Query-Context Aware Strategies for Performance Improvement of Search Engines

THESIS

Submitted in partial fulfillment
of the requirements for the degree of
DOCTOR OF PHILOSOPHY

by

N.MEHALA

Under the Supervision of

Prof. Poonam Goyal

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE
PILANI (RAJASTHAN) INDIA
2015

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE
PILANI (RAJASTHAN)

CERTIFICATE

This is to certify that the thesis entitled **"Query-Context Aware Strategies for Performance Improvement of Search Engines"** submitted by **N. MEHALA** ID. No. **2008PHXF004P** for award of Ph.D. Degree of the Institute, embodies original work done by her under my supervision.

(Signature in full of the supervisor)
Prof. Poonam Goyal

Associate Professor
Department of Computer Science and Information Systems
Birla Institute of Technology and Science-Pilani
Pilani – 333 031 (Rajasthan) INDIA

Date:

Dedication

Dedicated to my mother Late Vijayakumari, who wanted to see me educated and my father R. Natarajan who believed in equal education right for girl and boy child.

Acknowledgements

First and above all, I thank God for being with me every step of my life. It is my great pleasure to express my deep and sincere gratitude to those who made this PhD dissertation possible.

Foremost, I am deeply grateful to my supervisor Prof. Poonam Goyal for her continuous support and wonderful guidance throughout my entire PhD study. Prof. Poonam Goyal is a great advisor who is inspiring, perceptive, and patient. Prof. Poonam Goyal gave me the greatest encouragement to explore the fabulous research area of Information Retrieval and tremendously helped me to focus on the essential things. Most importantly, I learnt how to be a rigorous scholar. She is not only an academic advisor, but also a role model and a lifetime mentor.

I wish to express my sincere gratitude to Prof. Navneet Goyal, for his encouragement, brilliant ideas, valuable comments and his time.

I would also express my warmest gratitude to Prof. T.S.B Sudarshan and Prof. Shanmuga Sundar Balasubramaniam who introduced me to the wonderful field of Information Retrieval and gave me important guidance and encouragement during my initial attempts in academic research.

My sincere gratitude to Prof. Sudeept Mohan and Prof. J.P. Misra for providing suggestions and resources whenever it was required. Further, I sincerely thank members of the doctoral advisory committee for their critical remarks and for helping me in improving the work.

Special thanks are due to Prof. B. N. Jain, Vice Chancellor; Prof. A. K. Sarkar, Director; Prof. S.K. Verma, Dean, Academic Research Division, Prof. Rahul Banerjee, Head of Department, Computer Science & Information Systems, Birla Institute of Technology and Science, Pilani (BITS Pilani), (Raj.), for giving me an opportunity for PhD program.

I owe my warmest thanks to my entire family for their love and understanding. A special thanks to my in-laws, Sulochana & Chennaiah, for appreciating my endeavor towards research and for their care and concern. I am so grateful for unconditional love, patience, sacrifice and enormous support from my husband D.C. Kiran. He is my life time friend who stands always my side. Thank you for your encouraging words, which pushed me to work consistently in hard times.

I also thank my son K. Mano Srijan, who just turned to eight-year old, who missed mother's presence at most of the time and for not troubling too much when mom had to work.

My final debts to my mother (Chandra), grandmother and brothers for their support in every stage of my life.

Over the last almost nine enriching years at BITS Pilani, so many other people and things have been a part of this experience as well, that it is hard to choose and name only a few. My colleagues in the department, friends, students and the time we spent together will forever remain precious to me.

Abstract

Query-Context Aware Strategies for Performance Improvement of Search Engines

For a search engine, the challenge of finding relevant information from the web is becoming more and more difficult with rapid increase and change in the content of the web. This difficulty further increases as queries submitted by users are general, imprecise, short and ambiguous. Relevance between user's information need and documents returned by search engine is largely dependent on the query given by them. It has been shown in the literature that the terms in the queries alone are not good descriptors of the information needs of the users. Efficiency of search can be improved by formulating the query appropriately. One of the ways of formulating the query effectively is by query recommendation. So, we have extracted and selected query related concepts that are descriptive terms from the web snippets by measuring the relevance between query and concepts. These concepts can be recommended as query suggestions. As queries may have multiple and dynamic intentions, search engines are not able to understand the exact user context, and thus retrieve large volumes of results, most of which are irrelevant to the user. This problem can be resolved by grouping the search results based on the different intentions of the query and presenting grouped results every time whenever user poses the query. The work presented focuses on an approach that first identifies the associated topics of the query and represent them in the form of a set of concepts and then forms groups of documents by assigning each document to the appropriate topic and in the end it provides suitable labels to these topics. The query formulation effectiveness can be further improved through finding similar queries issued by the users by effectively exploiting the information lying in the query log/click-through log. This leads to a number of problems in dealing with similar queries. The problems include lack of common keywords, selection of different documents by the search engine and lack of common clicks, etc. These problems render traditional query clustering methods unsuitable for query recommendations. In our work, we have focused on a new query recommendation system. For this, we have identified conceptually related queries by capturing users' preferences using click-through graph containing extracted best features relevant to the queries from the snippets.

Query clustering is done by a new tripartite agglomerative clustering algorithm – Query-Document-Concept clustering (QDC Clustering) where the document nodes are used to decouple queries and features/concepts in a tripartite graph structure. This results in clusters of similar queries, associated clusters of documents, and clusters of concepts. Thus, it also produces document clusters. Documents in these document clusters are close to the context of the associated query clusters. A query context is represented as a set of features obtained from documents. This is one of the ways of doing query-context aware topical document clustering. Query recommendation system has been modelled in four ways: non-personalized content-ignorant models, personalized content-ignorant models, non-personalized content-aware models and personalized content-aware models.

Clustering relevant documents is also an essential step in improving efficiency and effectiveness of the information retrieval system. Topical document clustering and soft clustering are techniques to shape clusters into coherent and natural clusters. In our work, we have proposed a two-phase Split-Merge (SM) algorithm, which can be applied on the obtained topical clusters, to produce refined and soft topical document clusters. The proposed technique is a post processing technique which acquires the advantages of both document-pivot and feature-pivot based topical document clustering approaches. In SM algorithm, split phase splits the topical clusters by relating them to the topics which are obtained by disambiguating the web search results and also converts clusters into homogeneous soft clusters. Similar topical document clusters are then merged using similarity among topical clusters by feature-pivot approach.

Traditional query clustering algorithms are designed to work on previously collected data from query stream. These algorithms become less and less effective with time because users' interests, query meaning and popularity of topics etc., change over time. So, there is a need for incremental algorithms which can accommodate the concept drift that surface with new data being added to the collection without performing a complete re-clustering. We have proposed an incremental model for query and query-context aware document clustering. The model periodically updates new information efficiently and can be applied in a distributed environment. The proposed model can be applied to the results of hierarchical clustering algorithms that produce query, query based document, topical, and topical document clusters.

Last but not the least, performance of the web information retrieval system heavily depends on the organization of the web documents. Hierarchy generation is a potential solution to this.

Manually built hierarchies are popular, but they have their own problems. This work focuses on automatic & dynamic generation and refinement of a hierarchy. The hierarchy built is a topic hierarchy having nodes as topics described by the concepts obtained from the short texts such as snippets and titles. The snippets of documents, clicked for the queries, are clustered through query-context aware hierarchical clustering algorithms and then refined using a post processing technique to have coherent soft topical clusters representing one topic each. In this work, top four levels of the Open Directory Project hierarchy is used as base hierarchy. Obtained topical clusters are inserted into the base hierarchy by establishing relations with the categories in the hierarchy iteratively. We have also focused on a refinement method to reorganize the hierarchy to deal with the problems of topic drift and reduced homogeneity.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

Abbreviation	Details or Expanded Form
CatColl	Category Collection
CD	Coverage Degree
CE	Category Entropy
CQ	Core Query
CSC	Common Semantic Cotopy
DF	Document Frequency
F	F-Measure
GUI	Graphical User Interface
IDF	Inverse Document Frequency
IM	Incremental Model
IM	Importance Measure
IMC	Incremental Model with Clustered Data
IMR	Incremental Model with Raw Data
LF	Lexical F-/measure
LP	Lexical Precision
LR	Lexical Recall
M	Concept Similarity Matrix
MAP	Mean Average Precision
MI	Mutual Information
O	Topic Overlap
ODP	Open Directory Project
P	Precision
P_i	i^{th} Partition
PQC	Personalized Query-Concept
PQD	Personalized Query-Document
PQDC	Personalized Query-Document-Concept
PS	Popularity Score
QC	Query-Concept
QColl	Query Collection
QD	Query-Document
QDC	Query-Document-Concept

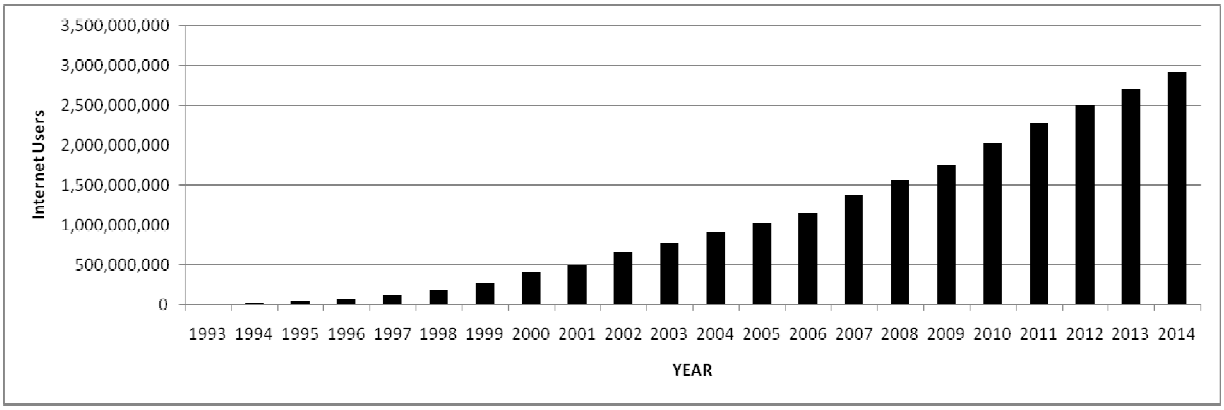
Abbreviation	Details or Expanded Form
QR	Query Recommendation
QRS	Query Relevance Score
R	Recall
R	Resultant Hierarchy
S	Support
SC	Semantic Cotopy
SM	Split-Merge
SRC	Search Result Clustering
STC	Suffix Tree Clustering
T	Target Hierarchy
TF	Term Frequency
TF	Taxonomic F-Measure
TLS	Topic Label Score
TO	Taxonomy Overlap
TP	Taxonomic Precision
TR	Taxonomic Recall
TS	Topic Set
W1	Weight Function 1
W2	Weight Function 2
WF	Weight Function
WPQC	Personalized Weighted Query-Concept
WPQD	Personalized Weighted Query-Document
WPQDC	Personalized Weighted Query-Document-Concept
δ	Concept Selection Threshold
δ_A	Document Assignment Threshold
δ_c	Correlation Threshold
σ	Cut-off Similarity Threshold

CHAPTER 01	*Introduction*

The World Wide Web (WWW) and the Internet has changed our lives in many ways and will continue to do so in the future as well. According to ILS (Internet Live Stats), the number of websites and the number of internet users is growing at an unprecedented rate. There are around 1 billion websites and number of internet users in the world is expected to cross 3 billion in 2015 (ILS 2015) (see Figure 1.1). The amount of data on the web is also growing at an alarming rate, making it very daunting to search for relevant information efficiently. Search engines have made our lives so much easier by giving us "relevant" information with almost zero latency. Search engines are programs that allow web users to search for documents/information on the World Wide Web. The information sought can be in the form of text, image or video and the search query could also be in any of these forms. Table 1.1 summarizes the history and evolution of search engines.

The tremendous increase in number of internet users and volume of data coupled with increasingly multimedia nature of the search has made it difficult for search engines to scale and adapt. Despite these challenges, the performance of search engines, in terms of efficiency, has remained impressive. However, many a times, the relevance of the search results is not up to expected levels, especially when the search queries are ambiguous and have multiple contexts.

Also, the dynamic nature of the web makes it difficult for search engines to efficiently incorporate new/updated web pages into their search algorithms. This necessitates the need for incremental approaches which can handle very large data within reasonable time & space and can accommodate the topic drift that surface with the new data being added to the previously collected data efficiently and effectively.



Source: ***Internet Live Stats*** (elaboration of data by *International Telecommunication Union (ITU)* and *United Nations Population Division)*

Figure 1.1. Global Internet Users

1.1 Search Engines – History & Evolution

Search engines have evolved in a big way during the last two decades and have made access to information on the web so simple. A brief history and evolution is summarized in Table 1.1.

Table 1.1. History and Evolution of Search Engines

Year of Launch	Search Engine	Current Status	Type	Language Support	Text/ Image/ Video Search support
1994	WebCrawler (2015)	Active, Aggregator	Meta Search Engine	English	ALL
	Go.com (2015)	Active, Yahoo Search	Web Portal	English	ALL
	Lycos (2015)	Active	Web portal and Hybrid Search Engine	Multilingual	ALL
1995	AltaVista (2015)	Inactive, redirected to Yahoo!	Web portal and Hybrid Search Engine	Multilingual	ALL
	Daum (2015)	Active	Web portal	South Korea, like Naver and Nate	ALL
	Excite (2015)	Active, Powered by Dogpile	Web Portal and Meta Search engine	English	ALL
	Yahoo! (2015)	Active, Launched as a directory	Hybrid Search Engine	Multilingual	ALL

	Dogpile (2015)	Active, Aggregator	Meta Search Engine	English	ALL
	Inktomi (2015)	Inactive, acquired by Yahoo!			
	HotBot (2015)	Active (lycos.com)	Hybrid Search Engine	English	Text
	Ask Jeeves (ASK 2015)	Active (rebranded ask.com)	Hybrid Search Engine	English	ALL
1996	Yandex (2015)	Active	Crawler Based Search Engine	Multilingual	ALL
	Google (2015)	Active	Crawler Based Search Engine	Multilingual	ALL
	Ixquick (2015)	Active also as Start page	Meta search Engine	Multilingual	ALL
1998	MSN Search (2015)	Active as Bing	Hybrid Search Engine	Multilingual	ALL
	AlltheWeb (2015)	Inactive (URL redirected to Yahoo! from 2011)	Crawler Based Search Engine	Multilingual	ALL
	GenieKnows (2015)	Active, rebranded Yellowee.com	Local business directory search engine		
	Naver (2015)	Active	Web portal	South Korea	ALL
	Teoma (2015)	Inactive, redirects to Ask.com	Crawler Based Search Engine	English	ALL
1999	Gigablast (2015)	Active	Meta Search engine interface	English	ALL
2003	Info.com (2015)	Active	Meta Search engine		ALL
	Yahoo! Search (Yahoo! 2015)	Active, Launched own web search (see Yahoo! Directory, 1995)	Hybrid Search Engine	Multilingual	ALL
2004	Sogou (2015)	Active	Search Engine	Chinese	ALL
	AOL Search (2015)	Active And Enhanced by Google	Hybrid Search Engine		ALL
2005	GoodSearch (2015)	Active And Powered by Yahoo!	Digital Coupon Database	English	ALL
	Soso (search engine) (2015)	Active	Smaller and vertical Search engine	Chinese	ALL
	Ask.com (ASK 2015)	Active	Hybrid Search Engine	English	ALL
	Live Search (2015)	Active as Bing, Launched as rebranded MSN Search	Hybrid Search Engine	Multilingual	ALL
2006	Blackle.com (2015)	Active, Google Search	Web portal	Multilingual	Text and Image

	Powerset (2015)	Inactive (redirects to Bing), Natural Language Search		Multilingual	
	Forestle (2015)	Inactive (redirects to Ecosia) supports google search as well		English	ALL
2008	DuckDuckGo (2015)	Active	Meta Search Engine	Multilingual	ALL
	Bing (2015)	Active, Launched as rebranded Live Search	Crawler Based Search Engine	Multilingual	ALL
2009	NATE (2015)	Active	Web portal and Web search engine	Korean	ALL
2010	Blekko (2015)	Active, Joined with IBM Watson Group	Hybrid Search engine	English	
2013	Halalgoogling (2015)	Active, Islamic / Halal filter Search	Meta search engine	English and Islam	Text

Search engines can be classified into four categories: crawler-based search engines, human-powered directories, hybrid search engines and Meta search engine. In crawler-based search engines (eg. Google), the entire web is crawled and the relevant pages are retrieved from the crawled pages as a response to a search query. In human-powered directories (eg. ODP 2015), experts are involved in the directory creation and updation. A search will retrieve relevant documents from the directory. Recently, crawler-type and human-powered directory results are combined for web search. Because, a hybrid search engine will favor one type of documents listing over another. For example, MSN Search generally present human-powered document from LookSmart. However, for obscure queries it presents crawler-based results. Meta search engines aggregate results from multiple search engines for a given query. Then these aggregated documents are ranked and returned to the user. DogPile (2015) and MetaCrawler (2015) are some examples of Meta search engines.

The basic architecture of the standard web search engine is shown in Figure 1.2.

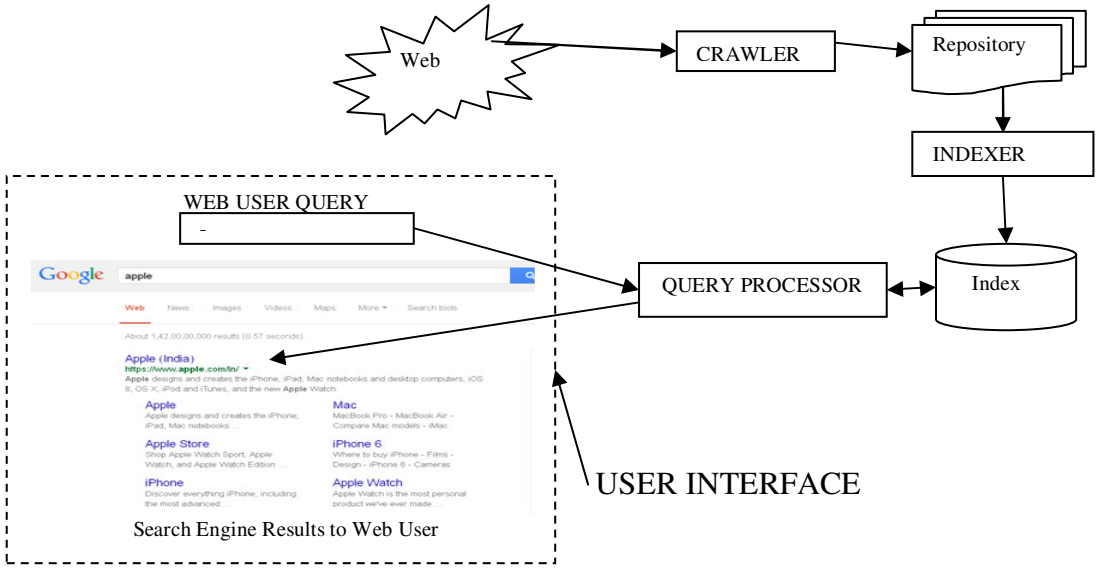


Figure 1.2. Basic architecture of a standard crawler-based web search engine

The main components of a crawler-based search engine are user interface, crawler, indexer and searcher/query processor (Brin and Page 1998).The user interface of a web search engine accepts user input (called as input user interface) and presents the results to the user (called as output user interface). The former focuses on the design of the user interface such that users' information need can be captured from the user easily and later focuses on how to present results to the user so that users' can access the required information easily. The input user interface allows users to describe their information need as query in the form of text, images and multimedia files, etc. Many researchers have focused on improving the input user interface and their work can be categorized as: work focusing on how to design search interfaces for individual sites (eg. Hearst 2009), work focusing on query reformulation system where users can refine their query to improve search performance (eg. Belkin 2000) and work focusing on specifying a search category along with query (eg. search engine such as ODP search). The output user interface mainly focuses on easy identification of required relevant documents by a user. The survey of research work on improving output user interface by search result clustering engines is given by Carpineto et al. (2009).

A Crawler locates, fetches and stores the content of the web. Crawling is difficult due to web characteristics like heavy internet traffic, fast rate of change of content of documents and dynamic page generation (Castillo 2004). Thus, a crawler should prioritize the pages to be

crawled by considering the following four different policies: Selection policy – identifies the pages to be downloaded; Revisit policy – sets the revisit frequency to check web pages for updates; Politeness policy – avoids overloading of websites by its crawler(s) by formulating guidelines, and Parallelization policy – states how to co-ordinate among the distributed web crawlers of the search engines with a view to reduce duplication and redundancy. Attaining a high page download rate is the most important performance objective for a crawler. Selection and implementation of appropriate data structures play a vital role in determining the efficiency of a sequential web crawler. Multi-threading and parallelization are the two ways to attain high scalability.

Indexer parses the downloaded content and represents it efficiently in a compact way to facilitate the process of retrieving necessary information. The main performance metrics for an indexer are deployment cycles length, compactness, and index updation speed. So far, most algorithmic improvements are concentrated on index compression. At the architectural level, the efficiency and scalability can be improved using index partitioning, pruning, and replication.

Query processor retrieves a set of documents in response to user query using the relevance between query and documents (Agichtein et al. 2006; Bendersky et al. 2011; Baeza-Yates et al. 2004a). The main challenge here is finding the best set of documents for a given query.

In this thesis, the main focus is on improving the performance of query retrieval. Web queries are usually short and imprecise (Jansen et al. 1998) due to which search engines are not able to retrieve documents of interest. Also, many users do not frame the query properly. As a consequence, traditional search engines return a large number of documents, which in turn results in increase in recall (ability to retrieve all relevant documents) and decrease in precision (ability to retrieve most relevant documents). Thus, users need to go through many documents to find the required information. These problems need to be solved to improve search result performance so that users can access required information efficiently and precisely. The major challenges in solving information retrieval problems are: dealing with poor quality and ambiguous queries, variation in users' search needs, handling enormous amount of information, discovering relevant information, finding latest information etc. Lots of methods have been recommended in the literature to address information retrieval problems. A list of these problems and possible solutions are listed in Table 1.2.

Table 1.2. Information Retrieval problems

Need of IR	***Possible solution(s)***
Discovering relevant information & handling enormous amount of information	Query Recommendation Document/Search Result/Topical Clustering Document/Information Organization Ranking etc.
Dealing with poor quality queries	Query Recommendation Search Result Clustering Document/Information Organization etc.
Variation in users' search needs	Query Recommendation Document/Search Result/Topical Clustering etc.
Finding latest information	Ranking Topic Identification Dynamic Information Organization etc.

Existing methods/techniques for search performance improvement can be broadly categorized as follows: 1. Ranking, 2. Query Recommendation, 3. Document Clustering, and 4. Information Organization. In the thesis, we have focused on the last three solutions. Query plays a vital role in improving the search engine performance as query acts as a mediator between users and the search engine. Reformulation of web queries is necessary as web queries are usually short, imprecise, and ambiguous in nature. Query reformulation can be done by Query Recommendation (QR) process which is capable of finding queries of similar intent issued by users in the past. QR is done in the following ways: Query Expansion (Carpineto and Romano 2012; Baeza-Yates & Ribeiro-Neto, 1999), Association Rules (Fonseca et al. 2003), Query-Flow Graph (Boldi et al. 2009) and Query Clustering (Beeferman and Berger 2000; Wen et al. 2001, 2002; Baeza-Yates et al. 2004, 2007; Leung et al. 2008; Goyal et al. 2013).

Query expansion is the process of reformulating the queries by adding terms and creating more complex queries. It can improve recall and therefore makes the retrieval effective. But, with the increase in the size of the corpus, precision becomes more important and query expansion may introduce noise and thereby rendering the retrieval less effective. In the second approach, similar queries are discovered by association rule mining among the queries in different sessions. This type of approach discovers query patterns frequently occurring in user's sessions that match the current query. This information is used to recommend related queries to the user. The major challenges in this approach include session segmentation and finding related queries effectively. Query-flow graph is a graph which stores information about the past queries issued or rephrased by the users according to their intention. Ambiguity is the major issue if the query-flow graph is

constructed based on keywords. Session segmentation is another major issue if the query-flow graph is constructed based on user sessions. Query clustering groups semantically related queries. Clustering can be done either by measuring the similarity of keywords (Salton and McGill 1986; Zaïane and Strilets 2002; Sahami and Heilman 2006) or by using the click-through logs (Beeferman and Berger 2000; Wen et al. 2001, 2002; Agichtein et al. 2006), or both (Leung et al. 2008, Goyal et al. 2013). Similarity measures using only users' feedback tend to cluster queries with the same or similar intention. Whereas, similarity measures using only the content tend to cluster queries with the same or similar terms. Both the approaches have drawbacks. Users' feedback may have noisy information and users' needs may not be fully captured by content alone. Recent research focuses on query clustering using click-through logs and content to incorporate the advantages of both the approaches and to overcome their disadvantages.

Document clustering methods (Ni 2004; Oikonomakou and Vazirgiannis 2005; Berkhin 2006; Aggarwal and Zhai 2012) facilitate in finding related documents effectively and efficiently. Initial document clustering methods focused on grouping the documents in the entire corpus as a means to improve the performance of the search engine. A recent application of document clustering is Search Result clustering (SRC) (Carpineto et al. 2009). SRC can help users to find necessary information very easily, to understand query related terms, to reformulate a query by disambiguating search results and to reduce the number of reformulation of queries to find required information. Further, SRC is an endeavor to cluster the documents by topics more effectively to improve the search performance.

Clustering documents by topics is referred to as topical clusters. Topical clustering methods (Riloff 1996; Schütze and Silverstein 1997; Zamir and Etzioni 1998; Vaithyanathan and Dom 1999; Yangarber et al. 2000; Blei et al. 2003; Osinski et al. 2004; Blei and Lafferty 2006; Surdeanu et al. 2006; Scaiella et al. 2012; Houlsby and Ciaramita 2013; Petkos et al. 2014a, b) can be broadly classified into two groups: document-pivot and feature-pivot methods (Aiello et al. 2013). Document-pivot methods group the related documents by finding the document similarity. Whereas, Feature-pivot methods groups the related documents by discovering the groups of similar features. The first type of approaches may cluster two documents which are related to two different topics into one cluster. This cluster may either be high level (broad) topic of the two different low level (specific) topics, or may be mixed type different topics which are not related (Petkos et al. 2014a). Pair-wise co-occurrences may produce mixtures of topics rather

than fine-grained topics in feature-pivot approach. However, one can take a larger subset of features to get fine-grained topics (Petkos et al. 2014b). SRC clustering can be further improved by integrating clicked-through log with the content.

Topic drift is the major concern in the static model. Therefore, there is a need for incremental models to incorporate recent changes.

Last but not the least, effective organization of documents can also play as catalyst in the progress of improving search performance. Web is huge, dynamic and incremental in nature. So, there is a need for organizing the information for effective and efficient information retrieval. Information can be organized either in flat structure or hierarchical structure. In both the structure, related documents are grouped and are referred as topics/categories. In flat structure organization, semantic relationships between topics/categories are not established. That is a flat organization structure merely gives us a set of topics. A hierarchical structure goes a step further by endowing with how these topics are related to each other. In hierarchy structure, topics are arranged from broader topics to more specific topics by identifying the relationships between categories/topics. This can act as knowledge base which will help in easy navigation, efficient access to relevant data, maintaining and enriching the collection etc. Hierarchical Organization of documents can be done manually or automatically. Hierarchical organization of documents can be referred as directory or topic/category hierarchy. Many directories like Yahoo! (Yahoo 2015), Open Directory Project (ODP 2015), Library of Congress Subject Headings (LCSH 2015) etc. are manually built directory. The top level of ODP (http://www.dmoz.org) is shown in Figure 1.3. These manually created directories are acting as a handy resource for the management of web content which allows us to browse and search information effectively & efficiently. However, manual building of directory needs more resources like man power, time and cost etc. It also requires users' to have the same view about the topics and their relations as that of hierarchy creator (manual label is a sign of hierarchy creator view). So, there is a need for building the hierarchy automatically which is a challenging task. The key challenges in this are: how to detect the topics hidden in the documents at a appropriate granularity?, how to evaluate the similarity between topics and assign semantic relations? and how to discover meaningful labels to the topics?. Further, it would be very useful, if users are allowed to use their own terminology to access, search/query and index documents in the hierarchy.

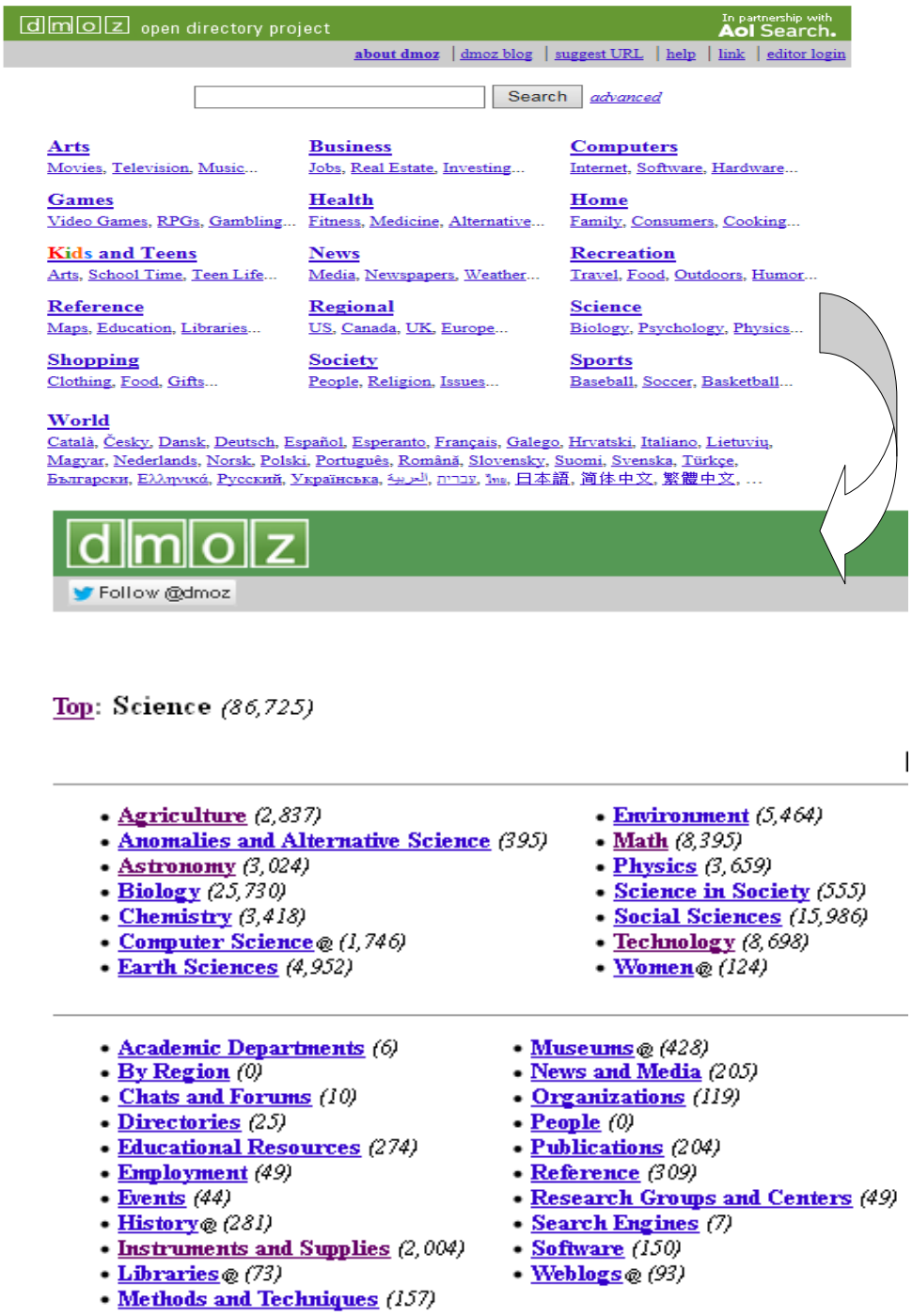


Figure 1.3. Top Level Open Directory Project (ODP) Categories and First Level Categories of ODP *Science* Category

The rest of this chapter is organized as follows. Thesis objectives are described in Section 1.2 and organization of thesis is described in Section 1.3.

1.2 Objectives

The main objective of the thesis is to improve the performance of search engines. The main techniques used are query recommendation and information organization. All the techniques developed for performance improvement of search engines are query-context/query-intention aware. The query-context/query-intention is extracted from the click-through data comprising of queries and the clicked documents' information using concept selection and clustering of queries, concepts & documents. Query recommendation is done using the clustering. Information organization system by building automatic topic hierarchy is developed. This system uses query-context aware topical clusters. These topical clusters are obtained using traditional and soft clustering techniques. Incremental clustering techniques have been developed to handle dynamic data efficiently. The overall system is presented in Figure 1.4. Details of important modules (in dashed boxes) of the system are given below:

1. **Concept Selection.**

   Documents can be represented as a set of keywords which describe/capture its context. In this work, query-context aware keywords, referred to as concepts, are extracted. As concept selection is used in text summarization, information retrieval, topic detection and tracking etc. rendering it as an important process. Thus, an online implementation of query-context aware concept selection is used for every incoming query automatically (***Concept Selection Module*** in Figure 1.4) (see Chapter 2 for more details).

2. **Identification and labeling of topic set of a query.**

   In this module, all the concepts selected for a query are grouped together to form a topic set. Automatic labeling of topics in a topic set is also done. This process is also referred to as query disambiguation. To capture all topics, query disambiguation is done every time a query is posed. A simple unsupervised model is developed. Supervised method is not found to be suitable as this require a lot of manual effort. Also, topic granularity may vary from query to query (***Topic Set Identification Module*** in Figure 1.4) (see Chapter 4 for more details).

3. **Finding similar queries.**

   This module is responsible for finding similar queries using content and click-through log. A clustering technique is used to cluster queries, the corresponding concepts and

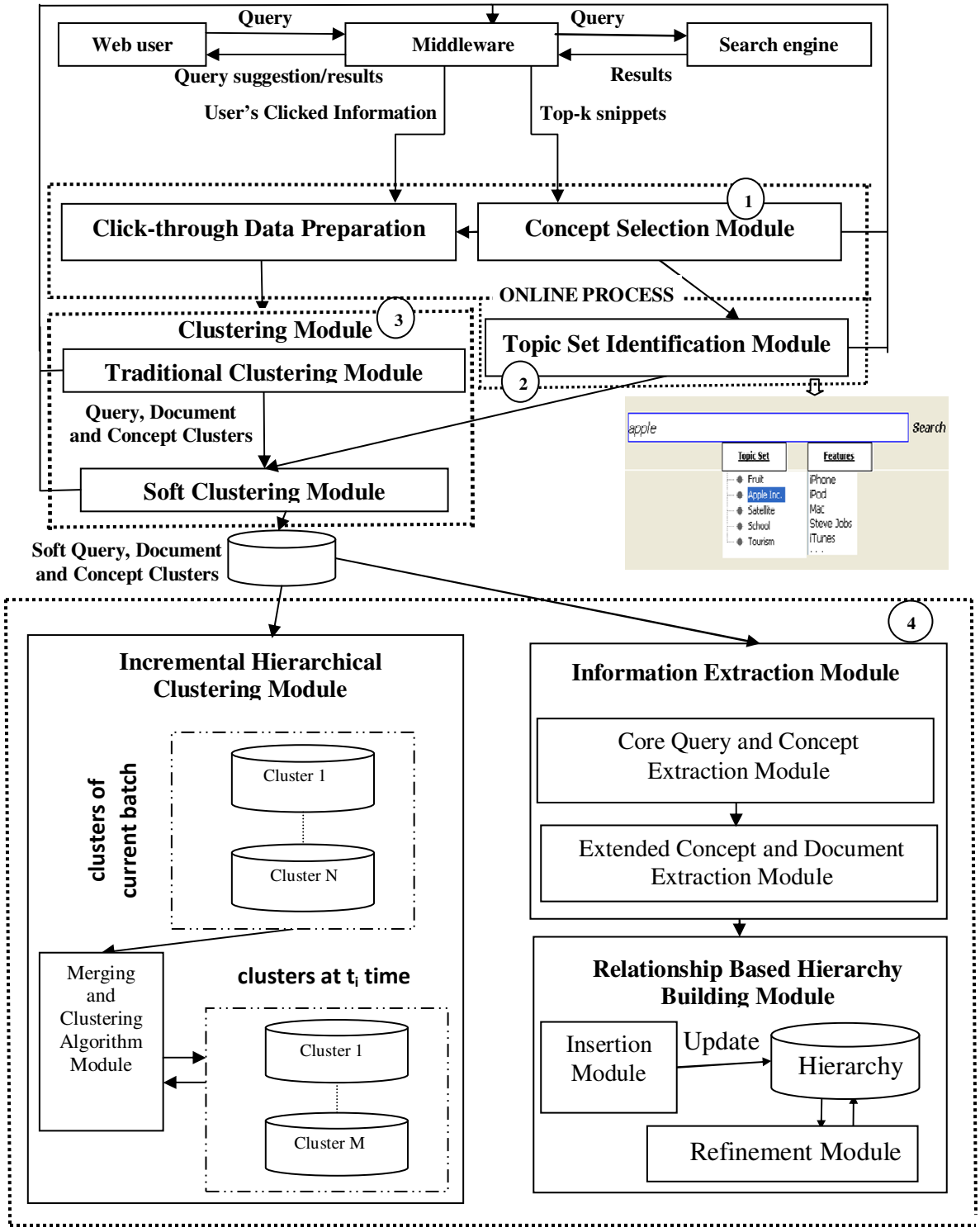


Figure 1.4. Overall System Model

documents (***Traditional Clustering Module*** in Figure 1.4) (see Chapter 3 for more details). A split-merge postprocessing algorithm is developed to obtain soft clusters to incorporate possibility of a query/document/concept belongs to more than one cluster (***Soft Clustering Module*** in Figure 1.4) (see Chapter 5 for more details).

4. **Finding similar queries.**

   This module is responsible for finding similar queries using content and click-through log. A clustering technique is used to cluster queries, the corresponding concepts and documents (***Traditional Clustering Module*** in Figure 1.4) (see Chapter 3 for more details). A split-merge postprocessing algorithm is developed to obtain soft clusters to incorporate possibility of a query/document/concept belongs to more than one cluster (***Soft Clustering Module*** in Figure 1.4) (see Chapter 5 for more details).

5. **Handling huge and dynamic data.**

   As the amount of information on the web is huge and dynamic, there is a need for organizing the information to facilitate effective and efficient information retrieval. Most popular ways to organize information are flat structures, where semantic relationships between topics are not established. Topic hierarchy structures are an alternative, which symbolize relationships between categories/topics such that these topics are arranged from general or broader topics to more specific topics. A flat structure merely gives us a set of topics. A hierarchal organization structure goes a step further by adding information about how these topics are related to each other. Hierarchy building can be either manual or automatic. The Open Directory Project (ODP) is one of the examples for manually built hierarchy/directory which allows us to browse and search information. However, building of hierarchy manually has multiple problems. This work focuses on the automation of hierarchy building by discovering topically coherent document clusters. Topical clusters (i.e. concept clusters) are formed from the click-through log at varying timestamps. These topical clusters are arranged in a hierarchy. This topic hierarchy is built and refined automatically to handle dynamic web data (***Incremental Hierarchical Clustering Module*** and ***Relationship Based Hierarchy Building Module*** in Figure 1.4) (see Chapter 6 and Chapter 7 for more details).

1.3 Organization of Thesis

Chapter 2 illustrates the process of concept/feature extraction and selection by filtering irrelevant concepts. These concepts reveal over-arching ideas of the underlying documents with respect to a query.

Chapter 3 demonstrates the models for finding similar queries and their related documents using only content or only usage information or using both.

Chapter 4 exhibits the process of identifying topic set where each topic is described by the set of descriptor concepts by revealing all meaning of the web search query.

Chapter 5 explains topical document clustering technique which groups the documents based on the topics using query based topic set by placing a query/document/concept in more than one cluster.

Chapter 6 exemplifies the incremental models which can accommodate the query/concept/topic drift efficiently with time.

Chapter 7 shows the evidence of importance of organizing the information automatically using the information obtained by the proposed methods in chapter 2-5 for efficient information retrieval.

Chapter 8 discusses about summary of achievements, limitation and future work on improving search retrieval performance.

List of references and List of publications by authors are appended to Chapter 8.

CHAPTER 02	*Concept Selection*

Document representation is an essential step in any Information Retrieval (IR) system to retrieve the documents relevant to the web user query. It is expected that the representation of documents should reflect the information intended to be conveyed by the documents. More the informative of the document representation, better will be the performance of IR using this representation. In general, documents are represented using a bag of words. But incorrectness and insufficiency of information are the major issues with this representation. So, documents can be represented as a set of keywords such that these keywords describe/capture the context of the underlying document. These keywords can act as descriptors of the documents and usually referred as features or concepts (It is to be noted that the terms *features* and *concepts* are used interchangeably). Various fields such as text summarization, information retrieval, topic detection and tracking, information visualization, automatic indexing etc. need the context of the documents. Thus, concept selection is an important process that is capable of expressing the associated document context. Manually assigning concepts is a tedious and time-consuming process. Therefore, there is a need for automatic selection of concepts from the document(s). When a query is supplied to a search engine, documents based on the common meaning of the query are returned. This causes the users to visit more number of pages to get the desired information. This problem can be resolved by identifying the concepts relevant to the query (i.e. topics of interest) from the documents to the end user. Concept selection is not only helpful for the effective query recommendation, but also helps search engine to find relevant information efficiently. This work focuses on selecting the concepts from the documents to identify ambiguity of the given query, to distinguish the underlying documents based on the meaning of the query and to help web user(s) & search engine to get the desired results. Most of the existing work focus on selecting the concepts from the whole document content. Downloading each document and selecting the concepts from it is a time consuming process. Also, documents may have text related to different topics for the detailed explanation. Concepts selected from these texts may deteriorate the document representation. Concept selection from the snippet of the document is studied in the literature (Zamir and Etzioni 1998; Osiński et al. 2004). In our work, snippets are used for concept selection instead of the whole content of documents. Generally, a

snippet includes web document's description, its title and URL. In this work, the snippet refers to the web document's description and its title (It is to be noted that the terms *snippet* and *document* are used interchangeably hereafter).

Concept selection techniques can mainly be classified into two groups: supervised techniques and unsupervised techniques. Supervised techniques train a function using training data having documents and its concepts assigned manually which is not always feasible (Witten et al. 1999; Turney 2000, 2003; Hulth 2003; Ernesto et al. 2004; Kerner et al. 2005). Unsupervised techniques select important concepts based on the scoring function(s) applied on the terms of a document. The proposed approach lies under the second technique. Unsupervised concept selection process can be further classified into two subcategories: linguistic approach and statistical approach. From the literature, it is observed that linguistic characteristics such as Part-Of-Speech (POS), syntactic structure and semantic qualities used add more values to the concepts as compared to concepts extracted using statistical approaches. Statistical approaches focus on non-linguistic features such as frequency, inverse document frequency, and position of a concept. However, the main advantages of these are: ease of use and limited computation requirements. We have chosen a statistical concept selection approach which can be applied online to incorporate query/concept drift. It is shown in (Yang and Callan 2009), that nouns and noun phrases are more capable of describing the intention of the underlying document. Therefore, in this work, nouns and noun phrases are considered as candidate concepts. Although this work uses noun phrase extraction method which is a part of linguistic approach, this is commonly used in both the approaches. Since, all the extracted candidate phrases from the snippets may not be good descriptors of the documents, so there is a need to filter unimportant concepts such that the intention/topic of the document can be captured easily and quickly. We have implemented a simple statistical method which can be applied online irrespective of the domain knowledge using only snippets.

The main contributions of this chapter are as follows:

1. Two measures, which are proposed and established, identify the relevant concepts for the given query by filtering irrelevant concepts.
2. A middleware has been developed for conducting experiments to compare proposed measures with the existing ones.

The rest of this chapter is organized as follows. Section 2.1 shows the related work and section 2.2 describes the methodology of the proposed method in detail. Experimental setup and results are described in Section 2.3. Section 2.4 states our basic findings and future work objectives.

2.1 Related Work

A concept is an important phrase which is an essential part in describing the document. Set of concepts from a single document can reveal the intention of the document which helps users' to understand the topic behind the document easily and quickly. Document concepts can be used in Information Retrieval (IR) and Natural Language Processing (NLP) tasks, such as document indexing (Frank et al. 1999), clustering (Hammouda et al. 2005), classification (Krulwich and Burkey 1996) and summarization (Berger and Mittal 2000) etc. These concepts can also be used in other applications such as thesaurus creation (Kosovac et al. 2000), metadata enrichment (Wu and Li 2008), query expansion (Song et al. 2006), automatic tagging (Medelyan et al. 2009) etc. Even though concepts are required for different applications, only a small set of documents are tagged with the concepts. But, manually assigning concepts to the documents is tedious, time-consuming and also it requires lots of resources like manpower, cost and time, etc. Therefore, automatic methods that generate concepts for a given document are preferred than manual concept selection process. According to (Witten et al. 1999), automatic concept generation can be broadly classified into two groups: 1. Concept assignment, where concepts are limited to a predefined vocabulary of terms (e.g., subject headings, thesaurus) and these concepts are not necessarily should occur in the documents. 2. Concept selection, where concepts are the most indicative phrases that present in the document itself, i.e. concepts do not depend on any vocabulary.

Concept selection process can be grouped into supervised and unsupervised approaches. In supervised approaches (Witten et al. 1999; Turney 2000, 2003; Hulth 2003; Ernesto et al. 2004; Kerner et al. 2005), the first model is constructed using the training documents along with manually assigned concepts and then the model is applied to extract the concepts for the unseen documents. In these approaches, the learning algorithm needs training data in order to construct an extraction system. However, acquiring training data with known concepts is not always feasible and human assignment is time-consuming. In addition, a model that is trained on a

specific domain may not always provide good results for the other domains. The unsupervised approaches do not require training data. These approaches first extracts the general set of concepts from the documents and then it selects the most important concepts by using some ranking strategy. Barker and Cornacchia (2000) first extracted noun phrase from the document content and then considered top ranked noun phrases as concepts. In this, noun phrases are ranked using simple heuristics based on their length, frequency, and the frequency of their head noun. In (Bracewell et al. 2005; Wan and Xiao 2008; Liu et al. 2009), extracted noun phrases from a document, and then clustered the terms which are semantically covered terms. The clusters are ranked based on term and noun phrase frequencies. Finally, top-n ranked clusters are selected as concepts for the document. Another kind of unsupervised approach which employs graph-based ranking methods is proposed in (Litvak and Last 2008). In such methods, a document is represented as a term graph based on term relatedness, and then a graph-based ranking model algorithm (similar to the PageRank algorithm (Brin and Page 1998)) is applied to assign scores to each term. Term relatedness is approximated in between terms that co-occur each other within a pre-defined window size. All of these above described methods have been focused on longer documents.

Finding important terms or concepts related to the given query can be classified into three categories. 1) Lexical approaches (language-specific) 2) Statistical approaches (corpus-specific) and 3) Statistical approaches (query-specific). *Lexical approaches* which influence global language properties, such as synonyms and other linguistic word relationships (e.g., Hypernyms). These approaches are typically based on dictionaries or other similar knowledge representation sources such as WordNet (Voorhees 1994). *Statistical approaches (corpus-specific)* are data-driven and attempt to discover significant word relationships based on term co-occurrence analysis and to select features. These relationships are more general and may not have linguistic interpretation. Early corpus analysis methods grouped words together based on their co-occurrence patterns within documents (Qiu and Frei 1993). These methods include term clustering (Jones 1971), which group related terms into clusters and Latent Semantic Indexing (Deerwester et al. 1990), which forms hidden orthogonal dimensions based on term-document co-occurrence. Later approaches attempt to reduce topic drift by looking for frequently co-occurring patterns only within the same context, as opposed to the entire document, where the context can be the same paragraph, sentence, or simply a neighborhood of n words (Jing and

Croft 1994; Xu and Croft 1996; Gauch and Wang 1997; Schütze and Pedersen 1997; Carmel et al. 2002). In contrast to global statistical approaches that consider the distribution and co-occurrence of words within an entire corpus, *Statistical approaches (query-specific)* use only a subset of the documents to identify significant co-occurrence patterns. This subset is typically a set of documents explicitly provided or tagged by the user as being relevant to the query. In relevance feedback systems, for example, the system modifies the query based on users' relevance judgments of the retrieved documents (Rocchio 1971). To eliminate or reduce the need for user feedback, some systems simply assume that the top N retrieved documents are relevant, where N is determined empirically and is typically between 20 and 100. This is motivated by the assumption that the top results are more relevant than that of a random subset. This approach is called pseudo-relevance feedback, or blind feedback (Xu and Croft 1996). Moreover, Glover et al. (2002) shown that the full-text of a Web page is not good enough for representing the Web pages.

This research proposes a simple statistical query-specific method which can be applied online and is capable of adapting changes occurring with time irrespective of the domain knowledge. We have used only document snippets instead of whole document content. Differences between categorization of short and long documents are identified by few researchers (Timonen et al. 2011a, 2011b; Timonen 2012). The same is present in concept extraction as well. The most obvious difference is the number of times a concept occurs in a document and in a snippet. Generally, concept frequency is one and that became a challenge.

Boldi et al. (2009) and Zahera et al. (2010) used the support measure, which is a well known measure in finding the frequent item sets in data mining, for measuring query and concept relevance. The basic idea behind using support measure is: if a keyword or a phrase appears frequently in the web-snippets of a query, it may represent an important concept. TFIDF is another well known and common measure to describe the text feature on vector space information retrieval paradigm introduced by Jones (1972). This research proposes and applies TFIDF based query and concept relevance measure and compares the results with the proposed two weight functions.

2.2 Methodology

The proposed approach is to extract the important concepts from the search results of the given query. Given a query as input the proposed approach works in two steps:

1. *Concept Extraction:* Extracts concepts from the document snippets returned by the search engine as a response to a given query.
2. *Concept Filtration:* Identifies the relevant concepts for the given query. This research proposes two weight functions to measure the relevance between query and concepts. Concepts with greater weights (greater than the threshold δ) are considered.

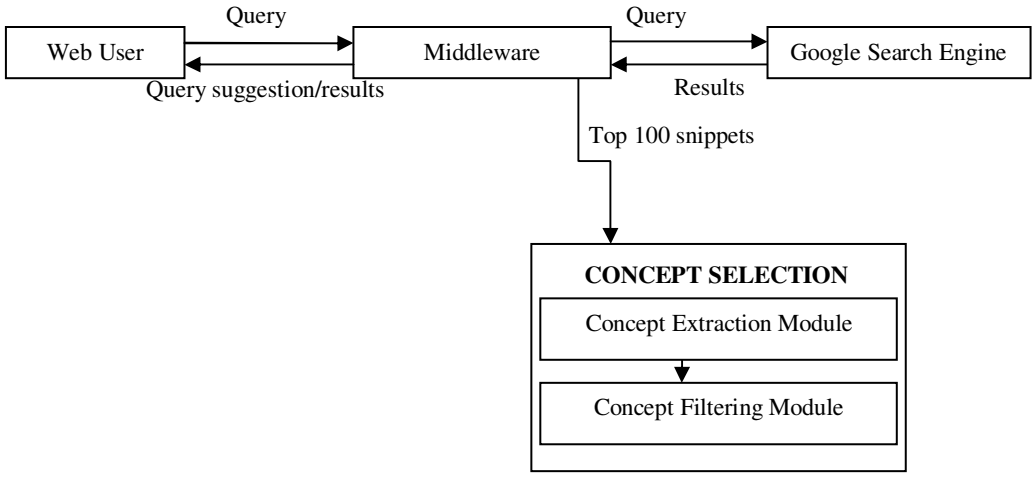


Figure 2.1. System Model

2.2.1 Concept Selection Module

The purpose of the concept selection is to build an index of the terms that occur in the document collection. Concept selection is performed in two steps. The first step involves extraction of nouns and noun phrases which are most predictive of the content of the documents. We have used MontiLinguatool (Liu 2004) to extract nouns and noun phrases. This tool is a free, end-to-end, lightweight, portable across platforms, open source, integral part of ConceptNet and natural language processing toolkit. It is implemented in Python and also available as a compiled Java library. It comprises 6 components: MontyTokenizer, MontyTagger, MontyREChunker, MontyExtractor, MontyLemmatiser, MontyNLGenerator.

All the nouns and noun phrases extracted from the search results in the first step might not be relevant to the intent of the given query. This necessitates filtering them. Concept filtration is done by first removing stop words and then followed by filtering the concepts by estimating query and concept relevance. Based on the relevance, we further eliminate concepts (weight $<\delta$) in the second step. For this we are using the proposed weight functions described next.

2.2.2 Relevance between Query and Concepts

This research proposes two weight functions, W1 and W2, to estimate the query and concept relevance. The function W1 is using the content of the retrieved documents alone, whereas the function W2 is making use of both content and the links associated with the documents. In this work, top N (=100) retrieved document's snippets of the query are used as a collection.

1. Weight Function 1 (W1)

In order to estimate the relevance between query-intent and concepts in the documents returned by the search engine, we have identified the following important factors:

1. Concept occurrence in the top 100 snippets
2. Concept occurrence within a snippet (usually measured by Term-Frequency (TF))
3. Concepts should also be good descriptors of the documents. Usually, if the terms appear many times in a document, then they might be good descriptors of the document. But many terms may occur frequently in many documents without having any relation to the document theme. Terms like "news", "articles" etc. are examples. These are frequently occurring terms in news documents and the general importance of these terms is high. The general importance of terms should be less for becoming good descriptors of the documents (usually measured by Inverse Document Frequency (IDF)).
4. The collective importance of the concept, with respect to the documents containing it.
5. Balance factor which maintains balance between collective TF and IDF

The following is the formula to calculate the weight W1 (q, c_i) of concept c_i with respect to query q:

$$W1(q, c_i) = \left(\sum_{d_j \text{Contains } c_i} \left(\frac{n_{ij}}{\sum_{k=1}^{z} td_{kj}} \log\left(N/m_i\right) \right) \right) \frac{m_i}{N} \quad (2.1)$$

where n_{ij} is a number that the number of times the concept c_i occurs in the document d_j, z is the total number of distinct terms in the document d_j. $\sum_{k=1}^{z} td_{kj}$ is the total number of terms in the document d_j. N is the total number of documents considered. m_i is the total number of documents in which the concept c_i appears. The weight is divided by the number of documents considered i.e. N. The concepts, for which the weights are greater than the threshold δ, are considered to be relevant to the query and thus potential concepts to be considered.

2. Weight Function 2 (W2)

This function exploits both the content and the in-and-out links of the documents to find the relevance between the concept and query. In W1, we have estimated the relevance between the concepts and the query based on the content of the document. It is also good to consider the content of the neighboring pages to estimate the importance of the concept with respect to the document. For example the concept “system” is occurring frequently in a document obtained for a query “apple”. The concept "system" might be considered as more related to the query than necessary if it is frequent. This effect can be balanced by considering the content of the neighboring pages. This is expected that these types of concepts would not also be highly frequent in the neighboring pages. For this, we have added an additional factor to the W1 that is the popularity score of the document based on the popularity score of the neighboring pages. Since, all the neighboring pages may not be related to the same query concept, we have considered neighboring pages which contain the concept.

The following is the formula to calculate the weight W2 (q, c_i) of concept c_i with respect to query q:

$$W2(q, c_i) = \left(\sum_{\forall d_j \text{ Contains } c_i} \left(\frac{n_{ij}}{\sum_{k=1}^{z} td_{kj}} \log\left(N/m_i\right) + PS(j) \right) \right) \frac{m_i}{N} \quad (2.2)$$

where, $PS(j)$ is the popularity score of the document d_j containing the concept c_i and is described by the following:

$$PS(j) = \frac{d}{N} + (1-d)\sum_{z=1}^{n} \frac{PS(W_z)}{C(W_z)} \quad (2.3)$$

where, $W_1, W_2, \ldots W_n$ are the documents that point to the document d_j which contains the concept c_i. $C(W_z)$ is the number of outgoing links from the page W_z that contains the concept c_i. d is a damping factor whose default value is set to 0.15.

The concepts, for which the weights are greater than the threshold δ, are considered to be relevant to the query.

W2 clearly separates the important relevant concepts from the unimportant concepts as compared to the W1. However, W2 uses the documents to explore the in/out links to the documents. Therefore, W1 can be treated as a light weight function in comparison to W2.

The performance of both the weight functions W1 and W2 is compared with that of support S and TFIDF.

Support (S):

Support is a well known measure in finding the frequent item sets in data mining as mentioned by Boldi et al. (2009), and Zahera et al. (2010). It is based on the fact that if a keyword or a phrase appears frequently in the web-snippets of a query, it may represent an important concept. The following formula for support S as a relevance measure is used by Leung et al. (2008) to measure the relevance between the query q and the concept c_i:

$$S(q, c_i) = \left(\frac{m_i}{N}\right) t_i \quad (2.4)$$

*w*here, t_i is the number of terms in the concept c_i. Other symbols are having the same meaning as described earlier.

TFIDF:

TFIDF is another well-known and common measure to describe the text feature on vector space information retrieval paradigm introduced by Jones (1972). TFIDF measure relevance between a document d_j and a concept c_i as follows:

$$TFIDF(d_j, c_i) = \frac{n_{ij}}{\sum_{k=1}^{z} td_{kj}} \log\left(N/m_i\right) \quad (2.5)$$

where, n_{ij} is a number that the number of times the concept c_i occurs in the document d_j, z is the total number of distinct terms in the document d_j. $\sum_{k=1}^{z} td_{kj}$ is the total number of terms in the document d_j. N is the total number of documents considered. m_i is the total number of documents in which the concept c_i appears. The weight is divided by the number of documents considered i.e. N.

TFIDF increases the weight of the concept c_i if the concept is more frequent in document d_j but rarely occur in other documents i.e. collections. In the literature, TFIDF measure is not used for measuring the relevance between query and concept.

We have measured relevance of the concept c_i with the query q using TFIDF as follows:

$$\mathrm{TFIDF}(q, c_i) = \left(\sum_{d_j \text{contains } c_i} \left(\frac{n_{ij}}{\sum_{k=1}^{z} td_{kj}} \log\left(N/m_i\right) \right) \right) \tag{2.6}$$

The concepts, for which the weights are greater than the threshold δ, are considered to be relevant to the query and thus potential concepts to be considered.

2.3 Experimental Setup and Results

In this section, we have presented the results obtained by the proposed weight functions. Experimental setup is described in section 2.3.1. Performance evaluation of the two proposed weight functions is given in section 2.3.2. The performance of the proposed weight functions is compared with that of S and TFIDF. The overall precision of the proposed weight functions is also given.

2.3.1 Experimental Setup

We have developed a middleware to collect the data automatically from the Google Search engine for the evaluation of the proposed models. This middleware uses Gson - open source Java API (GSON 2014) for the automatic extraction of web snippets. A query is submitted to the Google search engine through middleware without any modification and search results are processed (see Figure 2.1).

We have involved 20 users to collect data through the middleware. To avoid any bias, we have given 18 different categories for the search. Table 2.1 shows the categories which are considered. Each user processed maximum of five queries from the given categories.

Table 2.1. Topical categories of the queries

Cat. No.	Cat Description	Cat No.	Cat. Description	Cat. No.	Cat. Description
1.	Fruits	7.	Digital Cameras	13.	News
2.	Computer software	8.	Sports	14.	Biology
3.	Computer hardware	9.	World wide web	15.	Career
4.	Mobile Phone	10.	Business	16.	Shopping
5.	Scientist	11.	Travel	17.	Kids
6.	Animal	12.	Conference	18.	Research

Most users would examine only the top 10 results displayed by the search engine. However, we have considered top N (N=100) retrieved documents for concept selection so that all the concepts related to the query can be discovered. The concept extraction process is done using the tool called MontiLingua (Liu 2004). Extracted concepts are given to the concept filtering module to select the important concepts by estimating their importance using the weight function W1 and W2.

To evaluate the performance of the proposed functions, we have considered different type of queries such as *specific* (one meaning), *ambiguous* (multiple meaning), and the *general* queries. Table 2.2 gives the statistics of the type of the queries considered in the experiment.

Table 2.2. Sample Queries and number of queries in each category

Query Type	Sample Queries	No. of Unique Queries
General	career counselling, code, tax, world wide web, guide etc.	25
Specific	apple fruit, jaguar animal, guitar, Income tax, sony digital camera etc.	30
Ambiguous	Blackberry, apple, jaguar, cell, bow, home etc.	16

The threshold δ for concept extraction is set to 0.002. We have chosen small threshold value, so that no concept related to the query is missed. Table 2.3 presents the statistics of the data collected for concept selection experiments.

Table 2.3. Data statistics for concept selection experiments

Number of users	20
Max No. of queries assigned to each user	5
Number of test queries	85
Number of unique queries	71
Max No. of retrieved URLs for a query	100
Max No. of extracted concepts for a query	718
Min No. of extracted concepts for a query	309
Number of URLs retrieved	7100
Total Number of concepts retrieved	48723

2.3.2 Experimental Results

In this section, we present the performance analysis of the proposed weight functions. As discussed earlier (see Section 2.2.2), we have used four weight measures to find the relevance between queries and concepts. Now, we present the comparison of the performance of the proposed weights W1 and W2 with two other weight support (S) and TFIDF. All four weights are normalized in the range [0, 1] for comparison. Weight precisions (P) are calculated individually for every query from all three categories. For this all concepts are arranged by weight values. Precision is obtained by calculating the ratio (number of related concepts/number of total concepts (m)) for top m concepts. The concepts are marked related or unrelated manually by the cumulative feedback of 12 users, who were involved in the experiment.

For the first set of experiments, we have compared the precision of all four weights calculated individually for every query from all three categories, i.e. “General”, “Specific” and “Ambiguous”. The precision plots for few sample queries from each category are presented (see Figures 2.2-2.4). Precision (Y-Axis) is plotted against different values of m (top m concepts) (X-

Axis). In these figures, *NC* represents the total number of concepts extracted from top-100 snippets of the query and *NUC* represents the total number of unrelated concepts for the query.

Firstly, we discuss the results of *ambiguous* queries. Few of them are presented in the Figure 2.2. The performances of the two measures W1and W2 are much better than the measure support S and TFIDF for all queries in this category. The weight TFIDF is also performing better than support except for one query which is "Blackberry". In case of a query that is ambiguous, (i.e. having more than one well known meaning) W2 is the best among all e.g. query "Apple", "Blackberry" etc. This implies that W2 is capable of identifying the relevant concepts if a query is having multiple meanings. In some cases, where more than 70% of the documents are related to the most popular meaning of the ambiguous query, W1 shows better performance than the W2 e.g. "Jaguar" etc. otherwise it performs equally well as W2.

In case of *specific* type of queries, support is behaving as badly as in the *ambiguous* type. In most of the cases the support curve is increasing which shows that it is not capable of capturing the relevance between query and concepts. It is found that the values of precision for both W1 and W2 are almost same (e.g. query "World Wide Web Conference", "Soccer" etc.). Moreover, W1 and W2 attain highest values of precision among all measure for all queries of specific category. The precision of TFIDF is sometimes close to W1 and W2 e.g. query "World Wide Web Conference", "Tennis" etc., sometimes it is lowest e.g. query "Apple Fruit", "Sony Digital Camera" etc., and sometimes between support and W1 or W2 e.g. query "Soccer", "Income Tax" etc. (see Figure 2.3).

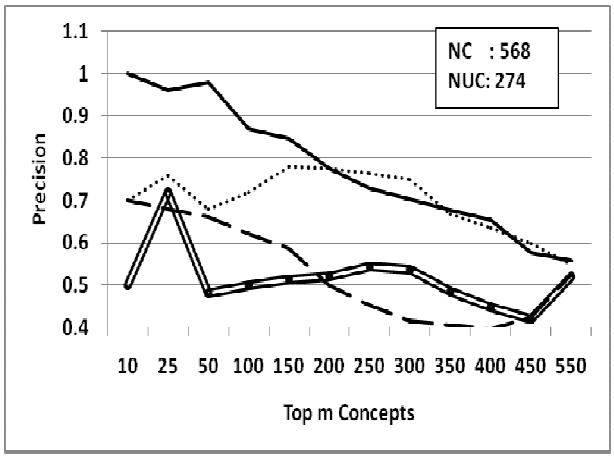
Precision for all measures is calculated and plotted for *the general category* of the queries also (see Figure 2.4). The precision for W1, W2 and TFIDF are quite close in some cases and the same behaviour is observed for support. In some cases, W2 is performing exceptionally well (attains high values close to 1) for the queries like "Guide", "News" etc. This is because the concepts related to these queries are popular and so as the linked documents containing these concepts.

The second set of experiments is conducted to estimate the performance of the measures for each category of the queries. The Mean Average Precision (MAP) for the measures have been calculated for query categories "General", "Specific" and "Ambiguous" and shown (see Figure 2.5). The measure support not only acquires the less precision (for all categories of queries) as compared to other measures, but also its graph is increasing for *general* and *ambiguous* queries.

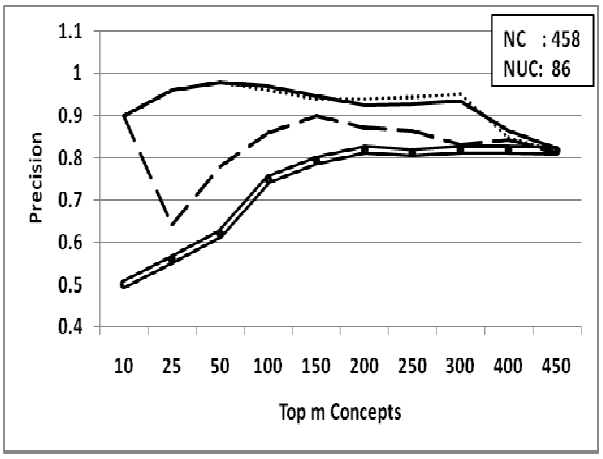
This indicates that the measure is not able to capture the high relevance of concepts with queries for its higher values.

The precision curves of the other three weights are almost same, showing the capability of identifying the relevant concepts for "General" category of the queries (see Figure 2.5a). However, the values of W2 are higher than the values of W1 and the values of W1 are higher than the values of TFIDF. The weight TFIDF is low as compared to W1 and W2 for the other two categories "Specific" and "Ambiguous". This indicates that if the query is *specific* or *ambiguous* the two weights W1 and W2 works better to identify relevant concepts. Both the measures W1 and W2 are performing distinguishably well in the case of the other two categories "Specific" and "General". It can be seen that W1 is performing slightly better than W2 in the "Specific" category, whereas W2 is better for "Ambiguous" category.

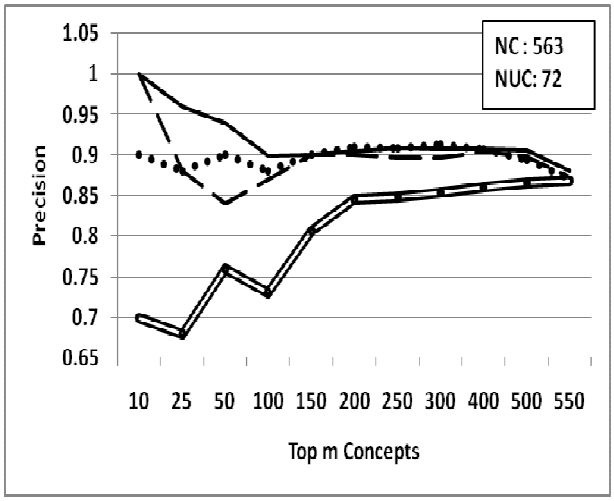
Lastly, the experiments are performed for the overall behaviour of the measures. The MAP of all weights for all the unique queries are calculated and presented (see Figure 2.6). It is clear from the figure that the MAP for W2 is highest among all and attains value around 0.9 till top 300 concepts. However, the MAP of W1 is slightly lower than that of W2 but having almost the same figure as W2. This shows that W2 and W1 weights are capable of identifying the relevant concepts. Moreover, it is observed that the concepts, identified by both W1 and W2, among top 150 are more relevant to the queries than the concepts from 150 to 300 approximately. Furthermore, the measure TFIDF is also showing the same behaviour that is capable of finding the relevance but with the lower precision. The measure support is not at all comparable to the above three measures because 1) Precision is quite low. 2) Curve is increasing rather than decreasing with m.



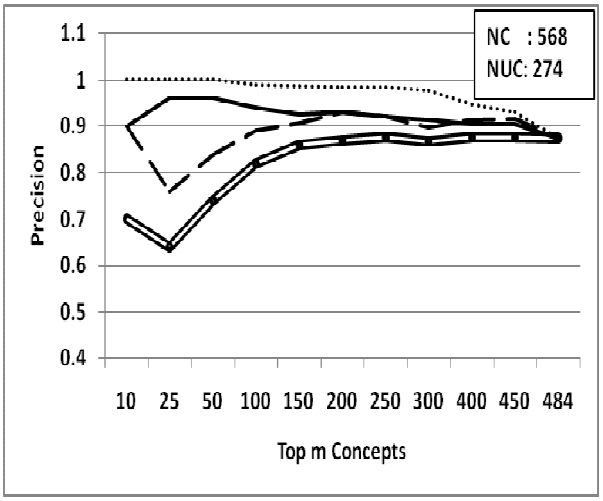
(a) Apple



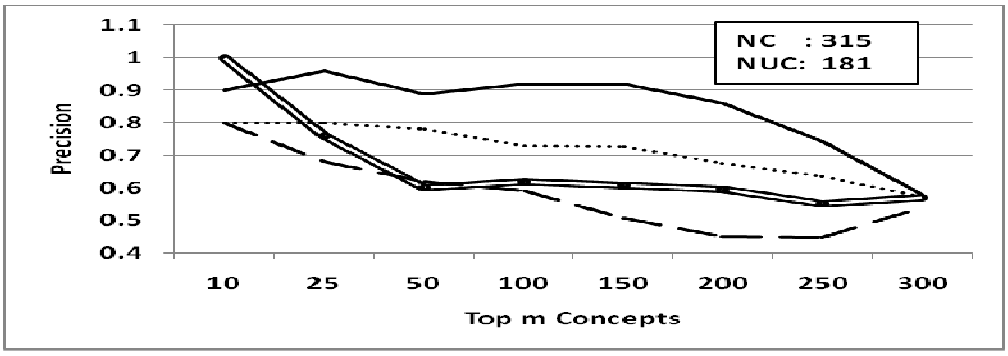
(b) Bow



(c) Cell

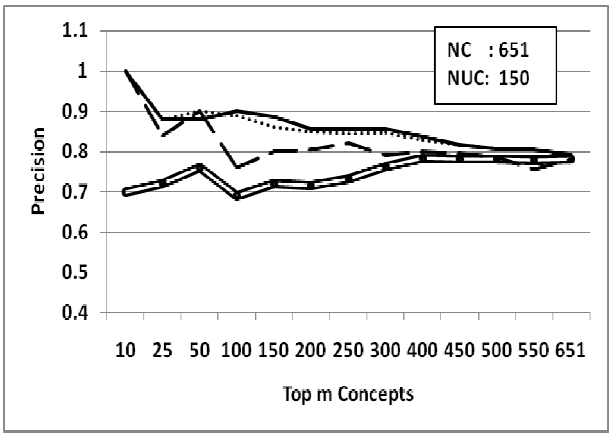


(d) Jaguar

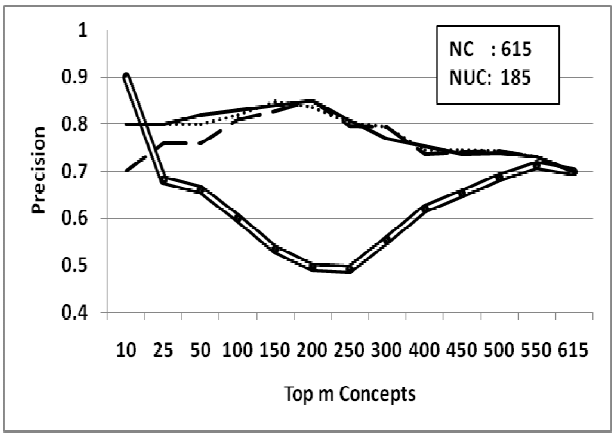


(e) Blackberry

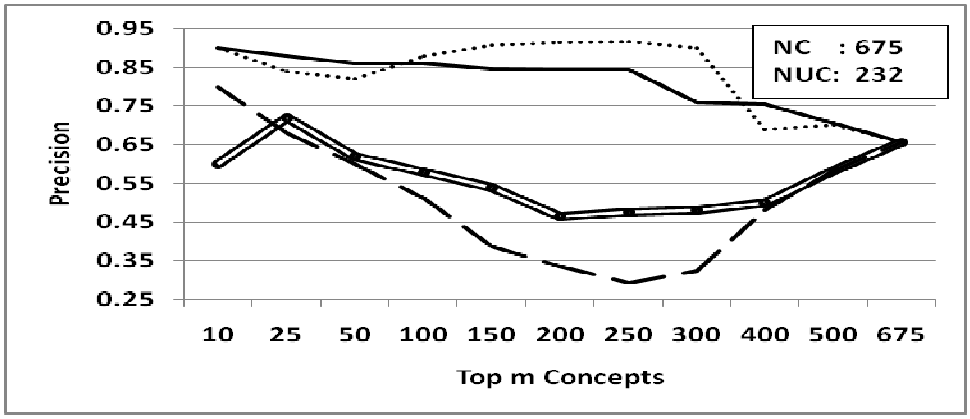
Figure 2.2. Precision of W1, W2, S and TFIDF for different queries of type "Ambiguous" (=== S, -------- TFIDF……. W1, ——W2)



(a) Soccer

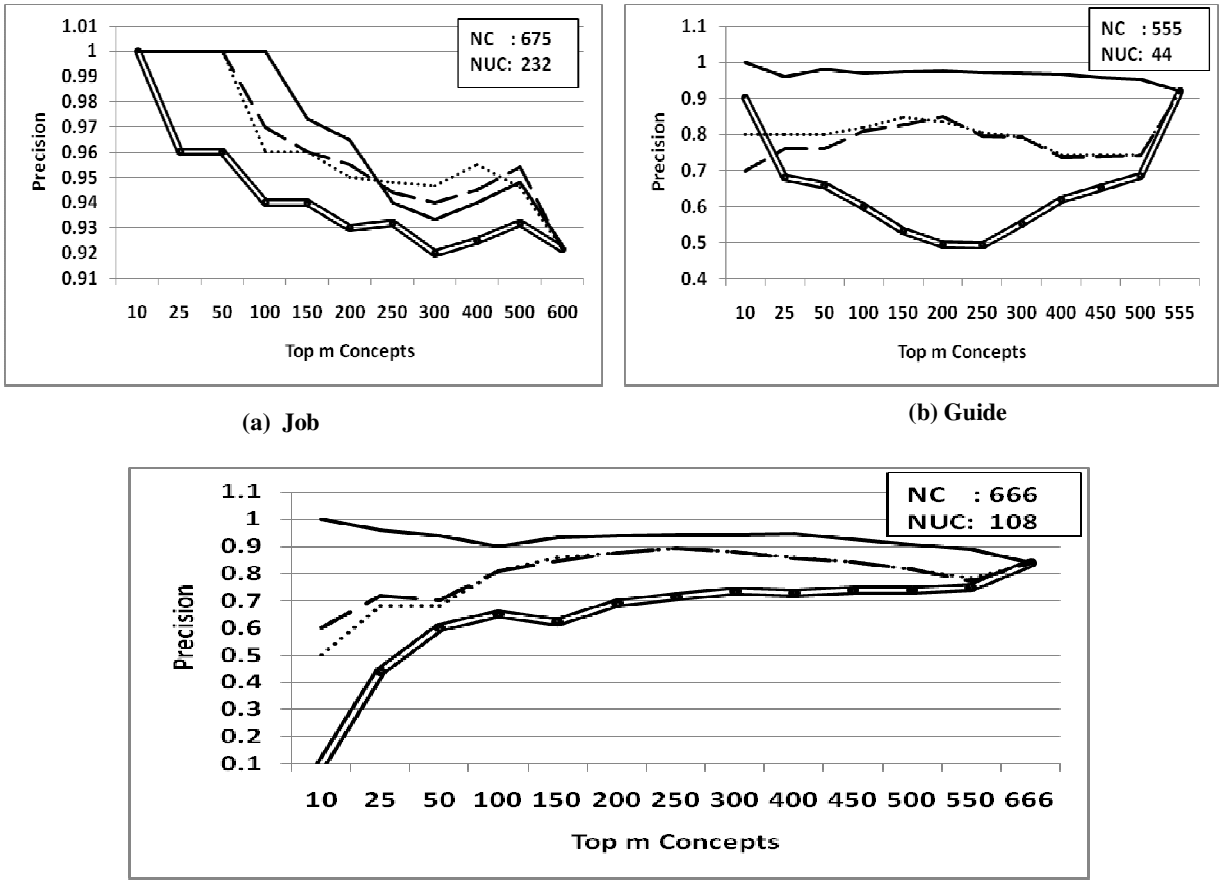


(b) WWW Conference



(c) Apple Fruit

Figure 2.3. Precision of W1, W2, Support and TFIDF for different queries of type "Specific"(=== S, -------- TFIDF……. W1, ——W2)



(a) Job

(b) Guide

(c) News

Figure 2.4. Precision of W1, W2, Support and TFIDF for different queries of type "General"
(=== S, -------- TFIDF……. W1, ——W2)

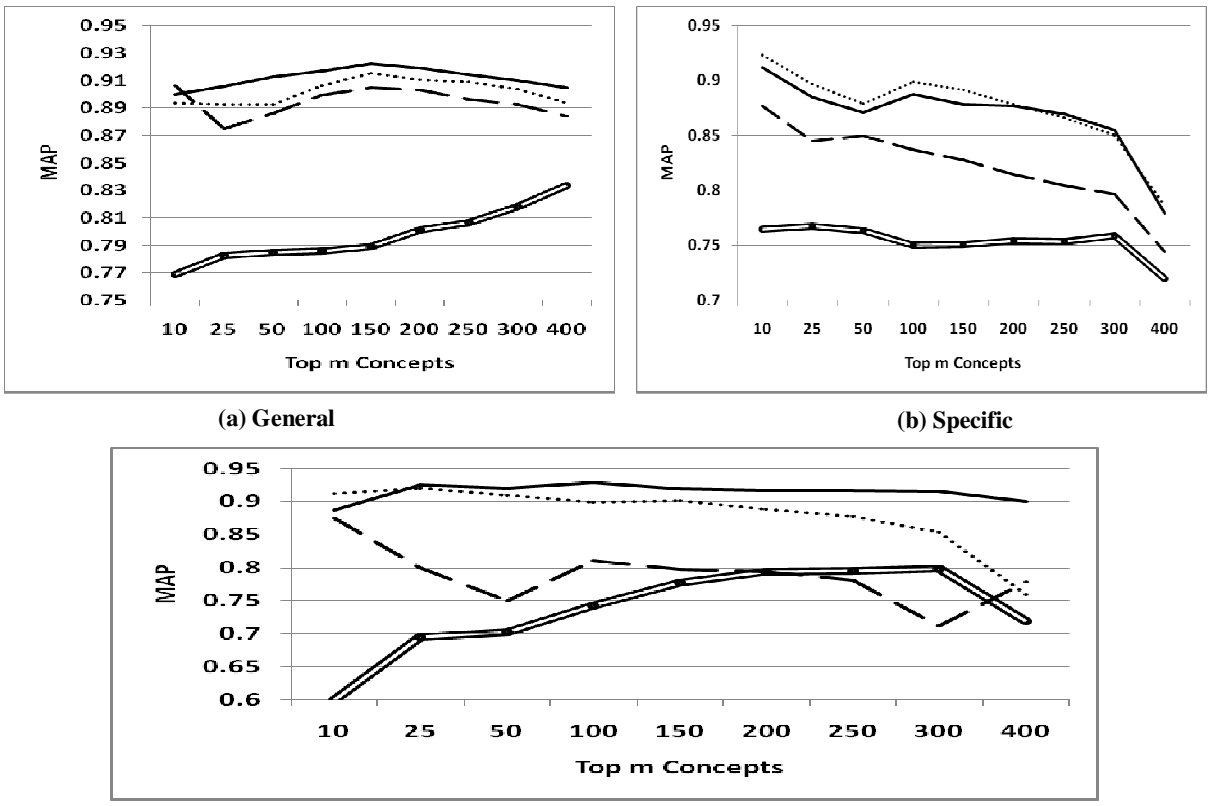


Figure 2.5. Mean Average Precision of W1, W2, S and TFIDF for different query types (=== S, -------- TFIDF……. W1, ——W2)

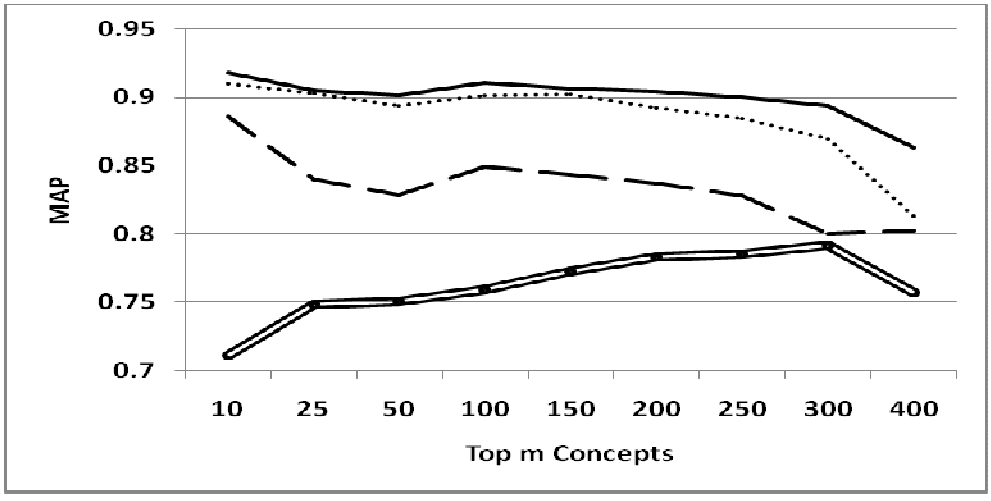


Figure 2.6. Mean Average Precision of measures W1, W2, S and TFIDF over all considered queries (=== S, -------- TFIDF……. W1, ——W2)

2.4 Conclusions

In this chapter, two weight functions are proposed for eliminating the irrelevant concepts which are extracted by the concept extraction process. The performance of the weight functions is compared and tested on many queries of different types against the two measures: support and TFIDF. The performances of the two measures W1 and W2 are superior for all queries of different types. The overall performance of the weight W2 is better than the weight W1. Whereas, W1 is a light weight process. The approach can be extended to cluster the documents returned by the search engine, i.e. Search Result Clustering and to cluster search engine queries.

CHAPTER 03	*Query Clustering*

The increasing growth of the web continually increases the information piled up by a search engine, thereby making it more and more difficult to find relevant information. This difficulty is further increased when queries submitted by the web users are short and ambiguous. For a query, traditional keyword similarity approaches do not incorporate user's context into the search process. Therefore, in spite of having different information needs, different users receive the same set of results for the same queries.

Query plays an important role in finding the relevance between query and documents. But, mostly queries specified by the users are imprecise, short and ambiguous. It has been shown that the average length of the query is less than three terms (Jansen et al. 1998). Thus, the terms in the queries are not adequate to represent the information needs of the users. Efficiency of search can be improved by reformulating the query appropriately. The query formulation can be done in two ways - by query recommendation and by effectively exploiting the information lying in the query log/click-through log.

Query recommendation has evolved as a powerful method to improve search by allowing users to formulate queries in a more effective manner. Many search engines offer query suggestions such as Yahoo's "Also Try" and Google's "Searches related to". These suggestions are semantically related but mostly start with the terms which users use in their queries. As a result, the ambiguity which was present in the original query may still remain. Moreover, the search engines provide same suggestions to all users without considering the specific needs of the individual user.

A search engine provides a long list of URLs along with the corresponding snippets as a response to a user query. Users then selectively click some of the URLs based on the snippet descriptions. Queries submitted and their respective clicked information are major parts of the query log which is also known as click-through log. Click-through log has emerged as a repository of feedback of the user's behavior. User's needs and intentions are implicitly lying in the click-through log which can be mined and used appropriately to improve search engine's efficiency. Beeferman and Berger (2000) and Agichtein et al. (2006) worked to mine the latent

knowledge hidden in the click-through logs. Though, click-through log may have some noise in it, still they convey important information on query and document relation embedded in them.

Query recommendation techniques can fall into one of the following four categories: 1) recommendation by query expansion (Carpineto and Romano 2012) 2) recommendation by association rules (Fonseca et al. 2003) 3) recommendation by query-flow graph (Boldi et al. 2009) and 4) recommendation by query clustering (Beeferman and Berger 2000; Wen et al. 2001, 2002; Baeza-Yates et al. 2004, 2007; Leung et al. 2008). Query expansion is the process of reformulating the queries by adding terms and creating more complex queries. It can improve recall and therefore make the retrieval effective. But with the increase in the size of the corpus, precision becomes more important and query expansion may introduce noise and thereby rendering the retrieval less effective. In the second approach, similar queries are discovered by association rule mining over the queries in different sessions. Session segmentation is the major issue with this approach. Query-flow graph is a graph which stores the information about the past queries issued or rephrased by the users according to their intentions. Ambiguity and session segmentation are major issues when the query - flow graph is constructed based on keywords and user-sessions respectively. Query clustering groups semantically related queries. Clustering is done either by measuring the keywords similarity or by using the click-through logs, or both. Query clustering using keywords is not successful because of short and imprecise queries. Recent research focuses on query clustering using click-through logs. Usually, query clusters are formed either by using query-document pair or term-document pair or term-term pair (term refers to query term). The analogy for components of query clustering is as follows: consider the document as “body”, query/query term as “spirit” and user’s intention about query as “soul”. Existing query clustering methods have proven to give better results using body and spirit. But they haven’t considered the soul. Quality search cannot be achieved without using the soul. A content-ignorant method is proposed in (Leung et al. 2008), for finding similar queries by considering query-concept pair but they haven’t taken documents into consideration. The following are the issues that are observed in the existing work:

1. The method given in (Salton and McGill 1986) considers only query terms for finding similar queries, is unable to distinguish two queries of different meaning with common terms. For example, “Apple computers” and “Apple Fruit”.

2. The method given in (Beeferman and Berger 2000; Wen et al. 2001, 2002) clusters queries based on the common clicks on the same pages. But, clicking on the same page even for the similar queries is infrequent because the search engine gives different results in different ranking order for different queries at different times.
3. The method given in (Leung et al. 2008) considers query and concept for query clustering, missing the following information:
   - Number of documents involved in relating the queries to the concepts.
   - Concepts which relate to a query are contained in the same or different documents.
   - If single concept is related to two or more queries, it is not known whether the concepts relating to a query are contained in the same or different sets of documents.
4. Most of the earlier methods did not measure the strength/weight of the relations among query terms themselves, between query and documents or between query and concepts based on the content. Many times, noise is introduced in the click-through log, if a document is clicked mistakenly by a user. So, considering the content is important in finding the strength of similarity between two queries in eliminating noise and in identifying user's intention (Refer to section 3.2 for details).

In this work, all three: spirit, body and soul, i.e., queries, documents, and concepts respectively are used. Concepts are important terms or phrases extracted from the search results and can be good descriptors of the intention of the queries. The proposed system has three phases. In the first phase of feature/concept selection: features/concepts are extracted and filtered by measuring the relevance with the given query (refer Chapter 2). During the second phase, click-through log is collected to incorporate user's feedback. Feature extraction and click-through log collection are online. Whereas, feature/concept filtering is done offline. In the third phase, QDC-Clustering is used to cluster conceptually related queries using a QDC tripartite graph structure. This is done in an offline mode. Finally, query recommendation is made for search refinement (It is to be noted that the terms *features* and *concepts* are used interchangeably henceforward).

The proposed system handles all the issues listed above by efficient feature selection and QDC-Clustering. The main contributions of the proposed system are as follows:

1. QDC-Clustering, a new tripartite agglomerative clustering algorithm, wherein the documents are innovatively used to decouple queries and features (QDC Tripartite Graph

Structure – refer to section 3.2.1). This structure is capable of capturing all the information present in the query log, unlike any other existing method mentioned above. In addition, different similarity measures are proposed to find similarities according to the subjective and objective conditions.

2. Four models for query recommendation are proposed. Non-personalized content-ignorant model, non-personalized content-aware model which strengthen the link between query and concepts based on the corresponding clicked documents. Another two models are personalized versions of content-ignorant and content-aware models. They incorporate each user's information individually.

The rest of the chapter is organized as follows. Related work is presented in Section 3.1. Section 3.2 describes the methodology of the proposed system in detail. Experimental results are described in Section 3.3. Section 3.4 states our basic findings and future work directions.

3.1 Related Work

Query clustering is the one of the major techniques to improve the web search experience. Researchers have approached query clustering in many different ways. One of the basic approaches for query clustering is by comparing query terms with document terms using various similarity functions such as cosine-similarity, Jaccard similarity, Dice-similarity etc. (Salton and McGill 1986). This approach is not appropriate because of the size of the queries (Silverstein et al. 1998; Wen et al. 2002). Moreover, the same term may refer to different semantic meaning and different terms may refer to the same semantic meaning. Thus, query terms alone cannot be useful in finding the semantic relationships between the queries.

A model for search engine queries and a variety of quasi-similarity measures to find the relevant queries is presented in Zaïane and Strilets (2002). Fonseca et al. (2003) presented a new method to find related queries based on association rules. Lu et al. (2009) proposed a mining algorithm called EARS. This algorithm uses the Vector Space Model to effectively mine association rules to connect query keywords and concepts to words in text documents. Fonseca et al. (2005) proposed a method which follows a concept based approach where the concept is extracted by analyzing and locating the cycles in a directed graph of query relations that are mined from

association rules. These concepts are then shown to the user who selects the most relevant concept to the current query.

Sahami and Heilman (2006) used query similarity based on the snippets. In this, snippets are used as queries to the search engine to find documents that contain the terms in the snippet. The returned documents are then used to create a context vector for the original snippet. However, the approach does not consider the feedback of the users and thus it is affected by web spam.

Improving the search engine results using the query logs has become the research focus for many researchers. Joachims (2002) was the first one to use click-through data as relevance feedback for the results retrieved for a given query. The paper focused on learning to rank the documents. Later, Joachim et al. (2005) analyzed the issues in the usage of click-through data. The usage of query logs as a source of implicit feedback to improve the search performance is surveyed by Agichtein et al. (2006).

Most of the previous work on the utilization of query logs focuses on finding the similarity among queries to provide query expansion, suggestion or reformulation. Boldi et al. (2009) analyzed search user sessions, classified query reformulation types and derived query-flow graphs for query recommendations. Spink et al. (1998) surveyed information related to successive Web searches and found that there are changes and shifts in search terms, search strategies, and relevance judgments. Jansen et al. (2009) analyzed successive queries in large web search query logs. Cao et al. (2008) and Sengstock et al. (2011) applied click-based clustering to session-based query suggestions and claimed that the context awareness helps to better understand user's search intent and to make more meaningful suggestions. Beeferman and Berger (2000) suggested a technique for query clustering based on the concept of similar strings between queries. This method is content-ignorant. It constructs a bipartite graph to cluster both the queries and URLs/documents based on the iterative clustering approach. This method can be used to generate a list of related queries formulated for a selected query. Primarily, this method is based on the co-occurrence of URLs within the click-through data. An algorithm for query clustering based on the context of the query, common clicked URLs between queries, similar strings between queries and the distance of the clicked documents in some pre-defined hierarchy is proposed by Wen et al. (2001). Query clustering algorithm by combining both the content-aware and link-based clustering approach is proposed by Wen et al. (2002). A method is proposed for the reformulation of a query by rewriting the original query to better match the

relatively small number of advertisers such that recall can be improved without affecting the original intent of the user (Jones et al. 2006). Chien and Immorlica (2005) shown that similarity of two queries can be measured from the temporal correlation coefficient of their frequency functions. Baeza-Yates et al. (2004a, 2004b, 2007) used the term-weight vector model for finding the semantic similarity among the queries. They considered terms in the clicked URLs in the model representation.

3.2 Methodology

This section, first explains the proposed QDC graph structure which holds queries, documents, and concepts in a special arrangement. The importance of the graph structure and how this resolves the existing problems in query clustering are explained by comparing it with the graph structure used in other methods. This section, then describes the different phases of the proposed query recommendation system. The phases are: Feature Selection i.e. feature extraction and filtering, QDC tripartite graph construction using click-through log data, and QDC-clustering. The query recommendation problem is modeled in four different ways. Two models are non-personalized and personalized content-ignorant models. Other two are non-personalized and personalized content-aware models. Clustering is applied on all four models. Overall system architecture is given in Figure 3.1.

3.2.1 QDC Tripartite Graph Structure

For a given query, a search engine returns documents which are searched based on the general context of the query. As illustrated earlier, other similarity finding approaches are inefficient because of ambiguity in the query. If we know the over-arching idea(s) of the underlying documents, then the conceptual similarities among queries can be measured easily. Concepts, which are the soul of a document, will play an important role in finding the theme of the document.

Users' feedback about their intention will be extracted from their own clicked behavior. Therefore, clicked documents are also an integral part of the process. There will be loss of

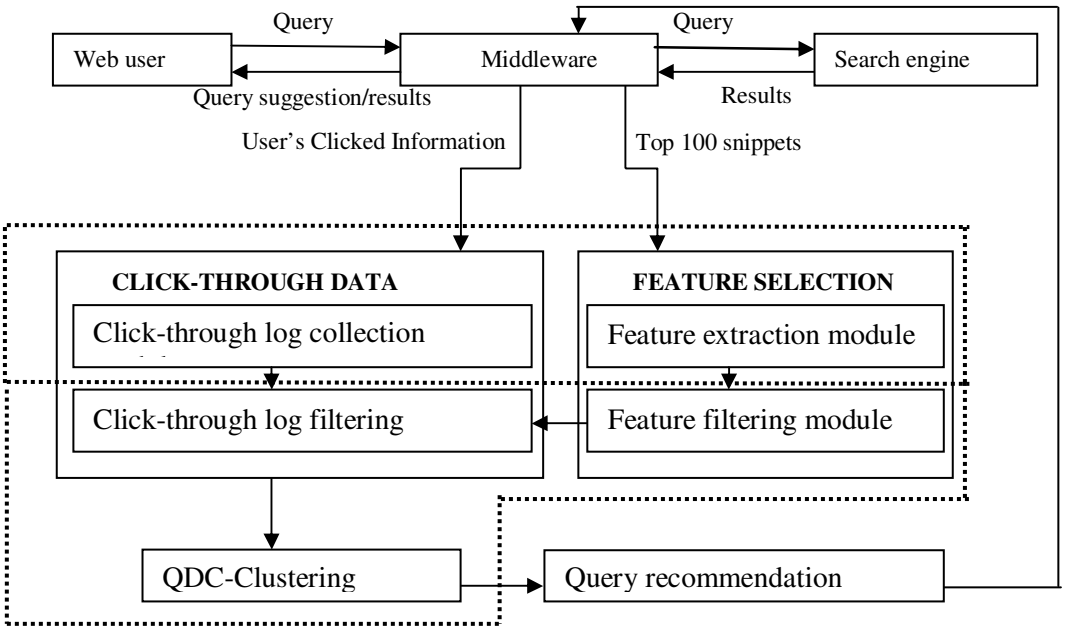


Figure 3.1. System Model

information if user feedback is not considered. Some examples are given next to support the above concepts.

Example 1: Consider Figure 3.2, in which for queries q_1 and q_2, documents $\{d_1\}$ and $\{d_2, d_3\}$ are clicked respectively. The documents d_1, d_2 and d_3 contain the concepts c_1, c_1 & c_2 and c_2 respectively. In Beeferman and Berger (2000), the above information is stored as Query-Document/URL (QD) pair as shown in Figure 3.2a. The queries q_1 and q_2 cannot be clustered as there is no common clicked document. However, there are common concepts. The same information, stored as Query-Concept (QC) pair (Leung et al. 2008), is shown in Figure 3.2b. In this case, two types of information are missing. First, the number of documents clicked for the queries q_1 and q_2 containing the concept c_1. Second, which document is clicked for which query. Therefore, one cannot cluster the queries based on the concepts alone. Thus, both documents and concepts have been considered for query clustering as shown in Figures 3.2c and 3.2d. The representation 3.2c (QCD) considers all three in order. In this case, although we have information about the documents clicked for the queries sharing the same concept, but the information about which document is clicked for which query is missing. To overcome these problems, the representation shown in Figure 3.2d (QDC) has been considered in this work. QDC representation has documents in the middle which decouples the relation between the queries and the concepts. It precisely stores information about all documents clicked for a query,

all concepts extracted from a document and documents involved in sharing the concepts for different queries. This structure clearly indicates the number of documents clicked for a query. The QDC representation is the best suited structure to precisely store all the information present in the query log.

Example 2: Consider queries q_1 and q_2 as "eye parts" and "Apple computer parts" respectively. The "Eyeball" concept (c_1) is present in snippets of document d_1 and d_2 which are clicked for q_1 and q_2 respectively. In the QC representation, there are chances of considering queries q_1 and q_2 as similar queries (Figure 3.3a & 3.3b) because extra information gets added that both the documents are clicked for both the queries.

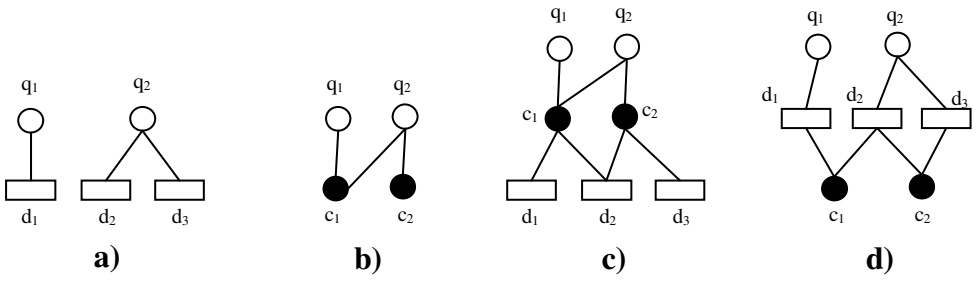


Figure 3.2. Graph of different methods

Example 3: Consider a query "Apple" and concepts, "iPod" and "iPhone", which are present in the snippet of the document d_1 clicked by the user (shown in Figure 3.3c). In this case, two or more concepts that are present in a single document have a greater chance of being similar. This information is missing in QC representation (as shown in Figure 3.3d). But, there are also fair chances of not being similar due to their occurrence in advertisements, downloads etc. on the same page. This situation can be handled by considering weights on the links of QDC representation (see Section 3.2.5.2).

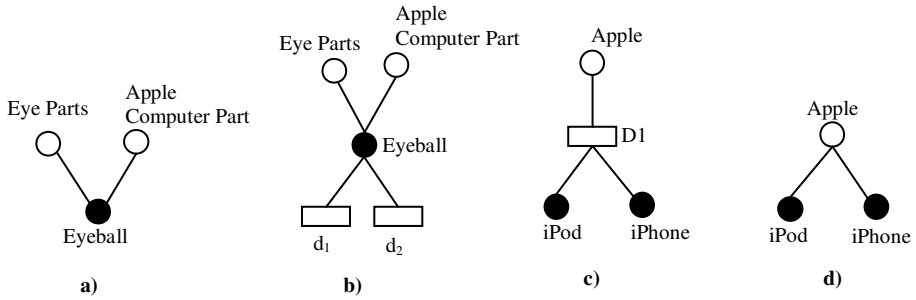


Figure 3.3. (a) & (b) show QC and QCD Graphs for Example 2 respectively and (c) & (d) show QDC and QC Graphs for Example 3 respectively

3.2.2 Feature Selection

The goal of feature selection is to select the necessary number of features. Less number of features will lead to loss of information content, whereas too many (irrelevant) features may lead to noise overshadowing the actual information. In the proposed system, feature filtering is carried using 'selection by elimination' approach (for more details refer chapter2). In this work, the concepts with greater weights W1 (> threshold δ) (refer Chapter2) are considered for the QDC clustering.

3.2.3 Click-through Data Preparation

The search engine gives a list of web pages along with snippets, titles and URLs in response to a query. By seeing these details, user clicks on some pages that appear to be most relevant to the query. The click information is stored in the click-through log. Therefore, click-through log can act as a repository of feedback of the web user's behavior. User's needs and intentions are implicit in the click-through log which can be mined and used appropriately.

There was no standard dataset available which had queries, associated clicked documents and its snippets when we carried out this work. So, the experiment was conducted by involving 20 users and is described in chapter 2. The table 3.1 describes the statistics about the click-through information. The same users are involved in collecting the click-through log. All the features are extracted online from the clicked URLs for a given query adding an entry of the respective concepts along with clicked URLs in the click-through log. Click-through log records an entry for every search. The following information from the click-through log are used:

$$Entry :=< U, Q, D_i, C >$$

where, U denotes the User-ID, Q denotes the query submitted (without any modification). D_i represents the i^{th} clicked document for the query Q and C represents the set of concepts related to the document D_i. The concepts in each entry of the query log are filtered offline using the method given in Chapter 2.

Table 3.1. Data statistics for clustering experiments

Number of users	20
Max No. of queries assigned to each user	5
Number of test queries	85
Number of unique queries	71
Max. No. of URLs clicked for a query	10
Min No. of URLs clicked for a query	2
Number of unique URLs clicked	595
Total Number of concepts selected (which is having greater threshold) for the clicked URLs	2967
Total Number of unique concepts selected (which is having greater threshold) for the clicked URLs	1669

3.2.4 QDC Clustering

The proposed methodology is based on agglomerative hierarchical query-document-concept clustering named as QDC-Clustering on tripartite graphs (See Section 3.2.4.1).

This research proposes four different models on which QDC-Clustering has been applied. The first model is content-ignorant (without weight) and the second is content-aware (with weight). The first two models are non-personalized. The third and the fourth models are the personalized version of first and second models respectively. The QDC-Clustering is comprised of the following steps:

1. The QDC tripartite graph is constructed whose left side vertices correspond to queries and right side vertices correspond to concepts and vertices between them correspond to documents. The graph construction algorithm is given in Section 3.2.4.1.
2. An agglomerative clustering algorithm is applied to the QDC tripartite graph constructed in step 1 and is described in section 3.2.4.2.
3. The clusters of queries and related clusters of documents and related clusters of concepts are extracted from the output of the step 2 of the algorithm (see Section 3.2.4.3).

1. QDC Tripartite Graph Construction

The steps involved in the QDC tripartite graph construction are given in the Algorithm 1 of Figure 3.4.

2. QDC Clustering Algorithm

QDC-clustering algorithm finds clusters of similar queries, the associated clusters of similar documents and the associated clusters of similar concepts. The algorithm alternatively merges the two most similar queries and two most similar concepts. This merging is an iterative process to merge similar clusters into a single cluster of the same type, which is based on the similarity between the cluster nodes. The merging process will terminate when the termination condition is attained. The terminating condition is as follows:

$$\max_{q_i, q_j \in Q} \mathrm{sim}(q_i, q_j) = \sigma_1 ; \max_{c_i, c_j \in Q} \mathrm{sim}(c_i, c_j) = \sigma_2$$

In case, σ_1 and σ_2 are zero, one single big cluster is formed for each of Q, C & D. The QDC Clustering algorithm is given in Algorithm 2 in Figure 3.4.

Algorithm1: Tripartite Graph construction	**Algorithm2:** QDC-Clustering
Input: Click-through data CT, Concept Set CS, Empty Graph G(V,E). **Output:** QDC Tripartite graph G={QUDUC, E}	**Input:** QDC Tripartite graph G, σ_1 , σ_2 **Output:** Graph G' containing clusters of queries, respective documents, and clusters of concepts
Build a set Q = {q_1, q_2, q_3 ….. q_n} of unique queries from CT for each q ∈ Q do Create a "Q" vertex in G Build a set D ={d_1, d_2, d_3,...,d_m} of unique documents from CT for each d ∈ D do Create a "D" vertex in G for each document d ∈ D clicked for a query q ∈ Q do add an edge (d,q) in E Build a set C = {c_1, c_2, c_3, ...,c_z} of unique concepts from CT for each c ∈ C do Create a "C" vertex in G for each document d ∈ D contains a concept c ∈ C do add an edge (d,c) in E	do for each unique query pair (q_i,q_j) in G do calculate sim(q_i,q_j) if sim(q_i,q_j)=max(sim(q_l,q_m)) ∀ q_l,q_m∈ Q and >σ_1 Merge (q_i,q_j) for each unique concept pair (c_i,c_j) in G do calculate sim(c_i,c_j) if sim(c_i,c_j)= max(sim(c_l,c_m)) ∀ c_l,c_m∈ Cand >σ_2 Merge (c_i,c_j) While (∃ some pair such that sim(q_i,q_j) > σ_1 or sim(c_i,c_j) > σ_2)

Figure 3.4. Tripartite graph construction algorithm and QDC-clustering algorithm

3. Extracting Clusters

The output of the QDC-clustering algorithm will have on one side, clusters of the queries, and on the other side, clusters of concepts. Documents with links to both sides are in between the clusters of queries and concepts. We extract the clusters of queries and the related clusters of documents by collecting all the documents pointing to them.

3.2.5 Models

In this section, the proposed four models are described. Models are classified as non-personalized and personalized. Each of these models is further classified as content-ignorant and content-aware**.**

1. Non-Personalized Content-Ignorant Model (QDC)

In the non-personalized content-ignorant model, the similarity between the nodes of the same type is based only on the user's feedback. A new similarity measure is proposed for merging two similar query clusters.

Merging of Query Clusters

A similarity measure to find similarity between two query clusters (initially each query is considered as a singleton query cluster) Q_i and Q_j is proposed. The similarity, which is based on the strength of the relation between the queries and concepts, is calculated. The most similar concept/concept-set is found for each query cluster and then the two query clusters are considered to be similar when the most relevant concepts for both are same and order of relevance with the concepts is also same.

The formula to calculate the relevance *rel(Q,c_i)* of concept c_i with query cluster Q:

$$\mathrm{rel}(\mathrm{Q}, \mathrm{c_i}) = \frac{\mathrm{m_i}}{\mathrm{N}} \tag{3.1}$$

Where, N is the number of documents linked/clicked for the query cluster Q and m_i is the number of the clicked documents which contain the concept c_i.

In QDC, different measures are used for finding query similarity and concept similarity unlike the existing methods QD (Beeferman and Berger 2000) and QC (Leung et al. 2008). This is because the intention behind the queries has to be found by considering the query and concept

relevance using clicked information. Otherwise, issues present in the earlier approaches will remain unaddressed.

Merging of Concept Clusters

The similarity between two concept clusters (initially each concept is considered as a singleton concept cluster) is given by the correlation between the concepts, i.e., the ratio of the number of documents containing both the concepts and the total number of documents containing at least one concept. All the documents considered here are the ones clicked for the same query cluster. The correlation is calculated using the following function (Fonseca et al. 2003):

$$\text{sim}\left(C_i, C_j\right) = \begin{cases} \frac{\left|n(C_i, C_j)\right|}{\left|n(C_i) \cup n(C_j)\right|}, & \text{if } \left|n(C_i) \cup n(C_j)\right| > 0 \\ 0, & \text{Otherwise} \end{cases} \quad (3.2)$$

where, $n(C_i,C_j)$ denotes the set of clicked documents linked to both C_i and C_j , $n(C)$ denotes the set of documents linked to C.

2. Non-Personalized Content-Aware Model (WQDC)

Similarity measures using only users' feedback tend to cluster queries with the same or similar intention. Whereas, similarity measures using only content tend to cluster queries with the same or similar terms. Both the approaches have drawbacks as user' feedback may have noisy information and users' needs may not be fully captured by query text and the relevant documents. In this model, both content and users' feedback are used to measure similarity between the nodes. This incorporates the advantages of both the approaches. In this work, two new similarity measures (one for query similarity and one for concept similarity) are proposed.

Consider a query "Apple" and the concepts "iPod" and "iPhone" from the clicked document D1. In this case, two or more concepts that are present in a single document have a greater chance of being similar. This information is missing in QC representation, but captured in QDC. But, two or more concepts may not be similar because of their occurrence on the same page in advertisements, downloads, etc. The consideration of the content of the documents will help in dealing with such situations.

In this model, the QDC-tripartite graph is constructed in the same way as given earlier, except the inclusion of the edge weight. Edge weights between queries and documents and between concepts and documents are measured as follows:

$$W(E(x,d)) = \left(\sum_{d_j \text{ related to } x}\left(\frac{n_{ij}}{\sum_{k=1}^{z} td_{kj}} \log(N/m_i)\right)\right) \quad (3.3)$$

Where, x can take two values, either query or concept. n_{ij} is a number that the number of times the concept c_i occurs in the document d_j, z is the total number of distinct terms in the document d_j. $\sum_{k=1}^{z} td_{kj}$ is the total number of terms in the document d_j. N is the total number of clicked documents linked to x. m_i is the total number of linked documents in which the concept c_i appears.

Merging of Query Clusters

The similarity between two query clusters (initially each query is considered as a singleton query cluster) is calculated by incorporating the content of the documents. We have extended the measure used for a content - ignorant model. The following is the formula proposed to calculate the relevance rel(Q,c_i) between Q and c_i:

$$rel(Q, c_i) = \left(\sum_{d_j \in Q \text{ and contains } c_i}\left(\frac{n_{ij}}{\sum_{k=1}^{z} td_{kj}} \log(N/m_i)\right)\right)\frac{m_i}{N_Q} \quad (3.4)$$

Where, all the symbols have the usual meaning. N_Q is the count of the number of documents clicked for the query cluster Q. After merging a pair of query clusters, the weight given to the new link between merged queries and the documents is the maximum of weights associated with the previous links.

Merging of Concept Clusters

The similarity between two concept nodes sim(C_i,C_j) is measured using the following proposed similarity measure:

$$sim(C_i, C_j) = \begin{cases} \frac{\sum_{d \in D_{ij}} min\left(W(E(d,C_i)), W\left(E(d,C_j)\right)\right)}{\left(\sum_{d \in D_i} w(d,C_i) + \sum_{d \in D_j} w(d,C_j)\right)} ; if\ D_{ij} \neq \emptyset \\ Diff\left(rel(q,C_i), rel(q,C_j)\right) ; if\ D_{ij} \neq \emptyset\ for\ some\ query\ q \\ 0; Otherwise \end{cases} \quad (3.5)$$

After merging, the minimum edge-weight is given to the new edge.

3. Personalized Models (PQDC & WPQDC)

Individual user's interest can be considered for further improving the query clustering by adding user_id along with the query in QDC-clustering. This will bring personalization in query recommendation. Users may give the same query with same or different intentions. Considering user with the query will disconnect the relation between the same queries submitted by different users with different intentions. This can be achieved in the following two ways:

1. Same queries issued in different sessions are considered as different queries, leading to the problem of session segmentation.
2. Queries issued by different users are considered as different queries, irrespective of whether the queries are same or different.

In this work, the second approach has been considered. Each individual query submitted by each user appears as an individual node in the graph G (see section 3.2.4.1) by labeling each query with a unique user identifier i.e. Q→UQ. The tripartite graph of PQDC is constructed in the same way as in the QDC tripartite graph (see Section 3.2.4.1). Here, the links between the two (UQ and D) are limited to the documents which are clicked for a query by the same user. The same measures used in QDC and WQDC models are used for the respective personalized versions of the models (PQDC & WQPDC). The Personalized QDC-Clustering algorithm is given in Figure 3.5.

3.2.6 Query Recommendation

In the last phase of the system, similar queries with the corresponding concepts are presented to the users as query suggestions. This allows users to build a proper search query with the knowledge domain terminology which will help the search engine to get the desired results. In the personalized version of the models, the query clusters will have queries that are issued by the same user with similar intentions. This will add a personalized query suggestion according to the users' conceptual needs.

Algorithm3: Personalized QDC-Clustering

Input: Personalized QDC Tripartite graph G, σ_1, σ_2

Output: Graph G' containing clusters of queries, respective documents, and clusters of concepts

Do
 for each unique query pair (q_i,q_j) in G
  do calculate sim(q_i,q_j)
   if sim(q_i,q_j)=max(sim(q_l,q_m)) $\forall\, q_l,q_m \in Q$ and $> \sigma_1$ and does not contain the same queries from different users
    Merge (qi,qj)
 for each unique concept pair (ci,cj) in G
  do calculate sim(ci,cj)
   if sim(c_i,c_j)= max(sim(c_l,c_m)) $\forall\, c_l,c_m \in C$and $>\sigma_2$ and does not contain the same queries from different users
    Merge (c_i,c_j)
 for each unique query pair (q_i,q_j) in G
  do calculate sim(q_i,q_j)
   if sim(q_i,q_j)=max(sim(q_l,q_m)) $\forall\, q_l,q_m \in Q$and $> \sigma_1$ and contain the same queries from different users
    Merge (q_i,q_j)
While $\exists$ some pair such that sim$(q_i,q_j) > \sigma_1$or sim$(c_i,c_j) > \sigma_2$

Figure 3.5. Personalized QDC-Clustering Algorithm

3.3 Experimental Results

This section presents the results which show the performance of the proposed models against various previously published methods on the dataset described in section 2.3.1 of chapter 2. The proposed models are compared with QD, QC and their variants. The personalized version of QD (PQD) and weighted QC (WQC) are developed by us for comparison purposes as they are not available in the literature.

All the models are applied on the collected data and the respective clusters of queries, concepts, and documents are formed. The statistics of the number of vertices in the graph for different models of QDC clustering and other methods are given in table 3.2. Purity and Inverse purity are the basic measures to measure the performance of the clusters. Purity measures the homogeneity of a clustering (Amigo et. al. 2009). Let *S* be the set of queries (documents) that exists in the predefined cluster and S' the set of related queries q_1, q_2,..., q_n (documents d_1,d_2,...,d_m) generated

by the algorithm and *N* be the number of clustered items, Purity is computed by taking the weighted average of maximal precision values using the following formula:

$$\text{Purity} = \sum_i \frac{|S'_i|}{N} \max_j \text{Precision}(S'_i, S_j) \quad (3.6)$$

Where the precision of a cluster S'_i for a predefined cluster S_i is calculated as follows:

$$\text{Precision}(S'_i, S_j) = \frac{|S'_i \cap S_j|}{|S'_i|} \quad (3.7)$$

Purity focuses on the frequency of the most common predefined cluster in each obtained cluster. Inverse Purity (Amigo et. al. 2009) favors small error in a big cluster to large number of small errors in small clusters. It focuses on the cluster with maximum recall for each predefined cluster. Inverse purity is computed by taking the weighted average of maximal precision values using the following formula:

$$\text{InversePurity} = \sum_i \frac{|S_i|}{N} \max_j \text{Precision}(S_i, S'_j) \quad (3.8)$$

Although, purity penalizes the noise in a cluster, but it does not reward grouping items from the same category together. Perfect scores for purity are obtained by assigning all items to separate clusters. Similarly, grouping all items together in one single cluster results in maximizing inverse purity.

Due to the above mentioned drawbacks of the measures Purity and Inverse Purity, Amigo et. al. (2009) proposed methods such as precision, recall and F-measure. These measures cover four properties including homogeneity, error in big clusters vs. error in small clusters, cluster completeness and pattern of noise present in clustering. Cluster completeness is about placing homogenous items in the same cluster and pattern of noise is associated with the placement of noisy items in heterogeneous cluster(s) rather than homogenous clusters. So, the measures recall, precision and F-measure are used for the comparison of results. These measures are calculated for the query clusters obtained from various models and compared with the predefined clusters. The predefined clusters are manually formed clusters by placing the relevant queries in a cluster. The number of predefined clusters and the number of clusters formed by various models and methods are given in Table 3.3.

The precision (*P*), recall (*R*), and F-measure (*F*) for a given query *q* and associated query cluster q_1, q_2,...,q_n produced by a clustering algorithm, are computed using the following formulae respectively:

$$P(q) = \frac{|S \cap S'|}{|S|} \tag{3.9}$$

$$R(q) = \frac{|S \cap S'|}{|S'|} \tag{3.10}$$

$$F = \left(2.\frac{(P.R)}{(P+R)}\right) \tag{3.11}$$

S is the set of queries that exists in the predefined cluster for *q* and S' is the set of related queries q_1, q_2,...,q_n generated by the algorithm. The precision and recall values from all queries are averaged. The predefined clusters are manually formed by placing similar intend queries into respective clusters, as there is no gold standard cluster information available for this dataset.

Table 3.2. The number of vertices in the graph for different Models of QDC clustering and other methods

Methods	**# of "C" vertex**	**# of "D" vertex**	**# of "Q" vertex**
QC & WQC	1669	-	71
QD	-	595	
QDC & WQDC	1669		
PQD	-		85
PQDC & WPQDC	1669		
PQC & WPQC		-	

Table 3.3. Number of clusters formed by different models and methods Vs Number of predefined clusters

Method	**# of clusters formed**	**# of predefined clusters**	**Method**	**# of clusters formed**	**# of predefined clusters**
QD	71	50	PQD	82	64
QC	25		PQC	62	
QDC	59		PQDC	71	
WQC	29		WPQC	41	
WQDC	48		WPQDC	61	

Most of the experimental results are presented in terms of best mean precision, recall and F-measure because every algorithm performs best at a different value of cut-off-similarity score σ.

For a particular value of σ, one algorithm may work the best, whereas other algorithm can be worse.

The three measures, precision, recall and F-measure, for various values of merging cut-off similarity score σ, is given in Table 3.4.

It is clear from the table that all QDC clustering models have achieved both higher precision and higher recall than other methods. The QDC clustering models are performing best at σ =0.3 except for WQDC which is best at σ =0.25. The values of both precision and recall increase as σ is changed from 0.25 to 0.3 except in WQDC. Whereas, the QD and QC methods are showing either increase in precision and decrease in recall or decrease in precision and increase in recall at all values of σ. It is also clear from the table that the QDC models are able to achieve F-measure above 0.7, whereas others methods have F-measure below 0.7. Moreover, WQDC and WPQDC could achieve 0.86.The best F-measure values achieved by different methods and models as well as the corresponding best cut-off similarity score σ are given in Table 3.5.

Table 3.4. Precision, Recall and F-Measure values of all models and methods for different cut-off similarity scores σ

σ	**0.25**			**0.3**			**0.35**			**0.4**		
Methods	P	R	F	P	R	F	P	R	F	P	R	F
QD	0.8333	0.4778	0.6073	0.7890	0.5055	0.6162	0.8333	0.4778	0.6073	0.8979	0.4501	0.5996
QC	0.5498	0.7833	0.6461	0.5786	0.7255	0.6438	0.6369	0.7167	0.6744	0.7255	0.5972	0.6552
QDC	0.7100	0.6267	0.6657	0.7694	0.6763	0.7199	0.7900	0.6433	0.7092	0.7900	0.6433	0.7092
WQC	0.6063	0.7063	0.5608	0.6915	0.6506	0.5797	0.7377	0.6219	0.5950	0.7411	0.5241	0.6141
WQDC	0.8890	0.8333	0.8603	0.9168	0.7918	0.8497	0.9168	0.7083	0.7992	0.9168	0.6900	0.7874
PQD	0.8083	0.5885	0.6811	0.7793	0.5226	0.6257	0.8519	0.5231	0.6482	0.9244	0.4578	0.6123
PQC	0.6207	0.7054	0.6603	0.6607	0.6917	0.6758	0.6607	0.6917	0.6758	0.6818	0.6542	0.6677
PQDC	0.7778	0.6372	0.7005	0.8000	0.6550	0.7203	0.8148	0.5972	0.6893	0.8148	0.5972	0.6893
WPQC	0.5919	0.5805	0.4978	0.6781	0.5813	0.5453	0.7724	0.6327	0.6287	0.7411	0.5420	0.6256
WPQDC	0.8890	0.8333	0.8603	0.889	0.8333	0.8603	0.8890	0.7223	0.7970	0.8890	0.7223	0.7970

From the Experimental results it is observed that the cut-off similarity score σ value will differ for different methods and models. So we have presented the F-Measure value for fixing up the best σ value for all the methods and models. These parameters are empirically analyzed in our

experiments. Table 3.5 summarizes the cut-off similarity score for all the methods and models based on the experimental results.

The precision-recall curves are also plotted for the clusters obtained by different models and methods. Figures 3.7a, b, c and d show the figures for (QD, QC, QDC), (WQC, WQDC), (PQD, PQC, PQDC) and (WPQC, WPQDC) respectively. Number labels show the number of clusters at that point. It can be seen in Figure 3.6a. that the large number of clusters formed by QD is at low recall. This is because QD purely depends on common clicks. But, the chance of clicking the common documents even for the same query by the different user is quite less. This results in the formation of more number of small clusters, lead to decrease in value of recall and increase in value of precision. Precision or recall values for most of the clusters formed by QC are low as the method results in less number of clusters (25). This is because QC is merging the queries if common concepts are linked. The values of the measures are even worst when two same queries with distinct intentions having common concept(s) (the average number of overlapping concepts between the queries is 56).

Table 3.5. Best precision, recall and F-Measure values of all the models and methods

Method	**P**	**R**	**F**	**σ**
QD	0.7890	0.5055	0.6162	0.30
QC	0.5786	0.7255	0.6438	0.35
QDC	0.7694	0.6763	0.7199	0.30
WQC	0.7411	0.5241	0.6141	0.35
WQDC	0.9168	0.7918	0.8497	0.25
PQD	0.8083	0.5885	0.6811	0.35
PQC	0.6607	0.6917	0.6758	0.35
PQDC	0.8000	0.6550	0.7203	0.30
WPQC	0.7724	0.6327	0.6287	0.35
WPQDC	0.8890	0.8333	0.8603	0.30

QDC overcomes the drawbacks of both QD and QC as concepts and queries are decoupled by the clicked documents. As a result precision and recall both are high. More than 50% of the

clusters formed by QDC have highest values of precision and recall. But the QDC treats all the common concepts equally whether or not the meaning of these concepts is related to the queries under consideration. The proposed WQDC model overcomes this problem by attaching weights to the links.

The weights grade the relevance of the concepts to the queries. Thereby, resulting in clusters with high precision and recall values (43/48 ratio where P=R=1) (see Figure 3.6b). The WQC method is worst amongst all methods and models as almost all the clusters formed are having low values of precision and recall. The performance of personalized QDC against other personalized methods is also analyzed. Figure 3.6c & d show the comparison of precision recall figures of PQDC with that of PQD and PQC and WPQDC with WPQC respectively. In the personalized approach, two queries with the same keywords posed by two different users are treated as different queries. This will help in finding/separating the queries with different meaning even if the query keywords are same. Therefore, the number of clusters formed by various personalized methods is more. It is observed that PQD behaves in the same manner as QD. The problem in QC discussed above persists in PQC as well. The clusters formed by PQDC and WPQDC are at higher precision and recall values and performing best among all the methods and models.

The percentage improvements in F-measure of our models over others for different values of σ are calculated and presented in Figure 3.7. In Figure 3.7a, percentage improvement of our QDC model over QD, QC and WQC has been plotted. It shows that the improvement is up to 18%, 12% and 25% respectively. In Figure 3.7b, the WQDC is compared with QD, QC, QDC and WQC. Around 30% of improvement over WQC at σ=0.4 and it increases up to 50% with decrease in σ. A similar increase in improvement is found for other algorithms. The improvement in F-measures of PQDC model with other personalized methods as PQD, PQC, WPQC (given in Figure 3.7c) are around 5% to 15% except in few cases such as WPQC is working poorly for σ=0.25 and 0.3. The improvement curves for WPQDC are given in Figure 3.7d. The behaviour is same as of PQDC in Figure 3.7c but the values are higher (between 20%-40%) except few cases where WPQC is performing badly. In all, it can be stated that the WQDC and WPQDC are performing outstandingly as compared to all other methods and models. However, QDC and PQDC are also better than all previously published algorithms.

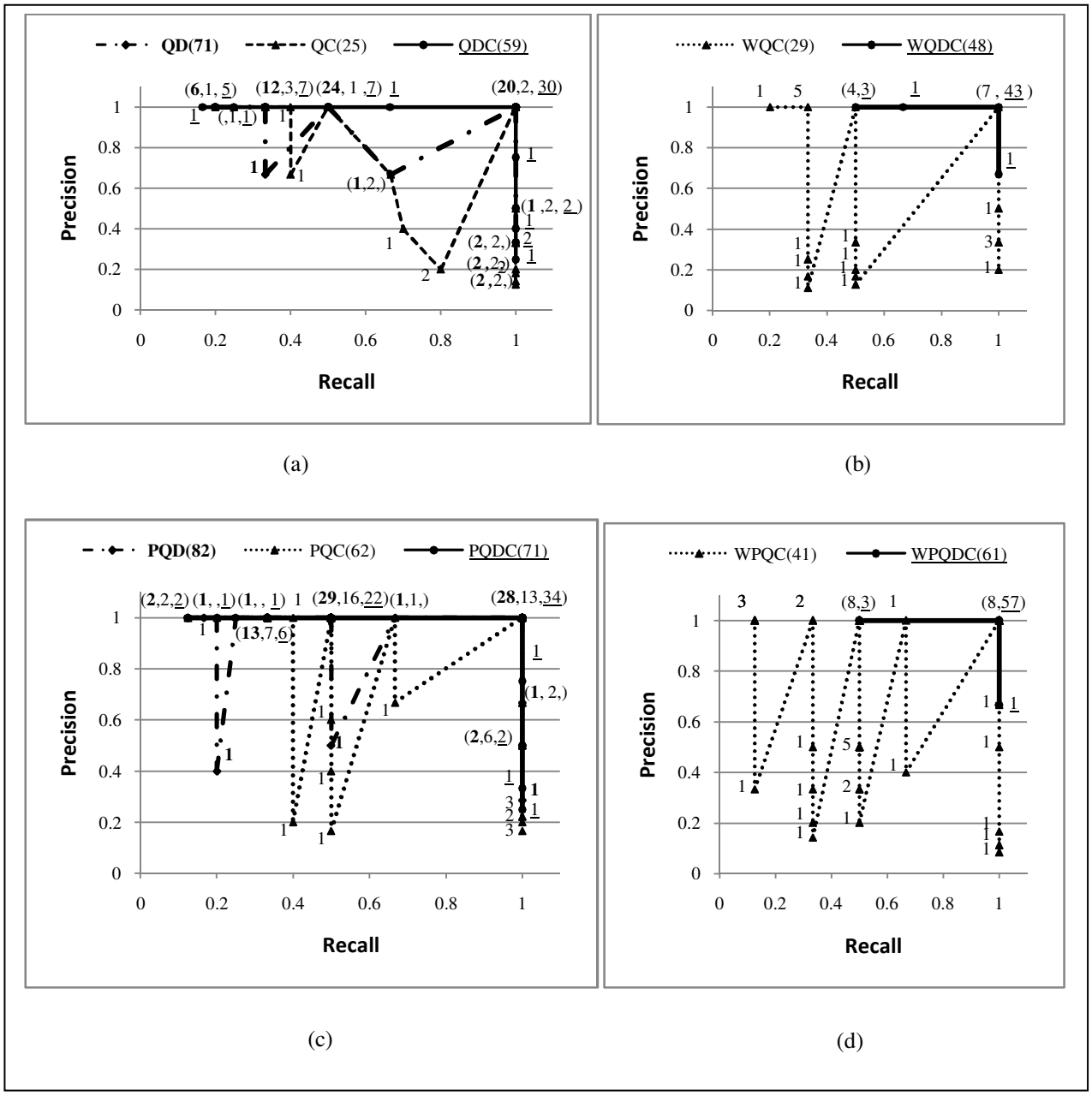


Figure 3.6. Precision vs. Recall Graph

A sample queries and clusters formed by all approaches are given in the Table 3.6. It is clear from the table that QD method is not able to find the related queries in most cases due to the lack of common clicked documents among similar queries. QC method merges queries which are not similar as well in a single cluster due to the absence of clicked document information in it. QDC model outperforms well as compared to QD and QC because of its own properties.

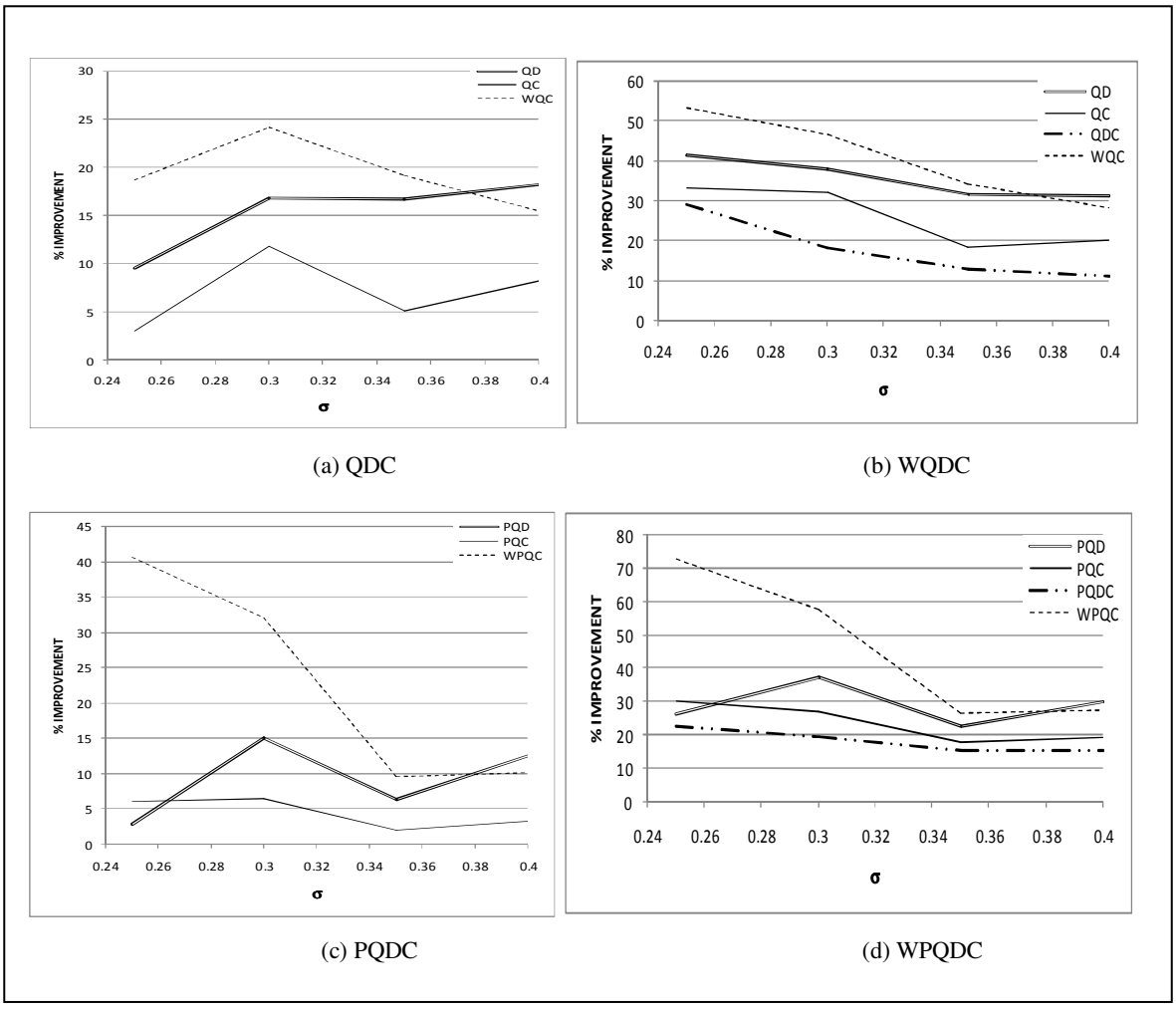


Figure 3.7. Percentage improvement Graph of different models over other models and methods

But QDC model is not able to differentiate the queries intention in the case of presence of similar concepts. For example, the concept "learning problems" has higher weight value with respect to both the query "Mobile handwritten recognition learning problems" and "automation learning problems" even though the query "automation learning problems" intention is robotics automation learning. WQDC surmounts this drawback by considering the document content in similarity finding.

Table 3.6. Sample queries and clusters formed by different models and methods

Sample Queries	**Clusters formed by QD**	**Clusters formed by QC**	**Clusters formed by QDC**	**Clusters formed by WQDC**
Mobile handwritten recognition learning problems, automation learning problems, robotics and automation, learning problems conference paper submission, machine learning, conference submission system, easychair, world wide web conference 2011 easy chair paper submission, AUSDM 2011 easy chair paper submission	[Mobile handwritten recognition learning problems], [automation learning problems], [robotics and automation], [learning problems conference paper submission, machine learning] [conference submission system, easychair], [world wide web conference 2011 easy chair paper submission], [AUSDM 2011 easy chair paper submission]	[Mobile handwritten recognition learning problems, automation learning problems, robotics and automation, learning problems conference paper submission conference submission system, easychair, world wide web conference 2011 easy chair paper submission, AUSDM 2011 easy chair paper submission], [machine learning]	[Mobile handwritten recognition learning problems, automation learning problems], [robotics and automation], [learning problems conference paper submission], [machine learning] [conference submission system, easychair], [world wide web conference 2011 easy chair paper submission], [AUSDM 2011 easy chair paper submission]	[Mobile handwritten recognition learning problems], [automation learning problems, robotics and automation], [learning problems conference paper submission], [machine learning] [conference submission system], [easychair], [world wide web conference 2011 easy chair paper submission], [AUSDM 2011 easy chair paper submission]

3.4 Conclusions

In this work, a system for query recommendation that clusters the queries, associated clusters of concepts and documents has been proposed. These clusters are hierarchically built using the proposed tripartite graph structure. User's preferences are captured by using click-through graph of web log. The effect of noise in the click-through data is suppressed by using the selected features which helps in identifying the user's intention. An efficient feature selection method is used to highlight the features which are relevant to the queries. Different new similarity measures are proposed and used effectively to measure the similarities of different entities. Four proposed QDC-clustering models are applied and analyzed. It is clear from the results that the proposed models can effectively group similar queries together as compared to existing methods. Moreover, it is shown that personalized versions of QDC-clustering models outperform the non-personalized version of QDC-clustering models' and content-aware models' performance is better than that of content-ignorant models. Never the less, the content-ignorant models performance is better than all other previously published methods. This work concludes that applications that need light weight models may use content-ignorant QDC models. This approach can be used to build not only query clusters, but also concept clusters, associated document clusters and user profiles. The proposed system is simple and is able to provide ease to web users to build a proper search query with the knowledge domain terminology which will help search engine to get the desired results.

The approach can be extended to the concept of query-context aware concept/topical document clustering, which will allow the web users to get more precise information as query results.

CHAPTER 04

An Approach for Search Result Topic Set Identification and Labeling

Over the past few years, the content of the web has grown at a brisk pace. Therefore, finding relevant information has become more and more difficult. Existing search engines deliver a considerable amount of results for a particular search query. They try to organize the results such that pages that are more relevant to the query appear first. This process is one of the challenging tasks for the search engine due to the various and dynamic intentions of a query. Moreover, queries may be short, imprecise, vague and ambiguous. It is shown by Jansen and Spink (2006) that approximately 30% of the queries, given to the search engines are single word. It is shown by Sanderson (2008) that about 7%-23% of the queries occurring in the search logs of large web search systems from Microsoft (labeled Web) and the UK's Press Association are ambiguous. Search engines are not able to understand the exact user context, and thus retrieve large volumes of results, most of which are irrelevant to the user. Therefore, many IR researchers have focused on finding relevance between query and document (Bendersky et al. 2011; Carpineto and Romano 2012).

Comprehensive presentation of search results is an important factor for improving search performance. Grouping similar search result documents of a query helps in swift understanding of topics/intensions behind a query. This also facilitates in organizing retrieved results thus helps in resolving the problem of information overlook. However, grouping alone cannot elucidate to the user the intentions behind the groups. Therefore, the need for labeling the groups arises that facilitates users for better understanding. Identification of different intentions of a query is referred as query disambiguation. The query intentions may change over time. For example, the query "green apple" is usually referred as "fruit". Now, the meaning of the query is also "tourism". To catch all the underlying meanings, it is required that the query disambiguation should take place every time when user poses a query (i.e. online). So, a simple unsupervised model is needed, which can be a suitable solution to incorporate such changes. The supervised methods are not feasible as they require a lot of manual effort and as topic granularity varies from query to query.

The unsupervised process of grouping search results can be broadly categorized as follows (Carpineto et al. 2009): *Data-Centric* (eg. Scatter/Gather (Hearst and Pedersen 1996), Tolerance

Rough Set Clustering (TRSC) (Ngo and Nguyen 2004), Agglomerative Hierarchical Clustering (AHC) (Everitt et al. 2001) algorithms apply conventional data clustering techniques (e.g. hierarchical agglomerative, k-means, spectral) to cluster search results and are usually modified to present textual depiction of their clusters to the user. The drawback of this approach is related to the quality of cluster labels which is the most critical part of any Web Search Results Clustering. Therefore, these approaches are out of consideration. *Description-aware* (eg. Suffix Tree Clustering (STC) (Zamir and Etzioni 1998), Hierarchical Suffix Tree Clustering (HSTC) (Masłowska 2003), SnakeT (Ferragina and Gulli 2005)) systems form labels from the features of clustered documents. They are based on the philosophy that if features are defined and relevant they can be used to describe the clusters to users. The limitation here is that clustering still precedes and governs the labeling procedure (Carpineto et al. 2009). *Description-centric* (e.g.. Lingo (Osiński et al. 2004), SRC (Zeng et al. 2004), DisCover (Kummamuru et al. 2004)) systems are based on the motto that "Description comes first" i.e. quality of a cluster description i.e. label is more important than that of a cluster. Thus, cluster label induction is usually followed by documents assigned to them. The performance of the approaches of this category degrades when ambiguous labels are selected (Marco and Navigli 2011). Moreover, a single label may not be able to describe a topic fully. STC assigns snippets to a cluster using phrases (labels) i.e. if a snippet does not include a phrase it will not be a part of cluster even though it is relevant to the topic (Osiński and Gotoh 2004; Carpineto et al. 2009). For example, consider the query *"B-52"* and the associated topic *"drink"*. Documents related to "cocktail" would not be identified as *"drink"* related documents if these documents are missing the term *"drink"*. Lingo suffers from ambiguity in the labels where such labels lead to unrelated snippet assignment to clusters. Consider a query "Beagle" from Ambient dataset (Carpineto and Romano 2008), lingo produces ambiguous labels such as "adopting", "search" etc. Here, label "adopting" may mean adopting beagle software, adopting beagle named child, adopting beagle pets, or adoption related resources. As a result, the cluster has documents related to multiple topics.

The proposed method provides a hybrid of description-centric and description-aware approaches. In this approach, we first cluster features through similarities in documents (i.e. description-aware approach) to form topics or categories and then place documents in these categories using features (i.e. description-centric approach). This resolves the problems with ambiguous labels since features (also referred as concepts in this work) describe the intention of a topic clearly.

The concepts under each topic help in rephrasing queries, mapping documents to the desired topics, comparing two topics, labeling topics etc.

We evaluate the performance of the proposed method by comparing it with both STC and Lingo algorithms that are examples of most popular clustering and labeling algorithms. Their open source implementations are available through Carrot2.

The proposed method uses a set of concepts for a topic/label to decide a snippet to be assigned and thus provides better clusters to users and overcomes STC's problem of frequent phrase labeling and the lingo's problem of ambiguous labels. Moreover, the method can also be used in other applications such as in question answering system where answers to questions can be described through labels and concepts in news articles management system where articles are tagged with various labels and concepts, etc. In these cases user has to just skim through categories in order to get most possible answer/similar articles according to search intention.

The main contributions of this chapter are:

1. Identification of various topics intended for a query and assignment of the documents to these topics.
2. Representation of topics through concepts and labels to describe search results in a more detailed and clear manner and to rephrase queries.
3. GUI has been developed for simple, quick, and easy search outcome.

The rest of the chapter is organized as follows: Section 4.1 provides related work. Section 4.2 details proposed methodology. Section 4.3 and 4.4 provides experimental setup and comparison of the proposed method with existing works respectively. Section 4.5 describes the visualization of topic set and the related search results. Section 4.6 concludes the work and provides future directions.

4.1 Related Work

Search Result Clustering (SRC) can be categorized as data-centric, description-aware and description-centric algorithms (Carpineto et al. 2009).

The main goal of data-centric algorithms is to form the clusters using conventional clustering techniques like partitional (eg. K-Means), hierarchical (eg. Agglomerative Hierarchical Clustering (AHC)), fuzzy (eg. FTCA) and density-based (eg. DBScan) clustering (Jain and Dubes 1988; Jain et al. 1999; Steinbach et al. 2000; Han et al. 2001; Berkhin 2006). These

algorithms mainly focus on clustering rather than labeling. Hence, they fail in explaining the clusters to the user. Scatter/Gather (Hearst and Pedersen 1996), is one of the examples of data-centric algorithms, which partitions a given document collection into small subsets of documents and then performs clustering to merge the obtained clusters. This process of partitioning and then merging will be repeated iteratively. Scatter/Gather labels each cluster by selecting most frequent words in its respective documents. Thus, in a few cases this algorithm faces the problem in bringing the cluster intention clearly. Tolerance Rough Set Clustering (TRSC) (Ngo and Nguyen 2004) is another example of data-centric algorithm. TRSC extracts features using the TRS model and clusters documents using K-means or AHC based on the vector space model. Frequent n-grams are selected as a label for a cluster. Again, same problem with respect to cluster labeling persists as this algorithm also uses frequent keywords to label the cluster. The data-centric algorithms are adapted to produce a comprehensible cluster description: cluster labels typically consist of single words recovered from the cluster representative, possibly expanded into phrases. In data-centric algorithms, cluster labels are nothing but a textual description of the underlying documents as these algorithms do not focus on the quality of cluster labels.

Description-aware SRC algorithms focus on resolving the cluster labeling problem by generating expressive cluster labels. Description-aware algorithms use techniques ranging over suffix trees (Zamir and Etzioni 1998; Masłowska 2003), spectral clustering (Cheng et al. 2005) and formal concept analysis (Carpineto and Romano 2004). The more popular algorithm of this class is Suffix Tree Clustering (STC) (Zamir and Etzioni 1998). The main objectives of STC are online processing and cluster label descriptiveness. STC first tokenizes the documents and forms Generalized Suffix Tree. Initially, each node that crosses minimum threshold of documents becomes a base cluster. STC selects phrases as label for each base cluster that occurs in minimum count of documents. Base clusters are then merged based on a simple similarity ratio and selected phrases, which occurs in maximum number of documents, as a label for the merged cluster. Thus STC was able to overcome the difficulties faced by data-centric techniques by enforcing the need and importance of sensible cluster labels. Another advantage offered by STC was overlapping cluster due to which a single document is assigned under more than one topic since documents often have multiple topics. This algorithm suffers from the problem of assigning a few related documents due to the absence of frequent phrases in them. This was

improved upon by SnakeT (Ferragina and Gulli 2005) where the authors tried to address this problem by introducing possible gaps in phrases.

Description-centric algorithms have a common philosophy of "description-comes-first". This class of algorithms focuses not only on generating superior labels, but also on providing worthy clusters. A few members of this class are (Lingo (Osiński et al. 2004), SRC (Zeng et al. 2004), DisCover (Kummamuru et al. 2004), SHOC (Zhang and Dong 2004)). The more popular algorithm of this class is Lingo (Osiński et al. 2004). The general idea behind Lingo is to first generate expressive cluster labels and then assign documents to these labels. Lingo tries to ensure that each label is distinct and simultaneously cover most of the documents. It uses vector space model coupled with Latent Semantic Indexing (LSI) using Singular Value Decomposition (SVD) to achieve its goal. Thus, it benefits from LSI's ability to capture the semantics of the document collection. However, effect of semantic retrieval is severely affected due to the small size of snippets. Moreover, SVD is computationally very expensive. The limitation of this class of algorithms is that cluster labels generated based on search results are not able to describe user intention. Hence the label may not be useful to the user.

The proposed approach is a hybrid version of the description-aware and description-centric algorithms. It is an online algorithm that aims to overcome the limitations faced by both the class of algorithms and provide an efficient solution by correlating features to form topics, using concepts assigned to the topic to assign documents and finally using both concepts and assigned documents to assign label to the topic. We evaluate our approach against STC (description-aware) and Lingo (description-centric). We used Carrot2 for the available implementation of Lingo and STC.

Further, SRC algorithms can be classified based on how documents are assigned to different clusters. Monothetic algorithms assign documents based on a single feature while polythetic algorithms assign documents based on multiple features. Advantages of monothetic approaches are described by Kummamuru et al. (2004). Monothetic approaches are well suited for browsing and summarizing search results. However, monothetic approaches are not capable of describing the document collection/group in a well defined manner. This issue can be addressed by the polythetic approach by describing a group with multiple features. While STC is a monothetic clustering algorithm, Lingo is a polythetic algorithm (Mecca et al. 2007). The proposed approach also uses multiple features while deciding documents to assign hence is a polythetic approach.

4.2 Methodology

A search result usually includes title, URL, and snippet. We consider document as a combination of title and its snippet instead of considering complete text of document because most of the users would be unwilling to wait for complete page to download in order to view the results.

Figure 4.1 describes the whole system for topic set identification and labeling. In the proposed method ***concept selection module*** extracts concepts from top 100 results of a query (refer Chapter 2). Concepts are grouped to form topics intended for a query by the ***Topic set Identification module*** (see section 4.2.1). The set of topics of a query indicates the different intentions of a query. Each topic is represented by a set of concepts. ***Document assignment module*** classifies and assigns the documents under each topic (see section 4.2.2). We have labeled each topic with appropriate phrase using the ***topic labeling module*** (see Section 4.2.3) to get the feel of the topic in a glance.

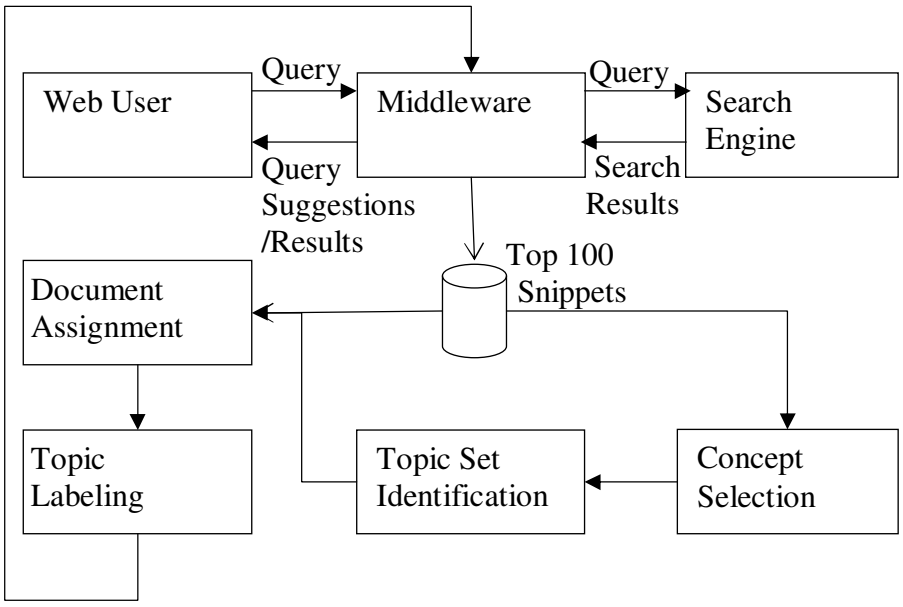


Figure 4.1. System Diagram

4.2.1 Topic Set Identification Module

In the proposed method query intentions are identified by grouping the extracted concepts. These intentions of a query form topic set for the query and each topic in the topic set is represented by a set of concepts.

As described earlier that we need a simple method for query disambiguation which is an online process. Therefore, we have used correlation among the concepts rather than other sophisticated

methods. We have formed the correlation matrix of the concepts obtained from snippets. The concepts which are not linked together (unrelated) are separated by partitioning the correlation matrix. These unrelated concepts have values lesser than the correlation threshold δ_C. Concepts in a partition will forms a topic. Each of the topics represents a different intention of the query. For example, for the query "apple", categories formed are "Fruit", "Apple Products" and "Tourism" etc. The detailed algorithm for the topic set identification of a query $\boldsymbol{q}$ is given below:

Step 1: Concepts similarity Matrix formulation

In this step, concepts similarity matrix (M) is formed. The similarity between two concepts c_i and c_j is calculated as follows:

$$\mathrm{sim}(c_i, c_j) = \frac{|c_i \cap c_j|}{|c_i| \cup |c_j| - |c_i \cap c_j|} \quad (4.1)$$

where $|\boldsymbol{c_i} \cap \boldsymbol{c_j}|$ is the number of documents contains both c_i and c_j and $|c|$ is the number of documents contains the concept *c*.

Step 2: Concept Grouping

In this step, concepts similarity matrix M obtained in the step 1 and correlation threshold δ_c have been given as input. Concepts in M are partitioned into the number of cohesive clusters by grouping the concepts with the correlation value greater than or equal to δ_c. As a result, each cluster will be represented by the set of related concepts representing a topic in the topic set of a query q. The algorithm for topic set formation is presented in Figure 4.2. We refer topic set of the query *q* as TS_q:$\{T_1, T_2, \ldots, T_n\}$ and a topic as T_i. This type of categorization has advantage that it does not require users to specify the number of categories well in advance. Further, this is a single pass process as it requires the data to be processed only once.

Input: Concepts similarity Matrix M, correlation threshold δ_c
Output: Topic Set TS_q ={T_i,T_j,….T_k} where, each topic is represented as set of concepts
Begin
//Initialization
C=0 // counter for new cluster formation
Concept_Set= [ ] // Cluster-Formulation
For j=1 to n // n is number of concepts
If c_j not in Concept_Set
then
C=C+1
Create new cluster with index C & add the concept c_j to it
Concept_Set = Concept_SetUc_j
End if
For k= (j+1) to n
If c_k not in Concept_Set
then
If similarity (c_j, c_k) >=δ_c
then
Add c_k to cluster C
Concept_Set = Concept_SetUc_k
End If
End if
End For
End For
End

Figure 4.2. Topic Set Identification Algorithm

4.2.2 Document Assignment

In this phase, we assign documents to a topic using a feature-centric document filter (F'). To initiate this phase, first we have selected a subset $D_{T_i}$of the documents that contain at least one of the concepts of the topic T_i. Documents of D_{T_i} are then filtered using document-topic relevance score. Document-topic relevance score is calculated as follows:

$$rel\ (d|T_i) = \frac{\sum_{w \in V^{(T_i)}} (\alpha * WF(T_i,w) * \frac{n_{wd}}{\sum_{k=1}^{z} td_k})}{|V^{(T_i)}|} \quad (4.2)$$

where $V^{(T_i)}$ is the set of concepts in a topic T_i, n_{wd} is the number of times concept w$\in T_i$ occurs in document d$\in D_{T_i}$, $\sum_{k=1}^{z} td_k$ is the total number of terms in document d. $WF(T_i, w)$ is the concept-topic relevance weight calculated based on D_{T_i} in place of top 100 snippets in Equation 2.1. Also, α is a factor used to include the importance of the concept based on its occurrence

either in the title or in the snippet. α is used because of the fact that most of the users would first try to skim through results using title. Hence, while assigning weight to document it is important to give more importance to concepts which occur in documents' title. This score is normalized using the number of concepts that belong to the topic. Either top-k documents or documents with the document-topic relevance score greater than threshold δ_A can be considered as the most relevant documents to the topic.

4.2.3 Topic Labeling

Importance of the terms is the key factor for considering a term as label and it is usually measured by its frequency as in STC. But, frequently occurring terms alone may not describe the topic well. In this work, we have measured the importance of the label using W1 measure which incorporates a number of factors (see Section 3.1). Moreover, the descriptiveness of the topic is also considered. That is label should contain the terms which are more specific to a topic as compared to other topics or the presence of the label in the document places high confidence that the document belongs to that topic.

In our work, candidate terms for labeling are concepts assigned to topic during concept categorization. On the other hand, documents being used for labeling are those assigned to the topic in document assignment phase. The Topic Label Score (*TLS*) for a *topic* (T_i) and *concept* (c_i) is computed based on following function:

$$\mathrm{TLS}\ (\mathrm{T_i}, \mathrm{c_i}) = \mathrm{WF}\ (\mathrm{T_i}, \mathrm{c_i}) * \frac{\mathrm{WF}\ (\mathrm{T_i}, \mathrm{c_i})}{\mathrm{WF}(\mathrm{q}, \mathrm{c_i})} \tag{4.3}$$

Where $WF\ (T_i, c_i)$ is the weight of concept (c_i) with respect to all the documents assigned to topic (T_i) and $WF(q, c_i)$ is the weight of the concept (c_i) in the query collection. Labels generated by $TLS\ (T_i, c_i)$ are both recurrent to the cluster and are definite to it i.e. incorporate both importance and descriptiveness. The concept with highest score acts as label for the topic while the other concepts are used as descriptors of the topic.

4.3 Experimental Setup

We have considered a subset of Open Directory Project (ODP) (ODP 2015) categories listed in Table 4.1 to evaluate the performance of the proposed approach. The method is tested against the STC and Lingo algorithms for individual and mixed ODP categories. The ODP categories are the largest human-edited directory of the web. It is maintained by experts of different domains.

Table 4.2 contains the details of mixing the ODP categories with similar and different intentions for experiments.

Table 4.1. ODP Categories for Evaluation

ID	Category	Content	# of Docs
G1	Blade Runner	Information about Blade Runner	16
G2	Lord of the Rings Series	Information about Lord of the Rings	19
G3	Java/FAQs Help and Tutorials/Tutorials	Java Tutorials	29
G4	DataWarehousing /Articles	Data Warehousing Articles	19
G5	MySQL	MySQL Database	12
G6	PostgreSQL	PostgreSQL Database	30
G7	XML Proprietary	Native XML Database	4
G8	Editors/Vi	Vi text Editor	15

Table 4.2. Merged Test Sets

Merged Groups	Test Set Rationale
G1+G2	Ability to separate two related categories
G1+G4	Ability to separate two unrelated categories
G4+G5+G6	Ability to Separate three categories with common parent intention
G1+G4+G8	Ability to separate three categories with different parent intention
G1+G4+G8+G7	Ability to separate four unrelated categories on significantly smaller (XML) than the rest
G1+G2+G3+G4+G5+G6+G7+G8	Ability to generalize categories (into Movies, databases, editor, java)

We have also evaluated the performance of the proposed method for ambiguous queries using Ambient (**Ambi**guous **Ent**ries) Dataset (Carpineto and Romano 2008). The dataset consists of 44 ambiguous queries selected based on Wikipedia disambiguation pages in 2008. Each query is assigned with 100 documents retrieved from a search engine. Subtopics for each query along with its documents are also presented in the dataset. In our experiments, we have used documents instead of top 100 snippets for each query in the dataset to verify the proposed approach capability in terms of topic set identification and document assignment under each topic.

We have used Carrot2 workbench-version 3.9.3. (Carrot2 2015) for getting results of STC and Lingo. In our experiments, α = 0.6 and 0.4 when concept occurs in the document's title and only in snippet respectively. δ_A and δ_C are 0.005 and 0.25 respectively. These values of thresholds are assigned empirically from the experimental analysis.

4.4 Experimental Results

This section presents the experiments conducted to advocate the following: How well the proposed approach is able to separate mixed ODP categories and their documents?, How well the proposed approach is able to handle ambiguous queries? and How well the proposed approach is able to assign labels appropriately?

Table 4.3 shows the generated topics for a few ODP categories. It can be observed that the proposed approach identifies topics that are better than Lingo and STC.

We observe that the topics in Lingo such as {*"Fan Fiction"* and *"Focused"*} with the same intention *"Focused Fan Fiction"* should have been merged into a single topic but are not able to merge in G1. Corresponding topic for the proposed algorithm is "*Focused Fan Site"*. Similarly, the topics { "*News*", "*Message*"} with the intention of *Lord of the Rings latest news* in G2, { "*Example*", "*Lesson*"} with the intention of *Java tutorial example lessons* in G3 and {"*Modeling*", "*Concepts*", "*Successful*"} with the intention of *successful data modeling* in G4 are similar (as the documents under these topics are similar intention documents). Our approach forms the corresponding topics as "*Rings news*", "*java tutorials*" and "*Successful modeling*" in G2, G3 and G4 respectively. The topics {*"Section"* and *"Links"*} generated by the Lingo are broad intention topics having a similar set of documents in each topic. The proposed approach generates a topic "Site" in place of the two topics *"Section"* and *"Links"*. Both the topics {"contains" and "Content"} in Lingo refer blade runner brand content. The proposed approach identifies the similarity among these two topics and forms a single topic {"Content"} in place of two different duplicate topics. Lingo is forming the topics {"interview", "awards"} to have documents related to interviews of various cast of movie blade runner & interviews published in Blade Runner Magazine and documents related to awards of blade runner movie respectively. The proposed approach adds these documents under the topic "Blade Runner" by having interview and awards as concepts under the topic. Whereas, STC forms the topic "Cast" in place of both the topics {"interview", "awards"}. The topic "Location" and "Blade runner Resource" which refers documents related to blade runner movie location and documents related to various resources like multimedia, essays, visionary films, etc. related to blade runner movie and game respectively is formed as a separate topics in the proposed approach which are not captured by Lingo and STC. The topic "Film" in Lingo is same as the topic "Movie" formed by the proposed approach. STC forms two different topics "film" and "movie" in place of one single topic

formation due to the reason that the phrases movie and film are frequent but chances of co-occurrence of these phases are less. This is also visible from the table that the number of topics formed by STC is less as compared to Lingo and the proposed approach.

Tables 4.4 and 4.5 respectively depict the results obtained on mixing documents of categories with similar (G1+G2) and distinct (G1+G4) broad intentions. It can be seen that most of the topics under categories G1+G2 and G1+G4 are identified by the proposed approach. It should also be noted that Lingo is not taking care of the semantic relationship between two categories due to lack of common co-occurred concepts between their documents. Thus, term document matrix in lingo returns 0 which is not able to satisfy lingo's snippet assignment threshold and hence not able to generate one topic. For example, "*Managing*" and "*Business Intelligence*" in Lingo are separate. The proposed approach is able to generate topic "*Quality Management*" which contains the following concepts including *business intelligence* and *management*:{"*business performance management*", "*data warehousing and data quality content*", "*industry based business intelligence*", "*data warehouse quality management*", "*Laura Hadley*", "*data warehouse environment*", "*ideas*", "*possible metrics*", "*warehouse value*"}.

Table 4.3. Labels of Topics for the categories G1 to G4

G1	**Lingo**	Blade Runner, Film, Focused, Interviews, Music, Awards, Content, Contains, Fan Fiction, Links, Section
	STC	Blade Runner, Movie, Cast, Film, Game
	Proposed Approach	Blade Runner, Site, Content, Movie, Blade Runner Resource, Location, Music, Book, Focused Fan Site
G2	**Lingo**	Cast and Crew, Discussion, Collection, Forum, Analysis, BBC, Download, Fan, Locations, Lord of the Rings News, Message Board, Pictures, Video
	STC	Lord of the Rings, Films, Movie, News, Image, Cast and Crew, Gallery, Links
	Proposed Approach	Lord Rings, Cast and Crew, Collection, Films, Image Gallery, BBC, Rings News, Locations, Movie, News, Discussion
G3	**Lingo**	Java Tutorials, Applications, Example, Java Code, Collection, Java and Web Development, Open Source, Automatically, Directory, Google Web Toolkit, Java Beans, Lesson, Package, Platforms, Process, Software Developers
	STC	Java, Tutorials, Developers, Examples, Applications, Code, Open Source, Google Web Toolkit
	Proposed Approach	Java, Development, code, Online Java Tutorials, Google Web Toolkit, command line, collection, Tutorial, Web, Affine transform lesson, haki, Ant, Web Services Messaging, databases, Layout Management
G4	**Lingo**	Data Warehousing, Managing, Business Intelligence, Concepts, Developing, Enterprise, Inmon, Modeling, Quality, Ralph Kimball, Review, Successful
	STC	Data Warehouse, Data Warehousing, Article, Management, Data Warehousing and Data
	Proposed Approach	Data Warehousing, Magazine, Ralph Kimball, Quality Management, Big Data, Data Warehouse Architecture, Successful modeling, Principles, Intelligent Enterprise Magazine, Bill Inmon Articles, Warehouse Assessment

Table 4.4. Topics created by different algorithms for merged categories G1+G2

Lingo	Blade Runner, Cast and Crew, News, Discussion, Image Galleries, Includes, Download, Fan Fiction, Focused, Locations, Trivia, BBC, Contains, Fan Club, Lord of the Rings News, Soundtrack, Trailers
STC	Lord of the Rings, Blade Runner, Cast and Crew, Film, Movie, Galleries ; Image Galleries, Fan Fiction, Images, Links, Discussion, Interviews, Articles, Game, Reviews
Proposed Approach	Blade Runner, Movie, Film, Cast, Game, Interviews, News, Trivia, Music, Image gallery, Lord of the Rings News, Analysis, Trailers, Site, Location, Discussion, Content, ring, Films spiritual Connections

Table 4.5. Topics created by different algorithms for merged categories G1+G4

Lingo	Blade Runner, Data Warehouse, Movie, Data Warehousing, Film, Managing, Focused, Business Intelligence, Concepts, Contains, Database, Enterprise, Fan Fiction, Inmon, Modeling, Ralph Kimball
STC	Blade Runner, Article, Data Warehouse, Data Warehousing, Movie, Management, Cast, Film, Game, Data Warehousing and Data
Proposed Approach	Movie, Blade Runner, Film, Cast, Data Warehouse, Data Warehousing, Content, Game, Site, Music, Quotes, Ralph Kimball, Quality Management, Data Mining Concepts, Location, Inmon, Awards

Table 4.6 depicts the sample results obtained using Ambient dataset and shows the capability of identifying the different intentions of ambiguous queries. We report the topics that contain at least three documents and ambiguous labels are highlighted in bold.

Consider the topic "*Issues*" from Lingo as a result of query *AIDA*. It assigns documents relating to the following intentions: {**I1)** AIDA issues new regulations over Colombian Spraying in Ecuadorian border region; **I2)** Issue of AIDA Mail and Newsletter; **I3)** Aida Pharmaceuticals chairman issues a letter to Shareholders Health}. As can be observed this problem needs to be resolved. On the other hand, STC is not able to form any topic relating to these documents due to lack of frequent phrases between them and hence places them in "Other Topics". In the proposed approach such documents can never be grouped together due to the underlying concepts in the topics. For the sake of brevity, in the proposed approach document with intention I1 goes in topic "*Colombian spraying*", I2 goes in "*association internationale*", I3 goes in "*Aida Pharmaceuticals*".

Since STC is based on recurring frequent phrases, smaller clusters are usually deemed insignificant in favor of larger clusters thereby leading to loss of topics. For example, in query "*Fahrenheit*" STC's result contains two topics "*32; Boiling; Temperature Scale*" and "*Scale; Temperature*" that indicate same intention with Fahrenheit as a scale of Temperature. These

topics could not be merged due to lack of documents containing these terms together. Even if merged and formed a large cluster that would not allow STC to form a cluster with label "*Celsius*" or "*Conversion*" which may be more illustrative to the users. Similarly, due to the lack of frequent phrases STC fails to identify important clusters like {"*Indigo books*", "*Shoes*"} for query "*Indigo*". Such topics are generated by both Lingo and the proposed approach.

Table 4.6. Sample Queries from Ambient Dataset and topics formed by different algorithms

AIDA	**Lingo**	Opera Aida, Giuseppe Verdi, Broadway Musical, Elton John and Tim Rice's Aida, **International**, **Classic**, Freeware Diabetic Software Simulation, Aida Association, Analysis, **English**, **Issues**, Leonyte Price, **Review**, Set in Ancient Egypt
	STC	Blood Glucose; Diet; Glucose Insulin Action, Elton John; Tim Rice's Aida, Giuseppe Verdi, Music, Verdi, Set; Set in Ancient Egypt, **Love**, Opera, Leonyte Prince, Nubian Princess Stolen, Love Story, Broadway, **Performance**, Princess
	Proposed Approach	Aida, Elton John, Opera, **Love**, Slave, Glucose, **Movies**, Princess, Feelings, Leonyte Price, Nubian Princess, Tim Rice, Verdi, association internationale, Fabric, Colombian spraying, Aida Pharmaceuticals, **Story**, Moviefone, **Archive**
Fahrenheit	**Lingo**	Fahrenheit 9/11, **Fahrenheit review**, Michael Moore, Indigo Prophecy, Temperature Scale, Fahrenheit 451 by Ray Bradbury, Fahrenheit 911, Fahrenheit 9/11 Trailer, Fahrenheit and Celsius, Adventure Game, Conversion, Quantic Dream, Terms, Fahrenheit Cologne by Christian Dior, Fahrenheit for PS2
	STC	Fahrenheit 9/11; Michael Moore, 32; Boiling; Temperature Scale, Indigo Prophecy, Fahrenheit 451; Ray Bradbury, Film; Fahrenheit 911, Number; Box Office, Game, Scale; Temperature, **Review**, Christian Dior, Preview, **News**, **Trailer**, **Movie**
	Proposed Approach	Fahrenheit 9/11, Fahrenheit 911, Indigo Prophecy, Scale, Game, Review, Previews, **Trailer**, **George**, Fahrenheit 451, Celsius, **Movie**, Christian Dior, **Bush**, Biotest Fahrenheit, **Definition**, Web Design, USAToday, Metabolism Breakthrough
Indigo	**Lingo**	Indigo Children, Indigo Music, Indigo Dyes, Color, Indigo Blue, Microsoft Indigo, India, Specialized, Indigo girls, Indigo Home Automation, Mixing Indigo, Indigo Clothing, Indigo Restaurant, **Internet**, **New Products**, Windows, Shoes for Woman
	STC	Indigo Children, Indigo Music, Systems, Dyes, Indigo girls, Software, **News**, Photos, **Services**, Blue, Microsoft, **Movie**, Natural, Family, Videos
	Proposed Approach	Music, Children, Shoes, Macintosh, Books, **Products**, **Questions**, Microsoft, **News**, Dyes, Plant, Folk Art, Indigo girls, imdb, Color, Clothing, Indigo Renderer

Since Lingo requires the terms in its label to be present in all the documents belonging to the topic it misses a few documents that should have been assigned to it. For example Lingo misses

the following two documents in topic "*Fahrenheit 451 by Ray Bradbury*": 1) IMDB page of Fahrenheit 451 movie (1966). 2) Sciflicks guide on Fahrenheit 451 movie collectibles. On the other hand, the proposed approach which uses the following concepts : *{Ray Bradbury, students, author, Fahrenheit 451 essays, edited, internal editing staff, book Fahrenheit 451, plot, author Ray Bradbury, insightful study}* assigns these pages under a topic "*Fahrenheit 451"*.

Another advantage of the proposed approach is that the user doesn't need to visit the set of documents if the topic is not clear through its label. For example, in Table 4.6 for a query "*indigo*" a user may be interested in "*wisefeet.com*" with the intention of a site for shoe recommendation. Both the approaches Lingo and STC bring up documents related to *"wisefeet.com"* under the topic "Shoes for Woman" and "Other Topics" respectively. So, user has to pickup the similar topics and go through all the underlying documents. The proposed approach is able to bring up this feature through the use of concepts. For example, the concept *wisefeet.com* under the topic "Shoes" guide user to find the appropriate documents easily and quickly. Not only such a feature allows user to view more number of documents in an organized manner, but also allows them to rephrase their query in order to fetch more intended documents.

Figure 4.3 presents Mean Average Precision (MAP) (Everitt et al. 2001) of STC, Lingo and the proposed approach for 15 sample queries from Ambient dataset. It can be observed in Figure 4.3, the proposed approach assigns documents to the topics with higher MAP as that of Lingo and STC.

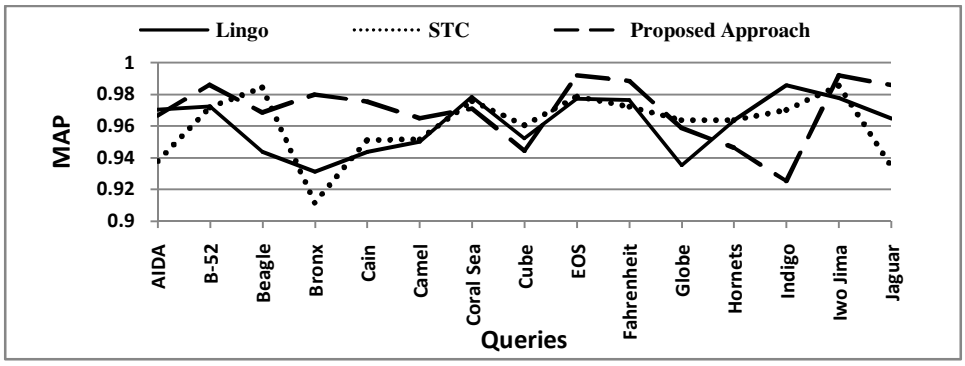


Figure 4.3. MAP for Document Assignment

Lingo algorithm precision is low due to the reason that it forms labels first by combining different terms and then assigns document based on term occurrence in the snippet. For example, the query *"Cain"* related topic "Helena Cain": intention of fictional character *"Helena Cain"* from Television Series *"Battlestar"*. But due to the above mentioned problem it also assigns

document related to *"Cain Vineyard located in St. Helena"*, which is different from the main intention of the topic. Another example is for the query *"Beagle"* with a topic *"Beagle photos"*: intention of the documents is photos of Dog Breed Beagle. However, Lingo includes documents related to photo tool in Beagle Software, which has significant differences from the main topic. On the other hand, the proposed approach solves these problems using concepts. For example, topic *"Helena Cain"* for query *"Cain"contains {Helena Cain, TOS, Battlestar wiki, series counterpart, original series character, cylons, Admiral Helena Cain, Molecay, fictional Character, reimagined science fiction}* so the intention of the topic is clear to the user. Moreover, if new documents related to the topic (for example, *Molecay*) come up they will be assigned here even if they do not contain "*Helena Cain"* directly. Such assignment is not possible in Lingo.

STC also suffers from the same problem. Consider the example query *"Cube"* with the topic *"Power Mac G4 Cube"* and intention of Macintosh Personal Computer from Apple Inc. The topic contains an irrelevant document related to *"Coherent Inc.: Cube"*, which is a laser diode system and the snippet contains term laser *"Power"*. The proposed approach is able to bring up this intention in the form of *G4 Cube* as topic which contains the following concepts:*{G4 Cube, Mac, Power, Apple, Power Mac G4 Cube, History, Company Computing World, Dramatic New Case Design, Apple Power Mac G4 Cube, Apple G4 Cube, Power Mac G4 Cube pricing and specifications, Steve Jobs, Address}* which completely clears the intention to users and assigns related documents.

It is also observed that STC is not able to form the topic with the label *"Helena Cain"*, because, the term *"Helena"* occurs in three documents while *"Cain"* occurs in ninety six documents which leads to a low similarity ratio according to their method and thus the phrase *"Helena Cain"* is not able to cross the threshold. In STC, documents which contain the term *"Helena"* is placed in the topic labeled *"Other Topics"* which is nothing but an indiscriminate documents collection.

It is also noted that in Figure 4.3, STC's higher precision in document assignment for a few queries is due to the formation of redundant groups. For example, for query *"Cain"* STC generates topics *{Abel, Brother, Brother Abel, Cain and Abel}:* intention of all these topics is Abel *("Second son of Adam and Eve and Brother of Cain")*. Similarly, consider the query *"Cube"* and the associated topics *{Person Shooter; Multiplayer, FPS; Indoor FPS Engine, IGN is the ultimate}* having the common intention *"IGN"*, which is a First Person Shooter (FPS)

Multiplayer game. Therefore all the topics in the above examples contain similar set of documents. Thus, even though documents are being correctly assigned which in turn leads to increase in MAP, such redundant topics are not required to the user.

The decrease in precision for the proposed approach in a few cases is due to inadequate length of the snippets. Thus, unrelated documents are able to have more weightage. Another reason for decrease in precision is the high weight of a few concepts which allows irrelevant documents to cross the threshold. For example, in query *"Coral Sea"* and the associated topic *"War",* concepts are *{War, World, Coast, World War ii, Nature, Region, Major Naval and Air Engagement, Battles, World War ii wreckage, south pacific, Namesake chain}*.Clearly this is about *World war ii*, but due to high weight of concept *"world"*, documents related to "*scuba diving in world's exotic destination*" is able to cross threshold and hence decrease precision. However, each concept is associated with the related documents in our approach. This helps user to retrieve the related documents by clicking on the respective concepts.

4.5 GUI: Visualization of Topic Set and the related search results

In the last step of the method the topic set along with their concepts are presented to the users as query suggestions. This interface allows users to search with the knowledge domain terminology which will help search engine to get the desired results. The screen shot of the developed GUI for the proposed approach is presented in Figure 4.4. The screen shot depicts the results of the query "Fahrenheit". The following Figure 4.5 shows the topic set and the respective grouped documents for the query "gunpowder" (as of June 2014). This figure 4.5 shows documents related to the selected topic "Gunpowder Plot of 1605".

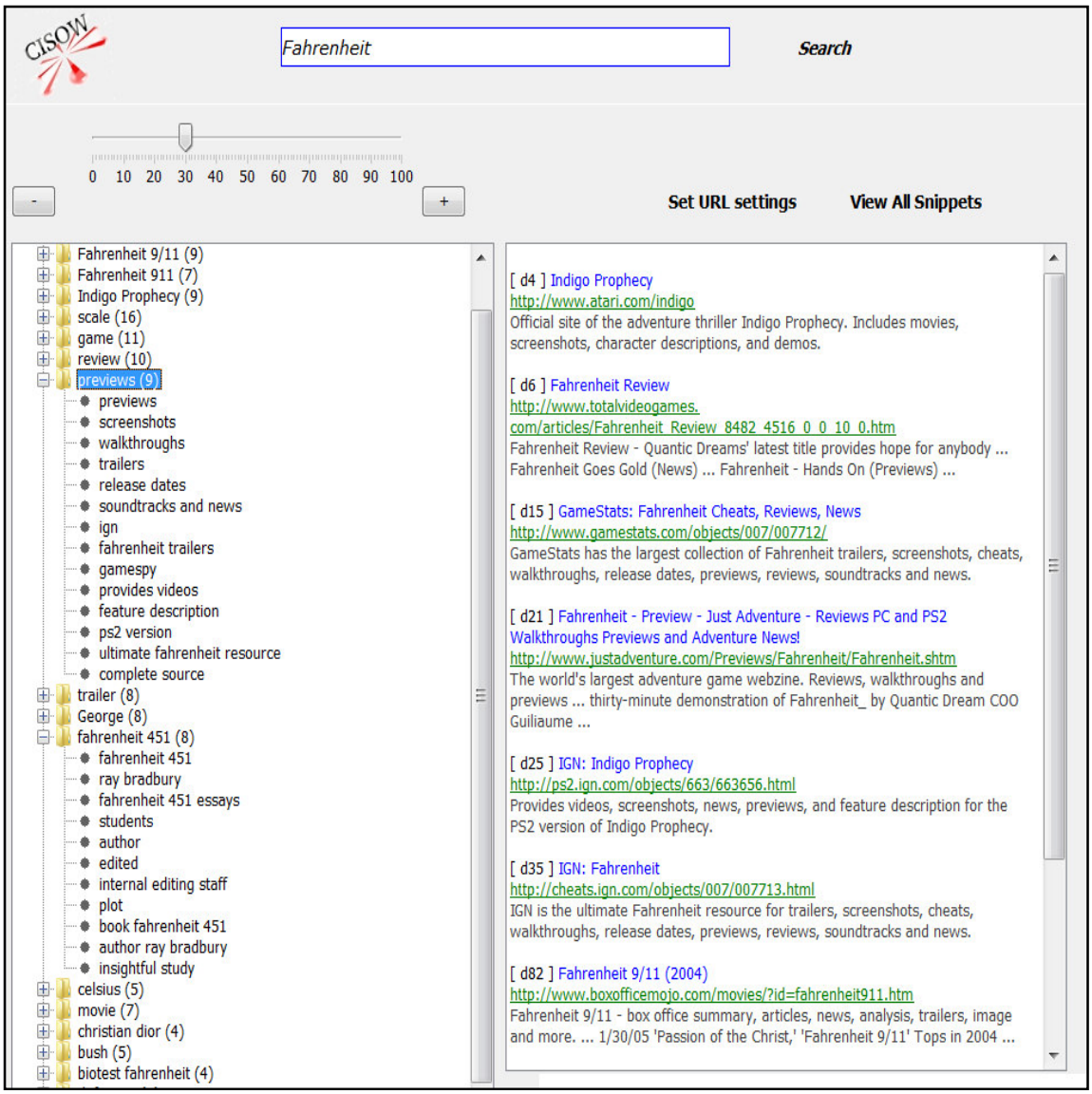


Figure 4.4. Screenshot of the proposed approach GUI for the query "Fahrenheit"

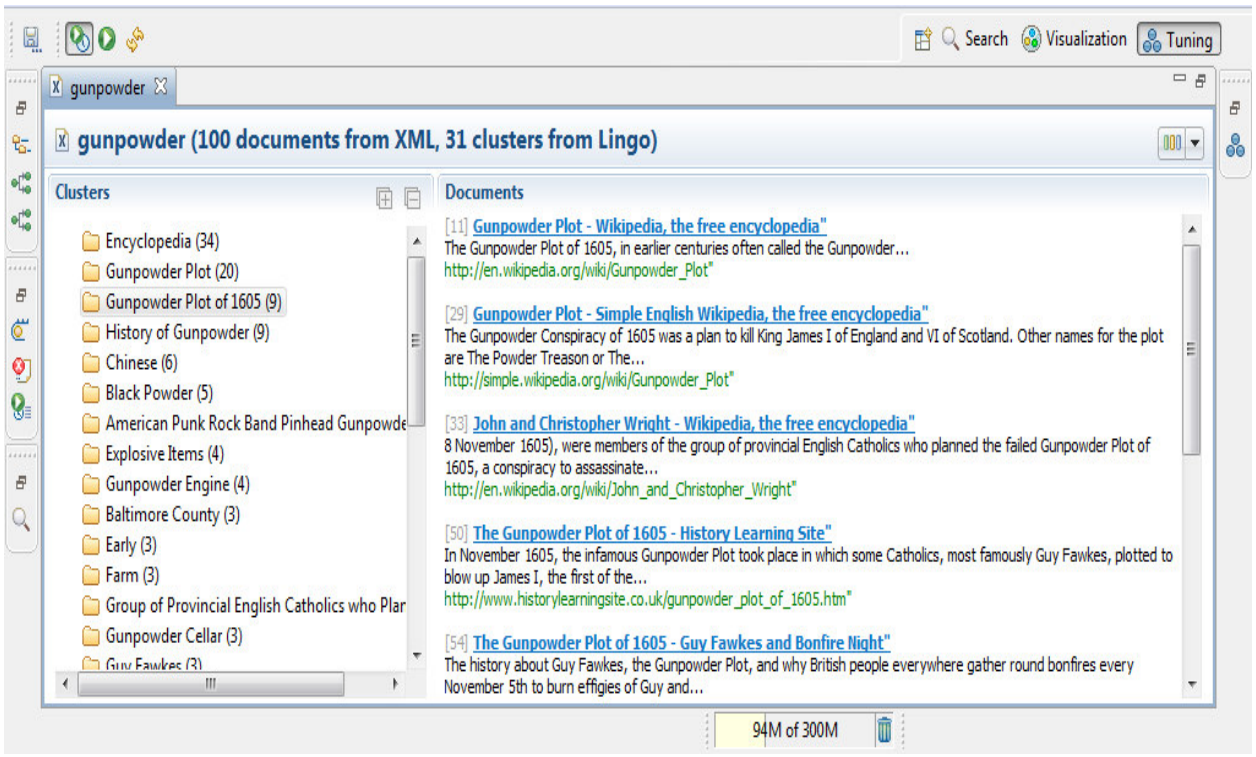


Figure 4.5. a) Screenshot Based on Lingo Algorithm GUI for the query "gunpowder"

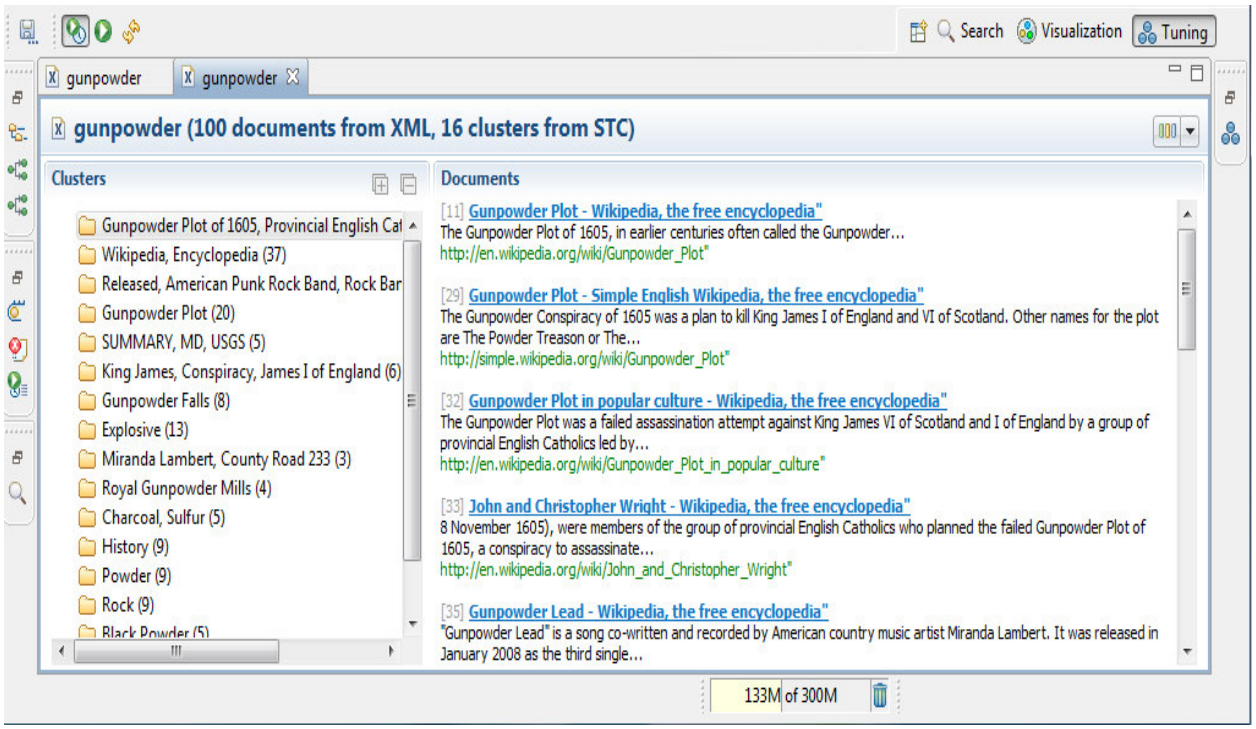


Figure 4.5. b) Screenshot Based on STC Algorithm GUI for the query "gunpowder"

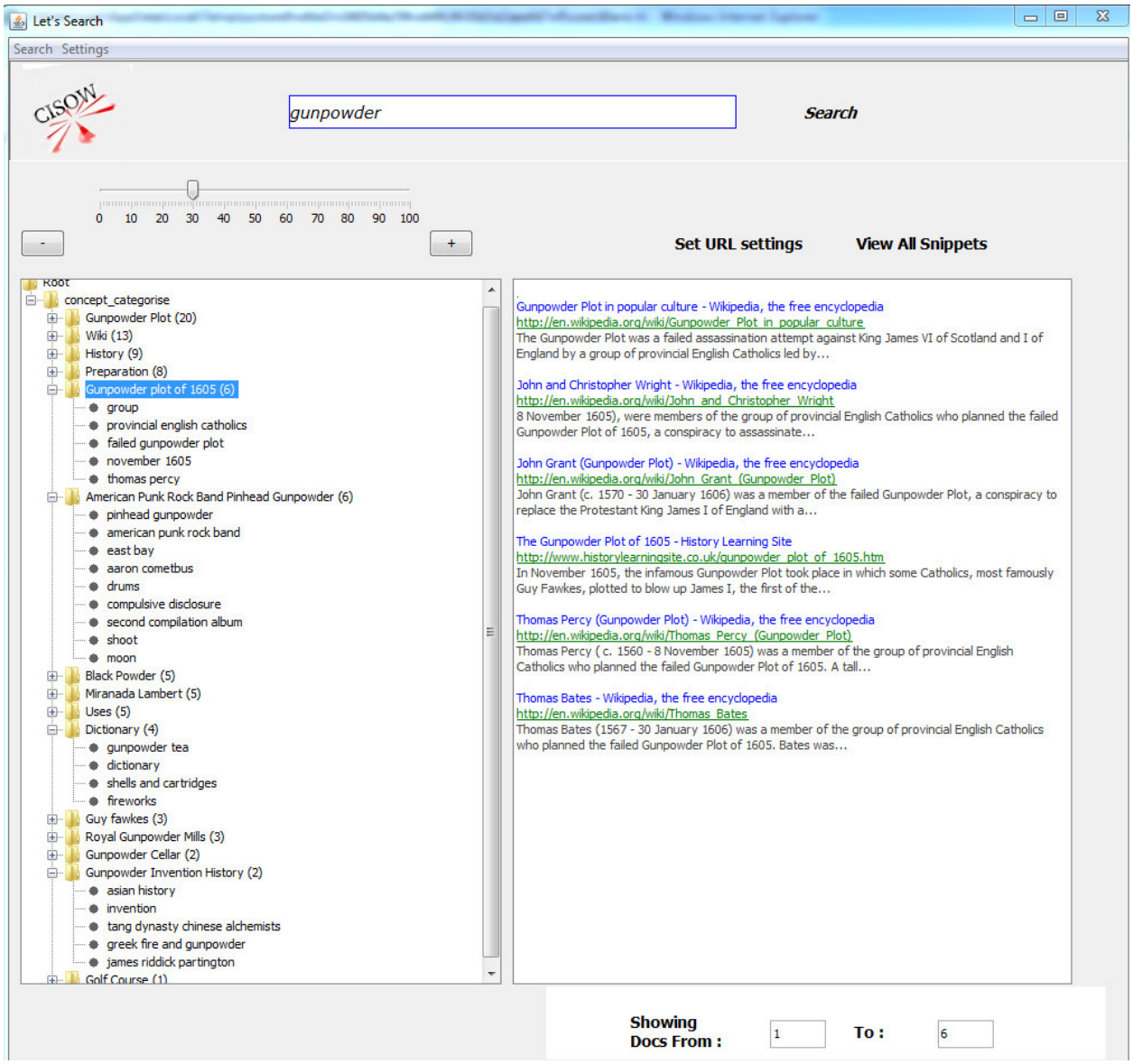


Figure 4.5. c) Screenshot Based on the Proposed Method GUI for the query "gunpowder"

This is clear from the Figure 4.5a and 4.5b that both STC and Lingo provide automatic topic labels. Instead, our method (see Figure 4.5c) suggests concept phrases as descriptors of the topics along with the label. In our approach, concept under the specific topic can be selected to view the respective concept related documents alone. As well as from the developed GUI, it is visible that users' can vary the correlation threshold according to their requirement. The correlation

threshold δ_C can be adjusted manually i.e. increasing and reducing δ_C allows us to get fine tuned categories and to get less number of categories respectively.

4.6 Conclusions

In this chapter, an approach that identifies various topics intended for a query and provides a concept representation for each topic has been proposed. It also assigns the relevant documents and labels to these topics so that users can understand the search results clearly and can use the associated concepts effectively to rephrase or enhance the query. The proposed method is evaluated against the performance of the well-known algorithms of search result clustering. It is clear from the results that the proposed approach produces encouraging results when applied on the standard datasets. The performance of the proposed algorithm is considerably good for all the modules (topic set identification, document assignment and labeling).

The proposed approach can be extended for personalized topic set identification. The proposed approach can be modified to be suitable for offline requirements by incorporating semantic and domain knowledge.

CHAPTER 05

Topical Document Clustering: Two-stage Post Processing Technique

Text document clustering is an essential means to organize a large amount of information into a small number of significant clusters to overcome the problem of low precision of the web search results. This allows users to find the necessary information easily. Effectiveness and efficiency of the results can be further improved if clustering is done by considering topics/theme of the documents. These topic based document clusters are referred as topical document clusters.

Approaches that identify topic(s) of the documents can be broadly classified into two categories: document-pivot where topic is represented by a cluster of documents formed by measuring some kind of similarities among the documents and feature-pivot where selected keyword/features are clustered based on their co-occurrences and topics are evolved from these clusters. Sometimes topics are detected through the distribution of the terms/features and it is known as a probabilistic topic model. These methods have their own pros and cons. The document-pivot approach may cluster two documents which are related to two different topics into one cluster. This cluster may either be high level (broad) topic of the two different low level (specific) topics, or may be mixed type different topics which are not related. The first one is acceptable, whereas the second is undesirable (Petkos et al. 2014a). Pair-wise co-occurrences may produce mixtures of topics rather than fine-grained topics in feature-pivot approach. However, one can take a larger subset of features to get fine-grained topics (Petkos et al. 2014b). In this chapter, we propose an idea to assign a document to cluster(s) with the additional query information which relates query with the documents during clustering process. This facilitates us to group documents based on query's conceptual meaning. Few algorithms such as Query-Document (QD) (Beeferman and Berger 2000), Query-Concept (QC) (Leung et al. 2008) and Query-Document-Concept (QDC) (Goyal et al. 2013) are available in the literature which cluster queries. These algorithms use concepts which are topic descriptors that are selected keywords from the documents clicked for the queries. These concepts are then linked with the queries and clustered based on the clicks. These approaches produce query clusters, associated document clusters and associated concept clusters (It is to be noted that concept cluster is also known as topical cluster). The obtained topical clusters are basically different thematic groups. For example, consider the queries "apple" and "apple fruit". Search result of these queries has

documents related to the topic apple fruit. This method will improve the homogeneity of the clusters inspired by the context of the query issued by the user.

Beefferman and Berger (2000) suggested a technique, Query-Document (QD), for query clustering based on the common clicks on the same pages. In this approach, the similarity between queries and between documents is found using users feedback i.e. click-through information. However, QD algorithm does not incorporate the content of the documents. Thus, affected by noisy clicks and less number of common clicked documents. The approaches QC and QDC consider both the content and the clicked information for query clustering. These approaches also produce document clusters and associated topical clusters. These topical clusters intend to have concepts related to a single topic and therefore respective document clusters produced are also expected to be pure. Normally, the quality of these topical and document clusters suffer because of two reasons: First, ambiguous query, noisy clicks, multi-topical documents, etc. (ambiguous and noisy input). Second, most of the traditional clustering algorithms produce hard clusters. An IR system should be more capable to relate a document/query with all its appropriate topics, rather than relating it with only a primary topic. This in turn leads to the requirement of soft clustering. The chapter presents a post processing technique (split-merge) applicable to the query-context aware clusters for soft clustering. It uses topic segmentation by disambiguating web search results, i.e. topics are formed by a feature-pivot approach that groups features which are extracted from the search results. This approach improves quality of the clusters and helps to overcome the problem of non-homogeneous and hard topical clusters. The proposed SM algorithm can be applied to the results of any query based clustering algorithm, which produce query clusters, associated document clusters and associated topical clusters such as QC and QDC. The algorithms that do not produce topical clusters such as QD, documents and queries can be processed to obtain topical clusters. In place of different clustering algorithms, here after we call them as algorithm A (i.e. A may refer to one of QD, QC and QDC).

We first illustrate an example which shows the requirement of the proposed approach and then we present the summary of the work done.

Example

Consider a simple example which has 12 queries, 24 associated clicked documents and concepts extracted for each clicked document (see Table 5.1). We have only presented sample concepts for each of the documents in Table 5.1 for clarity and the resultant clusters shown in Figure 5.3, 5.5 & 5.7 have sample concepts. This is the input to the clustering algorithm. The output of the algorithm is the input to our proposed approach. The result set consists of seven topical clusters (C1, C2,…, C7). Partial result is given in Figure 5.1. There can be more than one query clusters (QC' and QC'') associated with a single topical cluster (C1) as shown in Figure 5.1. The result is taken from QDC algorithm. The formation of C1 is due to same name concepts from documents in DC1' and DC1" rather than the related intensions. For example, the concepts 'job', 'interview' and 'questions' (Topic: programming language knowledge based job) are merged with the concepts 'Rod Johnson', 'comments' and 'views' (Topic: football player Rod Johnson interviews and comments) due to the common concepts like 'Rod Johnson', 'interview' etc. (see Figure 5.1) (It is to be noted that explicit topic labels are given to the clusters manually). This unrelated merging of concepts can be due to the following reasons: documents clicked for a query belong to more than one topic and/or noisy clicked documents (documents clicked on different topics/intension of the query).

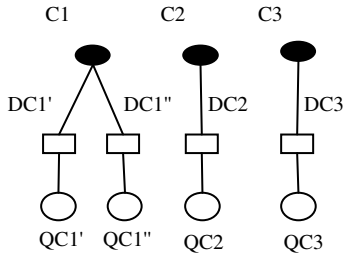


Figure 5.1. Sample Output of QDC Clustering Algorithm

From the Figure 5.1, it is also visible that related concepts did not merge in some cases. For example, the concept "Rod Johnson" is merged only with the concepts related to the query "football player". However, the concept "Rod Johnson" is also related to the query "spring" (Rod Johnson developed spring framework). This is because the concept "Rod Johnson" is ambiguous and clustering algorithm produces only hard clusters. Therefore, there is a need to post process the results of algorithm A to resolve the above issues.

Table 5.1. Sample Queries, clicked documents and Concepts in documents of the example

Queries	Programming Language (q1), code (q2), spring (q3), framework journal (q4), football player (q5), gunpowder wiki (q6), wikipedia black powder (q7), inventor of gunpowder (q8), dooney and bourke imitations (q9), dooney and bourke replicas (q10), American Girl dolls cost $ (q11), American girl doll retail price (q12)
Documents' and their Concepts	d1: language (c1), java (c3), source code (c5) d2: beginner (c10), java (c3), tutorial (c4) d3: java (c3), source code (c5), samples (c6), open source project (c9) d4: temperate (c21), season (c22), ideas (c23), spring (c12) d5: MVC (c14), android (c11), web application (c15), spring (c12) d6: MVC (c14), web application (c15), spring (c12), framework (c13), java (c3), tutorial (c4), development (c16) d7: java (c3), spring (c12), interview (c25), job (c26), questions (c27) d8: beginner (c10), tutorial (c4), code (c7) d9: sample code (c2), tutorial (c4), code (c7), programmer (c8) d10: java (c3), spring (c12), tutorial (c4), code (c7), framework (c13) d11: java (c3), framework (c13), Rod Johnson (c24) d12:spring (c12), seaters (c17), estate (c18), saloon (c19), taxi (c20) d13: framework journal (c30), framework (c13), cinema (c31), media (c32) d14: java (c3), spring (c12), tutorial (c4), code (c7), framework (c13) d15: interview (c25), Rod Johnson (c24) d16: Rod Johnson (c24), comments (c28), views (c29) d17:sale (c69), Dooney & Bourke heart bag (c70), tote style (c71), Bourke replica (c72), dimensions (c73) d18: gunpowder (c33), term black powder (c34), corning first (c35), fine black powder meal (c36), blocks (c37) d19: gunpowder (c33), encyclopedia article (c47), citizendium (c48), generic term (c49), black powder and smokeless powder (c50), smokeless powder (c51), nitrocellulose (c52), gun-cotton (c53) d20:history (c38), gunpowder (c33), earliest (c39), formula (c40), chinese Wujing Zongyao (c41), four great inventions (c42), ancient china (c43) d21: gunpowder (c33), online information article (c44), arms (c45), inventor (c46) d22: fashion accessories information business (c74), Dooney & Bourke (c75), designer (c76), leather handbags (c77), gloves (c78), organizer (c79), cell phone (c80), manufacturing buttons (c81), buckles (c82), imitation jewelry (c83) d23: ebay guides (c54), american girl dolls buying guide (c55), american girl dolls (c56) d24: doll market (c57), your favorite dolls (c58), gifted doll artists and collections (c59), Madame Alexander dolls (c60), cissy (c61), doll (c62), mattel barbie dolls (c63), american girl type dolls and outfits (c64), dollzone dolls (c65), pullip dolls (c66), lee middleton dolls (c67), retail price (c68)
Queries and their Clicked Documents information	q1({d1,d2,d3}:java language tutorial for the beginners:,{d4}:spring season:,{d5}:java spring framework:),q2({d6}:java spring web application tutorial:{d7}:java language based jobs and interview questions:,{d8}:java language tutorial for the beginners:{d9}:java code:),q3({d4}:spring season:,{d10,d11}:java spring :,{d12}:spring taxi:),q4({d13}:framework journal:, {d14}:java spring framework:),q5({d15,d16}:Rod Johnson:),q6({d19}:gunpowder encyclopedia:),q7({d18}:blackpowder:),q8({d20,d21}:gunpowder history:),q9({d22}:dooney & bourke fashion accessories:),q10({d17}:dooney & bourke bag sale:),q11({d23}:american girl dolls buying guide:),q12({d24}:american girl doll types:)

We first define a few factors before we present key points of our work.

- **Cluster Homogeneity:** Degree of similarity among elements in a cluster.
- **Multi-topical Document:** Multi-topical document is a document which belongs to more than one topic. For example, news article that covers sports and politics.
- **Topic Overlap:** Degree of similarity among the topical clusters.
- **Topic Set (TS) for a query:** Set of categories/topics, a query may intend to. Each topic is portrayed by a set of concepts found by disambiguating search results. (Refer Chapter 4).
- **Topic Segmentation:** Process of dividing a given topical cluster into a number of clusters using topic disambiguation.
- **Topic disambiguation**: Process of resolving conflicts that arise when a single topic has more than one intention.
- **Click-through Log**: It is a repository of feedback on the web user's behaviour.

The chapter is summarized as follows:

1. This work exploits the possibility of using existing query clustering algorithms for producing homogeneous soft topical document clusters.
2. A Split-Merge (SM) post processing technique is proposed to produce topical soft document clusters. The SM algorithm can be applied on the result of document clusters which has associated query clusters and concept clusters (optional).

    - Two factors, cluster homogeneity and multi-topical document characteristics decide whether a cluster can split or not. The homogeneity criteria dilutes if a topical cluster (obtained from hard clustering) has multi-topical documents or concepts. This manifest by two factors: noisy clicks and query ambiguity. In our approach, a cluster can split into multiple small clusters by forming coherent soft clusters through topic segmentation using topic set. This intra cluster split results in a number of homogeneous and soft clusters.
    - The inter cluster merging is used to merge related clusters in merging step of the proposed algorithm. Topic overlap is used to merge the clusters.

- The Performance of the proposed approach can be tuned by varying two thresholds – split-threshold & merge-threshold- to suit the application requirements, i.e. applications may require high-precision or high-recall or both. The lower the split-threshold, higher the precision. The lower the merge-threshold, higher the recall. For example, applications like spam filtering in email management system and Right to Information Act requires high precision. Applications like People finder in social network and Potential buyers' group discovery in marketing e.g. Amazon requires a higher recall. Applications like search engine, e.g. Google, Bing and Yahoo, etc. require both high precision and high recall.

3. Ambiguity often defeats the classical clustering approaches. But our approach is capable of handling ambiguity even though it is applied on snippets (short text).
4. Our experimental evaluation on two standard data sets demonstrates appreciable performance of the proposed approach.

The rest of the chapter is organized as follows: Section 5.1 presents related work. Section 5.2 illustrates the proposed SM algorithm in detail. Experimental set up and dataset descriptions are stated in Section 5.3. Section 5.4 demonstrates the performance of all the models concerning topical and topical document clustering. Section 5.5 concludes our basic findings and future work objectives.

5.1 Related Work

Document clustering is one of the most powerful tools and an active area of research in recent years to improve the performance of search engine results. A survey on different document clustering techniques is presented in Aggarwal and Zhai (2012), Oikonomakou and Vazirgiannis (2005) and Berkhin (2006). A survey of various mathematical models and the corresponding algorithms along with the survey of different document clustering algorithms is given in Ni (2004).

Topical clustering approaches, which group document collection by topic, have been proposed to improve effectiveness and efficiency of the results. These approaches can be grouped into two broad categories: Document-pivot approach and Feature-pivot approach (Aiello et al. 2013).

Document-pivot approaches group/identify the document's topics by measuring the relatedness among documents. Riloff (1996), Schütze and Silverstein (1997), Vaithyanathan and Dom (1999), Yangarber et al. (2000) and Surdeanu et al. (2006) have used document-pivot approaches to cluster documents. However, these approaches use whole documents in clustering that is computationally expensive and cannot be used online. Recently, work is started on snippets rather than whole documents. Ball and Hall (1967) focused on short text in clustering technique which can be applied for summarizing multivariate data. Metzler et al. (2007) provided a comparison of various similarity techniques for short text segments. The document-pivot approach(es) may cluster documents that belong to different topics into one cluster that results in broad topic which is the combination of two or more related specific topics or mixed type of different topics which are not related (Petkos et al. 2014a). Feature-pivot approaches (Zamir and Etzioni 1998; Blei et al. 2003; Blei and Lafferty 2006) groups the features by topic. Grouped features are incorporated to form topical document clusters. Pair-wise co-occurrence based feature-pivot approaches generally produce mixture of topics rather than fine-grained topics. However, this issue can be resolved to some extent by considering subset of features to get fine-grained topics (Petkos et al. 2014a). In this chapter, a solution for topical soft document clustering using the existing query clustering algorithms which produce query-context aware hard document clustering has been proposed. This proposed solution is hybrid of the two above mentioned approaches. The query clustering algorithms such as QD, QC and QDC are document-pivot algorithms and the proposed SM algorithm forms topic set for each query by disambiguating web search results based on feature-pivot approach. The latter approach disambiguates the mixed topical clusters produced by document-pivot algorithms.

Search result clustering is an attempt to improve search performance by grouping search results based on topics. Osinski et al. (2004) proposed a concept-driven algorithm for clustering search results by recognizing each snippet separately. The extensive survey of search result clustering is given by Carpineto et al. (2009). Some researchers proposed method for SRC using domain knowledge or external directories. In Scaiella et al. (2012) topics are detected by TAGME and clusters are produced by deploying semantics underlying the link-structure of Wikipedia. However, heuristic based scoring methods like TAGME require careful parameter tuning (Houlsby and Ciaramita 2013). In this chapter, SRC is used as a technique to form topic sets for query clusters.

Soft clustering is used to deal with multi-topical documents. The work proposed in Hung and Yang (2001) supports soft clustering using fuzzy C-means algorithm for topical clustering. But, fuzzy C-means algorithm based approach has the following limitations: performance dependency on its initialization and unable to handle very large document collections (Lu et al. 2011). Probabilistic clustering frameworks built upon Expectation-Maximization (EM) algorithms (Bradley et al. 2000; Ordonez and Omiecinski 2002) and Fuzzy Adaptive Resonance Theory (Fuzzy-ART) neural network (Kondadadi and Kozma 2002) are the some more examples of work on soft clustering. EM algorithm is extremely computationally intensive, especially for large problems and it requires a good initial guess for either the bias field or for the classification estimate. Otherwise, the EM algorithm could be easily trapped in a local minimum, resulting in an unsatisfactory solution (Ahmed et al. 2002). Fuzzy-ART depends critically upon the order in which the training data is processed regardless of the size of the data (Sarle 1995). The effect can be reduced to some extent by using a slower learning rate.

5.2 The Proposed Algorithm: Split-Merge (SM)

The aim of the proposed approach is to form soft document clusters on a precise topics. The system diagram of the proposed approach is given in the Figure 5.2. Middleware has been developed for extracting information from Google search engine. The proposed approach has three modules: 1. Topic Set Identification module (online module), 2. Clustering module (offline module) and 3. SM algorithm module (offline module). Topic Set Identification module (TSIM) identifies the topic set for each query by grouping related concepts (Refer chapter 4), Clustering module (CM) processes the click-through log and clusters related queries, documents and concepts using clustering algorithms mentioned in chapter 3. The proposed technique, which is a two phase Split-Merge algorithm, is applicable to results (topical clusters) of query-context aware clustering algorithm to produce refined topical clusters, associated document and query clusters. SM algorithm refines the topical clusters by relating them to the topic set of query clusters (split phase) and then by merging similar topical clusters according to the associated query clusters (merge phase).

The following are the problems due to which an unrelated agglomerative merging in algorithm A may occur during the clustering:

1. Different meanings of the concepts may lead users to click on the same document for unrelated queries.
2. Unrelated documents clicked for a query, i.e. noisy clicks.
3. One concept/document/query can be placed in only one cluster as a part of traditional hard clustering.

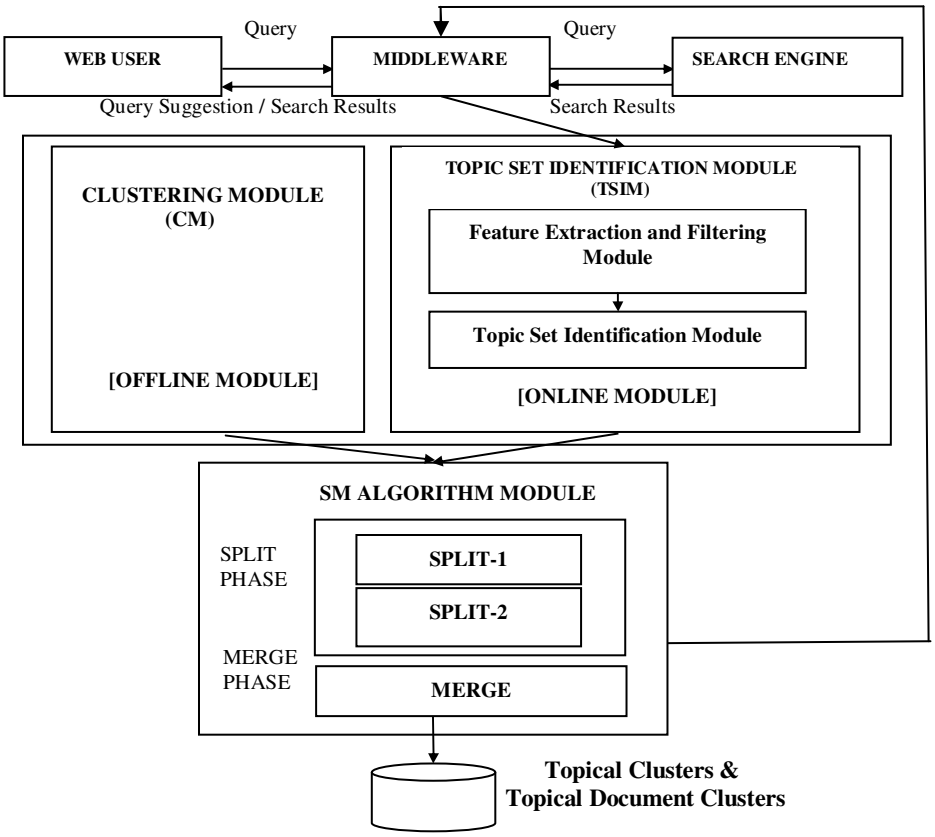


Figure 5.2. System Diagram

5.2.1 Split Phase

In split phase, a concept cluster that has more than one topic splits into two or more clusters using a two stage splitting process: split-1 followed by split-2. Two factors, cluster homogeneity and multi-topical document characteristics, have been used as deciding factors for a cluster to split. Every concept cluster obtained from the clustering algorithm goes through split-1 and split-2 processes. The aim of the split phase is to handle issues stated above.

1. Split-1

Split-1 process takes input as query, topical and document clusters to perform intra cluster splitting to resolve problems 1 and 3 (stated above). We will now consider the example (see Figure 5.1) to illustrate the splitting process:

Example

In Figure 5.1, topical cluster C1: {interview, job, questions, Rod Johnson, comments, views} is associated with the query clusters QC1': $\{q_2, q_3\}$ & QC1": $\{q_5\}$ (see Table 5.1) and respective document clusters DC1' & DC1". The DC1 has 11 documents $\{d_6\text{-}d_9\}$, $\{d_4, d_{10}\text{-}d_{12}\}$ and $\{d_{15}\text{-}d_{16}\}$ which are clicked for the queries q_2, q_3 and q_5 respectively. Also consider the topical cluster C2 which is associated with the query cluster QC2 and document cluster DC2. The topical cluster C2: {Java, beginner, spring, tutorial, code, framework} and QC2 has the queries q_1, q_2, q_3 & q_4. The DC2 has 14 documents d_1: d_{14} where, documents clicked for the queries are as follows: q_1: $\{d_1\text{-}d_5\}$, q_2: $\{d_6\text{-}d_9\}$, q_3: $\{d_4, d_{10}\text{-}d_{12}\}$ and q_4: $\{d_9, d_{13}\text{-}d_{14}\}$. Details of sample cluster C2 and the respective clicked documents and concepts after clustering algorithm are given in Table 5.2. It is clear from the table that topical clusters have concepts related to multiple topics. This is because of problems 1 and 3 stated above. For example, "spring" is a concept which is common in three different topics: "java spring framework", "spring season" and "spring taxi". Through the common concepts like “spring” and the associated queries, documents and concepts are merged and formed noisy clusters.

Split-1 as applied to C1 and C2 is shown in Figure 5.3a and 5.3b respectively. Split-1 splits C1 into two topical clusters, $C1^3$: (Topic: football player Rod Johnson's interview comments) with the query cluster $QC1^3$: $\{q_5\}$ and intermediate cluster IC1: {interview, job, questions, Rod Johnson} with the intermediate query cluster IQC1: $\{q_2, q_3\}$. Further, IC1 splits into two topical cluster $C1^1$: (Topic: Rod Johnson who created the Spring Framework) and $C1^2$: (Topic: job and interview related to code) with the associated query cluster $QC1^1$: $\{q_3\}$ and $QC1^2$: $\{q_2\}$ respectively. It is to be noted from the figure that the concepts “Rod Johnson” and “interview” belongs to more than one topical cluster. In this way, soft clustering is introduced in SM algorithm. This soft clustering is mainly due to the reason that common concepts are not

removed from the topical cluster as soon as it get split (refer SPLIT-1 algorithm given in Figure 5.4).

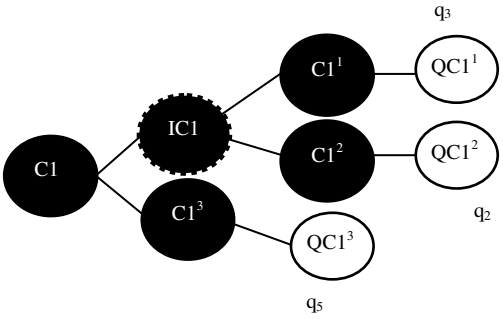
Similarly, split-1 identifies multiple topics of the queries and splits C2 into four topical clusters $C2^1$ (Topic: java language tutorial), $C2^2$ (Topic: java code), $C2^3$ (Topic: java language tutorial for the beginners) and $C2^4$ (Topics: java spring framework, spring taxi and framework journal). However, $C2^4$ shows more than one topic that is because of problem 2 stated above. It is handled by split-2.

The result set after applying split-1 will have high precision/purity. The recall would not decrease due to soft clustering.

The algorithm for split-1 has been described in Figure 5.4 where ***IDS*** describes an Iterative Deepening Search algorithm (Korf 1985). It is used to generate all the possible query subsets of a given query set Q of size ***size***. For example, given a query set Q: $\{q_1, q_2, q_3, q_4\}$ and for ***size***=3, the function returns the query subsets of size 3 in subQ: $\{\{q_1, q_2, q_3\}, \{q_1, q_2, q_4\}, \{q_1, q_3, q_4\}, \{q_2, q_3, q_4\}\}$. subQ is used to split the topical cluster C associated with Q. The split is based on the common concepts among the queries of a subset and topical cluster obtained from click-through log/feedback. The function find_common_concepts() returns the common concepts between $S \in subQ$ and C. For example, query cluster S has q_1, q_2 & q_3 and the queries q_1, q_2 and q_3 are related to concepts $\{c_1, c_2, c_4\}$, $\{c_1, c_2, c_3\}$ and $\{c_1, c_2, c_3\}$ respectively. Then the function returns c_1 and c_2 as a separate cluster C1 and returns a cluster C2 with remaining concepts c_3 and c_4. However, the query set S is associated with both the clusters C1 and C2. Splitting will be done for each query subset S of Q of different sizes related to the topical cluster C. This splitting process will be stopped when the cluster has all common concepts. Thus, C1 would not split again. However, the split would take place in C2. The process will be carried out for each topical cluster in CS. The resultant topical clusters and respective query clusters are stored in topical cluster CS1 and related query cluster CQS1.

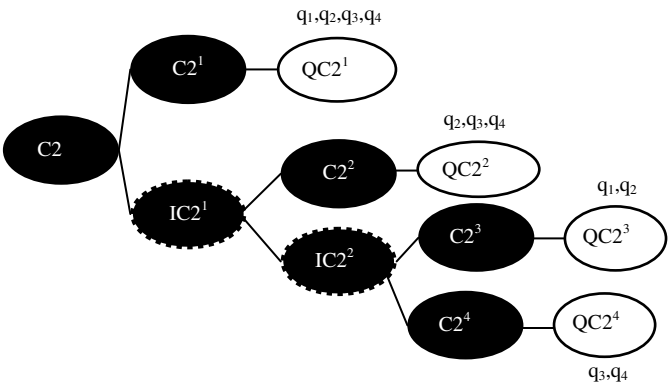
Table 5.2. Sample Cluster Details of cluster C2

Queries and the respective clicked documents' information
q_1: { {d_1,d_2,d_3}: java language tutorial for the beginners, {d_4}: spring season, {d_5}: java spring framework}
q_2: {{d_6}: java island, {d_7}: java language based jobs & interview questions, {d_8}: java language tutorial for the beginners, {d_9,d_{14}}: java code}
q_3: {{d_9}: java code, {d_{10},d_{11}}: java spring framework, {d_2}: java language tutorial for the beginners, {d_{12}}: spring taxi }
q_4: {{d_{13}}: framework journal, {d_{14}}: java spring framework, {d_9}: java code}
q_5: {{d_{15}, d_{16}, d_{17}}: Rod Johnson Football player }



Topical cluster	Concepts
C1	Interview, job, questions, Rod Johnson, comments, views
IC1	Interview, job, questions, Rod Johnson
C1¹	Rod Johnson
C1²	Interview, job, questions
C1³	Interview, Rod Johnson, comments, views

(a)



Topical cluster	Concepts
C2	Beginner, java, spring, tutorial, code, framework
IC2¹	Beginner, spring, code, Framework
IC2²	Beginner, spring, framework
C2¹	Java, tutorial
C2²	Code
C2³	Beginner
C2⁴	spring, framework

(b)

Figure 5.3. (a) & (b). Split-1 examples

```
Algorithm: SPLIT-1
Input: CS, CQS/* CS & CQS are Set of topical clusters and the associated set of
query clusters */
Output: CS1,CQS1/* CS1& CQS1are resultant  Set of topical clusters and the
associated set of query clusters */
for each topical cluster C ∈ CS
get QS associated to C// QS  ⊆ CQS
for each query cluster Q ∈ QS
        for each i from n to 1          // n is the size of query cluster Q
            remove all elements of Lcon
            /*Lcon: temporary concept set to store common concepts*/
            IDS(Q, 0, i, subQ)
        for each S in subQ
                if(Ct=find_common_concepts(S,C))!= null
                        add cluster Ct in CS1 &add cluster  S in CQS1
        add Ct to Lcon
        remove elements of Lcon from C

find_common_concept (Q, C)
if Qsize is 1 // Qsize is number of queries in Q
        return CL i.e. concepts related to Q
else
        temp=∩( CL_i ) ∀i such that q_i ∈ Q
return temp

IDS(Q, r, s, subQ)
    /*returns a set of query subsets of size i from Q in subQ. Each element in
    subQ refers a subset of size i */
if s is zero
        returnsubQ
else
        for each t from r to Qsize – s //Qsize is number of queries in Q
        subQ[len-s]=Q[t]/* Q[t] is t^th element in Q and len is length of subQ
which is equal to value of i in the calling function */
            IDS(Q,t+1,s-1,subQ)
```

Figure 5.4. Algorithm SPLIT-1

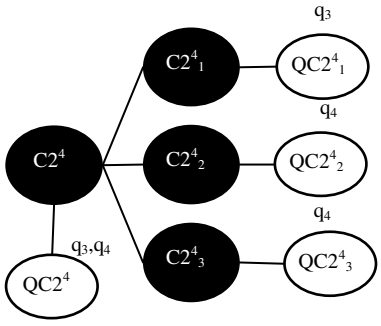
2. Split-2

Split-2 uses topic sets of queries (Refer Chapter 4) by finding a topic overlap to split multi-topical topical clusters that could not be corrected by split-1. The problem 3 mentioned above can be resolved by identifying topics of the query submitted and documents clicked by the user. The splitting process further reduces the effect of noise in the click-through data. We now present an example to understand.

Example

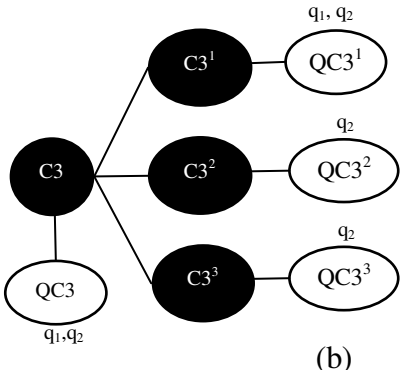
Consider the same example as in split-1. $C2^4$ is multi topical cluster i.e. concept of $C2^4$ belongs to three topics {T1: java spring framework, T2: spring taxi and T3: framework journal}. In split-2, topic overlap among the topics of multiple queries is calculated and if it is below split-threshold the cluster splits. Figure 5.5a shows the split-2 in detail where, $C2^4$ split in three clusters $C2^4_1$ (Topic: java spring framework), $C2^4_2$ (Topic: spring taxi), $C2^4_3$ (Topic: framework journal).

Similarly, the split-2 has been shown in Figure 5.5b. As mentioned earlier in the Figure 5.1, the topical cluster C3 :{ MVC, web application, android, source code, samples} is associated with a query cluster and a document cluster QC3 :{ q_1, q_3} and DC3: {d_5, d_8, d_9} respectively. C3 has different topics :{ spring architecture, Android OS, small code snippets that are helpful in programming}.



(a)

Topical cluster	Concepts
$C2^4$	Spring, Framework
$C2^4_1$	Spring, Framework
$C2^4_2$	Spring
$C2^4_3$	Framework



(b)

Topical cluster	Concepts
C3	MVC, web application, android, source code, samples
$C3^1$	MVC, web application
$C3^2$	android
$C3^3$	source code, samples

Figure 5.5. (a) & (b). Split-2 examples

The Split-1 process does not succeed to remove the noise of C3. However, the split-2 successfully splits C3 into three clusters C31 :{ MVC, web application} (Topic: spring architecture) and C32 :{ Android} (Topic: Android OS) and C33 :{ source code, samples} (Topic: small code snippets that are helpful in programming).

Topic Overlap

Consider a concept set C: $\{c_1, c_2, ..., c_k\}$ and corresponding query set Q: $\{q_1, q_2, ..., q_j\}$. For each query $q \in Q$, construct the topic set TS_q: {T1, T2, ...Tn} where, each Ti is set of concepts and has at least one concept $\in C$ with respect to the query q}. Topic overlap is found for each combination of topics from each topic set. For example, given a query set Q: $\{q_1, q_2\}$ and topic sets TS_{q1}: $\{T1_{q1}, T2_{q1}\}$ & TS_{q2} :{ $T1_{q2}$, $T2_{q2}$}, combinations are $\{T1_{q1},T1_{q2}\}$, $\{T1_{q1}, T2_{q2}\}$, $\{T2_{q1}, T1_{q2}\}$ and $\{T2_{q1}, T2_{q2}\}$ (see GenerateKCombinations() function in Figure 5.6). We find overlap for each combination. Topic overlap $O(T_i, T_j)$ is defined as:$O(T_i, T_j) = (|T_i \cap T_j|) / (|T_i \cup T_j|)$.

The result set after applying split-2 will have high precision/purity as compared to split-1. The recall would not decrease due to soft clustering (see Experimental results given in Table 5.5 & 5.6).

The algorithm for split-2 has been described in Figure 5.6. If topic overlap of the topical cluster C is less than the split-threshold, C is partitioned. The partition process is similar to split-1 except that the common concepts are extracted using the function common_cat_concept() in place of the function find_common_concepts(). The function finds the common concepts based on the topic sets for which the similarity is greater than split-threshold.

Algorithm: SPLIT-2
Input: ***CS1, CQS1*** */* CS1 & CQS1 refer Set of topical clusters and the associated set of query clusters obtained as a result of split-1 process*/*
Output: ***CS2***, ***CQS2*** */* CS2 & CQS2 refer Set of topical clusters and the associated set of query clusters obtained after split-2 process*/*

```
for each topical cluster C∈CS1
get QS associated to C // QS refers set of query clusters
for each query cluster Q∈QS
        for each i from n to 1          // n is the size of query cluster Q
        remove all elements of  Lcon  //*Lcon: temporary concept set to store common concepts*/
        IDS(Q, 0, i, subQ) /* returns set of query subset of size i from Q in subQ */
        for each S in subQ  // Each element in subQ refers one subset of size i
                k=S.size
                GenerateKCombinations(S,C,k,comb) /*returns topic index combinations for S with respect to
different intention of query for intention based query similarity finding*/
                for each  k2 from 0 to comb.size
                 // for each category combination
                        for each j from 0 to k-2
                         C1=Retrieve concept set related to the topic Tk for the query qj where,k= comb[k2][j]
                         C2=Retrieve concept set related to the topic Tk for the query qj+1 where,k= comb[k2][j+1]
                          calculate topic overlap O between C1 and C2
                if O< threshold
                          exit
                else
                common_cat_concept(S,C,comb[k2],Ctemp)    /* returns common topic related concepts in Ctemp */
                add cluster Ctemp in CS2
                add cluster S in CQS2
        add concepts of Ctemp in Lcon
        retrieve concepts such that ∀c ε C and ∀c ∉Lcon : Ctemp1
                add cluster Ctemp1 in CS2
                add  cluster S in CQS2
        remove the concepts of Lcon from C
        emptyLcon

GenerateKCombinations(Q,C,k,comb)
calculate count   /* total number of topics for all query q ∈ Q and those topic contains atleast one concept in C */
for each qi∈ Q
        RetrieveTSqi for the query qi and which contains atleast one concept in C
 k1=0
        for each I from 0 to count-1
j = 1
        for each  m from 0 to Q.size-1
                comb[k1][m]=indexof(TSm[(i/j)%TSm.length]) /*TSm is Topic set of  mth query qm,  TSm[x] is the xth
        topic in TSm, TSm.length is no. of  topic related to query qm */
                j = j* TSm.length
increment K1 by 1

Common_cat_concept(Q,C,cmb,Ctemp)
for each i from 0 to Q.size
    Retrieve Concepts related to TSik for the query qi and topic Tk.: C1     // where, k is the topic index for qi in cmb
  Retrieve common concepts in C1and C: temp1
  Retrieve common concepts in Ctemp and temp1 :Ctemp
return Ctemp
```

Figure 5.6. Algorithm SPLIT-2

The splitting will be done for each query subset related to the topical cluster C. This process will be repeated for each topical cluster in CS1. The resultant topical and respective query clusters are stored in CS2 and CQS2 respectively.

Split-1 increases purity using clicks from the users. However, the split-1 is not able to identify the noise due to multi-topical concepts. Split-2 provides us a way to refine the clusters based on an intention / topic using the content of the snippets of the document along with clicked information.

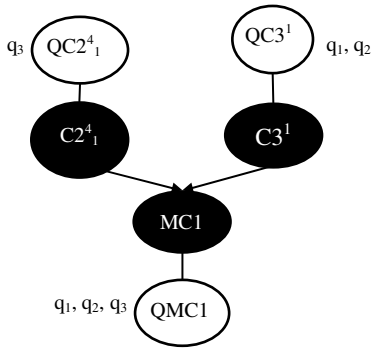
This idea of topic based split can also be applied in other applications such as mailing system etc.

5.2.2 Merge Phase

Split phase splits topical clusters into several homogeneous clusters in which each cluster has concepts and its own topic. In merge phase, similar topic clusters are merged together. This helps to overcome over-partitions. Similar topic clusters are found using two properties: Topic overlap, and Cluster homogeneity.

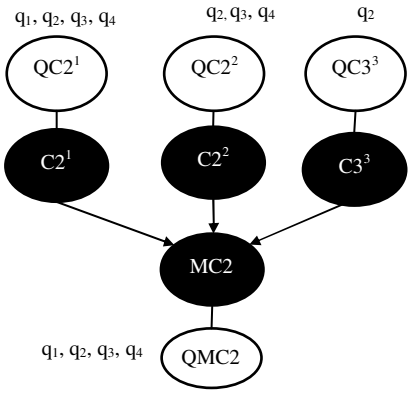
Example

The Sample results for merging are shown in Figure 5.7a and 5.7b through two examples. It is clear from the figure that the topical clusters $C2^4_1$: (Topic: spring as a framework) and $C3^1$: (Topic: web application) are merged into a cluster MC1. The resultant cluster MC1 has a topic “Developing web application using spring framework". In Figure 5.7b MC2 is formed by merging $C2^1$, $C2^2$ and $C3^3$ topics of java tutorial, code related to spring as a framework and some sample codes of a programming language respectively. Merged cluster MC2 refers the topic "code snippets or tutorials on spring development in java". However, no explicit topic labels are given to the clusters. This clearly indicates the role of merging process in successive steps.



Topical cluster	Concepts
$C2^4_1$	spring, framework
$C3^1$	MVC, web application
MC1	spring, framework, MVC, web application

(a)



Topical cluster	Concepts
$C2^1$	java, tutorial
$C2^2$	code
$C3^3$	source code, samples
MC2	java, tutorial, code, source code, samples

(b)

Figure 5.7. (a) & (b). Merge examples

Our approach selects the most similar clusters by looking at their closeness with the topics in the topic sets by measuring topic overlap. Similar topical clusters are found by the extended concept sets of topical clusters. Extended concept set of a topical cluser Ci is formed by taking the union of all concepts of every query in the associated query cluster of Ci. It is shown in the function named GenerateExpandedConceptSet() (see Figure 5.8). If the topic overlap/similarity between two topical clusters Ci and Cj is greater than the merge-threshold then these two clusters will be merged. This process will be repeated until there is no more pair that has similarity greater than the merge-threshold. Higher recall can be achieved by decreasing the merge-threshold. So, merge-threshold can be varied on-demand, according to the application requirements (see Table 5.5 and 5.6).

Algorithm: Merging
Input: ***CS2, CQS2*** */* Set of topical clusters and associated set of query clusters respectively obtained after split-2 process*/*
Output: ***CSfinal,CQSfinal***

for each concept cluster $Ci \in CS2$
for each concept cluster $Cj \in CS2$ and $Ci \neq Cj$
GenerateExpandedConceptSet(Ci, Qi): ECi
GenerateExpandedConceptSet(Cj, Qj): ECj
/*Qk is query cluster associated to Ck*/
calculate similarity between all pair of concept clusters Ci and Cj
as$|(ECi \cap ECj)|/|(ECi \cup ECj)|$: S
repeat
find concept clusters pair Ci and Cj with highest similarity (S_H)&&S_H>merge-threshold
merge Ci and Cj and add merged cluster MCi in $CS2$
remove Ci and Cj from $CS2$
merge Qi and Qj and add merged cluster $QMCi$ in $QCS2$
remove Qi and Qj from $QCS2$
recompute the similarity between MCi and each concept cluster in $CS2$
until there is atleast one merge progress
copy $CS2$ to $CSfinal$ and $QS2$ to $QSfinal$

GenerateExpandedConceptSet(*CSi*,*QSi*)
empty temp
for each query $q_i \in QSi$
retrieve Topic Set TS_{qi} for the query q_i *// TS_{qi} is topic set of the query q_i*
retrieve concept set $C \in Tj_{qi}$ where, $Tj_{qi} \in TS_{qi}$ and contains CSi
add each elements of C in temp
return temp

Figure 5.8. Algorithm MERGE

5.3 Experimental Setup and Data Sets

The proposed method is applied on the resultant of existing clustering algorithm discussed which is capable of producing concept/topical clusters and the associated query and document clusters. We have used two datasets for evaluation: 1. TREC 2011 session track data set (Jiang et al. 2011) and 2. Data set which is used in (Goyal and Mehala 2011; Goyal et al. 2013). Three datasets (DS1, DS2 and DS3) are constructed from the two datasets mentioned earlier for experiments. DS1 consists of full TREC 2011 session track dataset. Some of the TREC data set queries were singleton that make clusters of size one. Therefore, all the clustering algorithms may give same results as it suppresses the meaning of clustering. For this reason, we have removed queries that form clusters of size one and two from the TREC 2011 session track dataset and formed the dataset DS2. DS3 is the third dataset which is formed by combining DS2 and the Data Set used in (Goyal and Mehala 2011; Goyal et al. 2013) to verify the effect of the

proposed model on large dataset and as these two datasets also have some queries of same intention in common. For example, queries "Guide" and "tax" in (Goyal and Mehala 2011; Goyal et al. 2013) are having similar intention as that of the queries "AMT", "AMT and Tax" in DS2. The statistics of the data collected for concept extraction and clustering experiments for the datasets DS1, DS2 and DS3 are given in Table 5.3.

Table 5.3. Statistics from concept extraction and clustering experiments

	DS1	**DS2**	**DS3**
Number of unique queries	121	53	126
Max number of extracted concepts for a query	1089	617	712
Minimum number of extracted concepts for a query	450	308	308
Number of URLs retrieved for all queries	11779	5300	12600
Total number of concepts retrieved	85679	29753	79360
Maximum number of URLs clicked for a query	10	10	10
Number of unique URLs clicked	178	70	424
Total number of concepts considered (having weight>δ) for the clicked URLs	1541	653	2314
Total Number of unique concepts considered (having weight>δ) for the clicked URLs	1162	465	1972

The statistics on the basis of result of clustering algorithm A is given in Table 5.4. The predefined clusters are manually formed by placing similar intend concepts and similar intend documents into the respective clusters, as there is no gold standard cluster information available for any data set which has required information. Sample manually formed clusters are given in Table 5.8. Number of clusters formed by QC is too less than the expected actual number of clusters to be formed. Table 5.4 also shows that maximum number of concepts in a topical cluster formed by QC on DS1, DS2 and DS3 is 80, 74 and 221 respectively. That is QC is forming less number of big clusters which in turn reduces precision and increases recall (see Section 5.4). The same behavior is observed with respect to the topical document clustering results as well.

Table 5.4. Comparison of output generated by different clustering algorithms and ground truth

	QC			QDC		
	DS1	DS2	DS3	DS1	DS2	DS3
Actual Number of topical clusters generated manually	114	42	461	114	42	461
No. of Topical clusters formed	131	38	91	101	48	427
Maximum no. of concepts in a topical cluster	80	74	221	39	26	25
Average no. of concepts in a topical cluster	23	16	28	13	11	14
Actual Number of topical document clusters generated manually	114	42	394	114	42	394
No. of topical document clusters formed	82	38	91	114	49	409
Maximum no. of documents in a topical document cluster	8	10	63	6	5	16
Average no. of documents in a topical document cluster	2	3	10	2	2	2

5.4 Experimental Results

We now present results which show the performance of the SM algorithm. The performance of the proposed algorithms has been analyzed on three datasets DS1, DS2 and DS3. The measures recall, precision and F-measure are used for comparison of results (An analysis on the choice of measures for such evaluation is given in Amigo et al. (2009)). These measures are calculated for the topical clusters and topical document clusters obtained from the algorithm for various models and compared with predefined clusters.

The precision (*P*), recall (*R*) and F-measure (*F*) for a given concept *c* (a given document *d*) and associated topical cluster having the concepts $c_1, c_2, ..., c_n$ (document cluster having the documents $d_1, d_2, ..., d_m$) produced by a clustering algorithm, are computed using the following formulae:

$$P(x) = \frac{|S \cap S'|}{|S|} \qquad R(x) = \frac{|S \cap S'|}{|S'|} \qquad F = \left(2.\frac{(P.R)}{(P+R)}\right) \tag{5.1}$$

where *x* can take two values, either concept *c* or document *d*. *S* is the set of concepts (documents) that exist in the predefined cluster for *c(d)* and *S'* is set of the related concepts $c_1, c_2,...,c_n$ (documents $d_1,d_2,...,d_m$) generated by the SM algorithm. The precision and recall values from all concepts (documents) are averaged.

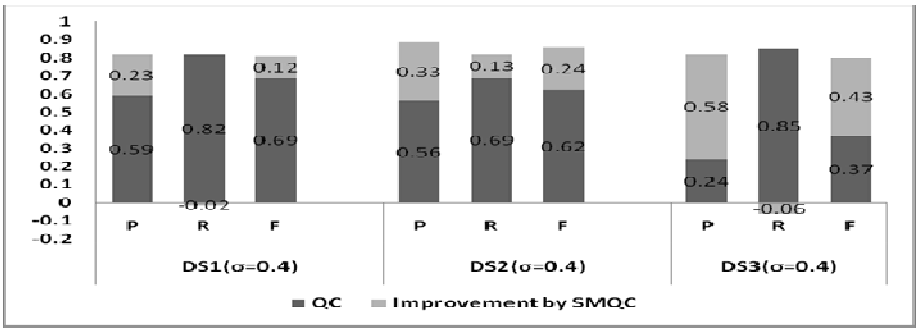
Most of the experimental results are presented in terms of best mean precision, recall and F-measure because every algorithm performs best at a different value of cut-off-similarity score σ.

For a particular value of σ, one algorithm may work the best, whereas other algorithm can be worse.

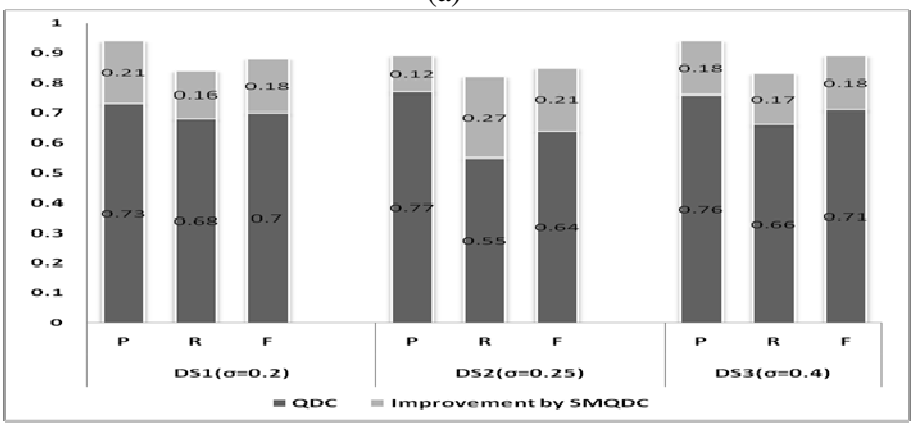
5.4.1 Topical Clustering

In this section, we present results for topical clusters obtained by the SM algorithm:

We will first present results which show the performance of the SM algorithm over the existing hierarchical clustering algorithms. From the experimental results, it is observed that the cut-off similarity score σ (for agglomerative merge in clustering algorithms) value will differ for different models. Therefore, results are obtained for different σ values for different algorithms to get the best performance. This parameter is empirically analyzed in our experiments. The best Precision, Recall and F-measure values are measured and shown in Figure 5.9a for QDC, SMQDC and in Figure 5.9b for QC, SMQC algorithms when applied on DS1, DS2 & DS3. Both precision and recall are increased when SM algorithm is applied on QC and QDC as shown in the figure. It can also be observed that SMQDC achieved both higher precision and higher recall than that of SMQC. However, recall of QDC and precision of QC are limited by the hard clustering and lack in usage of precise clicked document information respectively. Precision of most of the topical clusters formed by QC is low as the method results in less number of big topical clusters. This is because QC is merging the queries if these are linked via common concept(s). Similarly, concepts are merged if they are linked via common queries. The performance is becoming worse when same queries with distinct intentions are having common concept(s) (the average number of overlapping concepts between the queries is 56, which is very large). It is also clear from the figure that the SMQDC and SMQC algorithm are able to achieve F-measure above 0.85 and 0.8 respectively, whereas other methods have F-measure 0.7 or below 0.7. It is also visible that SM algorithm gives best result on the higher cut-off threshold (σ=0.4) in the agglomerative merging process in most of the cases. The improvements achieved are higher for SMQC because QC performance is quite low. Percentage improvement is also calculated for SMQDC and SMQC over QDC and QC respectively. The percentage improvement is around 20%, 27% and 21% in precision, recall and F-measure respectively, for SMQDC and 57%, 13% and 43% for SMQC.



(a)



(b)

Figure 5.9. Comparison of Precision, Recall and F-measure of topical clusters obtained with and without SM algorithm

The sample intermediate results for SMQDC and SMQC are analyzed and given in Table 5.5 and 5.6 respectively, for various split-thresholds and merge-thresholds. It is clear from the Table 5.5 that SMQDC split phase improves precision (by 0.23) without much deviation in the recall (0.04). It is also observed that the merge phase increases both precision and recall. The split-threshold does not change precision and recall much in split-2 process on DS3. The same behaviour is observed for DS1 and DS2. Unlike split, the merge-threshold has more impact on precision and recall. It is observed from the results that precision (recall) is increasing as merge-threshold increases (decreases). Merge-threshold can be varied according to the application requirements. For example, a merge-threshold can be lowered to give recommendation of products to the consumer. Whereas, high merge-threshold can be opted for the application like spam filtering. Cut-off similarity can be optimized for the highest value of F-Measure in information retrieval. From our experimental analysis, it is observed that F-measure value is highest at 0.15 for both split-thresholds and merge-thresholds.

Table 5.5. Sample intermediate results of topical clustering for SMQDC on DS3 at different split-thresholds and merge-thresholds

QDC				SPLIT-1			
P		R	F	P	R	F	
0.765		0.662	0.710	0.831	0.649	0.729	
SPLIT-2				MERGE			
split-threshold	P	R	F	merge-threshold	P	R	F
				0.10	0.882	0.802	0.840
				0.15	0.946	0.823	0.880
				0.25	0.977	0.798	0.879
0.10	0.996	0.62	0.764	0.50	0.982	0.779	0.869
				0.10	0.878	0.803	0.839
				0.15	**0.944**	**0.836**	**0.887**
				0.25	0.979	0.796	0.878
0.15	0.994	0.62	0.764	0.50	0.985	0.780	0.871
				0.10	0.883	0.804	0.842
				0.15	0.953	0.823	0.883
				0.25	0.981	0.793	0.877
0.25	0.996	0.62	0.765	0.50	0.990	0.779	0.872
				0.10	0.885	0.803	0.842
				0.15	0.954	0.820	0.882
				0.25	0.986	0.795	0.880
0.50	0.996	0.623	0.766	0.50	0.990	0.778	0.872

Table 5.6. Sample intermediate results of topical clustering for SMQC on DS3 at different split-thresholds and merge-thresholds

QC				SPLIT-1			
P		R	F	P	R	F	
0.238		0.846	0.371	0.327	0.843	0.471	
SPLIT-2				MERGE			
split-threshold	P	R	F	merge-threshold	P	R	F
				0.10	0.700	0.828	0.759
				0.15	0.786	0.816	0.801
				0.25	0.812	0.782	0.797
0.10	0.817	0.729	0.770	0.50	0.823	0.773	0.797
				0.10	0.700	0.828	0.758
				0.15	**0.816**	**0.790**	**0.803**
				0.25	0.820	0.779	0.799
0.15	0.817	0.730	0.771	0.50	0.826	0.771	0.798
				0.10	0.701	0.828	0.759
				0.15	0.793	0.814	0.803
				0.25	0.822	0.775	0.798
0.25	0.817	0.730	0.771	0.50	0.828	0.768	0.797
				0.10	0.707	0.824	0.761
				0.15	0.799	0.805	0.802
				0.25	0.825	0.774	0.799
0.50	0.820	0.730	0.772	0.50	0.832	0.758	0.793

It is clear from the Table 5.6 that in case of QC, split phase plays an important role as precision increases from 0.238 to 0.327 and 0.817 in split1 and split2 respectively. This is because QC forms a less number of big topical clusters (having poor precision). Recall in the split phase decreases, which later in the merge phase again improves as merge phase combines related clusters. As a whole without much decrease in recall, we could achieve high precision. Merge behavior in SMQC is similar to SMQDC.

F-measure increases from 0.371 to 0.803 for QC and from 0.710 to 0.887 for QDC.

5.4.2 Topical Document Clustering

We will now present the results of evaluation of topical document clustering. We have examined the quality of topical document clusters generated by QDC, QC, SMQC and SMQDC with that of predefined clusters in static scenarios.

Table 5.7 compares the best mean precision, recall and F-measure of topical document clusters generated by the clustering algorithms with and without SM algorithms. All the algorithm performs better at cut-off similarity threshold (σ) 0.4 except that QDC and SMQDC performs better at σ =0.2 on DS1. It is clear from the Table 5.7 that the SMQDC performs better as compared to QDC and QC for document clustering too. Percentage improvements in F-Measure of document clustering by SMQDC over QDC &SMQC over QC on DS1, DS2 and DS3are up to 11%, 12% and 13% & 1%, 26% and 38% respectively. Increase in precision and recall is observed in SMQDC and SMQC for topical document clustering too. SMQDC attains maximum F-measure as 0.944 and minimum 0.875 for datasets DS3 and DS2 respectively.

Table 5.7. Best precision, recall and F-Measure values of all algorithms on static data sets for topical document clustering

	DS1			DS2			DS3		
	P	R	F	P	R	F	P	R	F
QDC	0.865	0.722	0.787	0.918	0.644	0.757	0.946	0.714	0.814
QC	0.838	0.812	0.825	0.580	0.722	0.643	0.301	0.815	0.439
SMQDC	0.943	0.849	0.893	0.938	0.820	0.875	0.967	0.923	0.944
SMQC	0.881	0.782	0.829	0.950	0.857	0.901	0.860	0.786	0.822

The sample resultant and ground truth topical clusters and the respective query and document clusters are given in Table 5.8.

Table 5.8. Sample SMQDC resultant and ground-truth topical clusters and their respective query and documents' clusters

Sample Resultant Topical Clusters (RT): concepts list	Sample Resultant Query Clusters (RQ) and Document Clusters (RD): documents' id	Sample Manual Topical Clusters (MT): concepts list	Sample Manual Document Clusters (MD): documents' id
RT1: {c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8, c_9}	RQ1: {q_1, q_2, q_3, q_4} RD1: {d_1, d_2, d_3, d_6, d_7, d_8, d_9, d_{10}, d_{11}, d_{14}}	MT1: {c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8, c_9, c_{10}}	MQ1: {q_1, q_2, q_3, q_4} MD1: {d_1, d_2, d_3, d_6, d_7, d_8, d_9, d_{10}, d_{11}, d_{14}}
RT2: {c_{10}}	RQ2: {q_1, q_2} RD2: {d_2, d_8}		
RT3: {c_{11}}	RQ3: {q_1} RD3: {d_5}	MT2: {c_{11}, c_{14}, c_{15}}	MQ2: {q_1, q_2} MD2: {d_5, d_6}
RT4: {c_3, c_{12}, c_{13}, c_{14}, c_{15}, c_4, c_{16}}	RQ4: {q_1, q_2, q_3} RD4: {d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8, d_9, d_{10}, d_{11}}	MT3: {c_3, c_{12}, c_{13}, c_{14}, c_{15}, c_4, c_{16}}	MQ3: {q_1, q_2, q_3} MD3: {d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8, d_9, d_{10}, d_{11}}
RT5: {c_{17}, c_{18}, c_{19}, c_{20}}	RQ5: {q_3} RD5: {d_{12}}	MT4: {c_{17}, c_{18}, c_{19}, c_{20}}	MQ4: {q_3} MD4: {d_{12}}
RT6: {c_{21}, c_{22}, c_{23}}	RQ6: {q_3} RD6: {d_4}	MT5: {c_{21}, c_{22}, c_{23}, c_{12}}	MQ5: {q_3} MD5: {d_4}
RC7: {c_{24}}	RQ7: {q_3} RD7: {d_{11}}	MT6: {c_{24}}	MQ6: {q_3, q_5} MD6: {d_{11}, d_{16}}
RT8: {c_{25}, c_{26}, c_{27}}	RQ8: {q_2} RD8: {d_7}	MT7: {c_{25}, c_{26}, c_{27}}	MQ7: {q_2} MD7: {d_7}
RT9: {c_{24}, c_{25}, c_{28}, c_{29}}	RQ9: {q_5} RD9: {d_{16}}	MT8:{c_{24}, c_{25}, c_{28}, c_{29}}	MQ8: {q_5} MD8: {d_{16}}
RT10: {c_{30}, c_{31}, c_{32})	RQ10: {q_4} RD10: {d_{13}}	MT9: {c_{30}, c_{31}, c_{32})	MQ9: {q_4} MD9: {d_{13}}
RT11: {c_{33}, c_{34}, c_{35}, c_{36}, c_{37}, c_{38}, c_{39}, c_{40}, c_{41}, c_{42}, c_{43}, c_{44}, c_{45}, c_{46}}	RQ11: {q_6, q_8} RD11: {d_{18}, d_{20}, d_{21}}	MT10: {c_{33}, c_{34}, c_{35}, c_{36}, c_{37}, c_{38}, c_{39}, c_{40}, c_{41}, c_{42}, c_{43}, c_{44}, c_{45}, c_{46}}	MQ10: {q_6, q_8} MD10: {d_{18}, d_{20}, d_{21}}
RT12: {c_{47}, c_{48}, c_{49}, c_{50}, c_{51}, c_{52}, c_{53}}	RQ12: {q_7} RD12: {d_{19}}	MT11: {c_{47}, c_{48}, c_{49}, c_{50}, c_{51}, c_{52}, c_{53}}	MQ11: {q_7} MD11: {d_{19}}
RT13: {c_{54}, c_{55}, c_{56}}	RQ13: {q_{11}} RD13: {d_{23}}	MT12: {c_{54}, c_{55}, c_{56}}	MQ12: {q_{11}} MD12: {d_{23}}
RT14: {c_{55}, c_{56}, c_{57}, c_{58}, c_{59}, c_{60}, c_{61}, c_{62}, c_{63}, c_{64}, c_{65}, c_{66}, c_{67}, c_{68}}	RQ14: {q_{13}} RD14: {d_{24}}	MT13: {c_{55}, c_{56}, c_{57}, c_{58}, c_{59}, c_{60}, c_{61}, c_{62}, c_{63}, c_{64}, c_{65}, c_{66}, c_{67}, c_{68}}	MQ13: {q_{13}} MD13: {d_{24}}
RT15: {c_{69}, c_{70}, c_{71}, c_{72}, c_{73}, c_{74}, c_{75}, c_{76}, c_{77}, c_{78}, c_{79}, c_{80}, c_{81}, c_{82}, c_{83}}	RQ15: {q_{11}, q_{12}} RD15: {d_{17}, d_{22}}	MT14: {c_{69}, c_{70}, c_{71}, c_{72}, c_{73}, c_{74}, c_{75}, c_{76}, c_{77}, c_{78}, c_{79}, c_{80}, c_{81}, c_{82}, c_{83}}	MQ14: {q_{11}, q_{12}} MD14: {d_{17}, d_{22}}

The resultant cluster RT3 shown in Table 5.8 consists of the concept android (c_{11}). But the respective ground-truth actual cluster mentions that this cluster should have other concepts c_{14}(MVC) and c_{15}(web application) along with concept c_{11}(android). But, our method is not

capable of placing the missing concepts as the concepts c_{14}(MVC) and c_{15}(web application) is grouped under one topic referring the intention "java web application development" rather than placing these under the topic with the intention "android web application development". This is due to the hard topic set identification method. This type of issues can be resolved by introducing soft clustering in the topic set identification process. The same can be observed with respect to RT6 and MT5.

MT1 is having combined concepts of RT1 and RT2. Our method is not able to find the topic overlap among these two topical clusters, as RT2 has only one concept "beginner"(c_{10}). This cluster is formed as a result of the underlying clustering algorithm. Due to insufficient information, SM algorithm is not able to find the related topical cluster.

5.5 Conclusions

In this chapter, a two-phase Split-Merge (SM) algorithm, which is a post processing technique, has been proposed to produce soft topical document clusters along with topical and query clusters. The proposed algorithm is tested on resultant of two hierarchical clustering algorithms on different datasets including TREC session track 2011 dataset. Experimental evaluations show that the improvement achieved by the proposed SM algorithm on all data set is encouraging.

Clustering large document collections remains a challenging problem especially when documents are multi-topical. Traditional clustering algorithms are able to achieve limited success in this direction. In this work, the topic-segmentation based split phase not only achieves soft clustering but also eliminates noise from the clusters. Another advantage of the work is that the proposed algorithm can be applied on the resultant of any query-document clustering algorithm.

The work might be further refined by introducing soft topic segmentation to find topic sets and to apply the SM algorithm on the resultants of other clustering algorithms. Furthermore, the clusters obtained can lead to build a knowledge base by incorporating more and more data into the clusters.

CHAPTER 06

Incremental Models for Query and Query-Context Aware Document Clustering

Most clustering algorithms are designed to work on previously collected data for query stream. These algorithms become less and less effective with time because users interests, query meaning and popularity of the topics etc. change or evolve over time. So, there is a need for incremental algorithms which can accommodate the concept drift that surface with new data being added to the previously collected data. In addition, the incremental models should be capable of handling the following: very large data sets within reasonable time and space, clustering documents according to query-context by overcoming the drawbacks due to noise in the click-through logs, producing results with accuracy close to static models, and producing results without manually specifying the parameters such as number of clusters, session duration and tuning parameter etc. To the best of our knowledge, only few incremental models have been proposed for query clustering in the literature. An incremental model proposed in (Huang et al. 2000) finds similarity between queries using query terms similarity among different query sessions. In this approach, only query content is used which may not be able to discover the information need of the user. An incremental model presented in (Gupta et al. 2010) forms query clusters by placing similar queries in respective clusters. Query similarity is measured based on query content (common query terms) and click-through information (common clicked URLs). Generally queries are short, imprecise and ambiguous and lacks in having common terms. Moreover, chances of clicking the same URLs even for the same query are less as search engine's results are different for different users at different times. Clicking on the same URLs is possible only for popular queries. Broccolo et al. (2010) measured query similarity based on cosine similarity between the query vectors which consists of clicked URLs with their frequencies. This will also suffers with the problems mentioned above.

It would be better to extend existing hierarchical clustering algorithms into incremental hierarchical clustering algorithms, which can update new information effectively to leverage hierarchical nature of web data. This research proposes an incremental model and its variation which is applied on hierarchical query clustering algorithms. We have considered both feed-back based hierarchical query clustering algorithm (Beeferman and Berger 2000) and content-and-feedback based hierarchical query clustering algorithms (Leung et al. 2008; Goyal et al. 2013).

In particular, models are applied and tested on three algorithms Query-Document (feedback based), Query-Concept (feedback-content based) and Query-Document-Concept (feedback-content based). These approaches produce query clusters, associated document clusters and associated concept/topical clusters.

A query and document may have more than one intention/topic. An IR system should be capable of relating a document/query with all its appropriate topics, rather than relating it with only a primary topic. This in turn leads to the requirement of soft clustering. However, clustering algorithms used in this thesis are traditional hard clustering algorithms. Therefore, we applied SM algorithm to obtain soft topical and topical document clusters. We have also applied proposed incremental models on the clusters obtained after applying SM algorithm.

The main contribution of this chapter is as follows:

- An efficient incremental model along with a variation for query and document clustering has been proposed. The model, applicable to the resultant clusters of hierarchical clustering algorithms, updates query and document clusters at different time stamps and produces clusters that are very close to the respective clusters obtained by re-clustering on the entire data set (static model). The proposed incremental model is also applied to update topical and topical document soft clusters obtained by SM algorithm.

The rest of the chapter is organized as follows: In Section 6.1, details of existing query clustering and document clustering approaches are presented. Then, we have presented the proposed incremental clustering models in Section 6.2, Experimental setup and datasets used are described in Section 6.3. Results are presented in Section 6.4 and concluding remarks and future research directions are given in Section 6.5.

6.1 Related Work

Query clustering is an effective method for finding similar queries. Similar queries are found by adopting string matching features in (Wen et al. 2001). Content based approach for finding similar queries has been proposed by Zaiane and Strilets (2002). A method to predict a list of related queries by mining association rules from the logs of submitted queries to search engine has been proposed (Fonseca et al. 2003). Content of user's historical preferences that are recorded in query logs is used to describe the semantic meaning of the current query (Baeza-Yates et al. 2004b, 2007). Click-through data is clustered by iteratively merging two most

similar queries followed by merging two most similar URLs using bipartite graph and agglomerative clustering (Beeferman and Berger 2000). However, this method is based on only the feedback by ignoring the content. The feedback and content based hierarchical query clustering method is proposed by Goyal (Goyal et al. 2013) and Leung (Leung et al. 2008). Query similarity finding using click-through data is also addressed in (Dupret and Mendoza 2005). However the main focus is on document ranking. Similar queries are also found by 1) analyzing users' sessions, 2) classifying type of query reformulation and 3) deriving query-flow graphs (Boldi et al. 2009). Click-based clustering is applied to session-based similar query finding and claimed that the context awareness helps to find more similar queries and to better understand user's search intent (Cao et al. 2010; Sengstock and Gertz 2011). However, they did not comprehensively evaluate if the context awareness really improves the similarity finding utility mainly due to the lack of an adequate baseline.

Document clustering methods have been investigated by many researchers. Common document clustering techniques have been described in (Steinbach et al. 2000). They have presented the results of an experimental study of some common document clustering like hierarchical clustering algorithm and K-means clustering algorithm. An exhaustive survey of web document clustering approaches, by classifying the approaches into three main categories: text-based, link-based and hybrid, can be found from (Oikonomakou and Vazirgiannis 2006). Furthermore, they have presented a thorough comparison of the algorithms based on the various facets of their features and functionality. Finally, based on the review of the different approaches, it has been concluded that although clustering has been a topic for the scientific community for three decades, there are still many open issues that call for more research. Once the query-specific keywords have been extracted from the retrieved documents, they can be used to group the relevant documents. It has been shown (Tombros 2002) that query-specific clustering is more effective than traditional clustering.

An incremental clustering process by introducing a tree structure called the DC tree is proposed (Wong and Fu 2000). Incremental document clustering using cluster similarity histograms which relies only on pair-wise document similarity has been proposed (Hammouda and Karnel 2003). Enhanced incremental methods have been proposed (Shaw and Xu 2009) for web pages categorization which is an extension of the work proposed in (Hammouda and Karnel 2003).

An incremental query clustering approach proposed in (Huang et al. 2000) suggests terms to a user's query. These terms are extracted from similar query sessions rather from the content of the retrieved documents. Architecture proposed for incremental query clustering (Gupta et al. 2010) introduces methods that run online as well as offline to detect data duplication based on the used queries. An incremental query clustering approaches based on association rules and cover graph have been proposed Broccolo et al. (2010) and evaluated query recommendation algorithms using the effects of continuous model updates.

6.2 Incremental Clustering Models

In this section, we have presented the proposed incremental clustering model and its variation. The proposed model does not impose any additional constraints on hierarchical clustering algorithms. So, any existing hierarchical clustering algorithm can be adapted. The proposed incremental model is shown in Figure 6.1 and described below:

Step1. Clustering algorithm is applied on the data collected at time t_i. Set of clusters are obtained as per the algorithm.

Step2. New batch of data is received at time t_{i+1} and is clustered separately by the algorithm.

Step3. The two sets of clusters from step1 and step2 are appended and then clustered again by the same algorithm. Resultant set of clusters are obtained at time t_{i+1}. Now the model is ready to receive the next batch of data.

Variation in step 2 is introduced to come up with two incremental models, viz., IMC (Incremental Model with Clustered Data) and its variation IMR (Incremental Model with Raw Data). The two models are same except that IMR considers individual query as individual unit clusters in the step 2 in contrast to IMC.

It is evident from the experiments that the incremental results of both IMC and IMR are very close to that of static model. It also reduces the processing time many folds because clustering time for individual batch is less and merging two sets of clusters with clustering them again is very efficient as compared to clustering the entire data (see Section 6.4).

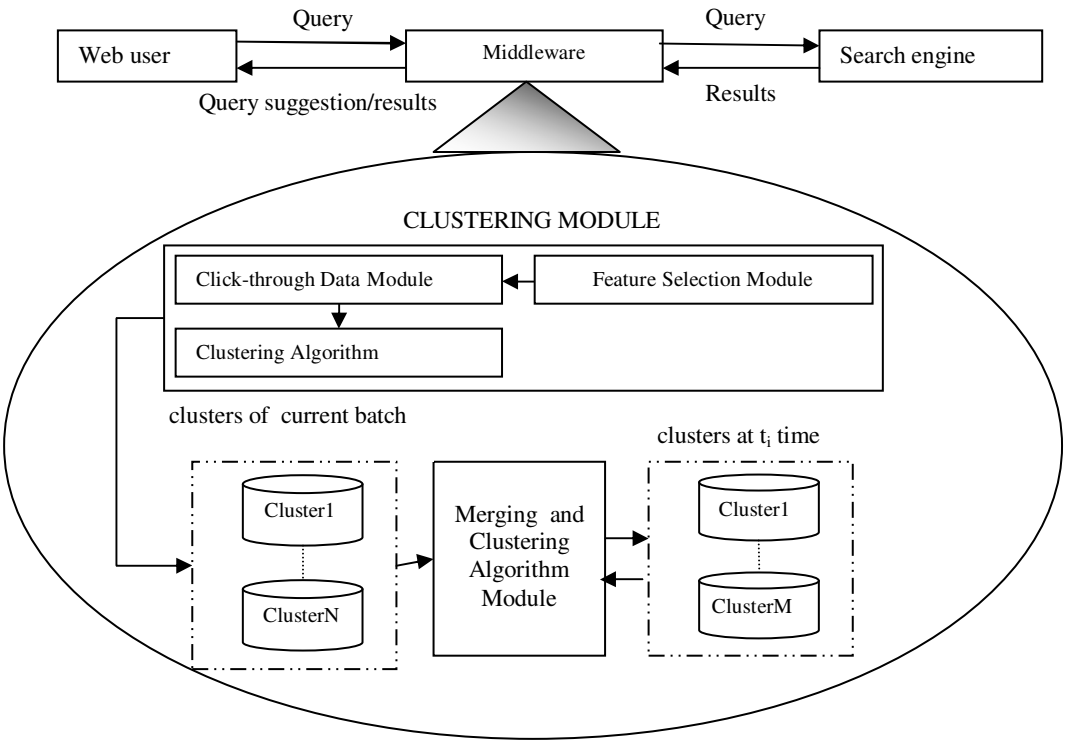


Figure 6.1. The Proposed Incremental Models

6.2.1 Complexity Analysis

We, now give the time complexity analysis of the two incremental models with QDC clustering algorithm. The time complexity with other two algorithms, QD and QC, is same as that of the QDC algorithm. The cost of an agglomerative clustering algorithm is $O(n^2)$ where n is the number of the points (Sibson 1973).

Let n_i and m_i be the number of queries and concepts in i^{th} batch data. Let k_{n_i} and k_{m_i}be the number of query and concept clusters generated by the incremental models after i^{th} iteration respectively.

Cost of processing i^{th} batch of data alone is $O({n_i}^2 + {m_i}^2)$. (6.1)

Cost of Incremental Models:

In model IMR, each object of new data forms its own cluster and added to the clusters obtained at the current time.

Cost of appending $(i+1)^{th}$ batch of clusters to the set of current clusters is O(1).

Now we derive the time complexity of the model by induction as follows:

Cost of processing first batch of data = O(n_1^2+ m_1^2) (6.2)

Cost of appending second batch of clusters to the first batch of clusters = O(1) (6.3)

Cost of clustering of the appended result = $O\left(\left(k_{n_1} + n_2\right)^2 + \left(k_{m_1} + m_2\right)^2\right)$

Since $n_2^2 >> k_{n_1}^2$ and $n_2^2 \gg k_{n_1} n_2$ and similarly, $m_2^2 >> k_{m_1}^2$ and $m_2^2 >> k_{m_1} m_2$, the cost of clustering of the appended result = O($n_2^2 + m_2^2$) (6.4)

Total cost after processing second batch of data = O($n_1^2 + m_1^2$) + O($n_2^2 + m_2^2$) (6.5)

Assume that the cost after i^{th} iteration

=O($n_1^2 + n_2^2 + \ldots\ldots + n_i^2$) + O($m_1^2 + m_2^2 + \cdots\ldots + m_i^2$) (6.6)

Cost of appending $(i+1)^{th}$ batch clusters to i^{th} iteration clusters is O(1). Cost of clustering of the appended data = $O\left(\left(k_{n_i} + n_{i+1}\right)^2 + \left(k_{m_i} + m_{i+1}\right)^2\right)$ (6.7)

Since $n_{i+1}^2 >> k_{n_i}^2$ and $k_{n_i} n_{i+1}$ and $m_{i+1}^2 >> k_{m_i}^2$ and $k_{m_i} m_{i+1}$

Cost of clustering of the appended data = O(n_{i+1}^2+ m_{i+1}^2) (6.8)

Total cost of IMR model with QDC algorithm after $(i+1)^{th}$ iteration is thus:

=O($n_1^2 + n_2^2 + \ldots\ldots + n_i^2 + n_{i+1}^2$) + O($m_1^2 + m_2^2 + \cdots\ldots + m_i^2 + m_{i+1}^2$) (6.9)

In IMC model, every batch is separately clustered. Let L_{n_i} and L_{m_i} be the number of query and concept clusters generated by the incremental model for i^{th} batch data.

Cost of processing first batch data separately=O($n_1^2 + m_1^2$) (6.10)

Cost of processing 2nd batch data separately=O($n_2^2 + m_2^2$) (6.11)

Cost of appending second batch clusters to the set of current clusters is O(1) (6.12)

Cost of clustering of the appended data = $O\left(\left(k_{n_1} + L_{n_2}\right)^2 + \left(k_{m_1} + L_{m_2}\right)^2\right)$ (6.13)

Cost after second iteration (add (6.10), (6.11), (6.12) and (6.13) and simplify)

= O($n_1^2 + m_1^2$) + O($n_2^2 + m_2^2$) (6.14)

Similar to IMR model, the cost of IMC model after $(i+1)^{th}$ iteration will be given by Equation 6.9.

Cost of static model

Cost of static algorithm for processing first batch data = O($n_1^2 + m_1^2$).

Cost of static algorithm for processing first and second batch data

=O($(n_1 + n_2)^2 + (m_1 + m_2)^2$)

After i^{th} iteration cost

$=O((n_1 + n_2 + \ ... + n_i)^2 + (m_1 + m_2 + \ ... + m_i)^2)$, comparing it with Equation 6.9, it is much greater than the cost of incremental models after i^{th} iteration.

6.3 Experimental Setup and Data Sets

The proposed method is applied on the resultant of existing clustering algorithm discussed which is capable of producing concept/topical clusters and the associated query and document clusters. So we have used the same data sets described in chapter 5. We have used two datasets for evaluation: 1. TREC 2011 session track data set (Jiang et al. 2011) and 2. Data set which is used in (Goyal and Mehala 2011; Goyal et al. 2013). Three datasets (DS1, DS2 and DS3) are constructed from the two datasets as mentioned in the previous chapter for experiments. DS1 consists of full TREC 2011 session track dataset. Some of the TREC data set queries were singleton that make clusters of size one. Therefore, all the clustering algorithms may give same results as it suppresses the meaning of clustering. For this reason, we have removed queries that form clusters of size one and two from the TREC 2011 session track dataset and formed the dataset DS2. DS3 is the third dataset which is formed by combining DS2 and the Data Set used in (Goyal and Mehala 2011; Goyal et al. 2013) to verify the effect of the proposed model on large dataset and as these two datasets also have some queries of same intention in common. For example, queries "Guide" and "tax" in (Goyal and Mehala 2011; Goyal et al. 2013) are having similar intention as that of the queries "AMT", "AMT and Tax" in DS2. The statistics of the data collected for concept extraction and clustering experiments for the datasets DS1, DS2 and DS3 are given in chapter 5. The statistics on the basis of result of clustering algorithm QD, QC and QDC is given in Table 5.3 in chapter 5.

We have partitioned the datasets DS1 and DS3 into five partitions (P_1, P_2, P_3, P_4 and P_5) and DS2 into three partitions (P_1, P_2 and P_3) randomly for the incremental model evaluation. We have partition DS2 into three partitions because its smaller size. All the partitions have been formed randomly i.e. by selecting data in a unit-information (query with its clicked documents irrespective of session and topic information) randomly.

6.4 Experimental Results

The performance of the proposed incremental model has been analyzed for different hierarchical clustering algorithms QD, QC, QDC, SMQC, and SMQDC on three datasets DS1, DS2, and DS3. We have named all the algorithms for static model as QDS, QCS, QDCS, SMQCS, and SMQDCS and for incremental models IMC and IMR as QDI, QCI, QDCI, SMQCI, SMQDCI and QDIR, QCIR, QDCIR, SMQCIR, SMQDCIR respectively.

We have considered three measures precision, recall and F-measure (described below) to evaluate the performance of the proposed model and its variation. These measures are calculated for the query, query based document, topical and topical document clusters obtained from various algorithms and models and then compared with our predefined clusters of queries and documents. The predefined clusters are manually formed by placing similar intend queries, similar intend documents and similar intent concepts into respective clusters, as there is no gold standard cluster information is available for any existing standard datasets.

6.4.1 Evaluation Measures

The following are the measures used for comparison of the results obtained by the different models and algorithms:

The precision (*P*), recall (*R*), and F-measure (*F*) for a given query *q* (or a given document *d or a given concept c*) and associated query cluster q_1, q_2,...,q_n (or document cluster d_1,d_2,...,d_m or topical cluster C_1,C_2,...,*Cl*) produced by a clustering algorithm, are computed using the following formulae respectively:

$$P(x) = \frac{|S \cap S'|}{|S|} \quad R(x) = \frac{|S \cap S'|}{|S'|} \quad F = \left(2.\frac{(P.R)}{(P+R)}\right) \tag{6.15}$$

Where *x* can take two values, either a query '*q'* or a document '*d' or a concept 'c'*. *S* is the set of queries (or documents/concepts) that exist in the predefined cluster for *q* (or *d/c*) and S' is set of the related queries q_1, q_2,...,q_n (or documents d_1,d_2,...,d_m */ concepts* c_1,c_2,...,c_l) generated by the algorithm. The precision and recall values from all queries (documents) are averaged.

Most of the experimental results are presented in terms of best mean precision, recall and F-measure because every algorithm performs best at a different value of cut-off-similarity score σ. For a particular value of σ, one algorithm may work the best whereas other algorithm can be worst.

6.4.2 Results: Query Clustering

Static Models

Before presenting the results for the incremental models, we present results of performance of hierarchical clustering algorithms QD, QC and QDC. The experiments have been conducted for different cut-off similarity scores σ. Best mean precision, recall, and F-Measure values of the three algorithms for the data sets DS1, DS2 and DS3 are given in Table 6.1.

Table 6.1. Best mean precision, recall and F-Measure of all algorithms for static datasets

	DS1			DS2			DS3		
	P	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
QD	0.990	0.623	0.764	1	0.400	0.535	0.949	0.425	0.567
QC	0.871	0.730	0.794	0.842	0.506	0.632	0.878	0.641	0.741
QDC	0.969	0.747	0.843	0.871	0.727	0.793	0.934	0.727	0.817

It is observed that QDC algorithm has achieved both higher precision and recall than others. QD algorithm gives better precision but low recall. This is due to less or no common click-documents in click-through log thereby forming a large number of small clusters. QC algorithm gives low precision values but recall values lie between that of QD and QDC algorithms. It can be observed from Table 6.1 that all the algorithms give better precision and recall for DS1. This is because of presence of many queries that form clusters of size one/two. DS1 has 44 singleton queries and 24 queries that form clusters of size two out of a total of 121 queries.

Percentage improvement in F-Measure of QDC algorithm over others for different values of σ is calculated and plotted in figure 6.2a and 6.2b for datasets DS2 and DS3 respectively. Improvement is up to 44% and 23% in Figure 6.2a and is up to 40% and 20% in Figure 6.2b over QD and QC respectively. It is clear that performance of QDC algorithm is remarkable for all the datasets considered.

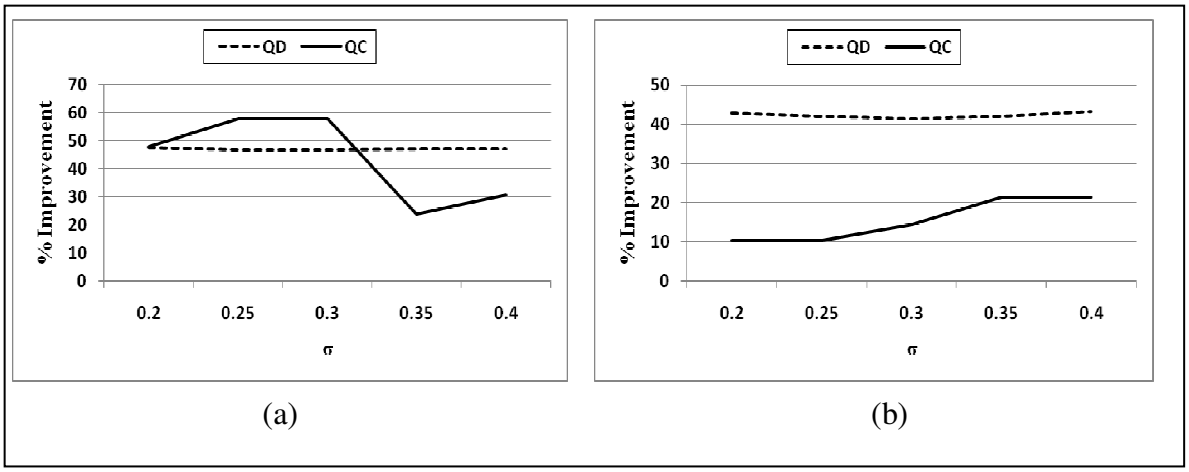


Figure 6.2. % Improvement in mean F-measure of QDC over QD and QC in static model on (a) DS2 and (b) DS3

Incremental Models

We now present results for the performance of the incremental models IMC & IMR. We have named all the algorithms for static model as QDS, QCS, and QDCS and for incremental models IMC and IMR as QDI, QCI, QDCI and QDIR, QCIR, QDCIR respectively.

Table 6.2 presents the best precision, recall and F-measure values obtained by cluster analysis over the dataset DS2 (partitions P_1, P_2, P_3). These are the best values at some value of σ. The incremental model is applied on DS2 two times for addition of P_2 and P_3 and results are presented in Table 6.2. It is observed that the incremental models give the measure values very close to that of static ones. QD results have not been presented for DS2 as the data set does not have enough common clicked documents. As a result, clusters are not formed and the results obtained by incremental models are same as that of static model (see Table 6.1).

Table 6.2. Best mean Precision, Recall and F-Measure of different algorithms for DS2

	P_2			P_3		
	P	R	F	P	R	F
QCS	0.556	0.611	0.582	0.842	0.506	0.632
QCI	0.695	0.500	0.582	0.889	0.456	0.603
QCIR	0.675	0.524	0.590	0.867	0.468	0.608
QDCS	0.913	0.630	0.746	0.917	0.693	0.789
QDCI	0.923	0.596	0.724	0.951	0.665	0.783
QDCIR	0.913	0.618	0.737	0.918	0.674	0.777

Precision, Recall and F-Measure values of all models for QC and QDC algorithms are also calculated for the dataset DS3. The percentage change in F-measure for incremental model against static model is plotted and presented in Figure 6.3. Both IMC and IMR, when applied to QDC are performing as good as the static model or even better than the static model except for P_5, where it is slightly lower than static model. Moreover, IMC and IMR are showing deterioration of up to 6 and 8 percent respectively, when applied to QC algorithm, whereas, more

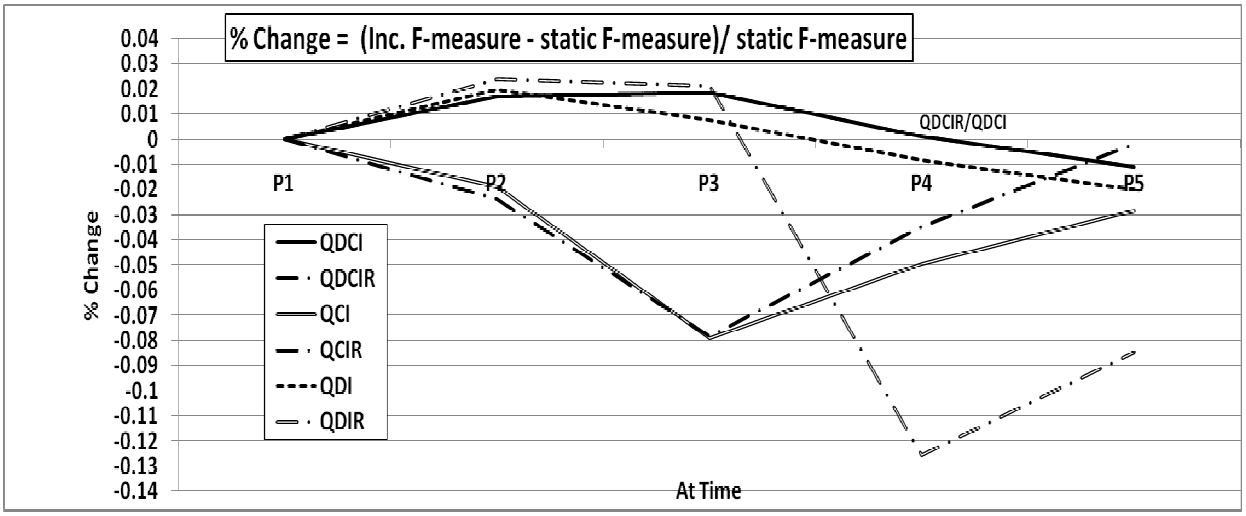


Figure 6.3. % Change in F-measure for different incremental models over respective static models for DS3

fluctuations are observed in QD. The results for DS1 are given in Table 6.3. In general, QDC static and incremental models give higher precision and recall compared to QD and QC. The deviation in precision and recall of incremental models with the static model is more in comparison to DS2. This is because DS1 has less clicked documents per-query. The models are performing better if data has sufficient clicked information.

It is clear from the results that both the incremental models, IMC and IMR, are performing close to the static model. Recall is getting better in the subsequent additions if the number of singleton query clusters is less. Experiments also show that the QDC algorithm is best among the three algorithms and the same is true for its corresponding incremental models.

In the next experiment, we have applied QCI and QDCI on different order of partitions. The results are presented in Table 6.4. It can be seen that QDCI is order independent unlike QCI. It shows that if the underlying algorithm is robust then the order of addition of the data/data clusters does not matter. Similar patterns are observed for DS1 and DS3.

Table 6.3. Best mean Precision, Recall and F-Measure values of different algorithms for DS1

Method	*P2*			*P3*			*P4*			*P5*		
(P,R,F) at P1	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
QDS (0.885,0.827,0.855)	0.955	0.846	0.897	0.951	0.778	0.856	0.929	0.668	0.778	0.99	0.623	0.764
QDI	1	0.788	0.882	0.974	0.735	0.838	0.962	0.643	0.770	0.969	0.567	0.716
QDIR	0.962	0.837	0.895	0.942	0.765	0.844	0.923	0.659	0.769	0.935	0.590	0.724
QCS (0.846,0.865,0.856)	0.824	0.923	0.871	0.773	0.823	0.797	0.776	0.764	0.770	0.871	0.730	0.794
QCI	0.904	0.853	0.877	0.843	0.774	0.807	0.795	0.670	0.727	0.824	0.584	0.684
QCIR	0.801	0.824	0.812	0.744	0.756	0.750	0.702	0.655	0.677	0.716	0.586	0.645
QDCS (0.885,0.788,0.834)	0.981	0.859	0.916	0.929	0.848	0.887	0.899	0.7800	0.836	0.969	0.747	0.843
QDCI	0.931	0.817	0.870	0.902	0.767	0.829	0.873	0.661	0.753	0.865	0.635	0.732
QDCIR	0.923	0.837	0.878	0.880	0.780	0.827	0.862	0.693	0.768	0.810	0.649	0.720

Table 6.4. Addition of partitions in different order for DS2

		P_1-P_2	P_3-P_1	P_2-P_3	P_1-P_2-P_3	P_3-P_1-P_2	P_2-P_3-P_1
QCI	**P**	0.695	0.906	0.767	0.889	0.824	0.788
	R	0.500	0.541	0.523	0.456	0.453	0.384
	F	0.582	0.677	0.622	0.603	0.584	0.517
QDCI	**P**	0.923	0.982	0.955	0.951		
	R	0.596	0.545	0.693	0.665		
	F	0.724	0.701	0.803	0.783		

Experiments have been conducted for the different number of partitions of the dataset and results are presented after applying incremental models on all partitions. Best precision, recall, and F-Measure of all QDC models with five and three partitions on dataset DS1 are presented in Table 6.5. The results illustrate that we obtain better recall and precision when the partition sizes are bigger.

Table 6.5. Performance of QDC models for different number of partitions on DS1

	On Five Partition			**On three Partition**		
	P	**R**	**F**	**P**	**R**	**F**
QDCS	0.969	0.747	0.843	0.969	0.747	0.843
QDCI	0.865	0.635	0.732	0.933	0.641	0.760
QDCIR	0.810	0.649	0.720	0.930	0.685	0.789

6.4.3 Results: Query-Context Based Document Clustering

Static Models

We have also carried out experiments for query-context based document clustering by different algorithms using different models. We have examined the quality of document clusters generated by QD, QC and QDC in both static and incremental scenarios. Table 6.6 compares the best mean precision, recall and F-measure of document clusters generated by three algorithms (static) with cutoff similarity score σ. It is clear from the table 6.6 that the QDC performs best on DS3 for document clustering too. Similar results were found for other datasets also. Percentage improvement in F-Measure of document clustering by QDC over other algorithms for different values of σ shows that the improvement is upto 32% and 25% of QDC over QD and QC respectively.

Table 6.6. Best mean Precision, Recall and F-Measure of document clustering on DS3

	P	**R**	**F**	**Threshold σ**
QD	0.678	0.718	0.697	0.35
QC	0.727	0.777	0.751	0.40
QDC	1	0.835	0.914	0.20

Incremental Models

Table 6.7 presents best mean precision, recall, and F-Measure values of document clustering by incremental models at different time stamps (partitions) for dataset DS3. The precision and recall of document clusters generated by QDC algorithm is higher than that of QD and QC throughout. Moreover, values of all the measures for incremental models are very close to that of the static model. Similar results were observed for other datasets.

It is also observed that both recall and precision values are increasing as we process partitions for QDCI and QDCIR. This shows that the incremental models have the capability to improve/retain the quality of clusters when applied to large datasets.

Table 6.7. Best mean Precision, Recall and F-Measure values of different algorithms for document clustering on DS3

Method	*P2*			*P3*			*P4*			*P5*		
(P,R,F) at P1	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
QDS (0.651,0.97,0.779)	0.683	0.866	0.764	0.688	0.798	0.731	0.673	0.763	0.715	0.671	0.716	0.693
QDI	0.675	0.867	0.759	0.648	0.774	0.706	0.651	0.744	0.694	0.64	0.684	0.662
QDIR	0.675	0.867	0.759	0.648	0.774	0.706	0.648	0.747	0.694	0.656	0.669	0.662
QCS (0.601,0.885,0.716)	0.68	0.896	0.773	0.738	0.829	0.781	0.731	0.734	0.733	0.727	0.777	0.751
QCI	0.672	0.872	0.759	0.698	0.812	0.751	0.702	0.783	0.74	0.709	0.685	0.697
QCIR	0.671	0.825	0.74	0.706	0.783	0.743	0.718	0.75	0.734	0.729	0.676	0.702
QDCS (0.949,0.788,0.861)	0.963	0.816	0.882	0.964	0.83	0.892	0.966	0.823	0.889	1	0.835	0.912
QDCI	0.947	0.808	0.872	0.955	0.813	0.879	0.965	0.807	0.879	0.963	0.798	0.873
QDCIR	0.956	0.805	0.874	0.956	0.813	0.88	0.967	0.807	0.879	0.963	0.775	0.859

6.4.4 Results: Soft Topical Clustering

We will now present the performance of the proposed incremental model when applied on different algorithms and data sets. We have named algorithms for static model as QCS, QDCS, SMQDCS and SMQCS and for the incremental model (IMC) as QCI, QDCI, SMQDCI and SMQCI respectively.

Table 6.8 presents the best precision, recall and F-measure obtained for topical clusters of DS1, DS2 and DS3 with SM algorithm. While, results without SM algorithm are given for DS3 in Table 6.9. These are the best values for an algorithm at some value of σ.

The incremental model is applied on DS1 and DS3 four times (at t_2, t_3, t_4 & t_5) for addition of partitions P_2, P_3, P_4 and P_5 in the existing partition P_1 at t_1. Whereas, the incremental model is applied on DS2 two times (at t_2 & t_3) for addition of partitions P_2 and P_3. It is observed that the incremental model gives the measure values very close to that of the static ones. The deviations in F-measure for datasets DS1, DS2 and DS3 using SMQDCI are 0.02, 0.023 and 0.062 respectively.

We have also conducted and compared the performance of the incremental model with and without applying SM algorithm. It is observed that the proposed incremental model with SM algorithm is showing better performance on all datasets. We have shown in Table 6.9 the performance of the incremental model on different clustering algorithms, i.e. without SM algorithm for DS3. SMQDCS shows improvement over QDCS in terms of precision by 0.179 and recall by 0.174. Similarly, SMQDCI shows improvement by 0.151 and 0.125 in precision

and recall respectively over QDCI. This improvement of SMQCI over QCI is more because of the clustering algorithm QC. Further, SM algorithm gives better performance while applying the algorithm on the clusters instead of applying the same on raw input data. The similar behavior is observed on other dataset as well.

Table 6.8. Sample intermediate results of topical clustering for SMQDC on DS3 at different split-thresholds and merge-thresholds

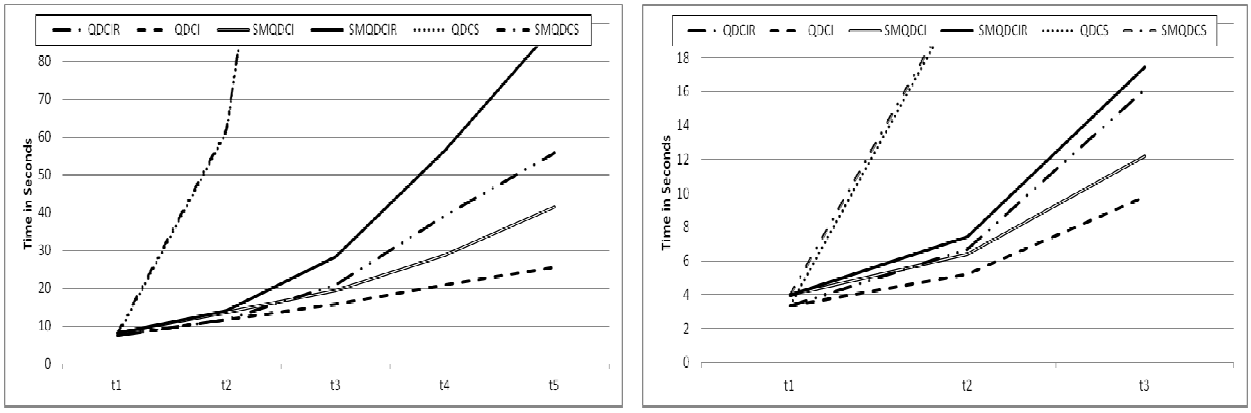
Data Set	Methods (P,R,F) at t1	t2			t3			t4			t5		
		P	R	F	P	R	F	P	R	F	P	R	F
DS1	SMQDCS (0.983, 0.917, 0.949)	0.979	0.815	0.890	0.960	0.843	0.898	0.965	0.843	0.900	0.965	0.751	0.844
	SMQDCI	0.969	0.827	0.892	0.939	0.822	0.876	0.912	0.840	0.874	0.926	0.743	0.824
	SMQCS (0.839, 0.956, 0.894)	0.790	0.865	0.826	0.776	0.840	0.807	0.766	0.863	0.811	0.819	0.802	0.811
	SMQCI	0.889	0.830	0.859	0.845	0.808	0.826	0.842	0.808	0.825	0.862	0.707	0.777
DS2	SMQDCS (0.806, 0.863, 0.833)	0.817	0.812	0.814	0.890	0.826	0.857						
	SMQDCI	0.867	0.825	0.846	0.882	0.792	0.834						
	SMQCS (0.750, 0.886, 0.812)	0.742	0.844	0.790	0.888	0.821	0.853						
	SMQCI	0.789	0.818	0.804	0.821	0.768	0.793						
DS3	SMQDCS (0.882, 0.915, 0.898)	0.940	0.890	0.914	0.932	0.823	0.874	0.920	0.822	0.869	0.944	0.836	0.887
	SMQDCI	0.920	0.893	0.906	0.913	0.798	0.851	0.892	0.808	0.848	0.885	0.773	0.825
	SMQCS (0.888, 0.927, 0.907)	0.910	0.828	0.867	0.870	0.782	0.824	0.861	0.816	0.838	0.816	0.790	0.803
	SMQCI	0.882	0.891	0.886	0.875	0.777	0.823	0.869	0.815	0.841	0.877	0.792	0.832

Table 6.9. Best precision, recall and F-Measure for topical Clustering by QDC and QC on static and IM models over DS3

Data Set	Methods (P,R,F) at t1	t2			t3			t4			t5		
		P	R	F	P	R	F	P	R	F	P	R	F
DS3	QDCS (0.637, 0.769, 0.697)	0.722	0.771	0.746	0.750	0.708	0.728	0.756	0.690	0.722	0.765	0.662	0.710
	QDCI	0.716	0.773	0.743	0.721	0.698	0.709	0.731	0.685	0.707	0.734	0.648	0.688
	QCS (0.202, 0.960, 0.334)	0.223	0.965	0.362	0.262	0.881	0.403	0.248	0.866	0.386	0.238	0.846	0.371
	QCI	0.219	0.942	0.356	0.248	0.867	0.386	0.228	0.872	0.361	0.200	0.834	0.323

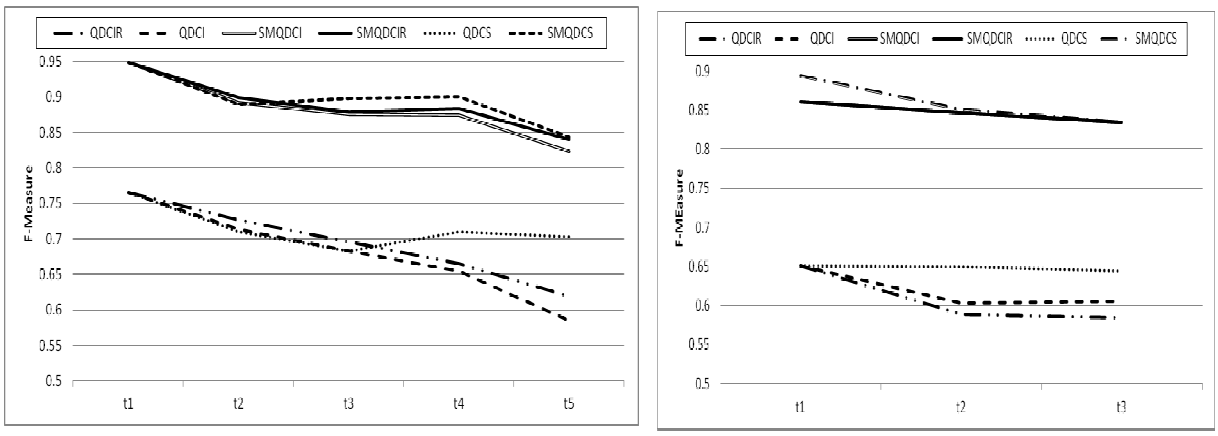
We report the comparison of time taken by the incremental model and the static model in Figure 6.4a & 6.4b for the datasets DS1 and DS2 respectively for SMQDC algorithm. It can be seen that the time taken by the static model is as high as 5 times, 16 times, 39 times and 63 times higher than that of the IM model at the time stamps t_2, t_3, t_4 and t_5 respectively. We haven't presented the time taken by the static model in the figure for all timestamps for clarity. The incremental model's time includes all individual clustering and merging. It is also evident from the figures that the time taken by the incremental model IM is much lesser than that of static and the time taken by SMQDCI is greater than QDCI. However, the SM incremental model shows the better performance with the little compromise in time.

We also report a percentage increase in F-measure of SMQDCI against QDCI as 18.3%, 17.9%, 19.3%, 21.9% and 23.9% on DS1 (see Figure 6.5a). Percentage increase in F-measure of SMQDCS against QDCS is also measured as 18.3%, 18%, 21.5%, 19% and 14.1% at different timestamps t_1 to t_5 on DS1. Similar behaviour is observed and shown in Figure 6.5b for DS2.



(a) (b)

Figure 6.4. Time taken by different algorithms for (a) DS1 and (b) for DS2



(a) (b)

Figure 6.5. F-measure of different algorithms for (a) DS1 and (b) for DS2

6.4.5 Results: Soft Topical Document Clustering

We will now present the results of evaluation of topical document clustering. We have examined the quality of topical document clusters generated by QDC, QC, SMQC and SMQDC with that of predefined clusters in both static and incremental scenarios.

Table 6.10 presents the best mean Precision, Recall and F-Measure values of topical document clustering by the incremental model at different time stamps for all the datasets considered with SM algorithm. While, results without SM algorithm are given for DS3 in Table 6.11. The precision and recall of topical document clusters generated by SMQDC algorithm are higher than that of the other algorithms. The best F-measure value achieved by SMQDC on the data set DS1, DS2 and DS3 is 0.882, 0.891 and 0.944 respectively. However, SMQDC performance can further be improved by incorporating soft topic set segmentation method in split and merge process. Moreover, all the measure values of incremental model are very close to the values of static model. We have also measured the deviation in F-measure achieved through incremental model with that of their respective static models. In case of on SMQDCS & SMQCS, deviation

on DS1: from 0 to 0.06 & from -0.023 to 0.026 respectively, on DS2: from -0.016 to 0.009 & from 0 to 0.066 respectively and on DS3: from 0 to 0.079 & from -0.024 to 0.009 respectively.

Table 6.10. Best precision, recall and F-Measure for Topical Document Clustering from SMQDC and SMQC on static algorithms and IM model

Data Set	Methods (P,R,F) at t1	t2			t3			t4			t5		
		P	R	F	P	R	F	P	R	F	P	R	F
DS1	SMQDCS (1,0.869,0.930)	1	0.847	0.917	0.965	0.863	0.911	0.967	0.859	0.910	0.965	0.813	0.882
	SMQDCI	0.984	0.839	0.906	0.974	0.816	0.888	0.942	0.805	0.868	0.929	0.737	0.822
	SMQCS (0.929,0.875,0.901)	0.897	0.868	0.882	0.878	0.833	0.855	0.862	0.821	0.841	0.881	0.782	0.829
	SMQCI	0.946	0.835	0.887	0.919	0.802	0.857	0.856	0.778	0.815	0.866	0.728	0.791
DS2	SMQDCS (1,0.921,0.959)	0.964	0.854	0.905	0.938	0.820	0.875						
	SMQDCI	0.962	0.863	0.910	0.977	0.818	0.891						
	SMQCS (0.869,0.867.0.868)	0.903	0.853	0.877	0.950	0.857	0.901						
	SMQCI	0.862	0.831	0.846	0.900	0.796	0.845						
DS3	SMQDCS (0.913,0.899,0.906)	0.968	0.930	0.949	0.956	0.894	0.924	0.947	0.879	0.912	0.967	0.923	0.944
	SMQDCI	0.947	0.923	0.935	0.937	0.841	0.886	0.927	0.832	0.877	0.910	0.828	0.867
	SMQCS (0.928,0.871,0.899)	0.940	0.846	0.891	0.912	0.791	0.847	0.906	0.780	0.838	0.860	0.786	0.822
	SMQCI	0.924	0.845	0.883	0.931	0.769	0.842	0.938	0.765	0.843	0.935	0.772	0.846

Table 6.11. Best precision, recall and F-Measure for Topical Document Clustering from QDC and QC on static algorithms and IM model over DS3

Data Set	Methods (P,R,F) at t1	t2			t3			t4			t5		
		P	R	F	P	R	F	P	R	F	P	R	F
DS3	QDCS (0.900, 0.793, 0.843)	0.889	0.800	0.842	0.914	0.748	0.822	0.925	0.750	0.828	0.916	0.732	0.814
	QDCI	0.869	0.824	0.846	0.882	0.755	0.813	0.896	0.742	0.811	0.894	0.733	0.806
	QCS (0.325, 0.925, 0.481)	0.304	0.951	0.461	0.348	0.869	0.497	0.367	0.816	0.506	0.301	0.815	0.439
	QCI	0.292	0.929	0.444	0.325	0.849	0.470	0.344	0.822	0.485	0.283	0.796	0.418

We have also compared the proposed incremental model with its variation where, raw data is appended in place of appending clusters. Then clustering algorithm followed by SM algorithm is then applied and final set of clusters are obtained. This model is referred as IMR. We have named algorithms for incremental model IMR as QCIR, QDCIR, SMQDCIR and SMQCIR for QCS, QDCS, SMQDCS and SMQCS respectively. Table 6.12 and 6.13 present best mean Precision, Recall and F-Measure values of topical clustering and topical document clustering by

incremental model IMR respectively with and without SM algorithm at different time stamps for the dataset DS3 as the behaviour of all algorithms and methods is same for other dataset as well. The precision and recall of topical and topical document clusters generated by SMQDCIR and SMQCIR algorithm are higher than that of the underlying clustering algorithms. SMQDCIR achieves better performance as compared to SMQCIR. The best F-measure value achieved by SMQDCIR for topical and topical document clustering on the data set DS3 is 0.837 and 0.865 respectively. The best F-measure value achieved by SMQDCI for topical and topical document clustering on the data set DS3 is 0.825 and 0.867 respectively. SMQDCIR shows comparable performance as compared to SMQDCI and also shows better performance in the scenario where ambiguous queries/concepts/documents are more. Time taken by the incremental model IMR is also given in Figure 6.4a & 6.4b for the datasets DS1 and DS2 respectively for SMQDC algorithm. We also report a percentage increase in F-measure of SMQDCIR against QDCIR in Figure 6.5a & 6.5b for the datasets DS1 and DS2 respectively.

Table 6.12. Best precision, recall and F-Measure for Topical Clustering from SMQDC, SMQC, QDC and QC on IMR model for DS3

Data Set	Methods	t2			t3			t4			t5		
		P	R	F	P	R	F	P	R	F	P	R	F
DS3	SMQDCIR	0.913	0.919	0.916	0.910	0.805	0.854	0.891	0.818	0.853	0.900	0.782	0.837
	SMQCIR	0.864	0.841	0.853	0.879	0.773	0.823	0.859	0.788	0.822	0.854	0.744	0.795
	QDCIR	0.698	0.768	0.731	0.702	0.695	0.698	0.714	0.681	0.697	0.714	0.641	0.675
	QCIR	0.220	0.946	0.356	0.244	0.869	0.381	0.216	0.873	0.346	0.197	0.836	0.319

Table 6.13. Best precision, recall and F-Measure for Topical Document Clustering from SMQDC, SMQC, QDC and QC on IMR model for DS3

Data Set	Methods	t2			t3			t4			t5		
		P	R	F	P	R	F	P	R	F	P	R	F
DS3	SMQDCIR	0.961	0.919	0.940	0.955	0.848	0.898	0.943	0.833	0.885	0.917	0.819	0.865
	SMQCIR	0.928	0.841	0.882	0.919	0.785	0.847	0.913	0.789	0.846	0.910	0.783	0.842
	QDCIR	0.898	0.809	0.851	0.908	0.743	0.817	0.923	0.738	0.820	0.919	0.715	0.804
	QCIR	0.292	0.932	0.445	0.325	0.849	0.470	0.337	0.823	0.479	0.278	0.800	0.413

6.5 Conclusions

In this chapter, we have proposed an incremental model and its variation for query, query-based document, topical and topical document clustering. The incremental models have been applied on three algorithms and evaluated against static models. The proposed models achieve accuracy as good as that of the static models for both query and document clustering. As expected, the time taken by the incremental approaches is much less than that of their static counterparts. The updation of clusters i.e. appending and then clustering merged data takes negligible time as compared to the clustering time in static mode. It is found that best results are obtained for the QDC algorithm among the three algorithms considered in the experiments. Query-context based document clustering approach clusters smaller group of relevant documents (relevant to query) by distinguishing irrelevant ones from search results. These clusters achieve high homogeneity.

The algorithms applied are conventional clustering algorithms that produce hard clusters (A query/document is placed in only one cluster). The work can be improved by using soft clustering instead of hard clustering. Soft clustering allows for a data point to be part of more than one cluster. So, we have applied SM algorithm on incremental models to convert hard to soft clusters and evaluated against static models. The proposed models also achieve accuracy as good as that of the static models for both soft topical and topical document clustering.

CHAPTER 07

Automatic Generation of Topic Hierarchy: A Query Context-Aware Approach

As the amount of information on the web is huge, dynamic and incremental, there is a need for organizing the information for effective and efficient information retrieval. Most popular ways to organize information are flat structure where semantic relationships between topics are not established or topic hierarchy structure which symbolizes relationships between categories/topics such that these topics are arranged from general or broader topics to more specific topics. A flat organization structure merely gives us a set of topics. A hierarchal organization structure goes a step further by endowing with how these topics are related to each other. This will help in easy navigation, efficient access to relevant data, maintaining and enriching the collection etc.

Hierarchy building can be a manual or an automatic process. The Open Directory Project (ODP) is one of the examples for manually built hierarchy/directory which allows us to browse and search information. Further, this is acting as a handy resource for the management of web content. Similarly, other manually built topic hierarchies are The WWW Virtual Library (VLIB 2015) and Yahoo (Yahoo 2015). However, building of hierarchy manually has the following issues: requires a lot of resources (domain experts, time and cost) and requires users' to have the same view about the topics and their relations as that of hierarchy creators (manual label is an indication of hierarchy creator view). So, automation of building the hierarchy is becoming an essential endeavor. The key challenges in this are: how to detect the "hidden topics" at the appropriate granularity?, how to evaluate the similarity and assign the semantic relations between these topics? and how to assign meaningful labels to the categories?.

Hierarchy building so far is based on either hierarchical clustering or based on the relation between categories where each category is represented by a term and then term relationship is established (Knijff et al. 2013). These methods have their own disadvantages: hierarchical clustering based hierarchy generation suffers with the problem of maintenance or periodic updates where hierarchy has to be built again. Term relationship based approach considers co-occurrence of terms in a document to find a relation. Term classification is difficult if the terms do not co-occur frequently in the documents and thus, this approach requires a large data to work with. Instead, in the hierarchy construction, a category can be represented by topical terms reflecting its conceptual meaning/intend. Thus, a relation between two categories can be

determined by describing the relations between their terms. Depending on the relationship established, the new category may be attached to the already existing hierarchy without rebuilding the structure of the master hierarchy.

In this chapter, we discuss a new approach based on the above idea to build a topic hierarchy. The hierarchy is query-context aware topic hierarchy which is built automatically and is able to integrate end-users' perceptions by letting the click through log grouped into topical clusters using query-context aware (query based topical clusters formed based on query similarities) agglomerative clustering approach. We mean query intend based topical clusters by query-context aware topics. The proposed approach involves users' feedback in the topical clusters formation to resolve the issue in the manually built hierarchy which insists end users' to have the same perception as that of hierarchy creators. Further, these topical clusters are refined by applying a post processing technique to produce soft and coherent clusters reflecting a single context. The proposed method also involves queries along with the terms of the categories, which reflect the topical context of the associated topical categories, in identifying the relationship between categories. We also generate and use the topic set, obtained by disambiguating query web search results, of each selected query to find its possible topics. Moreover, the approach is incremental as new categories/clusters can be inserted as and when they arrive based on the most recent information in the click-through logs. The categories in the hierarchy may become less cohesive due to the successive additions of new topics and/or documents. Therefore, a refinement approach is also proposed, which reorganizes the information into more topically cohesive categories. Our approach is based on both the viewpoints, i.e. hierarchical clustering and category relationship, and thus benefited from the advantages of both.

The main contributions of our work are:

1. A novel method to generate a topic hierarchy automatically by inserting topical clusters/categories iteratively into the current hierarchy.
2. An algorithm to refine the hierarchy in order to reorganize and to keep hierarchy more cohesive.
3. The proposed system also provides a query-context based procedure to prepare the data so that topical clusters and hierarchy categories can be compared and thus, relations can be established.

Two datasets have been prepared for the experiment: 1. data set which is prepared by taking the intersection of three standard data sets, i.e. AOL500k (AOL 2006), CABS120k08 (CABS 2015) and DMOZ (ODP 2015) data set, to get required data (referred as ODPDS) 2. data set obtained from the Indian Agricultural Statistics Research Institute (IASRI, Delhi, India) (referred as IASRIDS). The proposed approach is evaluated against a target hierarchy which is extended Open Directory Project (ODP) for both the data sets. The target hierarchy for the ODPDS is the ODP hierarchy. The ODP hierarchy is extended to get the target hierarchy for the data set ODPDS. We have added some categories which were missing in the ODP hierarchy in order to incorporate all the data. The target hierarchy for IASRIDS is prepared by research scientists of IASRI.

The rest of the chapter is organized as follows: Section 7.1 outlines the related work. In Section 7.2, hierarchy building methodology and refinement approach have been detailed. Experimental set up and results are illustrated in Section 7.3 and 7.4 respectively. Section 7.5 gives the concluding remarks and future directions.

7.1 Related Work

Information organization is one of the goals to improve search engine performance which can be accomplished either by using flat structure and hierarchical structure. Flat structure organizes the information as a set of topics/clusters. Distance based algorithm e.g. ISODATA (Ball and Hall 1965), K-Means (MacQueen 1967), Expectation-Maximization algorithm (Duda et al. 2000) and conceptual clustering algorithms e.g. ITERATE (Biswas et al. 1998) etc. are the popularly used algorithms to form a flat set of clusters. Query-context aware topical clustering methods are proposed to form flat clusters. Nguyen et al. (2009) and Scaiella et al. (2012) viewed topics as descriptive phrases of the respective document clusters and Wikipedia pages identified using topic annotators respectively for flat clustering. A flat and hierarchical search result clustering approaches are surveyed by Carpineto et al. (2009). However, flat structure is incapable of illustrating the structural relationship among the topics.

Organization of information in hierarchical structure shows semantic structural relationship among topics and can be done manually or automatically. DMOZ (ODP 2015), Yahoo! Directory (Yahoo 2015) and The WWW Virtual Library (VLIB) (VLIB 2015) etc. are the

examples of manually constructed web directories/hierarchies. Even though manually built hierarchy is accurate, it is continuous, time-consuming, tedious and costly process. This necessitates automatic hierarchy building which can be achieved through hierarchical clustering algorithms and term relationship based approaches (Knijff et al. 2013). Hierarchical clustering algorithms, which work on whole documents (Aggarwal and Zhai 2012) and those which work on web snippets (Ferragina and Gulli 2008), are surveyed. Different users may have different perceptions about the categories in the hierarchy. But, manually built hierarchy insists end users' to realize the view of hierarchy creators. This issue can be resolved by considering users' feedback in the hierarchy building. Some of the hierarchical clustering algorithms use users' feedback to cluster documents (Beeferman and Berger 2000; Goyal et al. 2013; Leung et al. 2008). Goyal et al. (2013) compare the performance of the hierarchical agglomerative clustering algorithms which produce topical clusters which are formed based on the query similarity that use click-through log as users' feedback.

Relationship based automatic hierarchy building uses either distance or statistical based measures to establish relations. In distance based approaches, each document is represented as a vector of a set of terms (Salton et al. 1996). Clusters are formed by grouping similar vectors. These approaches are not naturally incremental as they require similarities be re-computed and similarity thresholds are to be updated periodically. A term subsumption method, a statistical method, treats certain frequently occurring terms in documents as concepts/categories and provides a probabilistic way to find the relationship between pairs of terms to build a hierarchy (Sanderson and Croft 1999). Category subsumption, an extension of the term subsumption method, using fuzzy partial ordering is proposed for the construction of hierarchy (Kim and Lee 2002). In this method, a category is represented by topical terms that are extracted using the Chi-square statistical measure. Frequent item set based hierarchical clustering (HC) algorithm (Fung et al. 2003) finds frequent item sets based on the term statistics and measures cohesiveness of a category directly from frequent item sets. Knijff et al. (2013) have given a comparison of HC algorithm and relationship based method to show that HC algorithm performs better with the optimized settings for deeper hierarchy and relationship based method performs better for shallower hierarchy.

In this chapter, we have proposed an approach which combines hierarchical clustering (to reveal query-context aware topical clusters using click-through log and topic sets obtained by

disambiguating web search results) and category relationship based approach (to find, update and refine topic hierarchy) to take the advantages of both.

7.2 Hierarchy Building Methodology

The proposed method for topic hierarchy building, in a nutshell, combines the agglomerative and relationship based approaches. We take a current hierarchy and topical clusters which are obtained using agglomerative clustering and click-through log. We have considered click-through log as a source of incremental data. We then establish relation(s) from the hierarchy categories to the clusters (the terms *category* and *cluster* are referred as nodes of the hierarchy and a topical cluster to be inserted into the hierarchy respectively). The cluster is then added at the appropriate place(s). This process is repeated for all clusters formed. Newly arrived data can be processed in the similar manner, i.e. newly constructed topical clusters are added or merged into existing hierarchy. The hierarchy and/or category may become less cohesive because of successive additions of new topics and/or the documents. Thereby, a refinement approach is proposed to update the existing topic hierarchy by reorganizing the information into more topically cohesive categories.

The proposed approach is presented in Figure 7.1. The **input** to the hierarchy building algorithm is a base hierarchy and topical clusters. Initially, the base hierarchy is taken as the top four levels of the ODP hierarchy which has very general categories (see Figure 7.2). Topical clusters are candidate categories that are to be inserted into the (base) hierarchy. These clusters are soft topical clusters formed through two-stage process: 1. Applying agglomerative clustering algorithm on click-through log to produce traditional query-context aware topical clusters and 2. Applying a technique to transform hard topical clusters into soft and homogeneous topical clusters (see Section 7.2.1).

In order to establish relations from existing categories to clusters, we add necessary information into them. This is a one-time process presented via *Information Extraction Module* in Figure 7.1 and is detailed in Section 7.2.2. *Hierarchy Building Module* (see Section 7.2.3) shows insertion and refinement modules of the proposed system.

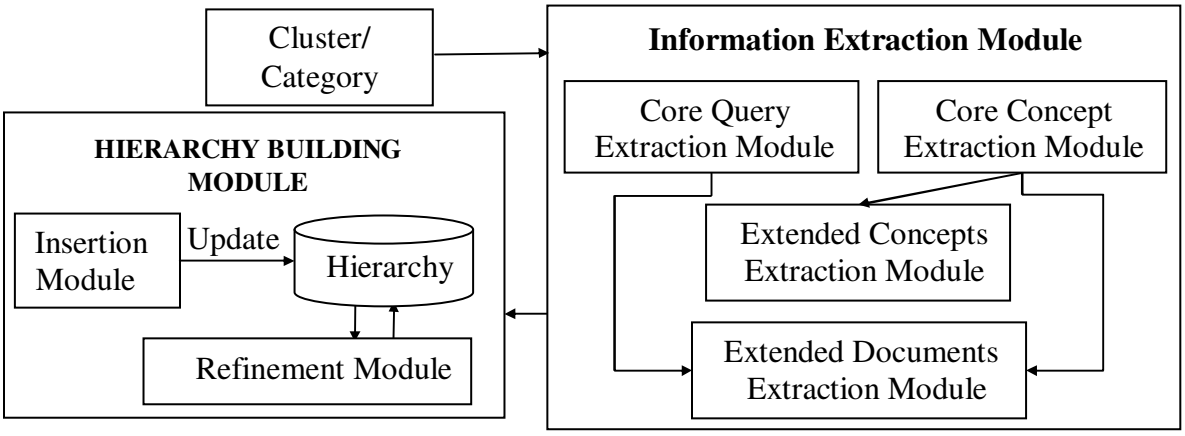


Figure 7.1. System model diagram

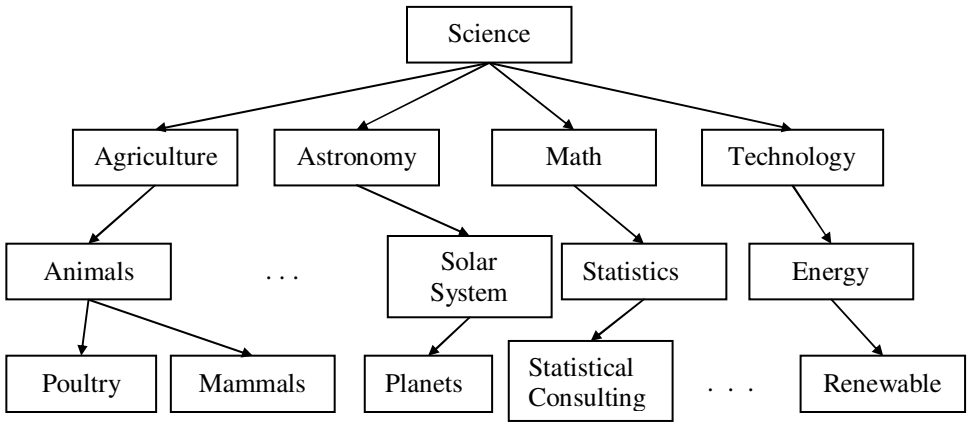


Figure 7.2. Partial Base Hierarchy

7.2.1 Input: Topical Clusters

Topical clusters, used as candidate categories to be inserted into the hierarchy, are query-context aware and soft. Query-context aware clusters are first constructed using Query-Document-Concept (QDC) hierarchical agglomerative algorithm (see Chapter 3). QDC belongs to a class of algorithms which works on query log such that, query based topical clusters are formed. A post processing technique, Split-Merge (SM) algorithm (see Chapter 5), is then applied to shape obtained clusters into coherent, and soft (natural) clusters such that one concept can belong to more than one cluster.

In QDC, first query-document-concept tripartite graph is constructed, where left side vertices are unique queries and right side vertices are concepts and vertices between them are clicked documents. Concepts are the filtered keywords, which are descriptors of the documents from the query's point of view, using a measure W1 (see Chapter 2). This graph structure precisely stores information about all documents clicked for a query, all concepts extracted from a document and documents involved in sharing the concepts for different queries and clearly indicates the number of documents clicked for a query, etc. An agglomerative clustering algorithm is then applied to obtain clusters of similar queries and the respective set of similar documents and the respective sets of concepts. These concept clusters are topical clusters and having concepts that relate query and document clusters with them. Similarly, topical clusters can be obtained from other algorithms of the same class. The document here refers snippet with the title rather than the whole document.

Topical clusters produced by the QDC are traditional clusters and having one concept only in one cluster. So, a post processing technique (refer Chapter 5) is applied, which is a two-phase SM algorithm, to produce refined and soft topical clusters. In SM algorithm, split phase splits the topical clusters by relating them to the topics that are obtained by disambiguating the web search results. Resultant clusters on similar topics are then merged using similarity among topical clusters.

It should be noted that the topical document clusters act as a basis for the building of automatic topic hierarchy. Topical document clusters should be constructed incrementally and should be added periodically into the current hierarchy. The incremental updates in the hierarchy will capture the new topics and changes. These updates are well captured by the query-context aware clustering algorithms.

7.2.2 Information Extraction Module

This module is a data preparation module that prepares data for categories of the base hierarchy and clusters to be inserted into the hierarchy so that relations between clusters and categories can be established. For this, we need the following:

1. Core Concepts

Core concepts are the descriptors of the documents in a category which are extracted from the snippets. The core concepts of the clusters are already computed as a part of the clustering. All the concepts of topical clusters are treated as core concepts as these are aggregated from the core concepts of each query in the associated query cluster. The core concepts for a category in the base hierarchy are extracted in the following way:

Nouns, verbs and noun phrases are extracted from the snippets associated with the ODP category by eliminating stop-words and then eliminating non-relevant terms by estimating category and concept relevance as all the concepts extracted from the documents might not be relevant to the given category. Concepts are filtered using weight function W1 (Goyal and Mehala 2011). The concepts with weight greater than threshold (δ) are considered as core concepts.

2. Core Query (CQ)

Core queries, a subset of queries associated with a category/cluster, describe the topical context of the categories unambiguously. Some queries that are ambiguous or short may belong to more than one category: e.g. for query "Apple", ODP categories are Science/Agriculture/Horticulture/Fruits/Apple, Science/Technology/Space/Satellite, and Computers/systems/Apple, thus, cannot be a core query.

Core Query extraction module selects queries that reflect the topical context of the associated topical category/cluster.

Core Query Extraction

CQ can be identified by estimating the relevance between the query and the respective documents. We believe that low-coherence among queries and documents imply more number of irrelevant documents in the category. In our work, we have considered top 15 queries with a high relevance score as core queries of a category/cluster. However, a threshold on relevance score can also be used for considering top queries. The relevance score for each query with respect to the category/cluster can be measured by finding the degree of distinctiveness between them. This is computed using relative entropy (Cover and Thomas 1991) between top-100 search result snippets. We call search results of a query Q as *QColl* and collective *QColl* of a category (for

each query) as *CatColl*. Greater the relative entropy, greater is the difference between the query and the category.

Query Relevance Score (QRS) between query *Q* and category *Cat* is calculated using relative entropy for core concepts only:

$$QRS(Q,\ Cat) = \sum_{c \in CS} P\left(c/Q\right) log \frac{P\left(c/Q\right)}{Pcoll(c)} \tag{7.1}$$

where *c* stands for a particular concept in *CS*, *CS* is the core concept set of *Cat*, extracted from its documents. $P\left(c/Q\right)$ is the probability of concept *c* occurring in the *QColl*. $Pcoll(c)$ is the probability of the concept *c* occurring in *CatColl*:

$$P_{coll}(c) = \frac{Total\ number\ of\ times\ c\ occurs\ in\ CatColl}{Total\ number\ of\ terms\ in\ CatColl} \tag{7.2}$$

Probability of concept c occurring in *QColl* can be estimated using two different ways, i.e. using probability and language model (Song and Croft 1999):

$$P\left(c/Q\right) = \frac{Total\ number\ of\ times\ c\ occurs\ in\ QColl}{Total\ number\ of\ terms\ in\ QColl} \tag{7.3}$$

and

$$P\left(c/Q\right) = \sum_{D \in QColl} P\left(c/D\right) P\left(D/Q\right) \tag{7.4}$$

respectively. where, *D* is a document in *QColl*. $P\left(D/Q\right)$ is the likelihood of a document *D* being relevant to the query *Q* and is estimated as follows (Song and Croft 1999):

$$P\left(\frac{Q}{D}\right) = \prod_{q \in Q} P\left(\frac{q}{D}\right) \tag{7.5}$$

where, *q* refers a term in the query *Q*. $P\left(D/Q\right)$ can be obtained by Bayesian inversion with uniform prior probabilities for documents in *QColl* and a zero prior for documents that contain no query terms. $P\left(\frac{q}{D}\right)$ and $P\left(\frac{c}{D}\right)$ can be estimated by the relative frequencies of terms linearly smoothed (Manning and Schütze 1999) with collection frequencies as

$$P\left(\frac{w}{D}\right) = \lambda P_{ml}\left(\frac{w}{D}\right) + (1-\lambda)P_{coll}(w) \tag{7.6}$$

where, $P_{ml}\left(\frac{w}{D}\right)$ is the relative frequency of term w in document D and λ is a smoothing parameter. $P_{ml}\left(\frac{w}{D}\right)$ is calculated as follows:

$$P_{ml}\left(\frac{w}{D}\right) = \frac{tf_{w,D}}{N_D} \tag{7.7}$$

where, $tf_{w,D}$ is the number of occurrences of term w in document D and N_D is the total number of terms in document D. *QRS* with Eq. (7.2) & (7.3) and *QRS* with Eq. (7.2) & (7.4) are referred as *P_QRS* (probabilistic *QRS* estimation) and *L_QRS* (Language model based *QRS* estimation) respectively.

We have conducted experiments for the categories of different types from different domains. Since, relevance score distribution is not known, a spearman rank correlation (Myers and Well 2003) test is applied. The resultant query score lists are replaced by the corresponding ranks and then correlation coefficient has been computed based on the resultant query rank list and expected query rank list. Perfect agreement between two rank lists results in 1 and total disagreement results in -1. We have estimated performance using d_based (where ties are known to be absent). We have also estimated the p-values that are probabilities which result in extreme or more extreme occur by chance (Myers and Well 2003). A comparison of *P_QRS* and *L_QRS* for 10 sample categories for spearman correlation coefficient and $\rho - values$ have been given in Table 7.1. *L_QRS* method works well with the lesser values of λ i.e. 0.1 to 0.5. This shows that better performance can be achieved with the increased weight of relative frequency of the term/concept in the category collection. The minimum and maximum correlation values achieved are -0.128 and 0.6362 respectively. This is also observed that this method is not proficient in predicting the core queries for the too specific categories (e.g. /Science/Agriculture/ Animals/ Insects/Bees). This is mainly because that the number of documents in these categories is less. P_*QRS* method works better than *L_QRS* in almost all the cases. The minimum and maximum correlation values achieved are 0.2101 and 0.9509 respectively. Low p-values in Table 7.1 indicate that there are less chances of correlation occurring by chance.

Table 7.1. Spearman correlation and ρ-value results

Agriculture Domain's Category Name (No. of Queries)	*Spearman Estimation methods*						ρ-value					
	L_QRS with λ value					*P_QRS*	*L_QRS with λ value*					*P_QRS*
	0.1	*0.3*	*0.5*	*0.7*	*0.9*		*0.1*	*0.3*	*0.5*	*0.7*	*0.9*	
Agriculture (358)	0.228	0.220	0.217	0.208	0.204	0.542	6.4E-06	1.3E-05	1.8E-05	3.7E-05	5.1E-05	4.8E-29
Animals (25)	0.612	0.616	0.611	0.609	0.593	0.759	0.00058	0.00052	0.00059	0.00061	0.00089	5.4E-06
Fruits (29)	0.090	0.071	0.071	0.055	0.053	0.385	0.32095	0.35636	0.35636	0.38908	0.39250	0.01973
Pests and diseases(146)	0.161	0.230	0.257	0.246	0.253	0.536	0.02600	0.00263	0.00088	0.00139	0.00102	1.6E-12
Soils(18)	0.636	0.530	0.476	0.434	0.421	0.951	0.00227	0.01185	0.02286	0.03599	0.04113	7.3E-10
poultry(42)	0.588	0.590	0.586	0.588	0.602	0.747	2.1E-05	1.9E-05	2.3E-05	2.1E-05	1.3E-05	6.8E-09
Bees (70)	0.428	0.423	0.429	0.425	0.425	0.588	0.00014	0.00016	0.00013	0.00015	0.00015	6.9E-08
Sustainable Agriculture (18)	0.372	0.394	0.394	0.394	0.428	0.623	0.06451	0.05274	0.05274	0.05274	0.03810	0.00286
Organic Farming(59)	0.561	0.560	0.564	0.566	0.564	0.509	1.9E-06	2.0E-06	1.7E-06	1.5E-06	1.6E-06	1.9E-05
Fisheries(145)	0.599	0.559	0.548	0.487	0.447	0.687	9.3E-16	1.4E-13	4.8E-13	2.6E-10	9.0E-09	6.9E-22

3. Extended Concept Set and Extended Document Set Extraction

Topic Set of a Query (*TSq*)

A query is likely to have multiple meanings because most of the queries posed are short and imprecise (Jansen et al. 1998) (ambiguous or belong to many topics). Moreover, new topics may arise in due course of time. For example, the word "green apple" is referring to "Tourism" and "apple" and is now being interpreted as "Ariane Passenger PayLoad Experiment" also. We construct a topic set for each query to identify different intentions of the query. For this, core concepts of top 100 search snippets of a query are generated (see Section 7.2.2.1) and are categorized (Goyal and Mehala 2011). To identify all topics including recent, it is required that the query disambiguation should take place at the same time when user poses a query. It is an unsupervised method, i.e. without using existing hierarchies like ODP, Yahoo, Google, etc. This is because not all the pages indexed by the search engine can be mapped with the categories of manually built hierarchies. The concepts which are not linked together (unrelated) are separated by partitioning the concept correlation matrix. These unrelated concepts will have low correlation values in the matrix. Concepts in a partition will form a category. Each category will represent a different topic that can be associated with the query. For example, for a query "apple", categories/topics formed are "Fruit", "Apple Products", "Tourism", etc. and the query "jaguar" related concepts can be categorized into "big cat", "jaguar car", "jaguar mining", etc. Each topic in the topic set is represented by the concepts referring an intention of the query.

Topic Overlap

For a category/cluster, consider its core concept set *C:* {*c1, c2, ..., ck*} and core query set *Q:* {*q1, q2, ..., qj*}. The topic *T* of a category/cluster is represented by a concept set *C*. Let TS_{qi}={$T1_i$, $T2_i$, ..., Tn_i} be the topic set for a query $qi \in Q$. Topic overlap between *T* and $Tj_i \in TS_{q_i}$ $O(T, Tj_i)$ can be defined as

$$O(T, Tj_i) = \frac{|T \cap Tj_i|}{|T \cup Tj_i|} \tag{7.8}$$

Extended concept set of a category T is computed by aggregating the concepts of topics for each query for which topic overlap is maximum: $= \cup_{q_i \in Q} Tj_i \; s.t. O(T,\ Tj_i)$ *is max.*

7.2.3 Hierarchy Building Module

In this section, we present a new algorithm to identify the relationship between a cluster to be inserted and a category in the hierarchy. Before presenting the algorithm, we will first describe the measures used for identifying relations:

1. Measures

A number of measures are used to establish a relation (merge, parent-child, ancestor-descendent, disjoint) between a category and a cluster. In the coming subsection, we use the symbols *C*, *Cat*, *N*, *c* and *m* denoting a cluster, a category, the total number of documents in *Cat*, a concept and number of documents that contain *c* respectively. The measures are as follows:

Category Entropy (CE)

Category entropy (Xu et al. 2008) measures the degree of relatedness of concepts of a cluster *C* to a category *Cat* (i.e. the amount of the information by which our knowledge about the category increases) and can be computed as an aggregation of individual concept *c* of *C* relatedness with the category:

$$\mathrm{CE} = -\sum_{\mathrm{c} \in \mathrm{C}} \mathrm{wt}\left(\frac{\mathrm{c}}{\mathrm{Cat}}\right) \; log \; wt\left(\frac{c}{Cat}\right) \tag{7.9}$$

where weight $wt(^{c}/_{Cat})$ of concept *c* from cluster *C* in Category *Cat* can be computed in two ways:

$CE1 = {}^{m}/_{N}$ which is fraction of documents of *Cat* that contain concept *c*.

$CE2 = cf/n_t$ which is the fraction of concept frequency in *Cat*. Here, *cf* is the collective concept frequency and n_t is the total number of terms in *Cat*.

Importance Measure (IM)

*IM (*Lang et al. 2008) has been used to quantify the importance of the concepts of a cluster *C* in a category *Cat* and can be computed in the following two ways:

First, *IM* is calculated based on inverse document frequency. This is because of concepts that are of greater importance and semantically significant tend to occur in a smaller fraction of a collection than other concepts. So, the importance of the concepts of the cluster *C* is aggregated *idf* i.e.

$$IM=\sum_{c\in C}\left(\frac{1}{m}\right) \tag{7.10}$$

Higher the *IM* value more is the importance of the concepts in a category.

Second, *IM* (say *V_IM*) is calculated based on variation in the frequency of concepts in *Cat* such that important concepts will have variation in the frequency or concepts of no importance will have a stable frequency or less variability in frequency in the documents. It is formulated as follows:

$$V_{IM} = \sum_{c\in C}\frac{V(c)}{MP(c)} \tag{7.11}$$

where, *MP*(*c*) is the mean of the probability of concept *c* in an individual document and *VP*(c) is the variance of the probability of concept *c*. These are defined as follows:

$$MP(c) = \frac{\sum_{i=1}^{N} P_i(c)}{N} \tag{7.12}$$

Each document in *Cat* is denoted as D_i. P_i(c) is the probability of concept *c* in D_i i.e. the fraction of terms in D_i.

$$VP(c) = \sqrt{\frac{\sum_{i=1}^{N}(P_i(c)-MP(c))^2}{N-1}} \tag{7.13}$$

Coverage Degree (CD)

Coverage degree measures the degree of coverage of concepts of cluster *C* with respect to a category *Cat.* This is used to identify the relation between a cluster *C* and a category *Cat* as follows:

$$CD = \sum_{c \in C} \frac{\sum_{i=1}^{N} P_i(c) * W(D_i)}{N} \tag{7.14}$$

where, D_i is a document in *Cat*, $W(D_i)$ is the weight of document D_i and $P_i(c)$ is the probability of concept *c* in D_i.

If a concept *c* has high occurrences in the documents, the *CD(c)* is considered to be high; by contrast, if *c* seldom appears in the documents, the *CD(c)* tends to be low. This clearly indicates that the higher the concept coverage, stronger the relation. This is to tell that cluster *C* can be a child of category *Cat* if concepts related to cluster *C* has less coverage with respect to *Cat* and vice versa.

Mutual Information (MI)

The similarity between two categories can be measured by comparing the probability of co-occurrence of the concepts with the probabilities of independent occurrence of the concepts in the documents of the categories. Mutual information measures the information that the two categories share. Two categories, Cat_i and Cat_j, are regarded as independent if knowing the concepts in Cat_i does not tell us anything about Cat_j. At the other extreme, if the concepts in Cat_i convey all the information of Cat_j then all information expressed by concepts in Cat_i can also be elucidated by concepts in Cat_j. That is mutual information is computed as the relative entropy between the joint distribution and the product distribution as (Cover and Thomas 1991; Ciminao et al. 2009):

$$MI(Cat_i, Cat_j) = \sum_{c_j \in Cat_j} \sum_{c_i \in Cat_i} p(c_i, c_j) \log_2 \left(\frac{p(c_i, c_j)}{p(c_i) * p(c_j)}\right) \tag{7.15}$$

where

$$p(c_i, c_j) = \frac{document\ frequency\ of\ joint\ conceptsc_i\ and\ c_j}{Total\ Number\ of\ documents} \quad (7.16)$$

and

$$p(c) = \frac{document\ frequency\ of\ concept\ c}{Total\ Number\ of\ documents} \quad (7.17)$$

2. Insertion Module

The idea of building the topic hierarchy is to insert topical clusters formed by the clustering algorithms at the appropriate locations of the base hierarchy. The base hierarchy (*M*) has borrowed top 4 levels of ODP hierarchy (see partial base hierarchy in Figure 7.2). The insertion module first prepares all the necessary data for each existing categories in *M* and for all the clusters to be inserted. Then, this module performs depth first traversal of the hierarchy to find the appropriate place(s) for each cluster to be inserted, i.e. the traversal does not stop when it finds an appropriate location for a cluster. But it continues to locate all the positions where the same cluster can be inserted. It is because of the fact that the same topic can be a subtopic of more than one topic.

There are four relations by which a cluster can be inserted into a hierarchy as a: parent-child, sibling, disjoint and exact match. These insertions are made from top to bottom in the hierarchy i.e. from general to specific category. If a match is found, then nodes in its subtree are examined. Otherwise, sibling is examined. In order to establish these relations, we have used the measures described above. We will now call a cluster also as a category for simplicity and write all the relations between two categories *X* and *Y* rather than between a cluster and a category.

Suppose category *X* is to be inserted into the hierarchy and it has to be compared with the existing category *Y*. *X* and *Y* have to go through a number of tests to form a relation between them. These tests are arranged in a manner so that a relation between *X* and *Y* can be established. For this, category entropy *CE*(*X, Y)* with both the variations are first checked against the respective thresholds (*Test1)* to find that the two categories are related. In case of failing the test, the two categories are declared as to be disjoint. If *X* and *Y* are found to be related by *CE*. A second test is then conducted to know the strength of the relation which is examined through importance measures *IM* and *V_IM*. We have also kept thresholds for *IM* and *V_IM*. If *X* and *Y* pass both the criterion, then the categories are said to be strongly related (as strong as at least related as siblings) otherwise disjoint. In the same way, a third test on coverage degree *CD* is

conducted for the parent-child relation. Coverage degree is calculated for both *CD*(*X, Y*) and *CD*(*Y, X)* and compared with a higher threshold for passing the test and then compared with each other to know whether *Y* is the parent of *X* or vice versa. The detailed test set for finding relation between categories *X* and *Y* is given in Figure 7.3.

We have observed that the documents in the categories are less for finding the relation between categories. So we have considered query based aggregated extended document and concept set (see Section 7.2.2.3) for identifying the relation.

Test1:*if test(CE1, CE2, CE1TH$_L$, CE1TH$_H$, CE2TH$_L$, CE2TH$_H$) == 1*
goto Test2
Test2:*if test(IM, V_IM, IMTH$_L$, IMTH$_H$, V_IMTH$_L$, V_IMTH$_H$) == 1*
goto Test3
Test3:*if(CD(X, Y)&& CD(Y, X))> CDTH$_H$*
if CD(X, Y)<=CD(Y, X) X is child of Y
else X is parent of Y
else X is a sibling of Y
function test(M1, M2, M1$_L$, M1$_H$, M2$_L$, M2$_H$) do
if ((M1(X, Y) && M1(Y, X))>***M1$_L$***&& (M2(X, Y) && M2(Y, X))>***M2$_L$***)
if ((M1(X, Y) && M1(Y, X))>***M1$_H$*** || (M2(X, Y) && M2(Y, X))>***M2$_H$***)
Categories X and Y are related ; return 1
else Categories X and Y are disjoint; Break
end

Figure 7.3. Test Set

We now explain why these tests are required to be in this form of execution. Category entropy is used to measure the document based overlap and frequency based overlap between two categories. Test1 ensures that minimum information shared between the categories makes them as a candidate of related categories. Once this test is passed, we examine that how important are the concepts of a cluster to the category through importance measure. This will indicate the concepts involved are either general or having multiple meanings or specific to the category using idf and variance. Thus, this cluster will appear in the hierarchy atleast as a sibling of the category. If the cluster concepts are then to be found to cover the documents of the category at a greater extend, then the parent-child relation can be formed. In this work, the weight of all the documents in coverage degree has been considered as one. However, more sophisticated method can be applied to give more weight to the core documents.

In our work, we have separated sibling relation into two: related sibling and disjoint sibling. For example, the categories "Science/Agriculture" and "Science/Math" are siblings but they both refer two different domains, thus referred as disjoint siblings. The categories "Algebra helper

software" and "Web based calculators" under the category "Science/Math/TeachingResources" are from the same domain so referred as related siblings.

We have observed that many related clusters are identified as disjoint siblings due to insufficient information in clusters or categories. But, these problems are rectified later in refinement stage when the sufficient information is added.

3. Refinement Module

Refinement module updates the topic hierarchy by reorganizing the categories to make it better reflect the topics in the hierarchy. This process is introduced because the hierarchy and/or category may become less cohesive due to successive additions of new topics and/or the documents. Reorganization is also required due to change in semantics over time. For example, two categories may not be related at an early stage of hierarchy creation, but later may relate to each other after receiving more data. Another reason for not identifying relations properly in the beginning is because of insufficient information about categories to be inserted. Another important reason for the same is that we have put tight thresholds for a number of measures to ensure the relation. However, one could relax these thresholds to get broader categories and smaller hierarchies.

Refinement consists of two phases: *related categories identification* (Ref1) followed by *equivalent categories identification* (Ref2).

In *Ref1*, we group related children or subcategories of a category at the same level in a bottom-up fashion. We create a new category for grouping these related categories. The new category becomes the parent of the group/subcategories and is inserted as a child of the parent category of the group. For example, the category *Science/Math/TeachingResources* (*Cat*) has the sub-categories *AlgebraHelperSoftware* (*Cat1*), *WebBased Calculators* (*Cat2*) and *MathWorksheets* (*Cat3*). C*at1* and *Cat2* should be identified as related siblings. So, new category referred as *Software* should be placed under *Cat* and both *Cat1* and *Cat2* should be placed as children of *Software* category. Since, refinement phase is to be run after many updates in the hierarchy. We perform our tests again on the categories to identify the relations. Similar to insertion of a category, the two subcategories X and Y undergo the tests 1 and 2 for related siblings. On clearing two tests, we check for the third test for both higher and lower thresholds. Here, we introduce a new threshold $CDTH_L$ for finding X and Y as related siblings.

In *Ref2*, we identify and merge equivalent categories which are separately placed in a parent-child relation or in a sibling relation in a bottom-up fashion. This is important because it not only reduces redundant categories in the hierarchy but also aims at improving the knowledge of each category of hierarchy by merging redundant/ equivalent categories. For this, we use 'Mutual Information' mentioned above. Mutual information measures how much information concepts of one category contain about the other. If MI is large enough than two categories are merged. Figure 7.4 depicts the equivalent categories identification process with the sample categories "Sugarcane crops usage" and "Sugarcane Usage". The dashed middle circle in Figure 7.4 signifies the mutual information shared by the two categories. If the overlapping area of circles with the dashed middle circle is greater than an empirical threshold, we merge the two clusters.

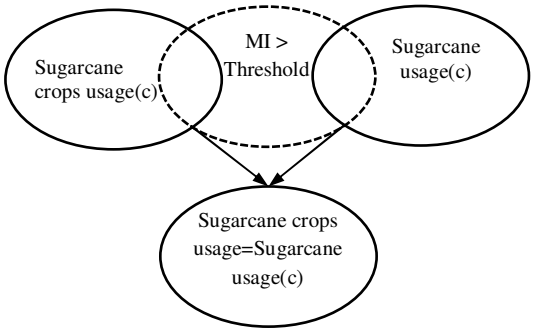


Figure 7.4. Ref2 algorithm illustration using sample example

Both *Ref1* and *Ref2* are bottom-up approaches because changes made at lower level may propagate from the current node to the root node. *Ref2* is applied after *Ref1* to first identify related categories and then to merge categories from the related ones.

7.3 Experimental Setup and Data Sets

The input required to build the topic hierarchy are queries, their clicked documents' URLs with titles & snippets and their ODP categories (assigned by DMOZ). DMOZ is the human-edited directory, which is the contribution of volunteers from all over the world, and is also recognized as Open Directory Project (ODP). There is no standard data set, which has all the required information, available to the best of our knowledge. So, we have used two datasets: 1. ODPDS: Data set obtained by preprocessing three data sets AOL500k data set (AOL 2006), CABS120k08

data set (CABS 2015) and DMOZ data set (ODP 2015) and 2. IASRIDS: Data set obtained from the Indian Agricultural Statistics Research Institute (IASRI, Delhi, India).

ODPDS

The AOL500k (2006) provides queries and click-through log with domain of URLs, but does not contain snippets of the documents. It consists of ~20M web queries collected from ~650k users over three months (01 March, 2006 - 31 May, 2006). The data set includes AnonID, Query, QueryTime, ItemRank and ClickURL. Here, ***AnonID*** represents an anonymous user ID, ***Query*** represents the query issued by the user, ***QueryTime*** represents the time at which the query was submitted to search, ***ItemRank*** represents the search result rank of the clicked item and ***ClickURL*** represents the domain portion of the clicked URL. The details of AOL 500k and CABS120k08 data set are given in Table 7.2.

Table 7.2. AOL 500k (2006) and CABS120k08 Data Sets

AOL 500k (2006)		**CABS120k08**	
Number of		Number of	
Queries	21,011,340	URLs	117,434
Click-through events	19,442,629	Categories	84,663
Queries w/o user click-through	16,946,938	Searches	2,617,326
Unique user ID's	657,426	Unique queries	13,33,900
Unique normalized queries	10,154,742	Users	3,383,571

The CABS120k08 data set contains 117,434 URLs with additional metadata including URL category in ODP which they have obtained using an intersection of AOL500k (2006) and the ODP data set. The details are shown in Table 7.2. We extract only queries, document's URLs and their respective ODP categories from CABS120k08 Data Set.

DMOZ data set incorporates document's URLs with snippets and assigned categories of the documents' etc. The number of documents in the ODP categories is mostly less. The resultant data in ODPDS after intersection of three data sets is adequate for "Science" category. Therefore, we have considered "Science" category alone for experiments. This has a number of non-leaf categories, number of leaf categories, the total number of documents, mean number of children per node and depth in the Science category as 2205, 9423, 96127, 5.27 and 2-11 respectively

before the intersection. The ODPDS data set is divided into four data set ODPDS1 (Science/ Agriculture), ODPDS2 (Science/ Math), ODPDS3 (Science/ Technology), and ODPDS4 (Science/ Astronomy).

We have downloaded these data sets in September 2011, so it is possible that the data sets might have been modified during this time. The number of ODP categories in the base hierarchy in ODPDS1, ODPDS2, ODPDS3 and ODPDS4 are 39, 64, 40 and 138 respectively.

We have evaluated the topic hierarchy R obtained by the proposed approach against the target hierarchy T. T is the manually prepared target hierarchy which is an extended version of ODP hierarchy. We added new subcategories in ODP because it is not wide and deep enough to cater more specific topics. For example, the categories "Rice crops" and "Water Sprinkler system" are not there in the existing ODP hierarchy. These should be under /Science/Agriculture/FieldCrops and /Science/Agriculture respectively.

The whole experiment is conducted in four parts: preprocessing, training, evaluation and refinement. In the preprocessing part, the data is extracted from the data sets and QDC clustering algorithm along with SM technique is applied to get query-context aware soft and pure concept/topical clusters. The information extraction module (see Section 7.2.2) is then run over the clusters and categories of the base/current hierarchy to get the required data.

The statistics of the input and output of clustering on ODPDS data set is shown in Table 7.3. The number of unique documents is less because same document is clicked for more than one query.

In the training part, two data sets ODPDS1 and ODPDS2 are considered for learning the thresholds. The hierarchies are built for various threshold values. The effect of low and high thresholds on the resultant topic hierarchy (R) is investigated using taxonomical measures empirically (see Next Section). We have learnt thresholds separately for all tests. We first started low value of thresholds and then increased it iteratively to achieve optimum. The threshold values used in the experiments are given in Table 7.4.

Table 7.3. Input & Output statistics of clustering for ODPDS and IASRIDS

Number of	**ODPDS**	**ODPDS1**	**ODPDS2**	**ODPDS3**	**ODPDS4**	**IASRIDS**
Input for clustering						
Unique queries	2306	548	300	1236	255	112
Unique docs	416	54	42	259	66	518
Unique concepts	2050	359	214	1256	263	1211
Output of clustering						
Clusters	359	39	33	242	49	93
Avg. queries per Cluster	6.4	14.1	9.1	5.1	5.2	4.494
Avg. concepts per Cluster	5.71	9.21	6.48	5.19	5.37	13.08

Table 7.4. Thresholds for various measures

	CE1	**CE2**	**IM**	**V_IM**	**CD**	**MI**
Low Threshold	12	0.9	116	6000	23	45
High Threshold	85	1.4	300	10000	65	

All the measures aggregate over the concepts of a category to get aggregated scores. The score grows with the increase in the number of concepts. But, the effect of the measure would not be x times if the score increases by x times. Thus, we scale down the measure values to dampen the effect. If ($m < \min(|cat1|,|cat2|) \leq n$) all thresholds are multiplied by 0.2, 0.4 and 1 for (m, n)=(0, 500), (500, 1000) and (1000, ∞) respectively.

IASRIDS

For the further verification and validation of our algorithm, we have obtained data from the *Indian Agricultural Statistics Research Institute* (IASRI, Delhi, India). The statistics of input and output of clustering IASRIDS are given in Table 7.3. For our experiments we have considered ODP Category */Science/Agriculture/Field Crops* (upto first 4 levels) as base hierarchy. Manual Target Hierarchy "T" was prepared by *IASRI* research scientists.

We tried mapping the categories in the target hierarchy given by IASRI with the categories in ODP. Table 7.5 shows the mapping till level 8 of the hierarchy. The categories given in italic are not present in the ODP hierarchy. It is clear from the table that categories are missing even in the base hierarchy. It is observed that 35 categories (out of 93 categories) related information are not

available in ODP. For example, consider the clusters with the intention {*green tea processing, agronomic practices on soil content and Drip System irrigation*}. Categories related to these clusters are Cash crops, Soil and Water management in Agriculture respectively. But, these categories and their related information are missing in ODP hierarchy.

Table 7.5. Mapping of categories in manual target hierarchy of IASRIDS with ODP categories

Level No.	**Categories in IASRI data set (No. of categories covered by ODP)**	**No. of Queries**
L2	Agriculture (1)	6
L3	Field Crops (1)	6
L4	Cereals, Oilseeds, *Cash Crops* (2)	22
L5	Cereals.Rice, Cereals.Wheat, *Oilseeed.mustard, Oilseed.seasame*, Oilseed.rapeseed, *Cash.tea*, *Cash.coffee*, Cash.sugarcane (4)	39
L6	*Rice.AgronomicPractices*, *Wheat.Agronomic Practices* (0)	6
L7	*Agronomic.Soil, Agronomic.CM* (0)	17
L8	*Soil.Water Management, Soil.Fertilizer* (0)	18

7.3.1 Hierarchy Evaluation Measures

There are many measures for reference based hierarchy evaluation. Mainly measures are divided into two groups: lexical and taxonomical (Dellschaft and Staab 2006). Lexical measures consider hierarchy as a collection of categories and evaluate the obtained hierarchy with the ground truth without considering the hierarchical relations among the categories. Given a target topic hierarchy or ground truth (T) and resultant topic hierarchy (R), lexical precision LP, recall LR and F-measure LF are described as follows:

$$LP(T,R) = \frac{|T \cap R|}{|R|} \tag{7.18}$$

$$LR(T,R) = \frac{|T \cap R|}{|T|} \tag{7.19}$$

$$LF = 2 * \frac{(LP * LR)}{(LP + LR)} \tag{7.20}$$

Taxonomical measures are more complicated than the lexical measures as they measure taxonomic overlap of the categories using local and global measures. The local measure compares local taxanomic overlap between the two categories by considering all their super and sub categories whereas global taxanomic measure is then computed by taking average of local measure results for category pair from learned and the reference hierarchy. In other words, local measure compares the position of two categories and the global measure is used to compare two hierarchies. Two type of taxonomic measures are proposed in the literature: one using semantic cotopy (SC) i.e. all its super- and sub-categories and another using common semantic cotopy (CSC) i.e. removing categories which are not present in other hierarchy. Taxonomic precision TP, taxonomic recall TR using semantic cotopy are described as follows:

$$TP_{sc}(T,R) = \frac{1}{|T|}\sum_{c\in T}\begin{cases} tp_{sc}(c,c,T,R) if c \in R \\ max_{c'\in R} tp_{sc}(c,c',T,R) if c \notin R \end{cases} \quad (7.21)$$

$$TR_{sc}(T,R) = TP_{sc}(R,T) \quad (7.22)$$

where, $tp_{sc}(c1,c2,T,R)$ and $tr_{sc}(c1,c2,T,R)$ are the local taxonomic precision and recall of a category and defined as:

$$tp_{sc}(c1,c2,T,R) = \frac{|ce(c1,T) \cap ce(c2,R)|}{|ce(c1,T)|} \quad (7.23)$$

$$tr_{sc}(c1,c2,T,R) = \frac{|ce(c1,T) \cap ce(c2,R)|}{|ce(c2,R)|} \quad (7.24)$$

Local taxonomic precision and recall find the similarity of two categories based on the characteristics extracts ***ce*** which categorizes the position of a category in the hierarchy i.e. two extracts should contain many common categories if the considered categories are at similar positions in the hierarchies (Dellschaft and Staab 2006).

Common semantic cotopy based TP and TR are defined as:

$$TP_{csc}(T,R) = \frac{1}{|T\cap R|}\sum_{c\in T\cap R} tp_{csc}(c,c,T,R) \quad (7.25)$$

and

$$TR_{csc}(T,R) = TP_{csc}(R,T) \tag{7.26}$$

Taxonomic F-measure (TF) is again the harmonic mean of TP and TR.

$$TF = 2 * \frac{TP*TR}{TP+TR} \tag{7.27}$$

The harmonic mean of lexical recall and TF is used to produce second order TF^1 value in place of TF for the cases where TF is not influenced heavily by the lexical level performance.

$$TF' = 2 * \frac{TF*LR}{TF+LR} \tag{7.28}$$

In addition, Taxonomic overlap TO(T,R) is evaluated like TP where local taxonomic overlap, TO(c1, c2, T, R), between two hierarchies T and R for categories c1 and c2 is defined as follows:

$$TO_{ce}(c1,c2,T,R) = \frac{|ce(c1,T) \cap ce(c2,R)|}{|ce(c1,T) \cup ce(c2,R)|} \tag{7.29}$$

7.4 Experimental Results

We report the topic hierarchy evaluation results for ODPDS1, ODPDS2, ODPDS3, ODPDS4 and ODPDS. It is required considering the hierarchy evaluation for the combined data set ODPDS because a new category constructed (e.g. NASA) may relate to two or more categories: For example, agriculture from ODPDS1 and technology from ODPDS4. Thus, it may be inserted at more than one place. We have also evaluated results of IASRIDS. We use both types of evaluation metrics lexical and taxonomical (see Section 7.3.2) for evaluation of the obtained hierarchies against their ground truths.

7.4.1 Experimental Results: ODPDS

We run insertion phase to get the hierarchy on all five ODP data sets. We also run refinement phase on the resultant hierarchies. We evaluate both stages of the hierarchies against target hierarchies. Lexical measures LP, LR and LF are presented in Table 7.6 and taxonomic measures TP, TR and TF with semantic cotopy and common semantic cotopy are presented in Table 7.7.

Table 7.6 shows that lexical precision, lexical recall and F-measures achieved and all values are above 79.7%, 79.4% and 80.7% after insertions respectively. These values (minimum) are corrected atleast upto 85.8%, 89% and 89.6% respectively after applying the refinement process. The F-measure achieved after refinement is 92.1%, 89.9%, 95.6%, 89.9%, 89.6% for data sets ODPDS, ODPDS1, ODPDS2, ODPDS3, ODPDS4 respectively. The lexical measure values achieved are higher because they do not cater to the positions of the categories in the hierarchies.
Semantic cotopy based TP, TR and TF after insertions are 76.8%, 67.4% and 71.8% respectively for ODPDS (see Table 7.7). These values are quite high for non-training data sets ODPDS3 and ODPDS4 (ODPDS1 and ODPDS2 are used for training). The measures are corrected after applying refinement phase in many cases.
We also report common semantic cotopy based TP, TR and TF for all the data sets in Table 7.7 and it is clear that the values are lower as compared to semantic cotopy. The maximum F-measure achieved before and after refinement are 77.8% (ODPDS4) and 81.5% (ODPDS4) respectively. Common semantic cotopy based measures are lower because an error in the insertion process propagates on all its descendants i.e. if a category is inserted incorrectly, all its descendants positions will be incorrect. It is observed that errors are mainly due to insufficient information in the categories of the hierarchy. The higher values of measures are expected by lowering the threshold values that will result in broader categories and smaller hierarchies. This will affect more in taxonomic measures because they use sub and super categories along with the candidate categories to be evaluated.

Table 7.6. Lexical measures on various data sets

Data Set	ODPDS			ODPDS1			ODPDS2			ODPDS3			ODPDS4		
	Ins	Ref1	Ref2	Ins	Ref1	Ref2	Ins	Ref1	Ref2	Ins	Ref1	Ref2	Ins	Ref1	Ref2
LP	0.797	0.757	0.858	0.864	0.853	0.875	0.885	0.861	0.933	0.905	0.907	0.908	0.922	0.922	0.903
LR	0.818	0.995	0.995	0.962	0.981	0.925	0.920	0.990	0.980	0.885	0.998	0.891	0.794	0.967	0.890
LF	0.807	0.860	0.921	0.911	0.912	0.899	0.902	0.921	0.956	0.895	0.950	0.899	0.854	0.944	0.896

Table 7.7. Taxonomic semantic cotopy and common semantic cotopy measures on various data sets

Data Set	*Semantic Cotopy Based Taxonomic Measures*														
	ODPDS			*ODPDS1*			*ODPDS2*			*ODPDS3*			*ODPDS4*		
	Ins	*Ref1*	*Ref2*	*Ins*	*Ref1*	*Ref2*	*Ins*	*Ref1*	*Ref2*	*Ins*	*Ref1*	*Ref2*	*Ins*	*Ref1*	*Ref2*
TP	0.768	0.693	0.739	0.946	0.943	0.955	0.944	0.924	0.958	0.963	0.952	0.963	0.988	0.942	0.949
TR	0.674	0.956	0.965	0.739	0.883	0.863	0.916	0.984	0.958	0.876	0.977	0.959	0.837	0.967	0.946
TF	0.718	0.804	0.837	0.830	0.912	0.907	0.929	0.953	0.968	0.917	0.964	0.961	0.906	0.954	0.948
TO	0.454	0.534	0.640	0.635	0.746	0.751	0.795	0.807	0.897	0.781	0.861	0.887	0.809	0.844	0.826
F'	0.766	0.889	0.908	0.891	0.945	0.916	0.920	0.976	0.974	0.901	0.981	0.958	0.847	0.960	0.918
	Common Semantic Cotopy Based Taxonomic Measures														
TP	0.527	0.635	0.728	0.745	0.758	0.755	0.722	0.722	0.784	0.775	0.785	0.816	0.782	0.799	0.816
TR	0.540	0.636	0.728	0.675	0.692	0.687	0.718	0.727	0.790	0.764	0.778	0.810	0.775	0.800	0.814
TF	0.533	0.636	0.728	0.708	0.723	0.719	0.720	0.725	0.787	0.770	0.782	0.813	0.778	0.800	0.815
TO	0.409	0.549	0.628	0.608	0.633	0.611	0.606	0.612	0.665	0.668	0.686	0.713	0.652	0.683	0.700
F'	0.646	0.776	0.841	0.816	0.833	0.809	0.804	0.840	0.873	0.823	0.877	0.878	0.786	0.875	0.851

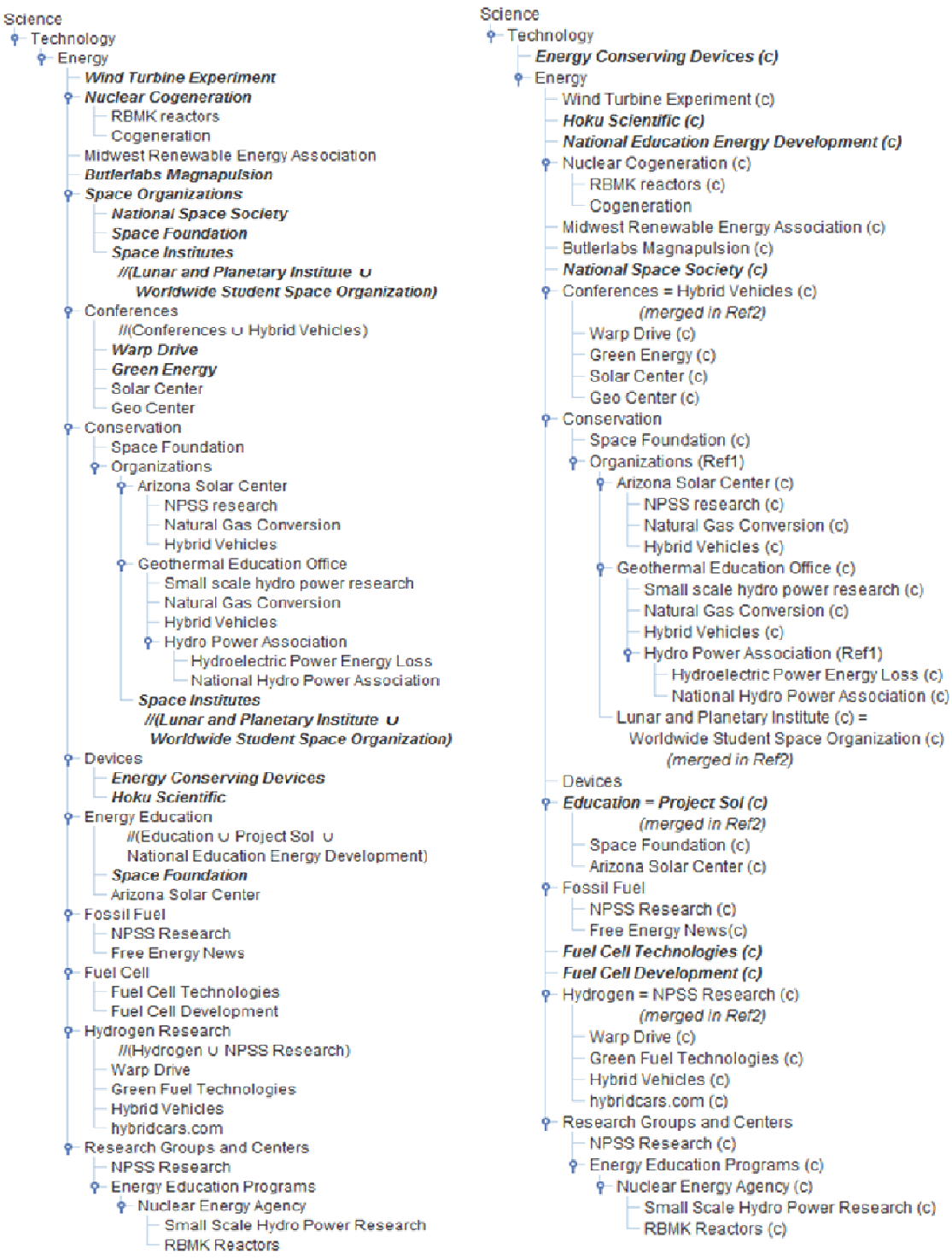
We now explain the reason why there is a decrease in taxonomic precision of semantic cotopy in few cases in the refinement phase with the following example: Consider the topic "technology/energy/publication" in dataset ODPDS4 and related topics such as *NPSS Research, Geothermal education office, rbmk reactors*. In insertion phase, these topics are correctly inserted as child of the category *publication*. But, in refinement phase, the topics *Geothermal education office and rbmk reactors* are actually two different streams of *technology* related to *geothermal* and *nuclear* studies respectively. Refinement phase is predicting a relation between them due to two reasons: occurrences of broad common concepts such as *generation, renewable energy, energy efficiency, electricity, environment, development, power, heating and cooling* and insufficient information in the categories. Thus, semantic cotopy results in higher taxonomic precision for insertion and lower taxonomic precision for refinement. As common semantic cotopy measures consider only common topics of the hierarchies for evaluation, it shows higher values for both insertions and refinement.

The above problem can be solved, if refinement phase is applied to a node after sufficient growth of its subtree with sufficient information.

Due to the same problem, the dataset ODPDS also shows a decrease in taxonomic precision in refinement phase in a few cases for semantic cotopy. For example, the topic "/science/math/statistics" related to *npss research (Nuclear and Plasma Sciences Society)* becomes a sibling of organizations in insertion phase correctly. Refinement phase creates a

common parent node among the topics chats and forums, *npss research* incorrectly due to its general concepts such as *news, forums, blogs, etc*. This error may affect successive insertions and refinements. However, one can avoid it by deferring the refinement phase till sufficient data is collected. It is advisable that refinement phase should be applied after sufficient growth in the hierarchy.

Figure 7.5a is the sample target hierarchy T which is an extension of ODP hierarchy as mentioned earlier. The categories which are added to extend ODP are highlighted in bold and italic fonts in the figure. Figure 7.5b presents the resultant hierarchy R. The clusters which are inserted in the base hierarchy and their labels are appended with *(c).* The categories in the resultant hierarchy which are inserted in the wrong places as compared to the target hierarchy are highlighted in bold and italic fonts in the figure. If a category is introduced in *Ref1* process, the category is labeled by appending *(Ref1)* with the label. If some categories are getting merged in Ref2 process, these are marked as "=" sign and also augmented with the process name. The remaining categories in R are from the base hierarchy. The category *"Energy conserving devices"* in R is inserted in the wrong place as compared to T where it is present under the category *"Energy/Devices"*. The data corresponding to this category have only three documents and one query. The position of the category may change to the right place if more data related to a category would be added in later insertion or refinement process. Consider two categories *"Fuel Cell"* and *"Hydro Power Association"* in T. Both categories have children {*"Fuel Cell Technologies"*, *"Fuel Cell Development"}* and *{Hydroelectric Power Energy Loss, National Hydro Power Association}* respectively. These four categories are inserted in the insertion phase. The category "*Hydro Power Association"* has been introduced as a parent of "*Hydroelectric Power Energy Loss"* and *"National Hydro Power Association"* in Ref1 process similar to T. However, the same refinement process couldn't construct parent category *"Fuel Cell"* for the categories *"Fuel Cell Technologies"* and *"Fuel Cell Development"*. This is also because of insufficient information in the categories. Similarly, the Ref2 process lacks in finding the exact match relation among the categories *"National Education Energy Development"*, *"Education"* and *"Project Sol"* due to the lack of information in the category *"National Education Energy Development"*.



(a) (b)

Figure 7.5. a. Partial Target and b. Partial Resultant hierarchy for ODPDS3 (“Science/Technology”)

7.4.2 Experimental Results: IASRIDS

Table 7.8 and Table 7.9 present the lexical and taxonomical measures obtained on IASRIDS respectively. LP, LR and LF achieved are 71.8%, 83.1% and 77.1% after insertions. LP, LR and LF values are 57.9%, 82% and 67.9% respectively after the refinement process. Semantic cotopy based TP, TR and TF after insertions are 52%, 59% and 55.3% respectively for IASRIDS (see Table 7.9). Semantic cotopy based TP and TR achieved after refinement is 47% and 60.8% respectively. Common semantic cotopy based measures are less as compared to semantic cotopy based measures. Further, both lexical and taxonomic measures are less on IASRIDS than that of ODPDS. This is due to the fact that ODP base hierarchy itself does not have the required information. Also considering the fact that we are aggregating the information for each category using all of their children in the current hierarchy. Thus, insertion of a few odd children may propagate the unrelated knowledge of category to the parent which in turn allows other unrelated categories to cross the thresholds and enter in the hierarchy. First level categories are too broad categories, so most of the unrelated topics may also be identified as relevant topics due to general terms/concepts. *Coverage Degree* (CD) plays an important role in finding the relation between the topics. Thus increasing CD thresholds in the upper level(s) in the current hierarchy resolves the problem of inserting unrelated topics. Therefore, we have increased CD thresholds as CD_L (28) and CD_H (145) for first level of current hierarchy and evaluated the performance again.

Table 7.10 and 7.11 present the lexical and taxonomical measures respectively obtained on IASRIDS (for variable threshold different for first and other levels). Table 7.10 and 7.11 show much better results than that of Table 7.8 and 7.9.

Lexical measures are high in the case of IASRI data set, as queries cover almost all the categories in the target hierarchy in comparison to other ODP data sets. On the contrary, taxonomic measures for IASRIDS are lower than that of ODPDS. This is because no data related to 37.634% of categories are present in the base hierarchy. Thus, the clusters are not placed in the proper position in the hierarchy. This problem can be resolved by deferring the refinement process when more related data is collected and updated in the hierarchy. Consider an example; the broad category *"field crops"* in the hierarchy does not have any information related to subcategory *Mustard*. Thus, we are unable to insert any subcategories like {*Mustard Types,*

Mustard Growth, Mustard Seed Shopping, etc} related to *Mustard* as a child of *field crops*. Therefore, these are inserted as siblings of *field crops*.

We will now compare results for different stages of the proposed algorithm. It can be observed from Table 7.11 that taxonomic precision decreases after insertion but increases in refinement phase. Increase in lexical and taxonomical precision in refinement phase is attributed to the fact that most of the new categories which contain less information have merged to a common broad category.

Precision and recall both need to be high in hierarchy building. Hence, the taxonomic F-measure represented as the harmonic mean of global taxonomic precision and recall gives a better representation of goodness. However, for cases where lexical measures do not impact taxonomic measures, we need to evaluate Taxonomic F'-measure, which is represented as the harmonic mean of lexical recall and taxonomic F-measure. It can be seen from table 7.11 that Taxonomic F' measure is greater than Taxonomic F-measure for both SC and CSC for both ODPDS and IASRIDS because of the high lexical recall. Taxonomic F' measure for Ref2 is lesser in a few cases from Ref1 in the SC of ODPDS2, ODPDS3, ODPDS4. It is due to a few unrelated small categories merging together. This is mainly because of insufficient information in the categories. Thus, it results in a decrease in lexical recall of the hierarchy. This problem doesn't have impact in CSC as only common categories in T and R are considered. Furthermore, SC achieves better TF' than CSC, as TP is observed to be higher than that of CSC (reasons explained above). Similar behavior can also be observed in the IASRIDS which points to the fact that we are observing a better recall even in situations where we are trying to insert a large number of unseen categories in the base hierarchy.

Table 7.8. Lexical Evaluation Results on IASRI Data Set without first level threshold

Data Set	*IASRIDS*		
Module	*Ins*	*Ref1*	*Ref2*
LP	0.718	0.576	0.579
LR	0.831	0.854	0.820
LF	0.771	0.688	0.679

Table 7.9. Taxonomic Evaluation Results on IASRI Data Set without first level threshold

Data Set	*Semantic CotopyBased Taxonomic Measures*			*Common Semantic CotopyBased Taxonomic Measures*		
	IASRIDS without first level threshold			*IASRIDS without first level threshold*		
	Ins	*Ref1*	*Ref2*	*Ins*	*Ref1*	*Ref2*
TP	0.520	0.463	0.470	0.400	0.394	0.402
TR	0.590	0.604	0.608	0.666	0.534	0.539
TF	0.553	0.524	0.530	0.500	0.453	0.460
TO	0.237	0.524	0.202	0.327	0.285	0.290
TF'	0.664	0.649	0.644	0.624	0.592	0.590

Table 7.10. Lexical Evaluation Results on IASRI Data Set with first level threshold

Data Set	*IASRIDS*		
Module	*Ins*	*Ref2*	*Ref3*
LP	0.735	0.754	0.933
LR	0.843	1.000	0.944
LF	0.785	0.860	0.939

Table 7.11. Taxonomic Evaluation Results on IASRI Data Set with first level threshold

Data Set	*Semantic CotopyBased Taxonomic Measures*			*Common Semantic CotopyBased Taxonomic Measures*		
	IASRIDS without first level threshold			*IASRIDS without first level threshold*		
	Ins	*Ref1*	*Ref2*	*Ins*	*Ref1*	*Ref2*
TP	0.843	0.854	0.854	0.723	0.756	0.767
TR	0.603	0.816	0.816	0.700	0.738	0.744
TF	0.703	0.835	0.835	0.711	0.747	0.755
TO	0.429	0.711	0.711	0.558	0.620	0.639
TF'	0.767	0.886	0.886	0.771	0.855	0.839

Figure 7.6a. shows the partial base hierarchy for *"Science/Agriculture"*. Figure 7.6b. shows the portion of a target hierarchy T prepared by IASRI experts in the area of field crops. The categories which are added to extend ODP are highlighted in bold and italic fonts in the figure. Figure 7.6c. depicts the resultant topic hierarchy after insertion of a few topical clusters. In the sample example, we have shown insertions of *sugarcane* and *agriculture research* related few clusters only and are denoted by label appended with "(c)". In the sample example, we have shown intermediate results such as after insertion (Figure 7.6c), after Ref1 at leaf level (Figure 7.6d), after Ref1 (Figure 7.6e) and after Ref2 (Figure 7.6f). It can be observed from the figure 7.6d. that related sibling categories in the leaf level like *Sugarcane*, *Sugarcane Crops Usage*, *Sugarcane Usage* and *Organic Sugarcane Use* are identified and grouped under newly inserted topic *Sugarcane*. The category/topic *Sugarcane* (which is inserted in Figure 7.6d.) effect is propagated and new parent node *Energy crops* is inserted and *Fiber Crops* & *Sugarcane* became its children. Ref2 is applied on the resultant hierarchy obtained from the Ref1 as shown in figure 7.6f. The equal sign (=) in the figure denotes the formation of cluster by merging two or more clusters in the resultant hierarchy. All labels are appended by *(Ref1)*, *(Ref2)* and *(c)* according to the updates happened due to *Ref1*, *Ref2* and *Insertion* process respectively. All other remaining categories are from the base hierarchy. The categories in the intermediate resultant and resultant hierarchy which are inserted in the wrong places as compared to the target hierarchy are highlighted in bold and italic font in the figures. It is clear from the resultant hierarchy (Figure 7.6f) that all the categories are finally placed at the appropriate locations except the one *"Organic Sugarcane Use"* which is not merged with the categories *"Sugarcane Crops Usage"* and *"Sugarcane Usage"*. The same behavior has been observed in most of the samples. Final taxonomic and lexical measures are given in the table 7.10 and table 7.11.

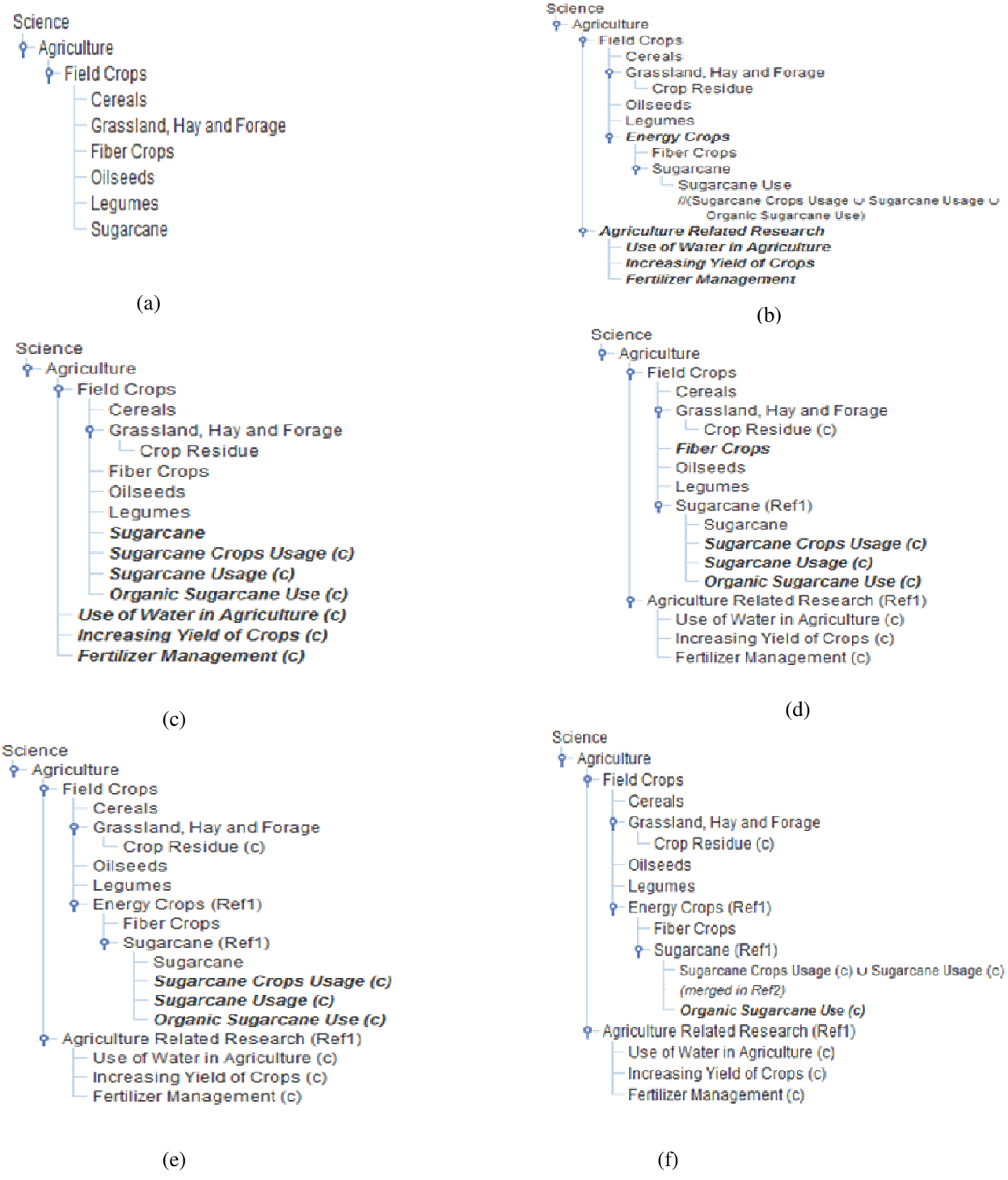


Figure 7.6. a) Sample partial base hierarchy b) Sample Target Hierarchy c) Hierarchy after insertions d) Intermediate hierarchy after Ref1at the leaf level e) Hierarchy after Ref1 f) Hierarchy after Ref2

7.5 Conclusions

This research proposes a system which can be used to construct a topic hierarchy using topical clusters and a base hierarchy. For this, a novel approach of inserting topical clusters as categories into a base hierarchy is proposed. This research also proposes a refinement procedure to incorporate the changes made in due course of updates to make the hierarchy more cohesive. The system also provides a way to prepare the data so that topical clusters obtained from existing hierarchical clustering algorithms can be compared with the categories of the hierarchy. The proposed system works as expected because we are able to generate homogeneous clusters and required data for hierarchy categories using queries' context.

The proposed method is tested using standard datasets by comparing resultant hierarchies with respective target hierarchies. Experimental evaluations on both lexical and taxonomical measures show encouraging results. Results after refining the hierarchy show considerable improvements with respect to all the measures. Splitting of the categories may also be required after many successive additions. The work might further be carried towards automatic labeling of categories in the hierarchy.

CHAPTER 08	*Achievements, Limitations, Future Work and Summary*

8.1 Summary of Achievements

The work in this thesis focuses on improving the search engine performance by effective query recommendation techniques and information organization. An attempt has been made to identify and incorporate context/intention of queries in information retrieval. A technique for automatic hierarchy building also proposed to improve the performance of search engines. The achievements of this thesis are enumerated below:

- An online process to identify the concepts which are relevant to the given query (Refer ***Concept Selection Module*** in Figure 1.4).
- A technique to identify various topics/intentions of a query has been proposed. The technique is applied on the search results of a query. Assignment of the relevant documents is done so that users can understand the search results clearly and can use the associated concepts to rephrase or enhance the query (Refer ***Topic Set Identification Module*** in Figure 1.4).
- Design and development of a hierarchical clustering approach that clusters queries with similar intentions and the concepts related to an intention and the associated documents. This approach captures the users' preferences by using click-through log. The click-through log is first represented as a graph using the proposed tripartite graph structure and then the proposed hierarchical agglomerative clustering algorithm is applied on it to produce clusters of queries, query-context aware clusters of documents and clusters of concepts. The concept clusters are referred as topical clusters. Because each concept cluster has concepts related to only one topic. The effect of noise in the click-through data is suppressed by using the selected features using the Concept Selection Module. This approach is referred as ***QDC Clustering (***Refer ***Traditional Clustering Module*** in Figure 1.4).

- QDC Clustering Module produces hard clusters. A two-phase Split-Merge (SM) algorithm is proposed to produce soft and homogeneous topical document clusters from the set of hard clusters. The proposed SM algorithm can be applied on hard clusters obtained using any traditional query-document clustering algorithm (Refer ***Soft Clustering Module*** in Figure 1.4).
- An efficient incremental model for query-document clustering has been proposed. This model helps in efficiently updating the clusters, as and when required, without re-clustering the complete data. This model is applicable on clusters produced by both traditional clustering algorithms (eg. QDC algorithm) and soft clustering algorithms (eg. SM algorithm) (Refer ***Incremental Hierarchical Clustering Module*** in Figure 1.4).
- A technique to build a topic hierarchy automatically using topical clusters and a base hierarchy is developed. For this, a novel approach of inserting topical clusters as categories into a base hierarchy is proposed. We also proposed a refinement procedure to incorporate the changes made in due course of time to make the hierarchy more cohesive. The system also provides a way to prepare the data so that topical clusters obtained from hierarchical clustering algorithm can be compared with the categories of the hierarchy. The proposed system works as expected because we are able to generate homogeneous clusters and required data for hierarchy categories using queries' context (Refer ***Relationship Based Hierarchy Building Module*** in Figure 1.4).
- The overall system is given in Figure 1.4. The interconnections between different modules can be clearly seen in the figure. Different modules can be used independently to improve the performance of an information retrieval system by addressing different issues. Two or more modules are also combined effectively to improve the quality of search results. In the automatic hierarchy building, all modules, except Incremental Hierarchical Clustering Module and Topic Labelling in Topic Set identification Module, are used together effectively to produce a hierarchy which is close to the target hierarchy. The overall system can be integrated with an existing search engine to improve the quality of search results and user experience.

8.2 Limitations and Future work

- SM algorithm can be further refined by introducing soft topic segmentation to find topic sets.
- In the automatic topic hierarchy building, splitting of the categories may also be required after many successive additions.
- The work might further be carried towards automatic labeling of categories in the topic hierarchy.
- The work can be extended to work on image retrieval systems.

8.3 Summary

Thesis starts with a brief description and needs of a search engine for the performance improvement in terms of effective and efficient search process. The challenges, motivations and a brief description of the steps involved in the proposed work are given.

The process of concept/feature extraction and selection from the set of keywords extracted from the documents' snippet obtained as a search results is illustrated in chapter 2. These concepts along with the query can be used as a query recommendation for the effective search process.

The models for finding similar queries and their related documents have been demonstrated in chapter 3. Four models for finding similar queries are proposed. Non-personalized content-ignorant model, non-personalized content-aware model which strengthen the link between query and concepts based on the corresponding clicked documents. Other two models are personalized versions of content-ignorant and content-aware models. They incorporate each user's information individually.

Identification of various topics (i.e. topic set) intended for a query and assignment of the documents to these topics has been demonstrated in chapter 4. Each topic in the topic set is described by the set of descriptor terms of the topic by revealing the ambiguous meaning of the web search query.

A split-merge (SM) post processing technique, which works over a set of document and concept and query clusters, to produce topical soft document clusters is explained in chapter 5. This

topical document clustering technique groups the documents based on the topics which are generated by disambiguating search results.

The incremental models which can accommodate the query/concept/topic drift efficiently have been exemplified in Chapter 6. The model, applicable to hierarchical clustering algorithms, updates query and document clusters at different time stamps and produces clusters that are very close to the respective clusters obtained by re-clustering on the entire data set (static model).

A system for organizing the information automatically and effectively by building the topic hierarchy is shown in chapter 7. Hierarchy nodes are topics represented by the concepts obtained from snippets. The snippets of documents, clicked for the queries, are clustered through query-context aware hierarchical clustering algorithms and then refined using a post processing technique to have coherent soft topical clusters representing one topic each. Obtained topical clusters are inserted into the base hierarchy by establishing relation(s) with the category(s) in the current hierarchy iteratively. A refinement method to reorganize the hierarchy is also illustrated to deal with the problems of topic drift and reduced homogeneity. The information stored for each node/topic in the hierarchy like queries, related documents and related concepts, etc. are useful in improving the search engine performance.

List of References

[Aggarwal and Zhai 2012]. Charu C. Aggarwal, and Cheng Xiang Zhai. 2012. A Survey of Text Clustering Algorithms. Mining Text Data. Charu C. Aggarwal, and Cheng Xiang Zhai (Eds.), Springer US, 77-128. DOI: http://dx.doi.org/10.1007/978-1-4614-3223-4_4.

[Agichtein et al. 2006]. Eugene Agichtein, Eric Brill, and Susan Dumais. 2006. Improving web search ranking by incorporating user behavior information. In Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '06), August 6-11, 2006, Seattle, Washington, USA. ACM, New York, NY, USA, 19-26. DOI= http://doi.acm.org/10.1145/1148170.1148177.

[Ahmed et al. 2002]. Mohamed N. Ahmed, Sameh M. Yamany, Nevin Mohamed, Aly A. Farag, and Thomas Moriarty. 2002. A modified fuzzy c-means algorithm for bias field estimation and segmentation of MRI data. In IEEE Transactions on Medical Imaging, Vol. 21, Article 3. IEEE, 193-199.

[Aiello et al. 2013]. Luca Maria Aiello, Georgios Petkos, Carlos Martin, David Corney, Symeon Papadopoulos, Ryan Skraba, Ayse Goker, Ioannis Kompatsiaris, and Alejandro Jaimes. 2013. Sensing trending topics in Twitter. In IEEE transactions on multimedia, Vol. 15, Article 6. IEEE, 1268-1282.

[AlltheWeb 2015]. AlltheWeb. http://search.yahoo.com/ (Last accessed April 9, 2015).

[AltaVista 2015]. AltaVista. http://www.altavista.com/ (Last accessed April 9, 2015).

[Amigó et al. 2009]. Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. 2009. A comparison of extrinsic clustering evaluation metrics based on formal constraints. In Information Retrieval, Vol. 12, Issue 4 (August 2009). Springer Netherlands, 461-486. DOI: http://dx.doi.org/10.1007/s10791-008-9066-8.

[AOL 2006]. AOL 2006: http://research.aol.com/ User Session Collection. Formerly available from research.aol.com (Last Accessed May 2006).

[AOL Search 2015]. AOL Search. http://www.aol.com/ (Last accessed April 9, 2015).

[ASK 2015]. ASK. http://www.ask.com/ (Last Accessed March 28, 2015).

[Baeza-Yates and Ribeiro-Neto 1999]. Ricardo Baeza-Yates and Berthier Ribeiro-Neto. 1999. Modern Information Retrieval. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 75–79.

[Baeza-Yates et al. 2004a]. Ricardo Baeza-Yates, Carlos Hurtado, and Marcelo Mendoza. 2004a. Query Clustering for Boosting Web Page Ranking. In Advances in Web Intelligence. Jesús Favela, Ernestina Menasalvas, and Edgar Chávez (Eds.), Lecture Notes in Computer Science, Vol. 3034. Springer Berlin Heidelberg, 164-175. DOI: http://dx.doi.org/10.1007/978-3-540-24681-7_19.

[Baeza-Yates et al. 2004b]. Ricardo Baeza-Yates, Carlos Hurtado, and Marcelo Mendoza. 2004b. Query recommendation using query logs in search engines. In Proceedings of the 2004 international conference on Current Trends in Database Technology (EDBT'04), March 14-18, 2004, Heraklion, Crete, Greece. Wolfgang Lindner, Marco Mesiti, Can Türker, Yannis Tzitzikas, and Athena I. Vakali (Eds.). Springer-Verlag, Berlin, Heidelberg, 588-596. DOI: http://dx.doi.org/10.1007/978-3-540-30192-9_58.

[Baeza-Yates et al. 2007]. Ricardo Baeza-Yates, Carlos Hurtado, and Marcelo Mendoza. 2007. Improving search engines by query clustering. In Journal of the American Society for Information Science and Technology, Vol. 58, Issue 12 (October 2007). Wiley InterScience,1793-1804. DOI: http://dx.doi.org/10.1002/asi.20627.

[Balcan and Blum 2008]. Maria-Florina Balcan, and Avrim Blum. 2008. Clustering with Interactive Feedback. In Algorithmic Learning Theory. Lecture Notes in Computer Science, Springer Berlin Heidelberg, Vol. 5254. Springer Berlin Heidelberg, 316-328. DOI: http://dx.doi.org/10.1007/978-3-540-87987-9_27.

[Ball and Hall 1965]. Geoffrey H. Ball and David J. Hall. 1965. ISODATA: a novel method of data analysis and pattern classification. Technical Report, Stanford Research Institute, California, USA, 72 pages.

[Ball and Hall 1967]. G. H. Ball and D. J. Hall. 1967. A clustering technique for summarizing multivariate data. In Behavioral Science, Vol. 12, Article 2 (April 1967). Wiley Science, 153-155.

[Barker and Cornacchia 2000]. Ken Barker and Nadia Cornacchia. 2000. Using Noun Phrase Heads to Extract Document Keyphrases. In Proceedings of the 13th Biennial Conference of the Canadian Society on Computational Studies of Intelligence: Advances in Artificial Intelligence (AI '00), May 14-17, 2000, Montreal, Quebec, Canada. Howard J. Hamilton (Ed.). Springer-Verlag, London, UK, UK, 40-52.

[Beeferman and Berger 2000]. Doug Beeferman and Adam Berger. 2000. Agglomerative clustering of a search engine query log. In Proceedings of the 6th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '00), August 20-23, 2000, Boston, MA, USA. ACM, New York, NY, USA, 407-416. DOI: http://doi.acm.org/10.1145/347090.347176.

[Belkin 2000] Nicolas J. Belkin. 2000. Helping people find what they don't know. In proceedings of the communications of the ACM, Vol. 43, Issue 8 (August 2000). ACM, ACM New York, NY, USA, 58-61. DOI=10.1145/345124.345143 http://doi.acm.org/10.1145/345124.345143.

[Bendersky et al. 2011]. Michael Bendersky, W. Bruce Croft, and Yanlei Diao. 2011. Quality-biased ranking of web documents. In Proceedings of the 4th ACM international conference on Web search and data mining (WSDM '11), February 09 - 12, 2011, Kowloon, Hong Kong. ACM, New York, NY, USA, 95-104. DOI: http://doi.acm.org/10.1145/1935826.1935849.

[Berger and Mittal 2000]. Adam L. Berger and Vibhu O. Mittal. 2000. OCELOT: a system for summarizing Web pages. In Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '00), July 24-28, 2000, Athens, Greece. ACM, New York, NY, USA, 144-151. DOI: http://doi.acm.org/10.1145/345508.345565.

[Berkhin 2006]. Pavel Berkhin. 2006. A Survey of Clustering Data Mining Techniques. In Grouping Multidimensional Data, Jacob Kogan, Charles Nicholas, Marc Teboulle (Eds.). Springer-Verlag, Berlin, Heidelberg, 25-71. DOI: http://dx.doi.org/10.1007/3-540-28349-8_2.

[Bing 2015]. Bing. https://www.bing.com/ (Last accessed April 9, 2015).

[Biswas et al. 1998]. Gautam Biswas, Jerry B. Weinberg, and Douglas H. Fisher. 1998. ITERATE: a conceptual clustering algorithm for data mining. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, Vol. 28, Issue 2 (May 1998). IEEE, 219-230. DOI: http://dx.doi.org/10.1109/5326.669556.

[Blackle.com 2015]. Blackle.com. http://www.blackle.com/ (Last accessed April 9, 2015).

[Blei and Lafferty 2006]. David M. Blei and John D. Lafferty. 2006. Dynamic topic models. In Proceedings of the 23rd international conference on Machine learning (ICML '06), June 25-29, 2006, Pittsburgh, Pennsylvania, USA. ACM, New York, NY, USA, 113-120. DOI: http://doi.acm.org/10.1145/1143844.1143859.

[Blei et al. 2003]. David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. In The Journal of Machine Learning Research, Vol. 3 (March 2003). JMLR.org 993-1022.

[Blekko 2015]. Blekko. http://blekko.com/ (Last accessed April 9, 2015).

[Boldi et al. 2009]. Paolo Boldi, Francesco Bonchi, Carlos Castillo, Debora Donato, and Sebastiano Vigna. 2009. Query suggestions using query-flow graphs. In Proceedings of the 2009 workshop on Web Search Click Data (WSCD '09), Feb 9, 2009, Barcelona, Spain. ACM, New York, NY, USA, 56-63.

[Bracewell et al. 2005]. David B. Bracewell, Fuji Ren, and Shingo Kuriowa. 2005. Multilingual single document keyword extraction for information retrieval. In Proceedings of 2005 IEEE International Conference on Natural Language Processing and Knowledge Engineering, (IEEE NLP-KE '05), October 30- November 1, 2005, Wuhan, China. Association for Computational Linguistics, Stroudsburg, PA, USA, 517-522. DOI: http://dx.doi.org/10.1109/NLPKE.2005.1598792.

[Bradley et al. 2000]. Paul S. Bradley, Usama M. Fayyad, and Cory Reina. 2000. Clustering very large databases using EM mixture models. In Proceedings of the 15th International Conference on Pattern Recognition, September 3-8 2000, Barcelona, Spain. Vol. 2, IEEE, 76-80. DOI: http://dx.doi.org/10.1109/ICPR.2000.906021.

[Brin and Page 1998]. Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hyper textual Web search engine. In Proceedings of the 7th international conference on World Wide Web 7 (WWW7), Philip H. Enslow, Jr. and Allen Ellis (Eds.). Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, 107-117.

[Broccolo et al. 2010]. Daniele Broccolo, Ophir Frieder, Franco Maria Nardini, Raffaele Perego, and Fabrizio Silvestri. 2010. Incremental algorithms for effective and efficient query recommendation. In Proceedings of the 17th international conference on String processing and information retrieval (SPIRE'10), October 11-13, 2010, Los Cabos, Mexico. Edgar Chavez and Stefano Lonardi (Eds.). Springer-Verlag, Berlin, Heidelberg, 13-24.

[CABS 2015]. CABS120k08: http://www.michael-noll.com/cabs120k08/. Retrieved March 3, 2011 from http://www.michael-noll.com/cabs120k08/ (Last Accessed February 28, 2015).

[Cao et al. 2008]. Huanhuan Cao, Daxin Jiang, Jian Pei, Qi He, Zhen Liao, Enhong Chen, and Hang Li. 2008. Context-aware query suggestion by mining click-through and session data. In Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '08), August 24-27, 2008, Las Vegas, NV, USA. ACM, New York, NY, USA, 875-883. DOI: http://doi.acm.org/10.1145/1401890.1401995.

[Carmel et al. 2002]. David Carmel, Eitan Farchi, Yael Petruschka, and Aya Soffer. 2002. Automatic query refinement using lexical affinities with maximal information gain. In Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '02), August 11-15, 2002, Tampere, Finland. ACM, New York, NY, USA, 283-290. DOI: http://doi.acm.org/10.1145/564376.564427.

[Carpineto and Romano 2004]. Claudio Carpineto and Giovanni Romano. 2004. Exploiting the potential of concept lattices for information retrieval with CREDO. In Journal of Universal Computer Science, Vol. 10, Issue 8. 985-1013.

[Carpineto and Romano 2008]. Claudio Carpineto, and Giovanni Romano. 2008. Ambient dataset, http://credo.fub.it/ambient/ (Last Access :October 15, 2014).

[Carpineto and Romano 2012]. Claudio Carpineto and Giovanni Romano. 2012. A Survey of Automatic Query Expansion in Information Retrieval. In ACM Computing Surveys, Vol. 44, Issue 1, Article 1 (January 2012). ACM, New York, NY, USA, 50 pages. DOI: http://doi.acm.org/10.1145/2071389.2071390.

[Carpineto et al. 2009]. Claudio Carpineto, Stanislaw Osiński, Giovanni Romano, and Dawid Weiss. 2009. A survey of Web clustering engines. In ACM Computing Surveys, Vol. 41, Issue 3, Article 17 (July 2009). ACM, New York, NY, USA, 38 pages. DOI: http://doi.acm.org/10.1145/1541880.1541884.

[Carrot2 2015]. Carrot2: http://sourceforge.net/projects/carrot2/files/carrot2/3.9.3/ [Last Accessed February 28, 2015).

[Castillo 2004]. Carlos Castillo. 2004. Effective web crawling. Ph.D. dissertation, University of Chile.

[Cheng et al. 2005]. David Cheng, Santosh Vempala, Ravi Kannan, and Grant Wang. 2005. A divide-and-merge methodology for clustering. In Proceedings of the 24th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'05), June 13 – 15, 2005, Baltimore, Maryland, USA. ACM, New York, NY, USA, 196–205. DOI: http://doi.acm.org/10.1145/1065167.1065192.

[Chien and Immorlica 2005]. Steve Chien and Nicole Immorlica. 2005. Semantic similarity between search engine queries using temporal correlation. In Proceedings of the 14th international conference on World Wide Web (WWW '05), May 10-14, 2005, Chiba, Japan. ACM, New York, NY, USA, 2-11. DOI: http://doi.acm.org/10.1145/1060745.1060752.

[Cimiano et al. 2009]. Philipp Cimiano, Alexander Madche, Steffen Staab, and Johanna Volker. 2009. Ontology learning. S. Staab and R. Studer (Eds.), Handbook on ontologies. Springer Berlin Heidelberg, 245-267. DOI: http://dx.doi.org/10.1007/978-3-540-92673-3_11.

[Cover and Thomas 2005]. Thomas M. Cover and Joy A. Thomas. 2005. Entropy, Relative Entropy, and Mutual Information, Elements of Information Theory, Second Edition, John Wiley & Sons, Inc., Hoboken, NJ, USA, 12-49. DOI: http://dx.doi.org/10.1002/047174882X.ch2.

[Daum 2015]. Daum. http://www.daum.net/ (Last accessed April 9, 2015).

[Deerwester et al. 1990]. Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. In Journal of the American Society for Information Science, Vol. 41, Article 6. 391–407.

[Dellschaft and Staab 2006]. Klaas Dellschaft and Steffen Staab. 2006. On how to perform a gold standard based evaluation of ontology learning. In Proceedings of the 5th international conference on The Semantic Web (ISWC'06), November 5-9, 2006, Athens, GA, USA. Isabel Cruz, Stefan Decker, Dean Allemang, Chris Preist, and Daniel Schwabe (Eds.). Springer-Verlag, Berlin, Heidelberg, 228-241. DOI: http://dx.doi.org/10.1007/11926078_17.

[Dogpile 2015]. Dogpile. http://www.dogpile.com/ (Last accessed April 9, 2015).

[DuckDuckGo 2015]. DuckDuckGo. https://duckduckgo.com/ (Last accessed April 9, 2015).

[Duda et al. 2000]. Richard O. Duda, Peter.E. Hart, and David.G. Stork. Pattern classification. John Wiley & Sons, 2000.

[Dupret and Mendoza 2005]. Georges Dupret, and Marcelo Mendoza. 2005. Recommending Better Queries from Click-Through Data. In Proceedings of the 12th International Symposium on String Processing and Information Retrieval (SPIRE'05), November 2-4, 2005, Buenos Aires, Argentina. Mariano Consens, and Gonzalo Navarro (Eds.). Springer Berlin Heidelberg, 41-44. DOI: http://dx.doi.org/10.1007/11575832_5.

[Ernesto et al. 2004]. D'Avanzo Ernesto, Bernardo Magnini, and Alessandro Vallin. 2004. Keyphrase extraction for summarization purposes: The LAKE system at DUC-2004. In Proceedings of the 2004 document understanding conference (DUC) workshop, Human language technology conference/North American chapter of the Association for Computational Linguistics Annual Meeting 2004, May 6-7, 2004, Boston, USA.

[Everitt et al. 2001]. Brian Everitt, Sabine Landau, and Morven Leese. 2001. Cluster Analysis. 4th ed, Oxford University Press, Oxford, UK.

[Excite 2015]. Excite. http://www.excite.com/ (Last Accessed March 28, 2015).

[Ferragina and Gulli 2005]. Paolo Ferragina and Antonio Gulli. 2005. A personalized search engine-based on Web-snippet hierarchical clustering. In Special interest tracks and posters of the 14th international conference on World Wide Web (WWW '05), May 10 – 14, 2005, Chiba, Japan. ACM, New York, NY, USA, 801-810.

[Fonseca et al. 2003]. Bruno M. Fonseca, Paulo B. Golgher, Edleno S. de Moura, and Nivio Ziviani. 2003. Using Association Rules to Discover Search Engines Related Queries. In Proceedings of the 1st Conference on Latin American Web Congress (LA-WEB '03), November 12, 2003, Santiago, Chile. IEEE Computer Society, Washington, DC, USA, 66-71.

[Fonseca et al. 2005]. Bruno M. Fonseca, Paulo B. Golgher, Edleno S. de Moura, and Nivio Ziviani. 2005. Concept Based Interactive Query Expansion. In Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM'05), October 31-November 05, 2005, Bremen, Germany. ACM, New York, NY, USA, 696-703. DOI: http://doi.acm.org/10.1145/1099554.1099726.

[Forestle 2015]. Forestle. https://www.ecosia.org/ (Last accessed April 9, 2015).

[Frank et al. 1999]. Eibe Frank, Gordon W. Paynter, Ian H. Witten, and Carl Gutwin et al. 1999. Domain-specific keyphrase extraction. In Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI '99), July 31 - August 6, 1999, Stockholm, Sweden. Morgan Kaufmann Publishers, San Francisco, CA, USA, Vol. 2, 668-673.

[Fung et al. 2003]. Benjamin C.M. Fung, Ke Wang and Martin Ester. 2003. Hierarchal Document Clustering using Frequent Itemsets. In Proceedings of the 3rd SIAM International Conference On Data Mining 2003 (SDM 2003), May 1-3, 2003, San Francisco, CA, USA. 59-70.

[Gauch and Wang 1997]. Susan Gauch and Jianying Wang. 1997. A corpus analysis approach for automatic query expansion. In Proceedings of the 6th international conference on Information and knowledge management (CIKM '97), November 10-14, 1997, Las Vegas, NV, USA. ACM, New York, NY, USA, 278-284. DOI: http://doi.acm.org/10.1145/266714.266910.

[GenieKnows 2015]. GenieKnows. http://www.genieknows.com/ (Last accessed April 9, 2015).

[Gigablast 2015]. Gigablast. http://www.gigablast.com/ (Last accessed April 9, 2015).

[Glover et al. 2002]. Eric J. Glover, Kostas Tsioutsiouliklis, Steve Lawrence, David M. Pennock, and Gary W. Flake. 2002. Using web structure for classifying and describing web pages. In Proceedings of the 11th international conference on World Wide Web (WWW '02), May 07-11, 2002, Honolulu, Hawaii, USA. ACM, New York, NY, USA, 562-569. DOI: http://doi.acm.org/10.1145/511446.511520.

[Go.com 2015]. Go.com. http://go.com/ (Last accessed April 9, 2015).

[GoodSearch 2015]. GoodSearch. http://www.goodsearch.com/ (Last accessed April 9, 2015).

[Google 2015]. Google. http://google.com/(Last Accessed March 28, 2015).

[Goyal and Mehala 2011]. Poonam Goyal and N. Mehala. 2011. Concept based query recommendation. In Proceedings of the 9th Australasian Data Mining Conference (AusDM '11), December 1-2, 2011, Ballarat, Australia, Peter Vamplew, Andrew Stranieri, Kok-Leong Ong, Peter Christen, and Paul J. Kennedy (Eds.), Vol. 121. Australian Computer Society, Inc., Darlinghurst, Australia, Australia, 69-78.

[Goyal et al. 2013]. Poonam Goyal, N. Mehala, and Ankur Bansal. 2013. A robust approach for finding conceptually related queries using feature selection and tripartite graph structure. In Journal of Information Science, Vol. 39, Article 5 (October 2013). SAGE, 575-592. DOI: http://dx.doi.org/10.1177/0165551513477819.

[Goyal et al. 2015]. Poonam Goyal, N. Mehala and Divyansh Bhatia. Topical Document clustering: Hard to Soft Two-stage Post Processing Technique. In Journal of Information Science (JIS). SCI, SAGE.

[GSON 2014]. GSON: Available at http://code.google.com/p/google-gson/ (Last Accessed: October 31, 2014).

[Gupta et al. 2010]. Ranjna Gupta, Neelam Duhan, A. K. Sharma, and Neha Aggarwal. 2010. Query Based Duplicate Data Detection on WWW. In International Journal on Computer Science and Engineering (IJCSE), Vol.2, Issue 4 (July 2010). 1395-1400.

[HaCohen-Kerner et al. 2005]. Yaakov HaCohen-Kerner, Zuriel Gross, and Asaf Masa. 2005. Automatic Extraction and Learning of Keyphrases from Scientific Articles. In Computational Linguistics and Intelligent Text Processing. Alexander Gelbukh, (Ed.).

Lecture Notes in Computer Science, Vol. 3406, Springer Berlin Heidelberg, 657-669. DOI: http://dx.doi.org/10.1007/978-3-540-30586-6_74 .

[Halalgoogling 2015]. Halalgoogling. http://halalgoogling.com/ (Last accessed April 9, 2015).

[Hammouda and Kamel 2003]. Khaled M. Hammouda and Mohamed S. Kamel. 2003. Incremental Document Clustering Using Cluster Similarity Histograms. In Proceedings of the 2003 IEEE/WIC International Conference on Web Intelligence (WI '03), October 13-17, 2003, Halifax, Canada. IEEE Computer Society, Washington, DC, USA, 597-601.

[Hammouda et al. 2005]. Khaled M. Hammouda, Diego N. Matute, and Mohamed S. Kamel. 2005. CorePhrase: keyphrase extraction for document clustering. In Proceedings of the 4th international conference on Machine Learning and Data Mining in Pattern Recognition (MLDM '05), July 9-11, 2005, Leipzig, Germany. Petra Perner and Atsushi Imiya (Eds.). Springer-Verlag, Berlin, Heidelberg, 265-274. DOI: http://dx.doi.org/10.1007/11510888_26.

[Han et al. 2001]. Jiawei Han, Micheline Kamber and Anthony K. H. Tung. 2001. Spatial Clustering Methods in Data Mining: A Survey. Geographic Data Mining and Knowledge Discovery. Taylor & Francis, Inc, Bristol, PA, USA, 1-29.

[Hearst 2009]. Search User Interfaces. Cambridge university Press, 2009.

[Hearst and Pedersen 1996]. Marti A. Hearst and Jan O. Pedersen. 1996. Reexamining the cluster hypothesis: scatter/gather on retrieval results. In Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '96), August 18-22, 1996, Zurich, Switzerland. ACM, New York, NY, USA, 76-84. DOI: http://doi.acm.org/10.1145/243199.243216.

[HotBot 2015]. HotBot. http://www.hotbot.com/ (Last accessed April 9, 2015).

[Houlsby and Ciaramita 2013]. N. Houlsby, and M. Ciaramita, 2013. Scalable probabilistic entity-topic modeling. arXiv preprint arXiv:1309.0337.

[Huang et al. 2000]. Chien-Kang Huang, Lee-Feng Chien, and Yen-Jen Oyang. 2000. Clustering Similar Query Sessions Toward Interactive Web Search. In Proceedings of Research on Computational Linguistics Conference XIII (ROCLING2000), August 24-25, 2000, Taipei, Taiwan. Association for Computational Linguistics and Chinese Language Processing (ACLCLP), 115-134.

[Hulth 2003]. Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In Proceedings of the 2003 conference on Empirical methods in natural language processing (EMNLP '03), July 11-12, 2003, Sapporo, Japan. Association for Computational Linguistics, Stroudsburg, PA, USA, 216-223. DOI: http://dx.doi.org/10.3115/1119355.1119383.

[Hung and Yang 2001]. Ming-Chuan Hung and Don-Lin Yang. 2001. An Efficient Fuzzy C-Means Clustering Algorithm. In Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM '01), November 29-December 2, 2001, San Jose, California, USA. Nick Cercone, Tsau Young Lin, and Xindong Wu (Eds.). IEEE Computer Society, Washington, DC, USA, 225-232.

[ILS 2015]. Internet Live Stats (ILS). http://www.internetlivestats.com/ (Last accessed April 9, 2015).

[Info.com 2015]. Info.com. http://info.com/ (Last accessed April 9, 2015).

[Ixquick 2015]. Ixquick. https://www.ixquick.com/ (Last accessed April 9, 2015).

[Jain and Dubes 1988]. Anil K. Jain and Richard C. Dubes. 1988. Algorithms for Clustering Data. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

[Jain et al. 1999]. A. K. Jain, M. N. Murty, and P. J. Flynn. 1999. Data clustering: a review. In ACM Computing Surveys (CSUR), Vol. 31, Issue 3 (September 1999). ACM New York, NY, USA, 264-323. DOI: http://doi.acm.org/10.1145/331499.331504.

[Jansen and Spink 2006]. Bernard J. Jansen and Amanda Spink. 2006. How are we searching the world wide web? A comparison of nine search engine transaction logs. In Information Processing and Management: an International Journal - Special issue: Formal methods for information retrieval, Vol. 42, Issue 1 (January 2006). Pergamon Press, Inc. Tarrytown, NY, USA, 248-263. DOI: http://dx.doi.org/10.1016/j.ipm.2004.10.007.

[Jansen et al. 1998]. Bernard J. Jansen, Amanda Spink, Judy Bateman, and Tefko Saracevic. 1998. Real life information retrieval: a study of user queries on the Web. In SIGIR Forum, Vol. 32, Article 1 (April 1998). ACM New York, NY, USA, 5-17. DOI: http://dx.doi.org/10.1145/281250.281253.

[Jansen et al. 2009]. Bernard J. Jansen, Danielle L. Booth, and Amanda Spink. 2009. Patterns of query reformulation during Web searching. In Journal of the American Society for Information Science and Technology, Vol. 60, Issue 7 (July 2009). Wiley Online Library, 1358-1371. DOI: http://dx.doi.org/10.1002/asi.v60:7.

[Jiang et al. 2011]. J. Jiang, S. Han, J. Wu, and D. He. Pitt at TREC 2011 session track. In Proceedings of the 20th Text REtrieval Conference (TREC '11), Gaithersburg, MD, USA.

[Jing and Croft 1994]. Yufeng Jing and W. Bruce Croft. 1994. An association thesaurus for information retrieval. In Proceedings of the Intelligent Multimedia Information Retrieval Systems (RIAO '94), October 11-13, 1994, New York, NY. 146-160.

[Joachims 2002]. Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In Proceedings of the 8th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '02), July 23-25, 2002, Edmonton, AB, Canada. ACM, New York, NY, USA, 133-142. DOI: http://doi.acm.org/10.1145/775047.775067.

*[Joachims et al. 2005].*Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2005. Accurately interpreting clickthrough data as implicit feedback. In Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '05), August 15-19, 2005, Salvador, Brazil. ACM, New York, NY, USA, 154-161. DOI: http://doi.acm.org/10.1145/1076034.1076063.

[Jones 1971]. Karen Sparck Jones. 1971. Automatic Keyword Classification for Information Retrieval. Butterworths, London, UK.

[Jones 1972]. Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. In Document retrieval systems, Peter Willett (Ed.). Taylor Graham Series In Foundations Of Information Science, Vol. 3. Taylor Graham Publishing, London, UK, 132-142.

[Jones et al. 2006]. Rosie Jones, Benjamin Rey, Omid Madani, and Wiley Greiner. 2006. Generating query substitutions. In Proceedings of the 15th international conference on World Wide Web (WWW '06), May 22-26, 2006, Edinburgh, Scotland, UK. ACM, New York, NY, USA, 387-396. DOI: http://doi.acm.org/10.1145/1135777.1135835.

[Kim and Lee 2002]. Han-joon Kim and Sang-goo Lee. 2002. Discovering Taxonomic Relationships from Textual Documents. In Proceedings in IASTED Applied Informatics, February 18 – 21, 2002, Innsbruck, Austria. 344-349.

[Knijff et al. 2013]. Jeroen De Knijff, Flavius Frasincar, and Frederik Hogenboom. 2013. Domain taxonomy learning from text: The subsumption method versus hierarchical clustering. In Data and Knowledge Engineering, Vol. 83 (January 2013). Elsevier, 54-69. DOI: http://dx.doi.org/10.1016/j.datak.2012.10.002.

[Kondadadi and Kozma 2002]. R. Kondadadi, and R. Kozma. 2002. A modified fuzzy art for soft document clustering. In Proceedings of the International Joint Conference on Neural Network, (IJCNN'02), May 12-17, 2002, Honolulu, Hawaii. IEEE, Vol. 3, 2545-2549. DOI: http://dx.doi.org/10.1109/IJCNN.2002.1007544.

[Korf 1985]. R. E. Korf, 1985. Depth-first iterative-deepening: An optimal admissible tree search. In Artificial intelligence, Vol. 27, Issue 1. 97-109.

[Kosovac et al. 2000]. Branka Kosovac, Dana J. Vanier, and Thomas M. Froese. 2000. Use of keyphrase extraction software for creation of an Aec/fm Thesaurus. In Electronic Journal of Information Technology in Construction, Vol. 5. 25-36.

[Koster 1994]. Martijn Koster. 1994. ALIWEB—Archie-like indexing in the WEB. In Computer Networks and ISDN Systems, Vol. 27, Issue 2 (November 1994). Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, 175-182. DOI: http://dx.doi.org/10.1016/0169-7552(94)90131-7.

[Krulwich and Burkey 1996]. Bruce Krulwich and Chad Burkey. 1996. Learning user information interests through extraction of semantically significant phrases. In Proceedings of the AAAI spring symposium on machine learning in information access (AAAI '96), March 1996, Stanford, California, USA. Marti A. Hearst, Haym Hirsh (Eds.). AAAI Press, California, USA, 100-112.

[Kummamuru et al. 2004]. Krishna Kummamuru, Rohit Lotlikar, Shourya Roy, Karan Singal, and Raghu Krishnapuram. 2004. A hierarchical monothetic document clustering algorithm for summarization and browsing search results. In Proceedings of the 13th International Conference on World Wide Web (WWW'04), May 17 – 22, 2004, New York, NY, USA. ACM, New York, NY, USA, 658–665. DOI: http://doi.acm.org/10.1145/988672.988762.

[Lang et al. 2008]. Hao Lang, Bin Wang, Gareth Jones, Jin-Tao Li, Fan Ding, and Yi-Xuan Liu. 2008. Query performance prediction for information retrieval based on covering topic score. In Journal of Computer Science and Technology, Vol. 23, Issue 4 (July 2008). Springer US, 590-601. DOI: http://dx.doi.org/10.1007/s11390-008-9155-6.

[LCSH 2015]. "Library of Congress Subject Headings (LCSH). http://lcweb.loc.gov/cds/lcsh.html (Last accessed February 28, 2015).

[Leung et al. 2008]. Kenneth Wai-Ting Leung, Wilfred Ng, and Dik Lun Lee. 2008. Personalized Concept-Based Clustering of Search Engine Queries. In IEEE Transactions on Knowledge and Data Engineering, Vol. 20, Issue 11 (November 2008). IEEE Computer Society, Washington, DC, USA, 1505-1518. DOI: http://dx.doi.org/10.1109/TKDE.2008.84.

[Litvak and Last 2008]. Marina Litvak and Mark Last. 2008. Graph-based keyword extraction for single-document summarization. In Proceedings of the Workshop on Multi-source Multilingual Information Extraction and Summarization (MMIES '08) held at CoLing'2008, August 23, 2008, Manchester, United Kingdom. Sivaji Bandyopadhyay, Thierry Poibeau, Horacio Saggion, and Roman Yangarber (Eds.). Association for Computational Linguistics, Stroudsburg, PA, USA, 17-24. DOI: http://dl.acm.org/citation.cfm?id=1613172.1613178.

[Liu 2004]. Hugo Liu. 2004. MontyLingua: An end-to-end natural language processor with common sense. Available at: web.media.mit.edu/~hugo/montylingua (Last Accessed : October 31, 2014).

[Liu et al. 2009]. Zhiyuan Liu, Peng Li, Yabin Zheng, and Maosong Sun. 2009. Clustering to find exemplar terms for keyphrase extraction. In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP '09), August 6-7, 2009, Singapore. Vol. 1. Association for Computational Linguistics, Stroudsburg, PA, USA, 257-266. DOI: http://dl.acm.org/citation.cfm?id=1699510.1699544.

[Live Search 2015]. Live Search. https://www.bing.com/ (Last accessed April 9, 2015).

[Lu et al. 2009]. Nan Lu, Zhou Chun-Guang, and Cui Lai-Zhong. 2009. The Application of Association Rules Algorithm on Web Search Engine. In Proceedings of the International Conference on Computational Intelligence and Security, (CIS '09), December 11-14, 2009, Beijing, China. IEEE, 102-108. DOI: http://dx.doi.org/10.1109/CIS.2009.270.

[Lu et al. 2011]. Qiang Lu, Jack G. Conrad, Khalid Al-Kofahi, and William Keenan. 2011. Legal document clustering with built-in topic segmentation. In Proceedings of the 20th ACM international conference on Information and knowledge management (CIKM '11), October 24-28, 2011, Glasgow, United Kingdom. Bettina Berendt, Arjen de Vries, Wenfei Fan, Craig Macdonald, Iadh Ounis, and Ian Ruthven (Eds.). ACM, New York, NY, USA, 383-392. DOI: http://doi.acm.org/10.1145/2063576.2063636.

[Lycos 2015]. Lycos. http://www.lycos.com/ (Last Accessed March 28, 2015).

[MacQueen 1967]. James MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, June 21-July 18, 1965, Berkeley, USA. L. M. Le Cam and J. Neyman (Eds.), Vol. 1. University of California Press, Berkeley, CA, 281-297.

[Manning and Schütze 1999]. Chris Manning and Hinrich Schütze. 1999. Foundations of Statistical Natural Language Processing, MIT Press. Cambridge, MA (May 1999).

[Marco and Navigli 2011]. Antonio Di Marco and Roberto Navigli. 2011. Clustering web search results with maximum spanning trees. In Proceedings of the 12th International Conference on Artificial Intelligence Around Man and Beyond (AI*IA'11), September 15 – 17, 2011, Palermo, Italy. Roberto Pirrone and Filippo Sorbello (Eds.). Springer-Verlag, Berlin, Heidelberg, 201-212. DOI: http://dx.doi.org/10.1007/978-3-642-23954-0_20.

[Masłowska 2003]. Irmina Masłowska. 2003. Phrase-based hierarchical clustering of Web search results. In Proceedings of the 25th European Conference on IR Research (ECIR'03). April 14 – 16, 2003, Pisa, Italy. Fabrizio Sebastiani (Eds.). Lecture Notes in Computer Science, Vol. 2633. Springer Berlin Heidelberg, 555–562. DOI: http://dx.doi.org/10.1007/3-540-36618-0_42.

[Mecca et al. 2007]. Giansalvatore Mecca, Salvatore Raunich, Alessandro Pappalardo. 2007. A new algorithm for clustering search results. In Data and Knowledge Engineering, Vol. 62, Issue 3 (September 2007). Elsevier, 504-522. DOI: http://dx.doi.org/10.1016/j.datak.2006.10.006.

[Medelyan et al. 2009]. Olena Medelyan, Eibe Frank, and Ian H. Witten. 2009. Human-competitive tagging using automatic keyphrase extraction. In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: (EMNLP '09), 6-7 August 2009, Singapore, Singapore. Vol. 3. Association for Computational Linguistics, Stroudsburg, PA, USA, 1318-1327. DOI: http://dl.acm.org/citation.cfm?id=1699648.1699678.

[Metzler et al. 2007]. Donald Metzler, Susan Dumais, and Christopher Meek. 2007. Similarity measures for short segments of text. In Advances in Information Retrieval. Lecture Notes in Computer Science, Vol. 4425 (April 2007). Springer-Verlag Berlin Heidelberg, 16-27. DOI: http://dx.doi.org/10.1007/978-3-540-71496-5_5.

[MSN 2015]. MSN. http://newsbot.msnbc.msn.com/ (Last Accessed March 28, 2015).

[Myers and Well 2003]. Jerome L. Myers and Arnold D. Well. 2003. Research Design and Statistical Analysis (second edition ed.), Lawrence Erlbaum.

[NATE 2015]. NATE. https://www.bing.com/ (Last accessed April 9, 2015).

[Naver 2015]. Naver. http://www.naver.com/ (Last accessed April 9, 2015).

[Ngo and Nguyen 2004]. Chi L. Ngo and Hung S. Nguyen. 2004. A tolerance rough set approach to clustering Web search results. In Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'04), September 20-24, 2004, Pisa, Italy. Lecture Notes in Computer Science, Vol. 3202. Springer Berlin Heidelberg, 515–517. DOI: http://dx.doi.org/10.1007/978-3-540-30116-5_51.

[Nguyen et al. 2009]. Cam-Tu Nguyen, Xuan-Hieu Phan, Susumu Horiguchi, Thu-Trang Nguyen, and Quang-Thuy Ha. 2009. Web Search Clustering and Labeling with Hidden Topics. In ACM Transactions on Asian Language Information Processing (TALIP), Vol. 8, Issue 3, Article 12 (August 2009). ACM, New York, NY, USA, 12:1-40. DOI: http://doi.acm.org/10.1145/1568292.1568295.

[Ni 2004]. Wenyi Ni. 2004. A Survey of Web Document Clustering. Southern Methodist University.

[ODP 2015]. Open directory project (ODP). http://dmoz.org/ (Last accessed February 28, 2015)

[Oikonomakou and Vazirgiannis 2005]. Nora Oikonomakou, and Michalis Vazirgiannis. 2005. A Review of Web Document Clustering Approaches. Data Mining and Knowledge Discovery Handbook. Oded Maimon, and Lior Rokach (Eds.), Springer US, Chapter 43, 921-943. DOI: http://dx.doi.org/10.1007/0-387-25465-X_43.

[Ordonez and Omiecinski 2002]. Carlos Ordonez and Edward Omiecinski. 2002. FREM: fast and robust EM clustering for large data sets. In Proceedings of the 11th international conference on Information and knowledge management (CIKM '02), November 5-8, 2002, McLean, VA, USA. ACM, New York, NY, USA, 590-599. DOI: http://doi.acm.org/10.1145/584792.584889.

[Osiński and Gotoh 2004]. Stanisław Osiński and Yoshi Gotoh. 2004. Dimensionality reduction techniques for search results clustering. Master's thesis, The University of Sheffield.

[Osiński et al. 2004]. Stanisław Osiński, Jerzy Stefanowski, and Dawid Weiss. 2004. Lingo: Search Results Clustering Algorithm Based on Singular Value Decomposition. In Intelligent Information Processing and Web Mining. Kłopotek, Mieczysław, Wierzchoń, Sławomir and Trojanowski, Krzysztof (Eds.). Advances in Soft Computing, Vol. 1494. Springer Berlin Heidelberg, 359-368. DOI: http://dx.doi.org/10.1007/978-3-540-39985-8_37.

[Petkos et al. 2014a]. Georgios Petkos, Symeon Papadopoulos, and Yiannis Kompatsiaris. 2014a. Two-level message clustering for topic detection in Twitter. In Proceedings of SNOW-2014 Data Challenge, April 8, 2014, Seoul, South Korea. CEUR, 49-56.

[Petkos et al. 2014b]. Georgios Petkos, Symeon Papadopoulos, Luca Aiello, Ryan Skraba, and Yiannis Kompatsiaris. 2014b. A soft frequent pattern mining approach for textual topic detection. In Proceedings of the 4th International Conference on Web Intelligence, Mining and Semantics (WIMS14), June 2-4, 2014, Thessaloniki, Greece. ACM, New York, NY, USA, Article 25, 10 pages. DOI: http://doi.acm.org/10.1145/2611040.2611068.

[Powerset 2015]. Powerset. https://www.bing.com/ (Last accessed April 9, 2015).

[Qiu and Frei 1993]. Yonggang Qiu and Hans-Peter Frei. 1993. Concept based query expansion. In Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '93), June 27 - July 01, 1993, Pittsburgh, PA, USA. Robert Korfhage, Edie Rasmussen, and Peter Willett (Eds.). ACM, New York, NY, USA, 160-169. DOI: http://doi.acm.org/10.1145/160688.160713.

[Riloff 1996]. Ellen Riloff. 1996. Automatically generating extraction patterns from untagged text. In Proceedings of the 13th national conference on Artificial intelligence (AAAI'96), August 4–8, 1996, Portland, Oregon. Vol. 2. AAAI Press 1044-1049.

[Rocchio 1971]. J. Rocchio. 1971. Relevance Feedback in Information Retrieval, Prentice-Hall Inc., Englewood Cliffs, NJ, Chapter 14, 313--323.

[Sahami and Heilman 2006]. Mehran Sahami and Timothy D. Heilman. 2006. A web-based kernel function for measuring the similarity of short text snippets. In Proceedings of the 15th international conference on World Wide Web (WWW '06), May 22-26, 2006, Edinburgh, Scotland UK. ACM, New York, NY, USA, 377-386. DOI: http://doi.acm.org/10.1145/1135777.1135834.

[Salton and McGill 1986]. Gerard Salton and Michael J. McGill. 1986. Introduction to Modern Information Retrieval. McGraw-Hill, Inc., New York, NY, USA.

[Salton et al. 1996]. Gerard Salton, Amit Singhal, Chris Buckley, and Mandar Mitra. 1996. Automatic text decomposition using text segments and text themes. In Proceedings of the 7th ACM conference on Hypertext (HYPERTEXT '96), March 16-20, 1996, Bethesda, MD, USA. ACM, New York, NY, USA, 53-65. DOI: http://doi.acm.org/10.1145/234828.234834.

[Sanderson 2008]. Mark Sanderson. 2008. Ambiguous queries: test collections need more sense. In Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR'08), July 20 - 24, 2008, Singapore, Singapore. ACM, New York, NY, USA, 499-506. DOI: http://doi.acm.org/10.1145/1390334.1390420.

[Sanderson and Croft 1999]. Mark Sanderson and Bruce Croft. 1999. Deriving concept hierarchies from text. In Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '99), August 15-19, 1999, Berkeley, USA. ACM, New York, NY, USA, 206-213. DOI: http://doi.acm.org/10.1145/312624.312679.

[Sarle 1995]. Warren Sarle. 1995. Why statisticians should not FART. ftp://ftp.sas.com/pub/neural/fart.doc (Last Accessed: November 1, 2014).

[Scaiella et al. 2012]. Ugo Scaiella, Paolo Ferragina, Andrea Marino, and Massimiliano Ciaramita. 2012. Topical clustering of search results. In Proceedings of the 5th ACM international conference on Web search and data mining (WSDM '12), February 8-12, 2012, Seattle, Washington, USA. ACM, New York, NY, USA, 223-232. DOI: http://doi.acm.org/10.1145/2124295.2124324.

[Schütze and Pedersen 1997]. Hinrich Schütze and Jan O. Pedersen. 1997. A cooccurrence-based thesaurus and two applications to information retrieval. In International Journal of Information Processing and Management, Vol. 33, Issue 3 (May 1997). Pergamon Press, Inc. Tarrytown, NY, USA, 307-318. DOI: http://dx.doi.org/10.1016/S0306-4573(96)00068-4.

[Schütze and Silverstein 1997]. Hinrich Schütze and Craig Silverstein. 1997. Projections for efficient document clustering. In Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '97), July 27-31, 1997, Philadelphia, PA, USA. Nicholas J. Belkin, A. Desai Narasimhalu, Peter Willett, William Hersh, Fazli Can, and Ellen Voorhees (Eds.). ACM, New York, NY, USA, 74-81. DOI: http://doi.acm.org/10.1145/258525.258539.

[Sengstock and Gertz 2011]. Christian Sengstock and Michael Gertz. 2011. CONQUER: a system for efficient context-aware query suggestions. In Proceedings of the 20th international conference companion on World wide web (WWW '11), March 28-April 01, 2011, Hyderabad, India. ACM, New York, NY, USA, 265-268. DOI: http://doi.acm.org/10.1145/1963192.1963305.

[Shaw and Xu 2009]. Gavin Shaw and Yue Xu. 2009. Enhancing an Incremental Clustering Algorithm for Web Page Collections. In Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT '09), September 15-18, 2009, Milano, Italy. Vol. 3. IEEE Computer Society, Washington, DC, USA, 81-84. DOI: http://dx.doi.org/10.1109/WI-IAT.2009.236.

[Sibson 1973]. R. Sibson. 1973. SLINK: an optimally efficient algorithm for the single-link cluster method. In The Computational Journal, Vol. 116, Issue 1. Oxford Journals, 30-34.

[Silverstein et al. 1998]. Craig Silverstein, Monika Henzinger, Hannes Marais, and Michael Moricz. 1998. Analysis of a Very Large AltaVista Query Log. Digital Equipment Corporation Systems Research Center (DEC SRC) Technical Note 1998-014.

[Sogou 2015]. Sogou. http://www.sogou.com/ (Last accessed April 9, 2015).

[Song and Croft 1999]. Fei Song and W. Bruce Croft. 1999. A general language model for information retrieval. In Proceedings of the 8th international conference on Information and knowledge management (CIKM'99), November 02-06, 1999, Kansas City, MO, USA. ACM, New York, NY, USA, 316-321. DOI: http://doi.acm.org/10.1145/319950.320022.

[Song et al. 2006]. Min Song, Il-Yeol Song, Robert B. Allen, and Zoran Obradovic. 2006. Keyphrase extraction-based query expansion in digital libraries. In Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries (JCDL '06), June 11-15, 2006, Chapel Hill, NC, USA. ACM, New York, NY, USA, 202-209. DOI: http://doi.acm.org/10.1145/1141753.1141800.

[Soso 2015]. Soso (search engine). http://www.sogou.com/?rfrom=soso (Last accessed April 9, 2015).

[Spink et al. 1998]. Amanda Spink, Judy Bateman, and Bernard J. Jansen. 1998. Searching heterogeneous collections on the Web: behaviour of Excite users. In Information Research, Vol.4, Issue 2 (October 1998). 317-328.

[Steinbach et al. 2000]. Michael Steinbach, George Karypis, and Vipin Kumar. 2000. A Comparison of Document Clustering Techniques. In Proceedings of the KDD Workshop on Text Mining at 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2000), August 20-23, 2000, Boston, MA, USA. ACM, New York, NY, USA, 109-111.

[Surdeanu et al. 2006]. Mihai Surdeanu and Jordi Turmo and Alicia Ageno. 2006. A hybrid approach for the acquisition of information extraction patterns. In Proceedings of the EACL 2006 Workshop on Adaptive Text Extraction and Mining (ATEM 2006), April 4, 2006, Trento, Italy. Association for Computational Linguistics, Stroudsburg, PA, USA, 48-56.

[Teoma 2015]. Teoma. http://www.ask.com/ (Last accessed April 9, 2015).

[Timonen 2011a]. Mika Timonen, Paula Silvonen, and Melissa Kasari. 2011a. Classification of Short Documents to Categorize Consumer Opinions. In Proceedings of 7th International Conference on Advanced Data Mining and Applications, (ADMA'11), December 17-19, 2011, Beijing, China. 1-14.

[Timonen 2011b]. Mika Timonen, Paula Silvonen, and Melissa Kasari. 2011b. Modelling a Query Space Using Associations. In Frontiers in Artificial Intelligence and Applications: Information Modelling and Knowledge Bases XXII, Vol. 255. Anneli Heimbürger, Yasushi Kiyoki, Takehiro Tokuda, Hannu Jaakkola, and Naofumi Yoshida (Eds.). IOS Press, Amsterdam, The Netherlands, The Netherlands, 77-96.

[Timonen 2012]. Mika Timonen. 2012. Categorization of Very Short Documents. In Proceedings of the Conference on Knowledge Discovery and Information Retrieval 2012 (KDIR'12), October 4-7, 2012, Barcelona, Spain. SciTePress Digital Library, 5-16.

[Tombros et al. 2002]. Anastasios Tombros, Robert Villa, and C.J. van Rijsbergen. 2002. The effectiveness of hierarchical clustering in information retrieval. In Information Processing & Management, Vol. 38. 559-582.

[Turney 2000]. Peter D. Turney. 2000. Learning Algorithms for Keyphrase Extraction. In Information Retrieval. Vol. 2, Article 4 (May 2000), Springer, Netherlands, 303-336. DOI: http://dx.doi.org/10.1023/A:1009976227802.

[Turney 2003]. Peter D. Turney. 2003. Coherent keyphrase extraction via web mining. In Proceedings of the 18th international joint conference on Artificial intelligence (IJCAI'03), August 9-15, 2003, Acapulco, Mexico. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 434-439.

[Vaithyanathan and Dom 1999]. Shivakumar Vaithyanathan and Byron Dom. 1999. Model Selection in Unsupervised Learning with Applications To Document Clustering. In Proceedings of the 16th International Conference on Machine Learning (ICML '99), June 27-30, 1999, Bled, Slovenia. Ivan Bratko and Saso Dzeroski (Eds.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 433-443.

[VLIB 2015]. The WWW Virtual Library: http://vlib.org/ (Last Accessed February 28, 2015).

[Voorhees 1994]. Ellen M. Voorhees. 1994. Query expansion using lexical-semantic relations. In Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval. (SIGIR '94), July 3-6, 1994, Dublin, Ireland. W. Bruce Croft and C. J. van Rijsbergen (Eds.). Springer-Verlag New York, Inc., New York, NY, USA, 61-69. DOI: http://dl.acm.org/citation.cfm?id=188490.188508.

[Wai-chiu and Fu 2000]. Wong Wai-chiu, and AdaWai-che Fu. 2000. Incremental Document Clustering for Web Page Classification. In Proceedings of the International Conference on Information Society in the 21st Century: Emerging Technologies and New Challenges (IS2000), November 5-8 Nov. 2000, Fukushima, Japan. Springer Japan. 101-110. DOI: http://dx.doi.org/10.1007/978-4-431-66979-1_10.

[Wan and Xiao 2008]. Xiaojun Wan and Jianguo Xiao. 2008. Single document keyphrase extraction using neighborhood knowledge. In Proceedings of the 23rd national conference on Artificial intelligence (AAAI'08), July 13-17, 2008, Chicago, USA. Anthony Cohn (Ed.), Vol. 2. AAAI Press, Chicago, Illinois, USA, 855-860. DOI: http://dl.acm.org/citation.cfm?id=1620163.1620205.

[WebCrawler 2015]. WebCrawler. http://www.webcrawler.com/ (Last accessed April 9, 2015).

[Wen et al. 2001]. Ji-Rong Wen, Jian-Yun Nie, and Hong-Jiang Zhang. 2001. Clustering user queries of a search engine. In Proceedings of the 10th international conference on World Wide Web (WWW '01), May 01 - 05, 2001, Hong Kong, Hong Kong. ACM, New York, NY, USA, 162-168. DOI: http://doi.acm.org/10.1145/371920.371974.

[Wen et al. 2002]. Ji-Rong Wen, Jian-Yun Nie, and Hong-Jiang Zhang. 2002. Query clustering using user logs. In ACM Transactions on Information Systems, Vol. 20, Issue 1 (January 2002). ACM New York, NY, USA, 59-81. DOI: http://doi.acm.org/10.1145/503104.503108.

[Witten et al. 1999]. Ian H. Witten, Gordon W. Paynter, Eibe Frank, Carl Gutwin, and Craig G. Nevill-Manning. 1999. KEA: practical automatic keyphrase extraction. In Proceedings of the 4th ACM conference on Digital libraries (DL '99), August 11-14, 1999, Berkeley, California, USA. ACM, New York, NY, USA, 254-255. DOI: http://doi.acm.org/10.1145/313238.313437.

[Wu and Li 2008]. Yi-fang Brook Wu and Quanzhi Li. 2008. Document keyphrases as subject metadata: incorporating document key concepts in search results. In Information Retrieval. Vol. 11, Article 3 (June 2008). Springer, Netherlands, 229-249. DOI: http://dx.doi.org/10.1007/s10791-008-9044-1.

[WWWW 2011]. World Wide Web Worm. http://www.cs.colorado.edu/home/mcbryan/WWWW.html.

[Xu and Croft 1996]. Jinxi Xu and W. Bruce Croft. 1996. Query expansion using local and global document analysis. In Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '96), August 18-22, 1996, Zurich, Switzerland. ACM, New York, NY, USA, 4-11. DOI: http://doi.acm.org/10.1145/243199.243202.

[Xu et al. 2008]. Kaifeng Xu, Rui Li, Shenghua Bao, Dingyi Han, and Yong Yu. 2008. SEM: Mining Spatial Events from the Web. In Proceedings of the Advances in Knowledge Discovery and Data Mining. May 20-23, 2008, Osaka, Japan. Lecture Notes in Computer Science, Springer Berlin Heidelberg, Vol. 5012. Springer Berlin Heidelberg, 393-404. DOI: http://dx.doi.org/10.1007/978-3-540-68125-0_35.

[Yahoo 2015]. Yahoo!. http://yahoo.com/ (Last Accessed March 28, 2015).

[Yandex 2015]. Yandex. https://www.yandex.com/ (Last accessed April 9, 2015).

[Yang and Callan 2009]. Hui Yang and Jamie Callan. 2009. Feature selection for automatic taxonomy induction. In Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval (SIGIR '09), July 19-23, 2009, Boston, Massachusetts, USA. ACM, New York, NY, USA, 684-685. DOI: http://doi.acm.org/10.1145/1571941.1572077.

[Yangarber et al. 2000]. Roman Yangarber, Ralph Grishman, Pasi Tapanainen, and Silja Huttunen. 2000. Automatic acquisition of domain knowledge for Information Extraction. In Proceedings of the 18th conference on Computational linguistics (COLING '00), June 19-22, 1980, Saarbrücken, Germany. Vol. 2. Association for Computational Linguistics, Stroudsburg, PA, USA, 940-946. DOI: http://dx.doi.org/10.3115/992730.992782 .

[Zahera et al. 2010]. Hamada M.Zahera, Gamal F. El Hady, and Waiel.F Abd El-Wahed. 2010. Query Recommendation for Improving Search Engine Results. In Proceedings of the World Congress on Engineering and Computer Science 2010 (WCECS 2010),October 20-22, 2010, San Francisco, USA. Vol. 1, 416-419.

[Zaïane and Strilets 2002]. Osmar R. Zaïane, and Alexander Strilets. 2002. Finding Similar Queries to Satisfy Searches Based on Query Traces. 2002. In Proceedings of the Workshops on Advances in Object-Oriented Information Systems (OOIS '02), September 2, 2002, Montpellier, France. Jean-Michel Bruel and Zohra Bellahsene (Eds.). Springer-Verlag, London, UK, UK, 207-216.

[Zamir and Etzioni 1998]. Oren Zamir and Oren Etzioni. 1998. Web document clustering: a feasibility demonstration. In Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '98), August 24-28, 1998, Melbourne, Australia. ACM, New York, NY, USA, 46-54. DOI: http://dx.doi.org/10.1145/290941.290956.

[Zeng et al. 2004]. Hua-Jun Zeng, Qi-Cai He, Zheng Chen, Wei-Ying Ma, and Jinwen Ma. 2004. Learning to cluster Web search results. In Proceedings of the 27th ACM International Conference on Research and Development in Information Retrieval, July 25 - 29, 2004, Sheffield, United Kingdom. SIGIR'04, ACM, New York, NY, USA, 210–217. DOI: http://doi.acm.org/10.1145/1008992.1009030.

[Zhang and Dong 2004]. Dell Zhang and Yisheng Dong. 2004. Semantic, Hierarchical, Online Clustering of Web Search Results, In Proceedings of the 6th Asia-Pacific Web Conference (APWeb 2004), April 14 - 17, 2004, Hangzhou, China. Advanced Web Technologies and Applications, Lecture Notes in Computer Science, Vol. 3007. Springer Berlin Heidelberg, 69-78. DOI: http://dx.doi.org/10.1007/978-3-540-24655-8_8.

List of Publications by Author

Conferences

[01] Poonam Goyal and N. Mehala. 2011. Concept based query recommendation. In Proceedings of the 9th Australasian Data Mining Conference (AusDM '11), December 1-2, 2011, Ballarat, Australia, Peter Vamplew, Andrew Stranieri, Kok-Leong Ong, Peter Christen, and Paul J. Kennedy (Eds.), Vol. 121. Australian Computer Society, Inc., Darlinghurst, Australia, Australia, 69-78.

[02] Poonam Goyal, N. Mehala, and Dipen Thakkar. 2012. Incremental Model for Query Clustering. In Proceedings of the FIRE 2012 Conference (FIRE '12), December 17-19, 2012, Kolkata, India.

[03] N. Mehala, Divyansh Bhatia, and Poonam Goyal. 2015. An Approach for Search Result Topic Identification and Labeling. In Proceedings of the 2nd IKDD conference on Data Sciences (*CODS '15)*, Mar 18-21, 2015, Bangalore, India. ACM IKDD, 978-1-4503-3436-5/15/03. http://dx.doi.org/10.1145/2732587.2732613.

Journals

[01] Poonam Goyal, N. Mehala, and Ankur Bansal. 2013. A robust approach for finding conceptually related queries using feature selection and tripartite graph structure. In Journal of Information Science (JIS), Vol. 39, Article 5 (October 2013). SCI, SAGE, 575-592. DOI: http://dx.doi.org/10.1177/0165551513477819.

[02] Poonam Goyal, N. Mehala and Navneet Goyal. 2014. Incremental Models for Query Clustering and Query-Context Aware Document Clustering. In Journal of Knowledge and Web Intelligence (IJKWI). Inderscience. (Revised)

[03] Poonam Goyal, N. Mehala and Divyansh Bhatia. 2015. Topical Document clustering: Hard to Soft Two-stage Post Processing Technique. In Journal of Information Science (JIS). SCI, SAGE. (Communicated)

[04] Poonam Goyal, N. Mehala, Divyansh Bhatia, and Navneet Goyal. 2015. Automatic Generation of Topic Hierarchy: A Query Context-Aware Approach. In Journal of ACM Transactions on Knowledge Discovery from Data *(*ACM TKDD). ACM. (Communicated)

Biographies

Brief Biography of the Candidate

Mrs. N. Mehala is a lecturer in the department of Computer Science and Information Systems at Birla Institute of Technology and Science Pilani (BITS-Pilani), Pilani campus, India. She obtained her B.E (CSE) from Madurai Kamaraj University, Madurai in 2001, and M.E (CSE) from Anna University, Chennai, in 2004.

She is teaching at BITS-Pilani since 2006.

Her research and teaching interests include Web Mining and Search Engine Algorithms.

Contact her at mehala@pilani.bits-pilani.ac.in.

http://universe.bits-pilani.ac.in/pilani/mehala/profile

Brief Biography of the Supervisor

Professor Poonam Goyal is an Associate Professor in the department of Computer Science and Information Systems at Birla Institute of Technology and Science Pilani (BITS-Pilani), Pilani campus, India. She ontained her M.E (Software Systems) from BITS-Pilani, Pilani, India and PhD degree in Mathematics from IIT-Roorkee, Roorkee, India.

She is with BITS-Pilani since 1995.

She is associated with the Advanced Data Analytics and Parallel Technologies Lab and also with Web Intelligence and Social Computing Lab.

Her research and teaching interests include data mining, high performance computing and information retrieval.

Contact her at poonam@pilani.bits-pilani.ac.in.

http://universe.bits-pilani.ac.in/pilani/poonam/Profile