-N terms of centroid matrix, the algorithm generates a directed graph, in which top components of the cluster centroid are the roots of graph which connect to other terms of cluster (by considering a threshold in the selection) by traversing the term similarity graph. A sample of graph is showed in Figure 5-1.

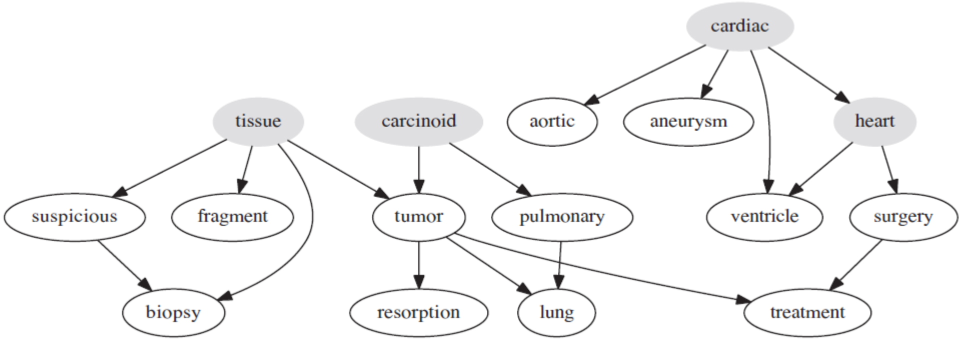


Figure 5-1: Graph representation of a cluster: the nodes in gray are the terms associated with the top components of the cluster centroid. The white nodes are the nodes that are reached from these root terms by traversing the term similarity graph.

Type of label	References
Flat labels	[73, 24]
	[42, 36, 46]
Concise labels (category label)	[57, 13, 16]
	[70, 23]
	[7, 30, 9]
Hierarchical labels	[73]
Graph based labels	[63]
Mixed labels	[62]

Table 5.1: Categorization of references based on the type of labels applied.

5.1.2 Feature Selection

The general goal of feature selection methods is to select a subset of relevant features, from an overall set of features, to be used in some subsequent model construction process. For what is related to the specific domain of cluster labeling, features can represent single words, n-grams or phrases; hence, feature selection methods are applied to reduce the noise coming from uninformative features within the training set [43]. Several cluster labeling approaches presented in this chapter rely on feature selection methods; in this section we provide a brief, yet informative overview of them.

Mutual Feature Selection

The goal of *mutual feature selection* methods is to compute a utility measure for each term of a vocabulary, in order to use only the most relevant ones during a subsequent classification. This measure is realized by means of the *expected mutual information* (MI) between a term t and a class c. Indeed, MI measures how much the presence or absence of t increases the likelihood of having a correct classification decision for c. In the end, only the top-k terms having the highest MI measure are retained for the subsequent classification process.

Formally, given a document D, a term t, a class c, a random variable $U = e_t$ which expresses the presence ($e_t = 1$) or absence ($e_t = 0$) of t in a D, and a random variable $C = e_c$ which expresses the fact that D belongs to c ($e_c = 1$) or not ($e_c = 0$), we express the mutual information between t and c as:

$$I(U;C) = \sum_{e_t \in \{1,0\}} \sum_{e_c \in \{1,0\}} P(U = e_t, C = e_c) \log \frac{P(U = e_t, C = e_c)}{P(U = e_t)P(C = e_c)'} \tag{5.1}$$

Chi-square (χ^2) Feature Selection

Another common feature selection method relies on the *chi-square* (χ^2) test. In statistics, the χ^2 test is used to determine whether there is a dependency between two variables. In the context of text classification and cluster labeling, these two variables are represented by the occurrence of a term t and a class c in a document D. In the following we introduce its formal definition.

First, given a document D, a term t and a class c, we define $N_{e_t e_c}$ to be the *observed* frequency of those cases where a term t is present/absent ($e_t = \{0, 1\}$) in D *and* D is correctly/wrongly assigned to a class c ($e_c = \{0, 1\}$); similarly, $E_{e_t e_c}$ represents the *expected* frequency. Then, the χ^2 test is defined as:

$$X^2(D,t,c) = \sum_{e_t \in \{0,1\}} \sum_{e_c \in \{0,1\}} \frac{(N_{e_t e_c} - E_{e_t e_c})^2}{E_{e_t e_c}} \tag{5.2}$$

Intuitively, high scores on x^2 indicate that the null hypothesis of independence is rejected, which in turn indicates that the occurrence of term t and class c are dependent; if this condition applies, the feature is then considered during the subsequent text classification process.

Frequency-based Feature Selection

The last class of feature selection methods which can be used by cluster labeling approaches is the *frequency-based* methods. Intuitively, these methods consider the *frequency* of features to select only the most frequent ones. Frequency can be mainly defined in terms of *document frequency* (DF) or *collection frequency* (CF).

Given a term t, its *document frequency* can be defined as the number of documents, in a class c, where t appears.

In the *collection frequency* case, the goal is to select the subset of features which occur frequently in a class (cluster). To this end, the idea is to count all the instances of a feature in a collection (multiple repetitions in documents of a collection are allowed) for a feature or document frequency (the number of documents in the class c that contain the term t), This subset of features has no specific information about the class (like days of week), and should be discarded. But on the other hand, if we compute the collection frequency values by the inverse document frequency (CFIDF) for a feature, then the top ranked values indicate the features that have more specific information about the class, and should be selected. TFIDF is another method with the same aim to select most informative features.

Final considerations

The mutual feature selection method and the chi-square feature selection method are supervised methods, while the last one belongs to the class of unsupervised methods. All these methods can be used in the context of *differential cluster labeling* to select cluster labels by comparing the distribution of terms in one cluster with that of other clusters. In order to using feature selection to label clusters, clustered documents are considered as the training classified set, and the aim is to select a subset of features as informative features for each cluster.

5.2 Direct Cluster labeling

Label(s) of a cluster may be directly extracted from the content of the cluster's documents [16, 24, 23, 63, 72, 70, 73, 46]. In this case, algorithm try to identify important terms in the cluster content that best represent the cluster topic. The algorithms of this approach are divided to two Differential cluster labeling and Cluster-internal labeling [43].

Differential cluster labeling selects labels of a cluster by identifying important terms in the cluster content that characterize the cluster in contrast to other clusters [16]. To this end, statistical techniques for feature selection can be used for differential cluster labeling. Important terms can be identified by using statistical feature selection techniques such as Mutual Information (MI), Information Gain, X^2-test [43], and Most Frequent Terms. For example, [23] use a modified version of the information gain measure to select words that are most representative of its contents and are least representative of the contents of the other cluster. [63] considers a list of terms with high weights in the centroid of the cluster as important terms; [73] extracts most frequent terms in the cluster. In [73] important terms are selected based on the term frequency and document frequency in the cluster and in general English language. There are some other approaches which focus on specific domain, like news articles, and consider specific term extraction, like named entities [72], as important terms.

Another family of algorithms are based on the idea of cluster-internal labeling, where labels of a cluster are selected by taking into account only information related to the cluster, ignoring the other clusters. Selecting the titles of documents which are mostly close to the cluster centroid as labels of cluster is an approach in cluster-internal labeling [43]. The Scatter/Gather application [16], which is one of the first approaches that consider cluster labeling, presents two algorithms, Buckshot and Fractionation, which can appropriately cluster a large number of documents within a time tolerance acceptable for user interaction. Here, their approach selects top-k terms with maximal weight from the cluster centroid as digest of clusters. [63] uses a flat clustering algorithm (k-means) to cluster documents. Their approach uses the clustering results to create a centroid matrix ,where each matrix column represents the centroid vector associated with a cluster. Finally, the approach by using this matrix and then finding meaningful relationships between each centroid vector of this matrix and cluster internal terms create a graph representation for label of each cluster.

There are some other works that combine intra-cluster and inter-cluster term extraction. In [23] authors present an approach in which candidate words for each cluster are selected by means of a modified version of the information gain (IG_m), this allows the selection of those words that are most representative of its contents and are least representative of the contents of the other clusters. Finally, in order to construct plausible labels, rather than simply using the list of the-scoring words (i.e. the ones that maximize IG_m), the approach looks within the titles of the returned web pages to look for a substring that best matches the selected top-scoring words.

5.3 Indirect Cluster labeling

Label(s) of a cluster may be extracted by using external resource. Which means that meaningful labels associated with a cluster may not occur in cluster documents. There are several approaches that tried to label a cluster by relying on external relevant label sources, e.g, Wikipedia's categories [9], Dbpedia's graph [30]. One of the significant reasons that brought researchers to rely on an external ontology is the observation that: When a human labels a document, she employs semantic relationships between the most important terms of the document and a pre-knowledge-base to select meaningful labels for the document. However, the meaningful proper label may not exist in the document. [9] illustrates an example in which the authors compare results of two different labeling methods: the JSD selection method, which is a symmetric version of the Kullback-Leibler divergence [10] which tries to maximize the distance between any cluster and the rest of the collection, and an alternative strategy which uses Wikipedia as an external ontology. The first method extracts set of top-5 important terms for six open directory project (ODP), and show that while the list of important terms fairly represents the content of the categories, these terms can serve as appropriate labels only for a few categories. On the other hand, Wikipedia's labels agree with human annotated labels much more. For example, top-5 important terms extracted by JSD for Electronics category of ODP are: voltage, high voltage, circuit, laser and power supply whereas by using Wikipedia ontology in this case, we can see label Electronics in top-5 labels that is much more meaningful for a user than top-5 JSD important terms.

In case of applying the ontology to describe a document or a cluster of documents, there are two major works; the first one indexes and classifies the documents by tagging each document with relevant terms from a well-defined ontology that represents semantic concepts, which are significant to describe the document. Some examples of these ontologies are the ACM Computing Classification System (ACM's CCS), which is used for the classification and indexing of the published literature of computing [14], Unified Medical Language System (UMLS) for documents related to medicine and SmartIndex tags for labeling News documents. Thirunarayan et al, in [70] present a simple technique to construct and select good cluster labels in context of News documents obtained in response to search queries involving entities (EN) and events (EV). The application in this work extracts a well-supported sentence, which contains phrasal references to EN and EV, from the cluster documents. The phrases corresponding to an entity or an event can be obtained from the domain knowledge used to stamp the documents with metadata terms (from a well-defined ontology) that reflect and abstract the document's content. After stamping phrases in sentences, a well-supported sentence is selected as a label by maximizing the number of documents that support the sentence, by maximizing the degree of overlap with a sentence in each document and by minimizing its length. The contribution of this work in using an external ontology is to determine and extract metadata terms from each sentence of a document and to use them instead of the document content.

Using the pre-designed ontology (like ACM's CCS) has many disadvantages [69].Once that it done, it must be maintained and modified, an important process in domains where the underlying concepts are evolving rapidly. ACM's CCS, for example, undergoes periodic reorganization and redesign and yet as a classification of computer science concepts, it always seems to be out of date or even quaint. As another problem, consider the process a person must follow in assigning ontology terms to a document. She has to be familiar with all of the possible choices or have some way to browse or search through them. She has to understand what each of the terms means, either the original meaning intended by the ontology designer or the possibly different current meaning as used by her community. Finally, she has to select the best set of terms from among the many relevant choices the ontology may present to her.

Many Web 2.0 systems have allowed users to tag documents and Web resources with terms without requiring them to come from a fixed vocabulary. In a social media context (e.g., del.icio.us or Flickr) an implicit ontology of tags can emerge from the community of users and subsequently influence the tags chosen by individuals, reinforcing a notion of a common ontology developed by the community. The use of an implicit ontology emerging from the tagging choices of a community of individuals solves some of mentioned problems, but also has significant disadvantages. Zareen et al. in [69] explain that Wikipedia as a emergent ontology has many advantages: it is broad and fairly comprehensive, of generally high quality, constructed and maintained by tens of thousands of users, evolves and adapts rapidly as events and knowledge change, and free and "open sourced". Moreover, the meaning of any term in the ontology is easy for a person to understand from the content on the Web page. Finally, the Wikipedia pages are already linked to many existing formal ontologies though efforts like DBpedia [3] and Semantic MediaWiki [33].

One of the first work used Wikipedia as an external ontology is [69]. The approach presented in this work tries to predict a common or general concept covering all relevant documents. The algorithm applies Spreading Activation technique which has been widely adopted for associative retrieval [15]. The idea in associative retrieval is that it is possible to retrieve relevant documents if they are associated with other documents that have been considered relevant by the user. The algorithm consider titles of Wikipedia article as concepts, links between articles as links between concepts and Wikipedia categories as generalized concepts. Consider a set of related documents, algorithm for each document in the set retrieval top N matching (based on cosine similarity) Wikipedia articles. The results of this retrieval generate initial activation nodes for spreading activation in category links graph. Thus algorithm starts with a set of activated nodes and in each repeat (Pulse) the activation of nodes is spread to associated nodes. By creating the category links graph and the article links graph, the algorithm implements three different methods to predict a common concept. During method one algorithm for each document in the relevant document set gets top N matching Wikipedia articles based on cosine similarity document and each article. For each article, Wikipedia category of that article is extracted, and by two simple scoring schemes scores them (all extracted Wikipedia categories). The top N Wikipedia categories resulted in prior method create initial set of activated nodes in the category links graph. During two last methods algorithm do some filtering and after k pulses of spreading activation, the category nodes are ranked based on some Activation Function.

As you can see common concept (label) for a set of relevant documents (cluster) extracted from an external ontology (Wikipedia) and might not occurs in any document. The results of experiments in [69] are satisfactory. For example, for a set document related to Genetics by considering only the scored categories in method 1 the prediction common concept is “Genetics”, and in case of spreading activation with 2 pulses the common concept is “Biology” further with three pulses the common concept is “Nature” which is an even broader concept than biology.

5.4 Supervised and Unsupervised Cluster labeling

In this section we try to individualize each cluster labeling approach with its supervised/unsupervised implicated methods.

Supervised learning.

In supervised learning or classification learning, there is supervision during the learning process. The supervision means the data labeled with pre-defined classes (it is done by human expert). Indeed, a learning algorithm is presented with a set of already classified, or labeled, examples. This set is called the training set.

Unsupervised learning.

In unsupervised learning there is no supervision during the learning process. No supervision means that there is no human expert who has assigned documents to classes. Unsupervised algorithms do not rely on a training set of labeled examples for building a model. Clustering is the most common form of unsupervised learning.

The approaches which are investigated in this chapter first apply either a supervised or unsupervised algorithm to classify textual data (depend on studied methods; documents, web pages, etc.) in relevant clusters, and next try to label each cluster. Some of approaches select top-N terms extracted in feature selection step of their classification [24]. On the other hand, some of the presented approaches utilize additional step(s) to label a cluster. In both kinds of methods, selection of representative labels for clusters depends of the cluster which is generated by either supervised or unsupervised methods. Thus, we address supervised and unsupervised approaches in cluster labeling according to the methods that are used to select labels, and also the methods that are used to yield clusters.

5.4.1 Supervised labeling

One group of the cluster labeling approaches utilize training data during either their clustering part or extracting labels. The methods included in this group, utilize a ground-truth dataset in which there are labeled classes implemented by human experts. There are several works that utilize supervised methods to label clusters. [73] represents 5 descriptive scores, which are based on 5 features for a phrase within a cluster, to measure amount of its utility as a label, and then by combining every feature into one descriptive score produces a linear model. The weights of each feature within the linear model are estimated using linear regression and training data. In order to train the linear regression model, since the correct descriptive score is not known for each label candidate, the descriptive score of a label candidate is estimated. To this aim, each label candidate's descriptive score is estimated based on how much the label overlaps with the correct category label in a set of training data. Another work which apply regression model to label clusters is presented in [79].

5.4.2 Unsupervised labeling

The cluster labeling approaches included in unsupervised group do not rely on a training set of labeled examples during the whole process of yielding labels for the cluster. There are several approaches that use unsupervised methods to label a cluster [9, 42, 16, 63, 72, 13, 70, 30, 46, 57, 69]. Carmel et al. in [9] proposed an unsupervised approach for cluster labeling by utilizing Wikipedia. Algorithm uses the meta-data of Wikipedia pages, such as categories and titles, for labeling the cluster. The algorithm selects top-k terms with maximal weights from the cluster's centroid as important terms, and then executes a query consist of important terms against the Wikipedia. Candidate labels, which are extracted from documents title and categories of search results, are evaluated by two judges; Mutual Information (MI) judge and Score Propagation (SP) judge. The MI judge scores each candidate by the average pointwise mutual information (PMI) of the label with the set of the cluster's important terms, with respect to a given external textual corpus (e.g., the web). The SP judge scores each candidate label with respect to the scores of the documents in the result set (result of query search) associated with that label. Indeed, this judge propagates documents' scores to candidates that are not directly associated with those documents, but share common keywords with other related labels.

Tseng in [74], proposed a generic cluster labeling method in which important terms are extracted by using chi-square (X^2) and correlation coefficient, and these descriptive terms are mapped generic terms based on a hypernym search algorithm which is generated based on WordNet. Feature selection method proposed in [23] selects candidate terms for labeling the generated clusters through a modified version of the information gain function. The method only considers the terms that positively describe the contents of a cluster. It means that, feature selection method in measuring of mutual information of term t and category c, only consider positive correlation of t. Therefore, the algorithm ignores negative correlation of t from IG formula and yields the modified version of IG: $IG_m(t,c) = P(t,c)log\frac{P(t,c)}{P(t)P(c)} + P(\bar{t},\bar{c})log\frac{(P(\bar{t},\bar{c}))}{(P(\bar{t})P(\bar{c}))}$.

One of the other works in cluster labeling that could be consider as supervised or unsupervised labeling or both of them, proposed by Roitman et al. in [62]. Indeed, the input of algorithm is a set of cluster labelers $\mathcal{L} = \{L_1, L_2, \text{âĂę}, L_m\}$, that could be supervised or unsupervised labelers, along with the cluster C wish to be label. Each labeler $L \in \mathcal{L}$

takes a cluster C as an input and may suggest a pool of total of nL distinct candidate cluster labels $L(C)$. It weights labels according to the estimated labeler's decisiveness with respect to each of suggested labels of labeler. The hypothesis of the approach is that, the label's choice of a cluster labeler for a given cluster should remain stable even in face of a slightly incomplete cluster data. To measure the stability of a cluster labeler's labeling choice, the algorithm forms an incomplete version of cluster C by sampling several sub-clusters $C_\theta = \{C_1, C_2, \text{âĂę}, C_N\}$ that each of them contains a subset of documents from the original cluster C. Each labeler L for each sub-cluster $C_i \in C_\theta$ presents a list of top-k labels $L^{[k]}(C_i)$. Therefore, there are a list of labels suggested by labeler L on the original cluster $L^{[k]}(C)$ and N lists of labels suggested by labeler L on the each of sub-cluster $C_i \in C_\theta$ called $L^{[k]}(C_i)$. Then, algorithm for all labels in $L^{[k]}(C)$ accounts the pairwise agreement between two top-k sub-cluster label lists $L^{[k]}(C_i)$ and $L^{[k]}(C_j)$ according to that specific label choice. The more such agreements (high score) are gathered for label l, the more it implies that labeler L may be decisive with respect to that specific label choice. Finally, by using two fusion methods; $CombSUM$ and $CombMNZ$, scores of labels, which are represented by various labelers in L, are summed over and boosted respectively. This approach presents a meta-cluster labeling solution for cluster labeling. In this investigation, we categorize some state of the art references based on the features they applied in order to labeling a cluster. The obtained result is presented in Table 5.2.

5.5 Topic labeling

Extracting topic word distributions are often intuitively meaningful, but the major challenge is to accurately interpret the meaning of each topic. Because user is not familiar with the source collection, it would be difficult for her to understand a topic only based on the multinomial distribution. There are several works trying to help user by labeling a topic model [42, 35, 46]. In this section we illustrate some of the remarkable works in topic labeling.

Mei et al. in [46] proposed first method to automatically generate labels for a topic model or a multinomial distribution of words, other than using a few top words in the distribution to label a topic. Their unsupervised proposed methodology uses a reference collection C (include: SIGMOD conference proceedings and Associated Press (AP)) to generate phrases as candidate labels, and then decide whether a phrase is good to label a topic. The algorithm first extracts candidate labels by using either Chunking/Shallow Parsing method or Ngram Testing method. The first method generates a set consist of the

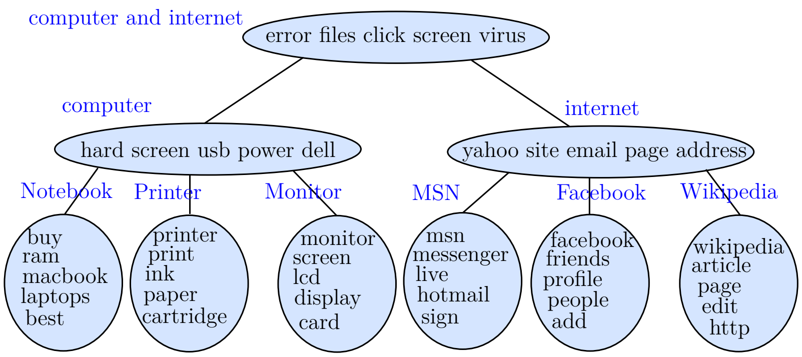

computer and internet
error files click screen virus
computer
hard screen usb power dell
internet
yahoo site email page address
Notebook
Printer
Monitor
MSN
Facebook
Wikipedia
buy ram macbook laptops best
printer print ink paper cartridge
monitor screen lcd display card
msn messenger live hotmail sign
facebook friends profile people add
wikipedia article page edit http

Figure 5-2: A topical hierarchy and labels obtained from [44]; The top 5 words are shown for each topic.

noun chunks/phrases frequently appearing in C. Another method extracts most significant N-grams based on statistical test, in which by using Student's T-Test measures the tendency of co-occurrence words in an N-gram with each other. In the second part, algorithm designs two relevance scoring functions to rank labels by their semantical similarity to a topic model θ. First relevance scoring function gives high score to a candidate label which contains more important words in the topic distribution. Assume candidate label $l = u_0 u_1 \ldots u_m$ (u_i is a word) the relevance scoring function define as:

$$Score = log\frac{p(l|\theta)}{p(l)} = \sum_{0 \leq i \leq m} log\frac{p(u_i|\theta)}{p(u_i)} \tag{5.3}$$

The second function, in order to prevent some problem may happen in the first function, utilizes a reasonable context to approximate a multinomial distribution decided by label l. And then measure the closeness of this approximated distribution and topic model θ using the Kullback-Leibler (KL) divergence. Indeed, this approach as a first approach in topic labeling proposes a method to accurately interpret the semantic of a topic instead of selecting the most frequent words of the empirical distribution as primitive labels [6], or manually generating more meaningful labels [45].

In other work, Magatti et al. in [42] proposed a method (ALOT) which uses a hierarchical topic model (obtained from the Google Directory service) implemented through a tree instead of approximate a multinomial distribution to be compared with existing topic. The algorithm implements an intermediate solution, between human and computer labeling, in which utilizes the available labeling schema for labeling. In the topics tree, each node is associated with a topic which has a label and the topics are linked by $IS-A$ relation.

The algorithm labels each topic through two main components; similarity measures and the labeling rules. It uses six similarity measures like; cosine similarity, overlap similarity, to find nearest topic t_j in topics tree to the topic t_i wish to be labeled. Finally algorithm labels topic t_i by exploiting some rules like: if all similarity measures agree on a specific topic t_j in topics tree then t_i will be labeled with the label of t_j, or if all similarity measures do not agree on one specific topic in topics tree then if all topics selected by similarity measures belong to common subtree then algorithm labels t_i with the label of the topic t_j which is the common deepest ancestor of topics in subtree. A topical hierarchy example is shown in Figure 5-2.

The algorithm proposed by Han Lau et al. [35], which is close to Mei et al, offers to generate topic label candidates using English Wikipedia, and then ranks the candidates to select the best topic labels. The algorithm generates topic label candidates by chunking parsing of primary candidates, the title of articles which extracted by querying top-10 topic terms on Wikipedia. In the next step, the algorithm uses several lexical association measures as the basis for an unsupervised and supervised model to ranking label candidates.

Hulpus et al. in [30], proposed a graph-based approach for topic labeling. The algorithm is based on a hypothesis that the words co-occuring in text likely refer to concepts that belong closely together in the DBpedia graph. Thus, the idea is to find the best relevant concept in DBpedia graph so that covers all other relevant concepts. There is a word sense disambiguation part (WSD) that represents an identified sense, which is extracted from DBpedia concepts, for each of the top-k words in topic model. As the extracted concepts of a topic are related, they should place near each other in DBpedia graph. Thus, the algorithm extracts one connected graph (topic graph) by expanding each concept for a few hops. The candidate labels are selected from which nodes of topic graph that play important structural role in the graph. To this aim, it uses several centrality measures (Closeness centrality, Betweenness centrality) [56] which are a well-known concept in social network science and are used to identify nodes that are most important for the network. Finally the graph based labeling algorithm ranks all nodes of topic graph and present top ones to the user as topic labels.

5.6 Evaluation of Cluster Labeler

Assessing the quality of a cluster labeling method is a subjective issue. Therefore, assessing objectively the quality of a cluster labeling method is a difficult problem. Indeed, there is no consensus that select a methodology as an authentic method to evaluate a cluster labeling method. For this reason some of the researchers apply a user study to evaluate the quality of a cluster labeling method. For example, Geraci et al. in [23], evaluate the label of each cluster by using several volunteer students as evaluators and proposing three questions: (a) Is the label syntactically well-formed?; (b) Can you guess the content of the cluster from the label?; (c) After inspecting the cluster, do you retrospectively consider the cluster as well described by the label? And then, the evaluator must choose one of the three possible answers (Yes; Sort of; No). Assessing the quality of a cluster label is performed by analyzing the volunteers' answers.

On the other hand, some of the researchers deal with the evaluation task by propounding a specific definition of cluster labeling task and then expose a specific assessment methodology in that case. One of these assessment methodologies which has attracted most of the researchers has proposed by [73]. In this approach the cluster labeling task is defined as descriptor-ranking problem. It considers the categories of Open Directory Project (ODP) as correct labels and then tries to evaluate a ranked list of proposed labels by defining the criteria in assessing of the quality. To this aim they proposed two definitions of a correct label: Exact match and partial match (for more details see [73]), in which it considers self-identity of a proposed label and a given category consist of a self-label and parent label. For both of these definitions of a correct label, they proposed the following evaluation measures.

- **Match at top N results (Match@N)**: Indicates whether the top N results contain any correct labels. It is a binary indicator, and monotonically increases as N increases.

- **Precision at top N results (P@N)**: Precision is computed as the number of labels in the top N results that match the correct categories label divided by N. P@N measures the percentage of correct answers that are displayed in ranks 1-N. In general, low precision is undesirable.

- **Mean Reciprocal Rank (MRR)**: Is the mean of the reciprocal of the rank of the first correct label. If the first correct label is ranked as the 3rd label, then the reciprocal rank (RR) is 1/3. If none of the first N responses contains a correct label, RR is 0. RR is 1 if the highest ranked label matches the correct label.

- **Mean Total Reciprocal Rank (MTRR)**: Sometimes there is more than one aspect to a category; for example, the category "acupuncture and Chinese medicine" has two correct aspects, "acupuncture" and "Chinese medicine". MTRR is similar to MRR, however, instead of considering only the rank of the first correct label as in MRR, MTRR takes into account all correct labels. Of the algorithm ranks "acupuncture" and “Chinese medicine” as the 2nd and the 4th labels, then the TRR (total reciprocal rank) is $\frac{1}{2}+\frac{1}{4}=\frac{3}{4}$ while $RR=\frac{1}{2}$.

Table 5.2: Categorization of some sate-of-the-art references based on the features they applied in order to labeling a cluster.

References	S	U	MSU	CC	ER	Intra	Inter	CII
[73]	✓	-	-	✓	-	-	✓	-
[7]	✓	-	-	✓	-	✓	-	-
[24]	✓	-	-	✓	-	-	✓	-
[23]	-	✓	-	✓	-	-	-	✓
[72]	-	✓	-	✓	-	✓	-	-
[36]	-	✓	-	✓	-	✓	-	-
[46]	-	✓	-	✓	-	✓	-	-
[70]	-	✓	-	✓	-	✓	-	-
[63]	-	✓	-	✓	-	✓	-	-
[16]	-	✓	-	✓	-	-	-	✓
[74]	-	✓	-	-	✓	-	✓	-
[35]	✓	-	-	-	✓	✓	-	-
[42]	-	✓	-	-	✓	✓	-	-
[30]	-	✓	-	-	✓	✓	-	-
[69]	-	✓	-	-	✓	✓	-	-
[9]	-	✓	-	-	✓	✓	-	-
[13]	-	✓	-	-	✓	✓	-	-
[57]	-	✓	-	-	✓	✓	-	-
[62]	-	-	✓	-	✓	-	-	✓
[79]	✓	-	-	-	-	-	-	-

Supervised (S)
Unsupervised (U)
Mix of Supervised and Unsupervised (MSU)
Extracting labels from cluster content (CC)
Extracting labels relying on external resources (ER)
Cluster internal labeling (Intra)
Differential cluster labeling (Inter)
Combine Intra and Inter labeling (CII)

6

A Fusion Approach in Improving clustering quality by Topic Models

Topic modeling algorithms are statistical methods that aim to discover the topics running through the text documents. Using topic models in machine learning and text mining is popular due to its applicability. In this chapter, we represent an enriching document approach, using state-of-the-art topic models and data fusion methods, to enrich documents of a collection with the aim of improving the quality of text clustering and cluster labeling. We propose a bi-vector space model in which every document of the corpus is represented by two vectors: one is generated based on the fusion-based topic modeling approach, and one simply is the traditional vector model. Our experiments on various datasets show that using a combination of topic modeling and fusion methods to create documents' vectors can significantly improve the quality of the results in clustering the documents.

6.1 Introduction

While we are overwhelming by increasing amount of available texts, we simply do not have the human power to read and study them to provide browsing and organizing experience over such the huge amount of texts. To this end, machine learning researchers have developed probabilistic topic modeling, a suite of algorithms that aim to discover and annotate large archives of documents with thematic information. Topic modeling algorithms are statistical methods which are able to find the themes (topics) running through the text documents by analyzing their words. Using topic models in machine learning and text mining is popular due to its applicability. In document clustering, a topic model could be directly used to map the original high-dimensional representation of documents (word features) to a low dimensional representation (topic features) and then applies a standard clustering algorithm like k-means in the new feature space, or we can consider each topic as a cluster and documents with highest proportion of same topic are located in the same cluster [41]. Lu et al. in [41] investigated performance of two probabilistic topic models PLSA and LDA in document clustering. Authors used the topic models to generate specific topics which are treated each one as a cluster. Therefore, for clustering, the documents are clustered into the topic with the highest probability. In similar way, [78] aims to elaborate on the ability of further other topic modeling algorithms CTM, Hierarchical LDA, and HDP to cluster documents.

We highlight two main problems here: first, we do not know the exact number of topics running through the corpus, besides, because of frequency-based nature of topic models, we cannot claim the topic with the highest probability for a document is the main topic by which the documents must be clustered. These two problems are considered as our hypothesis in dealing with topics running through the corpus.

In this work, we present a novel approach to improve the quality of clustering using topic models [5] and fusion methods [77]. The core idea of our approach is to **enrich** the vectors of the documents in order to improve the quality of clustering. To this end, we apply a statistical approach to discover and annotate a corpus with thematic information represented in form of different proportions over different **topics** for each document.

We first run topic modeling several times with different parameters over the collection, we then specify a set of topics in each iteration as the *special topics* for each document. Finally, we **combine** all the special topics of each iteration to generate a single topic for every document. These topics are treated as the vectors that are used in the clustering. Furthermore, we use these topics to generate labels for each cluster.

To the best of our knowledge, our work is the first to suggest a topic modeling solution to improve the quality of clustering and to perform cluster labeling based on the fusion methods.

6.2 Our Method

To create an enriched vectorial representation for documents of a corpus, we propose an unsupervised technique, called Fusion- and Topic-based Enriching (FT-Enrich). Let $\mathbb{D} = \{d_1, d_2, ..., d_n\}$ is the collection of documents that we wish to be clustered, we run topic models algorithm several times over the collection, every time with different specified number of topics. We start with the number of topics close to the number of clusters, for example, if K is the number of clusters we wish to have, the beginning number for topics is $K \pm \kappa$ and every time is increased by one. The reason of starting with this number of topics is to emphasize the topics of the iteration with number of topics close to the number of clusters. Finally, for every document d_j of $\mathbb{D}$ there is a set $\mathbb{B} = \{\mathcal{B}_1, \mathcal{B}_2, .., \mathcal{B}_m\}$ where $\mathcal{B}_i = \{\beta_1, \beta_2, .., \beta_s\}$ shows s topics belong to iteration i, and m indicates the number of iterations.

In every iteration, for each document, we generate a set of topics, namely, *special topics* where its elements are selected from the topics within iteration i. To generate these topics, we construct a graph G_i comprising the documents of $\mathbb{D}$ and the topics generated in iteration i. Figure 6-1 shows three examples of graph G in different iterations. Every circular node corresponds to a document of the collection, and the square nodes correspond to the topics generated in that iteration. The connection $\mathcal{X}_{jr}$ between a circular node d_j and a square node β_r indicates the proportion of the corresponding topic in the document. Therefore, if $\mathbb{P}_i = \{\theta_{1:s}, \theta_{2:s}, ..., \theta_{n:s}\}$ indicates topic proportions of the documents in iteration i where $\theta_j = \{\mathcal{X}_{j1}, \mathcal{X}_{j2}, .., \mathcal{X}_{js}\}$ shows topic proportions for document j in graph G_i where $\sum_{l=1}^{s} \mathcal{X}_{jl} = 1$. Therefore, the elements of *special topics* for document d_j, within iteration i, include:

- the topic with highest proportion of $\mathcal{X}jx$ for document d_j,
- the topic by which document d_j finds its best couple,
- the topic by which document d_j is selected as the best couple for a document.

Given the topics of iteration ith, the best couple for document d_j is a document d_k for which the following equation returns highest value:

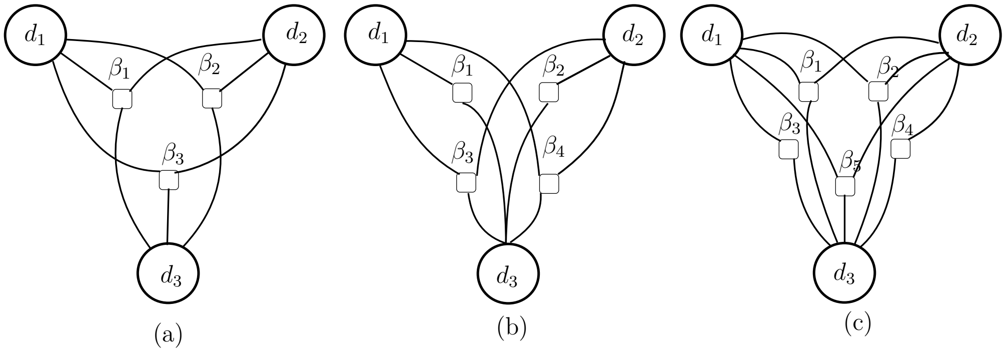


Figure 6-1: Three typical graphs of G for $\mathbb{D} = \{d_1, d_2, d_3\}$ in three different iterations with a) three topics, b) four topics, and c) five topics.

$$Couple(d_j, d_k|\mathcal{B}_i) = \arg\max_{\beta_l \in \mathcal{B}_i}\left(\frac{\mathcal{X}_{jl} \times \mathcal{X}_{kl}}{|\mathcal{X}_{jl} - \mathcal{X}_{kl}|}\right) \tag{6.1}$$

where the denominator for $\mathcal{X}_{jl} = \mathcal{X}_{kl}$ equals 1.

Therefore, for each document in a specific iteration, there is a special topics set $ST_i(d_j)$ where $|ST_i| <= |\mathcal{B}_i|$. We take into account the effect of special topics for each document by combining elements of $ST_i(d_j)$. Indeed, our goal is to generate a representing vector for each document to be used in clustering where this vector is a combination of some special topics. We use data fusion method CombSUM in two phases to generate a single topic (vector) for each document in the corpus.

In the first phase, all the topics within $ST_i(d_j)$ are combine to generate a single vector $\mathcal{V}_{ij}$ for each document d_j in the iteration i. Formally, let $S_\beta^{norm}(b|\mathcal{B}_i)$ denotes b's **normalized** score given in distribution (topic) β, the general form of CombSUM fusion method then simply sums over the normalized b'scores given by various topics in $ST_i(d_j)$.

$$CombSum(b|ST_i(d_j)) = \sum_{\beta \in ST_i} \mathcal{X}_{j\beta} \times S_\beta^{norm}(b|\mathcal{B}_i) \tag{6.2}$$

Where $\mathcal{X}_{j\beta}$ is the proportion of document j in topic β.

In the second phase, all the single vectors $\mathcal{V}_{ij}$ generated in m iterations are combined to generate a **unique** vector V_j for document j. Formally, given $AV(d_j) = \{\mathcal{V}_{1j}, \mathcal{V}_{2j}, ..., \mathcal{V}_{mj}\}$, let $S_\mathcal{V}^{norm}(b|\mathbb{B})$ denotes b's normalized score given in vector $\mathcal{V}$, therefore, the CombSUM fusion method then sums over the normalized b'scores given by various vectors in $AV(d_j)$.

$$CombSum(b|AV(d_j)) = \sum_{\mathcal{V} \in AV} S_{\mathcal{V}}^{norm}(b|\mathbb{B}) \tag{6.3}$$

Finally, a trade-off between V_j and traditional vector, which is a vector created based on *tf-idf* for document j, are used to generate the final vector. Which is the representing vector for jth document in clustering. Formally:

$$FV_j^{norm} = \alpha \times V_j^{norm} + (1 - \alpha) \times vt_j \tag{6.4}$$

Where vt_j indicates traditional vector for jth document, and $\alpha \in [0, 1]$.

6.3 Cluster Labeling

To label a cluster $C = \{FV_1, FV_2, ..., FV_c\}$, we use CombMNZ data fusion method which provides good results in combining several ranked lists [77, 62]. First, we **rank** all the vectors within C and create $\mathcal{L} = \{L_1, L_2, ..., L_c\}$ where L_j is the corresponding ranked vector to FV_j. We then create candidate-labels lists $L^{[N]}(C)$ which are top-N corresponding words to vectors L. Therefore, let $\mathcal{L}^{[N]}(C) = \bigcup_{L \in \mathcal{L}} L^{[N]}(C)$ denotes the overall candidate label pool which are based on the union of all top-N scored labels selected from $L \in \mathcal{L}$ for cluster C. The CombMNZ is to boost labels based on the number of top-N label lists that include each label. Formally:

$$CombMNZ(l|\mathcal{L}^{[N]}(C)) = \#\Big\{l \in L^{[N]}(C)\Big\} \times \sum_{L \in \mathcal{L}} S_L^{norm}(l|C) \tag{6.5}$$

Finally, top-K (i.e., $|K| < |N|$) labels of the combination result are selected as the label of the cluster C.

6.4 Experimental Setup

We explore the effectiveness of using representation vectors of documents generated by our method in addition to label the clusters. To this end, we used three different datasets *Classic4*, *BBC news*, and *20NG*. For our experiments with Classic4, we extract randomly 500 documents from each class.

Table 6.1: Clustering results of dataset BBC: (a) using traditional document representations $(\alpha = 0)$; (b) using FT-Enrich method $(\alpha = 1)$.

Cluster	Bus	Enter	Polit	Sport	Tech	Purity
Cluster 0	58	6	254	5	11	0.760
Cluster 1	320	2	15	4	5	0.925
Cluster 2	79	24	52	7	344	0.680
Cluster 3	30	16	15	441	5	0.870
Cluster 4	23	338	81	54	36	0.635
Total Purity						**0.763**

(a)

Cluster	Bus	Enter	Polit	Sport	Tech	Purity
Cluster 0	21	21	401	27	13	0.830
Cluster 1	473	5	9	1	8	0.954
Cluster 2	13	6	3	0	364	0.943
Cluster 3	1	0	1	482	4	0.980
Cluster 4	2	354	3	1	12	0.952
Total Purity						**0.932**

(b)

Table 6.2: Clustering results of dataset Classic4: (a) using traditional document representations $(\alpha = 0)$; (b) using FT-Enrich method $(\alpha = 0.1)$.

Cluster	Cacm	Cisi	Cran	Med	Purity
Cluster 0	323	30	11	21	0.839
Cluster 1	55	17	479	0	0.869
Cluster 2	47	6	4	454	0.888
Cluster 3	75	447	6	25	0.808
Total Purity					**0.852**

(a)

Cluster	Cacm	Cisi	Cran	Med	Purity
Cluster 0	334	9	2	0	0.968
Cluster 1	71	0	485	0	0.872
Cluster 2	43	0	6	485	0.908
Cluster 3	49	491	7	15	0.874
Total Purity					**0.898**

(b)

Preprocessing.

In addition to preprocessing explained in Section 2.6.2, we used Norm-2 to normalize the topics generated by MALLET.

6.5 Experimental Results

6.5.1 Evaluating Results of Clustering

In our experiments, we use the software package CLUTO[1] which is used for clustering low- and high-dimensional datasets. The algorithm adopted for clustering is Partitional, and the measure of similarity between two vector is Cosine similarity. Every document of the corpus is represented by two vectors: one is generated based on FT-Enrich method, and one simply is the traditional vector–classical *tf-idf* weighting of terms–model.

Indeed, we tested and evaluated clustering with/without applying FT-Enrich, to show the improvements in clustering purity due to a capable combination of fusion and topic

[1] http://glaros.dtc.umn.edu/gkhome/views/cluto

Table 6.3: Clustering results by grouping documents which have a same topic with highest probability (baseline): (a) on the BBC; (b) on the Classic4.

Cluster	Bus	Enter	Polit	Sport	Tech	Purity
Cluster 0	5	12	0	70	285	0.706
Cluster 1	0	348	6	180	5	0.533
Cluster 2	462	9	17	0	10	0.924
Cluster 3	18	12	375	11	10	0.880
Cluster 4	25	5	19	250	91	0.387
Total Purity						**0.773**

(a)

Cluster	Cacm	Cisi	Cran	Med	Purity
Cluster 0	211	386	11	10	0.625
Cluster 1	258	59	0	128	0.580
Cluster 2	25	8	484	0	0.936
Cluster 3	6	47	5	362	0.862
Total Purity					**0.745**

(b)

Table 6.4: The clustering results on 20NG using representation vectors generated by: traditional TF-IDF method, and FT-Enrich method with $\alpha = 1$.

	Purity of Cluster																				
Method	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	Total Purity
TF-IDF	0.25	0.39	0.16	0.5	0.21	0.3	0.46	0.48	0.25	0.3	0.16	0.29	0.71	0.24	0.84	0.35	0.41	0.4	0.71	0.45	**0.38**
FT-Enrich	1.0	0.96	0.89	0.99	0.96	0.64	0.63	0.43	0.23	0.42	0.28	0.47	0.95	0.59	0.88	0.95	0.94	0.94	0.35	0.81	**0.64**

modeling approaches. The obtained results of the such improvement are shown in Table 6.1 and Table 6.2. The obtained results in Table 6.1 indicate that representing documents by only FT-Enrich ($\alpha = 1$) significantly improve the quality of clustering. We can see in Table 6.1 the best improvement (more than %20) in total purity is obtained by entirely using FT-Enrich method ($\alpha = 1$). It further can be observed in cluster 4 we have about %50 improvement in purity of the cluster.

We have investigated the variation of α by considering the amount of dispersion of documents' sizes. Our experiments show that contribution of FT-Enrich method in creating the representation vectors for corpus with low Standard Deviation (SD)–considering its mean (M)–is major, in compare with the one with high SD. Table 6.2 shows the clustering result with $\alpha = 0.1$ on Classic4 dataset for which $SD = 143.34$ and $M = 158.47$, but on the other hand, the clustering result shown in Table 6.1 is obtained by $\alpha = 1$ for which ($SD = 123.64$, $M = 341.21$).

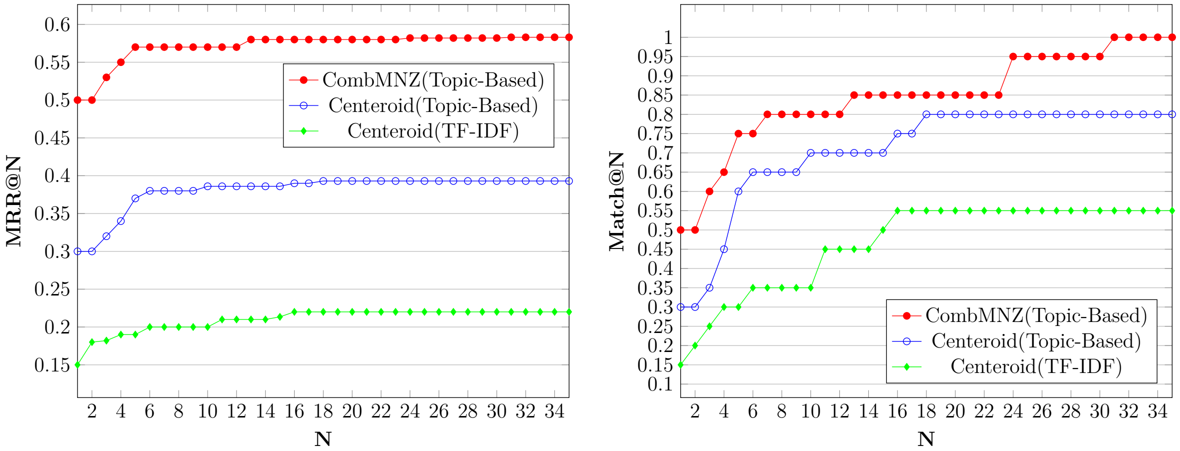
We further compared our method with the baseline approach [41] in which only the topic with the highest probability for each document is considered. The results of clustering are shown in Table 6.3. As can be observed, using this approach even returns worse result in clustering on dataset Classic4, in compare with using tradition representation vectors.

6.5.2 Evaluating Results of Cluster Labeling

We use 20NG benchmark for our experiments in cluster labeling. Therefore, we first show the result of clustering on this dataset using representation vectors generated by our method which indeed are used in cluster labeling. We further compare our result with the clustering result obtained by using the traditional representation vectors. The result are shown in Table 6.4. It shows a remarkable improvement in clustering (about %68) which lead to achieve significant result in cluster labeling as well.

The cluster labeling method represented in this work is a **direct** cluster labeling method in which the candidate labels for clusters are directly extracted from content of the clusters without using external sources (e.g. Wikipedia). One of the baseline direct approach that several clustering systems are applied for cluster labeling [16] is to select the top-k terms with maximal weights from the cluster' centroid as the candidate labels. In our experiments we use this approach as a baseline for comparison. Specifically, we explore the effectiveness of using candidate labels generated by our approach in addition to the highest weighted terms extracted from clusters' centroids provided by: TF-IDF and FT-Enrich method.

Figure 6-2 reports on the Match@N and MRR@N scores of each method for increasing values of N. As can be observed, using the highest weighted terms extracted from clusters' centroids provided by FT-Enrich method is more effective than one provided by TF-IDF. It further shows that using fusion method (CombMNZ(FT-Enrich)) on the representation vectors generated by FT-Enrich method provides the best performance for both label quality measures. We can further observe that, for the Match@N measure, baseline method with FT-Enrich cluster's centroid requires at list 18 terms to cover %80 of the clusters with a correct label, while the same effectiveness is achieved by a list of 7 terms only using FT-Enrich method. It is also interesting that with $N > 31$ CombMNZ(FT-Enrich) method covers %100 of the clusters with a correct label.



(a) *MRR@N*

(b) *Match@N*

Figure 6-2: Average MRR@N and Match@N values obtained for clusters of 20NG using fusion method over representation vectors generated by FT-Enrich, using top-N terms of cluster' centroid weighted by FT-Enrich method, and using top-N terms of cluster' centroid weighted by TF-IDF.

Conclusions

In this thesis we contributed to two main problems in natural language processing and information retrieval. Our contributions include document clustering and cluster labeling which are addressed in three parts. Our contribution in document clustering aims at improving the performance of classical text clustering algorithms, particularly, by investigating the effectiveness of various state-of-the-art techniques which are applied through novel approaches. Our contribution in cluster labeling is to explore state-of-the-art approaches in this field as a unique survey, presenting a novel method.

In the first contribution, we investigated the effectiveness of document summarization and document enriching approaches on improving the quality of clustering. First, we have presented a new graph-based algorithm for keyphrase extraction, in turn used to summarize big documents in a textual corpus, before applying a clustering algorithm. Our experiments indicate the big documents, i.e., document whose size is significantly larger than the mean size in the corpus, introduce noise that can worsen the quality of clustering result. We tested our keyphrase extraction algorithm to summarize these big documents, thus retaining only the terms that are relevant to the main topics discusses in the documents, and observed a significant improvement in clustering quality, using common human-annotated corpora.

Furthermore, we have presented a multi-strategy algorithm to extract most salient linked entities from a document, and then exploit such entities in enriching the document. To this end, we apply a single text summarization method in order to extracting the most salient linked entities, and also disambiguating sense of words which are selected to be expanded. Moreover, we utilize graph partitioning approach with two aims; discarding irrelevant terms to not be expanded, and applying a clustering ensemble approach to result better quality of clustering. Our experiments indicate that enriching documents with the latent information, extracted from properties of only most salient entities, significantly improve the quality of clustering results.

A further advantage of our approach is that the included features in enriched documents of a cluster are the best candidates for labels of the cluster, considered to describe in an informative way the content of clusters. Furthermore, using expanded representations of documents, especially considering predecessor/successor semantical relations, increase the ability of hierarchical clustering to achieve results with better quality. As a future work of our approach is to exploit it in cluster labeling in order to analyze qualification of the presented labels for the clusters.

The second contribution in this dissertation turns to the cluster labeling. Probably the greatest challenge in cluster labeling is building an algorithm to extract most important terms from the clusters. These terms are the best descriptors of the cluster content which are critical to label a cluster. Type of the labels as the descriptor of the cluster content could be consider as another challenge because it must be able to give a satisfactory sense of the cluster content to user. Another challenge in cluster labeling could be evaluating the quality of a cluster labeling method, whereas assessing the quality of a cluster labeling method is a subjective issue. Therefore, assessing objectively the quality of a cluster labeling method is a difficult problem.

In this contribution we provide a thorough understanding of different cluster labeling and topic labeling methods, as well as introducing a novel method in cluster labeling. In the first part, we propose a categorization of cluster labeling methods based on their methodologies to extract important terms from the cluster and to generate label(s) for it. Moreover, we investigate the recent advancement in case of applying external knowledge bases (Wikipedia, DBpedia, WordNet, etc.) to choose meaningful labels for a cluster which are close to human choice. Overall, This contribution provides a comprehensive overview of all the existing cluster labeling methods that could be useful for researchers to proceed and develop novel cluster labeling techniques.

As another part of this contribution, we have presented a fusion- and topic-based enriching approach in order to improve the quality of clustering. We have applied a statistical approach, namely topic model, to enrich the representation vectors of the documents. To this end, an ensemble topic modeling with using different parameters for each model are represented, and then, using a fusion approach, all the generated results are combined to provide a single vectorial representation for each document. Our experiments on the different datasets show significant improvement in clustering results. We further show that putting such the representation vectors in a fusion method provides interesting results in cluster labeling as well. As a future work, it would be interesting to exploit external sources (lexical and ontology) in both the clustering and cluster labeling to explore the effectiveness of using topic models as well as such the resources in these domains.

Bibliography

[1] Charu C Aggarwal and Philip S Yu. *Finding generalized projected clusters in high dimensional spaces*, volume 29. ACM, 2000.

[2] Charu C Aggarwal and ChengXiang Zhai. A survey of text clustering algorithms. In *Mining text data*, pages 77–128. Springer, 2012.

[3] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. *Dbpedia: A nucleus for a web of open data.* Springer, 2007.

[4] David Blei and John Lafferty. Correlated topic models. *Advances in neural information processing systems*, 18:147, 2006.

[5] David M Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.

[6] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[7] Christos Bouras, Vassilis Poulopoulos, and Vassilis Tsogkas. Perssonal's core functionality evaluation: Enhancing text labeling through personalized summaries. *Data & Knowledge Engineering*, 64(1):330–345, 2008.

[8] Tru H Cao, Thao M Tang, and Cuong K Chau. Text clustering with named entities: A model, experimentation and realization. In *Data mining: Foundations and intelligent paradigms*, pages 267–287. Springer, 2012.

[9] David Carmel, Haggai Roitman, and Naama Zwerdling. Enhancing cluster labeling using wikipedia. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 139–146. ACM, 2009.

[10] David Carmel, Elad Yom-Tov, Adam Darlow, and Dan Pelleg. What makes a query difficult? In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 390–397. ACM, 2006.

[11] Diego Ceccarelli, Claudio Lucchese, Salvatore Orlando, Raffaele Perego, and Salvatore Trani. Learning relatedness measures for entity linking. In *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management*, CIKM '13, pages 139–148. ACM, 2013.

[12] Youngchul Cha, Bin Bi, Chu-Cheng Hsieh, and Junghoo Cho. Incorporating popularity in topic models for social network analysis. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 223–232. ACM, 2013.

[13] Jackie Chi Kit Cheung and Xiao Li. Sequence clustering and labeling for unsupervised query intent discovery. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 383–392. ACM, 2012.

[14] Neal Coulter, James French, Ephraim Glinert, Thomas Horton, Nancy Mead, Roy Rada, Anthony Ralston, Craig Rodkin, Bernard Rous, Allen Tucker, et al. Computing classification system 1998: Current status and future maintenance. report of the ccs update committee. *Computing Reviews*, 39(1):1–62, 1998.

[15] Fabio Crestani. Application of spreading activation techniques in information retrieval. *Artificial Intelligence Review*, 11(6):453–482, 1997.

[16] 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*, pages 318–329. ACM, 1992.

[17] Dipanjan Das and André FT Martins. A survey on automatic text summarization. *Literature Survey for the Language and Statistics II course at CMU*, 4:192–195, 2007.

[18] Angela Fahrni and Michael Strube. Jointly disambiguating and clustering concepts and entities with markov logic. In *COLING*, pages 815–832, 2012.

[19] Christiane Fellbaum. *WordNet.* Wiley Online Library, 1998.

[20] Yansong Feng and Mirella Lapata. Topic models for image annotation and text illustration. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 831–839. Association for Computational Linguistics, 2010.

[21] Paolo Ferragina and Ugo Scaiella. TAGME: on-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada, October 26-30, 2010*, pages 1625–1628. ACM, 2010.

[22] Nathalie Friburger, Denis Maurel, and A Giacometti. Textual similarity based on proper names. In *Proc. of the workshop Mathematical/Formal Methods in Information Retrieval*, pages 155–167, 2002.

[23] Filippo Geraci, Marco Pellegrini, Marco Maggini, and Fabrizio Sebastiani. Cluster generation and labeling for web snippets: A fast, accurate hierarchical solution. *Internet Mathematics*, 3(4):413–443, 2006.

[24] Eric J Glover, Kostas Tsioutsiouliklis, Steve Lawrence, David M Pennock, and Gary W Flake. Using web structure for classifying and describing web pages. In *Proceedings of the 11th international conference on World Wide Web*, pages 562–569. ACM, 2002.

[25] Derek Greene and Pádraig Cunningham. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proceedings of the 23rd international conference on Machine learning*, pages 377–384. ACM, 2006.

[26] Thomas L Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101(suppl 1):5228–5235, 2004.

[27] Aria Haghighi and Lucy Vanderwende. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362–370. Association for Computational Linguistics, 2009.

[28] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.

[29] Andreas Hotho, Alexander Maedche, and Steffen Staab. Ontology-based text document clustering. *KI*, 16(4):48–54, 2002.

[30] Ioana Hulpus, Conor Hayes, Marcel Karnstedt, and Derek Greene. Unsupervised graph-based topic labelling using dbpedia. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 465–474. ACM, 2013.

[31] George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on scientific Computing*, 20(1):359–392, 1998.

[32] Jon M Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

[33] Markus Krötzsch, Denny Vrandečić, and Max Völkel. Semantic mediawiki. In *The Semantic Web-ISWC 2006*, pages 935–942. Springer, 2006.

[34] Takeshi Kurashima, Tomoharu Iwata, Takahide Hoshide, Noriko Takaya, and Ko Fujimura. Geo topic model: joint modeling of user's activity area and interests for location recommendation. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 375–384. ACM, 2013.

[35] Jey Han Lau, Karl Grieser, David Newman, and Timothy Baldwin. Automatic labelling of topic models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1536–1545. Association for Computational Linguistics, 2011.

[36] Dawn J Lawrie and W Bruce Croft. Generating hierarchical summaries for web searches. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 457–458. ACM, 2003.

[37] Wei Li and Andrew McCallum. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd international conference on Machine learning*, pages 577–584. ACM, 2006.

[38] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, volume 8, 2004.

[39] Marina Litvak and Mark Last. Graph-based word extraction for single-document summarization. In *Proceedings of the workshop on Multi-source Multilingual Information Extraction and Summarization*, pages 17–24. Association for Computational Linguistics, 2008.

[40] Tao Liu, Shengping Liu, Zheng Chen, and Wei-Ying Ma. An evaluation on feature selection for text clustering. In *ICML*, volume 3, pages 488–495, 2003.

[41] Yue Lu, Qiaozhu Mei, and ChengXiang Zhai. Investigating task performance of probabilistic topic models: an empirical study of plsa and lda. *Information Retrieval*, 14(2):178–203, 2011.

[42] Davide Magatti, Silvia Calegari, Davide Ciucci, and Fabio Stella. Automatic labeling of topics. In *Intelligent Systems Design and Applications, 2009. ISDA'09. Ninth International Conference on*, pages 1227–1232. IEEE, 2009.

[43] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[44] Xian-Ling Mao, Zhao-Yan Ming, Zheng-Jun Zha, Tat-Seng Chua, Hongfei Yan, and Xiaoming Li. Automatic labeling hierarchical topics. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 2383–2386. ACM, 2012.

[45] Qiaozhu Mei, Chao Liu, Hang Su, and ChengXiang Zhai. A probabilistic approach to spatiotemporal theme pattern mining on weblogs. In *Proceedings of the 15th international conference on World Wide Web*, pages 533–542. ACM, 2006.

[46] Qiaozhu Mei, Xuehua Shen, and ChengXiang 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.

[47] Qiaozhu Mei and ChengXiang Zhai. A mixture model for contextual text mining. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 649–655. ACM, 2006.

[48] Rada Mihalcea. Graph-based ranking algorithms for sentence extraction, applied to text summarization. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, page 20. Association for Computational Linguistics, 2004.

[49] Rada Mihalcea and Andras Csomai. Wikify!: Linking documents to encyclopedic knowledge. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 233–242. ACM, 2007.

[50] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into texts. Association for Computational Linguistics, 2004.

[51] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[52] David Milne and Ian H. Witten. Learning to link with wikipedia. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, pages 509–518. ACM, 2008.

[53] Soto Montalvo, Raquel Martínez, Arantza Casillas, and Víctor Fresno. Bilingual news clustering using named entities and fuzzy similarity. In *Text, Speech and Dialogue*, pages 107–114. Springer, 2007.

[54] Soto Montalvo, Raquel Martínez, Víctor Fresno, and Agustín Delgado. Exploiting named entities for bilingual news clustering. *Journal of the Association for Information Science and Technology*, 66(2):363–376, 2015.

[55] Roberto Navigli. Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR)*, 41(2):10, 2009.

[56] Mark Newman. *Networks: an introduction.* Oxford University Press, 2010.

[57] Tadashi Nomoto. Wikilabel: an encyclopedic approach to labeling documents en masse. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 2341–2344. ACM, 2011.

[58] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[59] Mohsen Pourvali, Salvatore Orlando, and Mehrad Gharagozloo. Improving clustering quality by automatic text summarization. In *Information Retrieval Technology*, pages 292–303. Springer, 2015.

[60] Dragomir R Radev, Eduard Hovy, and Kathleen McKeown. Introduction to the special issue on summarization. *Computational linguistics*, 28(4):399–408, 2002.

[61] Diego Reforgiato Recupero. A new unsupervised method for document clustering by using wordnet lexical and conceptual relations. *Information Retrieval*, 10(6):563–579, 2007.

[62] Haggai Roitman, Shay Hummel, and Michal Shmueli-Scheuer. A fusion approach to cluster labeling. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 883–886. ACM, 2014.

[63] François Role and Mohamed Nadif. Beyond cluster labeling: Semantic interpretation of clusters' contents using a graph representation. *Knowledge-Based Systems*, 56:141–155, 2014.

[64] Julian Sedding and Dimitar Kazakov. Wordnet-based text document clustering. In *proceedings of the 3rd workshop on robust methods in analysis of natural language data*, pages 104–113. Association for Computational Linguistics, 2004.

[65] Satoshi Sekine. Named entity: History and future. *Project notes, New York University*, page 4, 2004.

[66] Jonathan A Silva, Elaine R Faria, Rodrigo C Barros, Eduardo R Hruschka, André CPLF de Carvalho, and João Gama. Data stream clustering: A survey. *ACM Computing Surveys (CSUR)*, 46(1):13, 2013.

[67] Mark Steyvers, Padhraic Smyth, Michal Rosen-Zvi, and Thomas Griffiths. Probabilistic author-topic models for information discovery. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 306–315. ACM, 2004.

[68] Alexander Strehl and Joydeep Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *The Journal of Machine Learning Research*, 3:583–617, 2003.

[69] Zareen Saba Syed, Tim Finin, and Anupam Joshi. Wikipedia as an ontology for describing documents. In *ICWSM*, 2008.

[70] Krishnaprasad Thirunarayan, Trivikram Immaneni, and Mastan Vali Shaik. Selecting labels for news document clusters. In *Natural Language Processing and Information Systems*, pages 119–130. Springer, 2007.

[71] Ivan Titov and Ryan McDonald. Modeling online reviews with multi-grain topic models. In *Proceedings of the 17th international conference on World Wide Web*, pages 111–120. ACM, 2008.

[72] Hiroyuki Toda and Ryoji Kataoka. A search result clustering method using informatively named entities. In *Proceedings of the 7th annual ACM international workshop on Web information and data management*, pages 81–86. ACM, 2005.

[73] Pucktada Treeratpituk and Jamie Callan. Automatically labeling hierarchical clusters. In *Proceedings of the 2006 international conference on Digital government research*, pages 167–176. Digital Government Society of North America, 2006.

[74] Yuen-Hsien Tseng. Generic title labeling for clustered documents. *Expert Systems with Applications*, 37(3):2247–2254, 2010.

[75] Sandro Vega-Pons and José Ruiz-Shulcloper. A survey of clustering ensemble algorithms. *International Journal of Pattern Recognition and Artificial Intelligence*, 25(03):337–372, 2011.

[76] Xuerui Wang and Andrew McCallum. Topics over time: a non-markov continuous-time model of topical trends. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 424–433. ACM, 2006.

[77] Shengli Wu. *Data fusion in information retrieval*, volume 13. Springer Science & Business Media, 2012.

[78] Chyi-Kwei Yau, Alan Porter, Nils Newman, and Arho Suominen. Clustering scientific documents with topic modeling. *Scientometrics*, 100(3):767–786, 2014.

[79] Hua-Jun Zeng, Qi-Cai He, Zheng Chen, Wei-Ying Ma, and Jinwen Ma. Learning to cluster web search results. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 210–217. ACM, 2004.

[80] ChengXiang Zhai, Atulya Velivelli, and Bei Yu. A cross-collection mixture model for comparative text mining. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 743–748. ACM, 2004.

[81] Xiaodan Zhang, Liping Jing, Xiaohua Hu, Michael Ng, and Xiaohua Zhou. A comparative study of ontology based term similarity measures on pubmed document clustering. In *Advances in Databases: Concepts, Systems and Applications*, pages 115–126. Springer, 2007.