Document Clustering using various External Knowledge sources

Thesis submitted in partial fulfillment

of the requirements for the degree of

Master of Science (by Research)

in

Computer Science & Engineering

by

G V R Kiran

200602007

kiran.gvr@research.iiit.ac.in

gvrkirann@gmail.com

Center for Data Engineering

International Institute of Information Technology

Hyderabad - 500 032, INDIA

May 2011

International Institute of Information Technology

Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled **"Document Clustering using various External Knowledge sources"** by **G V R Kiran**, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Adviser: Prof. Vikram Pudi

Dedicated to my mother ***G V Lakshmi***

and grand parents ***Sitarama Sastry*** *&* ***Lakshmi***

for their love and support.

Acknowledgments

First and foremost, thanks to God, for giving me the strength and confidence to do this work.

I would like to thank my advisor Dr. Vikram Pudi for his support through out. He has been a great mentor, a good friend and gave me the freedom to explore my thoughts. There were many times when I was out of ideas and he helped me a lot with fresh ideas and motivated me.

I also express my sincere gratitude to Dr. Carolyn Rose, Carnegie Mellon, for inspiring me to think out of the box and giving me many opportunities to do so. My future would have been completely different from what it is now if I wouldn't have met her.

I would thank my mother G V Lakshmi, my grand parents Mr & Mrs. Sastry for their never ending love, support and encouragement. Special thanks to my brother Karteek, for constantly encouraging me to give out my best.

Last but not the least, many thanks to my good friend, project mate, lab mate, etc. Ravi Shankar, without whose companionship, all this work would not have been possible. Also, thanks to my friends PDSR Sandeep, Giridhar, Abhilash, Kiran, Kishore, Sandeep Thambi and all my other batch-mates for their support during my hard days and for making my days at IIIT so memorable.

Abstract

The problem of document clustering is to arrange a set of documents into groups such that documents within a group are related to each other. Document clustering has been studied extensively because of the huge increase in the amount of digital text content online and also due to its wide applicability in areas such as web mining, information retrieval, etc. Typical document clustering algorithms have issues like handling high dimensionality, scalability, building an efficient hierarchy, good clustering accuracy, etc.

In this thesis, we present a framework for document clustering that addresses all the above issues. The framework is a fusion of two broach approaches. The first approach consists of a topic-based document clustering algorithm, and in the second approach we provide methods to enhance the clustering produced using various external knowledge sources like Wikipedia and Social Content.

Next, we provide a specific implementation of the first approach of our framework using the concept of frequent itemsets. We define a frequent itemset based hierarchical document clustering algorithm that handles the high dimensionality of the documents by considering only important words which represent topics. A frequent itemset is a set of words that occurs frequently together in a set of documents. Using frequent itemsets also takes care of the scalability of the algorithm as we consider only the important (topic-indicating) words. We explain the drawbacks of the previous approaches and propose improvements to them in our approach to obtain a clustering. Next, we provide methods to process these clusters in order to construct a compact hierarchy of clusters. We also give meaningful labels to the clusters using Wikipedia to allow a user to easily browse through the clusters.

In the second approach of our framework, we enhance the clustering obtained previously using knowledge from various sources like Wikipedia and Social content like tags, comments, etc. There has been work done previously to enhance document clustering using various external knowledge sources like WordNet, Wikipedia, Open Directory project, etc. Most of these approaches deal with huge knowledge bases and hence are an overhead to the clustering process in terms of time taken for the enhancement. Our approach is fast because we only use the topic-indicating words for enhancing the document content. Moreover, our approach also handles noise efficiently.

Wikipedia is the biggest known online knowledge base that is constantly updated. Moreover, it contains various useful features like a manually tagged categorization for each document, a strong link structure, etc that can be used to improve the quality of a clustering. Using Wikipedia to enhance document clustering has been tried before. Though our approach has been inspired by them, the setting in which we use Wikipedia is an entirely different one and we also use a different set of features that provide a much better enhancement.

Using social content to enhance document clustering has never been tried before. The amount of social content being produced online is increasing rapidly because of the recent increase in the popularity of social web. A large amount of user generated content like tags, comments, reviews, etc are being generated. These provide a high quality meta-data that can be used to improve the clustering quality. We provide methods to use such meta data information to enhance document clustering.

We performed various experiments to show the validity of our approaches by comparing them with the state of the art approaches. We show that our results are better than most of the existing approaches in terms of various metrics like F-score, purity, NMI, etc on standard document datasets.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this age of information explosion, a large amount of text content in the form of query logs, emails and news documents is being generated. There is a lot of implicit information that is hidden in this data, which could be of great value if extracted. It would be very useful if we have methods to automatically interpret and make use of this data.

Data mining provides solutions to this problem of interpreting the data. Data mining is a method of extracting interesting knowledge from structured/unstructured data in large databases. Clustering is one such methods in data mining that could help us organize the data into groups that have similar properties.

Document Clustering, which is a sub-area of clustering help us in managing the large amounts of textual data being generated online daily. In this thesis, we present a hierarchical document clustering algorithm that allows us to easily browse through large document collections by clustering them and then constructing a meaningful hierarchy. We then present methods for enhancing this clustering using various knowledge sources. Our methods are fast, efficient and scalable to large document sets.

1.1 Motivation

Document clustering has been studied extensively because of its wide applicability in areas such as web mining, information retrieval, etc. Typical algorithms for document clustering have to handle many issues like:

1. *High Dimensionality*: Most of the clustering algorithms work well on low dimensional data, but fail to cluster data objects in high dimensional space, especially when the data is very sparse and highly skewed. Document clustering deals with a high dimensional space since each document typically contains thousands of words, making the dimensionality to be in thousands.

2. *Scalability*: Many clustering algorithms work fine on small datasets, however some of them fail to handle large datasets containing over tens of thousands of data points. Since document clustering deals with datasets containing data of the order of tens of thousands, the algorithms that we design must be scalable enough.

3. *A browsable hierarchy*: Efficiently building a hierarchy that is easily browsable is important. A hierarchy that is not easy to browse is of no use to the end user. Thus, the algorithms for document clustering must ensure that they prune out redundant clusters and present a compact yet perfect representation of the clustering to the user.

4. *Creating effective cluster labels*: For the user to easily browse through the clusters, the clusters must be labeled with short, yet meaningful labels that capture the essence of the entire cluster. Algorithms must be designed to produce good clustering labels to the clusters.

5. *Maintaining good cluster quality*: Standard clustering approaches that only make use of the document content typically produce clusters of poor quality as they do not capture the semantics of the document. This problem can be addressed by using external knowledge sources for enhancing the clustering.

Using knowledge sources: In the recent past, there has been a tremendous increase in the amount of open knowledge available online. Knowledge bases like Wikipedia, WordNet, Open Web Directory, etc come under this category. The amount of user generated content like tags, comments, reviews, etc has also been huge over the past few years. These repositories contain contain lots of knowledge that can be used for various purposes. Human knowledge and intuition are captured in some of these sources. Attempts have been made to use such

knowledge for enhancing document clustering, classification, improving search results, efficient recommendations, etc. In this thesis, we propose a method that can make use of any external knowledge to enhance document clustering.

The concept of hierarchical clustering and the weaknesses of the standard clustering methods (with respect to the issues described earlier) formulate the goal of this research: To provide an efficient, scalable and accurate clustering method that addresses the special challenges of document clustering. The resulting hierarchy of clusters should facilitate browsing by providing meaningful labels to the clusters. The clustering thus produced is then enhanced using external sources to produce a much finer clustering.

1.2 Problem Statement

In this section, we provide a formal definition of the problem that we are addressing in this thesis. Given a document collection D consisting of documents $d_1, d_2, d_3, ..., d_n$. Each document is represented using the vector space model as a vector of features ($\vec{d_i}$). Typically, these features constitute the important (key)words in the document. The basic problem of document clustering is to arrange these documents into groups (clusters) such that documents with in a group are similar to each other with respect to a similarity metric S. In a hierarchical clustering, these groups are arranged in a parent-child fashion, where a parent-child relationship represents topics and subtopics in the hierarchy.

We may also be given one or more external knowledge sources $K_1, K_2, K_3, ..., K_m$. We require that there is a method to extract features $K_1^{f_1}, K_1^{f_2}, K_2^{f_1}$, etc from these knowledge sources which can be used to enhance the document representation (vector of each document) to get the enhanced document vector $\vec{d_i}^e$. After enhancing, we use this enhanced vector $\vec{d_i}^e$ to get an improved clustering.

1.3 Contributions of this thesis

The major contributions of this thesis are:

1. We show a strong formal connection between the dual problems of document clustering and topic detection when seen in the context of frequent itemset mining.

2. We use the above connection to define a framework for topic-based clustering and uses external knowledge sources to enhance the clustering. Our framework can be extended to any external knowledge on any dataset.

3. We propose a frequent itemset based hierarchical document clustering algorithm that fits into our framework. We use the idea of generalized closed frequent itemsets to achieve a compact clustering. We also represent the clusters as a hierarchy, with appropriate labels to facilitate an easy browsing. Our approach is highly scalable and efficient since it deals with a low dimensional space.

4. We propose an approach for enhancing document clustering using knowledge from external sources like Wikipedia and Social Content. To our knowledge, we were the first one to use Social Content to enhance document clustering. The use of external knowledge improves the clustering because it considers the semantic association between the words.

1.4 Thesis Organization

The rest of the thesis is organized as follows:

- Chapter 2 - Background: Briefly discusses a few essential basics on frequent itemset mining, document clustering, external knowledges and our evaluation methodologies.

- Chapter 3 - Related Work: A survey of the recent methods that use frequent itemsets for clustering and a few methods that use various external knowledge sources like Wikipedia, Open Web Directory, etc for enhancing document clustering.

- Chapter 4 - Framework: A discussion of our framework for document clustering. It consists of two approaches, a basic topic-based document clustering approach and an approach for enhancing document clustering using various external knowledge sources.

- Chapter 5 - GenFIDoC- Generalized Frequent Itemset based Documnet Clustering: Describes our document clustering approach using frequent itemsets. We use the idea of generalized closed itemsets to produce a compact clustering.
- Chapter 6 - Enhancing GenFIDoC using various external knowledges: Describes methods for enhancing document clustering using knowledge from various sources like Wikipedia, Social Content, etc.
- Chapter 7 - Conclusions and Future Work: A brief summary of the contributions of this research and the possible directions to extend this work in the future.

Chapter 2

Background

This chapter briefly describes the various methods that we use in this thesis. The rest of this chapter is organized as follows: Section 2.1 describes the basic notion of what frequent itemset mining is and how they are useful. Section 2.1.2 describes a special type of frequent itemsets called generalized closed frequent itemsets that we use later for clustering. In Section 2.2, we describe the basics of what clustering means and what document clustering is about. We also describe metrics like Cosine Similarity and TF-IDF that are most commonly used in document clustering. In Section 2.3, we describe the various external knowledges that we use and their features. Finally, in Section 2.4, we define the various evaluation methods that we used for evaluating our clustering.

2.1 Frequent Itemset Mining

Frequent itemset mining has been an interesting problem in Data Mining. The input data for frequent itemset mining consists of a set of records, each having a set of items. From these set of items, the goal of a frequent itemset mining algorithm is to extract groups of itemsets that occur frequently, i.e. have a frequency greater than a certain user specified threshold. This threshold, which characterizes whether an itemset is frequent or not is commonly known as the "minimum support" (denoted *minsup*) of the frequent itemset.

Lets consider a small example of a frequent itemset mining. Table 2.1 shows some sample trasaction data on which we perform the frequent itemset mining. The frequent itemsets from

Table 2.1 Sample transaction data

Transaction Id	Transaction
t_1	a, b, c, d, f
t_2	b, c, d, e, g
t_3	b, d, e, f, g
t_4	a, c, e, f, h
t_5	a, b, d, f

this data for a $minsup \geq 50\%$, (i.e. which are present in more than half of the transactions) are: $\{a\}$, $\{b\}$, $\{c\}$, $\{d\}$, $\{f\}$, $\{a, c\}$, $\{a, f\}$, $\{b, d\}$, $\{b, d, f\}$, etc. On the other hand, the itemsets $\{a, b\}$ or $\{a, d\}$ are not frequent because they occur only in 40% (2/5) of the transactions. A frequent itemset of length k is called a *k-itemset*. In the above example, we have five $1 - itemsets$, three $2 - itemsets$ and one $3 - itemset$.

Frequent itemset mining is mainly used in mining association rules from the data, i.e. finding associations, correlations and frequent patterns among the items in the data. There are a wide range of other applications in frequent itemset mining can be used, like marketing, recommendation systems, agriculture, classification, clustering, etc. Many algorithms have been proposed for frequent itemset mining, and the most popular methods are *Apriori* [1] and *FP Growth* [11] algorithms.

In this thesis, we model document collections as transaction data and apply frequent itemset mining on them. This not only reduces the dimensionality we are dealing with but and also helps us in extracting important words. The way in which we model the documents as transaction data is explained below.

For each document in the dataset, we perform stopword removal by using a list of stopwords. Then, we stem the words using Porter stemmer to obtain their root form. After that, we represent each document as a transaction in a transaction database. Each transaction in a transaction database consists of what elements have been purchased. If an element is purchased, its value is 1 in the transaction and a 0 if it isn't purchased. Similarly, in each document an entry for a word is 1 if it is present in the document or 0 otherwise. A sample document representation for the data in Table 2.2 is shown in Table 2.3.

Table 2.2 Documents and the words they contain

Document	words contained in the document
D_1	w_1, w_2, w_3, w_4
D_2	w_2, w_3, w_4, w_5
D_3	w_1, w_3, w_5, w_6
D_4	w_4, w_5, w_6, w_7
D_5	w_1, w_2, w_4, w_8

Table 2.3 Sample Document Representation

	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8
D_1	1	1	1	1	0	0	0	0
D_2	0	1	1	1	1	0	0	0
D_3	1	0	1	0	1	1	0	0
D_4	0	0	0	1	1	1	1	0
D_5	1	1	0	1	0	0	0	1

2.1.1 Closed Frequent Itemsets

Closed frequent itemsets are a subset of frequent itemsets, which are much lesser in number. The number of frequent itemsets is a very important concept for large databases, because as the number of frequent itemsets increases, the number of redundant frequent itemsets increases. So, it would be better if we could prune out the frequent itemsets that are redundant. An itemset is said to be closed if it has no superset with the same support. If an itemset X has a superset Y with the same support, then any superset Z of Y can be considered as redundant because its additional support can be deduced to be equal to the support of $Z - (Y - X)$. Removing closed itemsets has desirable properties because (i) the itemsets that are discarded are truly redundant - they can be regenerated back along with their supports, and (ii) on large datasets the closed frequent itemsets are only a small fraction of all frequent itemsets, this reduces the processing time to a large extent.

2.1.2 Generalized Closed Frequent Itemsets

In this section, we briefly describe the concept of Generalized closed frequent itemsets [19] and motivate its use in document clustering. Typically, a huge number of frequent itemsets are

Table 2.4 Closed Frequent Itemsets

$< w_1 : 3 >, < w_2 : 3 >, < w_3 : 3 >, < w_4 : 4 >, < w_5 : 3 >$
$< w_1, w_2 : 2 >, < w_1, w_3 : 2 >, < w_3, w_5 : 2 >, < w_4, w_5 : 2 >, < w_5, w_6 : 2 >$
$< w_1, w_2, w_4 : 2 >, < w_2, w_3, w_4 : 2 >$

very common in frequent itemset mining process. These numbers make the frequent itemsets impractical for manual examination. Also, since frequent itemsets indicate cluster candidates and the number of clusters need to be small in number, the number of frequent itemsets produced is a concern. Increasing the support ($minsup$) might solve the problem, but it might also lead to a loss of information, which is not desirable.

Research in this area suggests that many of the generated frequent itemsets share support with one or more of their parent itemsets (subsets). These itemsets are uninteresting because they are just a specialization of the parent itemset. The closed itemset framework addresses this problem by pruning redundant frequent itemsets by using equal support pruning technique. That is, if an itemset and its superset have the exact same support, then we prune that itemset.

Though the number of closed frequent itemsets is lesser than the actual number of frequent itemsets, they guarantee no loss of information. Also closed itemsets have been extensively used in previous research [22] for clustering documents. Even though mining closed itemsets reduces the number of frequent itemsets to a certain extent, we argue that the notion of equal support pruning need not be strictly applied in several domains. This is because in several domains exact support equality is rarely achieved. Especially in the document clustering domain, two documents rarely have exactly the same items (keywords).

Generalized frequent itemset: An itemset X is called a generalized frequent itemset if there exists no proper super set Y such that Y has a support with in a tolerance factor (ϵ) range of the support of X.

An example of generalized closed itemsets is shown below. Table 2.2 shows the documents and their words. Closed frequent itemsets mined from this data, along with their supports are shown in Table 2.4. Table 2.5 shows the generalized closed frequent itemsets with ϵ = 1. We can see that out of 12 closed itemsets, 5 got pruned due to the use of generalized closed frequent itemsets.

Table 2.5 Generalized Closed Frequent Itemsets

$< w_4 : 4 >$
$< w1, w_3 : 2 >, < w_3, w_5 : 2 >, < w_4, w_5 : 2 >, < w_5, w_6 : 2 >$
$< w_1, w_2, w_4 : 2 >, < w_2, w_3, w_4 : 2 >$

Experimental studies on this approach [19] show that the number of generalized closed frequent itemsets is far lesser than the number of closed itemsets (especially for large datasets). The approach also guarantees that the supports of all the frequent itemsets can be calculated within a deterministic tolerance factor. There is no such guarantee given by any of the previous approaches. So, there is very less loss of information even as the number of frequent itemsets are drastically reduced [19].

2.2 Clustering

Clustering is one of the most commonly used data analysis techniques. It is the process of partitioning a set of data objects into a set of meaningful subclasses, called clusters. Formally, given a collection of n objects each of which is described by a set of p attributes, clustering aims to derive a useful division of the n objects into a number of clusters. A cluster is a collection of data objects that are similar to one another based on their attribute values, and thus can be treated collectively as one group. Clustering is useful in getting insight into the distribution of a dataset. Clustering may be highly intuitive as well as highly complicated. Clustering is different from other data mining techniques like classification because it is unsupervised. There is no need of any human intervention while clustering a group of objects. Hence clustering is also known as "Unsupervised Learning".

Each person can form different clusters from the same data. E.g. If we are given a group of animals, one could classify them based on their eating habits as herbivores, carnivores, etc while the other could classify them based on the number of legs they have into entirely different groups. Clustering has applications in a wide variety of fields like medicine, botany, sociology, marketing, agriculture, insurance, etc.

2.2.1 Document Clustering

In recent years, due to the rapid increase of online documents and the expansion of the Web, text document clustering has become an important task. Document clustering is the organization of documents into clusters such that documents within a cluster have high similarity in comparison to one another, but are very dissimilar to documents in other clusters.

An example usage of document clustering would be in a news agency that generates a large amount of news articles every day. Lets suppose that the agency wants to organize these articles according to some topic hierarchy, so that it is easy to go through them at a later period of time. Manually doing this task for hundreds of documents every day would be a very tedious job. Document clustering is clearly a solution for this problem because it automatically groups a stream of news articles based on their content similarities. Hierarchical clustering algorithms can also organize documents hierarchically. A parent-child relationship in the hierarchy can be viewed as topics and sub-topics in a subject hierarchy. These kinds of topic hierarchies are very useful particularly for news documents.

Nowadays, document clustering is being applied in many places like, clustering search engine results, for browsing a collection of documents, as a preprocessing step for document classification, and in building open web directories like Yahoo! subject hierarchy and Open Directory project, etc.

A document clustering approach that allows a document to be present in only on cluster is called a *hard clustering*. On the other hand, if a document can belong to muliple clusters, the clustering is called a *soft clustering*. A soft clustering makes more sense in the context of document clustering because a document might contain multiple topics and hence be assigned to multiple clusters corresponding to those topics.

2.2.2 Frequent Itemset based Document Clustering

Research in recent years has tried to apply the useful aspects of both frequent itemset mining and clustering to create highly scalable, fast and accurate algorithms. Around 4 algorithms belonging to this category have been proposed till date (discussed in Section 3.3). All of

them have the same structure: considering frequent itemsets as candidate clusters and then using score functions to refine the clustering. The method in which the frequent itemsets are mined might differ from each other. Some algorithms also propose methods for construction of hierarchy from the clustering.

2.2.3 Cosine Similarity

This section explains the cosine similarity measure and its importance in document clustering. Cosine similarity is one of the most generally used measure for calculating the similarity between text documents. The vector space model, where a document is represented as a vector of keywords is made use here. We make use of Cosine similarity of two documents in some of our score functions. Cosine similarity between two documents is given in eqn. 2.1.

$$CosineSim(\vec{d_1}, \vec{d_2}) = \frac{|\vec{d_1} \cdot \vec{d_2}|}{|\vec{d_1}||\vec{d_2}|} \tag{2.1}$$

where, $\vec{d_1}$ is the vector representation of document d_1 , "." represents the vector dot product, and $|d_1|$ represents the length of the document vector $\vec{d_1}$. The cosine similarity between any two vectors in general represents the cosine of the angle between the two vectors. When computed between two documents, it represents how similar one is to the other. If the documents are very similar, the angle between their corresponding vectors is close to zero and hence cosine similarity is close to 1. On the other hand, if the documents are not similar, cosine similarity values are close to 0. One advantage of using cosine similarity is that the similarities are naturally normalized to be in the range (0,1).

2.2.4 TF-IDF

TF-IDF (Term Frequency-Inverse Document Frequency) is one of the most common measure used in information extraction and text mining. TF-IDF is used to weigh the words of a document based on their occurrance in a document as well as in the entire corpus. There are several variations of this scheme. The most basic one of them is described here.

Let N be the total number of documents in the dataset. Let df_i be the number of documents in which term t_i appears. Let f_{ij} be the number of occurences of the term t_i in document d_j . Then, the normalized term frequency of t_i in d_j is defined as

$$tf_{ij} = \frac{f_{ij}}{\max f_{1j}, f_{2j}, ..., f_{|V|j}} \tag{2.2}$$

where $|V|$ is the vocabulary size of the collection.

The inverse document frequency (idf_i) of a term t_i is given by:

$$idf_i = \log \frac{N}{df_i} \tag{2.3}$$

and the TF-IDF term weight is given by:

$$TFIDF_{ij} = tf_{ij} \times idf_i \tag{2.4}$$

The idea behind TF-IDF is that if a term appears in a large number of documents in the collection, it is probably not important as it is not very discriminative. A term that appears frequently in a document as well as not so frequently in the entire collection has high TF-IDF value as it might highly discriminate a few documents (clusters) from the others.

2.3 Use of External Knowledges to enhance clustering

In the age of information explosion, there is a lot of open knowledge being created online in many forms. Examples of such open knolwedge includes Wikipedia, Open Web Directory and Social content like tags, reviews, etc. Previously, there have been attempts in research literature about the use of such knowledge for clustering documents, classification, enhancement of search results, etc.

In this section, we try to explain the need of the use of an external knowledge. Lets take an example of two documents containing words “Mercedes” and “Jaguar”. The similarity between these two document could be very less because of different words being used, though they talk about the same topic, cars. Thus, the current methods of just considering a document as a bag

of words fails in expressing the semantic relationship between two documents. A simple word matching algorithm doesn't capture the semantics properly as the meaning of a word is not defined explicitly, but depends on the context in which the word occurs. Many words have multiple meanings and hence if we do not consider the context in which the word occurs, it might distort the clustering.

By using knowledge contained in sources like Wikipedia, Open Web Directory, Social Content, etc. we can expand the scope of the document by adding words that indicate the context of the document, thus helping in increasing the accuracy of clustering. In this thesis, we propose a method for using existing knowledge from Wikipedia and Social content like tags, reviews, notes, etc. to enhance document clustering. We present two ways in which the enhancement can be done, showing the benefits and demerits of each one of them. Though we used only two knowledge sources, our method is easily extendable to many other external knowledge sources.

2.3.1 Wikipedia

This section contains the various features of Wikipedia that we make use of in enhancing document clustering. Out of the existing knowledge repositories, we chose Wikipedia because it has the following advantages:

1. Wikipedia is the world's biggest known knowledge repository.

2. It covers a wide range of domains, unlike WordNet (English vocabulary), MeSH (Health), etc.

3. It contains a manually tagged hierarchical categorization system, in which each article belongs to at least one category.

4. It is the most frequently updated and up to date.

5. Its link structure is well built and dense, and contains valuable information.

6. Equivalent concepts are grouped together with "redirect" links.

7. Gabrilovich et al. [9, 10] try to apply feature generation techniques on ODP and Wikipedia to create new features that augment the bag of words. Their application on text classification confirmed that ODP is more noisy when used as knowledge base.

The way we enhance the clustering using Wikipedia is detailed in Section 6.1. The features of Wikipedia that we used in our approach are given below.

Wikipedia Categories: Wikipedia contains preclassified topic labels to each document that we call $categories$. Each document of Wikipedia belongs to atleast one such topic. These can be accessed at the bottom of any Wikipedia page. e.g: $(Federer)^{category}$ = {World No. 1 tennis players, 21st-century male tennis players, Wimbledon Champions, ...}, etc. Since these categories are assigned manually, they make a lot of sense and act as a short summary of the contents of a page.

Wikipedia Links: We define $links$ of Wikipedia to be the words which have an internal hyperlink to a different Wikipedia document. e.g: A page on Federer in Wikipedia has $links$ like $(Federer)^{links}$ = tennis, world no. 1, Switzerland, Grand slam, ..., etc. $Links$ represent the major topics present in a document. We observed that there might be too many links for a page so we considered two approaches for picking up the important links from them: (i) rank them and take the top 10% of them, (ii) consider links only from the first 3 paragraphs of an article. Both these approaches have their own advantages as well as disadvantages.

Table 2.6 shows the results of ranking the terms and considering the top 10% of them and by considering only the first three paragraphs for an article on Roger Federer. We can see from that table that if the links from the first three paragraphs were considered as it is, they are no so clear and they represent a lot of noise like years (2004, 2008), unimportant names like Ivan Lendl, redundant information like "tennis", "ATP number one position", "Association of Tennis professionals"(expansion of ATP), etc. On the other hand, considering the top 10%, apart from being time consuming, could lead to the loss of certain information as certain words like "Rafael Nadal", "French Open" occur as different words, thus losing their importance as n-grams.

Wikipedia contains redirections for various pages. These redirection pages can be used for variety of purposes like finding alternative names, plurals, closely related words, alterna-

Table 2.6 Selecting important links

First three paragraphs	Ranking top words
Swiss, tennis, ATP number one position, Ivan Lendl, Grand Slam, Association of Tennis Professionals 2004 Wimbledon Championships, 2008 Australian Open, Pete Sampras, Rafael Nadal, Andre Agassi	tennis, ATP, Wimbledon, Rafael, Nadal, Pete, Sampras Australian, French, Open

Table 2.7 Redirections for some words

Word	Redirected to	Category
Edison Arantes do Nascimento	Pele	Alternative names
Greenhouse gases	Greenhouse gas	Plurals
Symbiont	Symbiosis	Closely related words
Hitler	Adolf Hitler	Less specific forms of names
AI	Artificial Intelligence	Abbreviations
Colour	Colour	Alternative spellings
Al-Jazeera	Al Jazeera	Punctuation issues

tive spellings or punctuation, etc. Table 2.7 shows the redirections for various words and the category they belong to.

2.3.2 Social Content

With the increased adoption of Web 2.0 and social web, the amount of user generated information is increasing rapidly. This information in the form of tags, reviews, notes, etc is very useful and can be used to provide highly informative meta data about an article/image/video.

Tags are short representatives of articles that summarize the content of the article in a single word or a group of words. Since tags are user given, their quality is very good. They also represents the different perspectives of a user, e.g. An article on the world cup 2003 cricket match between india and pakistan on cricinfo.com has been tagged $cricket$, $india$, $paksitan$, $sachin\ tendulkar$, $saurav\ ganguly$, $world\ cup$, $waqar\ younis$, $wasim\ akram$, $Lords$, $victory$, $United\ Kingdom$, $etc.$ indicating a wide range of additional knowledge like sachin and wasim akram, have played in the match; the match was played at Lords, United Kingdom, etc. Formally, each user u_i can tag an item i_j with a set of tags $T_{ij} = t_1, ..., t_p$, with

a variable number p of tags. After k users tagged i_j, it is described as weighted set of tags $T_{ij} = w_1t_1, ..., w_nt_n$, where $w_1, ..., w_n \leq k$.

Notes are free text describing the content of a web page. Users can write a note to describe what a web page contains. E.g. a note on flickr.com: *A photo and video sharing service site. Now, I use it to manage my photos and videos and publish them on my blog.* A **review** is a free text valuating a web page. Though this kind of annotations can initially look subjective, users tend to mix descriptive texts with opinions. StumbleUpon is a social bookmarking site relying on this kind of meta data. E.g.: a review for flickr.com: *flickr is a great place to share photos, learn more about photography and see the pictures of others.*

Reviews and notes are not as representative as tags because they are not straightforward and contain other unimportant words like opinions, sentiments (adjectives, adverbs), etc. But they do contain a lot of information in them. We preprocessed the notes/reviews, removed stopwords and used only Noun phrases extracted from them. All other parts of speech were ignored as they might add noise to the original document. We find from our experiments in Section 6.4.5 that enhancing document content using notes and reviews improves the clustering performance than the baseline by a good extent.

Highlights are the most relevant parts of a web page. With this kind of meta data, users specify the part of the web page they have special interest in, or the part they have considered relevant. The best-known site using highlights is Diigo. We didn't use highlights as external knowledge in this thesis because most of the web pages in our dataset did not consist of highlights. Otherwise, highlights would make a great source of external knowledge.

2.4 Evaluation Metrics

We used various evaluation metrics for comparing our method with the existing methodologies. Also, using a wide range of metrics for comparison will ensure that our approach is indeed better than the existing approaches. We used three metrics for evaluating our approaches: F-score, Normalized Mutual Information (NMI), Purity and Inverse Purity. In the sections that follow, we discuss each of these measures in detail.

2.4.1 F-score

F-score is one of the most commonly used measure for calculating the clustering quality. It can be used both for flat and hierarchical clusterings. It first identifies the cluster that can best represent a given natural class in the document set. Then, it measures the accuracy of the best cluster against the natural class. Finally, it calculates the weighted average on the accuracy of each natural class. It considers both precision and recall and produces a balanced measure from both of them. For a given set of documents, the Precision, Recall, and F-score are calculated as follows:

Given a particular class L_j of size n_j and a particular cluster C_i of size n_i, suppose n_{ji} documents in the cluster C_i belong to L_j. Then, the Precision, Recall and F-score are given by Equations 2.5, 2.6 and 2.7 respectively.

$$Precision(L_j, C_i) = \frac{n_{ji}}{n_i} \tag{2.5}$$

$$Recall(L_j, C_i) = \frac{n_{ji}}{n_j} \tag{2.6}$$

and F-score is defined as,

$$Fscore(L_j, C_i) = \frac{2 * Recall(L_j, C_i) * Precision(L_j, C_i)}{Recall(L_j, C_i) + Precision(L_j, C_i)} \tag{2.7}$$

The quality of a hierarchical clustering solution is determined by analyzing the entire hierarchical tree that is produced by the clustering algorithm. While computing $Fscore(L_j, C_i)$ in a hierarchical structure, all the documents in the subtree of C_i are considered as documents in C_i. The overall F-score $Fscore(C)$ of the entire clustering, is the weighted sum of the maximum F-score of all the classes.

$$Fscore(C) = \sum_{L_j \in L} \frac{n_j}{|D|} * \max_{C_i \in C} \{Fscore(L_j, C_i)\} \tag{2.8}$$

F-score is always a value between 0 and 1. An F-score of 1 indicates a perfect clustering, where every document belongs to the correct class and an F-score of 0 indicates that no doc-

ument was assigned correctly to its correct class. The higher the F-score, the greater is the quality of the clustering.

2.4.2 Purity and Inverse Purity

Purity and inverse purity are measures used together to depict the accuracy of a clustering. Purity of a clustering is defined as the weighted average of the maximum precision values of each cluster. It is defined as:

$$Purity(L, C) = \sum_{C_i \in C} \frac{n_i}{|C|} \max_{L_j \in L} (Precision(L_j, C_i)) \tag{2.9}$$

Inverse Purity focuses on the cluster with maximum recall for each category, rewarding the clustering solutions that gathers more elements of each category in a corresponding single cluster. Inverse Purity is defined as:

$$IPurity(L, C) = \sum_{L_j \in L} \frac{n_j}{|L|} \max_{C_i \in C} (Recall(L_j, C_i)) \tag{2.10}$$

where $Precision(L_j, C_i)$ and $Recall(L_j, C_i)$ can be calculated using Eqns. 2.5 and 2.6. Purity and inverse purity achieves maximum value of 1 when every cluster has one single member and when there is only one single cluster, respectively. High purity is easy to achieve when the number of clusters is large. Thus, we cannot use purity to trade off the quality of the clustering against the number of clusters. So instead, we use Normalized Mutual Information in such cases.

2.4.3 Normalized Mutual Information (NMI)

Mutual Information is a symmetric measure to quantify the statistical information shared between two distributions. In document clustering, it provides a sound indication of the information shared between a pair of clusterings.

Let $L = l_1, l_2, l_3, ..., l_j$ be the set of classes and $C = c_1, c_2, c_3, ..., c_i$ be the set of clusters. We interpret l_j as the set of documents in l_j and c_i as the set of documents in c_i.

$$NMI(L,C) = \frac{I(L,C)}{(H(L)+H(C))/2} \tag{2.11}$$

where, H is the entropy and I is the mutual information defined as:

$$I(L,C) = \sum_j \sum_i \frac{|l_j \cap c_i|}{N} \log \frac{N|l_j \cap c_i|}{|l_j||c_i|} \tag{2.12}$$

and, the entropy, $H(C)$ is defined as:

$$H(C) = -\sum_i \frac{|c_i|}{N} \log \frac{|c_i|}{N} \tag{2.13}$$

Equation 2.12 measures the amount of information by which our knowledge about the classes increases when we are told what the clusters are. The minimum of $I(L,C)$ is 0 if the clustering is random with respect to class membership. In that case, knowing that a document is in a particular cluster does not give us any new information about what its class might be. Maximum mutual information (=1) is reached for a clustering that perfectly recreates the classes- but also if clusters are further subdivided into smaller clusters. In particular, a clustering with $K = N$ one-document clusters has maximum MI. So MI has the same problem as purity: it favors larger number of clusters of small size.

The normalization by the denominator $[H(L)+H(C)]/2$ in Equation 2.11 fixes this problem since entropy tends to increase with the number of clusters. For example, $H(L)$ reaches its maximum $\log N$ for $K = N$, which ensures that NMI is low for $K = N$. Because NMI is normalized, we can use it to compare clusterings with different numbers of clusters. NMI is always a value between 0 and 1.

This chapter provides the background information required for the material provided in subsequent chapters. We first describe what frequent itemset mining is. Then we describe the various variants of frequent itemsets like closed itemsets and generalized closed itemsets. Next, we describe the various document clustering algorithms using frequent itemsets. We then describe basic metrics like Cosine similarity and tf-idf and motivate their usage in our approaches. Then, we describe what external knowledge sources are and the various types of

features in these knowledge sources that can be used to enhance document clustering. Finally, we describe the various evaluation metrics used in evaluating our approaches.

Chapter 3

Related Work

3.1 Partition based clustering algorithms

Partition based clustering algorithms like k-means were some of the earliest algorithms applied to document clustering. Each document is represented by a feature vector, and can be viewed as a point in a multi-dimensional space. A distance function is defined(e.g. Manhattan Distance, Euclidean distance, etc) to measure the distance from each point to the centroids. In this section, we describe two such partition based algorithms - (i) k-means, (ii) Bisecting k-means.

3.1.1 k-means

The k-means algorithm is described in Fig 3.1.

1. Choose k data points as initial centroids (cluster centroids).
2. For each data point, compute the distance of it from each centroid and assign the point to the closest centroid.
3. Re-compute the centroid using the current cluster memberships.
4. Repeat steps 2-3 until the stopping criterion is met.

Figure 3.1 Overview of k-means algorithm

3.1.2 Bisecting k-means

Bisecting k-means is the best among the family of k-means partitional clustering [24]. The bisecting K-means algorithm starts with a single cluster of all the documents and works in the following manner:

Bisecting k-means is briefly described in 3.2.

1. Pick a cluster to split. Usually, either the largest cluster or the one with the least overall similarity is chosen at this step.
2. Find 2 sub-clusters using the basic k-means algorithm
3. Repeat step 2, the bisecting step, for a fixed number of times and take the split that produces the clustering with the highest overall similarity
4. Repeat steps 1, 2 and 3 until the desired number of clusters is reached

Figure 3.2 Overview of Bisecting k-means algorithm

Both the basic and bisecting k-means algorithms are relatively efficient and scalable. The complexity of both algorithms is linear in the number of documents. In addition, they are easy to implement and hence are widely used in different clustering applications. Some of the major disadvantages of the k-means family of algorithms are

- We have to specify the number of clusters(k) before hand. This can not be known before hand and hence a wrong value of k may affect the clustering accuracy.
- k-means can be used to discover only spherical clusters. It is not suitable for discovering clusters of very different size which is very common in document clustering.
- Moreover, the k-means algorithm is sensitive to noise and outlier data objects as they may substantially influence the mean value, which in turn lower the clustering accuracy.
- It is not trivial to define a distance measure in the high dimensional space. Techniques like TF-IDF have been proposed precisely to deal with such problems.

- Moreover, the k-means algorithm is sensitive to noise and outlier data objects as they may substantially influence the mean value, which in turn lower the clustering accuracy.

- It is not trivial to define a distance measure in the high dimensional space. Techniques like TF-IDF have been proposed precisely to deal with such problems.

- The number of different words in the documents can be very large. Distance-based schemes generally require the calculation of the mean of document clusters, which are often chosen initially at random. In a high dimensional space, the cluster means of randomly chosen clusters will do a poor job at separating documents.

3.2 Hierarchical methods

Hierarchical algorithms are of two types: Agglomerative and Divisive. As the name suggests, agglomerative methods combine two clusters repeatedly to form larger clusters, while divisive methods divide larger clusters into smaller ones. Generally agglomerative clustering methods are popular than divisive methods, because divisive methods have to use a partitioning method like k-means in turn and are hence more computationally intensive. The general algorithm for agglomerative clustering is described in Figure 3.3.

Most hierarchical clustering algorithms are variants of the single-link, complete-link or average-link algorithms. These algorithms differ in the way they characterize the similarity between a pair of clusters. In the single-link method, the distance between two clusters is the minimum of the distances between all pairs of patterns drawn from the two clusters (one pattern from the first cluster, the other from the second). In the complete-link algorithm, the distance between two clusters is the maximum of all pairwise distances between patterns in the two clusters. In average link, the distance between two clusters is the average distance of all pair-wise distances between the data points in the two clusters.

In any case, two clusters are merged to form a larger cluster based on minimum distance criteria. An efficient average-link document clustering algorithm, UPGMA is given in the next section.

3.2.1 Unweighted Pair Group Method with Arithmetic Mean

Unweighted Pair Group Method with Arithmetic Mean (UPGMA) [16] is one of the best algorithms for agglomerative clustering [24]. In this algorithm, the similarity between two clusters is taken to be the average of all distances between pairs of objects in the two clusters, that is, the mean distance between elements of each cluster: the average cosine similarity for all the documents in those clusters. The pseudo-code for UPGMA is shown in fig. 3.3.

$$similarity(cluster_1, cluster_2) = \sum \frac{cosine(d_1, d_2)}{(size(cluster_1) * size(cluster_2))} \tag{3.1}$$

1. Compute similarity between all pairs of clusters
2. Merge the most similar (closest) two clusters
3. Update the similarity matrix
4. Repeat 2 & 3 until only a single cluster remains

Figure 3.3 Overview of an agglomerative clustering algorithm

UPGMA is has a major disadvantage that it is highly computationally intensive because we have to compute the similarity between all pairs of clusters. This makes UPGMA not scalable for large datasets, which are very common in case of document clustering. Hence, even though there is a good performance, UPGMA is not considered as a candidate for document clustering.

3.3 Frequent Itemset based methods

Both partitional and hierarchical clustering methods do not handle the problem of high dimensionality, which is very common in document clustering. Clustering using frequent itemsets has been a topic of extensive research in recent times, for reducing the dimensionality by only dealing with frequent sets of words for clustering.

Frequent itemset mining is an important topic in Data Mining. It finds out the sets of items that occur together frequently. We can use standard approaches like Apriori [1], FP-

growth [11], etc. to mine frequent itemsets from the documents. All the methods for frequent itemset based clustering are based on the intuition that frequently occurring sets of words define topics.

3.3.1 Hierarchical Frequent Term based Clustering

Hierarchical Frequent Term based Clustering (HFTC) [2] was the first algorithm to address the problem of high dimensionality in document clustering using frequent itemsets. HFTC greedily picks up the next frequent itemset (representing the next cluster) to minimize the overlap of the documents that contain both the itemset and some remaining itemsets. A score function is defined based on entropy overlap of the cluster, which indicates the distribution of the documents in that cluster over all other cluster candidates.

The clustering result very much depends on the order of picking up itemsets, which in turn depends on the greedy heuristic used. Then, a simple hierarchy of clusters is constructed with an empty term set in the root (covering the entire database), using frequent 1-term sets on the first level, frequent 2-term sets on the second level, etc. HFTC stops adding another level to the hierarchical clustering when there are no frequent term sets for the next level. The algorithm is tested against simple variants of k-means using F-score and the results do not show an improvement in the accuracy.

3.3.2 Frequent Itemset based Hierarchical Clustering

Though HFTC succeeded in some resolving some problems in document clustering, it failed badly in others. It was not scalable, the clustering accuracy was low and the results were not consistent as the clustering depends on the greedy strategy that we pick. Fung, et al came up with Frequent Itemset based Hierarchical Clustering (FIHC) [6] which outperforms HFTC. In FIHC, instead of choosing clusters sequentially, a document is assigned to a cluster with the highest similarity. Their method consists of two steps, construction of initial clusters and then making the clusters disjoint. The first step of their algorithm is to mine frequent itemsets using

any frequent itemset mining algorithm like Apriori. Then, they define a score function to make the clusters disjoint.

The score function used by FIHC contains the notion of global frequent itemsets and cluster frequent itemsets indicating itemsets which are frequent in the entire dataset and in the cluster respectively. The main motive behind this score function is the basic concept of clustering which maximizes intra cluster similarity and minimizes inter cluster similarity. The clustering thus produced is used in the construction of a hierarchical cluster tree, where the topic of a parent cluster is more general than the topic of a child cluster and they are similar to a certain degree. The tree is built bottom-up by choosing the best parent node at level $k-1$ for each cluster at level k using a score function. Then, they perform steps like child pruning and sibling merging for creating a compact tree.

The performance of FIHC is evaluated on standard datasets using F-score against HFTC, UPGMA and bisecting k-means. FIHC provides an improvement in cluster quality over HFTC, but not comparable to the other two [18]. It also proved to be more scalable when compared to HFTC.

Some of the drawbacks of FIHC are (i) using all the frequent itemsets to get the clustering (number of frequent itemsets may be very large and redundant) (ii) Not comparable with previous methods like UPGMA and Bisecting k-means in terms of clustering quality. [18] (iii) Use hard clustering (each document can belong to at most one cluster), (iv) Determination of different support parameters (like global support, cluster support) is difficult and can influence the quality of the clustering, (v) their score functions are highly computationally intensive.

3.3.3 Topic Directory Construction using Frequent itemsets

In 2004, Yu, et al came up with an efficient algorithm that addresses the drawbacks of FIHC to a certain extent. They proposed the use of closed frequent itemsets for clustering (TDC) [22] and the construction of a topic directory. The closed frequent itemset concept that they used to some extent handles the large number of frequent itemsets produced and also reduces the redundant frequent itemsets. They use a prefix tree data structure called FT-tree (similar to

FP-tree for frequent itemset mining [11]) to estimate the minimum support automatically. This method is based on the fact that the minimum support should be in such a way that all the documents are covered by at least one frequent itemset.

Like in FIHC, they follow a two step approach, the first one for mining the closed frequent itemsets and then using a score function to refine the initial clusters. Their score function is based on the TF-IDF scores of each frequent itemset in a document. They produce a soft clustering by allowing a document to be present in more than one cluster. Then, TDC constructs the hierarchy in the same way as FIHC. Experiments with standard datasets show that the performance of TDC is comparable to that of FIHC, but with improvements in computation time and scalability. In this thesis, we argue that closed itemsets may also be redundant and use the idea of generalized closed frequent itemsets.

3.3.4 Hierarchical Clustering using Closed Interesting Itemsets

Recently Hasan H Malik, et al. proposed Hierarchical Clustering using Closed Interesting Itemsets, (which we refer to as HCCI) [18] which is the current state of the art in clustering using frequent itemsets. They further reduce the closed itemsets produced by using only the closed itemsets which are "interesting". They use Mutual Information, Added Value, Chi Square, etc as interestingness measures of the closed itemsets. They show that this provides significant dimensionality reduction over closed itemsets and as a result an increase in cluster quality and performance.

The major drawback of using the various interestingness measures provided by them is that there might be a loss of information when we reduce the number of closed itemsets. They use methods proposed in [22, 6] to generate the final clustering and the hierarchy. In this thesis, the method we use to reduce the number of closed frequent itemsets guarantees minimum loss of information. We also show the drawbacks in the score functions used by the existing methods and propose new score functions that overcome these drawbacks.

3.4 Use of External Knowledge for enhancing clustering

Research is being done about improving the clustering quality by using a knowledge source to enhance the document representation. Using a knowledge base for clustering would help in including the context about which the document is talking. Some of the most commonly available knowledge sources include WordNet, MeSH, Wikipedia, Open Web Directory, etc. We need to keep in mind that the coverage of the knowledge source in the dataset must be good before enriching the dataset (document content) with an external knowledge, because lesser coverage could lead to introduction of noise in the data and hence a reduction in the cluster quality. In the further parts of this secion, we discuss the various attempts that have been made to enhance document clustering using various knowledge sources and present the advantages/drawbacks of them.

3.4.1 WordNet

WordNet was the first knowledge source to be used for enhancing document clustering. Dave et al. [5] made use of WordNet synsets as features for document representation to improve clustering. They did not perform word sense disambiguation and found that use of WordNet degraded the performance of clustering. Hotho et al. [12] later integrated WordNet into clustering and investigated word sense disambiguation methods and used hypernym relations from WordNet. Their results show a slight improvement in performance compared to the baseline (without using any external knowledges). However since WordNet is limited to English language words, the coverage of the terms in the document to indicate context and word sense disambiguation effect are quite limited. It might also happen that simply replacing/appending the words with their synonym/hypernym leads to loss of information or generalization which is an unwanted effect.

3.4.2 Open Web Directory

Work done by [7, 8] explains the importance of Open Directory Project (ODP) as a better knowledge source over other available ones. Gabrilovich, et al. [8] provides a framework

that views a document as a collection of local contexts and use ODP to perform word sense disambiguation and extract these contexts from the document. Gabrilovich, et al. [7] further enhance the knowledge in ODP with controlled web crawling. Gaurav Ruhela et al. [20] have also used Open Web Directory to improve text clustering. Their approach makes use of the hierarchy of ODP to know the exact context in which a particular word is being used. This helps them to enhance the document vector with related contexts from ODP to produce a better clustering.

3.4.3 Wikipedia

Many efforts have been made for making use of Wikipedia for enhancing document clustering. Gabrilovich and Markovitch [9, 10] propose a method to improve text classification performance by enriching document representation with Wikipedia concepts. However, the techniques they use are highly computationally intensive and need multiple scans over each document. Their method also produces too many related concepts for each document and there is no procedure to pick up those which are important. Moreover, the structural relations in Wikipedia are not fully used by them. Later, [13, 14] presented algorithms that make the most use of the concepts and categories in Wikipedia for document clustering. They propose methods that map documents to concepts in Wikipedia using different techniques and evaluate how each of them works.

3.4.4 Social Content

With the advent of the concepts of Web 2.0 and social web, the amount of social content generated by users on the web has been increasing drastically. User-generated annotations on social bookmarking sites can provide interesting and promising meta-data for clustering documents, images, video, etc. These user-generated annotations include diverse types of information, such as tags, comments, reviews and notes on different blog posts, web pages, images, videos, etc. This social content has a lot of potential information in it because they directly indicate the users intent in short terms.

Attempts are being made to make use of this social content for various purposes. We present here only those works that make use of social content as some kind of background knowledge. To the best of our knowledge, we are the first one to make use of social content for enhancing document clustering.

One of the recent work in the use of social annotations is for web object classification by Yin et al. [21]. Web objects consist of non textual objects like products, pictures, videos, etc. They consider the problem of web object classification as an optimization problem on a graph of objects and their tags. They then propose an efficient algorithm which makes use of social tags as enriched semantic features for the objects, and also infer the categories of unlabeled objects from other labeled objects based on the connection of social tags. Classification of web objects is very challenging because (i) objects like images, videos, etc do not have many features, only a few features like name, date of upload, etc are present, which do not indicate much information about the object, (ii) there are no interconnections between the objects, (iii) there are no default class labels to the objects, (iv) processing the content of the objects for classification would be highly expensive, etc. Tags make our job very easier by acting as meaningful features to these objects. On a graph of objects and their tags, interconnections between objects can be easily seen. Also, frequently occurring tags can be used as class labels for the objects and used in training classification models.

Zubaiga, et al. [25] use tags for web page classification. They propose methods for improving web page classification performance using social content like tags, reviews, notes, etc. In their approach, they just augment the content of the web page with related tags, comments and reviews and feed these extra features to a classifier to obtain an improved performance. Their method is highly computationally intensive because they consider the entire text of the web pages and use social content on that.

Tags have also been used for social interest discovery by Li et al. [17]. Social interest discovery is the problem of finding out the social interests shared by groups of users. This information can be used to connect people with common interests by making friend recommendations, etc. The approach proposed by them is based on the fact that in a social network, human users tend to use descriptive tags to annotate contents that they are interested in. They

used these patterns of frequent co-occurrences of user tags to characterize users common interests. Their approach, when tested on a real social network successfully detected communities of like-minded people.

In this chapter, we discussed the literature that is relevant to the methods that we describe in further chapters. We first discuss the various types of clustering algorithms, like partitional and hierarchical clustering. Then, we discuss the work that has been done till date in clustering using frequent itemsets. We explain the merits and demerits of each of these approaches. Finally, we discuss the usage of various types of external knowledge sources like Wikipedia, WordNet and Open Web Directory to enhance document clustering. In the subsequent chapters, we describe methods that we proposed and make comparisons with related work presented in this chapter where ever necessary.

Chapter 4

A Framework for Document Clustering

In this chapter, we define a framework for document clustering and provide methods to enhance the produced clustering using various external sources of knowledge. Our framework can be seen as a fusion of two broad approaches. In the first approach, we extract topics from the documents, which are treated as initial clusters (further described in Section 4.1.2). We refine these clusters using a score function to get the final clustering. In the second approach, we enhance the document representation using various external knowledge sources like Wikipedia, ODP, Social content, etc. In the remainder of this chapter, we use the terms topics, cluster representatives and initial clusters interchangeably.

Our framework is shown in Figure 4.1. The first approach, Generalized Frequent Itemset based Document Clustering (GenFIDoC), which forms a topic based document clustering approach (described in Chapter 5) is represented by path '1' in the figure. The second approach, which provides the enhancement strategies using external knowledge sources for refining the cluster quality (described in Chapter 6) is represented by path '2' in the figure. Using an external knowledge adds semantic information to these topics and hence the quality of the clustering is improved.

In the figure, the hexagons represent components where various techniques can be used. E.g. for extracting the topics, we can use various standard approaches from literature like TF-IDF/LDA/LSA, etc. Same is the case for refining the clusters. We develop two methods for refining the clusters, explained in detail in Section 5.2.2 and Section 6.1.3.

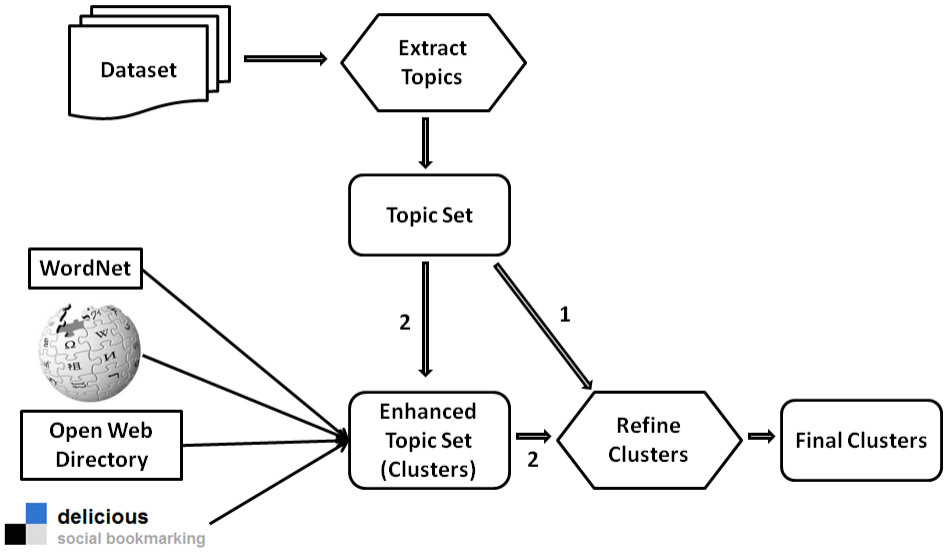

Figure 4.1 Our Framework

In our framework, any kind of external knowledge source can be incorporated. We only require that the external knowledge source provides additional information about documents and this information can be represented as a vector of features.

4.1 Document Clustering

This section describes the first approach of our framework, which performs a topic based document clustering. Our topic based clustering algorithm first extracts the topics from the documents using many standard methods like TF-IDF/LDA/LSA, etc. In Section 4.1.2, we show an equivalence between topics and clusters and hence consider these topics as candidate clusters. Later in Section 4.2, we provide functions to refine these candidate clusters and produce a final clustering.

4.1.1 Topic Extraction

Topic detection and extraction has been a study in text mining and information extraction since a long time. Given the document set, we can extract topics from the documents using various topic detection techniques like TF-IDF, Latent Dirchlet Allocation (LDA) [3], Latent Semantic Analysis (LSA), etc. The topics contained in a document set roughly indicate the clusters present in that set. So, we consider these topics as cluster representatives and the documents containing these topics as the candidate clusters. A detailed discussion supporting this claim is given below.

4.1.2 Topics as clusters

In this section, we formally show that the dual problems of topic detection and document clustering are related very closely. Intuitively, the document clustering problem is to cluster text documents by using the idea that *similar documents share many common keywords*. Alternatively, the topic detection problem is to group related keywords together into meaningful topics using the idea that *similar keywords are present in the same documents*. We show that both these problems are similar below. In the remaining of this section, by "important words", we mean words which are indicative of topics.

For the first problem (clustering), keywords in documents are treated as *items* and the documents (being treated as sets of keywords) are analogous to transactions in a market-basket dataset. This forms a transaction space, that we refer to as **doc-space** as illustrated below, where the d_i's are documents and w_{ij}'s are keywords.

$d_1 - [w_{11}, w_{21}, w_{31}, w_{41}, \ldots]$

$d_2 - [w_{12}, w_{22}, w_{32}, w_{42}, \ldots]$

$d_3 - [w_{13}, w_{23}, w_{33}, w_{43}, \ldots]$

$\ldots$

Then, in this doc-space, the set of important words (set of topics), that are common to a group of documents convey that those documents are similar to each other thereby help in

defining clusters. e.g: if (a, b, c) is a set of frequently occuring keywords, then (d_1, d_3, d_4) which are the documents that contain these keywords form a cluster.

For the second problem (topic detection), documents themselves are treated as *items* and the keywords are analogous to transactions – the set of documents that contain a keyword is the transaction for that keyword. This forms a transaction space, that we refer to as **topic space** as illustrated below, where the d_i's are documents and w_{ij}'s are keywords.

w_{11} – [$d_{i1}, d_{j1}, d_{k1}, \ldots$]

w_{11} – [$d_{i1}, d_{j1}, d_{k1}, \ldots$]

w_{12} – [$d_{i2}, d_{j2}, d_{k2}, \ldots$]

w_{13} – [$d_{i3}, d_{j3}, d_{k3}, \ldots$]

$\ldots$

Then, in this topic-space frequent combinations of documents (i.e., clusters) that are common to a group of keywords convey that those keywords are similar to each other and thereby help in defining topics.

Based on the above analogy between topic-space and doc-space provided above, we see that the set of important words (i.e., set of topics) in the doc-space that are common to a group of documents convey that those documents are similar to each other and thereby help in defining clusters. This analogy between topic-space and doc-space helps us in making the assumption that the topics extracted in the first approach of our framework constitute good candidate clusters.

4.2 Enhancement of the clustering

Now that we have the intial clusters (topics), we refine these using various external knowledge sources to obtain a finer clustering. Traditional approaches for clustering use the bag of words model where the semantic relations between the words is not considered. By enriching the cluster representation, we are adding more information to the clustering that helps us capture the semantics.

The main advantage of our approach over existing approaches is that we are adding semantic information only to the important topics. In all other existing methods [9, 10, 13, 14], the original document representation is changed, which might lead to addition of noise. There might be unimportant words/outliers for which the addition of extra information might lead to distortion of the original clustering. E.g. Consider the following document *"It was an exciting match between India and Pakistan. ... ... Pakistan prime minister Parvez Musharraf awarded the man of the match to Sachin Tendulkar. ..."*. This document is actually about a cricket match. If we enhance the entire document representation, we add unwanted information about Parvez Musharraf, prime minister, etc. which might distort the actual clustering.

Moreover, enhancing the document representation is a tedious process and increases the complexity. Our method will overcome this drawback too because we are processing only keywords/topics which are much fewer than the total words in the document. Using external knowledge is an optional step in our framework and need not be necessarily performed for obtaining the clusters. But, performing this step would increase the clustering quality very much, as shown in Section 6.4. The different types of external knowledge we use and the way we enhance the cluster representation is detailed in Chapter 6.

4.2.1 Refining the clusters

We now have the candidate clusters enhanced using knowledge sources. These candidate clusters were generated by extracting topics from the text and then finding the sets of co-occurring documents. So, it might happen that each document is present in multiple such initial clusters as a document can have multiple topics. In order to avoid this and produce a sharper clustering, we refine these candidate clusters to obtain the final clusters. We can use score functions that allow us to find the importance of a document in a cluster. The refining function that we used in our framework is shown in Equation 4.1.

The general score function that we use for refining the clustering using various external knowledges is shown below:

$$Score(D_i, C_j) = \sum_{k \in C_j, k \neq i} \frac{sim\,(D_{ij}, D_{kj})}{size(C_j)} \tag{4.1}$$

where D_i is the document that belongs to a cluster C_j. D_{kj} represents all other documents in C_j, where

$$sim(d_i, d_j) = csim(d_i, d_j)^{word} + \alpha * csim(d_i, d_j)^{feature_1} + \beta * csim(d_i, d_j)^{feature_2} + \dots \tag{4.2}$$

and $csim(d_i, d_j)^{word}$ represents the cosine similarity between the word vectors of the two documents, $feature_1, feature_2, ...$ represents the different features of the external knowledge that we use, like $categories$, $links$, $tags$, etc. If in Eqn. 4.2, we set $\alpha = 0, \beta = 0, etc$, we get the clustering that considers the bag of words model and is indicated by path "1" in Figure 4.1. We describe the details of Eqn. 4.2, like the various types of features possible and the different external knowledges that we use in the Chapter 6.

This chapter presents a framework for clustering documents. Our framework is a fusion of two broad approaches. The first approach is a topic based clustering algorithm and the second approach explains methods for enhancing clustering using various external knowledge sources. The enhancment step of our framework is optional and needs to be done only if we are not satisfied about the quality of the clustering. Recall our framework shown in Figure 4.1.

The subsequent chapters (Chapters 5, 6) indicate specializations of ideas provided in this framework. Chapter 5 provides an algorithm (which we call GenFIDoC) for clustering documents using frequent itemsets. This is a specialization of path 1 in our framework (shown in Figure 4.1), where frequent combinations of keywords are considered as topics. Chapter 6 gives an algorithm for enhancing clustering using various external knowledge sources, which is a specialization of path 2 of our framework.

Chapter 5

GenFIDoC - Generalized Frequent Itemset based Document Clustering

This chapter explains a specialized implementation of the first approach of our framework described in Chapter 4 - GenFIDoC: A Generalized Frequent Itemset based Document Clustering approach. The first approach of our framework consists of a topic based document clustering algorithm. We use the idea of frequent itemsets here to extract topics and produce a clustering. The rest of the chapter is organized as follows: Section 5.1 explains the theory behind using frequent itemsets for clustering. Section 5.2 explains our hierarchical document clustering using frequent itemsets. Section 5.3 explains the various processing steps involved in the construction of a hierarchy. Finally, we evaluate our approach in Section 5.4.

5.1 Theory behind Frequent Itemset based Clustering

In this section, we formally describe the problem of itemset-based clustering of documents. Our formulation captures the essence of related methods (described in Section 3.3) in an elegant manner. We have already shown the similarity between topic detection and document clustering in Section 4.1.2. Here, in particular, we show that the dual problems of document clustering and topic detection are related very closely when seen in the context of frequent itemset mining.

Intuitively, the document clustering problem is to cluster text documents by using the idea that *similar documents share many common keywords*. Alternatively, the topic detection problem is to group related keywords together into meaningful topics using the idea that *similar keywords are present in the same documents*. Both these problems are naturally solved by utilizing frequent itemset mining as follows.

For the first problem, keywords in documents are treated as *items* and the documents (being treated as sets of keywords) are analogous to transactions in a market-basket dataset. This forms a transaction space, that we refer to as **doc-space** as illustrated below, where the d_i's are documents and w_{ij}'s are keywords.

d_1 – [$w_{11}, w_{21}, w_{31}, w_{41}, \ldots$]

d_2 – [$w_{12}, w_{22}, w_{32}, w_{42}, \ldots$]

d_3 – [$w_{13}, w_{23}, w_{33}, w_{43}, \ldots$]

...

Then, in this doc-space, frequent combinations of keywords (i.e., frequent itemsets) that are common to a group of documents convey that those documents are similar to each other and thereby help in defining clusters. e.g: if (a, b, c) is a frequent itemset of keywords, then (d_1, d_3, d_4) which are the documents that contain these keywords form a cluster.

For the second problem, documents themselves are treated as *items* and the keywords are analogous to transactions – the set of documents that contain a keyword is the transaction for that keyword. This forms a transaction space, that we refer to as **topic space** as illustrated below, where the d_i's are documents and w_{ij}'s are keywords.

w_{11} – [$d_{i1}, d_{j1}, d_{k1}, \ldots$]

w_{11} – [$d_{i1}, d_{j1}, d_{k1}, \ldots$]

w_{12} – [$d_{i2}, d_{j2}, d_{k2}, \ldots$]

w_{13} – [$d_{i3}, d_{j3}, d_{k3}, \ldots$]

...

Then, in this topic-space frequent combinations of documents (i.e., frequent itemsets) that are common to a group of keywords convey that those keywords are similar to each other and thereby help in defining topics.

Table 5.1 Document Clustering and Topic Detection

Frequent Itemset Mining	Document Clustering	Topic Detection
Item	Keyword	Document
Transaction	Document	Keyword
Frequent Itemset	Document Cluster	Topic

In this context, the following lemma formally captures the relationship between the doc-space and topic-space representations.

Lemma: If $(w_1, w_2, w_3, \ldots)$ is frequent in the doc space, $(d_i, d_j, d_k, \ldots)$, the documents containing these words will also be frequent in the topic space and vice versa.

Proof: Let $W = (w_1, w_2, w_3, \ldots, w_n)$ be frequent in the doc space and $D = \{d_1, d_2, d_2, \ldots, d_m \}$ be the corresponding set of documents for these frequent words.

$\Longrightarrow$ In the topic space, $(d_1, d_2, d_2, \ldots, d_m)$ occurs in each transaction $w_i \in W$.

$\Longrightarrow$ Each $d_i \in D$ occurs in atleast n transactions.

$\Longrightarrow$ This means D is frequent in the topic space for all minimum supports $\leq n$, where n is the length of W.

The above analogies between document clustering, topic detection and frequent itemset mining are summarized in Table 5.1.

5.2 Hierarchical Document Clustering using Frequent itemsets

In this section, we present our hierarchical document clustering algorithm. The frequent itemset based document clustering approach is a topic based clustering approach (described in Section 5.1) and can be integrated as the first step in our framework described in the previous chapter.

We explain the series of steps in our process in detail in each of the subsequent sections. Documents contain hundreds-thousands of words. We represent the set of documents as a transaction database, where each item is either present or absent (shown in Table 2.3). We then use frequent itemset mining on that database and find out frequently occurring sets of

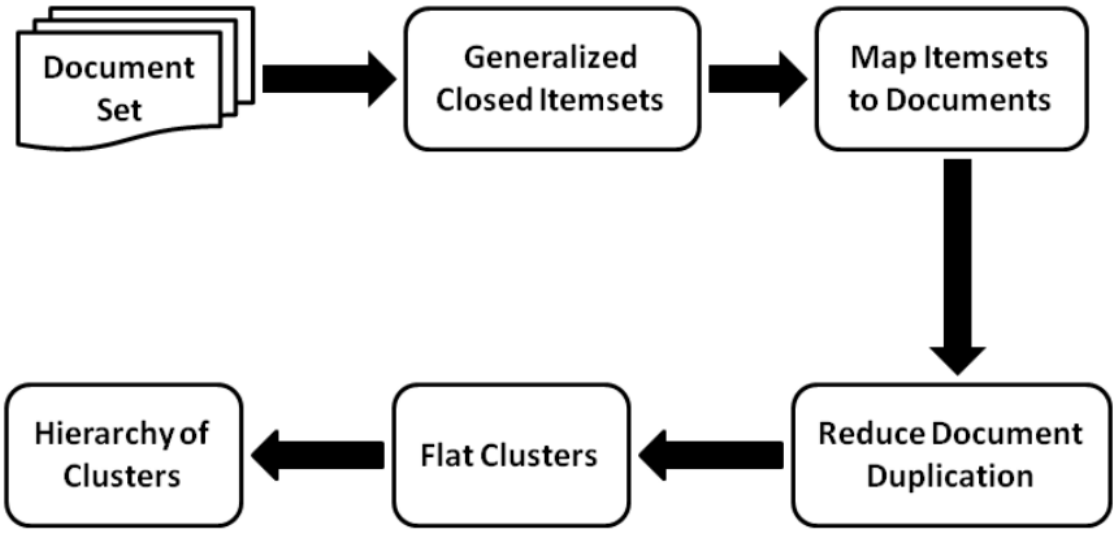

Figure 5.1 A block diagram of the Frequent Itemset based approach

words. These frequently occurring patterns form the initial (candidate) clusters (described in Section 5.2.1). These clusters are very coarse and there is a lot of overlap between them. In order to refine these candidate clusters, we present our score function for refining the clusters in Section 5.2.2.

Figure 5.1 shows a block diagram of our approach. Notice that this is a specialization of the overall framework in Figure 4.1 where it is represented by path 1.

5.2.1 Mining generalized closed itemsets and formation of initial clusters

After converting all the documents into the representation given in Table 2.3, we perform a closed frequent itemset mining using the "CHARM" toolkit [23]. Though the number of closed frequent itemsets is lesser than the actual number of frequent itemsets, they guarantee no loss of information. Also closed itemsets have been extensively used in previous research [22] for clustering documents. Even though mining closed itemsets reduces the number of frequent itemsets to a certain extent, we argue that the notion of equal support pruning need not be strictly applied in several domains. This is because in several domains exact support equality

is rarely achieved. Especially in the document clustering domain, two documents rarely have exactly the same items (keywords).

The number of clusters produced by any clustering algorithm needs to be small in number, yet represent the data compactly and perfectly. Many methods do this by removing some clusters or by merging some clusters but these will hinder the idea of natural clustering. We need a method that can actually merge clusters only because they are naturally closer and not to reduce the number of clusters. We also need the method to be quick enough with the least possible number of computations. To address all these problems, we use the concept of Generalized Closed Itemsets [19] (discussed in Section 2.1.2) to prune the redundant frequent itemsets and produce the initial clusters. The generalized closed frequent itemsets that are obtained are *compact* and are representative of the entire data.

Now that we have the frequent itemsets (sets of keywords), we have to map them to sets of documents. Using the ideas from Section 5.1, we map these keyword sets to document sets. The mapping process is based on the equalivalence of topic-space and doc-space mentioned in Section 5.1. e.g. if (a, b, c) is a frequent itemset, the documents that contain (a, b, c), say (d_i, d_j, d_k) forms a cluster. So, all such document sets constitute the initial clusters.

5.2.2 Removing document duplication

The initial clusters formed have a lot of overlaps between themselves leading to a document being present in multiple clusters. Since our method aims at a soft clustering, we need to make sure that each document is present in a certain small number of clusters only. In order to do this, we propose a score function to rank the documents based on their importance in the cluster.

TDC [22] uses a TF-IDF score based approach to limit document duplication to a maximum of max_dup (the maximum number of clusters in which a document can occur), a user defined parameter. A major drawback of this approach is that they are using the same max_dup for all the documents. But in a general scenario, all documents are not the same and hence fixing a single max_dup value is not right.

E.g. If a document A is present in 25 clusters and a document B is present in 5 clusters, A is an important document because it covers more number of topics. If we set max_dup to be 5, then we are neglecting the importance of A because A is restricted to 5 clusters only. So, instead, we use a threshold which is the percentage of clusters in which a document is present. e.g. In the above example if we change max_dup to be 20%, we can see that A can be present in 5 clusters and B can be present in 1 cluster. The score function used by TDC [22] and HCCI [18] is shown in Eqn. 5.1.

$$Score(d, T) = \sum (d \times t) \tag{5.1}$$

where $d \times t$ denotes the TF-IDF score of each t in T, the frequent itemset in document d.

This score function also has a drawback that the length of the frequent itemset is not being taken care. e.g. If a document d_1 belongs to 2 frequent itemsets say (a, b, c) and (d, e, f, g, h, i, j, k). Lets say the TF-IDF scores of d_1 for (a, b, c) be 2, 2 and 3 respectively. So the total score of d_1 for (a, b, c) is 7. On the other hand if the score of d_1 for (d, e, f, g, h, i, j, k) is 1 for each word, the score of d_1 in this frequent itemset will be 8. But this is not correct because generally a document explains about a set of topics (which could be small) and refers to many other topics (sub topics). But we are interested in finding the topics that are explained by the document rather than to those which are referred. Hence it would be better if we consider the frequent itemset with a small number of topics with a higher individual scores to the topics (a, b, c) than a bigger frequent itemset with lesser individual scores (d, e, f, g, h, i, j, k). We address the above problems by using the score function in Eqn. 5.2.

$$Score(d_i, T) = \sum_{t_j \in T} \frac{tfidf\,(t_j, d_i)}{len(T)} \tag{5.2}$$

where, T is the frequent itemset, d_i is a document and $len(T)$ indicates the length of the frequent itemset T. So, the score of a document in a cluster (frequent itemset) is the sum of TF-IDF's of each of the words (t_j's) in that document divided by the length of the frequent itemset.

Table 5.2 Removing redundant documents

Cluster Label	Documents
(a)	(d_1, d_2, d_3, d_5)
(b)	(d_1, d_3, d_4, d_5)
(c)	(d_3, d_4, d_5, d_6)
(e)	(d_2, d_4, d_6, d_7)
(a, b)	(d_1, d_3, d_5)
(c, e)	(d_4, d_6)
(a, b, c)	(d_3, d_5)

E.g. if (a, b, c) is a frequent itemset and (d_1, d_3, d_4) is the corresponding document set, score of d_1 in (a, b, c) is the sum of TF-IDF scores of a, b, c in d_1 respectively. The top $max\_dup\%$ of documents having the highest scores are put into their respective clusters by this score.

5.3 Generation of Hierarchy

The set of clusters produced in the previous stages can be viewed as a set of topics and subtopics contained in the dataset. This section first discusses various methods that we used to process the clusters. Then, we explain methods for construction of a hierarchy of topics based on the similarity among clusters. All the post processing steps are optional and there is a trade-off between the time taken and the quality of the hierarchy.

5.3.1 Removing redundant documents in the hierarchy

If multiple nodes in the same path of a hierarchy contain the same set of documents, to minimize redundancy in the hierarchy, we place such documents in the lowest node of the hierarchy and remove them from all other nodes. From the generalized closed frequent itemsets obtained, we find those itemsets whose supersets exists with the same document and remove the document from the subset itemset. E.g. In Table 5.2, (a), (a, b) and (a, b, c) have the same set of documents (d_3, d_5). So, using the above idea, we remove documents (d_3, d_5) from a and (a, b) and place them only in (a, b, c).

5.3.2 Reducing overlap between clusters

A general frequent itemset mining algorithm produces a large number of frequent itemsets of smaller lengths. Even after mining the generalized closed frequent itemsets, we found that the number of such smaller length frequent itemsets is large. These itemsets do not indicate much information as they have a lot of overlap between the document sets they represent. So, merging such nodes would help us present a compact, yet informative hierarchy to the user. We measure the extent of overlap between two clusters by using the standard Jaccard Coefficient between the two.

$$Overlap(C_i, C_j) = \frac{|C_i \cap C_j|}{|C_i \cup C_j|} \tag{5.3}$$

If $Overlap(C_i, C_j)$ (represented by δ) is greater than a specified overlap threshold, we merge the two clusters by creating a higher level node between the node and the root. We can see that δ is in the range [0,1]: when δ=0, these two clusters are disjoint and when δ=1, these two clusters have the same set of documents. In our experiments, we observed that setting the value of δ=0.7 performs the best, and the clustering accuracy is not sensitive to δ for values $>$ 0.7.

Figure 5.2 shows a sample hierarchy of clusters. In this hierarchy, the clusters (a), (b) and (c), (d) overlap and are merged together to obtain the hierarchy shown in Figure 5.3. Notice that the clusters which were below cluster b are now assigned to cluster a.

5.3.3 Generation of Hierarchy

Having applied a set of processing steps to the clusters, we start building our hierarchy. Here, we build a tree in a top-down fashion, starting with the root node at level 0. For the rest of this section, we represent the frequent itemset corresponding to a cluster by $freq(C_i)$. The root node contains all the unclustered documents. For each subsequent level, the clusters C_i having length of $freq(C_i)$ to be k appear at the k^{th} level. The depth of the tree is the length of the longest $freq(C_i)$ among all the clusters.

The steps in the construction of the hierarchy are:

1. Sort all clusters based on the lengths of $freq(C_i)$.

2. For each cluster (C_i) having length of $freq(C_i)$ to be k, find all clusters (C_{i-1}) with length of $freq(C_{i-1})$ to be $k-1$ and see if $C_{i-1} \subset C_i$.

3. Each such C_{i-1} is a parent of C_i in the hierarchy.

4. Perform steps to remove redundant clusters and reduce overlapping between clusters in the hierarchy using methods in Section 5.3.1 and 5.3.2 respectively.

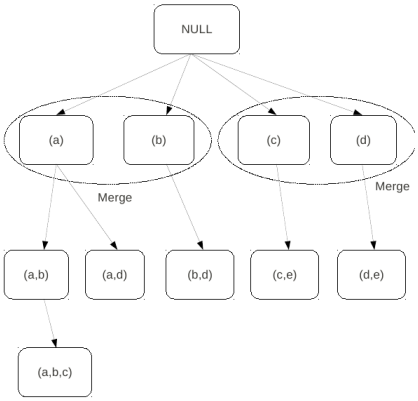

Figure 5.2 Hierarchy of clusters

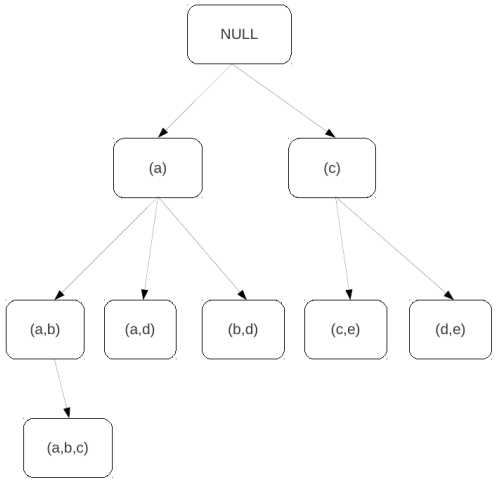

Figure 5.3 Hierarchy after merging similar nodes

5.4 Experimental Evaluation

Various types of experiments were performed to indicate the quality of GenFIDoC. We took a sample dataset consisting of 18 documents picked from 4 domains, (companies, places, birds, players) and experimented our algorithm. We also compared GenFIDoC with standard document clustering algoritms like bisecting k-means and UPGMA, and also with other methods using frequent itemsets (described in Section 3.3). Since there are hundreds of clustering algorithms, we limited our comparison to only those clustering methods that were highly relevant to our approach.

Table 5.3 Composition of our sample dataset

Cluster	Documents
Companies	Google, Microsoft, Yahoo
Places	Mumbai, Chennai, Hyderabad
Birds	Peacock, Pigeon, Parrot
Players	Sachin, Ganguly, Dravid, Federer, Sampras, Nadal

Table 5.4 Results on the sample dataset

Cluster Label	Documents contained in that cluster
City, population, capital	[hyderabad, mumbai, chennai]
Republic, democratic, world	[india, china, america]
India	[sachin, ganguly, dravid, india]
Develop, industry, population, culture	[china, india, mumbai]
Greatest, world, player	[sachin, sampras, federer, nadal]
Habitat, feathers, bird	[parrot, pigeon, peacock]
America, industry	[yahoo, google, america]

The rest of the section is organized as follows: We describe the results of our evaluation on a sample dataset in Section 5.4.1. Section 5.4.2 explains the various features of the datasets that we used. Section 5.4.3 details the experiments and analysis of GenFIDoC.

5.4.1 Evaluation on a Sample Dataset

We evaluated our approach on a sample dataset consisting of 18 documents taken manually from 4 domains, companies, places, birds and players. The exact documents that we took are described in Table 5.3. We picked the documents in such a way that they overlapped to a certain extent. We did so to check if our clustering algorithm would detect such overlap. E.g. The cluster places contains two sub clusters, India, China and America which belong to the cluster countries and Mumbai, Chennai and Hyderabad, which belong to the cluster cities.

Table 5.4 shows the results of our evaluation on the sample dataset. We can see meaningful cluster labels like Greatest, world, player which contains documents related to players.

5.4.2 Datasets

We compared GenFIDoC with the state of the art approaches in document clustering. We used the same datasets used in the previous work in order make a fair comparison. To compare against standard document clustering algorithms like bisecting k-means and UPGMA and against frequent itemset based clustering methods, we used datasets from Cluto clustering toolkit [4]. The results of various algorithms like UPGMA, Bisecting K means, FIHC [6], etc were taken from the results reported in HCCI [18]. Out of the different approaches that HCCI [18] presents for clustering, we considered the ones that perform the best for comparison. We used CHARM toolkit for mining closed frequent itemsets from the data.

Some of the datasets used by FIHC, HFTC, etc were directly from CLUTO toolkit and were in a particular format. We converted all other datasets that we used into that format to ensure consistency with the results of these algorithms. Each document was pre-classified into a single topic, i.e., a natural class. The class information is utilized in the evaluation method for measuring the accuracy of the clustering result. During the cluster construction, the class information is hidden from all clustering algorithms. We applied various preprocessing steps on the datasets like removing stopwords using a dictionary of stopwords and stemming using Porters suffix stripping algorithm.

The $Hitech$ dataset was derived from the San Jose Mercury newspaper articles that are distributed as part of the TREC collection. It contains documents about computers, electronics, health, medical, research, and technology. The Wap and $K1a$ datasets are originally from the WebAce project. Each document in this dataset corresponds to a web page listed in the Yahoo! subject hierarchy. Many recent works have used Wap dataset to represent the characteristics of web pages in a comprehensive comparison of document clustering algorithms. Dataset $Re0$ was extracted from newspaper articles. For this dataset, we only used the articles that were uniquely assigned to exactly one topic for evaluation purpose. Dataset $Ohscal$ contains text related to medicine. Detailed information about the various datasets we used and their properties are provided in Table 5.5.

Table 5.5 Datasets

Dataset	no. of classes	no. of docs	Source
Hitech	6	2301	San Jose Mercury news
Wap	20	1560	WebACE
Re0	13	1504	Reuters-21578
Ohscal	10	11465	Ohsumed-233445
K1a	6	13879	WebACE

5.4.3 Evaluation of GenFIDoC

We compared GenFIDoC with the standard hierarchical document clustering algorithm UPGMA [16] and the standard partitional clustering algorithm bisecting k-means. We also compared our approach with the state of the art algorithms for document clustering using frequent itemsets. We made use of the CLUTO clustering toolkit to generate the results of UPGMA and bisecting k-means. For HFTC and FIHC, we got the original source code from the authors and made use of it to generate the results.

GenFIDoC, stated in Section 5.2 reduces the document duplication and produces the final clusters. We proposed various improvements to the existing methods used in TDC [22] and HCCI [18] and designed a new score function to calculate the score of a document in a cluster. We then applied various processing steps on the clusters obtained to build a compact hierarchy.

We can see from the results in Table 5.6 that our approach performs better than UPGMA, bisecting k-means, FIHC and TDC for all the datasets. Compared to HCCI, our approach performs better for 3 datasets $Hitech$, $Ohscal$, $K1a$ and comparably for the remaining datasets.

Table 5.6 F-Score using TF-IDF scores

Dataset	UPGMA	Bisecting k-means	FIHC	TDC	HCCI	GenFIDoC
Hitech	0.499	0.561	0.458	0.57	0.559	**0.578**
Wap	0.640	0.638	0.391	0.47	**0.663**	0.611
Re0	0.548	0.59	0.529	0.57	**0.701**	0.605
Ohscal	0.399	0.493	0.325	N/A	0.547	**0.583**
K1a	0.646	0.634	0.398	N/A	0.654	**0.693**

In this chapter, we described our first approach for document clustering using frequent itemsets - GenFIDoC. GenFIDoC performs better than existing approaches because:

1. We used the idea of generalized closed itemsets that guarantee a minimum loss of information, unlike the previous methods.

2. We proposed improvements in the way we deal with the max_dup parameter, considering a percentage, rather than an actual number of documents.

3. We normalized the score function in order to avoid a bias towards longer clusters with lesser importance.

4. We used various processing steps that make the clustering more accurate.

The next chapter describes the path "2" described in our framework (Fig 4.1). We use knowledge from external sources like Wikipedia and Social Content to improve the quality of the clustering.

Chapter 6

Enhancing GenFIDoC using various external knowledges

In Chapter 5, we described our document clustering approach in detail. In this chapter, we describe the methods that we used to enhance the document content, which forms the second approach in our framework. Traditional approaches for clustering use bag of words model where the semantic relations between the words is not considered. By enriching the cluster representation, we are adding more information to the clustering that helps us capture the semantics. The external knowledge sources that we tried in this thesis were Wikipedia and Social Content, though practically, any other source that adds information to the documents can be used.

Enhancing the document representation using any external knowledge source is a tedious process and increases the complexity of clustering. The main advantage of our approach over existing approaches is that we are adding semantic information only to the important keywords/topics which are much fewer than the total words in the document.

Also, in all other existing methods, the original document representation is changed, which might lead to addition of noise. There might be unimportant words/outliers for which the addition of extra information might lead to distortion of the original clustering.

E.g. Consider a document about the tennis player Roger Federer. When looked at in the context of clustering tennis players together, words relating to his personal life like birth place, age, etc are unimportant and do not form topic indicating words. If we would have considered all the words of the document, it would have distorted the clustering.

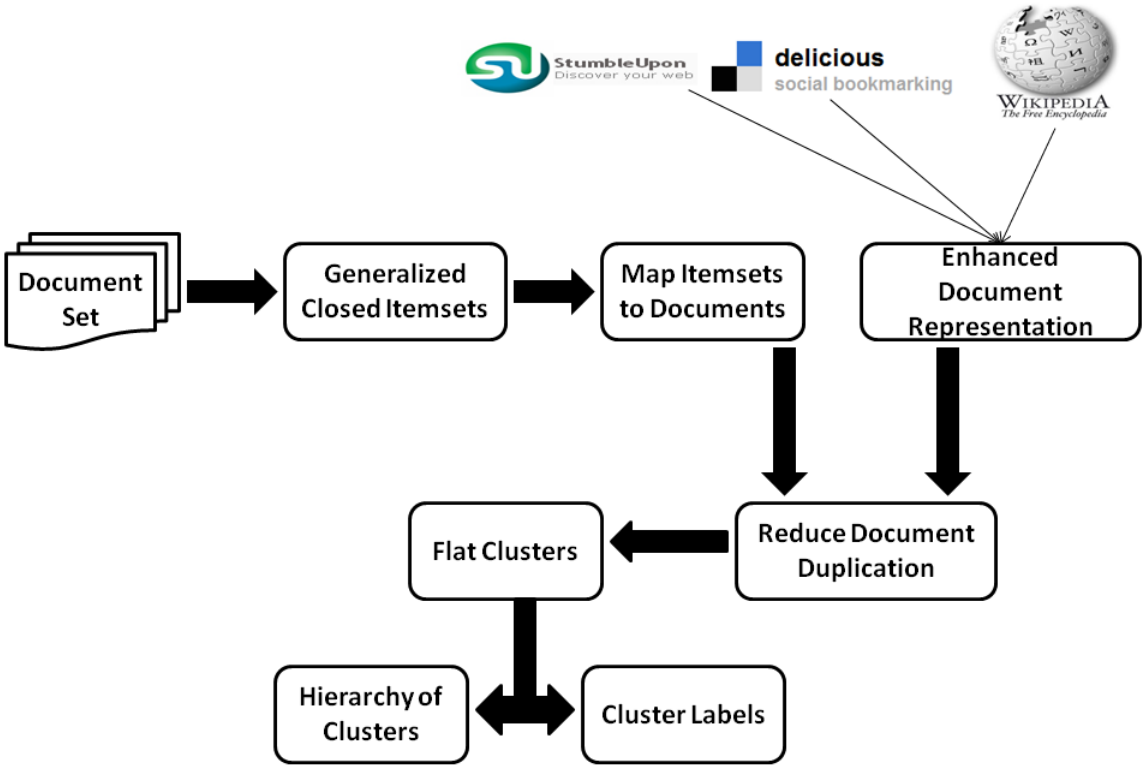

Figure 6.1 A block diagram of our approach using various External Knowledge sources

After enhancing the document content with an external knowledge, we use the score function similar to Eqns. 4.1 and 4.2 to refine the clustering. Figure 6.1 shows the block diagram of our system using any external knowledge source. We describe only the components of enhancing the document representation, reducing the document duplication and labelling of the clusters (shown in the figure) in this chapter. The remaining components have already been discussed in Chapter 5.

The rest of the chapter is organized as follows: We first describe the various features that we used to enhance the document representation using Wikipedia (Section 6.1) and Social content (Section 6.2). Then, we provide score functions that we used to enhance the clustering using each external knowledge. Later, we propose a novel method for labelling the clusters using Wikipedia in Section 6.3. We finally evaluate our approach by comparing with the state of the art methods using external knowledges on various standard datasets in Section 6.4.

6.1 Enhancing using Wikipedia

To enhance the document content using Wikipedia, we map the documents in the dataset to the $categories$ and $links$ in Wikipedia. We had to do the following preprocessing steps to ensure that the mapping of documents to corresponding Wikipedia pages is quick enough.

1. We extracted Wikipedia $links$ and $categories$ from each page of Wikipedia using the Wikipedia XML dump.

2. We built an inverted index separately for $categories$ and $links$ consisting of the title (treated as a bag of words) of the Wikipedia page and the corresponding $categories/links$.

3. We constructed a multi level index on this inverted index for fast access.

4. We built a separate index for all the redirect pages consisting of a particular page P as key and the pages that redirect to P as the value. e.g. The index for the page on Sachin Tendulkar will contain Tendulkar, Sachin Ramesh Tendulkar, S. R. Tendulkar, etc.

Then, for mapping the documents to these categories/links, we experimented with two methods, (i) Word-match and (ii) Topic-match.

6.1.1 Word-match

For each word in a document of the dataset, we check if a Wikipedia page with the same title exists and find the $links$ and $categories$ of that page. We maintain separate vectors for the category and links thus matched. The category and link vectors for a document is the union of the categories and links of all the words in the document. Most of the approaches that used Wikipedia previously (Section 3.4.3) made use this method. But this method has some flaws in it. When we add $categories$ and $links$ about all the words in the documents, the words which are not important (not stopwords) will also contribute to the addition of external knowledge, which might contain noise. Since we are considering only unigrams here for matching, this is an issue to be concerned. Also, since we are adding the $categories$ and $links$ to the entire

document, the category and link vectors thus produced are of very large size and it becomes very difficult to process them.

6.1.2 Topic-match

Instead of taking each word of the entire document, we first find the frequent itemsets in the entire data and then use only these frequent itemsets to find out the corresponding Wikipedia $categories$ and $links$. The frequent itemsets obtained from the data indicate the topics present in the data. So, by adding external knowledge to these, we can have better results. Moreover, we are now dealing with a very low dimension space and hence this method is computationally very effective when compared to the previous approaches. Though we experimented with both topic-match and word-match, we used topic-match as it is better and also fits well into our framework. We also showed this experimentally in Section 6.4.3.

Though redirections provide very good information, we found that only a small percentage of the pages have redirection links. So, instead of introducing a new vector for redirections, we just augmented the redirections of a particular word to the document/topics in Word-match and Topic-match respectively.

6.1.3 Refining the clustering using Wikipedia

Now that we enhanced the topics using knowledge from Wikipedia, we have to use this knowledge to enhance the clustering. To make use of the added knowledge, we propose a score function similar to Eqn. 4.2.

$$Score(D_i, C_j) = \sum_{k \in C_j, k \neq i} \frac{sim\left(D_{ij}, D_{kj}\right)}{size(C_j)} \tag{6.1}$$

where D_i is the document that belongs to a cluster C_j. D_{kj} represents all other documents in C_j. where

$$sim(d_i, d_j) = csim(d_i, d_j)^{word} + \alpha * csim(d_i, d_j)^{links} + \beta * csim(d_i, d_j)^{category} \tag{6.2}$$

In Eqn. 6.1, we divide by $size(C_j)$ in order to normalize the score. An advantage of using this score function is that we compute the similarity of documents unlike in Eqn. 5.2, where we calculate the similarity of a frequent itemset and a document. So this score function automatically handles the problems we explained in Section 5.2.2.

In our experiments we tried out a wide range of values for the parameters α and β and found that setting their values such that $\alpha = 1.2$ and $\beta = 0.7$ yield good clusters. This is because:

- α indicates the similarity between the links which are nothing but the words which contain hyperlinks to other pages in that document. These indicate a brief summary of the entire document. So, they need to have a higher weight than ordinary words.

- β indicates the similarity between the categories of Wikipedia which are generalizations of the topics contained in a document. So, if the categories of two documents match, we cannot be sure as to whether the two documents talk about the same topic as opposed to the case where two words are the same.

6.2 Enhancing using Social Content

Using social content for enhancing document clustering has never been tried before. We use a method to enhance the documents using social content that is similar to Wikipedia. We use tags, notes and reviews in place of categories and links. Unlike for Wikipedia, social content is attached to a document and hence there is no need for any mapping process as such. For each document (web page), we considered the tags and notes given to that web page from Delicious.com and the reviews given to that page from Stumbleupon.com as external knowledge.

Like above, we use two methods here, (i) using the entire content of the documents after stopword removal and stemming, and (ii) Mining the frequent itemsets from the data and using only these. The second method works better because it considers only the topic indicating words and also it deals with a lower dimension space. For each document, the number of tags was huge and hence we had to rank them based on the number of people who gave the same tag

Table 6.1 Sample tags, reviews and notes for the web page "www.bemboszoo.com"

Tags	typography, flash, learning, design, animation, animals, art, kids, alphabets, children, fun, zoo, illustration, graphics, education.
Notes	(i) preschool online book for kids. (ii) Alphabet letters linked to animals - but the animal images are created from the letters themselves. Prep - Year 2. (iii) Choose an alphabet and watch the animal's name turn into a picture of that animal. (iv) This is a great website for younger kids, probably grades PreK-1st or 2nd. It comes up with a page of the alphabet. Students can click on each letter of the alphabet and then a animal whose name starts with that letter will come up.
Reviews	(i) Really neat flash site making animal art out of the letters of the alphabet. (ii) I love this site! (iii) Silly, yet clever and funny. Probably helps if you have some interest in typography, and know that Bembo is a font. The sound-effects add to it. (iv) Animated animals made from letters, That was better than it should have been.

and took the top 20 tags. Notes contain facts about what a web page is about, whereas reviews contain a users opinion about a web page, along with some information about the web page. We processed notes and reviews and removed all stopwords and stemmed them.

Table 6.1 shows the tags, reviews and notes for the web page "www.bemboszoo.com", a site for teaching children alphabets through animations. We can look at the tags and find that most of them are to the point and indicate exactly what the page is about. Notes are like tags, but a bit longer. They too provide good information about what a page is about. Reviews are informative too, but they contain other unwanted information like "I love this site!" and "Silly, yet clever and funny", etc.

6.2.1 Refining the clustering using Social Content

For refining the clustering using Social Content, Eqn. 6.1 is the same, where as Eqn. 6.2 changes to Eqn. 6.3.

$$sim(d_i, d_j) = csim(d_i, d_j)^{word} + \alpha * csim(d_i, d_j)^{tags} + \beta * csim(d_i, d_j)^{notes} + \gamma * csim(d_i, d_j)^{reviews} \tag{6.3}$$

Here, we experimented with various values of the parameters and found that $\alpha > 1$, $\beta < 1$ and $\gamma < 1$ are the values that work the best. This might be because as we discussed earlier, tags are high quality knowledge and indicate a short summary of the page. Thus they are more important than individual words. Reviews and notes may contain some noise and hence they are considered to be of less importance than words in the document.

6.3 Labeling of clusters

Many previous approaches like FIHC [6], TDC [22], HCCI [18] propose a method of creating labels to clusters. The labelling given to a cluster is the frequent itemset to which the documents in the cluster belongs to. We propose a new method for labelling of clusters using Wikipedia which we find to be more efficient.

Though the labels provided by frequent itemset mining to the clusters contain good information, they have many problems like (i) they can be too general or too specific, (ii) they can be very long (frequent itemsets could go up to lengths > 30), (iii) they may contain unimportant words (which are not stop words) (iv) important words (frequently occurring) may not always provide suitable labels or are not meaningful enough for end users, etc.

To address this problem, we proposed a new method of making use of knowledge from Wikipedia to provide better labelling to the clusters. We note that there may be other external sources that can be utilized for this task such as domain-specific knowledge bases, ontologies or even more general sources such as the web. The main reason for focusing on Wikipedia is its attractive ability to provide high quality controlled content. Moreover, Wikipedia content has also been manually annotated by its users. These manual annotations could provide high quality meaningful labels.

The process of candidate label extraction of labels from Wikipedia is described below:

1. Perform all the preprocessing steps mentioned in Section 6.1 to build inverted indices on Categories and Titles on the Wikipedia dump.

Table 6.2 Sample labels to clusters

Manual Label	Using frequent itemsets	Our labels
Buddhism	Buddhist, Buddhism, Buddha Dharma	Buddhism, History of Buddhism, Buddhism in India, Buddhists
Electronics	Voltage, High Voltage, Power Circuit, Shock	Electronics, Power Electronics, Power Supplies, Electronic Terms, Diodes
Tennis	Defeat, Match today, Pete Sampras, Center Court	Tennis players, Tennis terminology, Tennis tournaments, Wimbledon
Bowling	Bowl, Bowler, Bowl Center, League	Bowling, Bowling (cricket), Bowling Competitons, Bowling lane

2. Given a frequent itemset, we search the index to find the set of titles and categories that match the frequent itemset.

3. The candidate label for the cluster consists of the conjunction of titles and categories matched for each item in the frequent itemset.

4. The final labels are generated by ranking the candidate labels based on their relevance in the cluster.

Table 6.2 shows some manual labels to the cluster, labels given by the frequent itemset based approach and the labels extracted by using our approach. We can observe that our labels are much more comprehensive and make sense when generating a hierarchy. We do not claim that our labels are the perfect ones for a clustering, but we just propose a method to give more meaningful labels to the clusters than the existing methods do.

6.4 Experimental Evaluation

In the previous sections, we have developed methods to enhance the clustering using various external knowledge sources. In this section, we try to evaluate these methods by comparing them with the state of the art approaches on standard datasets. Since there are many clustering algorithms that use various types of external knowledge sources, we only took the best ones out of them that use Wikipedia. We evaluated three major aspects of our approach here.

Table 6.3 Results on the sample dataset

Cluster Label	Documents contained in that cluster
State, domestic, international, country	[chennai, america, india, ganguly]
International, world, US, defend	[america, china, sampras]
Population, history, asia, religion	[china, india, peacock]
World corporate headquarters	[microsoft, google,mumbai, Yahoo]
Estimate, million, population	[america, chennai, peacock]

Table 6.4 Corrected clusters after using Wikipedia

Cluster (before using Wikipedia)	Cluster (after using Wikipedia)
Google, yahoo	[Google, yahoo, microsoft]
Sachin, dravid, federer, nadal	[Sachin, dravid, federer, nadal, ganguly, sampras]
india, america, chennai	[india, america, chennai,china,mumbai]

1. Evaluation of clustering results with Word match and topic match (Section 6.4.3).

2. Evaluation of our Wikipedia approach and comparison with other approaches that use Wikipedia for enhancing clustering (Section 6.4.4).

3. Evaluation of the enhancement in clustering using Social content (Section 6.4.5).

6.4.1 Evaluation on a Sample Dataset

We first evaluate our approach on a sample dataset that we took in Section 5.4.1. An example of the clusters obtained before using Wikipedia on the sample dataset shown in Table 5.3 is given in Table 6.3. After using Wikipedia knowledge, such clusters which were grouped together on the basis of just the words in the documents were removed. We also found that some documents that werent a part of the original clusters now belonged to them. An example of such clusters is shown in Table 6.4. We can see that Microsoft, which wasnt present in the cluster Google, Yahoo was added to that cluster after using knowledge from Wikipedia because using only words to cluster may not have had many word-matches in these documents.

Table 6.5 Datasets

Dataset	no. of classes	no. of docs	Source
Reuters-21578	30	1658	UCI ML Repository
20 News Groups	20	19997	UCI ML Repository
Social-odp-2k9	64	12616	nlp.uned.es

6.4.2 Datasets

In order to compare against approaches which use Wikipedia as external knowledge, we used 2 standard document datasets: (i) Reuters-21578 [1] is a news corpus containing 11,367 manually labeled documents classified into 82 clusters, with 9,494 documents uniquely labeled. For our experiments, we chose a subset of 1,658 documents from 30 clusters which contain more than 15 documents and less than 200 documents. We chose the same subset as that of [15, 13] so as to allow a fair comparison with them. (ii) 20-newsgroups dataset [2] contains 19,997 documents classified into 20 classes. We experimented using all the 19,997 documents. We applied general preprocessing techniques like stopword removal, stemming, etc before using them.

For computing the improvement in clustering quality using Social Content, we used the social-odp-2k9 dataset [3] provided in [25]. The dataset consists of 12,616 web pages and the tags, notes provided to them in Delicious and the reviews given to them in Stumbleupon, highlights of the pages from Diigo. We did not make use of the highlights information provided in the dataset as it was provided only for 1920 documents.

Wikipedia releases periodic dumps of its data available at http://download.wikipedia.org. For enhancing the document representation using Wikipedia, we used the latest release of Wikipedia dump (May 2010). Only the English documents from the dataset were considered which amounted to a size of 21GB. All the data was present in XML format. We processed the data and extracted the categories and links out of them.

Detailed information about the various features of each dataset is given in Table 6.5.

[1] http://kdd.ics.uci.edu/databases/reuters21578
[2] http://kdd.ics.uci.edu/databases/20newsgroups
[3] http://nlp.uned.es/social-tagging/socialodp2k9/

Table 6.6 Evaluation on word match and topic match schemes

	Reuters 21578			20 News Group			Social ODP 2k9		
	F	**P**	**I**	**F**	**P**	**I**	**F**	**P**	**I**
Word Match	0.623	0.557	0.618	0.381	0.297	0.426	0.364	0.389	0.325
Topic Match	0.732	0.684	0.778	0.369	0.303	0.422	0.422	0.506	0.414

6.4.3 Evaluation of clustering performance with Word match and Topic match

Table 6.6 shows the clustering performance in terms of F-score, Purity and Inverse Purity using Word match and Topic match schemes for all the datasets. F-score, Purity and Inverse Purity are described in Section 2.4. In the table, F denotes F-score, P denotes Purity and I denotes Inverse Purity. We can observe that the values of F-score, Purity and Inverse Purity using Topic match are better than using Word match, except for 20 News Group dataset, where the values are comparable. Using Word-match scheme may give lesser clustering accuracies because of the addition of noise when the document representation is enhanced. Using Topic match not only improves the clustering accuracy but also provides an improvement in the performance in terms of the amount of computation. Since Topic match performs better, we use it for further evaluation done in the later sections.

6.4.4 Evaluation of enhancement using Wikipedia

We compared our approach using Wikipedia with standard hierarchical document clustering algorithms like UPGMA, partitional clustering algorithms like bisecting k-means, and with other standard frequent itemset based clustering algorithms on datasets given in Table 5.5. In addition to these comparisons, we also compared our approach with the state of the art algorithms that use Wikipedia as external knowledge.

The Wikipedia approach proposed in Section 6.1 uses background knowledge from Wikipedia to enhance the document representation. We expect better results for this approach because of use of an ontology. The results in Table 6.7 compare our enhanced clustering performance with standard document clustering algorithms and other algorithms that use frequent itemsets.

The results illustrate that our approach performs better than all other existing approaches for 4 datasets ($Hitech$, Wap, $Ohscal$ and $K1a$). One drawback in our approach is that it might not be of great use for datasets which do not have sufficient coverage in Wikipedia (like $Re0$).

Table 6.7 F-Score using Wikipedia

Dataset	UPGMA	Bisecting k-means	FIHC	TDC	HCCI	Ours
Hitech	0.499	0.561	0.458	0.57	0.559	**0.691**
Wap	0.640	0.638	0.391	0.47	0.663	**0.695**
Re0	0.548	0.59	0.529	0.57	**0.701**	0.594
Ohscal	0.399	0.493	0.325	N/A	0.547	**0.626**
K1a	0.646	0.634	0.398	N/A	0.654	**0.702**

We also compared our algorithm with the best algorithms for enhancing clustering using external knowledges present till date on different datasets mentioned in Table 6.5. We compared the enhancement on Reuters-21578 dataset with Anna et al [15], Jian Hu et al [13], Gabrilovich et al [9] and on the 20-newsgroup dataset with Xiaohua Hu et al [14]. The results are shown in Table 6.8 and Table 6.9. Those entries in the table containing "-" were either not available or not applicable for that dataset. The results for our approach that we presented here are best values that we obtained from various support threshold ($minsup$) values. Support threshold value is the value above which we consider sets of words to be frequent. In the general framework, this could be described as the threshold above which we consider words as topics. We show the effect of the support threshold values on the performance of our algorithm in Tables 6.10 and 6.11, so as to indicate the variations in our performance with the change in support. Detailed analysis of the results is given in Section 6.4.6.

Table 6.8 Evaluation using Reuters-21578 dataset

	F-score	Purity	Inverse
Bag of Words	0.618	0.574	0.632
Gabrilovich et al	-	0.605	0.548
Hu et al	-	0.655	0.598
Huang et al	-	0.678	0.750
Ours	**0.732**	**0.684**	**0.778**

Table 6.9 Evaluation using 20 News Groups dataset

	F-score	**Purity**	**Inverse**	**NMI**
Bag of Words	0.232	0.286	0.362	0.168
Xiaohua et al	-	0.206	-	0.171
Ours	**0.369**	**0.303**	**0.422**	**0.193**

Table 6.10 Effect of threshold values on performance for Reuters data

Threshold	**F-score**	**Purity**	**Inverse**
0.01	0.636	0.604	0.653
0.02	0.702	0.624	0.745
0.03	**0.732**	**0.684**	**0.778**
0.04	0.681	0.617	0.728
0.05	0.639	0.565	0.681
0.06	0.610	0.518	0.597

6.4.5 Evaluation of enhancement using Social Content

For comparing using social content as external knowledge, we couldn't find any previous work to compare with. We present the results using baseline approach (Bag of Words) and after the enhancement. We compared the baseline performance to the enhanced performance using various types of social content like tags only, reviews only, notes only and tags,reviews and notes combined for two threshold values, 0.1 and 0.5. The results of these evaluations are given in Tables 6.12 and 6.13. Detailed analysis of the results is given in Section 6.4.6.

6.4.6 Discussion

From Table 5.6 and Table 6.7, we can clearly see that the scores using Wikipedia are better than those using the bag of words model (GenFIDoC) when compared on the same datasets. There is a sufficiently good improvement in the F-score because of the use of knowledge from Wikipedia. We justify the lesser scores using Wikipedia for some datasets like $Re0$ as they do not have enough coverage in Wikipedia, i.e. the documents in this dataset do not have corresponding pages in Wikipedia.

We can see from Table 6.8 that our algorithm performs much better than all the existing approaches for the Reuters dataset. From Table 6.9, we can see that our method outperforms

Table 6.11 Effect of threshold values on performance for 20 News Groups data

Threshold	F-score	Purity	Inverse
0.02	0.283	0.154	0.260
0.03	0.321	0.187	0.291
0.04	**0.369**	**0.303**	**0.422**
0.05	0.337	0.244	0.363
0.06	0.279	0.240	0.301
0.07	0.219	0.138	0.215

Table 6.12 Evaluation on Social-ODP-2k9 dataset for threshold 0.1

	F-score	Purity	Inverse
Bag of Words (BOW)	0.241	0.324	0.206
Tags + BOW	**0.422**	**0.506**	**0.414**
Reviews + BOW	0.366	0.391	0.366
Notes + BOW	0.382	0.422	0.385
Tags + Reviews + BOW	0.379	0.403	0.371
Tags + Notes + BOW	0.406	0.452	0.417
All	0.397	0.432	0.362

both Xiaohua et al and the bag of words approach in terms of F-score, Purity and Inverse Purity for the 20-News Group dataset. Our bag of words approach is better than many other approaches because we are considering only important words and neglecting the unimportant/outlier words. Our enhanced representation enriches only these important topics and hence much better than all other approaches.

We can infer clearly from Tables 6.10 and 6.11 that the performance of our approaches increases as we decrease the threshold. This is because reducing the threshold, we are considering more information and hence more topics. The results are not good if we reduce the threshold too much because unimportant topics are added. On the other hand, the performance is not good even if we increase the threshold too much because important topics are missed out due to the high threshold values.

Social content is very useful in enhancing document clustering because they represent shortly the topics that are discussed in a web page. From Tables 6.12 and 6.13, we can see that out of the various kinds of social content, using tags gives the best results. This is because a tag given by the user represents what topics are discussed in the web page. Enhancing topic

Table 6.13 Evaluation on Social-ODP-2k9 dataset for threshold 0.5

	F-score	**Purity**	**Inverse**
Bag of Words (BOW)	0.187	0.283	0.541
Tags + BOW	**0.295**	**0.346**	**0.792**
Reviews + BOW	0.241	0.337	0.690
Notes + BOW	0.279	0.340	0.777
Tags + Reviews + BOW	0.263	0.342	0.717
Tags + Notes + BOW	0.306	0.346	0.782
All	0.291	0.317	0.703

representations using tags will hence be very useful. Reviews and notes are not as useful as tags because they are not straightforward and contain other unimportant words like opinions, sentiments (adjectives, adverbs), etc.

Combining only tags and reviews performs better than just using reviews, same is the case with combining tags and notes. This is because tags being a good source of knowledge, add to the improvement of clustering quality. We can also see from the results that combining all these together also improves the results but not as much as tags. This might be because the information being added by reviews and notes contains some noise. All these approaches are much better than the baseline (bag of words) showing that use of external knowledge produces better clustering.

In this chapter, we described our approach for enhancing document clustering using various knowledge sources like Wikipedia and Social content. Although using external sources provides a significant improvement in the clustering quality, there is a significant improvement in the execution time. It is a drawback between the clustering quality and time taken for the clustering. Though we try to significantly reduce the time taken in the enhancement process over the previous approaches, it is still an overhead. As we described in Chapter 4, this step is optional and needs to be done only if a good quality clustering is needed.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

We presented a framework that first clusters documents and then enhances the clustering using various external knowledge sources. We discussed a formal connection between document clustering and topic detection and used these concepts to show how we can improve document clustering. We do not make any complex assumptions on the properties of either our dataset or the external knowledge. So, our framework can be used with any external knowledge on any dataset. The framework is divided into two broad approaches: the first approach presents a topic based document clustering algorithm and the next presents methods for enhancement of the clustering using various external knowledges.

Then, we presented our own hierarchical document clustering algorithm using frequent itemsets (GenFIDoC) that is a generalization of the first approach of our framework. We used the concept of generalized closed frequent itemsets to reduce the number of frequent itemsets drastically, yet maintain the cluster quality. We argued that these generalized closed frequent itemsets are perfect candidate clusters. But these candidate clusters are very coarse and need to be refined. So, we proposed score functions that overcome the drawbacks of the existing methods and refines the clusters. We then explained methods to efficiently construct a hierarchy of clusters by pruning the redundant clusters. We tested our approaches against standard document clustering algorithms like UPGMA and Bisecting k-means and also against state of the art algorithms on document clustering using frequent itemsets. We found that our re-

sults are comparable to the current state of the art methods and much better than the standard algorithms.

In the next part, we proposed methods to enhance the clustering obtained previously using background knowledge. In this thesis, we used knowledge from Wikipedia and Social Content like tags, comments, reviews, etc. We developed score functions to refine the clustering using these external knowledges. We then proposed a method for labelling the clusters using knowledge from Wikipedia. We evaluated our enhanced clustering against the best algorithms till date that use external knowledges for enhancing clustering. We show that our approach performs better compared to the existing approaches because we consider only the important topics and enhance them using external knowledge. All our results are much better than approaches that do not use any external knowledge. This shows that the addition of generalized terms provides the scope of calculating better similarity values and thus better clustering accuracy.

7.2 Future Work

Our framework presented consists of two approaches, one is a topic-based document clustering algorithm and the other is a method for enhancing the clustering using external knowledges. We proposed a frequent itemset based algorithm for the document clustering approach. We think any other topic based clustering that first enhances the clustering and then refines these topics can be applied here. We compared our work with other document clustering approaches that use frequent itemsets. Comparison with approaches that perform topic-based clustering would have been much wiser.

For the second part, we only used Wikipedia and Social content as external knowledges. Since there is no assumption on the structure of the external knowledge, any kind of external knowledge can be used. There are many other external sources of knowledge like Open Web Directory, WordNet, and domain specific knowledge bases like MeSH, ADAP, etc that we have not explored in our work.

We think that apart from document clustering, our ideas on enhancing using external sources can also be used in other applications like text classification, search, etc. We plan of extending

our ideas to other areas like evolutionary clustering, data streams, etc by using incremental frequent itemsets. We will explore more about utilizing the link structure of Wikipedia in clustering. In future, we plan to parallelize various steps used in our process(like computing similarity between documents, etc) using Hadoop.

One drawback of the approach we provided in this thesis is that it would not perform well if the dataset doesn't contain enough coverage in the knowledge source. Infact, this might degrade the performance. To address this issue, we would like to try other external knowledges, more domain specific ones(e.g. MeSH), in our framework. We plan to design a method that, given a dataset, can select from a set of knowledge sources the one which performs better. Also, we have used only Wikipedia and Social content as external knowledge, though there were many others like WordNet, Open Web Directory, etc that could have been used. This was because most work has already been done using WordNet and Open Web Directory. We would also like to think of better approaches for using other types of external knowledges like Web, where there is a wealth of information into our framework.

Chapter 8

Related Publications

- *Frequent Itemset based Document Clustering using Wikipedia as External knowledge*, In proceedings of 14th International Conference on Knowledge-Based and Intelligent Information and Engineering Systems, UK, 2010. Kiran G V R, Ravi Shankar, Vikram Pudi.

- *Enhancing Document Clustering using various External knowledge sources*, In proceedings of the 23rd International Conference on Software Engineering and Knowledge Engineering, USA, 2011. Kiran G V R, Ravi Shankar, Vikram Pudi.

- *Evolutionary Clustering using Frequent Itemsets*, In proceedings of the SIGKDD workshop on Data Streams (StreamKDD '10), USA, 2011, Ravi Shankar, Kiran G V R, Vikram Pudi.

Bibliography

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. Proceedings of the 20th International Conference on Very Large Data Bases, VLDB, September, 1994.

[2] F. Beil, M. Ester, and X. Xu. Frequent Term-based Text Clustering. Proceedings of International Conference on Knowledge Discovery and Data Mining, 2002.

[3] D. M. Blei, A. Y. Ng, and M. Jordan. Latent dirchlet allocation. Journal of Machine Learning Research, 2002.

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

[5] D. M. P. K. Dave and S. Lawrence. Mining the peanut gallery: opinion extraction and sentiment classification of product reviews. Proceedings of WWW, 2003.

[6] B. Fung, K. Wang, and M. Ester. Hierarchical Document Clustering using Frequent Itemsets. Proceedings of SIAM International Conference on Data Mining, 2003.

[7] E. Gabrilovich and S. Markovitch. Feature generation for text categorization using world knowledge. In International Joint Conference on Artificial Intelligence(IJCAI'05), 2005.

[8] E. Gabrilovich and S. Markovitch. Harnessing the expertise of 70,000 human editors: Knowledge-based feature generation for text categorization. In The Journal of Machine Learning Research, 2006.

[9] E. Gabrilovich and S. Markovitch. Overcoming the Brittleness Bottleneck Using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge. Proceedings of The 21st National Conference on Artificial Intelligence., 2006.

[10] E. Gabrilovich and S. Markovitch. Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. In Proceedings of The 20th International Joint Conference on Artificial Intelligence., 2007.

[11] J. Han, J. Pei, Y. Yin, and R. Mao. Mining frequent patterns without candidate generation. Data Mining and Knowledge Discovery, 2004.

[12] A. Hotho, S. Staab, and G. Stumme. Wordnet Improves Text Document Clustering. Proceedings of Semantic Web Workshop, the 26th Annual International ACM SIGIR Conference., 2003.

[13] J. Hu, L. Fang, Y. Cao, and et al. Enhancing Text Clustering by Leveraging Wikipedia Semantics. Proceedings of 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval., 2008.

[14] X. Hu, X. Zhang, C. Lu, E. K. Park, and X. Zhou. Exploiting Wikipedia as External Knowledge for Document Clustering, In Proceedings of Knowledge Discovery and Data Mining KDD'09. 2009.

[15] A. Huang, D. Milne, E. Frank, and I. H. Witten. Clustering documents using a wikipedia-based concept representation. Proceedings of Pacific Asia Conference on Knowledge Discovery and Data Mining(PAKDD), 2009.

[16] L. Kaufman and P. J. Rousseeuw. Finding Groups in Data, An introduction to Cluster Analysis. New York John Wiley & Sons, Inc, March, 1990.

[17] X. Li, L. Guo, and Y. E. Zhao. Tag-based social interest discovery. Proceedings of WWW, 2008.

[18] H. H. Malik and J. R. Kender. High Quality, Efficient Hierarchical Document Clustering Using Closed Interesting Itemsets. Proceedings of International Conference on Data Mining (ICDM'06), 2006.

[19] V. Pudi and J. R. Haritsa. Reducing rule covers with deterministic error bounds. In Proceedings of 7th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD 03), 2003.

[20] G. Ruhela. Exploring open web directory for improving the performance of text document clustering. Masters thesis, IIIT Hyderabad, 2010.

[21] Z. Yin, R. Li, Q. Mei, and J. Han. Exploring social tagging graph for web object classification. Proceedings of KDD, 2009.

[22] H. Yu, D. Searsmith, X. Li, and J. Han. Scalable Construction of Topic Directory with Nonparametric Closed Termset Mining. Proceedings of International Conference on Data Mining (ICDM'04), 2004.

[23] M. J. Zaki and C.-J. Hsiao. Efficient algorithms for mining closed itemsets and their lattice structure. IEEE Transactions on Knowledge and Data Engineering, 2005.

[24] Y. Zhao and G. Karypis. Evaluation of Hierarchial Clustering Algorithms for Document Datasets. Proceedings of International Conference on Information and Knowledge Management, November 2002.

[25] A. Zubiaga, R. Martinez, and V. Fresno. Getting the most out of social annotations for web page classification. Proceedings of DocEng 2009, the 9th ACM Symposium on Document Engineering,, 2009.