School of Information Systems

Improving the Relevance of Web Search Results by Combining Web Snippet Categorization, Clustering and Personalization

Dengya Zhu

This Thesis is presented for the Degree of
Doctor of Philosophy (By Research)
of
Curtin University

October 2010

Declaration

To the best of my knowledge and belief this thesis contains no material previously published by any other person except where due acknowledgement has been made.

This thesis contains no material which has been accepted for the award of any other degree or diploma in any university.

Signature: ……………………………………………

Date: ……………………………….

Abstract

Web search results are far from perfect due to the polysemous and synonymous characteristics of nature languages, information overload as the results of information explosion on the Web, and the flat list, “one size fits all” strategies of search engines to present search results without concentrating on user personal information needs.

Re-organizing Web search results, or Web snippets, by means of text categorization and clustering are two dominant approaches to attack the issues above. Text categorization uses a collection of labeled documents to train a classifier which can then predict labels for new unlabeled documents; while text clustering groups unlabeled documents by finding common properties shared among the documents in the same group. The issue related to categorization is human labeled training documents are very expensive to obtain and thus surprisingly scarce at the moment; while how to label the generated groups is still an open research question for text clustering. In addition, a Web snippet, returned from search engines, contains only the title of a webpage and an optional very short (less than 30 words) description of the page. The less-informative aspect of Web snippets is another challenge for both text categorization and clustering.

The primary objective of this research is to improve the relevance of Web search results and thus provide the user with a better search experience. To achieve this objective, the research combines Web snippet categorization, clustering and personalization techniques to recommend relevant results to search users. Using design research methodology, the study develops an IT artifact named RIB – Recommender Intelligent Browser. RIB categorizes Web snippets using a socially constructed Web directory such as the Open Directory Project (ODP) for which the semantic characteristics of the categories in ODP are extracted to generate a series of labeled document sets. At the same time, the Web snippets are clustered to boost the quality of the categorization. Based on search preferences in a user profile which is automatically generated by using information extracted from user personal computer with the approval of the user for information collection, the proposed search method will recommend personalized search results to users. Experimental data demonstrate that the mean average precision improvement of RIB over *Yahoo Search Web Services API* based on 25 search-terms with 1250 Web snippets is 7.84%, from 55.55% of Yahoo to 64.29% of RIB.

A novel *boostingUp* algorithm is also proposed in this research to improve the performance of text categorization by leveraging the power of text clustering and vice versa. Experimental results illustrate that *boostingUp* can marginally improve the performance of both Web snippet categorization and clustering in terms of Adjusted Rand Index and F_1. *BoostingUp* is able to produce 0.97% improvement of macro-averaged F_1 from 24.51% to 25.48% for

Naïve Bayes with combination of K-Means, 2.04% improvement of micro-averaged F_1 from 32.17% to 34.21%. On the other hand, the improvement in terms of Adjusted Rand Index of K-Means with combination of Naïve Bayes is 2.35% (from 13.17% to 15.52%), and the improvement of F_1 is 2.37% (from 21.45% to 23.82%).

The issues of lack of labeled data set that can be used for Web snippet categorization and used as benchmark document collection to evaluate text categorization/clustering algorithms is addressed by extracting semantic characteristics of ODP categories to generate a series of labeled *categoryDocument* sets. Statistical information about the generated data sets is provided as well. The generated *categoryDocuments* are used to evaluate the performance of Naïve Bayes, Adaboost, and kNN text categorization algorithms when a list of feature selection algorithms including Chi-square, Mutual Information, Information Gain, Odds Ratio, are employed to pick up 50, 80, 100, 200, 300, 500,1000, 2000, 3000, 5000, and 10000 features. Other text categorization algorithms such as SVMlight and Statistical Language Model based algorithm and feature selection algorithms such as GSS Coefficient, NGL Coefficient, and Relevancy Score are also evaluated based on a specially designed small data set. Two other proposed algorithms, R^2Cut thresholding strategy and Z-tfidf, are at the same time evaluated, and demonstrate the ability of slightly improving the performance of text categorization. Text clustering algorithms such as K-Means and Hierarchical Agglomerative Clustering are also evaluated by using the generated *categoryDocument* sets. All algorithms involved in this research were implemented in Java.

In addition, this research is the first to present the detailed information about the hierarchy of the ODP, the world's most comprehensive human-edited Web directory, by analyzing the data in two publicly accessible files under Free Use License. Although ODP is adopted as core directory services for the World's most popular search engines such Google, AOL Search, Netscape Search, Lycos, HotBot and hundreds of other; and used for a wide range of research purposes, there is no detailed hierarchical information about ODP published so far.

The research further verifies the relationship between precision improvement and relevance judgment convergent degree when the effectiveness of an information retrieval system is evaluated based on the results of human relevance judgment; and reveals that the two variables are to some extent co-related in terms of correlation coefficient.

Improving the relevance of Web searching is challenging. This research proposes to combine text categorization, clustering and personalization to provide better search experience to users. Comprehensive experimental evidence and favorable comparisons against search results of Yahoo API demonstrate the designed search objectives have been achieved.

Acknowledgements

I would like to express my gratitude to all those who gave me the possibility to complete this thesis.

I owe an immense debt of gratitude to my supervisor, Professor, Dr. Heinz Dreher, for providing me with the opportunity to conduct this research. His research knowledge, technical advices, supervision, guidance, patient, collaboration, and encouragement are essential throughout the course of my doctoral program. He provided me a significant amount of freedom to explore the topics I was interested in. I am also grateful to him and Associate Professor Dr. Vanessa Chang for providing me a part time job to support my last stage research.

Special thanks to my associate supervisor, Professor Dr. Elizabeth Chang for providing continuous encouragements, supports, advice, generous help, constructive suggestions, and assistance in all aspects during my research. It is difficult to overstate my gratitude to her. Thanks to another associate supervisor, Dr. Ashley Aitken for his reviewing of my thesis and suggestions and feedbacks provided.

I am also like to acknowledge my deep gratitude to Dr. Alan Parkinson for his detailed review, valuable feedbacks and suggestions on this thesis. I would also like to thank Dr. Robert Williams and Associate Professor, Dr. Laurie Dickie for their detailed review and feedbacks on some chapters of the thesis.

Profound thanks and gratitude to Professor Dr. Hai Zhuge from Chinese Academy of Sciences for his valuable discussions and feedbacks on my research ideas, future research directions, and possible collaboration fields. My gratitude and appreciation also presented to Professor Dr. Jiawei Han from University of Illinois, Dr. Michael Brodie, Chief Scientist of Verizon Services Operations, and Professor Dr. Karlheinz Kautz from Copenhagen Business School, Assistant Professor, Dr. Christian Gütl from Graz University of Technology, for their constructive suggestions, feedbacks and discussions on my research.

Thanks to Ms. Jo Boycott, CBS Research Student Coordinator, for her assistance and help with regarding to facilities and administration throughout the study, especial she managed to arrange two computers for me to speed up my experiments. Thanks to administration staffs, Mrs. Karen Clarke, Mrs. Muriel Bijoux, Mrs. Ruth Ernstzen, Ms. Francesca Vallini, and Ms. Karen Stoute from School of Information Systems for their assistance and advices during my candidature.

I also owe debt of gratitude to all participants as human judges to conduct the relevance judgment in this research project. Thanks to Mr. Wei Liu and Mr. Christopher Jones for their help in some early stage programming.

This research was supported by Curtin University Postgraduate Scholarship and DEBII top-up scholarship. Thanks to Mrs. Delia Giggins and Professor Elizabeth Chang for application and management advice.

Lastly and most importantly, I wish to express my deep and sincere gratitude to my wife Wei Chen. During the long and sometime tough period of my study, she supported our family by diligent working and optimistic attitude towards life. I might have not finished my study without her love and devotion. Thanks also to my son Rongduan Zhu for his understanding of me and his lust for knowledge which in return encouraging me. I would also like my parents to receive my deepest gratitude for their everlasting support, encouragement and understanding. Thanks to my sister, brother and all members in their families for their help, support, and encouragement during my study journey.

Table of Contents

List of Figures

List of Tables

Publications Related to This Study

1) Zhu, D & Dreher, H 2010, 'Exploring semantic characteristics of socially constructed knowledge repository to optimize Web search', in HO Nigro & SEG Cisaro (eds), *Ontologies Driven Web Mining: Concepts and Techniques*, IGI Global, Hershey. (Accepted)
2) Zhu D, Dreher H Characteristics and Uses of Labeled Datasets – ODP Case Study. Proceedings of the sixth International Conference on Semantics, Knowledge, and Grids; 2010; Ningbo, China: IEEE Computer Society; November 1-3, 2010. To appear.
3) Wang H, Zhu D. Information Ecological Imbalance and Information-flow Intervention. Library and Information Service. 2010; 54(8):85-88.
4) Zhu D. Improving the Relevance of Search Results: Search-term Disambiguation and Ontological Filtering. Saarbrucken: VDM Verlag; 2009; p. 252.
5) Zhu D, Dreher H. Discovering Semantic Aspects of Socially Constructed Knowledge Hierarchy to Boost the Relevance of Web Searching. Journal of Universal Computer Science. 2009; 15(8):1685-1710.
6) Zhu D, Dreher H Chang E, Damiani E, Hussain FK, editors. A Gyroidal Pyramid Searching Model in Digital Ecosystems. Proceedings of the Third IEEE Digital Ecosystems and Technologies Conference; 2009; Istanbul, Turkey: IEEE; May 31st - June 3rd, 2009. p. 365-370.
7) Zhu D, Dreher H. Improving Web Search by Categorization, Clustering, and Personalization. In: Tang C, editor. LNAI 5139, Advanced Data Mining and Applications, The Fourth International Conference ADMA. Berlin, Heidelberg: Springer Verlag; 2008. p. 659-666.
8) Zhu D, Dreher H Personalized Information Retrieval in Digital Ecosystems. Proceedings of the Second IEEE Digital Ecosystems and Technologies Conference; 2008; Phitsanulok, Thailand: IEEE; February 26-29, 2008. p. 580-585.
9) Zhu D, Dreher H IR Issues for Digital Ecosystems Users. Proceedings of the Second IEEE Digital Ecosystems and Technologies Conference; 2008; Phitsanulok, Thailand: IEEE; February 26-29, 2008. p. 586-591.
10) Zhu D Using the Open Directory Project for Web Information Retrieval: A Survey. Proceedings of the 2008 Curtin Business School Doctoral Colloquium; 2008; Perth, Australia: Curtin University of Technology; October 2-3, 2008. p. 1-10.
11) Zhu D RIB: A Personalized Ontology-based Categorization/Clustering Approach to Improve the Relevance of Web Search Results. Proceedings of Curtin Business School Doctoral Colloquium; 2007; Perth, WA Australia: Curtin University of Technology; August 30-31, 2007.

12) Zhu D, Dreher H Determining and Satisfying Search Users Real Needs via Socially Constructed Search Concept Classification. Proceedings of the Inaugural IEEE Digital Ecosystems and Technologies Conference; 2007; Cairns, Australia: IEEE; February 21-23, 2007. p. 404-409.
13) Zhu D, Dreher H An Integrating Text Retrieval Framework for Digital Ecosystems Paradigm. Proceedings of the Inaugural IEEE Digital Ecosystems and Technologies Conference; 2007; Cairns, Australia: IEEE February 21-23, 2007. p. 367-372.

List of Denotations

C	Set of categories (labels)
D	Set of documents, or Web snippets, or distance between two vectors
T	Set of terms
v or V	Number of terms, Or features in D
n or N	Number of documents in D
|C|, m or M	Number of categories in C
k or K	Number of clusters, Or neighbours in K-Means or kNN
$\mathbf{d}_j$	Document vector = $\{t_{1j}, t_{2j}, \ldots, t_{vj}\}$, or $\{w_{1j}, w_{2j}, \ldots, w_{vj}\}$
t_{ij}	The frequency of i-th term/feature in j-th document
w_{ij}	Weight of i-th term in j-th document
x_i	The frequency of i-th element/term/feature in an instance/document
$E(X)$, $\mu(X)$ or μ	The mean value of variable $X = \{x_1, x_2, \ldots, x_n\}$
$\sigma(X)$, or σ	The standard deviation of variable $X= \{x_1, x_2, \ldots, x_n\}$
Φ	A true target function which decides how a document is categorized
$\hat{\Phi}$	An approximate function to Φ

1 Motivation and Problem Statement

This chapter introduces issues to be addressed, research objectives, and strategies to approach the research goals. Section 1.1 discusses several issues and challenges related to the field of Web information retrieval; Section 1.2 presents research questions and objectives that will be the focuses of some of the following chapters; significance and contributions of the study are presented in Section 1.3, how the thesis is organized is introduced in Section 1.4; and Section 1.5 is the summary of the chapter.

1.1 Challenges and Issues in Web Information Retrieval

Due to the exponential increasing of the information available on the World Wide Web, and the polysemous and synonymous characteristics of natural languages, Web information retrieval is facing challenge of retrieving as much relevant and only relevant information as possible in response to a query that is usually a single term or short phrase preferred by users. Numerous efforts have been made to improve Web user experiences. However, providing effective and efficient Web information searching demanded by users are far from satisfactory; despite that sophisticated algorithms have been developed and employed by search engines that intend to present relevant information to users. Smyth (2007) points out that approximately 50% of Web search sessions fail, and only 14% of the time the target results are included in the top ten search hits.

The low quality of Web search results (Gauch, Chaffee, and Pretschner 2003; Glover et al. 2001; Pierralos et al. 2003; Smyth 2007) in terms of *recall* and *precision*[1] stems from 1) the synonymous and polysemous characteristics of natural languages (Deerwester et al. 1990); 2) information overload on the Web (Cohn and Herring 2005; Montebello 1998; Zhu 2007); 3) the imperfection of the information retrieval models so far developed (Limbu, Connor, and MacDonell 2005); and 4) insufficient consideration of personal search interests and preferences (Chirita et al. 2005; Gauch, Chaffee, and Pretschner 2003; Godoy and Amandi 2006; Pitkow et al. 2002; Micarelli et al. 2007; Gauch et al. 2007; Castellano et al. 2009; Smyth 2007; Stamou and Ntoulas 2009; Ma, Pant, and Sheng 2007). These are the four main issues and challenges facing Web information retrieval.

Numerous approaches have been proposed to address the issues, including Web search results re-organization. However, re-organization of the gigantic volume of Web information

[1] See glossary for the explanation/definition of this and all others special terms

to reveal its semantic structure at different levels of detail is a challenge for the reason that semantic analysis and large-scale text document process are two demanding tasks to deal with. Text categorization (Chakrabarti 2003; Mitchell 1997; Sebastiani 2002a; Han and Kamber 2006; Qi and Davison 2009) and clustering (Chakrabarti 2003; Húsek et al. 2007; Jacsó 2007a, 2007b; Jain, Murty, and Flynn 1999; Han and Kamber 2006; Carpineto and Stanisław. 2009; Jain 2009; Kurland and Lee 2009) are predominant approaches used to address the problem of large amounts of information and the polysemous characteristics of natural languages. Text categorization, which is traditionally classified as supervised learning (Qi and Davison 2009), is the automatic assigning of predefined categories to free documents (Yang 1999); while document clustering, or traditionally classified as unsupervised learning (Jain 2009; Carpineto and Stanisław. 2009; Manning, Raghavan, and Schütze 2008), tries to discover groups in a document collection such that similar documents are grouped together (Rijsbergen 1979). Ontology, "a formal, explicit specification of a conceptualisation" (Gruber 1993, p.908), is frequently adopted as predefined categories for text categorization purpose. In fact, ontology is a description of the real world from a certain point of view by employing a special vocabulary to describe the entities, classes, properties, and functions related to that perspective (Fonseca et al. 2002).

For text categorization, the main issue is that it is expensive to obtain sufficient human labeled training data (Chakrabarti 2003; Davidov, Gabrilovich, and Markovitch 2004; Lewis et al. 2004), and consequently the readily labeled data sets are surprisingly scare (Davidov, Gabrilovich, and Markovitch 2004). Further, labeling a document involves relevance judgments by human experts, whereas the objectiveness of relevance judgment per se is an arguable topic (Hjørland 2010; Saracevic 2007b, 2007a; Borlund 2003; Mizzaro 1998, 1997).

The main challenge for clustering algorithms is the automatically formed cluster hierarchy often mismatches the human mental model (Zhu 2007; Zhu and Dreher 2008b). In addition, when only Web snippets[2], which are not as informative as full-length text documents, are available, the developed algorithms for text categorization/clustering have not been verified sufficiently. Lack of 'informativeness' also makes the relevance judgment of these Web snippets become difficult; while relevance judgment is the core of information retrieval (Mizzaro 1997; Saracevic 2007a).

Using unlabeled data to improve the accuracy of prediction of supervised algorithms is the main concern of semi-supervised learning with the assumption that data points are drawn independently and identically distributed from a common distribution (Chapelle, Scholkopf,

[2] A piece of Web snippet contains only the title of a Web page and an optional very short (less than 30 words) description of the page.

and Zien 2006). Nevertheless, can labeled data be leveraged to improve the performance of unsupervised learning which intends to discover the patterns among the unlabeled data seems not involved according to the question to be answered by semi-supervised learning (Chapelle, Scholkopf, and Zien 2006). Another issue is the assumption that labeled and unlabeled data are identically distributed from the same distribution is in some case not valid. For example, in Web search categorization and personalization, unlabeled data are the Web snippets, and the labeled data items which are used to train a classifier are the meta data of Open Directory Project[3] (Gauch, Chaffee, and Pretschner 2003; Dumais and Chen 2000).

Personalization is regarded as a promising approach to improve the relevance of Web search results because it concerns not only retrieval based on literally relevant information, but also a user's information consumption patterns, search history, searching strategies, and applications used (Pitkow et al. 2002). Google and other major search engines are providing personalized Web search services with the approval of users to collect personalization information or simple based past search information linked to a cookie (http://www.google.com/support/accounts). However, there are two main issues for personalized searching: concept drift (Pitkow et al. 2002; Tsymbal 2004; Webb, Pazzani, and Billsus 2001) and privacy protection (Kobsa and Schreck 2003; Shen, Tan, and Zhai 2007).

In brief, the main challenges for Web retrieval are: synonymy and polysemy of natural languages; information overload; the imperfection of information retrieval models; and the insufficient consideration of personalization.

1.2 Research Questions and Objectives

To address the above issues, *RIB* – Recommender Intelligent Browser is proposed (Chapter 3). The main purpose of *RIB* is to combine text categorization and clustering techniques to address synonymy, polysemy, and information overload problems by re-ranking, hierarchically organizing, and ontologically filtering Web snippets; to personalize Web search result by means of building a user profile based on a ontology - "a shared taxonomy of entities" (Smith 2004, 158) - created from a Web directory (such as the ODP). Finally, *RIB* will recommend to user re-ranked relevant results according to the created user profile.

To achieve this goal involves answering the following questions.

> **Q1** *How does the use of Web snippet categorization and personalization enhance the relevance of Web search results as measured by user feedback if it does at all?*

[3] Open Directory Project, also referred to as DMOZ, is the most comprehensive human-edited Web directory (www.dmoz.org). Refer to Chapter 4 for further introduction.

Q2 *To what degree does the combination of categorization and clustering improve the performance (in terms of Adjusted Rand Index and F_1[4]) of Web search results categorization and clustering?*

Another concern of this research is to explore the hierarchical information of ODP, the world's most comprehensive human-edited Web directory, and to generate a series of labeled document sets to enrich the existing labeled benchmark document collection for the purpose of evaluation of text categorization and clustering algorithms, and of search or Web content categorization (Sahami et al. 2004). Consequently, the following three questions are raised:

Q3 *How are the categories in Open Directory Project organized?*

Q4 *Can the semantic characteristics of ODP categories be utilized to generate a series of labeled document sets to enrich the existing benchmark document collection?*

Q5 *Can the generated labeled data set be used to train a text classifier to categorize Web search results?*

By answering the question **Q3**, **Q4** and **Q5**, the research is able to provide detailed information about the hierarchical information of ODP which has been applied by a list of researchers (Zhu 2008a). The generated labeled documents, the ODP *categoryDocument* as elaborated in Chapter 4, will not only enrich the existing benchmark document collection to estimate the performance of text categorization and clustering algorithms (Chapter 5), but also be applied to train a text classifier to re-organize Web search results (Chapter 7).

The effort to answer question **Q2** in this study leads to the suggestion of a *boostingUp* algorithm (Chapter 3) that aims to improve the performance of text categorization by leveraging the ability of clustering; and at the same time to increase the performance of text clustering by using external knowledge via text categorization.

With the answering of question **Q1**, the study can estimate whether RIB is able to provide better user search experiences. Consequently, the main objectives of this research can be briefly described as:

RO1 *To provide better user search experience by combining text categorization, clustering and personalization;*

RO2 *To evaluate the performance of the proposed* boostingUp *algorithm;*

RO3 *To generate a series of labeled data sets to evaluate the performance of text categorization and clustering algorithms;*

RO4 *To provide hierarchical information about ODP Web directory.*

[4] Refer to Chapter 2 Section 2.2.6 and Chapter 6 Section 6.1.4 for the detailed definition of the two criteria.

1.3 Significance and Contributions of the Research

1.3.1 Contributions

The contributions of this research include:

1) A novel search browser – RIB – that combines text categorization, clustering and personalization to improve the relevance of Web searching (Chapter 3).
2) A *boostingUp* algorithm which leverages the abilities of categorization and clustering to improve not only the performance of categorization, but also the performance of text clustering (Chapter 3 and Chapter 6).
3) The first research presents the detailed hierarchical analysis of the world's most comprehensive human-edited Web directory – Open Directory Project. For example, the number of categories at the different levels of the ODP taxonomy tree, how deep is the taxonomy tree for the different top-level ODP categories (Chapter 4).
4) Using *cut&combine* strategy to generate a series of labeled document sets which not only enrich the existing benchmark document collection for the purpose of evaluating text categorization and clustering algorithms, but also can be used to train a text classifier to re-organize Web search results into the human-edited Web directory (Chapter 4, Chapter 5).
5) An R^2*Cut* thresholding strategy and a *Z-tfidf* term-weighting strategy are proposed to improve the performance of text categorization (Chapter 5).
6) Further verified the relationship between the precision improvement and relevance *Judgment Convergence Degree* which are proposed at an earlier stage of this research (Zhu 2007; Zhu and Dreher 2010) (Chapter 7).
7) Several concepts introduced in the process of the generation of ODP *categoryDocuments* such as empty ODP category and *semantic-granularity* of an ODP *categoryDocument* (Chapter 4).

1.3.2 Significance

1) The knowledge of effectiveness of text categorization and feature selection algorithms when applied to Web snippet will be added by accomplishing **RO1** and **RO3** (Chapter 5, Chapter 7);
2) A new approach intends to enhance the quality of Web snippet categorization will be evaluated by accomplishing **RO2** (Chapter 6). The approach first estimates the inter-similarities between the Web snippets and the categories of a socially constructed ontology such as the ODP; and then estimates the intra-similarities among the Web

snippets to form several clusters. Combining the formed clusters with the categorized results is expected to improve the quality of categorization (Chapter 3);

3) Recommender Intelligence Browser (RIB), a novel Web information retrieval paradigm aims at recommending refined results to users based on automatically learned user profiles or users own selection of interest topics to ontologically filtering search results, is developed and its performance is expected to at least comparable with the results of *Yahoo! Search Web Services API* by achieving **RO1**.
4) Detail information about the ODP Web directory will help other researchers and practitioners to have a well understanding of the hierarchy of the ODP taxonomy tree when it is employed for the purposes of personalization, categorization, cluster labeling, a basis for evaluation, feature generation and a list of others.

1.4 Organization of the Thesis

The thesis is organized into eight chapters followed by seven appendices and references.

After the introductory Chapter 1, Chapter 2 reviews relevant literature with regards to text categorization, clustering, personalization, and research methodology with the intention to provide foundations about the research fields, state of the arts research progress in the fields, and critical reviews on the literatures. Details of the algorithms discussed in this chapter are purposely separated and presented in Appendix 5 to make sure the chapter focuses on research ideas rather than to be entangled in algorithm details.

Chapter 3 discusses the assumptions made in the research and defines research objectives formally. The core of the chapter is the introduction of the framework of RIB, the mechanism of the proposed *boostingUp* algorithm, and research method in this s.

Chapter 4 presents detail information about the ODP taxonomy tree, and how to extract semantic characteristics of the ODP categories to generate a series of labeled *categoryDocument* sets by using *cut&combine* strategy. This chapter achieves research objective four – **RO4** as discussed in Chapter 1.

Chapter 5 concentrates on the accomplishment of research objective three – **RO3** by carrying out various experiments to evaluate the performance of text categorization algorithms when the generated labeled ODP *categoryDocuments* are utilized as benchmark data sets. The proposed R^{2}Cut and Z-tfidf algorithms are also evaluated in this chapter.

Research Objective two – **RO2** is achieved in Chapter 6 which firstly introduces how the novel *boostingUp* algorithm is implemented, and then evaluates the performance of the algorithm which combines Naïve Bayes text classifier and K-Means clustering algorithm with different number of features are selected by Information Gain, Odds Ratio, and

Relevancy Score feature selection algorithms. Two clustering evaluation criteria, Rand Index and Adjusted Rand Index which are used as the category/clustering evaluation metrics are also introduced and implemented in this chapter.

Chapter 7 evaluates the performance of RIB by comparing the results directly from *Yahoo! Search Web Services API* and the recommended personalized results of RIB based on automatically created user profile, and based on allowing user to choose interest topics as well. 25 human judges are employed to conduct relevance judgments which are summarized and taken as the standard of the relevance decisions against 1250 Web search results. The chapter is to achieve research objective one – **RO1**.

Finally, Chapter 8 draws conclusions according to the experimental results presented in the previous chapters and future research directions are outlined.

Figure 1-1 illustrates the structure of the thesis and the research objectives of the research.

1.5 Summary

This chapter introduced the issues and challenges facing Web information retrieval, defined research questions and concisely presented the approaches to address the issues. After the contributions of the research were discussed, the significances of the study were presented as well. The structure of the thesis was provided with a diagram that shows how the research

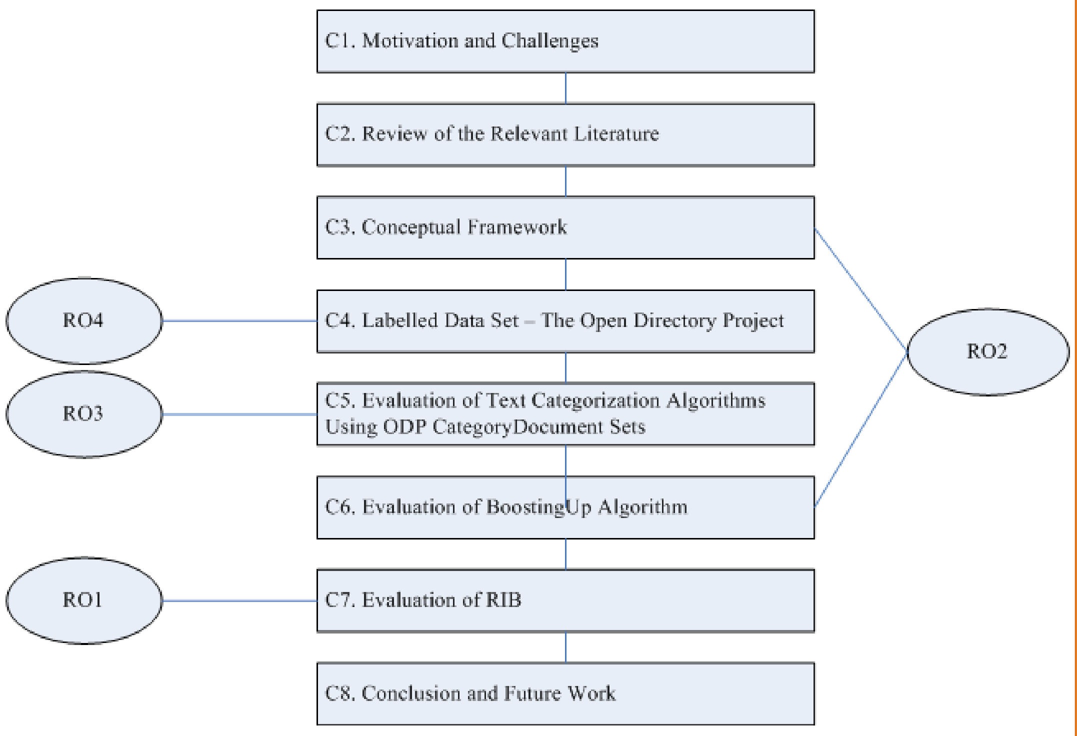

C1. Motivation and Challenges
C2. Review of the Relevant Literature
C3. Conceptual Framework
RO4
C4. Labelled Data Set – The Open Directory Project
RO2
RO3
C5. Evaluation of Text Categorization Algorithms Using ODP CategoryDocument Sets
C6. Evaluation of BoostingUp Algorithm
RO1
C7. Evaluation of RIB
C8. Conclusion and Future Work

Figure 1-1 Structure of the thesis and research objectives

objectives are related to the different chapters. In the next chapter, literatures will be reviewed and foundations are to be introduced within the scope of the research.

2 Review of the Relevant Literature

This chapter concentrates on reviewing relevant literature on text categorization, clustering, feature selection, personalization, and research methodology applied in this research. Section 2.1 first provides the foundations of popular text categorization algorithms and performance evaluation measures, and then reviews works that employ the algorithms to categorize Web search results. Section 2.2 introduces text clustering algorithms, how to use external knowledge to improve text clustering, performance measurements of clustering algorithms, and research that groups Web results into clusters. This is followed by Section 2.3 that suggests how the high dimension of term space can be reduced by feature selection algorithms. Section 2.4 reviews Web search results personalization; and lastly, Section 2.5 summarizes the chapter.

Detailed introduction of text categorization, clustering and feature selection algorithms involved in this chapter is presented in Appendix 5. Implementation details of the algorithms are provided in Chapter 5 and Chapter 6.

2.1 Categorization

Text categorization, also referred to as classification and traditionally classified as supervised learning (Chakrabarti 2003; Cios et al. 2007; Han and Kamber 2006; Larose 2005; Liu 2007; Manning, Raghavan, and Schütze 2008; Mitchell 1997; Sebastiani 2002b, 2005; Qi and Davison 2009), is the automatic assigning of predefined categories to free documents (Yang 1999). Voluminous research has been conducted on text categorization, including techniques based on probability and naïve Bayes theorem (Lewis 1998; Lewis and Gale 1994; Li and Jain 1998) , support vector machines (Joachims 1998), Rocchio's method (Hull 1994; Joachims 1997; Schapire, Singer, and Singhal 1998), decision tree (Mitchell 1997), decision rule (Apté, Damerau, and Weiss 1994; Cohen and Singer 1999), regression methods (Schütze, Hull, and Pedersen 1995; Yang, Chute, and Clinic July 1994; Yang and Liu 1999), on-line (incremental) methods (Dagan, Karov, and Roth 1997; Ng, Goh, and Low 1997; Schütze, Hull, and Pedersen 1995; Wiener, Pedersen, and Weigend 1995b), neural networks (Dagan, Karov, and Roth 1997; Lam and Lee 1999; Ng, Goh, and Low 1997; Ruiz and Srinivasan 2002; Weigend and Pedersen Oct 1999), and boosting (Schapire and Singer 2000; Schapire, Singer, and Singhal 1998), k-Nearest Neighbours (Mitchell 1997).

A supervised machine learning process suggested by Kotsiantis (2007) is presented in Figure 2-1. The figure demonstrates that selection of an appropriate algorithm and parameter tuning is essential for a specific application domain because for the same data sets, different algorithms and different settings of parameters produce different categorization results. However, this process is from the perspective of machine learning applications. From the point of view of machine learning research, the process suggested by Larose (2005) as demonstrated in Figure 2-2 pays more attentions to the data sets applied to evaluate a supervised learning algorithm. The principle to be keep in mind is that test and validation data set should not be a part of training data set.

It would be hard in this project to enumerate and discuss in detail all the algorithms. According to the popularity, effectiveness, availability and ease of implementation, Naïve Bayes classification (NB), k-Nearest Neighbours (kNN), support vector machines (SVMs), Adaboost and statistical language model are selected and discussed in the following sections. More detailed information about the algorithms is presented in Appendix 5. Before the discussion of the algorithms, bias-variance trade-off is introduced.

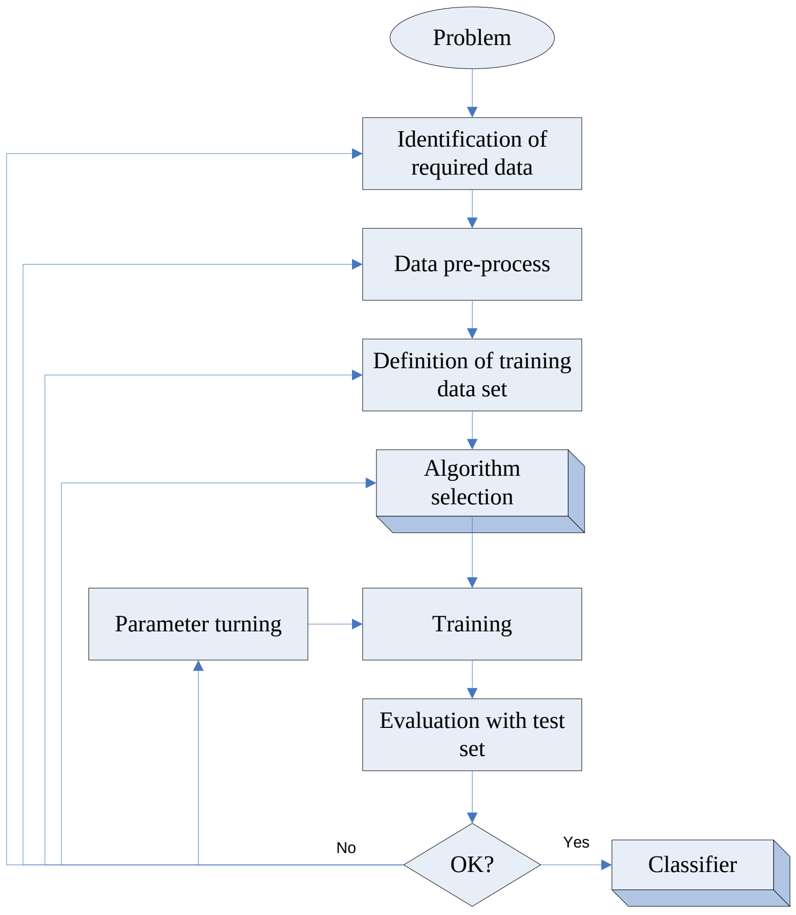


Figure 2-1 The process of supervised machine learning (Kotsiantis 2007).

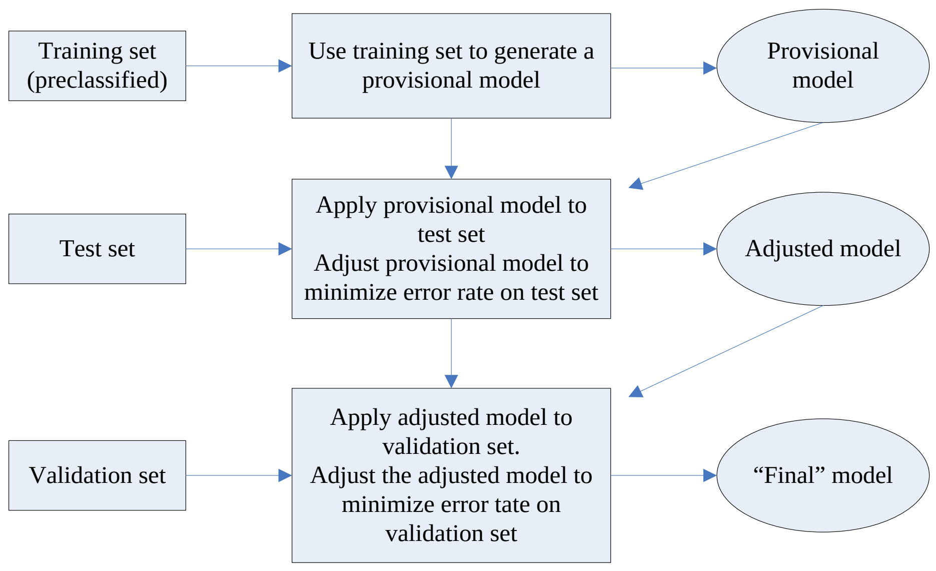

Training set (preclassified)
Use training set to generate a provisional model
Provisional model
Test set
Apply provisional model to test set
Adjust provisional model to minimize error rate on test set
Adjusted model
Validation set
Apply adjusted model to validation set.
Adjust the adjusted model to minimize error tate on validation set
"Final" model

Figure 2-2 The process of supervised machine learning (Larose 2005).

2.1.1.1 Bias-Variance Trade-off

Figure 2-2 is a supervised modeling process proposed by Larose (2005). In this process, test set is divided into two parts, one for test and one for validation. The provisional model is adjusted to minimize error rate on the test data, and the adjusted model, over-fitted by the test data set, is to be tested against the validation set to make sure optimal results, in terms of minimum error rate, are obtained. However, decreasing error rate is usually at the cost of increasing the complexity of the supervised model as well as the variance[5] (Geman, Bienenstock, and Doursat 1992), as shown in Figure 2-3 (Nelles 2001; Yu et al. 2006). Hand (2006) indicates that simple categorization algorithms perform almost as good as complex models in real applications. In this research, performance of the text categorization algorithms in terms of F_1 (refer to Section 2.1.1.7) is to be evaluated in Chapter 5.

2.1.1.2 Naïve Bayes Classification

As pointed out by Lewis (1998), Naïve Bayes classifier and its variations has long been a core technique in text categorization and information retrieval. Suppose the vector random

[5] Bias of an estimator is the difference between the estimator's expected value and the true value of the parameter being estimated. The variance of a random variable or distribution is the expectation, or mean, of the squared deviation of that variable from its expected value or mean. For detailed description, refer to Geman, Bienenstock, and Doursat (1992), and Yu et al. (2006)

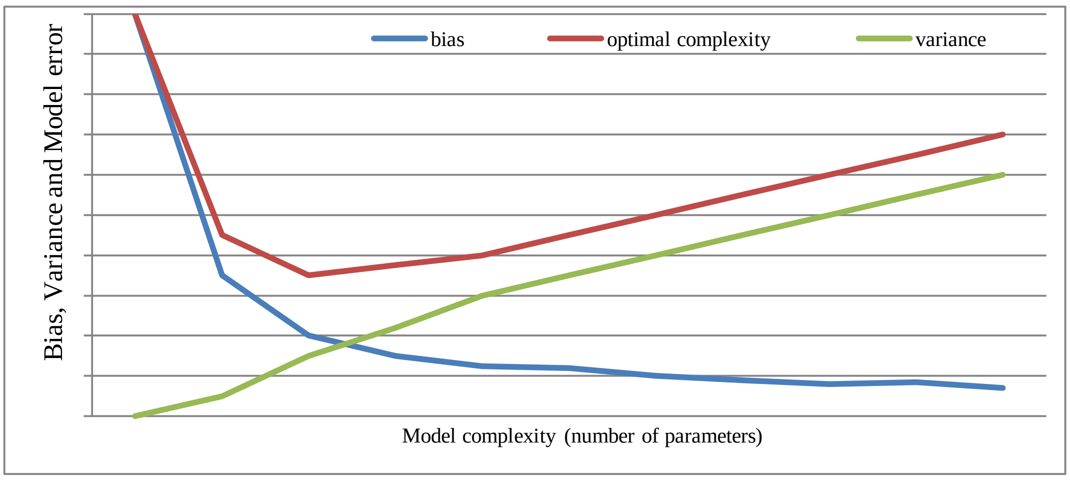


Figure 2-3 Relationship among bias, variance and complexity (Nelles 2001). (p. 165)

variable D = {$\mathbf{d}_j$ | $\mathbf{d}_j \in$ D, j = 1, … N} where N is the total number of documents in the event space D, and a document $\mathbf{d}_j$ is represented as a feature vector $\mathbf{d}_j$ = (t_1, t_2, … t_v) where v is total number of features in D; the random variable C = {c_i | $c_i \in$ C, i = 1, … M} where M is the total number of categories in C. If one document falls into exactly one of the category in C, the probability of a given document $\mathbf{d}_j$ in category c_i can be expressed by applying Bayes formula as

$$P(C=c_i \mid D=\mathbf{d}_j) = P(C=c_i) \times \frac{P(D=\mathbf{d}_j \mid C=c_i)}{P(D=\mathbf{d}_j)}$$

where P(C = c_i | D = $\mathbf{d}_j$) is the conditional probability of category C = c_i given document $\mathbf{d}_j$, P(D = $\mathbf{d}_j$ | C = c_i) is the conditional probability of document $\mathbf{d}_j$ appears in given category C = c_i, P(D = $\mathbf{d}_j$) is the prior probability of observing document D = $\mathbf{d}_j$, and P(C = c_i) is the prior probability of document occurring in C = c_i.

Naïve Bayes, as simple as it is, has been described as “remarkably successful” (Lewis 1998), or “surprisingly well”, “often more effective than sophisticated rules” (Hand and Yu 2001). As one of the most widely used algorithm, it is implemented and evaluated in this research.

2.1.1.3 kNN Classifier

k-Nearest Neighbours (kNN) is another popular machine learning algorithm. kNN is also referred to as lazy learning (Mitchell 1997) because it stores all the training instances, and classifies an unlabeled instance by simply comparing the similarities between the instance and the stored, labeled instances (Larose 2005).

Three issues related to kNN need to be addressed. The first one is how to measure the distance, or similarity between two instances **x** and **y**. Table 2-1 summarizes some distance estimation approaches for two instance x and y. Cosine similarity (Salton and Buckley 1988) has been verified is one of the most effective similarity measurements in the field of

Table 2-1 Approaches to measure distance between two instance **x** and **y** (or set X and Y)

Distance	**Measure**
Minkowsky	$D(\mathbf{x},\mathbf{y}) = \left(\sum_{i=1}^{T} \lvert x_i - y_i \rvert^r\right)^{1/r}$
Manhattan	$D(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{T} \lvert x_i - y_i \rvert$
Euclidean	$D(\mathbf{x},\mathbf{y}) = \left(\sum_{i=1}^{T} \lvert x_i - y_i \rvert^2\right)^{1/2}$
Chebychev	$D(\mathbf{x},\mathbf{y}) = \max_{i=1}^{T} \lvert x_i - y_i \rvert$
Cosine similarity	$D(\mathbf{x},\mathbf{y}) = \frac{\mathbf{x} \bullet \mathbf{y}}{\mathbf{x} \times \mathbf{y}} = \frac{\sum_{k=1}^{T} x_k y_k}{\sqrt{\sum_{k=1}^{T} x_k} \times \sqrt{\sum_{k=1}^{T} y_k}}$
Camberra	$D(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{T} \frac{\lvert x_i - y_i \rvert}{\lvert x_i + y_i \rvert}$
Jaccard	$D(X,Y) = \frac{(X \cup Y) - (X \cap Y)}{(X \cup Y)}$
Pearson's Correlation coefficient	$D(\mathbf{x},\mathbf{y}) = \frac{(x_i - \bar{x}_i) \bullet (x_j - \bar{x}_j)}{\lVert x_i - \bar{x}_i \rVert \cdot \lVert x_j - \bar{x}_j \rVert}$
Normalized Google Distance	$NGD(\mathbf{x},\mathbf{y}) = \frac{\max\{\log f(\mathbf{x}), \log f(\mathbf{y})\} - \log f(\mathbf{x},\mathbf{y})}{\log N - \min\{\log f(\mathbf{x}), \log f(\mathbf{y})\}}$ N is number of pages indexed by Google, f(**x**), f(**y**) are the number of Google hits for search terms **x** and **y**, f(**x,y**) is the number of Google hits for tuple of search term **x** and **y** (Cilibrasi and Vitányi 2007).

information retrieval (Baeza-Yates and Ribeiro-Neto 1999) and thus will be implemented in this research; another widely used measure is the Minkowski distance (Jain, Murty, and Flynn 1999) which is also implemented when the order *r* in Table 2-1 equals two.

The second issue is how to decide k, the number of neighbours. One approach is to exhaustively try different k and then choose the one that produces the best results, more practically, to try some possible k with randomly selected training sets to determine which k can minimize the classification error rate (Larose 2005). Since the category distribution in training data set is not even, a fixed value of k is not appropriate. Li, Lu, and Yu (2004) suggest an adaptive kNN to adjust k based on the size of samples for different categories. How to effectively and efficiency choose an appropriate k is still an open research question.

For kNN, the third issue is how to match a test instance to the most appropriate class[6]. Majority voting is an intuitive and simple un-weighted approach which selects the majority label from the k neighbours (Zhu 2009). Weighted voting schema estimate the influence of an instance in the training document set by its inverse proportion to the distance between the instance and the test document. Supposed D(**x**, **y**) is the distance between instance **x** and **y**, then the weighted voting score of D(**x**, **y**) can be (Larose 2005)

$$\frac{1}{D^2(\mathbf{x},\mathbf{y})}$$

kNN is one of the conceptually straightforward approaches to classifying text documents and can perform comparably with even SVMs (Yang and Liu 1999). Experimental results of Cardoso-Cachopo and Oliveira (2003) by using LSA - Latent Semantic Analysis (Appendix 5) to measure the similarity support Yang and Liu's results. However, the biggest obstacle to the application of this algorithm is its poor efficiency when processing large-scale training data set. Using LSA as a measure of similarity makes the situation worse because LAS also suffers from computational expensive. Reducing the high dimensionality of vector space by means of an efficiency and effective feature selection algorithm can to some extent address the issue.

2.1.1.4 Support Vector Machines

Support Vector Machines (SVMs), introduced in 1995 (Vapnik 1999) based on computational learning theory, is one of the state of the art and top performance learning algorithms in a range of applications such as pattern recognition and text categorization

[6] Assigning a single label/category to each document is referred to as hard categorization. Soft categorization allows an instance to be assigned more than one labels/classes.

(Joachims 1998; Yang and Liu 1999; Treeratpituk and Callan 2006). The basic idea of SVMs is to optimize a hyperplane for linearly separable object/patterns, and can be extended to patterns that are not linearly separable by a kernel function to transform the original data into a new space.

SVMs have been applied in a range of applications include text categorization (Joachims 1998, 1999; Dumais and Chen 2000), and experimental results demonstrate SVMs are usually the top ranked text classifier with Reuters-21450, Reuters-21578, Ohsumed corpus, RCV1, and some other collections in UCI Repository (Section 2.1.1.9) as benchmark collections (Lewis et al. 2004; Yang and Liu 1999; Yang, Zhang, and Kisiel 2003; Joachims 1998; Cardoso-Cachopo and Oliveira 2003). However, the performance of SVMs, like other machine learning algorithms, is influenced by feature selection algorithms, application domain, and a set of parameters that need to be carefully tuned.

2.1.1.5 Boosting

Boosting algorithm is a kind of ensemble learning algorithms that combine a list of learned, most probably weak classifiers to form a composite model which is hoped to boost the accuracy of classification/prediction (Han and Kamber 2006). Bagging and Boosting are two examples of ensemble learning techniques. While Bagging assigns a label to a given example by a simple majority voting result of several trained independent weak classifiers; each weak Boosting classifier, which is trained in sequence, updates the weights of training examples according to the performance of the previous weak classifier so as more attention is paid to those misclassified examples. A final learner is produced by combining the votes of the weighted individual classifiers where the weight of each classifier is estimated with respect to its accuracy (Han and Kamber 2006).

The idea of Boosting stems from the Probably Approximately Correct (PAC) learning model (Kearns and Vazirani 1994; Valiant November 1984), which is a binary “weak” learning algorithm that performs only slightly better than random guessing, say 50%, and the weak classifiers can be boosted into one with extremely high accuracy (Freund and Schapire August 1997). Schapire and Singer (1999) suggest several improvements to AdaBoost algorithm proposed by Freund and Schapire (August 1997). A comprehensive review of the boosting algorithms from its very beginning to the most recent research achievements are presented by Schapire (2003). BoosTexter (Schapire and Singer 2000) is one of the boosting algorithms which can be used for text and speech categorization tasks. AdaBoot.MH and AdaBoost.MR are two versions of BoosTexter; the former concentrates on minimizing Hamming loss function (Schapire and Singer 1999), the latter intends to train a learner that arranges the correct labels on the top of a ranked label list.

Boosting algorithm and its variants have been evaluated on very large data sets (more than 10000 training examples) such as Reuter-21578 (Lewis et al. 2004; Bloehdorn and Hotho 2004; Schapire, Singer, and Singhal 1998; Iyer et al. 2000; Sebastiani, Sperduti, and Valdambrini 2000), AP Title (Lewis and Gale 1994), DSO (Escudero, Màrquez, and Rigau 2000), OHSUMED (Yang and Liu 1999), and UseNet data (Lang 1995). Experimental results of Lewis et al. (2004) reveal that *AdaBoost.MH with real-valued predictions* (AdaBoost.MHR) outperforms all the other competitor algorithms, including Naïve Bayes (Mitchell 1997), Rocchio (Sebastiani 2002b; Joachims 1997), probabilistic TF-IDF (Joachims 1997), Sleeping-experts and RIPPER (Cohen and Singer 1999). Experimental data also demonstrate that *AdaBoost.MH with real-valued predictions and abstaining* is only marginally inferior to the best player. This leads to the suggestion that with a huge training set, boosting algorithm is an appealing alternative for text categorization problem. It should be best utilized for complex multiclass problems when large number of training data is available (Schapire and Singer 2000), at the cost of computational complexity (Schütze, Hull, and Pedersen 1995), and the assumption that there is little or no classification noise in training data set (Dietterich 2000). However, this assumption cannot always be satisfied in real world applications.

2.1.1.6 Statistical Language Modelling

Statistical Language Modelling (Liu and Croft 2003; Manning, Raghavan, and Schütze 2008; Ponte and Croft 1998; Zhai 2008, 2009; Kurland and Lee 2009; Miller, Leek, and Schwartz 1999), or simply Language Modelling, has seen a long history in the fields of speech recognition, named entity finding, optical character recognition, topic identification, and other natural language techniques since 1980 (Rosenfeld 2000; Miller, Leek, and Schwartz 1999). A language model involves estimating a probability distribution P(**d**) over all possible documents **d** (Rosenfeld 2000). Experiments carried out by researchers show that Language modelling approach is comparable to the best information retrieval techniques so far developed (Liu and Croft 2003; Zhai 2008).

Since the first language modelling approach proposed by Ponte and Croft (Ponte and Croft 1998), the model has been extended in a variety of ways, such as a list presented by Liu and Croft (2003), and Zhai (2008, 2009). However, the extended models are usually more computationally expensive than the basic query likelihood model while only merely marginally improve on the basic model. The crucial factor which influences the performance of the language modelling approach is the smoothing strategy (Liu and Croft 2003; Manning, Raghavan, and Schütze 2008; Song and Croft 1999; Zhai 2008, 2009; Zhai and Lafferty 2001).

All the application of the developed statistical language models are concentrated on the information retrieval systems and seldom are focused on text categorization, especially on Web snippet categorization.

2.1.1.7 Evaluation Measures of Text Categorization

The performance of an algorithm usually involves two aspects, effectiveness and efficiency. In the field of text categorization, as pointed out by Sebastiani (2002b), the ability of an algorithm to make a correct classification decision is more important than its efficiency. Therefore, the following widely used effectiveness measurements are introduced first.

Precision, recall, F_1, fallout, accuracy and error

Precision and recall are two measures borrowed from the field of classical information retrieval where precision is defined as the proportion of the retrieved documents that is relevant, and recall is the proportion of relevant documents that has been retrieved (Baeza-Yates and Ribeiro-Neto 1999). From the perspective of probability, precision is defined as the conditional probability that given a category c, the probability that assign the category to a test document **d** is correct. The recall is also defined as a conditional probability that if **d** ought to be assigned c, this decision is taken (Sebastiani 2002b). A contingency table (Yang 1999) for a category c is usually applied to define the two measures and others. The cells in Table 2-2 are:

TP: True Positive, the number of documents correctly assigned to c

FP: False Positive, the number of documents incorrectly assigned to c

FN: False Negative, the number of documents incorrectly rejected by c

TN: True Negative, the number of documents corrected rejected by c

With this contingency table, the measures for text categorization are defined as:

Accuracy (a) = (TP + TN) / n where n = TP + FP + FN + TN;

Error (err) = (FP + FN) / n;

Table 2-2 A contingency table for a category c

Category c		**Expert judgments**	
		Yes is correct	No is correct
Assigned Yes by c	Yes	TP	FP
Assigned No by c	No	FN	TN

Fallout (f) = FP / (FP + TN) if FP + TN > 0; otherwise undefined;

Precision (p) = TP / (TP + FP) if TP + FP > 0; otherwise undefined;

Recall (r) = TP / (TP + FN) if TP + FN > 0; otherwise undefined.

To obtain the overall precision and recall across all categories, two different averaging approaches are available: micro-averaging and micro-averaging. Micro-averaging first calculates precision and recall “locally” for each of the categories, and then averages the results of different categories; Micro-averaging on the other hand, accumulates all individual precision and recall, and then calculates the two measures by definition.

$$\textit{Macroaveraged precision and recall}$$

$$precision = \frac{\sum_{i=1}^{|C|} p_i}{|C|}, \quad recall = \frac{\sum_{i=1}^{|C|} r_i}{|C|}$$

$$\textit{Microaveraged precision and recall}$$

$$precision = \frac{TP}{TP+FP} = \frac{\sum_{i=1}^{|C|} TP_i}{\sum_{i=1}^{|C|} (TP_i + FP_i)}, \quad recall = \frac{TP}{TP+FN} = \frac{\sum_{i=1}^{|C|} TP_i}{\sum_{i=1}^{|C|} (TP_i + FN_i)}$$

where p_i and r_i are precision and recall for category c_i, |C| is the total number of categories, TP_i, FP_i, and FN_i are true positive, false positive, and false negative for category c_i.

The two averaging approaches may lead to distinct results. Micro-averaging gives equal weight to every document; and micro-averaging assign equal weight to every category.

Since increasing precision is to certain extent at the cost of sacrifice to recall, and vice versa, therefore, the performance of a text categorization algorithms is usually evaluated by F_β function (Rijsbergen 1979) which is defined as

$$F_\beta = \frac{(\beta^2 + 1) \cdot p \cdot r}{\beta^2 \cdot p + r}$$

where $\beta \in (0, +\infty)$ is a parameter which allows assigning different weights to precision and recall. If $\beta \in (0, 1)$, F_β gives more weights to precision, and if $\beta \in (1, +\infty)$, more weights are assigned to recall. When $\beta = 0$, F_β coincides with precision, when $\beta = \infty$, F_β equals recall. Specifically, when $\beta = 1$, F_β balances the weights of precision and recall by assigning them the same weight, and is referred as to F_1, which is widely employed by a list of researchers (Lewis et al. 2004; Schapire and Singer 2000; Yang 1999; Yang and Liu 1999; Chen et al. 2009; Toutanova et al. 2001; Campos and Romero 2009; Sun, Lim, and Ng 2002; AI-Mubaid and Umair 2006; Wang and Chiang 2007; Ifrim, Bakir, and Weikum 2008; Yang, Slattery, and Ghani 2002; Caropreso, Matwin, and Febastiani 2001; Li, Lu, and Yu 2004; Debole and Sebastiani 2005; Lam, Ruiz, and Srinivasan 1999).

Mean Reciprocal Rank – MRR

Mean Reciprocal Rank is proposed by Voorhees (1999) in the TREC-8 question answering track report. The definition of MRR is "an individual question received a score equal to the reciprocal of the rank at which the first correct response was returned, or 0 is none of the five responses contained a correct answer." Voorhees (1999) The final score is the mean of individual reciprocal rank scores. MRR is intuitive and easy to calculate; it is closely related to the average precision, and bounded inclusively to $\{0, 1/R \mid R \in \{1, N\}$, N is the number answers to be considered$\}$.

Supposed each document is assigned one and only one category, MRR can be taken as a measure to evaluate the performance of text categorization algorithms as well (Cardoso-Cachopo and Oliveira 2003). However, the measure is not accepted in this research for the reason that it is more suitable for question-answer systems, and only occasionally used as a measurement for text categorization.

Eleven-point average precision

As pointed by Voorhees (Voorhees 2005a), precision and recall are set-based measurements which evaluate the quality of an unordered set of search results. When ranking order of retrieved results are taken into account, as the case for search engines and most large-scale information retrieval systems, eleven-point average precision versus recall curve is often employed. For eleven recall level 0 to 1.0 with increments of 0.1, the corresponding precision can be plotted, and a precision versus recall curve can be drawn based on the 11 <recall, precision> tuples (Baeza-Yates and Ribeiro-Neto 1999). Examples of the recall-precision curves are provided by Baeza-Yates and Ribeiro-Neto (1999) and Voorhees (2005a). Experiments take this measures to evaluate text categorization algorithms include for example, Schapire and Singer (2000), Lam, Ruiz, and Srinivasan (1999), Yang and Pedersen (1997), Yang (1999). This measurement is to be utilized to evaluate the performance of RIB.

Breakeven point

Breakeven point is the value at which precision equals recall. If there is no such a point, the value that is closest to both precision and recall is selected as the breakeven point. Several researchers utilize break even to estimate text categorization algorithms, such as Joachims (1998), Cohen and Singer (1999), Lewis (1992), Ng et al. (1997), and Yang (1999).

Both breakeven point and F_1 (or F_β) capture balance between recall and precision. Since arithmetic mean (breakeven point) is no less than harmonic mean (F_1), F_1 is more strict than breakeven. Therefore, F1 is taken as the measurement in this research to evaluate the performance of text categorization algorithms.

Efficiency

It is very hard to find in literature an accurate measure of efficiency of text categorization algorithms, except for Dumais et al (1998), Joachims (1999), and Ifrim, Bakir and Wiekum (2008). The research of Yang, Zhang and Bryan (2003) is the first that provides an analysis of computational complexity of SVMs, kNN, ridge regression, least square fit and logistic regression, and then present their experimental results on these algorithms. However, they tested kNN and SVMs only on Ohsumed data collection, and large-scale experiments are needed to empirically demonstrate the efficiency of the algorithms.

2.1.1.8 Applications of Text Categorization in Web Snippets

Klas and Fuhr (Klas and Fuhr 2000) propose to group Web documents under the hierarchical structure of Yahoo! directory. They create a so-called *megadocument* for each of the categories in *Yahoo! Web Directory* by downloading all webpages indexed under the categories and then concatenating all text part of the downloaded documents belonging to a given Yahoo! Category. The terms in the *megadocuments* are weighted by tf-idf[7] technique. To classify a document, the first *n* best terms (according to their idf values) are selected as a query vector. A similarity search is conducted based on a probabilistic model, and a document is assigned to the category (represented by the *megadocument*) with the highest similarity. Experimental results demonstrate that their approach achieves relatively good results with Web documents. When applied to the Reuters collection[8], the Klas and Fuhr approach was comparable to the other classical text categorization methods.

Dumais and Chen (2000) suggest an SVMs based classifier which is trained by using the data set derived from a hierarchical structure that exists in the *Looksmart Web Directory* (http://search.looksmart.com/). Only the top two levels of the hierarchy are utilized because most search results can be disambiguated at this level. The webpages are represented by binary vectors. The reason is that good performance can be achieved when binary vectors are used in SVMs, and this representation improves efficiency as well. To reduce the feature space, words that appear in only one page are eliminated; 1000 words with the highest mutual information among each category are then selected as features. Compared with the

---

[7] tf: term frequency, idf: inverse document frequency. tf-idf is a term-weighting strategy which estimates the informativeness of a given term for a document in an information retrieval system if the document contains the term.

[8] or Routers Corpus, refer to http://www.daviddlewis.com/resources/testcollections/rcv1/, and http://about.reuters.com/researchandstandards/corpus/

flat, non-hierarchical models, Dumais and Chen's experimental results indicate small advantage is achieved in the F_1[9] accuracy score for this hierarchical model.

Han and Karypis (2000) indicate that centroid-based document classification, which uses traditional Vector Space Model to compare similarity between documents, performs very well compared with Naïve Bayes, kNN, and C4.5 (Quinlan 1993). Term distributions are also explored to improve the performance of centroid-based text categorization (Lertnattee and Theeramunkong 2004). Surprisingly, experimental results demonstrate that the performance of traditional Vector Space Model based algorithms for information retrieval is hard to beat, if parameters are well tuned (Manning, Raghavan, and Schütze 2008).

Zhu (2007), Zhu and Dreher (2007) propose to classify Web search results into a socially constructed knowledge hierarchy, such as the ODP – the largest, most comprehensive human-edited Web directory. The semantic characteristics of each category in the ODP are extracted, and category-documents are formed based on the extracted semantic characteristics of the categories. A Special Search Browser (SSB) is developed which obtains Web search results by utilizing *Yahoo! Search Web Services API*. Similarities between vectors represent Web search results and the category-documents are compared by using API provide by Lucene (Gospodnetić and Hatcher 2005); a majority voting strategy is used to assign a Web snippet to the proper category without overlapping. The experimental results based on five ambiguous search-terms demonstrate that the improvement of precision of SSB is about 23.5%. One weakness of the research is if users are unfamiliar with the hierarchy of the ODP, the special search browser lacks the ability to recommend to users which category can more likely satisfy the users' information needs. The second limitation is that the experimental results are based on only five search-terms, which is obviously not sufficient. A third weakness is while the *precision* is improved, there is a decrease in *recall*.

This research is largely based on SSB and aims at addressing the above three weaknesses. An updated version of SSB – Recommender Intelligent Browser is designed and discussed in detail in Chapter 3.

2.1.1.9 Test collections

As discussed in the beginning of this section, a labeled benchmark document collection is a necessity for text categorization because a text classifier has to be trained by using the labeled collection before it can be used to predict a label for an unlabeled document. Performance of a classifier might vary dramatically when evaluated by different document collections. The following is a list of text collections used in text categorization research.

[9] F_1 is a measurement of effectiveness of information retrieval systems based on *precision* and *recall*.

Reuters-21578 and RCV1.

One of the most widely used labeled benchmark document collection is the Reuters-21578 (Sebastiani 2002a; Yang 1999; Debole and Sebastiani 2005). Recently, Reuters released Reuters Corpus Version 1 (RCV1) that greatly expanded the earlier Reuters-21578. RCV1 intends to overcome some weaknesses, such as few documents, lack of the full document text, inconsistent or incomplete category assignments, and particular textual properties of the existing test collections. RCV1 categorizes more than 800,000 newswire stories from August 20, 1996 to August 19, 1997 under three categories: *Topics*, *Industries* and *Regions*. Lewis et al. (2004) analyse the different aspects of the RCV1 from the perspective of test collection, remove errorful data, and produced a new version, RCV1-v2, for the purpose of text categorization.

OHSUMED.

OHSUMED (http://ir.ohsu.edu/ohsumed/ohsumed.html), another very popular text set, consists of 348,566 references from 270 medical journals spanning from 1987 to 1991. Each reference is assigned no less than one MeSH categories (*Introduction to MeSH* 2007; Hersh et al. 1994).

UCI Machine Learning Repository

Created by David Aha in 1987, the repository intends to provide a collection of data sets available to machine learning community to evaluate machine learning algorithms (http://archive.ics.uci.edu/ml/). The current 187 data sets can be used for categorization, clustering, regression and other purpose. The most widely used text categorization data set, Reuters-21578 and Twenty Newsgroups are also included in this repository.

Open Directory Project.

"The Open Directory Project is the largest, most comprehensive human-edited directory of the Web. It is constructed and maintained by a vast global community of volunteer editors." (http://www.dmoz.org/about.html) A number of researchers have utilized the ODP for different purpose. However, to the best of our knowledge, how OPD is utilized by different researchers is not well studied and there is a lack of literature on summarizing the application features of the ODP. A survey paper (Zhu 2008a) on the usage of ODP as part of this research provides a comprehensive review of the usage of ODP. Table 2-3 presents a concise summarization of the applications of ODP based on the survey paper.

Although many researchers apply ODP for different purposes, there is no paper so far to provide statistical information of the most comprehensive human-edited Web directory. This research will fill this void by presenting statistical information about ODP and generating ODP *categoryDocument* sets as described in Chapter 4.

Table 2-3 A survey of the usage of ODP (Zhu 2008a)

Application	**Researchers**
Personalization	Middleton, De Roure and Shadbolt (2001), Pitkow et al (2002), Tanudjaja and Mui (2002), Haveliwala (2002), Gauch, Chaffee and Pretschner (2003), Fang. Liu, Yu & Meng (2004), Chirita et al. (2005), Ma, Pant and Sheng (2007), Zhu and Dreher (2008c)
Feature generation	Gabrilovich (2006), Santamaria, Gonzalo and Verdejo (2003)
Text categorization	Zhu (2007), Zhu and Dreher (2007), Zhu (2009), Zhu and Dreher (2009)
Hyperlink-based usage	Haveliwala (2002), Jeh and Widon (2003), Qiu and Cho (2006)
Ground truth for evaluation	Chowdhury and Soboroff (2002), Ch and Chaudhary (2006), Treeratpituk and Callan (2006)
Spam filter	Wu and Davison (2006)
Other	Maguitman et al.(2006), Qiu et al. (2007), Jianhui. Liu and Birnbaum (2007), Davidov, Gabrilovich & Markovitch (2004), Varma (2002)

2.2 Text Clustering

Text clustering, usually categorized as a kind of unsupervised learning (Carpineto and Stanisław. 2009; Chakrabarti 2003; Húsek et al. 2007; Jain, Murty, and Flynn 1999; Mirkin 2005; Rokach and Maimon 2005; Han and Kamber 2006; Manning, Raghavan, and Schütze 2008; Kurland and Lee 2009), tries to discover groups in a document set such that similar documents are grouped together (Rijsbergen 1979). Text clustering considers how to represent a document, how to measure the similarity among the documents, design/choose a clustering algorithm, evaluation and if necessary, data abstraction (Jain, Murty, and Flynn 1999). Clustering techniques are well researched and a vast collection of algorithms have been published. With the algorithm arsenal the questions then are: how to choose a similarity measure, how should domain knowledge be leveraged, and how to improve effective and efficiency are dilemmas in practical applications (Jain, Murty, and Flynn 1999).

Clustering algorithms are typically classified into two majority types, hierarchical and partitional (or flat), as shown in Figure 2-4 (Jain, Murty, and Flynn 1999), although some different classifications exist (Rokach and Maimon 2005). In this research, the focus is on how similar Web snippets are grouped (not a thorough research on clustering algorithms), thus two representative algorithms, K-Means which is considered a typical partitional algorithm, and Hierarchical Agglomerative Clustering (HAC) which represents the hierarchical clustering algorithm family, are considered. More specifically, the *K-Means* with its generative variety, the *Expectation-Maximization*, and an efficient HAC algorithm is taken into account. Jain, Murty and Flynn point out (1999) with an appropriate initial partition which is obtained by other techniques/heuristic, K-Means works very well on large

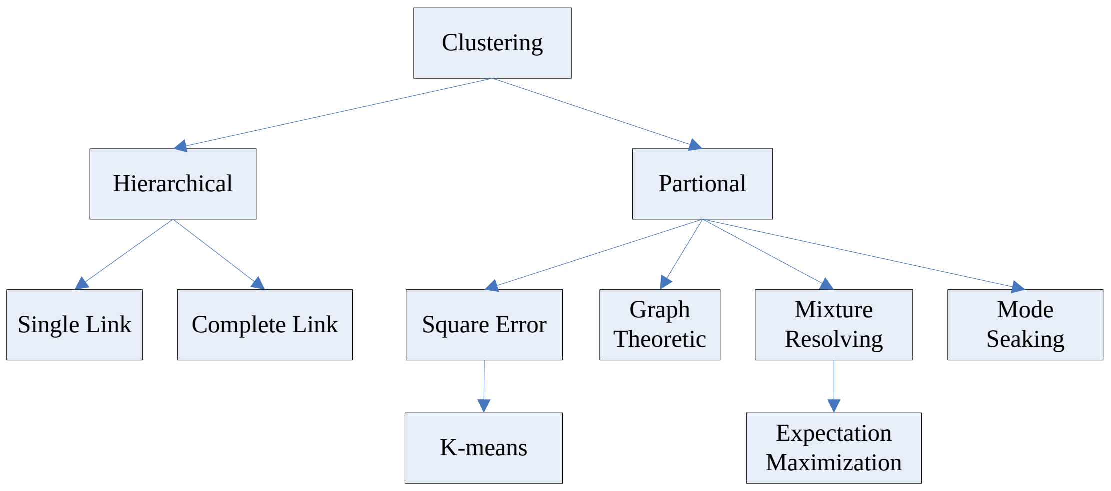


Figure 2-4 A taxonomy of clustering algorithms (Jain, Murty, and Flynn 1999)

data set; and domain knowledge will further improve the performance of the algorithms. However, partitional algorithms suffer from unstructured set of clusters and need to assign an arbitrarily specified number of clusters as input. On the other hand, HAC can successfully alleviate this issues; nevertheless, low efficiency is one of the major shortcomings of HAC algorithms (Manning, Raghavan, and Schütze 2008).

As in text categorization, documents, which are also called entities (Mirkin 2005), patterns, feature vectors, observations (Jain, Murty, and Flynn 1999), are represented by vectors in R^v, a v dimensional real space (feature space) where v is the number of terms, or features, or sometimes attributes, in the document set. A formed cluster is usually represented by a centroid (or prototype), which is the mean of the vectors in the cluster, or simply a representative vector of the cluster. A distance measure quantifies or approximates the similarity between two vectors in the feature space.

2.2.1 K-Means

K-Means clustering algorithm (MacQueen 1967; Kanungo et al. 2000; Han and Kamber 2006; Jain 2009) is a very popular clustering algorithm for its clear mathematical properties, computational ease and efficiency (Mirkin 2005). It clusters the data set into K non-overlapping groups that are usually represented by the centroids of the groups. A partition S = $\{S_1, S_2, \ldots S_K\}$ is composed of K subsets $S_i \in S$, i = 1, 2, …, K, with a set of centroids c = $\{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K\}$.

Three core factors in K-Means are the initial set of centroids, the number K, and measurement of distance between data items.

K-Means employs minimum distance rule to arrange data items into different centroids. It compares the distances between a data item and each of the centroids and assigns the data item to the cluster with nearest centroid. Many distance measures are proposed as presented in Table 2-1. In the field of text clustering, the most popular measures are cosine similarity and Manhattan distance. Euclidean distance is also a widely accepted distance measure.

The number of K can be selected randomly, or using MaxMin and Anomalous Pattern (AP) approaches for its simplicity and effectiveness (Mirkin 2005). Chiang and Mirkin (2006) compare eight approaches to select the number of clusters. However, the data set used in their experiments is not a text document set so their conclusion needs to be further verified in the circumstance of text clustering.

The centroids can also be selected by using the MaxMin or AP algorithms for which the details are presented in Appendix 5. The algorithms are all computational expensive because the complexity of the algorithm is O(knt) where k is the number of clusters, n is the total number of data items, and t is the iterations to make the generated clusters stable (Han and Kamber 2006).

Note that some of the generated clusters have only one pattern, or less than a specified number of patterns. These clusters are to be merged with other clusters and a final cluster number K is then decided. A predefined threshold K_0 can also be a stop condition if the number of formed clusters reached K_0. Standard K-Means algorithm can then be employed with the formed number of K and the centroids $\mathbf{c}_i$, i = 1, 2, … K. This approach is called *Intelligent K-Means algorithm*, or *iK-Means* which works well on low dimensional space, and may fail when pattern are scattered too far away from each other. In other words, iK-Means is not good at dealing with sparse data set (Mirkin 2005).

For example, Beyer et al. (1999) indicate that in a certain circumstances, it is hard to tell the difference between the maximum and minimum distances, and they suggest using "ε-Radius Nearest Neighbors" to retrieve for meaningful data points within a hypersphere. While the distance metrics utilized in K-Means is usually *Euclidean distance*, some researchers find *Manhattan distance*, the case when the order *r* of Minkowsky distance (Table 2-1) is one, outperforms *Euclidean distance*. Furthermore, fractional distance metric, that is when *r* is less than 1, demonstrated theoretically and empirically provides more significant improvement on effectiveness for clustering algorithms such as K-Means (Aggarwal, Hinneburg, and Keim 2001).

However, choosing the number of clusters K and initializing the centroids are all computational expensive, and using categorization results to determine K and initialize the centroids have not be been proposed in literature so far.

2.2.2 Model-based Clustering - Expectation-Maximization (EM) Algorithm

Model-based clustering postulates that documents are generated according to a probability model, and is directed at recovering the original model from the observed data. The recovered models can then specify the models and estimate the probability of a given document to the clusters (Manning, Raghavan, and Schütze 2008). Maximum-likelihood is a very popular technique to estimate the model parameters, and more generally, the expectation-maximization algorithm that iteratively estimates the model parameters are frequently utilized to obtain more accurate parameter estimation because the clustering is unknown a priori.

Expectation-maximization algorithm was firstly summarized by Dempster, Laird and Rubin for computing maximum likelihood estimates from incomplete data (Dempster, Laird, and Rubin 1977). Based on the application fields and the property of random variables, EM algorithms are presented from different perspectives (Chakrabarti 2003; Dempster, Laird, and Rubin 1977; Han and Kamber 2006; Kung, Mak, and Lin 2004; Moon 1996; Prescher 2003). EM algorithms have two steps, E-step estimate a set of parameters; M-step maximizes the likelihood with respect to the estimated parameters. The process iterates until a given stop condition is satisfied.

For the reason of its computational cost, the EM algorithms is not evaluated in this research although it may be evaluated in the future, however, its specific form K-Means is to be implemented and evaluated.

2.2.3 Hierarchical Clustering Algorithms

One of the popular hierarchical clustering algorithms is Hierarchical Agglomerative Clustering (HAC) algorithm. HAC has three steps to cluster a collection of data as described in Figure 2-5 (Jain, Murty, and Flynn 1999).

When merging clusters, there are different strategies to compare the similarity between clusters. Based on the different similarity strategies, hierarchical clustering algorithms are

1. compute the proximity matrix containing the distance between each pair of patterns. Treat each pattern as a separate cluster.

2. Find the most similar pair of clusters using the proximity matrix. Merge these two clusters into one cluster. Update the proximity matrix to reflect this merge operation.

3. If all patterns are in one cluster, stop. Otherwise, go to step 2.

Figure 2-5 Hierarchical Agglomerative Clustering algorithm (Jain, Murty, and Flynn 1999)

Table 2-4 similarity notions of HAC algorithms (Manning, Raghavan, and Schütze 2008)

HAC algorithms	**Definition**	**Calculation**
Single-link	The similarity of their most similar members	Max(SIM(i, k_1), SIM(i, k_2))
Complete-link	The similarity of their most dissimilar members	Min(SIM(i, k_1), SIM(i, k_2))
Centroid	All similarities between documents in two different clusters	$(\frac{1}{N_m}\vec{v}_m)\cdot(\frac{1}{N_i}\vec{v}_i)$
Group-average	All similarities among documents, including pairs from the same cluster	$\frac{1}{(N_m+N_i)(N_m+N_i-1)}[(\vec{v}_m+\vec{v}_i)^2-(N_m+N_i)]$

classified into single-link, complete-link, centroid-based, and grouped-averaged. Table 2-4 summarizes the four notions of cluster similarities (Manning, Raghavan, and Schütze 2008). In this table, C_k represents the k-th cluster, C_{k1} represents the merged cluster of C_{k1} and d C_{k2} respectively; N_m and N_i are the number of documents in $C_{k1} \cup C_{k2}$ and C_i, $\vec{v}_m$ and $\vec{v}_i$ are the vector sums of $C_{k1} \cup C_{k2}$ and C_i respectively.

Another kind of hierarchical clustering techniques is the divisive hierarchical clustering, for instance, the Bisecting K-Means (Steinbach, Karypis, and Kumar 2000). The basic idea of the bisecting K-Means is to split one cluster into two sub-clusters and repeat the process until a stop condition is satisfied. The produced clusters can be taken as an initialization set and further refined by applying regular K-Means algorithm. Experimental results demonstrate that bisecting K-Means outperforms HAC for large-scale document collection (more than 1000 documents). For small sample document set, HAC performs better (Wang and Hodges 2005). Since the algorithms are computationally expensive ($O(n^3)$ where n is the number of documents to be clustered (Zhao and Karypis 2002)), from the perspective of Web search results clustering, HAC and K-Means are two algorithms evaluated.

2.2.4 Latent Semantic Analysis for Clustering

Latent Semantic Analysis (LSA) or Indexing (LSI) is a “mathematical/statistical technique for extracting and inferring relations of expected contextual usage of words in passages of discourse.” (Landauer, Foltz, and Laham 1998) It takes “advantage of implicit higher-order structure in the association of terms with documents (“semantic structure”) in order to improve the detection of relevant documents on the basis of terms found in queries”. (Deerwester et al. 1990) LSA is proposed to alleviate the polysemous and synonymous issues which seriously affect the performance of information retrieval, in terms of recall and precision, based on “lexical matching” (Furnas et al. 1988; Dumais 2004). In addition to information retrieval, LSA also has been applied to a wide range of applications such as classification, and filtering (Kontostathis and Pottenger 2006), cross-language/multilingual information retrieval (Dumais 2004) and clustering (Wei, Yang, and Lin 2008). The power of LSA stems from the ability of SVD - Singular Value Decomposition (Golub and Loan 1996) to encapsulate high-order term co-occurrence information, which plays an essential role in the effectiveness of information retrieval and data mining (Kontostathis and Pottenger 2006).

One assumption of LSA is that there should exist a latent structure in the terms by documents matrix which is used to represent the set of documents because “some closely related documents should contain nearly identical patterns of terms, and synonymous terms should have highly similar patterns of occurrence across documents” (Furnas et al. 1988). The latent structure can represent the document set in a more “parsimonious” way by squeezing out redundancy and noises. Due to the polysemous and synonymous characteristics of natural languages, “terms actually observed” in a document are “only a sample of the true, larger pool of terms that might have been associated with it.” (Furnas et al. 1988) It is therefore concluded that “the observed term-document matrix can be thought of as a true association matrix obscured by some sort of sampling error.” (Furnas et al. 1988) Because it is almost impossible to access the “true” structure, estimation is needed. The proposed approach to this issue is the SVD factor analysis technique (Golub and Loan 1996).

Deerwester et al. (1990) find that in their experiment, precision increases with the number of the largest singular values k ranges from 10 to 100, and then begin to turn down. Dumais (2004) presents a similar experimental result for selecting the number of k. Dupret (Dupret 2003) indicates lower ranked dimensions can discern relationship among terms, and synonyms can be well discriminated if a higher rank is assigned.

When LSA is applied in text clustering, the number of clusters is also experiment-based. With an appropriately selected threshold, hierarchical agglomerative clustering algorithm (Jain and Dubes 1988; Jain, Murty, and Flynn 1999) can be employed to group the documents based on semantic similarities estimated by $\hat{A}^T\hat{A}$ (Hasan and Matsumoto 1999) where $\hat{A}$ is the term-document matrix in the SVD reduced spaces, and $\hat{A}^T$ is the transpose of matrix $\hat{A}$.

The well-know problem of LSA is its computational complexity that makes it very hard to be applied in an interactive application scenario such as Web snippet clustering. Therefore, in this research, LSA is only evaluated for effectiveness comparison purpose by using a small size data set.

2.2.5 Using External Knowledge to Boost Text Clustering

Feldman and Sanger (2006) suggest the use of unlabeled data to improve classification, because labeling a large data set involves huge human labour and is thus too expensive; whereas unlabeled data exists in abundance and can be obtained with low cost. The two approaches introduced are EM and co-training. In EM algorithm (refer to Appendix 5 for details), a probability model is first trained over the labeled document set. Then, E and S steps are iterated until the convergence of a local maximum. In E-step, the unlabeled documents are classified by the current model; in M-step, the model is trained over the new combined model.

The co-training algorithm represents each document in two different forms/views, for example, a webpage's content and the anchor text of the page. Co-training uses bootstrapping strategy "in which the unlabeled documents classified by means of one the views are then used for training the classifier using the other view, and vice versa."

This EM approach is kind of semi-supervised machine learning, for which the main attention is to explore the huge amount of available unlabeled data to boost the performance of supervised learning (Zhu 2008b). However, if the anchor text is not available, the co-training will not work.

Kyriakopoulou (2008) summarizes three potential approaches of text categorization aided by clustering. The first method is feature compression or extraction by clustering algorithm. This approach is actually a kind of feature extraction as will be discussed in the following section (Section 2.3). A second technique is semi-supervised learning which utilizes both labeled and unlabeled documents as training examples for the reason that human labeled document set are rare and expensive to obtain. Clustering in large-scale classification problems is the latest research area in text categorization. The focus of the approach is to reduce the training time by using clustering as a down-sampling pre-process to reduce the scale of training set. This approach differs from feature selection/extraction in that it intends to remove training examples that might not helpful for text categorization.

Supervised learning is a process which leverages external knowledge - the labeled training document set - to train a prediction model for a list of labels (categories, or classes), and the trained model can then predict a label for an unlabeled test document. This process compares the similarity between a test document and training documents, and the similarity is thus the inter-similarity. Unsupervised learning, without external knowledge available, groups test documents into clusters by exploring the similarities among the test documents, and this kind of similarity is intra-similarity. To the best of our knowledge, combining inter- and intra-similarities to boost the performance of supervised and unsupervised learning has not been proposed so far. An innovative algorithm *boostingUp* is to be discussed in detail in Chapter 3, and experimental results are to be presented in Chapter 6.

2.2.6 Evaluation Metrics of Clustering

Criteria for evaluating clustering are mainly divided into two categories: external and internal, although there is no agreement on the definition of what a good clustering is (Rokach and Maimon 2005). The most widely accepted internal criterion is the Sum of Squared Error (SSE) (Rokach and Maimon 2005), and the external criteria are Normalized Mutual Information (NMI), Rand Index (RI), and F-measure (Manning, Raghavan, and Schütze 2008).

2.2.6.1 Internal Criterion – Sum of Square Error (SSE)

Let C_k is k-th cluster, $|C_k|$ is the number of members in C_k; $\boldsymbol{\mu}_k$ is the centroid of cluster k, $\boldsymbol{\mu}_k = \frac{1}{|C_K|}\sum_{x_i \in C_k} \mathbf{x}_i$, then, SSE is defined as (Rokach and Maimon 2005):

$$SSE = \sum_{k=1}^{K} \sum_{x_i \in C_k} \| \mathbf{x}_i - \boldsymbol{\mu}_k \|^2$$

SSE is suitable for well separated compact clusters (Rokach and Maimon 2005). However, performing well under SSE does not guarantee effectiveness in a user-centred application, such as information retrieval and search results clustering (Manning, Raghavan, and Schütze 2008).

2.2.6.2 External Criteria

Normalized Mutual Information

Supposed that C = $\{c_1, c_2, \ldots, c_K\}$ is the set of classes that groups a collection of documents. Let $\Omega = \{\omega_1, \omega_2, \ldots, \omega_K\}$ is the set of K clusters generated by a clustering algorithm. Mutual Information[10] is defined as (Manning, Raghavan, and Schütze 2008):

$$\begin{aligned} MI(\Omega, C) &= \sum_{j=1}^{K}\sum_{i=1}^{K} P(\omega_i \cap c_j) \log \frac{P(\omega_i \cap c_j)}{P(\omega_i)P(c_j)} \\ &= \sum_{j=1}^{K}\sum_{i=1}^{K} \frac{|\omega_i \cap c_j|}{N} \log \frac{N|\omega_i \cap c_j|}{|\omega_i||c_j|} \end{aligned}$$

[10] The notion of Mutual Information used by Manning, Radhavan and Schütze (2008) is referred to as Information Gain by Sebastiani (2002). In this research, Sebastiani's notion is used except in this section.

Where N is total number of documents in the collection, P(ω_i), P(c_j), and P($\omega_i \cap c_j$) are the probabilities of a document being in cluster ω_i, class c_j, and in the intersection of ω_i and c_j. Note that these probabilities are estimated by maximum likelihood estimation.

MI reaches maximum when K = N. This is problematic because intuitively a smaller K is more preferable. To overcome the issue, large number of K should be penalized, and one of such strategy is to divide MI by the sum of entropies of Ω and C which getting bigger as the members in Ω or C increase. Particularly, |H(Ω) + H(C)|/2, the tight upper bound on MI(Ω, C) is chosen to guarantee that NMI(Ω, C) ∈ [0,1].

$$NMI(\Omega, C) = \frac{MI(\Omega, C)}{[H(\Omega) + H(C)]/2}$$

Where

$$\begin{aligned} H(X) &= -\sum_{k=1}^{K} P(\omega_k) \log P(\omega_K) \\ &= -\sum_{k=1}^{K} \frac{|\omega_k|}{N} \log \frac{|\omega_k|}{N} \end{aligned}$$

Rand Index and F_β Measures

According to Manning et al. (2008), supposed the number of documents in a given document set is N, the number of possible pairs of the N documents is N × (N – 1) / 2. There are four possible decisions for each of the documents pairs which are similar to that defined by Yang (1999) in a contingency table (Table 2-2) for evaluating text categorization algorithms.

True Positive (TP): two similar documents are assigned in a same cluster;

True Negative (TN): two different documents are assigned to different clusters;

False Positive (FP): two dissimilar documents are assigned into a same cluster;

False Negative (FN): two similar documents are assigned to different clusters.

$$RandIndex(RI) = \frac{TP + TN}{TP + FP + FN + TN}$$

Usually, arranging dissimilar documents in the same cluster (false positive) is more tolerant than separating similar documents into different clusters (false negative). Since Rand Index gives identical weights to the two scenarios, F_β (refer to Section 2.1.1.7) measure with $\beta > 1$ is suggested to overcome the issue (Manning, Raghavan, and Schütze 2008).

2.2.7 Web Snippet Clustering

Some early works on Web snippet clustering is *Scatter/Gather* (Cutting et al. 1992; Hearst and Pedersen 1996) which uses a partitional algorithm named Fractionation. It is found in the research that search results clustering can significantly improve similarity search ranking, and thus validate the cluster hypothesis that relevant documents tend to be more similar to each than non-relevant documents (Rijsbergen 1979). *Northern Light* (Notes 1998), one of the early commercial search engines, organizes search results into different Custom Search Folders (clusters). These folders are formed based on keywords, information sources and other criteria. Each folder is labeled according to a single word or a two words phrase; and documents that contain the specified label will be organized under that label's folder. However, as pointed out by Zamir and Etzioni (1998), *Northern Light* does not reveal the techniques of how to create the folders.

Grouper (Zamir and Etzioni 1998, 1999) is another example of early research on clustering Web search results. Suffix Tree (Farach 1997), a linear time clustering algorithm, is utilized to re-organize the Web results. A suffix tree is a rooted, directed tree; each node of the tree is a cluster of search results with a common phrase that labels the node. The Suffix Tree Clustering (STC) algorithm identifies and uses frequency-words to form base clusters – nodes in the Suffix Tree, which are further merged to form the final clusters. Experimental results demonstrate STC performs better than other popular clustering algorithms in a number of conditions. The experimental data also show that "clusters based on snippets are almost as good as clusters created using the full text Web documents." However, this statement needs to be further verified by extensive experiments and the judgment of what is meant by "good clusters", needs to be clarified.

Lingo (Osiński, Stefanowski, and Weiss 2004; Osiński and Weiss 2005) aims at improving the quality of cluster descriptions, which can be long, ambiguous and sometimes meaningless. It focuses on producing readable and unambiguous descriptions of formed clusters. *Lingo* applies Singular Value Decomposition (Deerwester et al. 1990) to find any existing latent structure of diverse topics in the search results. Frequency terms or phrases, which are derived by a suffix array algorithm from the returned Web snippets, are used to represent the abstract concepts in the left singular value matrix in SVD. *Lingo* then matches the derived cluster descriptions with the extracted topics by comparing the cosine similarities between frequency terms/phrases and the extracted abstract concepts. Problems associated with *Lingo* are that it does not consider cluster merging (Osiński, Stefanowski, and Weiss 2004), and the formed clusters are not hierarchically organized. In addition, SVD is time consuming, the complexity for an $m \times n$ matrix is $O(m^2n + n^3)$ (Osiński and Weiss 2005).

Zeng et al. (2004) propose the Web snippets clustering problem can be dealt with as a salient phrase ranking problem. With a given query and a list of ranked titles and snippets returned by a search engine, salient phrases can be extracted and ranked as candidate cluster names based on a regression model learned from human labeled training data. The documents are assigned to relevant salient phrases to form candidate clusters, which are then merged to form the final clusters. Experimental results reveal that accurate clusters with short names can be produced. The main weakness of this algorithm is that it only generates flat clusters, and cannot reveal the hierarchical structure of the Web snippets.

Geraci et al. (2006) suggest a modified furthest-point-first method for a k-centre clustering algorithm. By exploring the triangular inequality, instead of computing the distances of each point from all other points, they calculate the distances of each point from each cluster centre. This method is fast and accurate, and comparable or better than recently proposed, fast versions of K-Means algorithms. The distance function is also refined by weighting the title and snippet of the returned Web snippet differently to avoid calculating the cosine-normalized tf-idf weight. Experimental results on the information snippets from the ODP show that this algorithm is efficient and effective. However, the formed clusters are organized in a flat structure.

Ferragina and Gulli (2005) have developed a clustering engine named as *SNAKET* which organizes on-the-fly meta-search engine results into hierarchically labeled folders. To select and rank sentences, *SNAKET* takes advantages of two knowledge bases: one is the anchor texts ("the segment of a webpage surrounding an hyperlink" (Ferragina and Gulli 2005)) extracted from more than 200 million webpages to enrich the poor content of the corresponding snippets; the other is the ODP hierarchy used to help ranking the gapped, variable length labels extracted from the snippets. *SNAKET* employs an innovative bottom-up hierarchical clustering algorithm aimed at constructing a hierarchical folder that takes consideration of the number of created folders and the balance among its offspring. Experimental results demonstrate the effectiveness and efficiency of *SNAKET* is comparable to that of the most popular *Vivisimo.com* (www.vivisimo.com).

Boley et al. (Boley et al. 1999) propose to utilize graph based partitioning algorithms, the Association Rule Hypergraph Partitioning and Principal Direction Divisive Partitioning algorithm, to cluster Web documents; and find that in some cases the suggested algorithms outperform traditional distance based and probabilistic based clustering algorithms.

One serious question in the field of text clustering is if there exists a "real" cluster structure, or, what is the real "K". This question is out the scope of this research and a lot of literature is available for this topic (Monti et al. 2003; Sugar and James 2003; Hamerly and Elkan 2003; Jain 2009).

2.3 Dimensionality Reduction

The high dimensionality of the term space may be problematic in the field of text categorization because many sophisticated learning algorithms used for classifier induction can hardly scale to the high dimensionality (Sebastiani 2002b; Mladenic and Grobelnik 2003). Dimensionality reduction is the process to reduce the high vector space for the purpose of efficiency. In fact, dimensionality reduction can not only boost categorization efficiency, but also moderately improve the effectiveness (less than 5%) of categorization because noise features are at the same time removed (Yang and Pedersen 1997).

Dimensionality reduction problem could be described as trying to construct a optimal function $f:\Re^{|\mathcal{F}|}\rightarrow\Re^{|\mathcal{F}'|}$ according to certain criterion to maximize the effectiveness and efficiency of text categorization algorithms, where $|\mathcal{F}|$ and $|\mathcal{F}'|$ are the dimensions of original feature set $\mathcal{F}$ and reduced feature set $\mathcal{F}'$.

Generally speaking, there are two types of dimensionality reduction: feature selection and feature extraction. Feature selection aims at selecting (or picking up) from the original feature set $\mathcal{F}$ the set of $\mathcal{F}'$ features ($|\mathcal{F}|$ « $|\mathcal{F}'|$) with a given criterion, such as a predefined number of features. $\mathcal{F}'$ is a subset of $\mathcal{F}$. Feature extraction, on the other hand, intends to generate from the original feature set $\mathcal{F}$ the set of $\mathcal{F}'$ of "synthetic" features ($|\mathcal{F}|$ « $|\mathcal{F}'|$) that maximize effectiveness. Features in $\mathcal{F}'$ might not of the same type of the features in $\mathcal{F}$, but of the form of combinations or transformations (Sebastiani 2002b). Since feature extraction is usually more complex and more computational expensive than feature selection, it is not considered in this research.

2.3.1 Dimensionality Reduction by Feature Selection

2.3.1.1 Main feature selection algorithms suggested by Sebastiani (2002)

A number of feature selection algorithms have been developed for text categorization since 1980's. Following Sebastiani (2002b), some of the popular feature selection approaches are summarized as shown in Table 2-5, where N is the total number of documents involved, probabilities are interpreted on an space of documents and are estimated by counting occurrences in the training set. $P(t_k, c_i)$ indicates the probability that for a random document *d*, feature t_k occurs in *d* and *d* belongs to category c_i), and $P(\bar{t}_k, c_i)$ indicates the probability that for a random document *d*, feature t_k does not occur in *d* and *d* belongs to category c_i. All functions are specified "locally" to a specific category c_i; in order to assess the value of a term t_k in a "global", category-independent sense, either the sum $f_{sum}(t_k)=\sum_{i=1}^{|C|} f(t_k, c_i)$,

Table 2-5 Main functions used for selection (Sebastiani 2002b)

Function	**Denoted by**	**Mathematical form**
Frequency-based	FR	Document frequency, term frequency
Information gain	IG(tk, ci)	$\sum_{c\in\{c_i,\bar{c}_i\}}\sum_{t\in\{t_k,\bar{t}_k\}} P(t,c)\log\frac{P(t,c)}{P(t)P(c)}$
Mutual information	MI(tk, ci)	$\log\frac{P(t_k,c_i)}{P(t_k)P(c_i)}$
Chi-square	χ^2 (tk, ci)	$\frac{N[P(t_k,c_i)P(\bar{t}_k,\bar{c}_i)-P(t_k,\bar{c}_i)P(\bar{t}_k,c_i)]^2}{P(t_k)P(\bar{t}_k)P(c_k)P(\bar{c}_k)}$, N is total document number
NGL coefficient	NGL(tk, ci)	$\frac{\sqrt{N}[P(t_k,c_i)P(\bar{t}_k,\bar{c}_i)-P(t_k,\bar{c}_i)P(\bar{t}_k,c_i)]}{\sqrt{P(t_k)P(\bar{t}_k)P(c_k)P(\bar{c}_k)}}$
Relevancy score	RS(tk, ci)	$\log\frac{P(t_k\mid c_i)+\rho}{P(\bar{t}_k\mid \bar{c}_i)+\rho}$ ρ is a constant damping factor, for example, takes the value 1/6.
Odds ratio	OR(tk, ci)	$\log\frac{P(t_k\mid c_i)(1-P(t_k\mid \bar{c}_i))}{P(t_k\mid \bar{c}_i)(1-P(t_k\mid c_i))}$
GSS coefficient	GSS(tk, ci)	$P(t_k,c_i)P(\bar{t}_k,\bar{c}_i)-P(t_k,\bar{c}_i)P(\bar{t}_k,c_i)$

or the weighted sum $f_{wsum}(t_k)=\sum_{i=1}^{|C|}P(c_i)f(t_k,c_i)$, or the maximum $f_{\max}(t_k)=\max_{i=1}^{|C|} f(t_k,c_i)$ of their category-specific values f(t_k, c_i) are usually computed. Note that to estimate the probabilities in practice, Maximum-Likelihood Estimates are usually utilized. A detailed discussion of the algorithms is presented in Appendix 5.

2.3.1.2 Other feature selection algorithms

In addition to the popular feature selection algorithms discussed above, there are still list of other algorithms proposed to pick up informative features to construct a reduced feature space for text categorization or clustering algorithms intends to not only improve its efficiency but also effectiveness as well. For instance, the feature competitive algorithm (Ye and Lo 2000), Principal Component Analysis (Abdi and Williams 2010), Independent Component Analysis (Hyvärinen and Oja 2000), Discrete Fourier Transform (Candan, Kutay, and Ozaktas 2000), Bi-Normal Separation, Odds Ratio Numerator, Power, and Probability Ratio (Forman 2003).

The effectiveness of feature selection algorithms depends on the document collections involved, number of features selected, evaluation criteria, application domains and the text categorization or clustering algorithms applied. However, in general, experimental results from literatures indicate that OR, GSS, NGL are usually performs better than λ^2 and IG, and MI is almost always present poor results. In this research, in addition to the factor of effectiveness, efficiency is also an important element to be considered.

2.3.2 Dimensionality Reduction by Feature Extraction

Feature selection algorithms intend to select the most informative features directly from the set of original features; term extraction algorithms, on the other hand, attempt to generate a new set of “synthetic” features that maximize categorization effectiveness. It is believed that due to the problems of polysemy and synonymy, the original features may not be optimal dimensions for document content representation. Feature extraction maybe an effective solution to solve the problems by generating artificial features that do not suffer from them. Two steps are needed for feature extraction, the first is extracting new features from the old one, and the second is representing the document using the extracted features (Sebastiani 2002b).

The two widely used feature extraction approaches are Latent Semantic Analysis and Term Clustering. LSA, as discussed in the previous section, reduces the dimensionality of term-document matrix via singular value decomposition. Applying LSA to both training and testing collections to form a new term-document set, and then good performance is to be expected when testing documents are classified (Schütze, Hull, and Pedersen 1995).

Term clustering approach measures the relationship between terms and consequently forms some term clusters, the terms in a cluster are then represented by the centroid of the cluster and subsequently reduces the dimension of term-document matrix (Sebastiani 2002b). Experimental results presented earlier by some researchers (Li and Jain 1998; Slonim and Tishby 2001; Lewis 1992) demonstrate this approach can sometimes marginally improve the performance of text categorization. However, some recent research (Bekkerman et al. 2003; Dhillion, Mallela, and Kumar 2003) indicates that term clustering is a effective approach to improve the performance of text categorization.

2.3.3 Wrapper, Filter and Embedded Approaches

Feature selection algorithms can also be categorized into wrapper, filter and embedded approaches based on three common aspects for all feature selection (not for feature extraction) algorithms, that are 1) feature subset generation or search strategy, 2) evaluation criterion definition, and 3) evaluation criterion estimation or assessment methods (Guyon et al. 2006). Filters differ from wrappers in that they select a feature subset by utilizing evaluation criteria not involving any specific learning algorithms; whereas wrappers select a subset of features based on criteria to evaluate machine learning algorithms. The common aspect of the two types of algorithms is they both employ search strategy to exhaustively explore feature combination space. On the other hand, embedded methods choose features during the process of training of a given learning machine and uses different search strategies from wrappers (Guyon et al. 2006).

Since this research focuses on Web snippet categorization, therefore, only feature selection algorithms that have been evaluated in the field of text categorization are considered in this thesis. Obviously, a large-scale and comprehensive evaluation of feature selection algorithm for text categorization is necessary in the future.

2.4 Personalization

Personalization involves not only retrieving syntactically relevant information, but also involves taking into consideration a user's information consumption pattern, searching strategies, application used and the nature of the information (Pitkow et al. 2002).

2.4.1 Techniques for Personalization

According to Pitkow et al. (2002), the most widely applied approaches of personalization are search-term augmentation and search results re-organization. Personalization can also be classified according to how to collect user-feedback and subsequently profiling procedure. Within this framework, Micarelli et al. (2007) suggest personalization can be based on Current Context, Search History, Rich User Models, Collaborative approaches, Result Clustering, and Hypertextual Data. Personalization can be classified by distinguishing if the user information is collected explicitly or implicitly (Micarelli et al. 2007).

2.4.2 User Profiling

One crucial factor in personalization is how to construct a user profile which can accurately express the user's search interests, although it is obviously a challenging task when considering concept drift (Pitkow et al. 2002; Tsymbal 2004; Webb, Pazzani, and Billsus 2001; Koychev 2001) and privacy protection (Kobsa and Schreck 2003; Shen, Tan, and Zhai 2007). Rumpler (2001) suggests user profiles can be generated via different approaches, such as sociological approach (explicit), human-machine approach (implicit), cognitive approach (contextual), and case-based reasoning. In this research, the user profiling approaches are classified into implicit, explicit and hybrid (a combination of implicit and explicit) approaches as shown in Table 2-6.

Explicit approach obtains users information via explicit user-feedback. It allows user to choose from a range of personal interests that can then be used to expand queries and filter search results. Explicit approaches assume users can accurately express their search preferences and thus provide better search experience. However, there are several drawbacks of explicit approaches. As pointed out by Gauch, Chaffee, and Pretschner (2003), this approach imposes an extra burden on users to take their time to express their search interests; the description of users interests may not accurate; users' interests shift over time whereas few users are willing to update their interests changing. If users decline to explicitly express their interests, no user profile is created.

Implicit approaches construct user profiles by means of implicitly collecting and analysing users' search history (such as Page Interest Estimators (Chan 1999)), downloaded webpages, and search log file which includes meta-data describing the content of webpages (Stermsek, Strembeck, and Neumann 2007). While implicit approaches intend to address the issues of explicit methods, there are different opinions on the effectiveness of implicitly created user profiles(Dou, Song, and Wen 2007). Another issue to be attacked for implicit user profiling approach is the privacy protection that is out the scope of this research.

A hybrid user profiling method which combines the advantages of the above two approaches has been applied by several researchers as shown in Table 2-6. This research also uses the hybrid approach to build user profiles.

In fact, interactivity between a search browser and its users is also a kind of personalization, and it is a kind of more effective and efficiency personalization approach compared with the personalization methods so far discussed. The advantages of the interactive personalization are 1) it involves no privacy protection problems which is one of the biggest obstacles to prevent users to allow search engines to provide personalized results because the fear of the leaking of privacy information; 2) it enables users to express their real information needs more accurately during the interactive process since even human beings need to communicate several rounds to exchange information before they can understand each other well; 3) it is efficient because it does not need complex algorithms aiming at building accurate user profiles.

In this research, RIB entitles users to explicitly express their search interests, it also collects users' search information to enrich and update the user profile to address the search concept drift issues. In addition, RIB allows users interactively deliver their search preference by selecting interest categories where the Web search results are grouped into.

Table 2-6 Personalization techniques

	Researcher	**Information collected**	**Interests representation**	**Matching/expanding approach**
Explicit	Chirita et al. (2005)	A list of interested ODP categories.	ODP Web directory	Semantic distance function + PageRank
	Glover et al. (2001)	Preferred topics and resources	Hierarchical topic tree	Search results classifying, kNN + VSM
	Gentili, Micarelli and Sciarrone (2003)	Pages visited	Semantic network	Query expansion, filtering, feedback, VSM + K-Means
	Micarelli and Sciarrone (2004)	User interests	Semantic network	Artificial neural networks
	Panzzani, Muramatsu and Billsus (1996)	Keywords of pages, rating of users	Rated Web pages	Search results grouping, Bayesian classifier
	Seher (2007)	Search interests	Domain ontology	Decision tree
	Shavlik et al. (1999)	Keywords from Web pages advised	Advised Web pages	Search results classifying, Neural networks
	Tanudjaja and Mui (2002)	Keywords and concepts	“Coloured” ODP categories	Filtering results, Personalized HITS
Implicit	Gauch, Chaffee and Pretschner (2003)	Browsing history	Top four level ODP categories	Search results grouping, VSM
	Barrett, Maglio and Kellem (1997)	URLs visited, text in the URLs	Concepts with a cluster of terms	Not revealed
	Chen & Sycara (1998)	Browsing activity	keywords	Keyword expansion + VSM
	Dumais et al. (2003)	All user activity	Predefined filters	Search results filtering, probabilistic model
	Stamou and Ntoulas (2009)	Web pages visited	ODP categories + WordNet	Re-rank results, a new rank algorithm
	Liu, Yu and Meng (2002)	Search logs	Categories in first three levels of ODP	Grouping results, Rocchio-based algorithm
	Godoy and Amandi (2006)	Web pages visited	Hierarchical category structure	Re-organize results, cosine similarity, kNN
	Mobasher (2007)	Web logs	Ontologies, keywords	Available data mining techniques
	Sieg, Mobasher and Burke (2004)	Search logs	Categories in Yahoo! Directory	Grouping results, Rocchio algorithm
	Trajkova and Gauch (2004)	Browsing activity	Categories in top three levels of ODP	Grouping results, cosine similarity
	Speretta and Gauch (2005)	Queries, Web snippets	Categories in first three levels of ODP	Re-organize results, cosine similarity
	(Xu 2008)	Queries in Log files	Centroid of generated clusters	Re-ranking results, Latent semantic analysis
Combination	Pitkow et al. (2002)	Web pages visited	1000 top ODP categories	Re-organize results, cosine similarity
	Stermsek, Strembeck and Neumann (2007)	Web pages, log file, user activities,	Hierarchical category from meta-data	
	Zhu and Dreher (2008c)	Web pages visited	ODP categories	Re-organize results, cosine similarity, kNN

2.5 Summary

This Chapter reviewed some of the widely employed text categorization, clustering and feature selection algorithms for the purpose of evaluating the algorithms by using the labeled document sets which are generated by exploring the extracted semantic characteristics of the ODP categories. Pertinent literatures related to personalization are also reviewed, especially the work about user profiling approaches.

Supervised learning (categorization) and unsupervised learning (clustering) are two well researched fields, and suggestions have been put forward to combine the two approaches to boost the supervised learning. The proposed *boostingUp* algorithm introduces a new approach to combine supervised and unsupervised learning from the perspective of inter-similarity and intra-similarity. The following Chapter will introduce the conceptual framework of RIB and the mechanism of the *boostingUp* algorithm.

3 Research Overview and Method

This chapter presents the conceptual framework of this research that focuses on improving Web searching via categorization, clustering, personalization and recommendation. Section 3.1 introduces assumptions made in this study and objectives to be achieved. Section 3.2 reviews research methodologies, discusses design research method applied in this project. Section 3.3 presents the framework of RIB – Recommender Intelligent Browser that is a concrete implementation of the proposed approach to enhance Web searching; followed by an introduction of a novel *boostingUp* approach and z-tfidf term-weighting strategy. Section 3.4 introduces how RIB is to be evaluated by collecting users relevance judgment for a set of search results returned from a search engine for 25 ambiguous search-terms. Section 3.5 summarizes this chapter.

3.1 Assumptions and Research Objectives

3.1.1 Definitions

ODP Category

An ODP category is a specifically defined classificatory division within the Open Directory Project knowledge hierarchy. For example, "Databases" is an ODP category of the ODP topic "Top: Computers: Software: Databases".

Web snippet

A Web snippet contains only the title of a webpage and an optional very short (usually less than 30 words) description of the page. Similar to the representation of a document in an information retrieval systems, a Web snippet is usually represented as a vector $\mathbf{d}_j = \{w_{1j}, w_{2j}, \ldots w_{mj}\}$, *where* w_{ij} *is the weight of the term* t_i *in vector* $\mathbf{d}_j$*. T is the term set in an information retrieval system and m is the size of T. tf-idf (Salton and Buckley 1988) is the most popular term-weighting strategy.*

Web snippet categorization Similar to the definition given by Sebastiani (2005),

Web snippet categorization is the task of approximating the unknown target function $\Phi : D \times C \rightarrow [a, b]$ *(that describes how Web snippets ought to be classified, according to a supposedly authoritative expert) by means of a function* $\hat{\Phi} : D \times C \rightarrow [a, b]$ *called the classifier, which returns a categorization status value for a given testing document. a, b are two real numbers and* $a < b$, $C = \{c_1, \ldots, c_{|C|}\}$ *is a predefined set of categories and* $D = \{\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_n\}$ *is a (possibly infinite) set of Web snippets. The categorization status values of* $\hat{\Phi}(\mathbf{d}_j, c_i)$ *for* $i = 1, 2, \ldots |C|$ *are ranked so an appropriate category can be decided for the given Web snippet.*

Web snippet clustering

Web snippets clustering is defined as the task of approximating the unknown *target function Ψ which assigns a class label $l_i \in L$ to $\mathbf{d_j} \in D$ by a function $\hat{\Psi}$. The set of all labels for a Web snippet set D is $L = \{l_1, l_2, . . . , l_K\}$, with $l_i \in \{1, 2, …, K\}$, where K is the number of clusters.*

Personalization

*For a given query/search-term **q**, a set of Web snippet $D = \{\mathbf{d_1}, \mathbf{d_2}, …\}$ returned from a search engine, and a set of user search preferences $S = \{\mathbf{s_1}, \mathbf{s_2}, ... \mathbf{s}_{|S|}\}$ which is subset of a predefined concept collection C which is used to represent all possible user search preferences. Personalization continuously collects user information to maintain an updated S, and compares the similarity $Sp(\mathbf{s}_i, \mathbf{d}_j)$ and re-rank $\mathbf{d}_j \in D$ by the descending order of $Sp(\mathbf{s}_i, \mathbf{d}_j)$ accordingly.*

3.1.2 Assumptions

Assumption 1 The hierarchical structure of ODP categories created by human reflects the hierarchical structure of the content in the Web; it provides a way to hierarchically organize the content of the Web and thus can be employed as a predefined Web directory to categorize Web snippets.

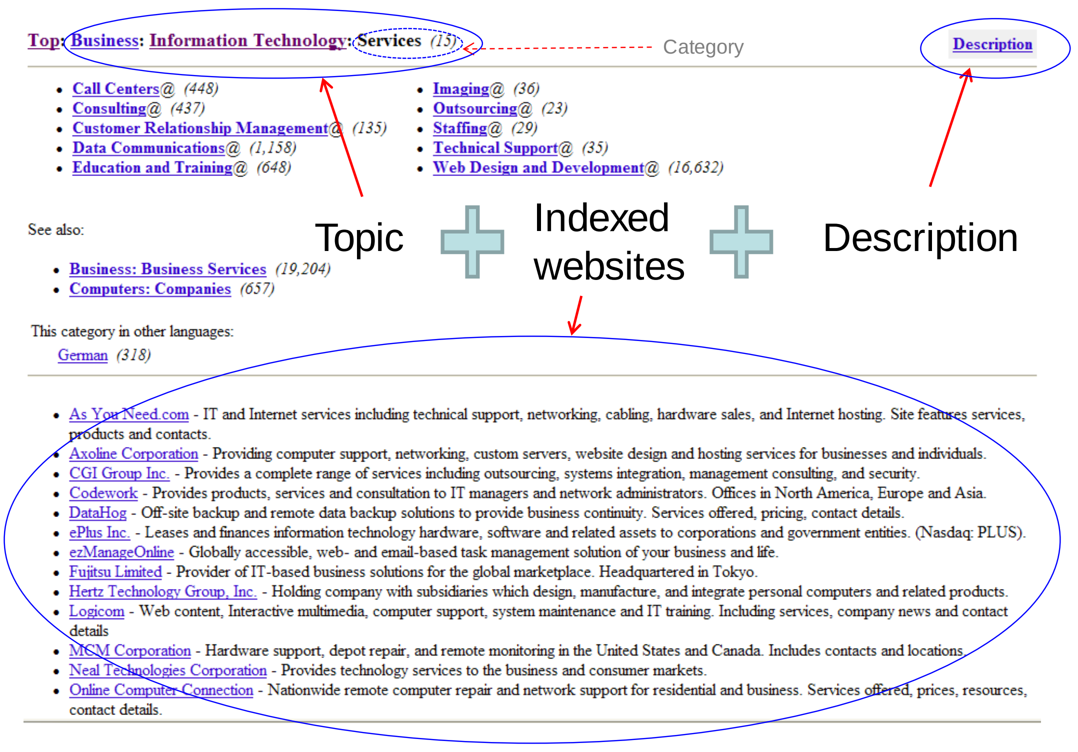
The hierarchical structure of ODP is constructed by human experts with the intention to present a comprehensive, top-down view of the Web content. It is utilized in this research as a knowledge framework to re-organize the research results (Web snippets) of Web search engines. The categorized search results will facilitate Web users to select interest topics. By ontologically filtering irrelevant results, the relevance of Web searching can be enhanced.

Assumption 2 The semantics of an ODP category can be represented by a *categoryDocument* composed of the semantic characteristics of the category.

Semantic characteristics of an ODP category include as illustrated in Figure 3-1 the *topic* of the category, the *description* of the category, and a list of *indexed websites* under the category. Chapter 4 will discuss in details how a *categoryDocument* set is constructed. *CategoryDocument* is an essential concept in this research because it is to be employed to manifest the semantic of an ODP category. *CategoryDocument* set is to be used as a labeled data set to evaluate categorization/clustering algorithms (Chapter 5 and Chapter 6); and as a training data set to train a text classifier to group Web snippets (Chapter 7).

Assumption 3 Information stored in desktop/laptop computers expresses users' Web search interests; privacy issues related to personalization are ignored in this study.

This assumption is based on the intuition that users will store interesting and important information in their personal computers; uninteresting and unimportant information is seldom kept in hard disk. In addition, they usually store interesting information as much as possible. By using this assumption, users' search interests can be collected easily by crawling folders created by users in their desktop computers while the privacy issues (Shen,



Top: Business: Information Technology: Services (15)

Description

- Call Centers@ (448)
- Consulting@ (437)
- Customer Relationship Management@ (135)
- Data Communications@ (1,158)
- Education and Training@ (648)
- Imaging@ (36)
- Outsourcing@ (23)
- Staffing@ (29)
- Technical Support@ (35)
- Web Design and Development@ (16,632)

See also:

- Business: Business Services (19,204)
- Computers: Companies (657)

This category in other languages:

German (318)

- As You Need.com - IT and Internet services including technical support, networking, cabling, hardware sales, and Internet hosting. Site features services, products and contacts.
- Axoline Corporation - Providing computer support, networking, custom servers, website design and hosting services for businesses and individuals.
- CGI Group Inc. - Provides a complete range of services including outsourcing, systems integration, management consulting, and security.
- Codework - Provides products, services and consultation to IT managers and network administrators. Offices in North America, Europe and Asia.
- DataHog - Off-site backup and remote data backup solutions to provide business continuity. Services offered, pricing, contact details.
- ePlus Inc. - Leases and finances information technology hardware, software and related assets to corporations and government entities. (Nasdaq: PLUS).
- ezManageOnline - Globally accessible, web- and email-based task management solution of your business and life.
- Fujitsu Limited - Provider of IT-based business solutions for the global marketplace. Headquartered in Tokyo.
- Hertz Technology Group, Inc. - Holding company with subsidiaries which design, manufacture, and integrate personal computers and related products.
- Logicom - Web content, Interactive multimedia, computer support, system maintenance and IT training. Including services, company news and contact details
- MCM Corporation - Hardware support, depot repair, and remote monitoring in the United States and Canada. Includes contacts and locations.
- Neal Technologies Corporation - Provides technology services to the business and consumer markets.
- Online Computer Connection - Nationwide remote computer repair and network support for residential and business. Services offered, prices, resources, contact details.

Figure 3-1 A categoryDocument includes topic, indexed websites and Description of an ODP category

Tan, and Zhai 2007; Kobsa and Schreck 2003) are not dealt with in this research. Instead, an imaginary user is stimulated for the purpose of evaluating the proposed approach.

3.1.3 Research Objectives

The ultimate objective of this research is to improve Web searching via using ODP as an ontology to re-organize Web snippets to allow users to interactively personalize Web search results. The objectives presented in Chapter 1 can be formally expressed as:

> ***RO1**: Supposed the performance in terms of precision, recall and F_1 of a set of Web snippet D returned by a search engine for a given test query q are Pr_q, Rc_q, and F_{1q}, re-organizing D by means of combining Web snippets categorization/clustering and personalization obtains Pr_q', Rc_q', and F_{1q}'. The research intends to investigate approaches to make the following inequality true,*
>
> $$Pr_q' - Pr_q + Rc_q' - Rc_q + F_{1q}' - F_{1q} > 0$$
>
> *and if possible, maximize it.*

To achieve this objective, according to the experimental results and discussions of Zhu (2007), an important factor is to improve recall while keeping high precision when categorizing Web snippets into ODP categories, this leads to the second research objective:

> ***RO2**. Supposed the performance in terms of precision, recall and F_1 of a categorization algorithm Φ on a given data set D (which is divided into training set and test set) and category set $C = \{C_1, C_2, \ldots, C_{|C|}\}$ are Pr_q, Rc_q, and F_{1q}*

supposed a clustering algorithm produces k = |C| clusters C' = { C_1', C_2', ..., $C_{|C|}$'} on the same data set D, this research intends to explore the combination of C and C' to yield an enhanced performance for both the categorization and clustering algorithms .

Further, to achieve **RO1**, a predefined set of labels (categories, classes) are needed when a text classifier is trained to predict a label for Web snippets. This leads to the third research objective:

RO3*: Using the semantic characteristics of the Open Directory Project categories to generate a set of labeled data sets. Using the generated data sets to evaluate Naïve Bayes, Adaboost, and kNN text categorization algorithms and K-Means and HAC clustering algorithms when combined with chi-square, Odds Ratio, Mutual Information, Information Gain, Relevancy Score, and GSS coefficient feature selection algorithms.*

Before extracting the semantic characteristics of the ODP categories to generate the labeled data sets, the hierarchical information of the ODP taxonomy tree such as how many categories at the different level of the ODP taxonomy tree, how deep is the taxonomy tree, is need. Therefore, the fourth objective of the research is described as:

RO4*: Uncovering detailed information of the ODP taxonomy tree and presenting statistical information of the generated ODP categoryDocument sets.*

3.2 Research Methodology

The Association for Information Systems (http://home.aisnet.org/) classifies research methodologies into three categories: quantitative, qualitative and design research. In the field of information systems, quantitative positivist research (QPR) intends to answer "questions about the interaction of humans and artifacts such as computers, systems, and applications" (Straub, Gefen, and Boudreau 2005). For QPR, "the interpretation of the numbers is viewed as strong scientific evidence of how a phenomenon works." (Straub, Gefen, and Boudreau 2005) The epistemology of QPR is that objective reality of the world can be captured and translated into testable hypotheses in forms of statistical or numerical analyses. According to Straub, Gefen and Boudreau (2005), the purpose of empirical testing of a theory is to falsify it with data, not to verify the predication of the theory because with proper selection, almost any theory can be verified. In QPR, theories can never be shown to be correct, and should arise through deduction, rather than be based on observation. This falsification principle is the core of positivism (Straub, Gefen, and Boudreau 2005). The advantage of QPR is it utilizes statistical methods to objectively examine theories (Straub, Gefen, and Boudreau 2005); however, if the data collected in experiments are not good, the conclusion derived by QPR can be deceptive.

Avison and Myers (2005) indicate qualitative research is widely adopted in social sciences. They point out that in qualitative research there are three philosophically distinct research epistemologies: positivist, interpretive and critical; and four research methods: action, case study, ethnography and grounded theory. Qualitative research methods are independent of the underlying epistemologies. Researchers usually adopt one of these underlying philosophical assumptions otherwise it will be more difficult to defend. The strong interpretive aspects of qualitative research make it suitable for explaining social phenomena.

Design Research intends to understand, explain, and consequently, boost the behavior of aspects of information systems by analyzing the use of and performance of designed artifacts, such as algorithms (for example, information retrieval) and human/computer interfaces (Vaishnavi and Kuechler 2007). Design means creating artifacts that do not exist naturally. The design activities are categorized into a science of the artificial (also known as design science) which "is a body of knowledge about artificial (man made) objects and phenomena designed to meet certain desired goals" (Vaishnavi and Kuechler 2007). Design Research is sometimes called improvement research, i.e. research which emphasizes the problem-solving/performance-improving nature of the activity. The iterative circumscription process of problem-awareness/problem-solving/performance-improving is a formal logical method[11], and can contribute "knowledge to the understanding of the always-incomplete-theories that abductively motivated the original design" (Vaishnavi and Kuechler 2007). "knowledge is used to create works, and works are evaluated to build knowledge" (Owen 1998). Venable's model (Venable 2006a, 2006b) emphasizes the role of theory building, as well as activities for evaluating solution technologies in either positivist or interpretivist perspectives.

The Design Research method is employed in this research. The ultimate purpose of this research is to improve the relevance of Web search results. To approach this goal, - an intelligent search browser (a human-machine interface with information retrieval algorithms) is to be developed to understand, estimate, and refine the behavior of aspects (in terms of *precision*) of the text categorization and clustering algorithms so far developed in the context where only information snippets are available. From the perspective of the epistemology of design research – "knowing from making: objectively constrained construction within a context" (Vaishnavi and Kuechler 2007), and compared with the other two research methods, design research method fits naturally into this research.

Vaishnavi and Kuechler (2007) suggest that the design research process is generally organized into five iterative stages: awareness of problem, suggestions, development, evaluation and conclusions. The corresponding outputs of each of these five stages are: proposal, tentative design, artifact, performance measures and results. Following this

[11] Formal logic is the study of inference with purely formal content, where that content is made explicit. http://en.wikipedia.org/wiki/Logic, retrieved on 27 November, 2007.

research process steps and with consideration of research objectives, this research is also to be conducted in five stages as elaborated below.

Research stage 1 (RS1): Awareness of issues in Web search; Literature review to find research gaps

An observation of search results returned from search engines such as Google, Yahoo!, Clusty.com and others suggests that search engines are far from perfect (Zhu and Dreher 2008b). Literature review at this research stage mainly focuses on the topics of information retrieval models and evaluation, search engines, machine learning, Web data mining, text categorization/clustering, feature selection, personalization, and concept drift. These topics deal with organizing huge amounts of information effectively and efficiently, and improving the relevance of Web search results.

Research stage 2 (RS2): Suggestion of Recommender Intelligent Browser (RIB) which addresses the research questions of this study.

The aim of stage 2 is to suggest a conceptual model which addresses the research questions. Details of this stage are presented in the following sections.

Research Stage 3 (RS3): Implementation of *RIB*

The task at this research stage is to implement the *RIB*. The programming language used in the project is Java; the Integrated Development Environment (IDE) is NetBeans of Sun Microsystems. The main reasons for selecting NetBeans as the IDE are 1) it is an open source integrated project, 2) it supports multi-programming languages; and 3) it is free.

Because only a prototype of *RIB* is to be developed, a linear sequential software process model is to be adopted (Pressman 2001). However, the design and development process of *RIB* are Object-Oriented. Machine learning algorithms will be encapsulated in separate classes. Object-Oriented implementation enables code reuse (for example, cosine similarity class can be used in VSM, kNN and K-Mean), and enables *RIB* easy to maintain.

Research Stage 4 (RS4): Evaluation of categorization/clustering algorithms and the performance of *RIB*

Measures: The performance of an information retrieval system is usually evaluated by *recall* and *precision*. For search engines, the results are also to be evaluated by TREC[12] style *precision* and *recall* (Voorhees 2005a, 2005b). To evaluate categorization results, IR style

[12] The Text REtrieval Conference (TREC) was started in 1992. Its purpose was to support research within the information retrieval community by providing the infrastructure necessary for large-scale evaluation of text retrieval methodologies. http://trec.nist.gov

recall and *precision*, including *accuracy* and F_1 are widely accepted (Sebastiani 2002a; Yang 1999; Yang and Pedersen 1997).

Search-term selection: Thirty search-terms with ambiguous meanings provided by Zeng et al. (2004) to evaluate their learning algorithm will be selected and submitted to the meta-search engine.

Evaluation: search results obtained from *Yahoo! Search Web Services API* and the results of RIB are compared in terms of *precision*, *recall* and F_1.

RIB will also generate a series of labeled document sets by means of extracting semantic characteristics of ODP categories. The generated document sets will be used as benchmark documents to evaluate text categorization algorithms implemented in RIB such as AdaBoost, kNN, Naïve Bayes, Statistical Language Model, and feature selection algorithms chi-square, information gain, mutual information, Odds Ratio, GSS coefficient, and NGL.

RIB will also evaluate the effectiveness of a novel R^{2}Cut thresholding strategy proposed in this research, and a new version of tf-idf, the Z-tfidf.

Research Stage 5 (RS5): Data analysis and conclusion

This research stage will analyse the experimental data collected in stage 4. Based on data analysis results, conclusions are to be drawn, and further work is to be suggested.

Table 3-1 demonstrates the relationship among research stages, purposes of the stages, methods applied, outcomes, and corresponding chapters in this thesis. Note that RO4 is achieved first because it is a prerequisite of the RO3. Similarly, RO2 and RO3 are prerequisites of RO1 and the achievement of RO1 relies on both of them.

Table 3-1 Research stages and outcomes

Stage	**Purpose**	**Method**	**Outcome**	**Research Objective and chapter**
RS1	Aware of issues	Literature review	Determine research questions and research objectives	Chapter 1, Chapter 2 and part of Chapter 3.
RS2	Suggestion	Algorithm design, software design	*boostingUp*, R^{2}Cut, Z-tfidf, schematic overview of RIB	Chapter 3
RS3	Implement RIB and proposed algorithms	Programming, coding in Java	List of text categorization and clustering algorithms; *boostingUp*, R^{2}Cut, Z-tfidf, RIB	RO4 - Chapter 4
RS4	Evaluation	Quantitative comparision by using effectiveness evaluation criteria	Verification of *boostingUp*, R^{2}Cut, Z-tfidf algorithms and RIB	RO3 – Chapter 5, RO2 – Chapter 6, RO1 – Chapter 7.
RS5	Data analysis & Conclusion	Statistical	Conclusions of this research	Part of Chapter 4 to 7, Chapter 8.

3.3 RIB – Recommender Intelligent Browser

The proposed solution to address the challenges discussed in Chapter 1 is RIB – Recommender Intelligent Browser (Zhu and Dreher 2008a), as illustrated in Figure 3-2. The design goal of RIB is to improve the relevance of Web search results by Web snippets categorization, clustering, and personalization. Then, experiments are to be carried out to verify to what degree RIB can provide a user better search experience in terms of precision, recall, and F_1.

RIB intends to provide and compare the following three collections of Web snippets/search results:

1) the Web snippets directly returned from a meta-search engine;
2) the personalized Web snippets based on automatically created user profile; and
3) the re-ranked categorized Web snippets of RIB based on user interaction.

RIB is also able to check to what degree the combination of categorization and clustering boosts the performance (in terms of *recall*, *precision*, and F_1) of Web snippet categorization. The categorized results by a trained text classifier such as k-Nearest Neighbours (Mitchell 1997), AdaBoost (Schapire and Singer 2000), and Naïve Bayesian (Lewis 1998), will be combined with the results clustered by K-Means, and hierarchical clustering algorithms such as HAC (Jain and Dubes 1988). Obviously, *RIB* is not going to simply put all the algorithms together that will do nothing better except dramatically increase the computational complexity. The algorithms are mentioned here because one purpose of this research is to

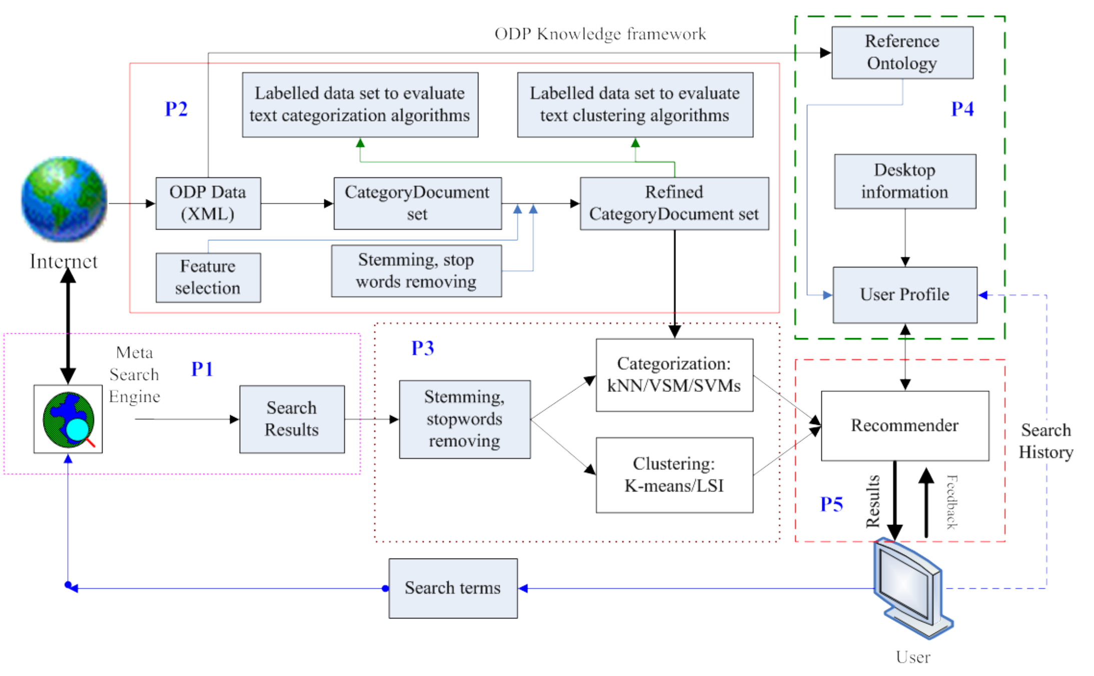


Figure 3-2 The architecture of Recommender Intelligent Browser

evaluate the effectiveness of the algorithms for the short Web snippets, which are not as informative as full text documents for which the algorithms have been developed and evaluated extensively. In addition, documents in the generated *categoryDocument* sets in Chapter 4 have different lengths; the performance of the text categorization algorithms when different length documents are used as training data set will also be compared.

3.3.1 Components of RIB

RIB is constructed with the following five main components which are introduced in details in the following subsections.

3.3.1.1 Meta Search Engine

A meta-search engine obtains search results from multiple existing search engines (Meng, Yu, and Liu 2002). The Meta-search engine in RIB (**P1**) is simplified to obtain 50 search results for a query directly from *Yahoo! Search Web Services API* after an application ID is applied. *Yahoo! Search Web Services API* allows developers to retrieve from Yahoo's Web databases directly by providing application interfaces for a list of programming languages such as Java, C++, PHP, Visual Basic etc. In this research, Java is selected to access Yahoo!'s Web search database. For non-commercial licenses, the maximum number of results per query from Yahoo! is 100. Take into account the factor that relevance judgments have to be conducted by human experts; and as stated by Jansen (2006), users are viewing few results pages, and more than half user browse only the first page. Therefore, in this research, top 50 results retrieved from *Yahoo! Search Web Services API* are collected which are then categorized into different ODP categories.

3.3.1.2 The *CategoryDocument* Extraction and Feature Selection

This component (**P2**) extracts and analyzes semantic characteristics of ODP categories to generate a series set of labeled *categoryDocuments* which are used as a benchmark document collections to evaluate text categorization algorithms and to re-organize Web search results from the meta-search engine. The *categoryDocument* sets are created based on two Resource Description Framework (rdf or RDF) files, *structure.rdf* and *content.rdf* that are available to download from ODP under the Open Directory License[13]. RDF is a family of W3C specification designed as a standard data model for data exchange on the Web. As suggested by Zhu and Dreher (2009), data in the two rdf files are extracted and analyzed by using SAX (Simple API for XML) and JAXP (Java API for XML Processing). Data in *content.rdf* include webpages submitted to the ODP categories and the descriptions of the webpages;

[13] http://www.dmoz.org/license.html

data in *structure.rdf* contains information of the ODP knowledge hierarchy which is also taken as a reference ontology to represent users' search preferences to personalize Web search results (Zhu and Dreher 2009). Chapter 4 will discuss how to use *cut&combine* approach to produce a series of labeled ODP *categoryDocument* sets with different statistical information.

The generated *categoryDocument* sets are actually human labeled document sets which can be applied in a range of applications such as to evaluate text categorization/clustering algorithms, and to re-organize Web search results into different ODP categories to provide users a better search experience. Before the *categoryDocument* sets extracted from the ODP can be used in the applications, the data sets need to be refined by one of the following feature selection algorithms that have been implemented in RIB, chi-square (χ^2), Mutual Information, Odds Ratio, Information Gain, Relevancy Score, NGL coefficient, and GSS Co-efficient (Sebastiani 2002b) for the purpose of efficiency and effectiveness (Yang and Pedersen 1997). A pre-processing step, which first removes stop words based on a stop words list provided by the Snowball[14] string processing language and then stems the words in the *categoryDocuments* by using Porter's stemming algorithm (Porter 1980), is conducted prior to feature selection.

The refined *categoryDocument* sets are utilized as a human labeled benchmark document collection to evaluate the following text categorization algorithms, Adaboost.MHR, Naïve Bayes, k-Nearest Neighbours (kNN). Chi-square, Mutual Information, Information Gain, and Odds Ratio feature selection algorithms are applied to select 10,000, 5,000, 3,000, 2,000, 1,000, 500, 300, 200, 100, 80, 50 features when the text categorization algorithms are evaluated by using the ODP *categoryDocument* sets. Experimental results are to be presented in Chapter 5.

3.3.1.3 Categorization/Clustering of Web Snippet

The component **P3** uses the generated ODP *categoryDocuments* as a training data set to train a list of text classifiers to categorize Web snippets obtained from the meta-search engine; at the same time, the Web snippets are grouped into different clusters by using K-Means clustering algorithms. The clustered results are then utilized to improve the categorized results by a proposed *boostingUp* approach. Section 3.2.2 will discuss how the clustered results improve the categorized results.

[14] Snowball is a small string processing language designed for creating stemming algorithms for use in information retrieval (http://snowball.tartarus.org).

Categorization algorithms implemented in RIB include Vector Space Model based k Nearest Neighbours (Larose 2005; Mitchell 1997), Support Vector Machine Light (Burges 1998; Joachims 1998, 2008), BoosTexter (Schapire and Singer 2000), Statistical language model (Kurland and Lee 2009; Ponte and Croft 1998; Zhai 2008), and Naïve Bayes (Lewis 1998; Manning, Raghavan, and Schütze 2008). Clustering algorithms include partitional based K-Means, and hierarchical clustering algorithm HAC (Húsek et al. 2007; Jain, Murty, and Flynn 1999; Mirkin 2005). Max-Min and Anomalous Pattern (AP) algorithms are used to choose the number of K (Mirkin 2005). Detailed descriptions of the algorithms are presented in Appendix 5, how the algorithms are implemented is described in Chapters 5 and Chapter 6.

3.3.1.4 User Profile Creation

Component **P4** creates user profiles by extracting information stored in desktop computers, and subsequently updates the created profiles whenever a website is visited. Indexed information stored in a given list of directories in a desktop computer is extracted by a specially developed recursive algorithm. The extracted document set is used to initialize a user profile that uses the categories at the second level of the OPD Web directory as an ontology to express the user's search preference.

To initialize a user profile, a text classifier such as Naïve Bayes or Adaboost is trained by using a generated labeled *categoryDocument* set. The labels appeared in the training document set is the same as the ODP categories used to express the user's search preferences. Using the above trained Naïve Bayes or Adaboost text classifier, each of the indexed documents extracted from the user's personal computer is assigned a label, or categorized under a category in a user profile. The number of documents categorized under a given category in the user profile indicates the degree of interest of the user to the category, and thus can be used to weight the user's search preference to that category. This is based on the intuition that users usually collect and store interesting information rather than uninteresting information (Assumption 3).

When a webpage is visited, the created user profile is to be updated, and the time factor which may cause search concept drift is considered (Zhu and Dreher 2008c). The impact of concept drift will be a weighting factor which represents a user's search preferences (Zhu and Dreher 2008c). Let w_i be the weight of concept $\mathbf{c}_i$ in user profile, the width of slide time window is 400 recently visited webpages, t is the sequence number in the time window, then,

$$w_{i-updated} = u(t) \times w_i, \quad u(t) = \begin{cases} 0.95 & current - most\ current\ 200\ searches \\ 0.75 & recent - past\ 201 - 400\ searches \\ 0.30 & historical - searches\ earlier\ than\ 400 \end{cases}$$

The time window is used to “smooth” the dramatic concepts shift, for example, from jaguar car to jaguar flight. If a user always visits Web sites related to jaguar car, and only visited two times of webpage about jaguar flight, it is reasonable to rank the webpages related jaguar car higher than jaguar flight. However, Mihalkova and Mooney (2009) suggest short session information is better than long term history information in disambiguating queries terms.

3.3.1.5 Recommender

The component Recommender (**P5**) presents filtered results to a user based on the created user profiles. Search results returned from the meta-search engine are categorized into the ODP knowledge hierarchy. Suppose the Web snippets are categorized into a category $\mathbf{c_i}$ belonging to the set of categories used to express the user’s search preferences, and its corresponding category weight in the user profile is w_i (i = 1, 2, …|C|) where |C| is the number of categories in the user profile. According to the descending order of w_i, the search results are re-organized and recommended to users in the same order. Users can adjust the number of categories to be recommended.

The design and implementation of component **P5** is based on the implementation of the components **P1** to **P4**, it aims at reaching research objective **RO1** as discussed previously.

3.3.2 Combining Clustering to Boost Categorization - ***BoostingUp***

Given a set of text documents or Web snippets D = {$\mathbf{d}_1$, $\mathbf{d}_2$, …, $\mathbf{d}_n$}, text categorization and text clustering are two distinct sort of algorithms to re-organize the documents into groups according to some common properties shared among the documents. For text categorization, supposed that the predefined label (category) set is $\Omega = \{\varpi_1, \varpi_2, ... \varpi_{|\Omega|}\}$, |Ω| is number of elements in set Ω, documents in D are assigned to category set C = {C_1, C_2, ..., $C_{|C|}$} $\subseteq \Omega$ where |C|≤ | Ω| is the number of categories in C, $\cup_{i=1}^{|C|} C_i = C, C_i \cap C_j = \phi \ for\ i \neq j$, and a document is assigned one and only one category.

For each document $\mathbf{d_j} \in$ D, a similarity score between categories in C and $\mathbf{d_j}$ is estimated by a classifier, for example, when Naïve Bayes classifier (Chapter 2, Section 2.1.1) is employed, the estimated probability of $\mathbf{d_j}$ belong to category C_i can be simply taken as the similarity score. The estimated similarity scores can be arranged in a matrix

$$A = [a_{ij}]_{|C| \times n}$$

where i = 1, 2, …, |C|, j = 1, 2, … n (the number of documents in D), a_{ij} is estimated by

$$P(C=c_i \mid D=d_j) = P(C=c_i) \times \frac{P(D=d_j \mid C=c_i)}{P(D=d_j)}$$

A simple strategy to categorize document is to assign C_i to document $\mathbf{d_j}$ if $a_{ij} = \max(a_{1j}, a_{2j}, \ldots a_{|C|j})$.

For text clustering, let the generated clusters, or a partition of D, are C' = $\{C_1', C_2', \ldots C_K'\}$. $\cup_{i=1}^{K} C_i' = C', C_i' \cap C_j' = \phi \ for\ i \neq j$, K is the number of clusters. Supposed that one document is arranged into one and only cluster, since documents are represented as vectors, a centroid can be generated for each of the formed clusters,

$$\mathbf{c}_i = (\mathbf{d}_{i1} + \mathbf{d}_{i2} + \ldots + \mathbf{d}_{in_i}) / n_i$$

n_i is the number of documents in cluster C'_i.

Similar to text categorization, for a document $\mathbf{d_j} \in D$, the similarities between the document $\mathbf{d_j}$ and the K centroids can constitute a matrix

$$B = [b_{ij}]_{K \times n}$$

where i = 1, 2, …, K; j = 1, 2, … n. b_{ij} is the similarity score between document $\mathbf{d_j}$ and centroid $\mathbf{c}_i$. For example, b_{ij} can be simply estimated by the cosine value of the two vectors $\mathbf{d}_j = \{w_{1,j}, w_{2,j}, \ldots w_{m,j}\}^T$ and $\mathbf{c}_i = \{w'_{1,j}, w'_{2,j}, \ldots w'_{m,j}\}^T$ as

$$b_{ij} = sim(\mathbf{d_j}, \mathbf{c_i}) = \frac{\sum_{k=1\ldots m} w_{k,j} \times w'_{k,i}}{\sqrt{\sum_{k=1\ldots m} w_{k,j}^2 \times \sum_{k=1\ldots m} w'^{2}_{k,i}}}$$

$w_{i,j}$ and $w'_{i,j}$ are the tf-idf weights of term i in document $\mathbf{d}_j$ and centroid $\mathbf{c}_j$ respectively. The weights are calculated based on centroid set C' = $\{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K\}$. Document $\mathbf{d_j}$ is arranged into cluster C'_i if $b_{ij} = \max(b_{1j}, b_{2j}, \ldots b_{Kj})$.

C and C' are two partitions of the document set D, therefore, a contingency table can be produced to compare the two partitions C and C' (Table 3-2). In the table, $n_{ij} = | C'_i \cap C_j |$ is the number of documents that appear in both category C_i and cluster C'_j, or the size of the intersection of $C'_i \cap C_j$. $n_{imax} = \max\{n_{i1}, n_{i2}, \ldots n_{i|C|}\}$, $n_{imin} = \min\{n_{i1}, n_{i2}, \ldots n_{i|C|}\}$. n_{maxj} and n_{minj} can be obtained the similar way. Note that when calculate the minimum value of n_{ij}, zero is not taken into consideration, that is, $n_{imin} \geq 1$, $n_{minj} \geq 1$ for i = 1, 2, … K, j = 1, 2, … |C|. An empty intersection implies both categorization and clustering algorithms agree on that no documents should be arranged in the intersection.

Table 3-2 A contingency table for comparing two partitions of document set D

Partition	**C_1**	**C_2**	**…**	**$C_{\|C\|}$**	**Max**	**Min**
C'_1	n_{11}	n_{12}	…	$n_{1\|C\|}$	n_{1max}	n_{1min}
C'_2	n_{21}	n_{22}	…	$n_{2\|C\|}$	n_{2max}	n_{2min}
⋮	⋮	⋮		⋮		
C'_K	n_{K1}	n_{K2}	…	$n_{K\|C\|}$	n_{Kmax}	n_{Kmin}
Max	n_{max1}	n_{max2}		$n_{max\|C\|}$		
Min	n_{max1}	n_{max2}		$n_{max\|C\|}$		

In an ideal scenario, C = C' and thus K = |C|, that is, the documents are arranged into the same number of clusters and categories, and there is only one non-zero element in each row and column of the contingency table.

However, because of the imperfection of text categorization and clustering algorithms, C' may differ from C, and there are more than one elements have non-zero value of n_{ij}.

A small number of n_{ij} such as one indicates that there is only one document that appears in the intersection of set C'_i and C_j. The document $\mathbf{d}_s \in D$ in the intersection is consequently considered as a potential outlier[15] for C'_i and C_j. However, whether the document is an outlier for C'_i and C_j needs to be further verified by checking the values in matrix A and matrix B that present the similarity scores of document between the different categories, and between the different centroids of the generated clusters.

Because the document is assigned into the category C_j, in matrix A, $a_{sj} = \max(a_{1j}, a_{2j}, \dots a_{|C|j})$. Similarly, in matrix B, $b_{sj} = \max(b_{1j}, b_{2j}, \dots b_{Kj})$.

The proposed *boostingUp* algorithm looks up the i-th row of the contingency table to locate the maximum n_{ij} for j = 1, 2, …, |C|. If the p-th column in this row has the maximum value, *boostingUp* then checks the similarity scores a_{sp} in p-th column of matrix A. If the score is ranked as the second or third highest scores, the document is considered as an outlier of category C_i, and thus is removed from C_i to category C_p.

To decide whether the document $\mathbf{d}_s$ is an outlier of cluster C'_i, *boostingUp* checks the j-th column in the contingency table to obtain q from $n_{qj} = \max\{n_{1j}, n_{2j}, \dots, n_{Kj}\}$. It then looks up the i-th column in matrix B to check whether b_{si} is far away from the centroid $\mathbf{c}_i$ of C'_i, that is, if b_{si} is among the one third elements that are not as close or not as similar to $\mathbf{c}_i$ as the rest two thirds element in C'_i. If b_{si} is far away from $\mathbf{c}_i$, *boostingUp* further checks if b_{sq} is ranked as the second or third highest scores among b_{sk}, k = 1, 2, …, K, $k \neq i$. If it is, document $\mathbf{d}_s$ is

[15] "An outlying observation, or outlier, is one that appears to deviate markedly from other members of the sample in which it occurs" (Grubbs 1969). The term used in this research refers to a document that may be arranged an inappropriate label by a text categorization algorithm, or be arranged into an inappropriate cluster by a text clustering algorithm.

taken as an outlier of cluster C'$_i$, and thus is remove from C'$_i$ to C'$_q$. Otherwise, $\mathbf{d}_s$ is not regarded as an outlier and will be kept in C'$_i$.

BoostingUp repeats the above process until there are no n_{ij} equals to one. The algorithm is presented in Figure 3-3. The design and implementation of *boostingUp* algorithms aims to achieve research objective **RO2** as discussed in Section 3.1.

3.3.3 Z-tfidf Term-weighting Strategy

3.3.3.1 tf-idf Term-weighting Strategy

There are two naïve yet essential ideas that dominant information retrieval systems, one of them is taking words as they stand, the other is counting their stances (Jones 2003). Salton and Buckley (1988) indicate there are two crucial factors for the enhancement of retrieval effectiveness: features (or terms) probably relevant to users information needs ought to be retrieved; and orthogonal features should be excluded. This is summarized as two monotonicity constraints (Witten, Moffat, and Bell 1999):

```
Check the contingency table CT
For i = 1, 2, … K;
    For j = 1, 2, … |C|;
        check if $n_{ij}$ equals 1;
        if $n_{ij}$ equals 1
            get the document $\mathbf{d}_s$ which lies in the intersection of C'$_i$ and $C_j$
            check the ith row in CT
            get the column p from $n_{ip} = \max\{n_{i1}, n_{i2}, \dots n_{i|C|}\}$
            get the row q from $n_{qj} = \max\{n_{1j}, n_{2j}, \dots, n_{Kj}\}$

        check the $a_{sp}$ in A to decide if $\mathbf{d}_s$ is an outlier of $C_j$
        if $a_{sp}$ ranked the second or third highest value among
            $\{a_{s1}, a_{s2}, \dots a_{s|C|}\}$
            document $\mathbf{d}_s$ is taken as a outlier for category $C_j$. Remove $\mathbf{d}_s$
                from $C_j$ to $C_p$.

        check the $b_{si}$ in B to decide if $\mathbf{d}_s$ is an outlier of C'$_i$
        1.  check if $b_{si}$ is among one third of the elements in C'$_i$
             that are far away from the centroid of C'$_i$.
        if $b_{ai}$ is far away from the centroid,
            then check if $b_{sq}$ is ranked the second or third higest value among
                $\{b_{s1}, b_{s2}, \dots, b_{s|C|}\}$
            if it is, then $\mathbf{d}_s$ is taken as an outlier of cluster C'$_i$, remove $\mathbf{d}_s$ from
                 C'$_i$ to cluster C'$_q$.
    end For j
End for i
```

Figure 3-3 The boostingUp algorithm

A term that appears in many documents should not be regarded as being more important than a term that appears in a few, and a document with many occurrences of a term should not be regarded as being less important than a document that has just a few.

Let N be total number of documents, n_i be the number of documents in which term t_i appears, $freq_{i,j}$ be the frequency of term t_i in document $\mathbf{d}_j$, using $\max_l freq_{l,j}$, the maximum frequency of a term appeared in $\mathbf{d}_j$, as the normalization factor, term frequency is expressed as $tf_{i,j} = \frac{freq_{i,j}}{\max_l freq_{l,j}}$, inverse document frequency is estimated by $idf_i = \log(1+\frac{N}{n_i})$. The famous tf-idf term-weighting scheme is thus $w_{i,j} = tf_{i,j} \times idf_i$ (Baeza-Yates and Ribeiro-Neto 1999).

3.3.3.2 Z-tfidf Term-weighting Strategy

One weakness of tf-idf weighting strategy is that the term frequency (tf) is calculated only against a single document $\mathbf{d}_j$, and inverse document frequency (idf) concerns only whether the term occurs in other documents, not matter how many times the term appears. If the term occurs frequently in other training documents as well, that is, the weights of the term frequency in other training documents are also very high; the weight of the feature in the documents should be weighted less. Therefore, a z-tfidf term-weighting strategy is proposed to address the situation in which inverse document frequency does not help much because it involves only whether a term appears in other documents, the frequency of the term in these documents are not taken into consideration.

Let $f_{i,j}$ is the term frequency of the i-th document and j-th feature, μ_j is the average of all $f_{i,j}$ for feature j, $\sigma_j = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(f_{i,j} - \mu_j)^2}$ is the standard deviation. Standardizing $f_{i,j}$ to relief the document length effects which might put long documents in advantages,

$$z_{i,j} = \frac{f_{i,j} - \mu_j}{\sigma_j}$$

With $z_{i,j}$, the z-term frequency of j-th feature in i-th document is estimated by

$$tf_{i,j} = \frac{z_{i,j}}{1 + \sum_{k=1,\ldots N, k\neq i} w_{k,j}}$$

where N is the number of document in the information retrieval system. Finally, z-tfidf is calculated by $w_{i,j} = tf_{i,j} \times idf_i$ where $idf_i = \log(1+\frac{N}{n_i})$ which is the same as in tf-idf.

3.4 Effectiveness Evaluation of RIB

The effectiveness of RIB is evaluated by different steps that include evaluation of text categorization algorithms based on the generated *categoryDocument* sets, evaluation of the *boostingUp* algorithm, evaluation of categorized results of RIB, and evaluation of the recommender of RIB.

3.4.1 Evaluation of Text Categorization Algorithms

The first step involves using RIB to extract the semantic characteristics of the ODP categories to generate a series of *categoryDocument* sets that can be used as benchmark data sets to evaluate text categorization algorithms such as Naïve Bayes, kNN, and Adaboost.MHR. The statistical information of the generated ODP *categoryDocument* sets is presented in Chapter 4. RIB can also be used to compare the performance of other text classifiers including SVM$^{\text{light}}$ (Joachims 2008) and Statistical Language Model (Kurland and Lee 2009; Zhai 2008) by using a small specially designed data set. Implementation details of the text classifiers and the experimental results are provided in Chapter 5.

The above algorithms are to be evaluated by using n-fold cross validation methods (Han and Kamber 2006), and in this research, n is five. That is, the data set D is randomly separated into five partitions D = {D_1, D_2, …, D_5},

$$D = \sum_{i=1}^{n} D_i ,\ D_i \cap D_j = \phi \ \text{for}\ i \neq j\ .$$

The evaluation is to be repeated five times, for each time, one of the D_i, i = 1, 2, … 5 is selected as a testing data set and the rest four partitions are used together to train an algorithm. The evaluation results presented are the averaged results against the five round experiments.

3.4.2 Evaluation of the Effectiveness of ***BoostingUp*** Algorithm

The second evaluation step is to verify to what degree the proposed *boostingUp* algorithm enhances the performance of Naïve Bayes text classifier by using K-Means (Jain 2009) text clustering algorithm. A set of 50 search results obtained from Google Directory (http://directory.google.com/) using "jaguar" as a search term is employed as a labeled Web snippet set. Each of the returned Web snippets has an ODP topic assigned by Google[16]. The

[16] Google is now using ODP as it Web directory, the categories assigned to the results of Google Directory should come from ODP. However, some of the categories assigned by Google are not included in ODP (Chapter 5, Section 5.4.1)

topic is taken as the truth topic for the Web snippet. After removing the topics assigned by Google, the Web snippet set is used as a testing data set.

Naïve Bayes classifiers are trained by the generated ODP *categoryDocument* set. The trained classifiers are used to predict a label (topic) for each of the testing Web snippet. The topic predict by Naïve Bayes classifier is then compared with the topic given by Google. F_1 is taken as the measure to evaluate the performance of the classifier.

The testing Web snippets are then grouped into clusters by using K-Means clustering algorithm. *boostingUp* algorithm is employed to combine the results of K-Means and Naïve Bayes. The combined results are evaluated by F_1 measure which is expected better than the results without using *boostingUp* algorithm.

The testing Web snippet set is also used to evaluate the text clustering algorithms such as K-Means and HAL in terms of Rand Index (Chapter 2, Section 2.2.5). Experimental results are discussed in Chapter 6.

3.4.3 Evaluation of Web Search Results of RIB

To evaluate how RIB improves the relevance of Web search results, a similar experiment as suggested by Zhu (2007) is designed and to be conducted. The main shortcoming of Zhu's experiment is that there are only five ambiguous search-terms were employed and thus the experimental results may not be statistical sound. To overcome the weakness, 25 out of 30 ambiguous search-terms from Zeng et al (2004) are chosen as a set of queries, because according to Buckley and Voorhees (2000), a good experiment needs 25 to 50 queries to produce desired level of confidence about experimental results. *Yahoo! Search Web Services API* is employed to obtain 50 search results for each of the search-terms. Similar to experiments conducted by Zhu (2007), for each of the ambiguous search-terms, an information need is clearly defined (refer to Appendix 4 for the 30 search-terms and the specified information needs). Human judges are employed to make a decision if a returned Web snippet is relevant (R), partial relevant (P), irrelevant (I) to the specified information need, or not sufficient information to make a judgment (N). For each of the $25\times50 = 1250$ returned Web snippets, relevance judgment results from five different judges are collected.

A final binary relevance judgment result is made for each of the Web snippets by assigning a numeric value to each of the four possible judgment results, and then combining the values together to reach a final score. As suggested by Zhu (2007), the four possible judgments R, P, I and N are assigned value 3, 1, -3, and 0 respectively. If the sum of the five judges' decision is bigger than zero, the Web snippet is believed relevant to the specified information need; if the sum is less than zero, the Web snippet is taken as irrelevant. If the summed value equals zero, the result is re-checked by visiting the website linked by the snippet, and according to the topic of the website, a binary relevance judgment decision is made.

When the search-terms are submitted to RIB, the returned Web snippets are categorized into the 14 top-level ODP categories as well. Following Zhu's (2007) experiment, two categories which have the most relevant results are chosen as the top ranked results of RIB. The results are evaluated and compared with the results directly from Yahoo! in terms of precision, recall, P@10[17], and the interpolated precision-recall curve - the widely employed performance measurements of information retrieval systems and search engines. The re-ranked results of RIB can also be taken as the recommended results suppose that the two categories are also the choices of users'.

3.4.4 Evaluation of Recommender

RIB is able to recommend interest categories to a user based on the automatically generated user profile, which collects the user's search preferences by means of data stored in the user's personal computer and search history.

To evaluate how the recommender in RIB can provide a better user search experience, a set of simulated data is produced for an imaginary user. The simulated data is used to create a user profile, and supposed that the search preference of the user is related to two top-level ODP categories, one is about "Computers", and another is about "Sports". Given a test data set of Web snippets, RIB is to re-organize the snippets according to the generated user profile. The re-ranked results can be used to evaluate the performance of RIB.

Experimental results are to be presented in Chapter 7.

3.5 Summary

This chapter presented a brief overview of the research objectives, the proposed solution to address the challenges introduced in Chapter 1, the *boostingUp* algorithms, research methodology, and the design of experiments. The definitions of important concepts, assumptions made in this project, and research objectives were firstly provided. The research methodology, design research method employed in this research and the five research stages were presented as well. Components in RIB – Recommender Intelligent Browser were then discussed in details to demonstrate how the objective **RO1** is achieved. A novel *boostingUp* algorithm that combines text categorization and clustering to boost the performance of text

[17] P@10 is the precision of the top ten returned results of an information retrieval system which returns a ranked list of results. Refer to Chapter 7, Section 7.2.3.

categorization was introduced in details to approach the research objective **RO2**. The design of a series of experiments that evaluate the effectiveness of RIB was introduced and discussed with the aim to achieve **RO3**, **RO4** and **RO1**. The experimental results of RIB will be presented in the following chapters.

4 Labeled Data Set – The Open Directory Project

Text categorization, also referred to as classification, is the automatic assigning of predefined categories or labels to unlabeled documents (Yang 1999). This involves using a labeled data set to train a prediction model that then makes classification decisions. The size and the quality[18] of labeled data set are two essential factors that affect the performance of the trained prediction model. However, a labeled document set, especially an extremely large one, is generally very expensive to obtain because it involves much human labour to categorize the document set under the predefined label set. Fortunately, Open Directory Project (ODP), which is a collection of innumerable volunteers' efforts and intelligence on websites labeling, is a source of a free labeled document set that can be used to generate large-scale labeled data sets.

The ODP has been employed in a range of research and applications. For example, in the research of Dumais and Chen (2000), they use ODP data extracted from the first two level of ODP repository to organize Web content by means of Support Vector Machines (Appendix 7). Gauch, Chaffee and Pretschner (2003) use the fourth level of ODP data to re-arrange Web search results to personalize Web searching. Seng et al. categorize Web search results into ODP categories and then map the categories to another set of predefined concepts (Shen et al. 2006). A detailed review of the applications and research of ODP in the field of information systems is presented by Zhu (2008a) who discussed more than 30 papers that utilizing ODP for different purposes.

However, there are some questions to be considered before using the most comprehensive human-edited Web knowledge hierarchy. For instance, how the labeled data sets employed by the researchers are constructed, what kind of information is extracted to construct the data sets, what are the detailed steps and approaches employed in constructing the data sets? How many categories at the different levels of the ODP knowledge hierarchy? What are the average/maximum lengths of the documents in the training data sets? Why using the second level ODP data or the four level ODP data set but not the third level? There is lack of detailed information about these questions when the data are extracted from the different levels of the ODP *taxonomic tree* to serve as a labeled data set. This kind of information,

[18] Not all of labeled documents can be assigned an appropriate label because text categorization is inherently a subjective procedure that depends on individuals and other factors when relevance judgments are made (Mizzaro, 1997). The most widely used benchmark document collection, the Reuters Corpus, also contains some conflicting assignments (Lewis et al., 2003).

.

such as document length, if is known and can be exploited by a retrieval system, has the potential to improve retrieval performance (Liu and Croft 2003).

This chapter presents detailed statistical information about the ODP category, one of the most comprehensive human-edited Web knowledge hierarchies, and statistical information about the generated *categoryDocument* sets based on the semantic characteristics of ODP. After a brief introduction of the ODP in section 1, section 2 explains how the semantic characteristics are to be extracted for each of the categories in the OPD. Section 3 discusses how the ODP *categoryDocument* sets are constructed, how many are subcategories for each of the top-level ODP categories. This section also includes statistical information about the maximum and average length of the ODP *categoryDocuments*, and how the extracted semantic characteristics of the ODP categories are combined to construct a labeled data set used to evaluate text categorization and clustering algorithms that are to be discussed in the following chapters. The last section summarizes this chapter.

The information provided in this chapter is to help to the readers understand how semantic characteristics distributed among the hierarchical structure of the ODP *taxonomic tree* when using ODP as a labeled data set for different research and application purposes. To the best of our knowledge, this is a pioneering piece of research that not only demonstrates the hierarchical information of the ODP repository, but also presents how are the semantic characteristics of each of the ODP categories extracted. In addition, the chapter also discusses how to accumulate the semantic characteristics to form a *categoryDocument* set that is to be employed as a labeled data set for the purpose of evaluating text categorization and clustering algorithms, or for the purpose of re-organizing Web snippets. The new constructed ODP *categoryDocument* sets enrich the existing text categorization and cluster data sets such Reuters-21578 (Sebastiani 2002a; Yang 1999), MeSH categories (*Introduction to MeSH* 2007), UseNet data (Lang 1995), and so on; and provide some alternatives for the researchers in the fields of text mining, machine learning, information retrieval and other related disciples when labeled benchmark document collection is a necessary.

4.1 The Open Directory Project (ODP)

Spurred by the success of open source movement, Rich Skreta and Bob Truel set up the Open Directory Project[19] in June 1998 in response to the shortcomings of the Yahoo! Web Directory that is maintained by only a small group of editors (Sherman 2000). ODP is now the largest, most comprehensive human-edited directory of the Web, it has more than 84517 volunteer editors from all of the world, 590,000 categories, and more than 4.5 million human

[19] www.dmoz.org, or www.dmoz.com. ODP is also known as simply dmoz.

indexed websites that are submitted to ODP (as at 7th December 2009). It is also employed as a Web directory by search engines such as ABC.net, Google, Lyrics.com, Meta-search, and a number of others organizations[20].

4.1.1 The ODP Hierarchy

The ODP is organized as a *Directed Acyclic Graph* (Figure 4-1), which is formed by using symbolic links. "*A symbolic link is a hyperlink which makes a directed connection from a webpage along one path through a website to a page along another path*" (Perugini 2008). Symbolic links has the potential to help to assign a document to multiple labels, which is another research topic outside the scope of this thesis when analysing the structure of the ODP. Put aside the symbolic links, ODP is arranged as a tree, as shown in Figure 4-2 (Zhu 2007), and the tree is referred to as the ODP *taxonomic tree*. Similar to one of the definitions given by Randall (Randall 1976), the ODP *taxonomic tree* is defined as:

> *Starting from the unique beginner, the ODP root, there is a chain of* direct precedence *proceeding downward through other categories to a terminal (leaf) category.*

For instance, the ODP *taxonomic tree* includes category "Computers", "Computers" includes "Algorithms", and "Algorithms" includes a leaf category "Sorting and Searching".

From the root category, the ODP (or TOP), there are 15 first level categories. In addition to the 15 categories, category "World" supports the ODP in different languages. Figure 4-3 shows the 15 + 1 categories and some of their subcategories. In fact, ODP also contains a hidden category "Adult" which is not visible from its website. Therefore, there are in fact all together 17 top-level ODP categories.

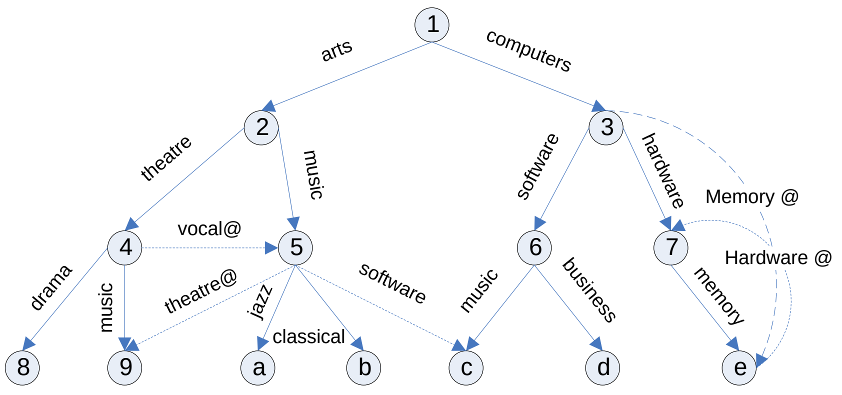


Figure 4-1 Sample Web directory, with characteristics similar to the ODP.

Nodes correspond to Web pages and directed edges correspond to hyperlinks between pages. Symbolic links are indicated by dashed edges and hyperlink labels ending with @ character (Perugini 2008)

[20] http://www.dmoz.org/Computers/Internet/Searching/Directories/Open_Directory_Project/Sites_Using_ODP_Data Note that the new Microsoft Bing, the world's third largest search engine, has not a Web directory at the moment.

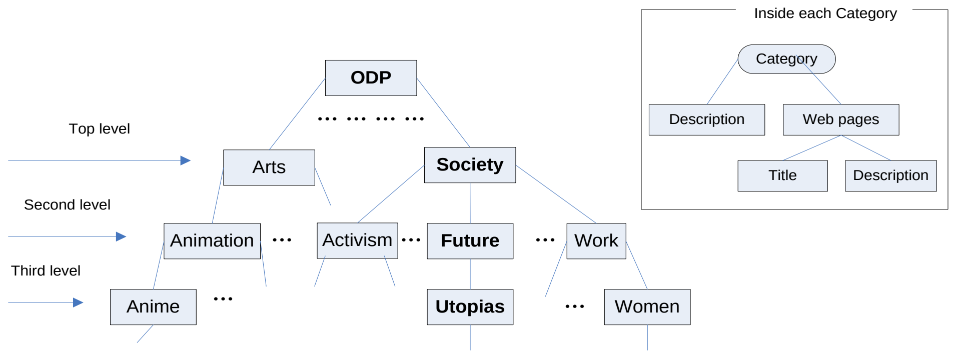
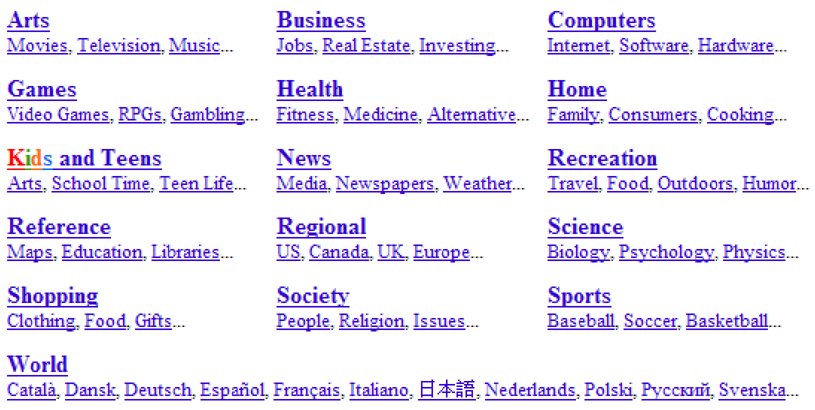


Figure 4-2 The ODP taxonomic tree, adaption of the hierarchical structure of the ODP (Zhu 2007)

Each of the fifteen first level ODP categories has its own subcategories. For example, under category "Arts", the immediately following subcategories are: …, "Movies"; …; "Television"; …; "Music"; … and so on. Subcategories under the category "Arts" may have their own sub-subcategories, and these sub-subcategories may in turn have their own subcategories, until such an up-down path reaches the end (leaf node) of the tree structure. The fifteen first level categories are also referred to as top-level categories; their immediate subcategories are referred to as second level categories; similarly, the immediate subcategories of the categories in the second level are referred to as the third level categories; so on and so forth, until the path reaches its leaf node. Refer to Figure 4-3 for an illustration of the *level* in the ODP *taxonomic tree*. In addition, the term subcategory or subcategories maybe instituted by category or categories in case no ambiguousness may occur.

Each of the categories in the ODP has a unique *identification*, the *topic* of the categories. For instance, "Top: Computers: Programming: Software Testing" is the *topic* of the category titled "Software Testing"; its direct supercategory is "Programming"; its first level supercategory (immediately after "Top") is "Computers".



Arts Movies, Television, Music...	**Business** Jobs, Real Estate, Investing...	**Computers** Internet, Software, Hardware...
Games Video Games, RPGs, Gambling...	**Health** Fitness, Medicine, Alternative...	**Home** Family, Consumers, Cooking...
Kids and Teens Arts, School Time, Teen Life...	**News** Media, Newspapers, Weather...	**Recreation** Travel, Food, Outdoors, Humor...
Reference Maps, Education, Libraries...	**Regional** US, Canada, UK, Europe...	**Science** Biology, Psychology, Physics...
Shopping Clothing, Food, Gifts...	**Society** People, Religion, Issues...	**Sports** Baseball, Soccer, Basketball...

World
Català, Dansk, Deutsch, Español, Français, Italiano, 日本語, Nederlands, Polski, Русский, Svenska...

Figure 4-3 The first 15 + 1 first level categories of ODP (www.domz.org)

4.1.2 Semantic Characteristics of the ODP and *Zero-level CategoryDocuments*

Each of the ODP categories contains

1) the *topic* of the category;
2) an informative *description* of what the category is about;
3) a list of human *indexed websites* (submitted websites), each of the websites has a descriptive title and a brief description of the website;
4) a list of its subcategories;
5) a list of category in other languages. The first three parts are all semantic characteristics of a given category, and thus can be extracted and used to manifest the category, as suggested by Zhu (2007), Zhu and Dreher (2009).

A *categoryDocument* which is composed of the three parts mentioned above is constructed for each of the ODP categories. That is, a *categoryDocument* is a combination of:

1) The topic of the category;
2) The description of category;
3) A list of indexed websites: title of each of the sites, informative description of each of the sites.[21]

The above information can be extracted from two *rdf*[22] format files, *content.rdf* and *structure.rdf*, available to download from the webpage of the ODP. Usage of the data is under the Open Directory License[23]. Zhu and Dreher (2009) provide a detailed description and analysis of the two *rdf* files, how the *categoryDocument* set is constructed, and some implementation details.

The generated *categoryDocument* set is referred to as *zero-level categoryDocument* set because during the construction of the document set, no hierarchical elements in the ODP *taxonomic tree* are involved.

A crucial assumption made in this research is that the semantics of each of the ODP categories can be represented by a *categoryDocument* that is composed of the semantic characteristics of a given category (Assumption 2 in Chapter 3). Further, the *categoryDocument* can contain the semantic characteristics acquired from its subcategories. The assumption is based on the fact that the ODP knowledge structure is a top-down

[21] A best category for a website should be identified by the creator of the website before the site is submitted to ODP (www.domz.org/add.html). Therefore, some of the websites are indexed at the leaf-category and others are not.

[22] Rdf: Resource Description Framework, is a family of World Wide Web Consortium (W3C) specifications designed as a standard data model for data exchange on the Web.

[23] http://www.dmoz.org/license.html

organized tree (a taxonomic hierarchy), and if subcategories SC_{ij} (j = 1, 2, …) are classified under a given category C_i, the semantic characteristics of the subcategories should be naturally a part of the semantics of the category C_i. In other words, semantic characteristics of subcategories SC_{ij} (j = 1, 2, …) constitute a subset of the semantic characteristics of the given category C_i. This assumption leads to the introduction of the concept of *semantic-granularity* to be discussed in Section 4.2.1.

The constructed *categoryDocument* for each of the ODP categories is in fact a representation of the semantics of the category. Each of the *categoryDocuments* can be taken as a <topic, document> tuple, where the topic is the unique *identification* of the *categoryDocument*, that is, the whole path from root of the ODP to the specific category; and the *document* is composed of the three parts described above. ODP has approximately three quarters of a million categories counting all levels and consequently there is the same number of *categoryDocuments*, if no further processes such as combining a group of subcategories to form a new one are conducted on the categories. With the generated *categoryDocuments*, it is a simple matter to use the *categoryDocument* set for text categorization and clustering purposes.

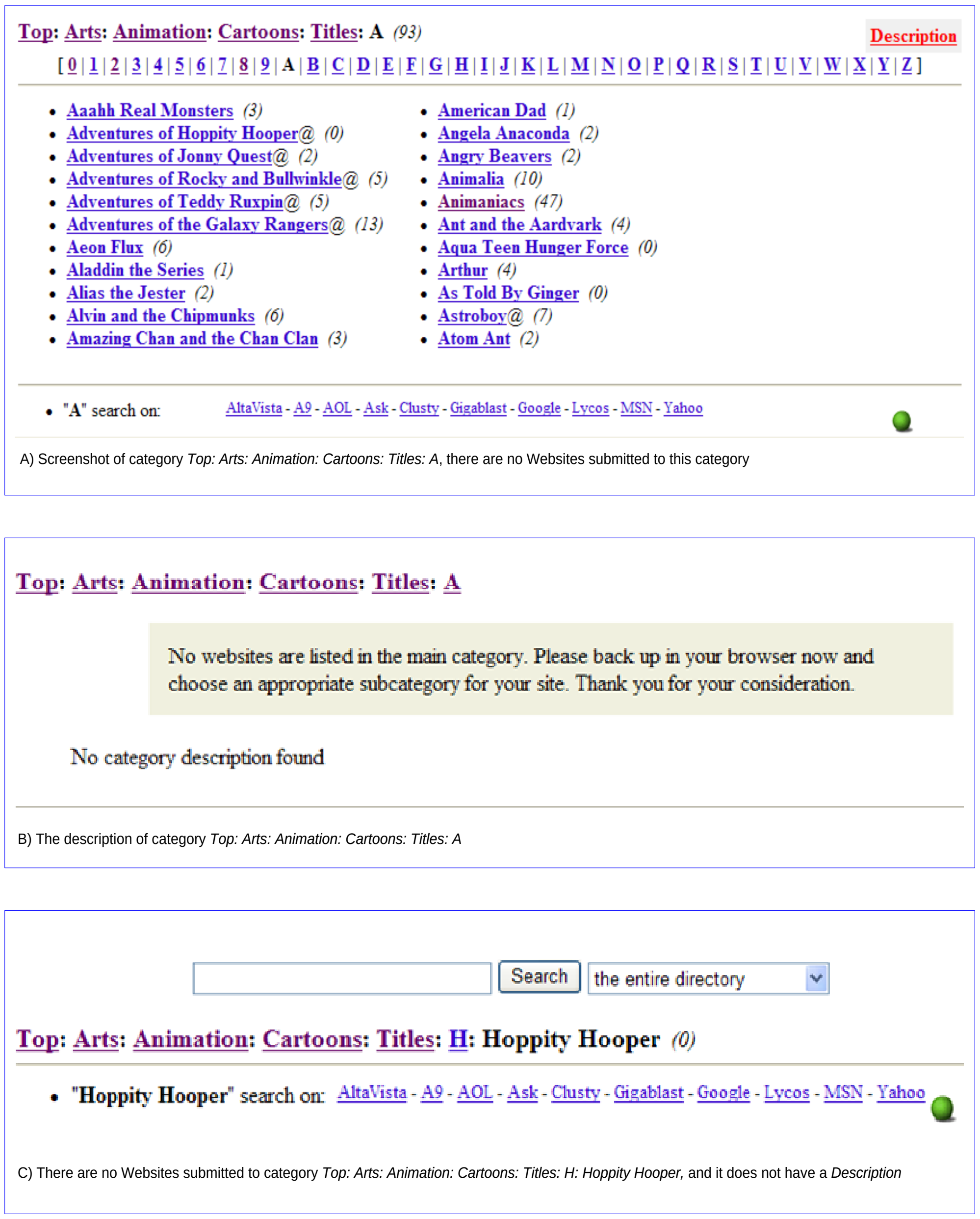
4.1.3 Using 14 but not all the 17 Top-level ODP Categories

In this research, for the entire 17 top-level ODP categories, three of them are not taken into account when constructing a *categoryDocument* set. The first of these categories is "World" which is designed for multi-language purpose and contains many specific characteristics that are not readable for English, thus not possible to carry out syntactic and semantic analysis on its subcategories. Processing multi-language is out the scope of this research. The second ignored category is "Adult" because it is not visible by public from the homepage of ODP. The last category which is not taken into account is "Regional" for the reason that in practice, classifying a data item or a Web snippet under the category "Regional" provides almost no semantic clues to help a user to understand the data item or the content of a Web snippet besides that it is local to a region. This category is a good portal for Web navigation when a user intends to find some websites in a specific area geographically, but not a proper category for text categorization and clustering purpose (without considering geographic clustering); especially when the ODP data set is used to classify Web snippets to actually help Web users to improve their Web search experiences.

4.1.4 *Empty-CategoryDocuments*

Attention needs to be paid to the fact that some of the *categoryDocuments* are *empty-categoryDocuments*. If an ODP category does not contain "Description" part, or the

"Description" part contains only editorial information (Zhu and Dreher 2009), and there are no websites indexed under the category, the constructed *categoryDocument* will include only the topic of the category, then the *categoryDocument* is referred to as an *empty-categoryDocument*. For example, as illustrated in Figure 4-4, the *categoryDocument* for "Top: Arts: Animation: Cartoons: Title: A" is an *empty-categoryDocument* because there are no websites indexed under the category (part A) of this figure), and although the category has a "Description" page, however, the page includes only editorial information which instructs not to submit websites to this category (part B) of this figure), and thus nothing



Top: **Arts**: **Animation**: **Cartoons**: **Titles**: **A** *(93)* **Description**

[0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z]

- Aaahh Real Monsters *(3)*
- Adventures of Hoppity Hooper@ *(0)*
- Adventures of Jonny Quest@ *(2)*
- Adventures of Rocky and Bullwinkle@ *(5)*
- Adventures of Teddy Ruxpin@ *(5)*
- Adventures of the Galaxy Rangers@ *(13)*
- Aeon Flux *(6)*
- Aladdin the Series *(1)*
- Alias the Jester *(2)*
- Alvin and the Chipmunks *(6)*
- Amazing Chan and the Chan Clan *(3)*
- American Dad *(1)*
- Angela Anaconda *(2)*
- Angry Beavers *(2)*
- Animalia *(10)*
- Animaniacs *(47)*
- Ant and the Aardvark *(4)*
- Aqua Teen Hunger Force *(0)*
- Arthur *(4)*
- As Told By Ginger *(0)*
- Astroboy@ *(7)*
- Atom Ant *(2)*

- "A" search on: AltaVista - A9 - AOL - Ask - Clusty - Gigablast - Google - Lycos - MSN - Yahoo

A) Screenshot of category *Top: Arts: Animation: Cartoons: Titles: A*, there are no Websites submitted to this category

Top: **Arts**: **Animation**: **Cartoons**: **Titles**: **A**

No websites are listed in the main category. Please back up in your browser now and choose an appropriate subcategory for your site. Thank you for your consideration.

No category description found

B) The description of category *Top: Arts: Animation: Cartoons: Titles: A*

Top: **Arts**: **Animation**: **Cartoons**: **Titles**: **H**: **Hoppity Hooper** *(0)*

- "**Hoppity Hooper**" search on: AltaVista - A9 - AOL - Ask - Clusty - Gigablast - Google - Lycos - MSN - Yahoo

C) There are no Websites submitted to category *Top: Arts: Animation: Cartoons: Titles: H: Hoppity Hooper,* and it does not have a *Description*

Figure 4-4 Examples of emptyCategory

A) shows there are no websites submitted to category Top: Arts: Animation: Cartoons: Titles: A, B) is the Description of the category in A), C) illustrates there are no websites submitted to category Top: Arts: Animation: Cartoons: Titles: H: Hoppity Hopper, and the category does not have a Description.

related to the semantic characteristics of the category. Part C) in this figure demonstrates another example of *empty-categoryDocument*. Under the category "Top: Arts: Animation: Cartoons: Titles: H: Hoppity Hopper", there are no indexed websites, and the category simply does not have a related "Description" as in A). The generated *categoryDocuments* in these two scenarios are *empty-categoryDocument*. The corresponding category of an *empty-categoryDocument* is referred to as an *emptyCategory* accordingly.

In this research, *empty-categoryDocuments* are not taken into account when using ODP as a data set to evaluate the performance of text categorization, clustering and feature selection algorithms. An *emptyCategory* may cause missing values of training document sets and thus is an important character of a learning task (Kotsiantis 2007). Ignoring *emptyCategories* is assumed not impose negative impact on the *categoryDocument* set which serves as a training data set, because the *emptyCategory* contributes little to the semantic characteristics of the category, and the percentage of *emptyCategory* for all the 17 ODP top-level categories is 12.17%; and only 6.04% for the 14 ODP categories. Section 4.3 of this chapter provides detailed statistical information about the number of categories and *emptyCategories*.

Using this approach, a semantically rich *categoryDocument* set is generated by extracting semantic characteristics from the world's most comprehensive human editor Web directory. Furthermore, excluding the *empty-categoryDocuments* and take only the top 14 ODP categories into consideration will not affect the richness of the semantics of the generated *categoryDocument* sets.

4.2 Using ODP as Labeled Data Set

Different data sets are to be generated if the constructed *categoryDocument* set (original set) is to be further processed by *cutting* the ODP *taxonomic tree* at different levels. The data items in the generated data sets can then be said have different *semantic-granularities*. The original set, which has the finest semantic-granularity, can also be used directly without being processed from the perspective of the level of the ODP *taxonomic tree*.

Since each of the generated *categoryDocuments* is presented in a form of a <topic, document> tuple, the *topic* is actually the label of the *document*. Therefore, all the generated *categoryDocument* sets are labeled data sets.

4.2.1 *Semantic-granularities* of the Generated *CategoryDocument* Sets

The notion of *semantic-granularity* employed in this thesis is similar to that of semantic granularity proposed by Fonseca et al. (2002) that "*semantic granularity addresses the different levels of specification of an entity in the real world*". It is worth to discern *resolution* and *granularity* at first. *Resolution* emphasizes the "the amount of detail in a representation" whereas *granularity* "*refers to the cognitive aspects involved in selection of features*" (Fonseca et al. 2002) and it is this kind of granularity is referred to as "*semantic granularity*". With regard to the situation of the construction of ODP *categoryDocuments*, the focus is the human-edited knowledge hierarchy (cognitive aspects in constructing the knowledge hierarchy); therefore, the term *semantic* rather than *resolution* is chosen.

The notion of "*semantic granule*" suggested by Lin and Chiang (2006) is "*if n tokens have additional meaning, we say they form a semantic n-granule*". This definition emphasizes additional meaning of a group of tokens. At first glimpse this definition differs from the one suggested by Fonseca et al. (2002). However, if taken "n tokens" in Li and Chiang as the "different levels of specification", the two notions are in some way similar with each other in that both employ a group of words/tokens to manifest a "real world entity" or a concept, although the "entity" or concept is not explicitly appeared on Lin and Chiang's definition. A difference between the two definitions is that Fonseca et al. address also "different level" which implies a hierarchical structure which is hard inferred from Lin and Chiang's notion.

Albertoni et al. (Albertoni et al. 2006) argue that semantic granularity should be defined dynamically rather than statically, and they propose to utilize it for Semantic Web browsing purpose. With a set of information sources, a *quality* based on which the sources are organized, and an ontology to depict a data schema in which the sources and the *quality* are entities, a sequence of granularities $\mathbf{G} = <G_1, G_2, \dots G_n>$ is generated so that G_{i+1} provides a more detailed view on the information source than G_i does.

In this research, ODP *categoryDocuments* are to be used to evaluate the text categorization and clustering algorithms, and be used as a training data set to classify Web snippets. A generated *categoryDocument* $\mathbf{d}_j$ for an ODP topic, for example, a subcategory of "Computers", may contain only 10 terms. However, in an extreme case, a *categoryDocument* can also be generated for a top-level ODP category, for instance "Computers", by combining all the features in the *categoryDocuments* that are subcategories of "Computers". In this case, $\mathbf{d}_{\text{Computers}}$ contains all of the terms that are indexed under the subcategories about "Computers" in the ODP knowledge repository. Obviously, the two documents contain quite different volume of semantic characteristics that will affect the evaluated performance of text categorization / clustering algorithms, and the performance of Web snippet categorization. This leads to the definition of *semantic-granularity*, which is a factor to be considered in generating the different *categoryDocument* sets.

Semantic-granularity of an ODP topic is defined with respect to the different levels of the ODP *taxonomy tree* and quantified by the number of features to manifest the topic. Let the vocabulary set of a *categoryDocument* $\mathbf{d}_j$ corresponding to the ODP topic t_j at level m be S_{jm}.

Subject to feature selection algorithms employed, the *semantic-granularity* of topic t_j , or its corresponding *categoryDocument*, is defined as

$$\Gamma_{mt_j} = \log_{10} | S_{jm} |$$

where $|S_{jm}|$ is the number of features contained in the *categoryDocument*. With this definition, if the generated *categoryDocument* for the topic t_j contains 100 different features, the *semantic- granularity* of t_j is 2; whereas the *semantic-granularity* of “Computers” may be as bigger as 5.

4.2.2 Using Original *CategoryDocument* Set Directly

The generated *categoryDocument* set can be used directly to evaluate the performance of text categorization and clustering algorithms.

To directly use the original *categoryDocument* set, note that a *categoryDocument* is a <topic, document> tuple, where topic is a *path* from the root of the ODP knowledge hierarchy to a specific category. For example, “Top: Computers: Programming: Software Testing” is the topic of category “Software Testing”. Removing the word “Top” which is the same for all topics and considering only the top-level category, all category documents with the topic starting with “Computers” constitute a training document set for the category “Computers”. That is, each of the documents is an example of the category “Computers”. Since there are all together 8469 categories arranged under the category “Computers”, there is consequently the same number of training documents or training examples available for category “Computers”. Similarly, *categoryDocuments* generated for other top-level categories can be used as training data set for the categories.

One of the applications of the generated *categoryDocument* set is to employ it as a training data set by a text classifier such as kNN (Mitchell 1997) to classify Web snippets. kNN is a kind of lazy classifier which has no training session. It compares the similarities between a test data item with all the training data items, ranks the similarities in descending order, and then uses majority voting algorithm to choose an appropriate category. The data set can also be used to evaluate the performance of different text classifiers. Zhu (2007) uses a data set produced the similar way to classify Web search results by kNN and the experimental results are encouraging. The advantages of the approach are its simplicity, and each of the *categoryDocument* is not too big (refer to 4.3 for statistical information of ODP categories). However, because the *semantic-granularities* of these *categoryDocuments* are too fine, or they are not semantically rich, they are thus very sensitive to a specific training data item. To address the issue, in the following section, data sets are to be constituted by a so-called *cut&combine* approach.

4.2.3 Using *Cut&combine* Method Produces 2nd-Level *CategoryDocument* Set

There are 17 top-level ODP categories, each of them has its own subcategories. Table 4-1 shows a list of the number of subcategories of the 14/17 top-level ODP categories based on the ODP data downloaded on 3rd June 2009.

As discussed above, the *semantic-granularities* of the *categoryDocuments* generated in the previous section are too fine to construct a semantic richness data set. To overcome the shortcoming, a *cut&combine* approach is proposed which firstly obtains a set of categories by *cutting* from a given level at the ODP *taxonomic tree*. For example, if *cut* the ODP *taxonomic tree* immediately after the root node "Top", it will obtain a category set ODPRoot = $\{C_i \mid i$ = "Arts", "Business",…, "Sports"$\}$. For any of the obtained categories $C_i \in$ ODPRoot, it has its own subcategory set $SC_i = \{c_x \mid c_x$: starting with "Top: C_i:"$\}$. For instance, if i = "Computers", $SC_{Computers}$ is a *categoryDocument* set which contains 45 elements (refer to Table 4-1), and "Computers: Programming" is an example of such elements. The category set $SC_{Computers}$ is obtained by *cutting* the *taxonomic tree* at the second level for category "Computers". *Cut* from the second level of the ODP taxonomy tree will yield 527 second level ODP categories. This is the *cut* step of the *cut&combine* approach.

Obviously, the category "Computers: Programming" has its own subcategory set, and the topic of all the elements in the subcategory set starting with "Computers: Programming".

The next step of the *cut&combine* approach is to combine the document part in the <topic,

Table 4-1 Subcategories of each of the 17 top-level ODP categories

No.	**Category**	**Number of subcategories**
1	Arts	41
2	Business	48
3	Computers	45
4	Games	24
5	Health	37
6	Home	20
7	Kids and Teens	14
8	News	19
9	Recreation	32
10	References	23
11	Science	52
12	Shopping	35
13	Society	30
14	Sports	107
Sub-total		527
15	Adult	12
16	Regional	36
17	World	81
total		656

document> tuple of all *categoryDocuments* if their topic starting with "Computers: Programming". This will constitute a new *categoryDocument* with topic "Computers: Programming". That is, for all the <topic, document> tuples, if the topic starting with "Computers: Programming", their document parts are added up, and the newly constructed content will replace the document part of the tuple <"Computers: Programming", document>.

Following this approach, for category "Arts", 41 new *categoryDocuments* are generated which corresponding to the 41 subcategories under the top-level ODP category "Arts"; category "Business" has 48 generated *categoryDocuments*, "Computers" has 45 such new documents, and so on. Refer to Table 4-1 for the number of categories by *cutting* at the second level ODP *taxonomic tree*.

When different levels of ODP *categoryDocument* sets are concerned, a simple cut approach can be applied to the *zero-level categoryDocument* set to yield groups of *categoryDocuments*. The *cut* approach *cuts* the ODP *taxonomic tree* at a given level to obtain a list of subcategories SC_i. All the *categoryDocuments* produced in the previous section (Section 4.2.2) starting with a given topic $t_j \in SC_i$ are taken as the examples of the t_j. Taking the "Top: Computers" as an example, using the *cut* approach, there are will be 8469 instead of 45 *categoryDocuments*. The *cut&combine* approach, on the other hand, firstly *cuts* at a given level of the ODP *taxonomic tree*, and then *combines* a bundle of documents with a desired property - all the topic of the *categoryDocuments* sharing the same top-level category.

One of the advantages of using *cut&combine* approach is the number of documents can be reduced sharply. The *zero-level categoryDocument* set has totally 187170 elements for the 14 top-level ODP categories, or 765459 elements for all the 17 top-level categories. Using *cut&combine* approach, the numbers are now 527 and 656; the ratio of the size of data set before and after applying the *cut&combine* approach is 0.028% and 0.00088% respectively.

Another advantage is that it provides an enriched set of semantic characteristics for a given higher-level ODP category compared with the previous *cut* approach because it aggregates all the available semantic features to manifest a given category. It is expected to provide more satisfactory experimental results when employed to evaluate the performance of text categorization and clustering algorithms.

The disadvantages of the approach are that it takes time to generate a new *categoryDocument*, and the generated documents are very long which might mean they will be costly to process compared to some shorter testing documents if no proper feature selection processes are conducted. The time taken to produce the original *categoryDocument* set and new *categoryDocument* set for the level two, three, and four is shown in Table 4-2.

It can be seen from the table that the time taken to generate a new *categoryDocument* set for the 14 top-level ODP categories is acceptable[24].

The lengths of *categoryDocuments* generated are provided in Section 4.2.4. The length of a *categoryDocument* is defined as the number of tokens in that document. *Token* is defined as any character string separated by space, tab, and new line, carriage-return, and form-feed characters, similar to the definition in the Java programming language. The delimiters in the ASCII character set is " \t\n\r\f".[25] Since most text categorization and clustering algorithms usually employ feature selection algorithms and length normalization techniques, long documents are generally not an issue when served as training data item.

4.2.4 Using *Cut&combine* Approach for Different Levels

It is natural to apply the *cut&combine* approach to not only the top 14/17 ODP categories, but also to the third and fourth level ODP categories in the ODP *taxonomic tree*. *Cut* at the root of the ODP *taxonomic tree*, there are only 14/17 categories, when a more fine-grained *CategoryDocument* set is require, third level, or even fourth level ODP categories are alternatives. For instance, if considering top-level ODP categories, there are all together 527 *CategoryDocuments* for the 14 top ODP categories. If more fine-grained categories are required, the *cut&combine* can be applied to the third level ODP *taxonomic tree*, and this will provide 6185 *CategoryDocuments* which have more fined *semantic-granularities* than the *CategoryDocuments* produced at the second level ODP *taxonomic tree*.

Table 4-2 Time taken to produce categoryDocument set (hh:mm:ss)

No.	Category	Original set	2nd level	3rd level	4th level
1	Arts	00:03:54	00:12:26	00:04:33	00:00:33
2	Business	00:05:47	00:01:03	00:00:12	00:00:04
3	Computers	00:02:03	00:01:13	00:00:11	00:00:02
4	Games	00:00:49	00:03:30	00:00:12	00:00:02
5	Health	00:00:50	00:00:24	00:00:04	00:00:01
6	Home	00:00:16	00:00:07	00:00:01	00:00:01
7	Kids and Teens	00:00:29	00:00:35	00:00:07	00:00:01
8	News	00:00:08	00:00:01	00:00:01	00:00:01
9	Recreation	00:01:22	00:01:10	00:00:27	00:00:18
10	References	00:00:57	00:04:36	00:03:41	00:01:53
11	Science	00:01:04	00:02:36	00:01:07	00:00:21
12	Shopping	00:01:05	00:00:07	00:00:01	00:00:01
13	Society	00:04:51	00:10:53	00:06:09	00:03:00
14	Sports	00:01:35	00:39:33	00:00:17	00:00:08
Sum		00:25:49	00:39:33	00:17:06	00:06:27

[24] Computer used: HP Compaq dc7700, Inter® Core™ 2 Quad CUP Q9400 @ 2.66GHz, 1.97GHz, 3.49GB of RAM.

[25] http://java.sun.com/j2se/1.4.2/docs/api/java/util/StringTokenizer.html

To produce the third level *categoryDocument* sets, considering the category "Computers: Programming" as an example. Under this category, there are 26 subcategories[26]. Applying the *cut&combine* approach obtains 26 third level *categoryDocuments*, each of the documents is semantically related to the category "Computers: Programming", and therefore can be used as a training data item for this category.

There are all together 527 third level ODP categories, and 6185 generated *categoryDocuments* for these categories. The number of categories at the fourth level of the ODP *taxonomic tree* is 6185, and there are 25749 generated *categoryDocuments* for the categories at the fourth level (refer to Table 4-3).

The generated *categoryDocument* set for category "Computers: Programming" can also be used as training data set for its supercategory "Computers". These documents have more fined *semantic-granularities* than the *categoryDocument* set generated in the previous section, but have more coarse *semantic-granularities* than the original *categoryDocument* set.

The statistical information of the ODP categories at different levels is presented in the following section.

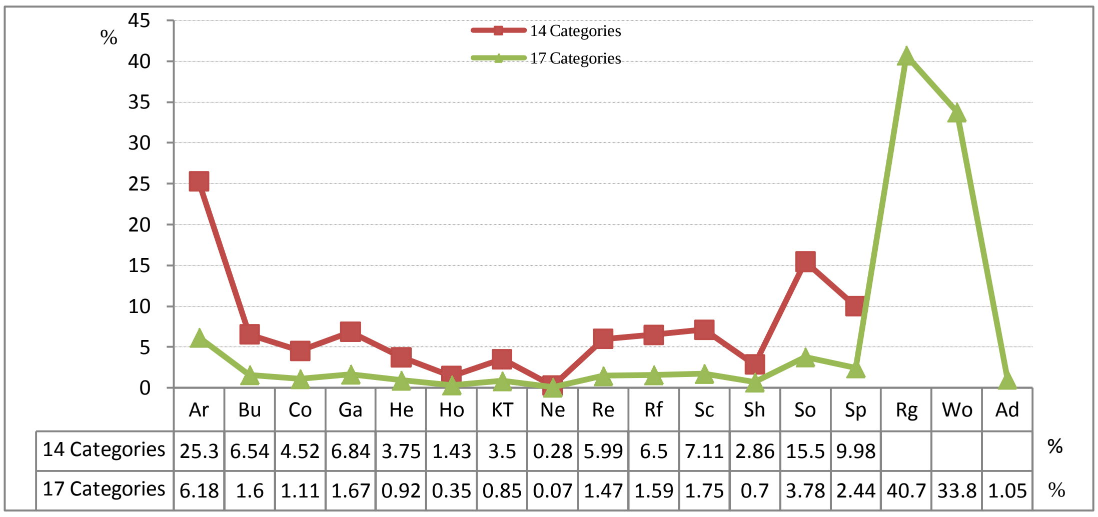
4.3 Statistical Information of ODP Data based on Level 2, 3 and 4

This section provides statistical information of the ODP categories based on the *content.rdf* and *structure.rdf* issued on 3rd June 2009. After unpacking, the size of the two files are 1,989,159KB, and 711,169KB, the compressed file size are 311,183KB and 74,304KB respectively. The statistical information presented in this section includes the number of categories and *emptyCategories* for the 14 top-level ODP categories; statistical distribution of the categories at the different levels of the ODP *taxonomic tree*; the depth of the ODP *taxonomic tree*, the maximum and average length of the *categoryDocuments* generated by *cut* and *cut&combine* approaches at different levels of the ODP *taxonomic tree*; and the number of words contained in the generated *categoryDocument* sets.

4.3.1 Number of Category and *EmptyCategory*

An overall view of percentages of the number of subcategories is presented in Figure 4-5. Figure 4-6 provides the number of categories and *emptyCategories* of ODP under the top 17 ODP categories. Figure 4-7 is the summary of categories and *emptyCategories* for the top 17 ODP categories at the different level of the ODP *taxonomic tree*. Note that the vertical

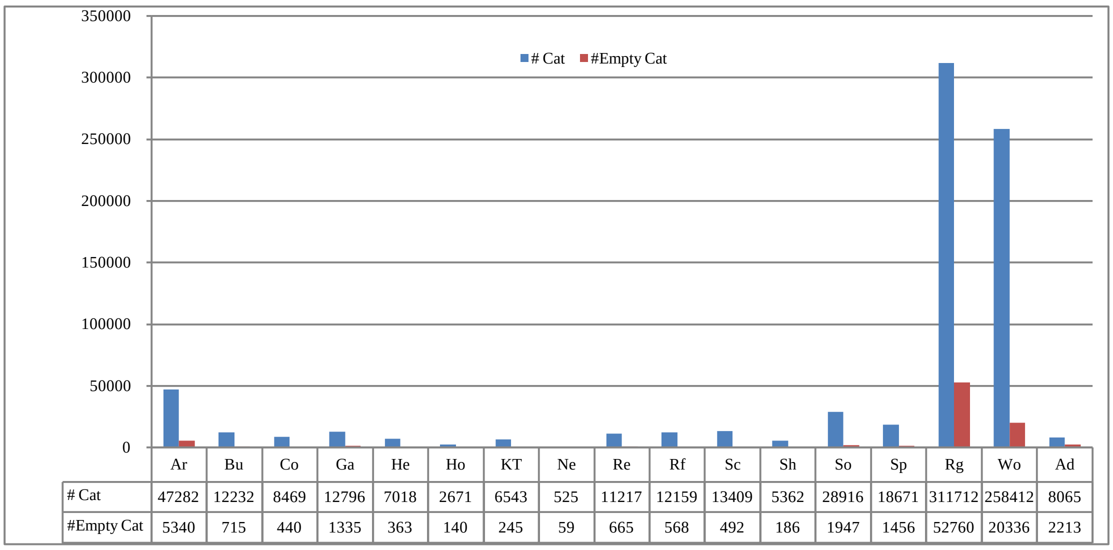
[26] Refer to http://www.dmoz.org/Computers/Programming/. Note that categories starting with character @ are symbolic links, and are not taken into account (retrieved: September 8, 2010).



	Ar	Bu	Co	Ga	He	Ho	KT	Ne	Re	Rf	Sc	Sh	So	Sp	Rg	Wo	Ad	
14 Categories	25.3	6.54	4.52	6.84	3.75	1.43	3.5	0.28	5.99	6.5	7.11	2.86	15.5	9.98				%
17 Categories	6.18	1.6	1.11	1.67	0.92	0.35	0.85	0.07	1.47	1.59	1.75	0.7	3.78	2.44	40.7	33.8	1.05	%

Figure 4-5 Percentages of number of categories in the 14/17 top ODP categories

axes presents the natural logarithm of the number of categories, that is, ln(number of categories). Detailed information of the number of categories and *emptyCategories* for each of the top 17 ODP categories at the different levels of the ODP *taxonomic tree* is provided in Appendix 3 from Appendix Figure 3-1 to Appendix Figure 3-15. Figure 4-8 presents the total number of categories and *emptyCategories* for the 17 top-level ODP categories at the different levels in the ODP *taxonomic tree*. Figure 4-9 is similar to Figure 4-8 except that it excludes category "Adult", "Regional", and "World" which are not involved in constructing the OPD *categoryDocument* sets.



	Ar	Bu	Co	Ga	He	Ho	KT	Ne	Re	Rf	Sc	Sh	So	Sp	Rg	Wo	Ad
# Cat	47282	12232	8469	12796	7018	2671	6543	525	11217	12159	13409	5362	28916	18671	311712	258412	8065
#Empty Cat	5340	715	440	1335	363	140	245	59	665	568	492	186	1947	1456	52760	20336	2213

Figure 4-6 Number of categories and *emptyCategories* in the 17 top ODP categories.

Ar: Arts; Bu: Business; Co: Computers; Ga: Games; He: Health; Ho: Home; KT: Kids & Teens; Ne: News; Re: Recreation; Rf: Reference; Sc: Science; Sh: Shopping; So: Society; Sp: Sports; Rg: Regional; Wo: World; Ad: Adult, The same abbreviation is also used in the following figures

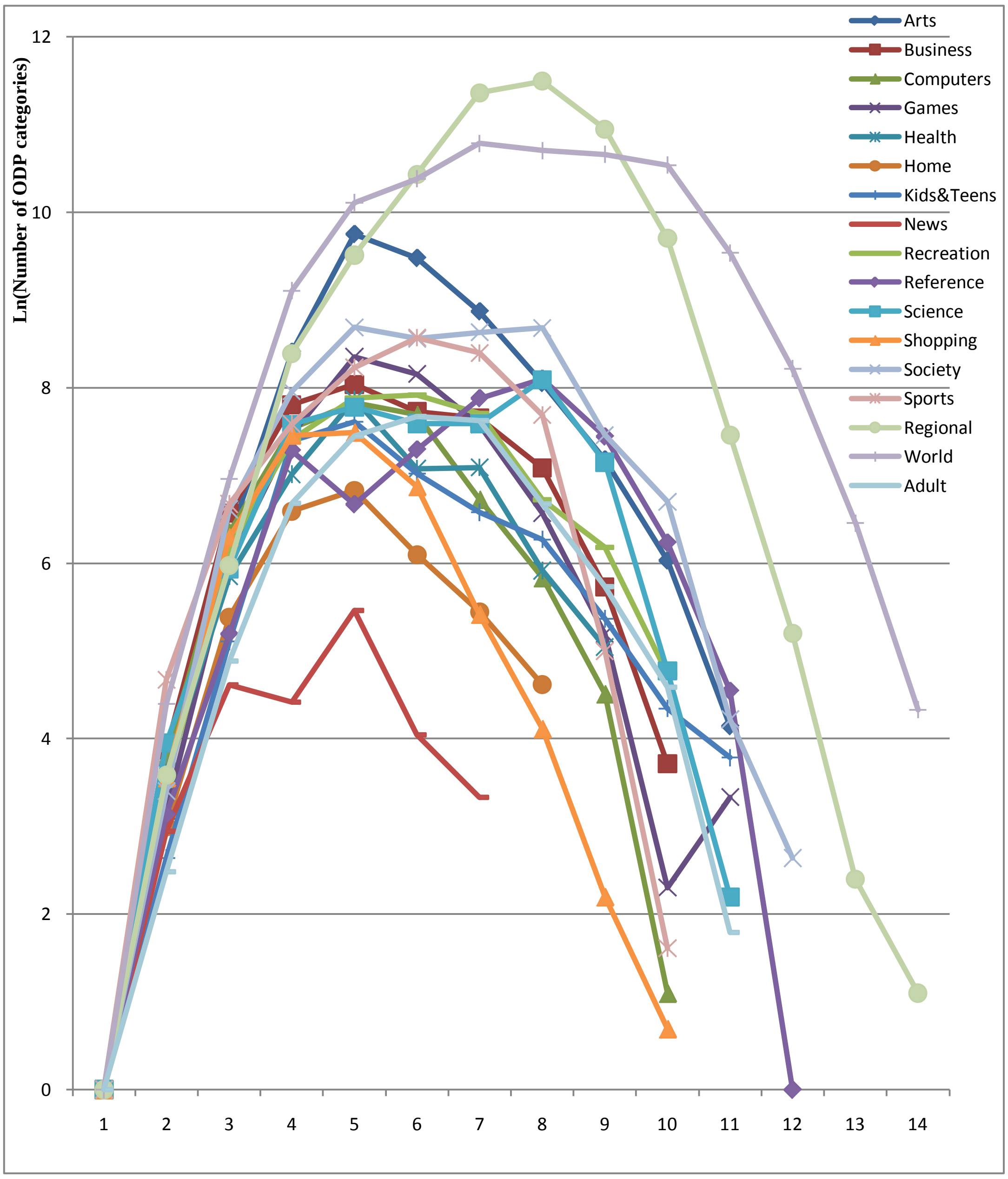
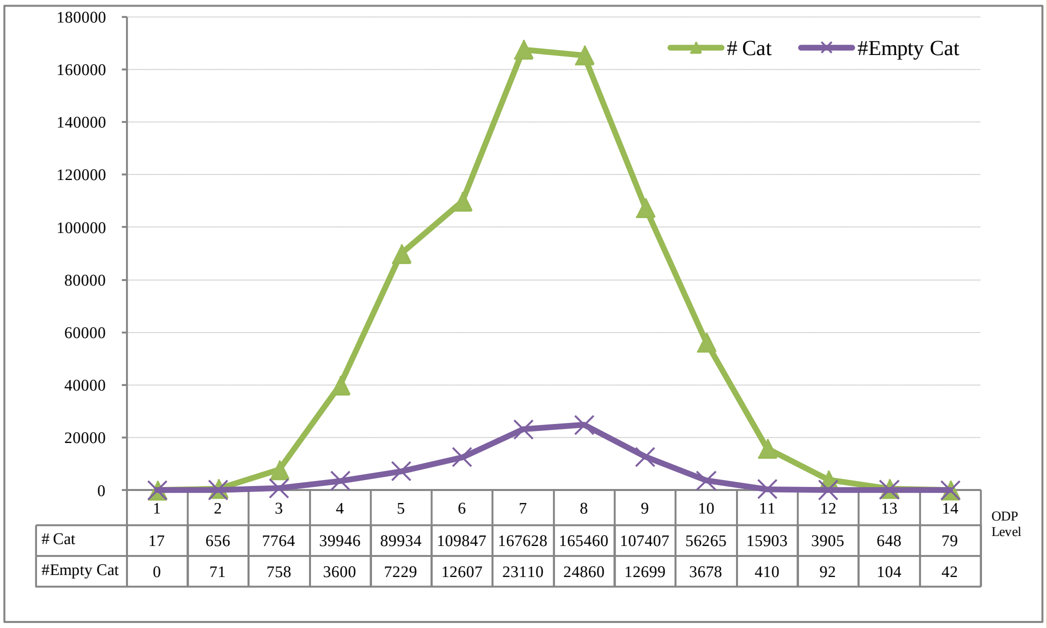


Figure 4-7 Number of categories in the top 17 ODP categories at different levels

The following two observations can be obtained from Figure 4-5 and Figure 4-6.

Firstly, an interesting feature of Figure 4-6 is that the number of subcategories of the category "Regional" and "Word" are much bigger than the number of subcategories in other first level ODP categories. While "Regional" has 311712 subcategories and "World" has 258412 subcategories, for the rest top-level ODP categories, 47282 is the highest number of subcategories arranged under the category "Arts". Figure 4-5 demonstrates that percentages of number of categories under "Regional" and "World" are as high as 40.70% and 33.80% respectively, these takes more than three quarters of all the ODP categories. All the 14 top-level ODP categories employed in this research as data set have only 178170 subcategories,



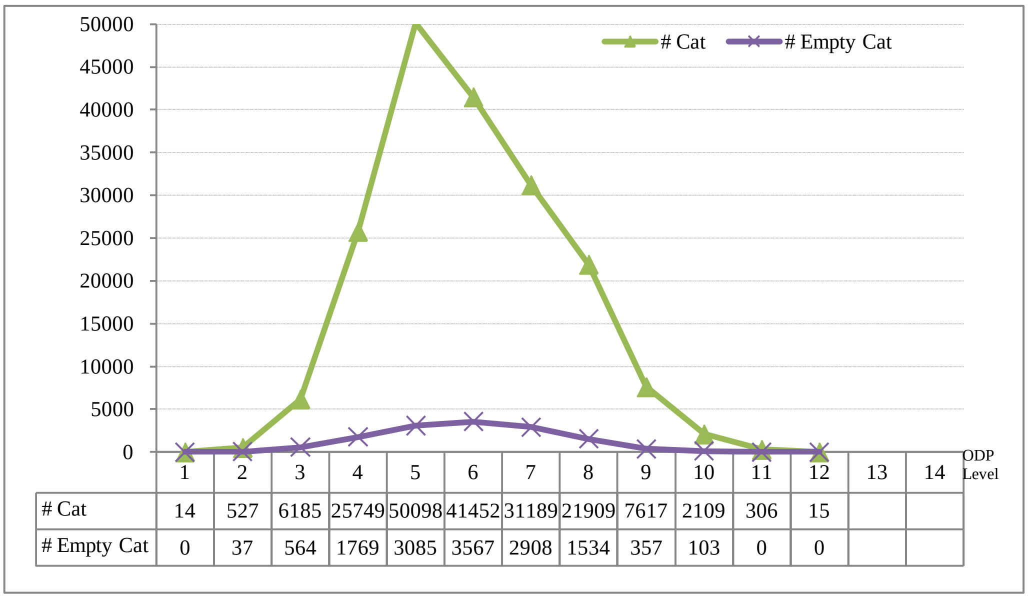
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# Cat	17	656	7764	39946	89934	109847	167628	165460	107407	56265	15903	3905	648	79
#Empty Cat	0	71	758	3600	7229	12607	23110	24860	12699	3678	410	92	104	42

Figure 4-8 Total number of categories and *emptyCategories* in the top 17 ODP categories

only around 25% of all the subcategories in the whole ODP knowledge repository. These figures imply that editors of the ODP categories have not taken into account the balance of the ODP *taxonomic tree*.

Another interesting discovery from Figure 4-6 is that the number of *emptyCategories* of “Regional” is 52760; this is even bigger than the number of categories under “Art” (47282) which has the most categories except “Regional” and “World”.

As can be seen from Figure 4-7 and the 17 figures in Appendix 3, generally speaking, the



	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# Cat	14	527	6185	25749	50098	41452	31189	21909	7617	2109	306	15		
# Empty Cat	0	37	564	1769	3085	3567	2908	1534	357	103	0	0		

Figure 4-9 Total number of categories and *emptyCategories* in the top 14 ODP categories

majority of these figures (Appendix Figure 3-1, Appendix Figure 3-4, Appendix Figure 3-10, Appendix Figure 3-12, Appendix Figure 3-14, Appendix Figure 3-16, and Appendix Figure 3-15) demonstrate the distribution of *emptyCategory* in the 17 top-level ODP categories is close to a normal or Gaussian distribution, especially when only 14 top-level ODP categories are taken into account, as illustrated in Figure 4-9. The figure clearly reveals that the number of *emptyCategories* is normally distributed, with μ = 1162.58, and σ = 1362.44. However, the shapes of some of the figures of the number of *emptyCategories* against different ODP levels are irregular, for instance, the shapes shown in Appendix Figure 3-2, Appendix Figure 3-6, Appendix Figure 3-7, and Appendix Figure 3-11.

On the other hand, the number of subcategories of some of the 17 top-level ODP categories may have a Poisson distribution, as demonstrated by some of the figures, such as Appendix Figure 3-1 to Appendix Figure 3-7, Appendix Figure 3-12, and Appendix Figure 3-15. Except some irregular shapes as shown in Appendix Figure 3-8, Appendix Figure 3-11, Appendix Figure 3-9 and Appendix Figure 3-13, the rest of the figures show that the number of subcategories in the top-level ODP categories has a somewhat normal distribution.

Figure 4-7 and Figure 4-8 demonstrate that most of the ODP subcategories are arranged at level seven and level eight. If considering only the 14 top ODP categories, as illustrated in Figure 4-7 and Figure 4-9, most of the subcategories (about (50098 + 41452) / 187170 = 48.91%) are assigned at level five and six.

Figure 4-10 illustrates the shape of the ODP *taxonomic tree* with respect to the number of categories for all of the 17 top-level ODP categories at the different levels of the tree. From top to bottom, the different levels of the ODP *taxonomic tree* are represented vertically by the nodes in the figure; and the number of categories is represented by the width between two horizontal nodes. The figure illustrates the shape of the ODP *taxonomic tree* is somewhat like a spindle. Figure 4-11 shows the percentage of *emptyCategories* of the 14/17 top-level ODP categories at the different levels of the ODP *taxonomic tree*. The overall average of *empty-categoryDocuments* is 6.94% for the 14 top-level ODP categories; and 10.44% for the 17 top-level ODP categories. The figure also reveals that at the 14th level of the ODP *taxonomic tree*, more than half (53.16%) of the categories are *empty*; and at 13th level, the percentage is 16.05%. It is interesting to note that the high percentages of *emptyCategory* in the two deepest levels are sourced from the category “World”. One of the explanations of this phenomenon is that as the ODP path from the root to the leaf getting longer and longer (or deeper and deeper), users may at the same time being less inclined to explore such a deep category. It is also hard to avoid missing such a deeply located category when travelling from the root of ODP. The advantage of deeply locating a category is that it provides more accurate description of the category in the ODP knowledge hierarchy;

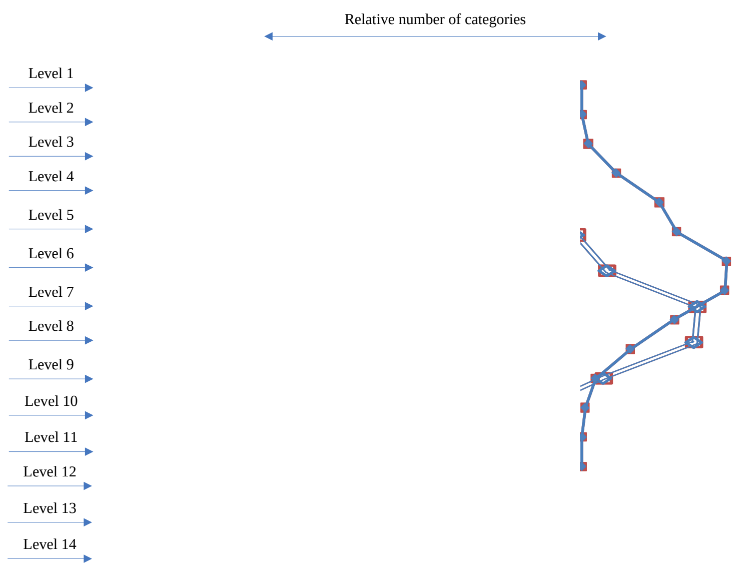
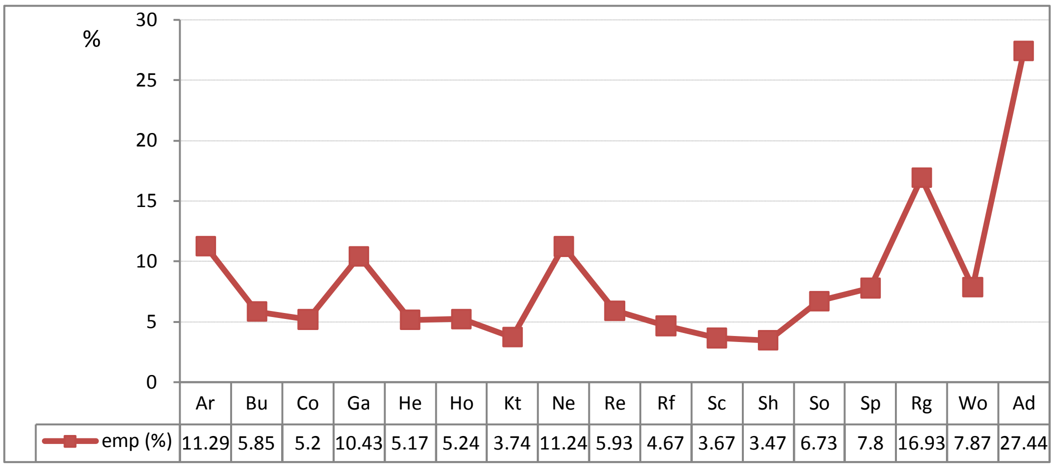


Figure 4-10 Relative number of categories for all of the 17 top-level categories at the different levels of the ODP taxonomic tree

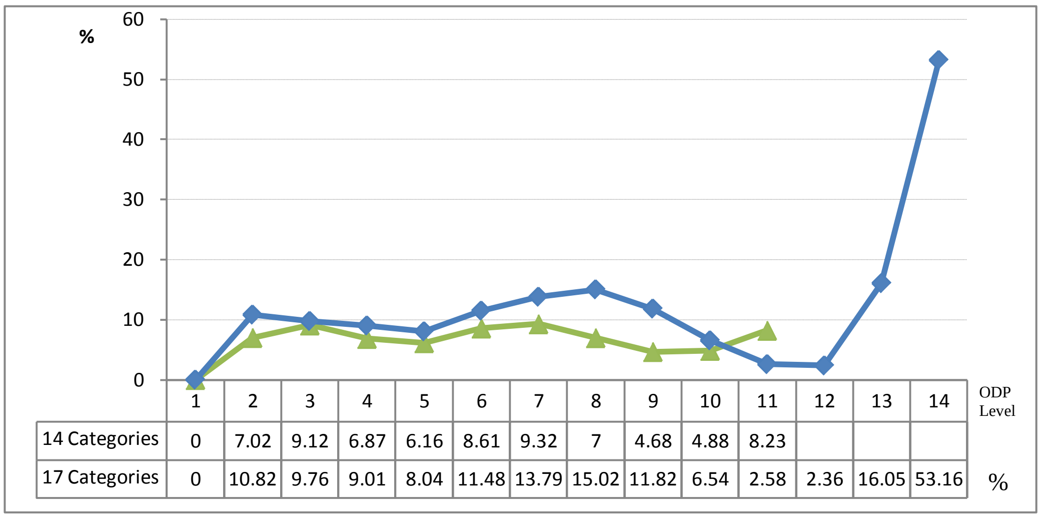
however, even if one believes such a kind of deep hierarchy is more appropriate to locate a category, it takes much more time to browse a category like this.

Figure 4-12 provides the percentage of *empty-categoryDocuments* with regard to the 17 top-level categories. Note that more than 27% of subcategories of "Adult" are *emptyCategories*. This is maybe because the category is invisible from the homepage of the ODP and users might not even be aware of this category. The second highest percentage of *emptySubcategories* is the category "Regional", for which the percentage is 16.93%. One possible reason for this second highest *emptyCategory* percentage is that there are too many subcategories under the category "Regional" – it has 311712 out of the total 765459 ODP categories, the huge number of subcategories makes it hard to locate an appropriate (too



	Ar	Bu	Co	Ga	He	Ho	Kt	Ne	Re	Rf	Sc	Sh	So	Sp	Rg	Wo	Ad
emp (%)	11.29	5.85	5.2	10.43	5.17	5.24	3.74	11.24	5.93	4.67	3.67	3.47	6.73	7.8	16.93	7.87	27.44

Figure 4-11 Percentages of *emptyCategories* in the 14/17 top-level ODP categories



ODP Level	1	2	3	4	5	6	7	8	9	10	11	12	13	14
14 Categories	0	7.02	9.12	6.87	6.16	8.61	9.32	7	4.68	4.88	8.23			
17 Categories	0	10.82	9.76	9.01	8.04	11.48	13.79	15.02	11.82	6.54	2.58	2.36	16.05	53.16

%

Figure 4-12 % of *emptyCategories* in the 14/17 top-level ODP categories at different levels

many choices) subcategory, and thus lead some subcategory has no webpage indexed. Some might try other ODP categories that are also relevant to topic of their webpages.

It is also worth to note that the percentage of *emptySubcategories* of "World" is relatively low, only 7.87% of the subcategories of "World" are *emptyCategories* (Figure 4-11). Note also that when observing from different levels of the ODP *taxonomic tree*, as pointed out previously, at level 13 and 14, the percentage of *emptyCategories* of "World" is as high as 104/637 = 16.33% and 42/76 = 55.26% (Appendix Figure 3-15). The data reveal that ODP users all of the world are using ODP as an indexing tool to publish their webpages; however, in spite that they prefer relatively concise topic, they are reluctant to delve downwards by following the ODP tree path deeper than 12 levels to locate an proper category. On the other hand, all subcategories at 13th and 14th levels of the "Regional: North America" are not *emptySubcategory*, there are always some webpages indexed under these deepest subcategories.

Table 4-3 summaries the number of subcategories of the 14/17 top-level ODP categories at different levels of the ODP *taxonomic tree*. Table 4-4 presents the number of *emptySubcategories* under the 14/17 top-level ODP categories at different levels. The total number of categories of the ODP knowledge hierarchy is 765459, and for the 14 top-level categories, the number is 187170. That is, 14 out of 17, or 82.35% of the top-level ODP categories take only 25% of all ODP subcategories; while 2 out of 17, or 11.76% of the top-level ODP categories have been assigned as much as 75% of all ODP categories. Category "Regional" itself (1/17 = 5.88%) takes 40.70% of all ODP categories. There are all together 89260 *emptyCategories* against all the ODP categories, and 13951 *emptyCategories* for the 14 top-level ODP categories. Similar to the ratio of categories assigned to the two categories,

Table 4-3 Number of subcategories under the 14/17 top-level ODP categories (Lv: level)

	Lv1	Lv2	Lv3	Lv4	Lv5	Lv6	Lv7	Lv8	Lv9	Lv10	Lv11	Lv12	Lv13	Lv14	Sum
Arts	1	41	536	4479	17138	13048	7104	3142	1314	416	63				47282
Business	1	48	714	2457	3093	2271	2105	1195	307	41					12232
Computers	1	45	574	1878	2506	2194	835	342	91	3					8469
Games	1	24	413	1753	4236	3481	1957	713	180	10	28				12796
Health	1	37	347	1113	2609	1183	1202	371	155						7018
Home	1	20	218	729	926	444	232	101							2671
Kids &**T**eens	1	14	165	1629	2027	1123	721	528	214	77	44				6543
News	1	19	101	83	236	57	28								525
Recreation	1	32	478	1646	2660	2744	2229	830	114						11217
Reference	1	23	180	1454	788	1470	2638	3295	1703	509	94	1			12159
Science	1	52	378	1962	2392	1981	1879	3263	1274	118	9				13309
Shopping	1	35	542	1732	1791	964	225	61	9	2					5362
Society	1	30	746	2863	5940	5217	5599	5888	1736	814	68	14			28916
Sports	1	107	793	1971	3756	5275	4435	2180	148	5					18671
Sub-Total	14	527	6185	25749	50098	41452	31189	21909	7617	2109	306	15			187170
Regional	1	36	393	4399	13514	33955	96032	98245	56816	16388	1738	181	11	3	311712
World	1	81	1054	9002	24600	32288	48263	44513	42665	37670	13853	3709	637	76	258412
Adult	1	12	132	796	1722	2152	2044	793	309	98	6				8065
Total	17	656	7764	39946	89934	109847	167628	165460	107407	56265	15903	3905	648	79	765459

"Regional" and "World", the two out of 17 categories (2/17 = 11.76%) have (89260-13951)/89260 = 84.37% *emptyCategories*; while the rest top-level categories have around 15% of *emptyCategories*. The two groups of data reveal that the distribution of number of categories and *emptyCategories* follow the so-called 20-80 principle[27]. However, unlike what is emphasized by the principle, in this research, the attention is paid to the 20% (of all the categories) that is produced by the 80% (14 top-level ODP categories).

4.3.2 Average and Maximum Length of *CategoryDocuments*

Each of the categories in the ODP has a corresponding *categoryDocument* that contains the *identification* of the category – the path from the root of the ODP to the category; and the *content* of the *categoryDocument* – the semantic characteristics related to the category. The

Table 4-4 Number of empty subcategories under the 14/17 top-level ODP categories (Lv: level)

	Lv1	Lv2	Lv3	Lv4	Lv5	Lv6	Lv7	Lv8	Lv9	Lv10	Lv11	Lv12	Lv13	Lv14	Sum
Arts		1	44	357	1505	1455	1255	557	124	37	5				5340
Business		1	21	88	133	110	244	79	36	3					715
Computers		1	59	81	110	126	33	24	6						440
Games			97	292	291	308	285	36	22	0	4				1335
Health		2	36	68	69	104	16	6							363
Home			27	3	33	45	11	21							140
Kids &**T**eens		1	2	109	28	44	55	4	2						245
News			37	12	6	1	3								59
Recreation		1	19	105	155	220	119	21	17	8					665
Reference		1	34	179	36	88	54	99	52	20	5				568
Science		27	4	83	69	114	96	88	7	4					492
Shopping		1	19	51	66	36	12	1							186
Society			106	169	234	448	398	489	59	31	13				1947
Sports		1	59	172	350	510	239	99	26						1456
Sub-Total		37	564	1769	3085	3567	2908	1534	357	103	27				13951
Regional		26	43	522	1939	5735	15662	18288	8897	1573	71	4			52760
World		8	129	1006	1719	2712	3978	4884	3374	1980	312	88	104	42	20336
Adult			22	303	486	593	562	154	71	22					2213
Total	0	71	758	3600	7229	12607	23110	24860	12699	3678	410	92	104	42	89260

[27] http://en.wikipedia.org/wiki/Pareto_principle

following figures and tables present the maximum and average length of these generated *categoryDocument* sets at different levels of the ODP *taxonomic tree*.

Table 4-5 shows the maximum lengths of the *categoryDocuments* for the 14 top-level ODP categories. The maximum length of all the *categoryDocuments* is 15080, and the minimum of the maximum length is only 31. The *categoryDocument* with the maximum length is located at the eighth level of the ODP *taxonomic tree*, and the *categoryDocument* which has the minimum length is found at the top-level of the tree. Note that the minimum length of the *categoryDocuments* is not listed here for the reason that the number of tokens (or length of a document) of the "topic" in the <topic, document> tuple changes dramatically from only one, such "arts", up to 28, for instance, "Reference Education Colleges and Universities North America United States California California State University San Diego State University Departments and Programs College of Professional Studies and Fine Arts Programs". Also note that the *empty-categoryDocuments* are not taken into account when calculating the length of the *categoryDocuments* because the lengths of all the *empty-categoryDocuments* are zero.

Table 4-6 provides the information about the average lengths of the categories for the 14 top-level ODP categories. The data reveal that the average length of the *categoryDocuments* for

Table 4-5 Maximum length of the *categoryDocuments* for the 14 top-level ODP categories

		Lv1	**Lv2**	**Lv3**	**Lv4**	**Lv5**	**Lv6**	**Lv7**	**Lv8**	**Lv9**	**Lv10**	**Lv11**	**Lv12**	**Max**	**Min**	**µ**	**σ**
Arts		60	665	2246	7246	4776	3473	2871	1871	1216	790	716		7246	60	2357	2147
Business		195	1546	7409	7684	6320	5861	5955	4442	4122	903			7684	195	4444	2712
Computers		197	1181	4690	3996	8615	6943	1901	1895	766	439			8615	197	3062	2906
Games		138	388	1897	1394	1922	998	1124	1180	1293	225	111		1922	111	970	666
Health		135	949	1698	4262	2405	3206	2795	1883	920				4262	135	2028	1285
Home		31	706	2623	1475	2495	1527	494	340					2623	31	1211	981
K&T		35	973	1211	1559	1735	1422	954	1464	1754	633	91		1754	35	1076	608
News		646	1899	1119	2388	2814	1646	646						2814	646	1594	840
Recreation		33	759	3415	3700	6099	1966	3801	3982	1096	341			6099	33	2519	1976
Reference		32	1050	4170	2254	2034	2205	2830	1532	1418	677	481	50	4170	32	1561	1221
Science		36	1249	2919	3168	3741	2704	2503	1211	1172	685	430		3741	36	1802	1244
Shopping		54	571	2498	4246	3245	2568	1426	836	309	63			4246	54	1582	1473
Society		136	614	2372	3959	3393	5553	2171	15080	8072	3773	302	118	15080	118	3795	4294
Sports		37	612	3579	2262	2180	7883	10246	1766	616	50			10246	37	2923	3468
summary	Max	646	1899	7409	7684	8615	7883	10246	15080	8072	3773	716	118	15080			
	Min	31	388	1119	1394	1735	9982	494	340	309	50	91	50		31		
	µ	161	1004	3284	3818	4026	3723	3331	3754	2371	1029	407	95				
	σ	162	418	1645	1959	2043	2224	2568	3839	2168	1033	238	48				

Max: Maximum value; Min: Minimum value; µ: Average; σ: standard deviation; Lv: level

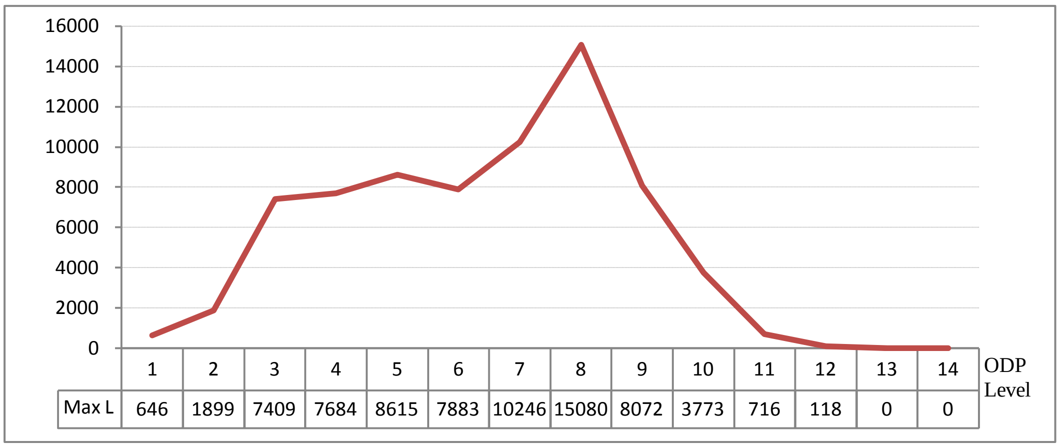
Table 4-6 Average length of the *categoryDocuments* for the 14 top-level ODP categories

		Lv1	Lv2	Lv3	Lv4	Lv5	Lv6	Lv7	Lv8	Lv9	Lv10	Lv11	Lv12	Max	Min	μ	σ
Arts		60	149	265	141	76	87	74	74	71	74	80		265	60	105	61
Business		195	264	569	484	365	258	193	186	170	117			569	117	280	147
Computers		197	236	345	258	212	224	137	106	83	252			345	83	205	78
Games		138	128	132	80	61	70	80	77	84	78	43		138	43	88	31
Health		135	200	294	202	118	167	141	99	108				294	99	163	62
Home		31	295	291	187	123	118	53	72					295	31	146	103
K & T		35	299	102	107	109	158	105	114	142	94	47		299	35	119	69
News		646	438	253	259	289	105	228						646	105	317	175
Recreation		33	207	307	192	136	134	158	145	78	128			307	33	152	74
Reference		32	297	331	85	169	90	75	75	98	86	113	50	331	32	125	95
Science		36	343	392	260	189	137	81	53	52	129	123		392	36	163	121
Shopping		54	148	404	284	198	133	152	166	104	50			404	50	169	107
Society		136	132	245	196	131	129	103	131	194	163	73	55	245	55	141	53
Sports		37	126	168	128	111	91	77	75	60	30			168	30	90	43
summary	Max	646	438	569	484	258	228	186	194	252	123	55		646			
	Min	31	126	102	80	61	70	53	53	52	30	43	50		30		
	μ	126	233	293	205	163	136	118	106	104	109	80	53			162	
	σ	162	95	119	105	83	53	51	41	44	61	33	4				

Max: Maximum; Min: Minimum; μ: Average; σ: standard deviation

different top-level ODP categories does not vary drastically. The maximum of the average length is 317, the shortest of the average document length is 88, the average value of the average document length is 162, and the standard deviation is 67.

The maximum lengths of *categoryDocuments* at different levels of the 14 top ODP categories are illustrated in Figure 4-13 for visualization purpose; the average and standard deviation of the average length over the 14 top ODP categories at different levels of the ODP *taxonomic tree* are demonstrated in Figure 4-14, in case that the corresponding level has



	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Max L	646	1899	7409	7684	8615	7883	10246	15080	8072	3773	716	118	0	0

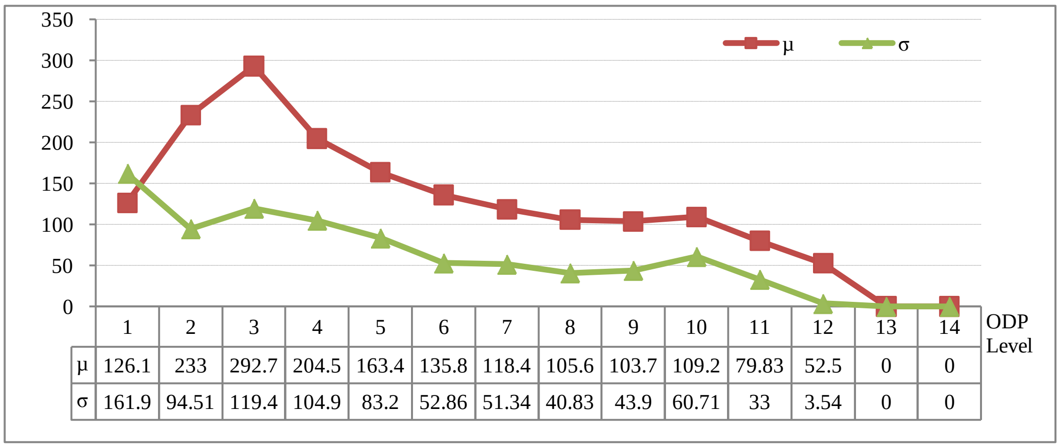
Figure 4-13 Maximum length of *categoryDocuments* for the 14 top ODP categories at different levels

categoryDocuments. For example, at level 12, only category "Reference" and "Society" have *categoryDocuments*, and the average length of these documents are 50 and 55 respectively (Table 4-6), therefore, the average of the two data is (50 + 55) / 2 = 52.5 ≈ 53, and the standard deviation is sqrt((((50-52.5)*(50-52.5) + (55-52.5)*(55-52.5)) /(2-1)) = 3.54 ≈ 4. The data show that the average length of all the *categoryDocument*, not include the *empty* ones, is 143.72 ≈ 144 (average of the 12 values in Figure 4-14); and maximum length of all the *categoryDocuments* is 15080.

Figure 4-14 demonstrates that the maximum average length of *categoryDocuments* is that of the third level *categoryDocuments*, with the length of 292.72 terms. The standard deviation of the length of these documents is 119.41. *CategoryDocuments* of level 12 has the minimum average length, the average length is only about 53 tokens; however, the standard deviation of the average length of *categoryDocuments* at this level is also the smallest, the standard deviation is as small as 3.54 terms (refer to the previous paragraph about how the 3.54 is calculated, however, note that there are only two data points). While the average length of the *CategoryDocuments* at level one is about 126 terms, it also has the biggest deviation, about 162 terms. Refer to Figure 4-13, it can be found that the maximum length of *categoryDocuments* at third level is 7409 terms; and although a *categoryDocument* at level eight has the maximum length of 15080 terms, the averaged length of *categoryDocuments* at this level is about 106 terms.

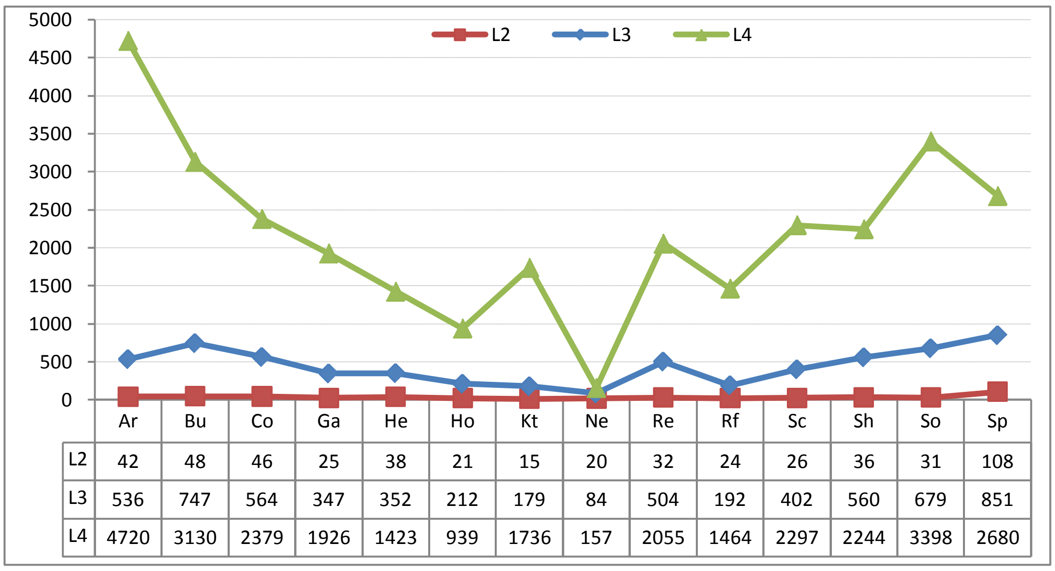
4.3.3 ODP Data Set based on Semantic Characteristics from Level 2, 3 and 4

As discussed in Section 4.2, an ODP *categoryDocument* set can be produced by using *cut&combine* approach. *Cut* from different levels will produce different *categoryDocument* sets. Figure 4-15 illustrates the number of *categoryDocuments* produced for the 14 top-level



	1	2	3	4	5	6	7	8	9	10	11	12	13	14
μ	126.1	233	292.7	204.5	163.4	135.8	118.4	105.6	103.7	109.2	79.83	52.5	0	0
σ	161.9	94.51	119.4	104.9	83.2	52.86	51.34	40.83	43.9	60.71	33	3.54	0	0

Figure 4-14 Average and standard deviation of the lengths of *zero-level categoryDocument* for the top 14 ODP categories at different levels



	Ar	Bu	Co	Ga	He	Ho	Kt	Ne	Re	Rf	Sc	Sh	So	Sp
L2	42	48	46	25	38	21	15	20	32	24	26	36	31	108
L3	536	747	564	347	352	212	179	84	504	192	402	560	679	851
L4	4720	3130	2379	1926	1423	939	1736	157	2055	1464	2297	2244	3398	2680

Figure 4-15 Number of *categoryDocuments* generated when cutting from the second level of the ODP taxonomic tree

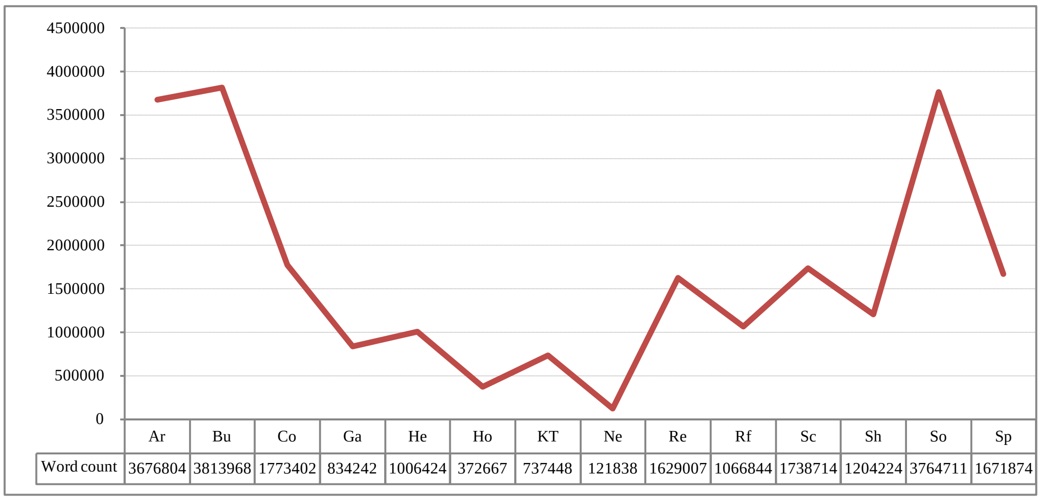
ODP categories when *cut* from the second, third, and fourth level of the ODP *taxonomic tree*.

Figure 4-15 illustrates when *cut* from the second level, category "Sports" has the maximum 108 *categoryDocuments,* whereas "Kids and Teens" has the minimum 15 *categoryDocuments*. There are all together 512 second level *categoryDocuments,* the average number of documents is 36.57, and the standard deviation is 22.86.

When *cut* from the third level of the ODP *taxonomic tree*, once again the category "Sports" has the maximum number of 851 *categoryDocuments*. However, the category "News" has the minimum number of 84 *categoryDocuments*. The average number of third level *categoryDocuments* is 443.5, the standard deviation is 230.28. The total number of third level *categoryDocuments* is 6209.

For the fourth level *categoryDocuments*, the total number of documents is 30548, the average and the standard deviation of the number of *categoryDocuments* is 2182 and 1114.26 respectively. Category "Arts" has the maximum number of *categoryDocuments*, and category "News", the same as *cut* from third level, has the minimum number of 157 *categoryDocuments*.

Table 4-7 illustrates the maximum length and average length of the *categoryDocuments* generated when cutting from the second level of the ODP *taxonomic tree*. As can be seen from the figure, one of the *categoryDocument* in category "Society" has the maximum length of 1588435 tokens, and the average length of the *categoryDocuments* in this category is 121442, which is also the largest number for all of the 14 top ODP categories. The average of the average length of the second level *categoryDocuments* is 47928.93, the standard deviation is 31662.43.



	Ar	Bu	Co	Ga	He	Ho	KT	Ne	Re	Rf	Sc	Sh	So	Sp
Word count	3676804	3813968	1773402	834242	1006424	372667	737448	121838	1629007	1066844	1738714	1204224	3764711	1671874

Figure 4-16 Terms in the *categoryDocuments* generated for the 14 top-level ODP categories

Number of terms which appear in the *categoryDocuments* for the 14 top ODP categories is demonstrated in Figure 4-16. Note that number of tokens for different top ODP categories is independent from the levels of the ODP *taxonomic tree*; that is, no matter the *categoryDocument* sets are generated by *cutting* from whatever level of the ODP *taxonomic tree*, the tokens appear in the *categoryDocument* set for a given top-level category is always the same. The *categoryDocuments* for category “Business” has the maximum number of 3813968 tokens. The average and the standard deviation of the number of tokens for the top 14 ODP categories are 1672291 and 1230941 respectively. The total number of token for all of the ODP categories is 23412077.

Table 4-7 contains the maximum and average length of the *categoryDocument* set produced by *cutting* at level two, three, and four of the ODP *taxonomic tree* for the 14 top-level ODP categories. It also provides the number of tokens included in these generated *categoryDocument* sets.

Table 4-8 presents the overall statistical information of the generated *categoryDocument* sets. As can be seen from the figure, as the level increasing from two to three to four, the number of *categoryDocuments* is also increasing from 512 to 6209 to 30548; the average length is at the same time decreasing from 47929 to 3605 to 711; the maximum length and the standard deviation of the length are decreasing as well.

Table 4-9 provides the same type of information for the two other top-level ODP categories, “Regional” and “Adult”. The information for category “World” is not presented here because in this research multi-language problems are not considered. The information of the two categories is presented separately from the other 14 top ODP categories since it is not taken into account when generating the ODP *categoryDocument* set which is to be used in the experiments in the following chapters.

Table 4-7 *CategoryDocument* set generated from level 2, 3, and 4 of the ODP taxonomic tree for the 14 top-level categories

		Lv2	**Lv3**	**Lv4**
Arts	No of Docs	42	536	4720
	Ave Length	87542	6859	778
	Max Length	1154588	452165	143634
	Tokens	3676804		
Business	No of Docs	48	747	3130
	Ave Length	79457	5105	1218
	Max Length	492004	270566	116480
	Tokens	3813968		
Computers	No of Docs	46	564	2379
	Ave Length	38552	3144	745
	Max Length	536257	221780	147218
	Tokens	1773402		
Games	No of Docs	25	347	1926
	Ave Length	33369	2404	433
	Max Length	617569	119555	35347
	Tokens	834242		
Health	No of Docs	38	352	1423
	Ave Length	26484	2859	707
	Max Length	233147	73326	37550
	Tokens	1006424		
Home	No of Docs	21	212	939
	Ave Length	17746	1757	396
	Max Length	152802	32158	12007
	Tokens	372667		
Kids and Teens	No of Docs	15	179	1736
	Ave Length	49163	4119	360
	Max Length	272014	87129	60294
	Tokens	737448		
News	No of Docs	20	84	157
	Ave Length	6091	1450	776
	Max Length	51946	47218	33297
	Tokens	121838		
Recreation	No of Docs	32	504	2055
	Ave Length	50906	3232	792
	Max Length	379920	251005	223145
	Tokens	1629007		
References	No of Docs	24	192	1464
	Ave Length	44451	5556	728
	Max Length	811171	667944	494520
	Tokens	1066844		
Science	No of Docs	26	402	2297
	Ave Length	66873	4325	756
	Max Length	443412	263962	125284
	Tokens	1738714		
Shopping	No of Docs	36	560	2244
	Ave Length	33450	2150	536
	Max Length	125769	50990	26899
	Tokens	1204224		
Society	No of Docs	31	679	3398
	Ave Length	121442	5544	1107
	Max Length	1588435	1192587	858491
	Tokens	3764711		
Sports	No of Docs	108	851	2680
	Ave Length	15479	1964	623
	Max Length	236099	155471	128673
	Tokens	1671784		

Table 4-8 Statistical information of data sets generated from level 2, 3, & 4 of the ODP taxonomic tree

Level	**Total number of docs**	**Average length of docs**	**Maximum length of docs**	**StDev of average length of docs**	**Number of words**
Lv2	512	47928.93	1588435	31662.41	23412077
Lv3	6209	3604.86	1192587	1673.48	
Lv4	30548	711.07	858491	243.45	

4.3.4 Assign Labels to Books, Papers, and Pieces of Information

Note that the average length of the generated labeled ODP *categoryDocument* sets when cutting from second, third and fourth levels are around 48,000 words, 3,600 words and 700 words respectively (Table 4-8). These are the lengths similar to that of a book, a paper, and a long paragraph. The average length of the documents in the *categoryDocument* set produced without using *cut&combine* approach is about 160 words (Table 4-6), this is somewhat the length of an abstract or a piece of information such as Web snippets.

According to the different application scenarios, an appropriate labeled ODP *categoryDocument* set with different average length can be generated. It is hard to obtain satisfactory categorization performance if a text classifier is trained by a labeled document set where documents are short pieces of information, and the trained classifier is then used to assign labels to a set of testing documents in which documents are lengthy books. Since cutting at the different levels of the ODP taxonomy tree can generate labeled *categoryDocuments* with different average length, it is thus very flexible to produce a training data set that has average length that matches the average length of a testing document set. This is an obvious advantage of the ODP *categoryDocument* sets generated by applying *cut&combine* approach over other benchmark labeled document sets in that the average length of the documents is fixed.

Table 4-9 Statistical information for category "Adult" and "Regional"

		Lv2	**Lv3**	**Lv4**
Adult	No of Docs	13	137	1015
	Ave Length	35332	3352	452
	Max Length	167937	50396	22575
	Word count	459328		
Regional	No of Docs	11	403	4467
	Ave Length	1850462	50508	4556
	Max Length	13047062	11569086	2060479
	Word count	20355087		

4.3.5 Reading the Data in Table 4-3, Table 4-4 and Table 4-7

The data presented in Table 4-3, Table 4-4 and Table 4-7 hold a kind of relationship. For example, in Table 4-3, the category "Business" has one top-level (Lv1) category, the "Business" itself, 48 second level (Lv2) categories, and one *emptyCategory* at Lv2 as illustrated in Table 4-4. The generated *categoryDocuments* for this category should be 1 + 48 – 1 = 48, as demonstrated in column L2 of Table 4-7. The category "Arts", as shown in Table 4-3, has one top-level (Lv1) category, the "Arts" itself, and 41 second level (Lv2) categories. Meanwhile, Table 4-4 shows that "Arts" has one *emptyCategory* at level two (Lv2), the number of documents ("No of Docs") provided in Table 4-7 should be one category at top-level (Lv1) plus 41 categories at level two (Lv2) minus one empty category. However, there are 42 generated *categoryDocuments*.

Figure 4-17 illustrates that "Top: Arts: Comics" is a second level *emtpyCategory* and thus there will be no *categoryDocument* produced for the category according to the definition presented in Section 4.1.4. Nevertheless, the topic contains a list of subcategories that will be combined to generate a new *categoryDocument* with the topic "Top: Arts: Comics". That is, although the topic itself is an *emptyCategory*, when *cut&combine* approach is employed, the category still can have a *categoryDocument* which is a combination of the semantic characteristics of its subcategories. In this scenario, the number of generated *categoryDocuments* in Table 4-7 is bigger than that simply adds up the categories at level one (Lv1) and level two (Lv2) in Table 4-3 then minus the number of *emptyCategories* of the two levels (Lv1 and Lv2) in Table 4-4.

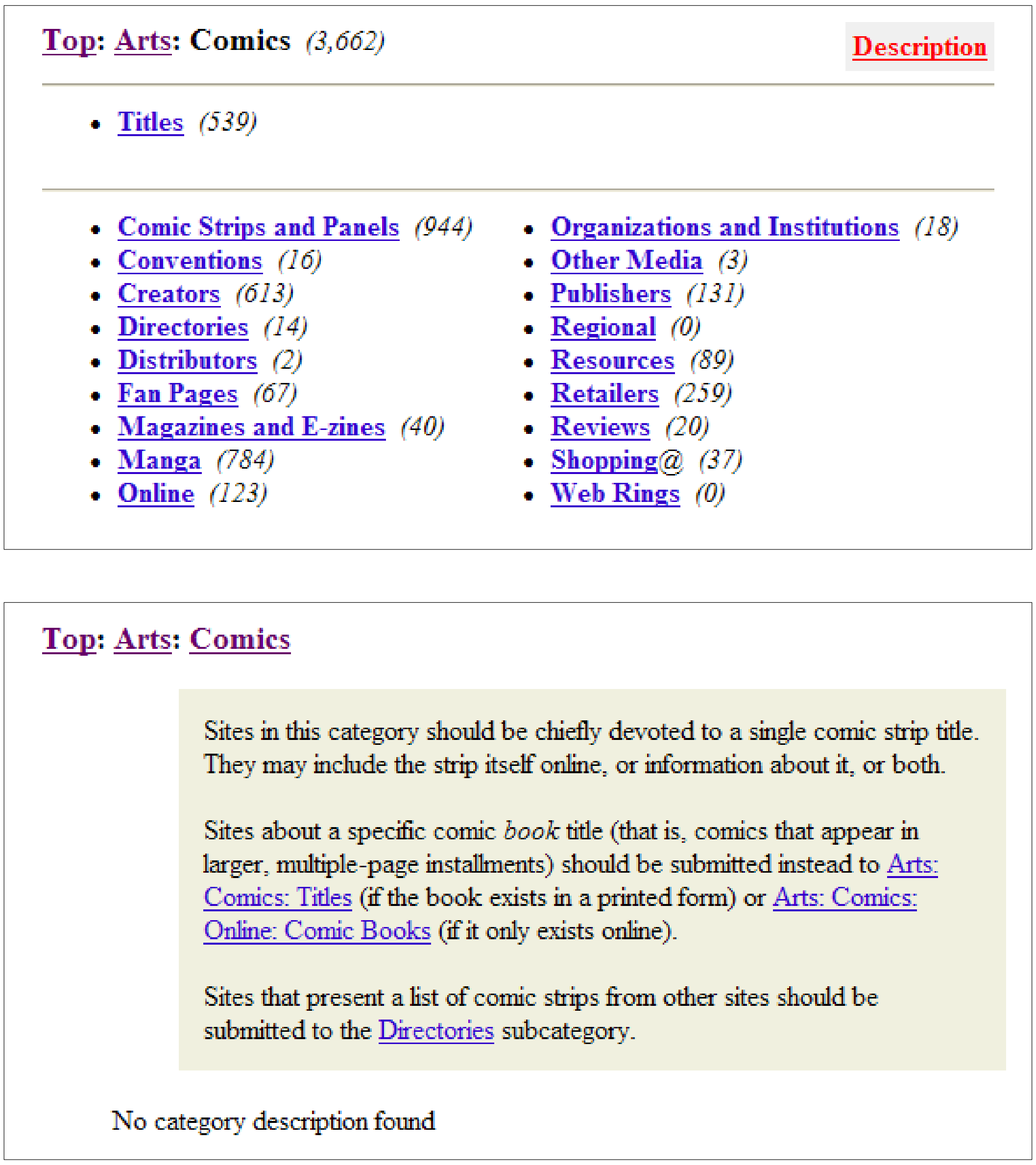
4.4 Limitation of Using ODP Labeled Data Sets

One criticism of the ODP is that competing companies may register as volunteer editors and try to remove or reject their competitors' website from ODP, a similar issue is facing Wikipedia (Giles 2005). However, since the focus of the research is to extract semantic features of the ODP categories, therefore, what concern us is only the presentation of the semantic characteristics of a given ODP category, no matter which competition side the information comes from.

The other issue is the dynamic features of the ODP. The ODP *rdf* files from which the semantic characteristics of each of the categories are extracted are supposed to be updated fortnightly, and consequently, there is no guarantee the experimental results based on one such data set are exactly the same as others, not like the static Reuters-21578 data set or others. In fact, it is interesting to compare the data in Table 4-1 with the data provided in 2007 by Zhu (2007). It can be found that even the number of categories in the first level of

ODP have changed a lot since 2007. This limitation is to further investigated in the future by comparing the testing results based on the *rdf* files on different time periods, for instance, every six months for a period of five years.

Another shortcoming of using ODP as a knowledge hierarchy is that while the two top-level categories, “Regional” and “World”, have 75% of subcategories out of all of the 765459 ODP categories, the two categories not help much for grouping Web snippets to disambiguate search terms. For example, when “jaguar” is used as a search term to retrieve information about animal jaguar, Web snippets about the animal and about jaguar car could both be grouped under the category “Regional” or “World”. If 75% of the ODP categories are not used, it is arguable that the semantic characteristics of the ODP categories are not



Top: Arts: Comics *(3,662)* Description

- Titles *(539)*

- Comic Strips and Panels *(944)*
- Conventions *(16)*
- Creators *(613)*
- Directories *(14)*
- Distributors *(2)*
- Fan Pages *(67)*
- Magazines and E-zines *(40)*
- Manga *(784)*
- Online *(123)*
- Organizations and Institutions *(18)*
- Other Media *(3)*
- Publishers *(131)*
- Regional *(0)*
- Resources *(89)*
- Retailers *(259)*
- Reviews *(20)*
- Shopping@ *(37)*
- Web Rings *(0)*

Top: Arts: Comics

Sites in this category should be chiefly devoted to a single comic strip title. They may include the strip itself online, or information about it, or both.

Sites about a specific comic *book* title (that is, comics that appear in larger, multiple-page installments) should be submitted instead to Arts: Comics: Titles (if the book exists in a printed form) or Arts: Comics: Online: Comic Books (if it only exists online).

Sites that present a list of comic strips from other sites should be submitted to the Directories subcategory.

No category description found

Figure 4-17 Topic “Top: Arts: Comics” is an *emptyCategory*

well explored. However, subcategories of "World" are mainly designed for multi-language which is out the scope of this research. Category "Regional" intends to provide a portal for those who want to arrange their webpages geographically or locally; it is therefore reasonably to ignore these two categories if ODP categories are utilized as a knowledge hierarchy to re-organize Web snippets; or as a data set to evaluate the performance of text categorization and clustering algorithms.

4.5 Summary

This chapter presents statistical information of the socially constructed hierarchical knowledge repository, the Open Directory Project; and of the *categoryDocument* sets generated for the 14 top-level ODP categories. From the perspective of the ODP *taxonomic tree*, the statistical information such as the number of categories, the average and maximum lengths of the *categoryDocuments* at different levels, and number of tokens are calculated for each of the 17 top ODP categories. Then, by using *cut&combine* approach, *categoryDocument* sets for level two, level three, and level four of the 14 top-level ODP categories are constructed. Except for categories "Adult", "Regional", and "World", there are 512 second level ODP *categoryDocuments* which are to be utilized as a document set to evaluate the text categorization and clustering algorithms in the following chapters.

Although ODP has been employed as a data repository for different research purpose, this research, to the best of our knowledge so far, is the first one to provide detailed statistical information about the most comprehensive human-edited Web directory. In the following chapters, some of the text categorization and clustering algorithms are to be evaluated by using the constructed second level ODP *categoryDocuments*.

5 Evaluation of Text Categorization Algorithms Using ODP *CategoryDocument* Sets

This chapter discusses implementation details of several popular text categorization algorithms, and presents experimental results of the performance of the algorithms when ODP *categoryDocuments* are used as example data sets. A novel R^2Cut smoothing strategy is put forward and evaluated as well. Section 5.1 provides the implementation details of three text categorization algorithms Naïve Bayes, AdaBoost.MHR, KNN; and similarity comparison approaches Vector Space Model, language model, and Euclidean distance. Section 5.2 presents the experimental results of the three text categorization algorithms when features are selected by chi-square, information gain, mutual information, and odd-ratio feature selection approaches. In Section 5.3, more text categorization algorithms are evaluated and compared by utilizing a specially constructed small data set in which a document is somewhat like a Web snippet. Section 5.4 utilizes ODP second level *categoryDocuments* to train the above three text categorization algorithms, and then presents the experimental results of the trained classifiers to categorize Web snippets from Google Directory by using "jaguar" as a search-term. Section 5.5 summarizes this chapter.

The contributions of this chapter include firstly, it uses a newly constructed labeled data set to evaluate text categorization algorithms such as Adaboost, kNN, and Naïve Bayes when features are selected by a list of feature selection algorithms; the experimental results would help practitioners and researcher when a text classifier and feature selection algorithm needs to be selected. Secondly, a novel R^2Cut thresholding strategy aims at improving the performance of text categorization algorithms is implemented and experimental results are presented. Lastly, a new version of tf-idf, z-tfidf, that takes into account term frequency not only in one document, but also in other documents, is introduced and experimental results for kNN text classifier are presented.

5.1 Implementation of Categorization Algorithms

There are four factors to be considered when evaluating text categorization algorithms: the variants of the algorithm to be evaluated, the feature selection algorithm employed to pick up informative features; the thresholding strategy; and the data set used for the evaluation.

Almost any of the popular text categorization algorithms have their variants. For example, Naïve Bayes has at least two models, multinomial NB model and multinomial unigram language model (Manning, Raghavan, and Schütze 2008); AdaBoost has Adaboost.MH and

AdaBoost.MR (Schapire and Singer 2000). Theoretical aspects of the algorithms are provided in Appendix 5. Since the different versions of the algorithms may yield different results on different data set, it is therefore important to accurately state the above factors before an experiment is conducted so that they can be reproduced if necessary.

The details of feature selection approaches are presented in Appendix 5 as well. The data set generated based on the second level of the ODP taxonomic tree is described in Chapter 4. Before the discussion of the implementation details of the text categorization algorithms presented in Appendix 5, a small data set employed to evaluate the performance of text categorization algorithms is designed at first; this is followed by an introduction of several existing thresholding strategies; finally, a novel R^2Cut thresholding strategy is put forward which assigns two categories to one documents if conditions are satisfied.

5.1.1 A Small Data Set S-Set

Some text categorization and clustering algorithms are computational expensive, especially when the number of documents or terms are huge. For example, when using the second level ODP *categoryDocuments* as a training data set, in spite of there being only 512 documents, there are 23,412,077 words to be process. When using feature selection algorithms to choose 10,000 features, the term-document matrix (Manning, Raghavan, and Schütze 2008) has the dimension of 512×10000. Using such a huge amount of data to evaluate a text classifier algorithm such as boosting in a personal computer is obviously time consuming.

To address the issue of the poor computational capacity for text processing of personal computers, a compromise strategy is to seek for or create a concise data set. In this research, a small data set with 25 documents that mimic Web snippets and three categories is designed. All the terms and categories in the data set are selected from the ODP categories. The data set, which is initially constructed for debugging purpose, is found later can also be utilized as a data set to evaluate the performance of text categorization and clustering algorithms. Appendix 1 includes the 25 documents and the statistical information of the data set, which will be referred to as S-Set.

Although S-Set is concise and the data are only a tiny portion of ODP categories, to a certain extent it can be taken as a representative of Web snippets or a kind of small size data set. And more importantly, it allows the comparison of text categorization and clustering algorithms in an acceptable time span.

5.1.2 Thresholding Strategy

Let $D = \{\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_n\} \in \Re^m$ be a training document set, $C = \{c_1, c_2, \ldots, c_{|C|}\} \in \{0, 1\}$ be a set of labels (also referred to as class, category). For each label c_i, the task of text categorization is to define a threshold $\tau_i > 0$ and use D' $\subseteq$ D to train a classifier $\hat{\Phi}_i(\mathbf{d}_j) \rightarrow [0,1] \in \Re$ which assigns c_i to a give test document **d** if $\hat{\Phi}_i(\mathbf{d}_j) \geq \tau_i$.

5.1.2.1 Existing Thresholding Strategy

Various existing thresholding policies are available for text categorization algorithms (Fan and Lin 2007; Lewis et al. 2004; Yang 1999, 2001). Five of the strategies (Yang 2001) are summarized in Table 5-1.

In this research, one purpose is to filter Web search results based on user's selection of interest categories. PCut strategy is not suitable because the training data set is generated from the semantic characteristics of the ODP category, whereas the test data set are independently taken from Web search results, the assumption that training data set share the same distribution with test data set is obvious not valid. SCutFBR is not considered as well, as the training set is big enough to train a classifier (except for the SVM$^{\text{Light}}$ algorithm which will be discussed in details in Section 5.1.5). SCut thresholding policy is also not suitable for RIB because this classifier-pivoted strategy may in some circumstance leave test documents un-labeled, which is unacceptable in RIB.

Table 5-1 Thresholding Strategies (Yang 2001).

	Approach	**Features**
RCut	For a document **d**, rank labels and assign the top t labels to **d**.	Suitable for document filtering, document-pivoted categorization.
PCut	For a label c_j, sort test documents and assign top $k=P(c_j)\times x\times m$ documents to c_j, where $P(c_j)$ is the prior probability of c_j, m is the number of labels, and x is an arbitrary real number specified by users.	Assume training data set distribution is the same as test data set, use the power of this distribution to optimize categorization.
SCut	Tune a τ_j for each label j with a validation set to ensure local optimal.	Optimize macro-average performance; dominated by the performance of rare categories.
RTCut	$f(c\mid\mathbf{d}) = r(c\mid\mathbf{d}) + \dfrac{s(c\mid\mathbf{d})}{\max_{c'\in C}\{s(c'\mid\mathbf{d})\}+1}$ **d** is the test document, r(c|**d**) is the rank of category c with respect to **d**, s(c|**d**) the original score of the classifier which assign **d** to c.	Prefer high-precision end of recall-precision space.
SCutFBR	If the micro-averaged F-measure is less than a pre-defined value FBR, set the threshold to infinity (SCutFBR.0), or to the highest decision value of validate data (SCutFBR.1).	Suitable for training example set is small.

Document-pivoted strategies are the proper choice for RIB. However, a simple RCut strategy is quite coarse, and suffers from the risk of performance degradation resulting from low precision if a document is allowed to have more than one label while one document is actually assigned only one. On the other hand, if an algorithm assigns a document to one and only one label, experimental results in the early stage of this research demonstrate that the recall is degraded by about 14% (Zhu 2009).

An ODP *categoryDocument* can be assigned more than one labels. There are about less than 11% ODP categories has multiple classification links that connect two distinct top-level categories (Perugini 2008). This implies that the same number of training documents have multi labels. Therefore, some of the Web search results should also be assigned multi labels. However, in this research, one document is assigned one and only one label except the proposed R^2Cut strategy assigns at most two categories to one document.

5.1.2.2 R^2Cut Thresholding Strategy

A strategy named R^2Cut (Relative RCut) is proposed which is able to assign a second label to a test document if the following condition is satisfied:

$$\frac{\hat{\Phi}_t(\mathbf{d}) - \hat{\Phi}_s(\mathbf{d})}{\hat{\Phi}_t(\mathbf{d})} \leq 0.1\rho$$

where $\hat{\Phi}_t(\mathbf{d})$ is the score estimated by the top ranked classifier with respect to document **d**, $\hat{\Phi}_s(\mathbf{d})$ is the score returned by the second ranked classifier, ρ is a control parameter with a default value 1.0, indicates that if the relative error of the estimated scores of the top-two ranked classifiers is less than 10%, the document is assigned the two labels given by the two top ranked classifiers. Increasing the value of ρ enables more documents have the chance to be assigned more than one labels, and vice versa.

The parameter ρ is adjustable for different text categorization algorithms, and furthermore, even adjustable for different categories. Note also that a bigger ρ increases recall and a smaller ρ increases precision, and this eventually affects the value of macro-average / micro-average F_1.

Since a document in the training set has assigned only one category, utilizing R^2Cut algorithm might at the risk of low precision in cases in which the top ranked category is the correct one. However, if the top ranked category is not correct and R^2Cut strategy predicts the correct category, both precision and recall can be improved.

5.1.3 Naïve Bayes Classifier

The multinomial Naïve Bayes (NB) model (Appendix 5) has been implemented in this research. NB consists of two parts, training and testing. The implemented Naïve Bayes source package has three classes, *NaiveBayes*, *Category*, and *SimilarityScore*. Class *SimilarityScore*, which is reusable by other text categorization algorithms, is designed to keep the calculated similarity score for a given category when a test document is given. Here the similarity score is the estimated probability of the given test document belongs to the given category. Class *Category* calculates document frequency, prior probability in the training phase; and the probability of a given document with respective to a category. Class *NaiveBayes* reads in training document set, pre-processes the documents (stop words removing and stemming), extracts a category set, calculates term frequency, and finally, generates a ranked list of categories for a given test document based on the calculated probabilities in descending order. Figure 5-1 illustrates both the training and testing part of the algorithm. In the training phase, the conditional probability is calculated by Equation 2-2 in Appendix 5, Laplace smoothing scheme (Manning, Raghavan, and Schütze 2008) is utilized to eliminate zero probability that might leads to mathematical overflow when the probability is taken as a divisor in the testing phase.

```
Training:
NaiveBayes(Training set D)
1 read in D
2 extract category set C from D
3 extract vocabulary set V from D
4 get Document number N
5 for each c in C
6       Nc ← Number of document in c
7       calculate prior probability ppro[c] = Nc/N
8       txtc ← concatenate all text of document d belong to c
9       for each term t in V
10            Tct ← get term frequency from txtc
11      for each term t in V
12            cpro[c][t] ← get conditional probability
13 store V, ppro, cpro
14 return

Testing:
NaiveBayes(Testing document d)
1 W ← extract terms from d if the terms also in V
2 for each c in C
3       simScore[c] ← log(ppro[c])
4       for each t in W
5             simScore[c] += log(cpro[c][t])
6       insert c in a ranked list rList based on simSocre[c]
7 return rList
```

Figure 5-1 pseudo code of the implemented Naïve Bayes algorithm

As can be seen from Figure 5-1, the pseudo code demonstrates that in the training phase, the Naïve Bayes algorithm first reads in the training data set D (Line 1). Each of the elements in D is a <topic, document> tuple where topic is an ODP topic which has the format of, for instance, "Computers: Software: Software Testing"; document is a generated *categoryDocument* which contains the semantic characteristics of the given topic. The categories C, here the top-level ODP categories and the vocabulary set V in D are extracted (Line 2, 3). The number of training documents N is counted subsequently (Line 4). A classifier is trained for each of the category c in C (Line 5 to 12). In the training phase, the number of training documents Nc which belongs to the same category c is counted first (Line 6), the prior probability ppro[c] = Nc/N is estimated accordingly (Line 7). All the documents belong to c are concatenated to construct a new text txtc, and terms in txtc are extracted (Line 8). For each of the terms in V (Line 9), the term frequency Tct is calculated against to the generated txtc (Line 10). After term frequency Tct is counted, another "for" loop is conducted (Line 11) to calculate the conditional probability cpro for each of the terms in txtc given the category c (Line 12). After all the classifiers are trained, the extracted vocabulary set V and the approximated probability ppro and cpro are stored for testing purpose (Line 13), and this ends the training phase (Line 14).

The testing phase estimates the probabilities of a given testing document **d** belonging to each of the categories in C. Terms which occurring in **d** and also included in V are extracted at first to generate a term set W (Line 1). Then, for each of the trained classifier $c \in C$ (Line 2), the similarity score SimScore[c] is initialized with the prior probability ppro[c] of c (Line 3). The similarity score is accumulated for each of the terms in W (Line 4) by accumulating the log value of the conditional probability cpro[t][c], the probability of term t given category c (Line 5). The accumulated similarity score represents the probability of the testing document belongs to category c. The score is then inserted into a ranked list rList of categories in descending order (Line 6). rList is initialized as an empty list, each time a classifier is tested against **d**, and the category of the classifier is inserted into rList according to the calculated similarity score in a descending order. After all classifiers are estimated against the testing document, the rList is returned and this ends up the testing phase.

5.1.4 kNN Classifier

The k-Nearest Neighbors (kNN) text categorization algorithm is a lazy learning approach because it has no training phase. Without any training procedure, the similarity between a testing document and a training document is compared directly by means of approximating the distance between them. Three factors that affect the performance of kNN are considered when the algorithm is implemented for text categorization purpose. One of the essential factors is how to estimate the distance – the similarity between a testing document and a category, that is, how to select the k nearest neighbours of the testing document. The second factor is how to choose the number k – it is usually obtained via experiments. The third factor to be considered is how to process the selected k neighbours to make a final decision.

5.1.4.1 Similarity Estimating

There are a wide range of similarity estimating approaches, such s cosine similarity (Salton and Buckley 1988), Manhattan distance, Tanimoto similarity, Jaccard similarity coefficient, and Euclidean distance (Han and Kamber 2006; Mitchell 1997). In fact, any strategy that compares the similarity between two documents can be applied in kNN. In this research, cosine similarity and Euclidean distance are used to decide the k nearest neighbours of a testing document based on the estimated k top ranked similarity scores.

To calculate the cosine similarity and Euclidean distance, documents are represented as a terms vector, and tf-idf term-weighting strategy is employed (Baeza-Yates and Ribeiro-Neto 1999; Salton and Buckley 1988). In addition, the proposed z-tfidf strategy in Chapter 3 is implemented as well, and experimental results regarding to the two term-weighting strategies are presented in Section 5.3.4.

5.1.4.2 The Number k

kNN is not an efficient algorithm and experimentally selecting an appropriate k is time consuming. In this research, k = 5, 10, and 14 are evaluated. Since there are 14 categories, any k bigger than 14 is unrealistic and thus is not taken into consideration.

5.1.4.3 Majority Voting Strategy

A majority voting algorithm intends to output a final decision from as input k labeled documents. This kind of algorithms is limited to assign a testing document to only one category. However, in some cases, multi-labeling strategy is necessary. To address the weakness of one document one label strategy, two algorithms are put forward to make a final decision that allows to assign more than one category to a test document. The first algorithm is named *Dominant Majority* (DM), and the second one is referred to as *Majority Ranking* (MR). Both algorithms return a ranked list of labels that can be assigned to a given testing document.

Dominant Majority

Considering one of the majority voting algorithm presented by Zhu (2009), let k be the total voting member and m be the majority number. Supposed that each of the k member is a <category, document> tuple. *Dominant Majority* is defined as the majority number m > k/2.

In this case, the test document is assigned to only one category decided by the majority members, no matter whether the category of the top ranked member is the same as the category of the majority group. *Weak majority* refers to the case when m <= k/2. In this scenario, if the category of the top ranked member is the same as the category of the majority group, the test document is assigned only one category which is the same as the top ranked member; otherwise, the test document is assigned two categories, one is decided by the weak group, another is decided by the top ranked member. Figure 5-2 illustrates the proposed DM algorithm in the form of pseudo code.

Majority Ranking

Majority Ranking (MR) repeatedly applies majority voting algorithm as proposed by Zhu (2009) to select one category in each loop, until all the k members selected as the nearest neighbours are ranked. The pseudo code of the algorithm is illustrated in Figure 5-3.

The first step of MR reads in k nearest neighbours k <category, document> tuples of a testing document into kNbs (Line 1). Suppose there are n categories amongst the k tuples. The second step applies majority voting algorithm (Zhu 2009) to select amongst kNbs one majority category cs, put cs in a ranked list rList (Line 2). rList is a queue which follows the rule of first in first out. Step 3 removes from kNbs the tuples whose category equals the selected category cs. If there are x such tuples, then k is set to the value of k – x (Line 3). The last step checks if k is zero. If k is zero, it implies all the k nearest neighbours have been process and the rList is returned, the algorithm is stopped. Otherwise, if k is still bigger than zero, go to step 2 to repeat the above process (Line 4).

5.1.5 Support Vector Machines

Many SVMs programs are publicly available today, and the implementation of SVMs algorithms involves quadratic optimization problem with bound constraints and one linear

```
if (m > k/2)
        Assign the document the category decided by the majority members
else if (m <= k/2) {
                if (top ranked member is among the majority group)
                        Assign the document the category decided by the top
                        ranked member
                else
                        Assign the document two categories, one is the same as
                        the top ranked member, another is decided by the
                        majority member.
}
```

Figure 5-2 Dominant majority voting algorithm

```
1. Read in k nearest neighbours (k <category, document> tuples) into kNbs
2. Apply majority voting algorithm to select one category cs, put cs in a ranked
   list rList.
3. Suppose there are x tuples whose category equals cs. Remove from kNbs all
   <category, document> tuples with category equals cs, and set k ← k-x
4. If k is zero, return rList and stop, otherwise, goto step 2.
```

Figure 5-3 pseudo code of the Majority Ranking algorithm

equality constraint (Joachims 1999). Some numerical calculation techniques are required for the SVMs to deal with large volume training set (Joachims 1999; Lu et al. 2006). To avoid recreating the wheel, in this research, SVM^{light} (Joachims 2008), one of the popular implementation of SVM employed by a list of researchers (Lewis et al. 2004; Lu et al. 2006; Campos and Romero 2009), is selected as a off-the-shelf SVM classifier.

SVM^{light} has a list of parameters. Lewis et al. (2004) suggest all the parameters should be left at default values, except for 1) parameter –j, a cost factor which adjusts the weighting of positive training examples over negative training examples; 2) parameter –x, which is not decided by the algorithm of SVM^{light}, but decided by SCutFBR.1 thresholding strategy (Yang 2001). The values of –j are chosen from 0.1, 0.2, 0.4. …1.0, 1.25, … to 15.0 in case it yields the best performance in terms of F_1 for a given classifier. In this research, all the parameters are set as their default values; include the parameter –j (default value 1). Since R^{2}Cut strategy is employed, instead of adjusting parameter –x, parameter ρ in R^{2}Cut is to be adjusted and tested.

Lewis et al. (Lewis et al. 2004) indicate linear SVM outperforms competitors if leave all parameters of SVM^{light} as default, with the SCutFBR.1 thresholding strategy when RCV1, a pre-processed version of Reuters 21578 document set is used as a benchmark collection.

5.1.6 Boosting

In this research, AdaBoost.MHR, the AdaBoost.MH with real-valued predictions is implemented due to the fact that it is the most effective one among the all four different versions of AdaBoosting (Schapire and Singer 2000), and it is also studied by other researchers (Sebastiani, Sperduti, and Valdambrini 2000; Comité, Gilleron, and Tommasi 2003).

The flow chart of the AdaBoost.MHR is illustrated in Figure 5-5. How to train a weak hypothesis is illustrated in Figure 5-4. A source package *Boosting* which contains five classes is defined to implement AdaBoostMHR in RIB.

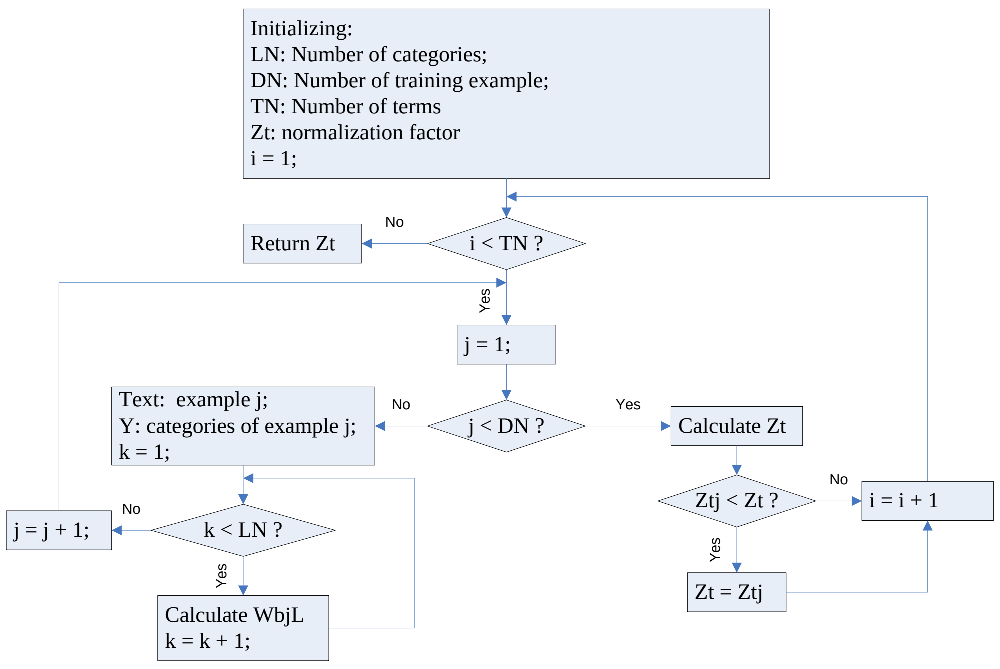


Figure 5-4 Training weak hypothesis.

R^2Cut thresholding strategy is applied to make a final prediction of categories assigned to a given test document. The estimated value given by the final $h_t(x)$ implies a measure of “confidence” (Freund and Schapire 1999) to assign a category to a test document, if the two top ranked categories are both predicted with very high and very close confidence values, it is reasonable to assign the two categories to the test document.

Training round T is selected as an arbitrary number, say, 200, 1000, 10000, or other, in some of boosting algorithm experiments (Lewis et al. 2004; Schapire and Singer 2000; Sebastiani, Sperduti, and Valdambrini 2000). An alternative approach is to set up a terminate condition and stop the training round when the condition is satisfied (Schapire, Singer, and Singhal 1998). One candidate terminate condition is to run a boosting algorithm until the training error reaches a predefined minimal value ε, or simply zero. Let the training round is T_0 when the terminate condition is reached, continuing the training process for another βT_0 rounds is suggested (Schapire, Singer, and Singhal 1998) with the intention of further reducing the testing error, because even if training error reaches zero, testing error keeps going down if the training continues.

When ODP is used as the data set to evaluate the performance of the algorithm, the only one parameter T is set to 10 in this research, this is mainly due to the fact that the limited computing power of the desktop computer used to execute the algorithm. In this research, T = 10, 100, 1000, and 10000 are evaluated with 100 features selected by Odds Ratio feature selection algorithm.

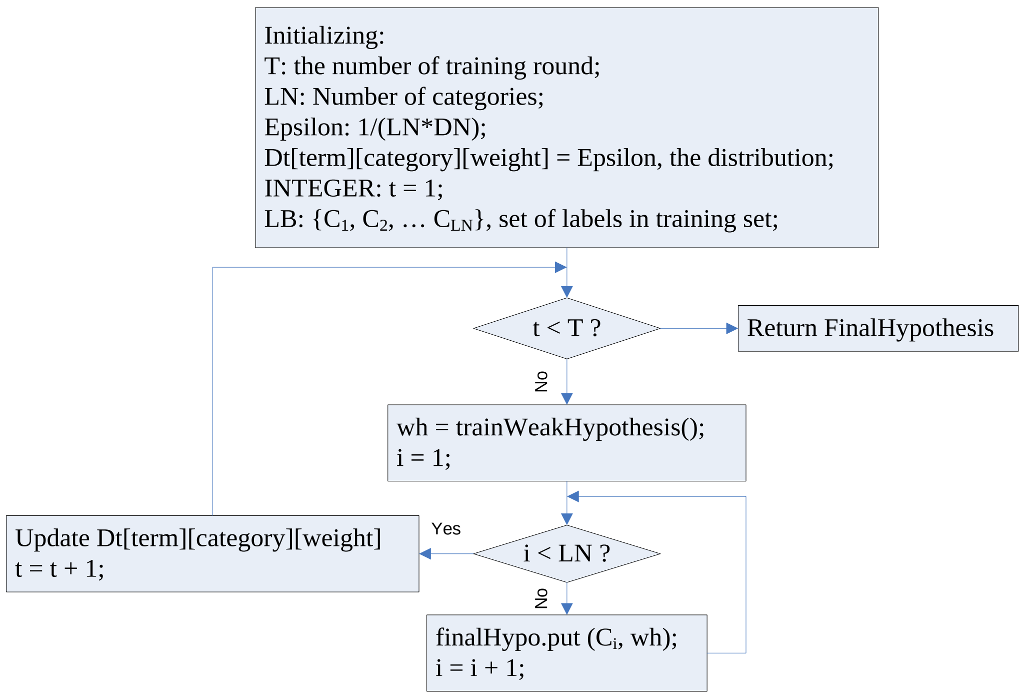

Initializing:
T: the number of training round;
LN: Number of categories;
Epsilon: 1/(LN*DN);
Dt[term][category][weight] = Epsilon, the distribution;
INTEGER: t = 1;
LB: {C_1, C_2, … C_{LN}}, set of labels in training set;
t < T ?
Return FinalHypothesis
No
wh = trainWeakHypothesis();
i = 1;
Update Dt[term][category][weight]
t = t + 1;
Yes
i < LN ?
No
finalHypo.put (C_i, wh);
i = i + 1;

Figure 5-5 Flow chart of AdaBoost.MHR.

5.1.7 Statistical Language Modelling

Smoothing is an essential factor that influences the performance of a statistical language model, therefore, tuning the relevant parameters to make the model reaches its best performance is inevitable and time consuming.

There are two ways to utilize statistical language models in text categorization. One is simply to employ statistical language model as a means of estimating similarities between a test document and different categories, just as cosine similarity is utilized in kNN. Another approach is to train a classifier for each of the categories appeared in training data set. Supposed there are C categories can be extracted from a training data set, the training set are firstly divided into C groups based on the categories they belong to. Then for each of the C sub-data set, a language model is generated. In the testing phase, each of the language models evaluates a similarity score for a test document that is consequently assigned the category with the highest similarity score.

In this research, the first approach is employed, and the only case considered is when k equals one. This is primarily because of the time limitation for this research and tuning parameters is time consuming. Further research on language models is scheduled and comparison between the two approaches is to be conducted as well.

The pseudo code of the statistical language model is illustrated in Figure 5-6. Class *LaguageModel* is designed to implement the model. Two *maximum likelihood* estimators, *document language model* and *collection language model*, are calculated in the training phase respectively as

$$\textit{MaxLhdD} = \textit{(raw term frequency in a document) / (terms in the document)}$$
$$\textit{MaxLhdC} = \textit{(raw term in the collection) / (terms in the collection)}$$

A final language model is the combination of the two models with a smoothing parameter lambda.

P(t|d) = lambda * MaxLhdD + (1 - lambda) MaxLidC

This smoothing strategy is called *LinearInterpolation*. Another smoothing strategy is referred to as *Bayesian Smoothing* which is expressed as

$$P(t|d) = \frac{\textit{(raw term frequency in a document)} + \textit{alpha} \times \textit{MaxLhdC}}{\textit{tokens in the document} + \textit{alpha}}$$

The class *LaguageModel* first reads in raw training documents from a hashmap <string, string> structure where the first parameter is the category of the document, and the second parameter is the content of the document. The class then calculates the *document language model* and the *collection language model*. The two implemented smoothing versions can then be called with a string as parameter which can be taken as a query for an information retrieval system, or as a test document for text categorization purpose. The class returns an estimated probability of statistical language model.

One issue to be addressed here is that if none of the terms contained in a testing document are included in the training data set, a zero probability is returned to indicate that there is no matched category returned. This is problematic in text categorization because a testing

```
Training
1. Read in training document set D
2. Calculate document language model and collection language model
3. Calculate category prior probability
4. Choosing one of the following smoothing strategy
            Linear Interpolation, or
            Bayesian Smoothing

Testing
For a given testing document, select one of the smoothing strategy to obtain
a ranked list of categories, the top ranked category is assigned to the document
```

Figure 5-6 pseudo code of the implemented statistical language model document is assumed to be assigned at least one label. The proposed solution for this problem is to utilize the prior category probability as the estimated similarity scores. The prior category probability for a category c in the category set C is estimated by

$$\text{Prior probability (c)} = \frac{\text{number of documents within category c}}{\text{number of training documents}}$$

The prior probabilities for the categories are ranked and returned in case of a zero probability are encountered.

5.2 Experimental Results Based on ODP CategoryDocuments

This section presents the evaluation results of text categorization algorithms of Naïve Bayes, kNN, and Adaboost in terms of F_1 by using the second level ODP *categoryDocuments* as a labeled data set. The statistical information of the ODP data set is presented in the previous Chapter. Feature selection algorithms involved in the experiments include chi-square, mutual information, information gain, and odd-ratio (Appendix 5, Section 3). The number of features selected by the algorithms is 10,000, 5,000, 3,000, 2,000, 1,000, 500, 300, 200, 100, 80, and 50. Five-fold cross validation is used to make the experimental results more statistically sound.

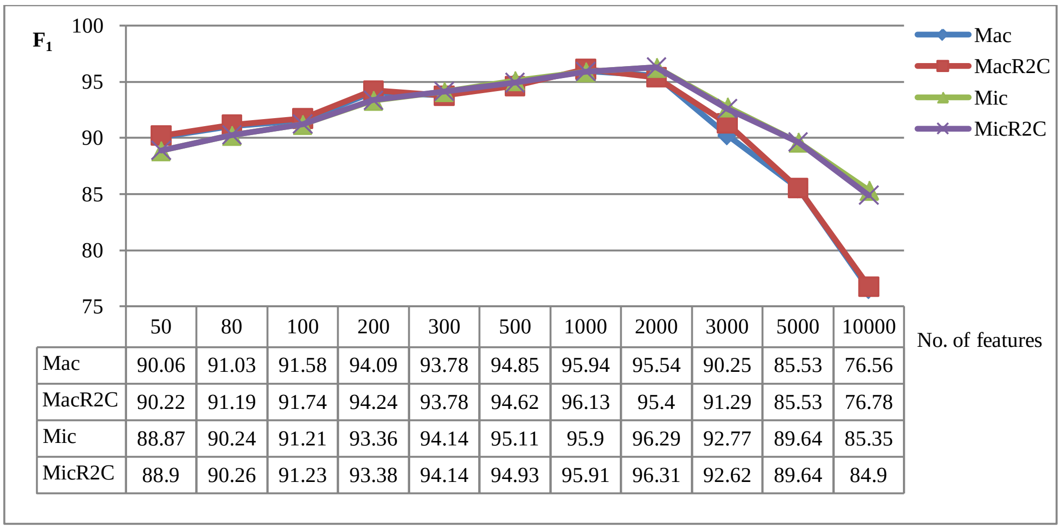
The measures employed to estimate the effectiveness of the text categorization algorithms are micro-averaged and macro-averaged F_1, which are calculated based on micro-averaged and macro-averaged precision and recall.

Since five-fold cross validation is used in the experiments, with regard to the micro-averaged / macro-averaged F_1 obtained in each round, a final F_1 can be calculated by averaging the five micro-averaged / macro-averaged F_1, in terms of micro-averaging; or by first calculating the averaged precision and recall, and then calculating the F_1 by definition, in terms of micro-averaging. In this research, the first approach is used to estimate a final averaged F_1.

5.2.1 Experimental Results of Naïve Bayes

Figure 5-7 to Figure 5-10[28] show the micro-averaged and macro-averaged F_1 for the four different feature selection algorithms with / without R^2Cut smoothing strategy. The four feature selection algorithms are chi-square, information gain, mutual information, and odd-ratio respectively. The experimental data reveal that:

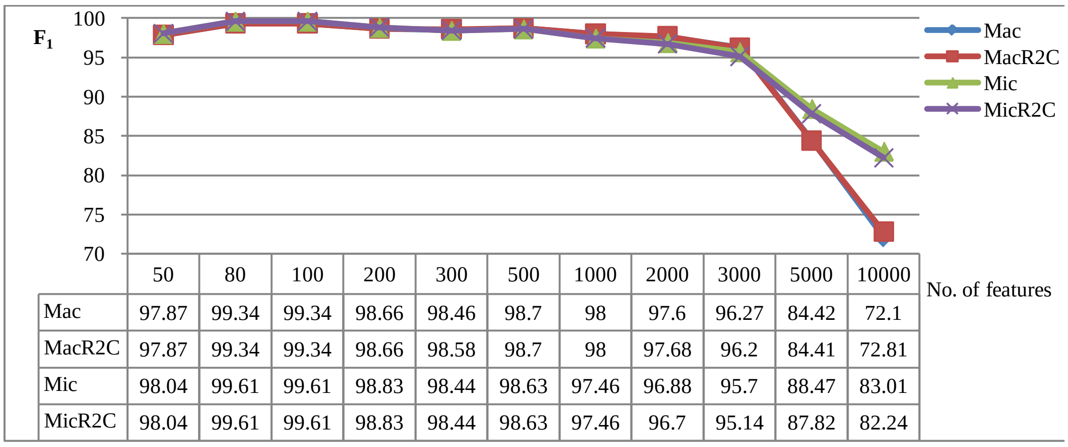
[28] Abbreviations in the figures: Mic/Mac, micro-averaged/Macro-averaged, without R^2Cut smoothing strategy; MicR2C/MacR2C, micro-averaged/Macro-averaged, with R^2Cut smoothing strategy.



	50	80	100	200	300	500	1000	2000	3000	5000	10000
Mac	90.06	91.03	91.58	94.09	93.78	94.85	95.94	95.54	90.25	85.53	76.56
MacR2C	90.22	91.19	91.74	94.24	93.78	94.62	96.13	95.4	91.29	85.53	76.78
Mic	88.87	90.24	91.21	93.36	94.14	95.11	95.9	96.29	92.77	89.64	85.35
MicR2C	88.9	90.26	91.23	93.38	94.14	94.93	95.91	96.31	92.62	89.64	84.9

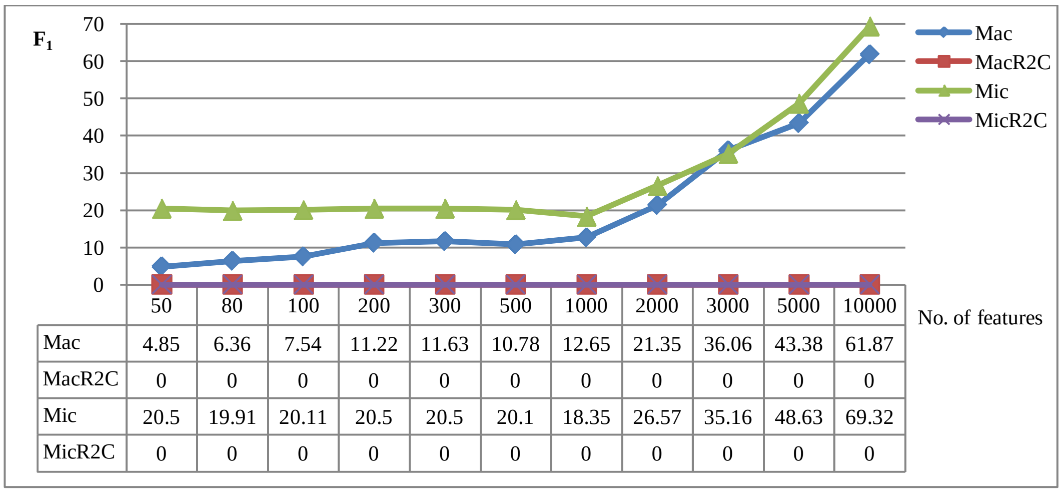
Figure 5-7 Micro/Macro averaged F_1 with/without R^2Cut smoothing strategy of Naïve Bayes algorithm with chi-square feature selection algorithm ($\rho = 0.4$)

1) The number of features affects the performance of Naïve Bayes significantly. For chi-square (Figure 5-7), Naïve Bayes reaches its best performance when features range from 500 to 2000. For information gain (Figure 5-8), features range from 50 to 500 enable Naïve Bayes performs very well. Mutual information is not as good as other feature selection algorithms for Naïve Bayes, and it seems the more features are selected, the better performance can be achieved (Figure 5-9).
2) Odd ratio is one of the best feature selection algorithms for Naïve Bayes (Figure 5-10). It outperforms chi-square and mutual information, and is slightly better than information gain. The preferable number of feature ranges from 300 to 500.
3) Both odd ratio and information gain can enable Naïve Bayes performs extremely well



	50	80	100	200	300	500	1000	2000	3000	5000	10000
Mac	97.87	99.34	99.34	98.66	98.46	98.7	98	97.6	96.27	84.42	72.1
MacR2C	97.87	99.34	99.34	98.66	98.58	98.7	98	97.68	96.2	84.41	72.81
Mic	98.04	99.61	99.61	98.83	98.44	98.63	97.46	96.88	95.7	88.47	83.01
MicR2C	98.04	99.61	99.61	98.83	98.44	98.63	97.46	96.7	95.14	87.82	82.24

Figure 5-8 Micro/Macro averaged F_1 with/without R^2Cut smoothing strategy of Naïve Bayes algorithm with information gain feature selection algorithm ($\rho = 0.4$)

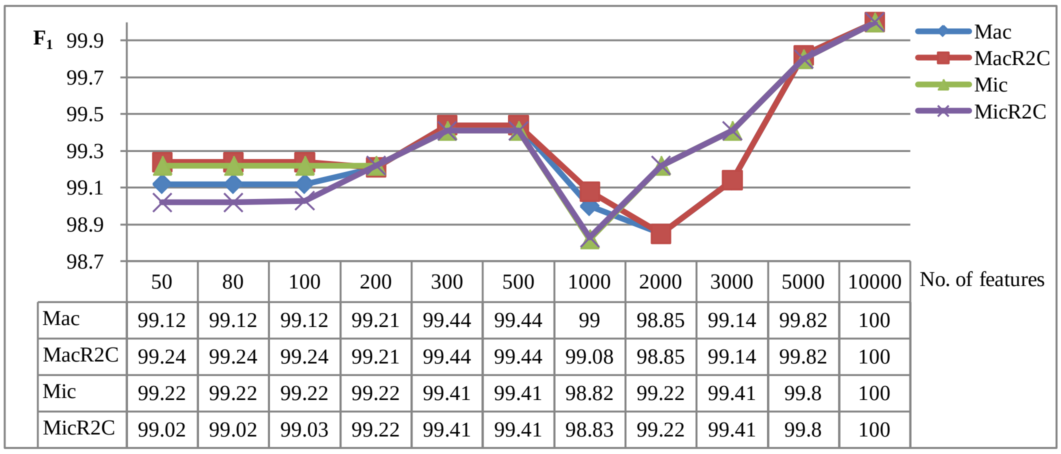


	50	80	100	200	300	500	1000	2000	3000	5000	10000
Mac	4.85	6.36	7.54	11.22	11.63	10.78	12.65	21.35	36.06	43.38	61.87
MacR2C	0	0	0	0	0	0	0	0	0	0	0
Mic	20.5	19.91	20.11	20.5	20.5	20.1	18.35	26.57	35.16	48.63	69.32
MicR2C	0	0	0	0	0	0	0	0	0	0	0

Figure 5-9 Micro/Macro averaged F_1 with/without R^2Cut smoothing strategy of Naïve Bayes algorithm with mutual information feature selection algorithm

when 50 to 500 features are selected. With these selected features, the F_1 is almost always bigger than 99% (Figure 5-8, Figure 5-10). This performance is never reported so far in literature.

4) The proposed R^2Cut smoothing strategy can marginally improve the performance of Naïve Bayes compared with the one document one category strategy when ρ is properly selected. When ρ is too small, the R^2Cut strategy has no effect on the results, and if ρ is too big, the performance of Naïve Bayes is not enhanced, but on other hand is deteriorated. In this research, the results of ρ = 0.1, 0.2, 0.25, 0.3, 0.4, 0.5 and 1.0 are compared, and the value of ρ is as presented in the caption of the figures. The average improvement for the 11 feature number points (50, 80, 100, 200, …, 10000) of F_1 values are summarized in Table 5-2. It can be seen from the table that the overall average



	50	80	100	200	300	500	1000	2000	3000	5000	10000
Mac	99.12	99.12	99.12	99.21	99.44	99.44	99	98.85	99.14	99.82	100
MacR2C	99.24	99.24	99.24	99.21	99.44	99.44	99.08	98.85	99.14	99.82	100
Mic	99.22	99.22	99.22	99.22	99.41	99.41	98.82	99.22	99.41	99.8	100
MicR2C	99.02	99.02	99.03	99.22	99.41	99.41	98.83	99.22	99.41	99.8	100

Figure 5-10 Micro/Macro averaged F_1 with/without R^2Cut smoothing strategy of Naïve Bayes algorithm with odds ratio feature selection algorithm (ρ = 0. 25)

Table 5-2 The improvement of F_1 of NB with / without R^2Cut smoothing strategy (%).

Features		**50**	**80**	**100**	**200**	**300**	**500**	**1000**	**2000**	**3000**	**5000**	**10000**	**AVG**
Chi-square	Mac	0.16	0.16	0.16	0.15	0	-0.23	0.19	-0.14	1.04	0	0.22	0.16
	Mic	0.03	0.02	0.02	0.02	0	-0.18	0.01	0.02	-0.15	0	-0.45	-0.06
	µ	0.1	0.09	0.09	0.09	0	-0.2	0.1	-0.06	0.45	0	-0.12	0.05
Info. Gain	Mac	0	0	0	0	0.12	0	0	0.08	-0.07	-0.01	0.71	0.08
	Mic	0	0	0	0	0	0	0	-0.18	-0.56	-0.65	-0.77	-0.2
	µ	0	0	0	0	0.06	0	0	-0.05	-0.32	-0.33	-0.03	-0.06
Odd ratio	Mac	0.12	0.12	0.12	0	0	0	0.08	0	0	0	0	0.04
	Mic	-0.2	-0.2	-0.19	0	0	0	0.01	0	0	0	0	-0.05
	µ	-0.04	-0.04	-0.04	0	0	0	0.05	0	0	0	0	-0.01
AVG	Mac	0.09	0.09	0.09	0.05	0.04	-0.08	0.09	-0.02	0.32	0	0.31	0.09
	Mic	-0.06	-0.06	-0.05	0.01	0	-0.06	0.01	-0.05	-0.23	-0.22	-0.4	-0.15

improvement for macro-averaged F_1 is 0.09%. It is interesting to note that when 200 or less features are selected, R^2Cut strategy can always improve the macro-averaged F_1 for chi-square, and the improvement of F_1 in this case is 0.09% on average. The biggest improvement of macro-averaged F_1 is 1.04%, and the -0.23% is the worst case.

5) Table 5-2 also demonstrates R^2Cut strategy is good on macro-averaged F_1 but has negative effects on micro-averaged F_1, further study is needed to address the issue.

6) The 100% F_1 is reached when odd ratio is used to select 10000 features; and 99.82 % is reached when 5000 features are selected (Figure 5-10). Using information gain, F_1 value is as high as 99.34% if 80 or 100 features are chosen (Figure 5-8).

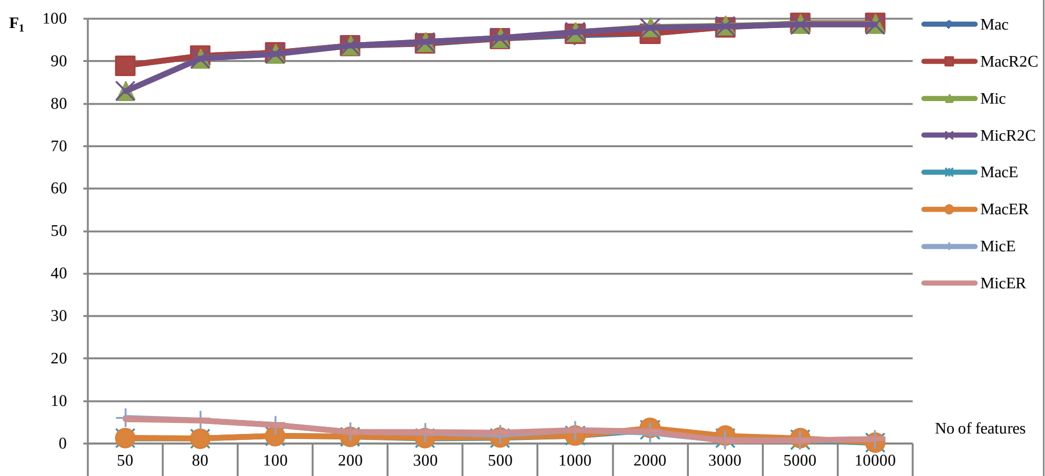
7) Based on the experimental results, it can be found that when Naïve Bayes is employed to classify documents, information gain and odd ration are two preferable feature selection algorithms. When information gain is used, 80 to 100 features are enough to ensure a satisfactory performance with F_1 above 99.3%; and if odd ration is used to pick up 300 to 500 features, the F_1 value is always bigger than 99.4%.

5.2.2 Experimental Results of k-Nearest Neighbours

kNN is tested not only against to the feature selection algorithms, number of features selected, R^2Cut smoothing strategy as discussed in the previous Section 5.2.1, but also against the number of k = 5, 10, 14, and the distance measure algorithms include cosine similarity and Euclidean distance.

Figure 5-11 to Figure 5-14[29] demonstrate the macro-averaged and micro-averaged F_1 for kNN when the feature selection algorithms chi-square, information gain, mutual information,

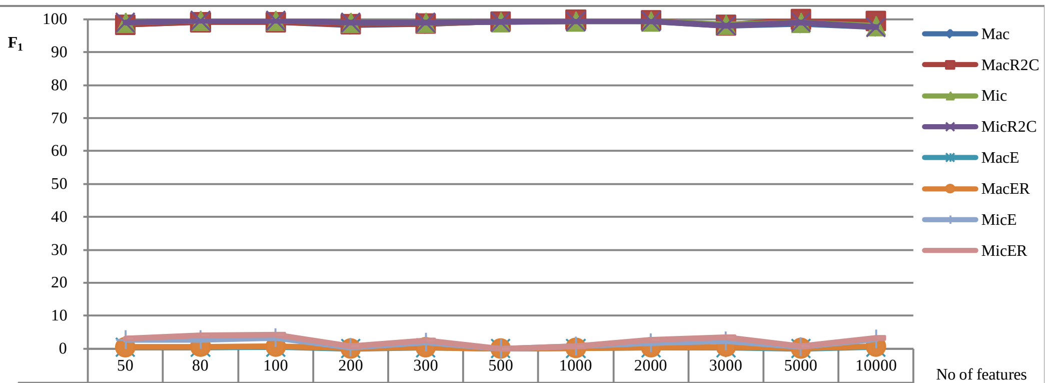
[29] Abbreviations in the figures: MacC/MacC, Macro/Micro-averaged using cosine similarity without R^2Cut smoothing strategy; MacR2CC/MinR2CC, Macro/Micro-averaged using cosine similarity with R^2Cut strategy; MacE/MacE, Macro/Micro-averaged using Euclidean distance without R^2Cut smoothing strategy; MacR2CE/MinR2CE, Macro/Micro-averaged using Euclidean distance with R^2Cut strategy;



	50	80	100	200	300	500	1000	2000	3000	5000	10000
Mac	88.97	91.34	92.04	93.68	94.19	95.36	96.15	96.58	98.08	98.8	98.79
MacR2C	88.97	91.34	92.04	93.68	94.26	95.36	96.52	96.54	98.03	99.02	98.99
Mic	83.01	90.62	91.79	93.75	94.53	95.5	96.88	98.05	98.44	98.83	98.83
MicR2C	83.01	90.62	91.79	93.75	94.53	95.5	96.91	97.87	98.25	98.64	98.65
MacE	1.22	1.12	1.73	1.57	1.23	1.31	1.83	3.19	1.3	0.87	0.15
MacER	1.26	1.12	1.73	1.57	1.29	1.47	1.83	3.63	1.81	1.2	0.24
MicE	6.07	5.48	4.31	2.73	2.54	2.14	3.13	2.54	0.78	0.78	0.98
MicER	5.75	5.48	4.31	2.73	2.69	2.61	3.08	2.9	0.61	0.68	1.1

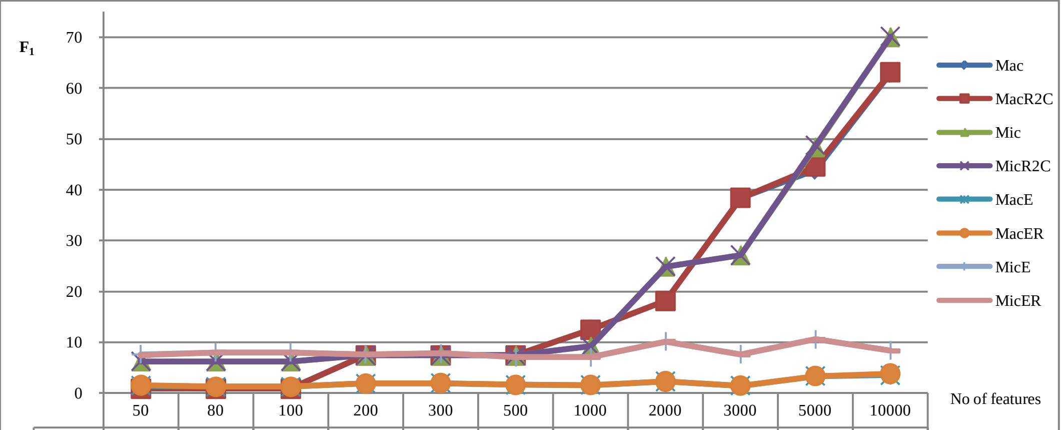
Figure 5-11 Micro/Macro averaged F_1 with/without R^2Cut smoothing strategy of kNN with chi-square feature selection algorithm for cosine and Euclidean measures (ρ = 0. 8)

and odd ratio are used to choose 50, 80, …, 10000 features. The figures also include the results for the two distance measure approaches: cosine similarity and Euclidean distance.



	50	80	100	200	300	500	1000	2000	3000	5000	10000
Mac	98.42	99.14	99.14	98.34	98.56	99.35	99.64	99.41	98.07	98.74	97.74
MacR2C	98.42	99.14	99.17	98.48	98.71	99.35	99.92	99.66	98.25	100	99.52
Mic	99.02	99.61	99.61	99.02	99.02	99.22	99.42	99.42	98.44	99.02	98.05
MicR2C	99.02	99.61	99.61	99.03	99.03	99.22	99.42	99.42	98.08	99.05	97.58
MacE	0.38	0.38	0.45	0	0.27	0	0.11	0.24	0.33	0	0.43
MacER	0.43	0.57	0.63	0.11	0.35	0	0.17	0.39	0.55	0.14	0.73
MicE	2.73	2.73	3.32	0.39	1.95	0	0.78	1.75	2.34	0.58	2.93
MicER	3.11	4.05	4.28	0.75	2.49	0	0.73	2.59	3.52	0.75	3.23

Figure 5-12 Micro/Macro averaged F_1 with/without R^2Cut smoothing strategy of kNN with information gain feature selection algorithm for cosine and Euclidean measures (ρ = 0. 8)

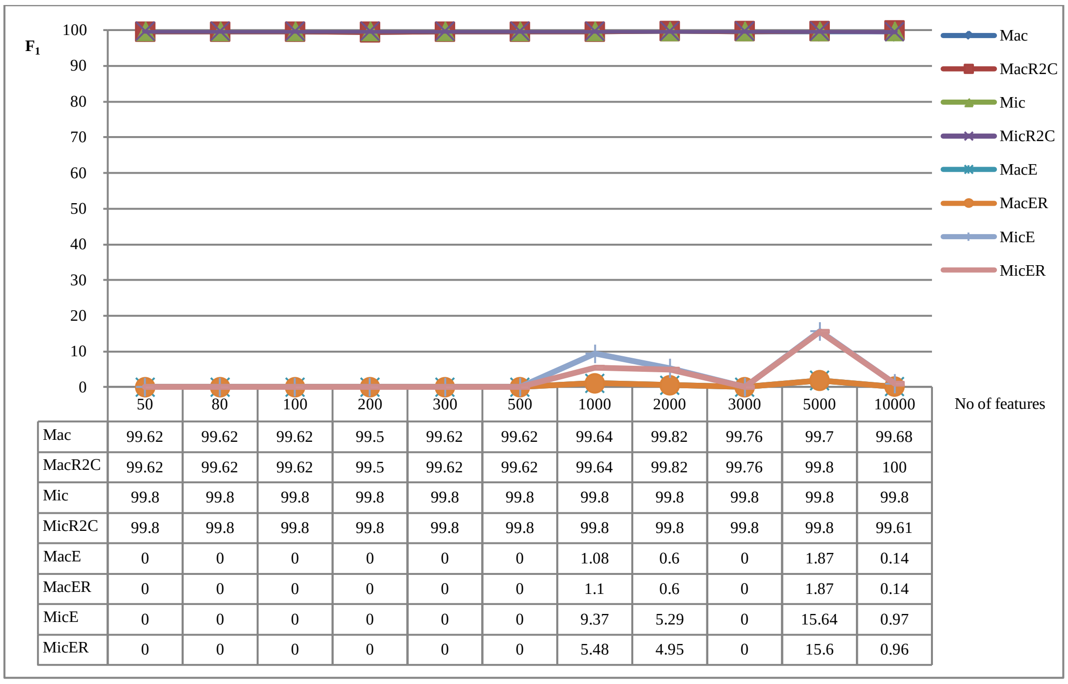


	50	80	100	200	300	500	1000	2000	3000	5000	10000
Mac	0.84	0.84	0.84	7.43	7.43	7.43	12.46	18.16	38.4	43.91	62.95
MacR2C	0.84	0.84	0.84	7.43	7.43	7.43	12.46	18.16	38.4	44.56	63.11
Mic	6.25	6.25	6.25	7.42	7.42	7.42	9.17	24.92	27.14	48.42	70.1
MicR2C	6.25	6.25	6.25	7.42	7.42	7.42	9.17	24.92	27.14	48.74	70.15
MacE	1.46	1.23	1.24	1.88	1.97	1.64	1.58	2.28	1.45	3.31	3.52
MacER	1.53	1.23	1.24	1.88	1.97	1.64	1.58	2.28	1.45	3.31	3.8
MicE	7.62	8.01	8.01	7.61	7.81	7.02	7.03	10.15	7.62	10.55	8.4
MicER	7.42	8.01	8.01	7.61	7.81	7.02	7.03	10.15	7.62	10.55	8.28

Figure 5-13 Micro/Macro averaged F_1 with/without R^2Cut smoothing strategy of kNN with mutual information feature selection ($\rho = 0.8$)

The following observations can be obtained from the data shown in the figures.

1) The number of features is an essential factor which dominants the performance of kNN in terms of F_1. Similar to NB, for information gain, features range from 50 to 500 enable kNN to perform very well (Figure 5-12). For chi-square, kNN reaches its best performance when features range from 500 to 2000 (Figure 5-11). Mutual information is not a good feature selection algorithm for kNN, and it seems performing better with more features (Figure 5-13).
2) Odd ratio is one of the best feature selection algorithms for kNN as well. It outperforms chi-square and mutual information, slightly better than information gain, and the preferred feature range for kNN is from 300 to 500 (Figure 5-14).
3) Although conceptually simple, kNN, if combined with Odds Ratio and information gain, can perform extremely well. When R^2Cut thresholding strategy is used and 1000 features are selected using information gain, the micro-averaged F_1 is as high as 99.92% (Figure 5-12). And what is astonishing is that when odd ration is employed to selected features, the F_1 values are always above 99.5%; and when 200 features are chosen, the F_1 is reached its submit at 99.82% (Figure 5-14).
4) Compared with the one document one category strategy, the R^2Cut smoothing strategy can also marginally improve the performance of kNN as the case of Naïve Bayes. To estimated the effect of R^2Cut strategy, experimental results are obtained when ρ = 0.1, 0.2, 0.3, 0.4, 0.5, 0.8 and 1.0. For the five cases, the peak performance is reached when ρ



	50	80	100	200	300	500	1000	2000	3000	5000	10000
Mac	99.62	99.62	99.62	99.5	99.62	99.62	99.64	99.82	99.76	99.7	99.68
MacR2C	99.62	99.62	99.62	99.5	99.62	99.62	99.64	99.82	99.76	99.8	100
Mic	99.8	99.8	99.8	99.8	99.8	99.8	99.8	99.8	99.8	99.8	99.8
MicR2C	99.8	99.8	99.8	99.8	99.8	99.8	99.8	99.8	99.8	99.8	99.61
MacE	0	0	0	0	0	0	1.08	0.6	0	1.87	0.14
MacER	0	0	0	0	0	0	1.1	0.6	0	1.87	0.14
MicE	0	0	0	0	0	0	9.37	5.29	0	15.64	0.97
MicER	0	0	0	0	0	0	5.48	4.95	0	15.6	0.96

Figure 5-14 Micro/Macro averaged F_1 with/without R^2Cut smoothing strategy of kNN with Odds ratio feature selection algorithm (ρ = 0. 8)

= 0.8. The improvement of R^2Cut is illustrated in Table 5-3. As can be seen from the table, the average improvement of F_1 over the four feature selection algorithms is 0.13% for macro-averaged F_1, and -0.03% for micro-averaged F_1. The maximum improvement is 1.78%.

5) Euclidean distance performs very poorly when combined with tf-idf strategy. The macro-averaged (micro-averaged) F_1 of its best with respect to the different features selected is only 10.55 (mutual information).

6) To evaluate the effectiveness of k, mutual information algorithm is employed to select

Table 5-3 The improvement of F_1 of kNN with / without R^2Cut smoothing strategy (%).

Features		50	80	100	200	300	500	1000	2000	3000	5000	10000	AVG
Chi-square	Mac	0	0	0	0	0.07	0	0.37	-0.04	-0.05	0.22	0.2	0.07
	Mic	0	0	0	0	0	0	0.03	-0.18	-0.19	-0.19	-0.18	-0.06
	μ	0	0	0	0	0.04	0	0.2	-0.11	-0.12	0.02	0.01	0.01
Info. Gain	Mac	0	0	0.03	0.14	0.15	0	0.28	0.25	0.18	1.26	1.78	0.37
	Mic	0	0	0	0.01	0.01	0	0	0	-0.36	0.03	-0.47	-0.07
	μ	0	0	0.02	0.08	0.08	0	0.14	0.13	-0.09	0.65	0.66	0.15
Mutual Info.	Mac	0	0	0	0	0	0	0	0	0	0.65	0.15	0.07
	Mic	0	0	0	0	0	0	0	0	0	0.32	0.05	0.03
	μ	0	0	0	0	0	0	0	0	0	0.49	0.1	0.05
Odd ratio	Mac	0	0	0	0	0	0	0	0	0	0.1	0.32	0.04
	Mic	0	0	0	0	0	0	0	0	0	0	-0.19	0.01
	Ave	0	0	0	0	0	0	0	0	0	0.05	0.07	0.03
AVG	Mac	0	0	0.01	0.04	0.06	0	0.16	0.05	0.03	0.55	0.57	0.13
	Mic	0	0	0	0	0	0	0	-0.05	-0.16	0.04	-0.2	-0.03

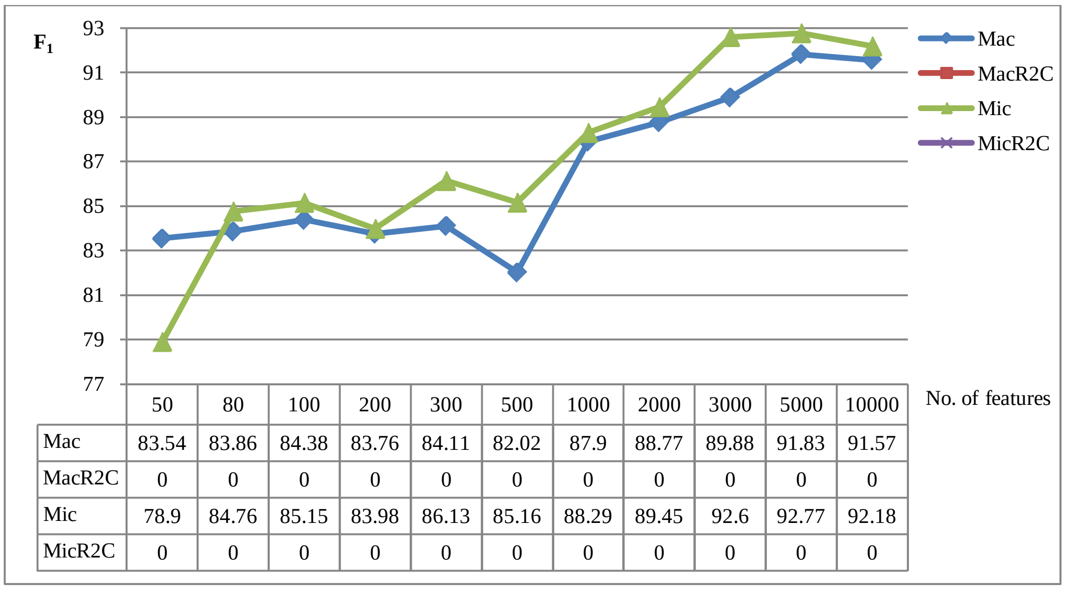
2000 features. When k is 10, the values of Mac, MacR2C, Mic, MicR2C are 24.13%, 24.13%, 29.9%, and 29.9% respectively; when k = 14, the corresponding values are 21.85%, 21.85%, 25.62%, 25.62. As can be seen from the data, a bigger but not the biggest (same as the number of categories) k can improve the performance of kNN in this research. However, a further comprehensive evaluation is needed in the future.

5.2.3 Experimental Results of AdaBoost

For AdaBoost.MHR, in addition to the number of features selected by the four feature selection algorithms, another two adjustable parameters are the number of training rounds T and the ρ of the R^2Cut smoothing strategy. Training an AdaBoost classifier is time consuming. Limited by the computational power of the personal computer used to conduct the experiments, the parameter T is chosen as 10, although no less than 1000 is more preferable (Schapire and Singer 2000).

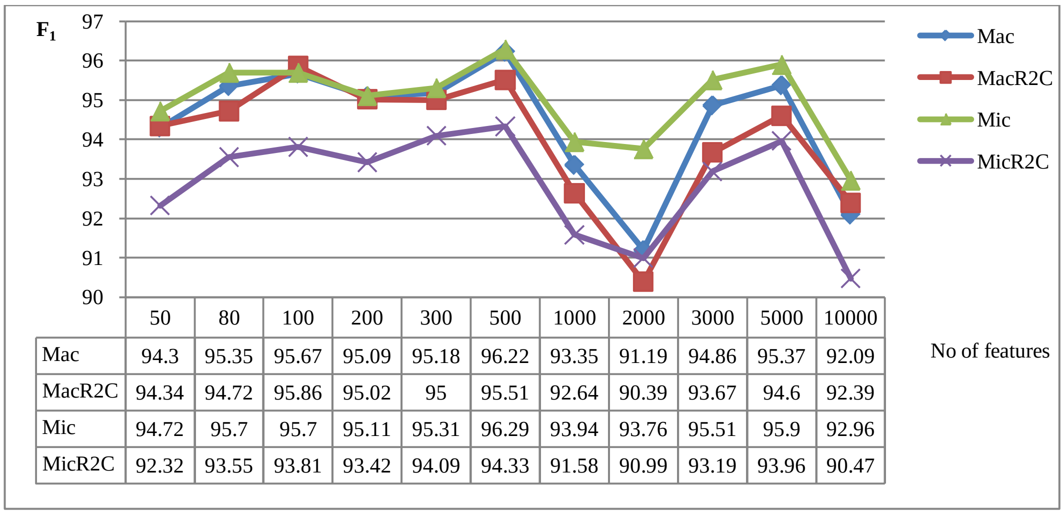
Figure 5-15 to Figure 5-17 demonstrate the experimental results of AdaBoost.MHR when 50, 80, 100, 200, 300, 500, 1000, 2000, 3000, 5000, and 10000 features are selected by chi-square, information gain, mutual information, and Odds Ratio feature selection algorithms. Based on the experimental results in these figures, the following observations can be derived.

1) Similar to the previously evaluated two algorithms, the performance of F_1 is dominated by the number of features selected by the different feature selection algorithms. AdaBoost.MHR reaches its peak 92.77% in terms of micro-averaged F_1 when 5000 features are selected by chi-square (Figure 5-15). When 500 features are chosen by



	50	80	100	200	300	500	1000	2000	3000	5000	10000
Mac	83.54	83.86	84.38	83.76	84.11	82.02	87.9	88.77	89.88	91.83	91.57
MacR2C	0	0	0	0	0	0	0	0	0	0	0
Mic	78.9	84.76	85.15	83.98	86.13	85.16	88.29	89.45	92.6	92.77	92.18
MicR2C	0	0	0	0	0	0	0	0	0	0	0

Figure 5-15 Micro/Macro averaged F_1 with/without R^2Cut smoothing strategy of AdaBoost.MHR algorithm with chi-square feature selection algorithm (ρ = 4.8)



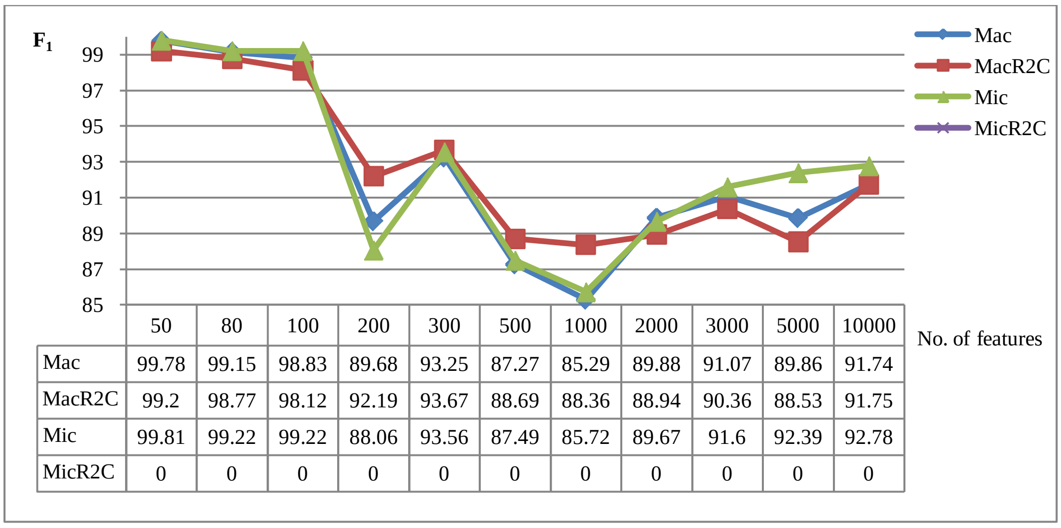
	50	80	100	200	300	500	1000	2000	3000	5000	10000
Mac	94.3	95.35	95.67	95.09	95.18	96.22	93.35	91.19	94.86	95.37	92.09
MacR2C	94.34	94.72	95.86	95.02	95	95.51	92.64	90.39	93.67	94.6	92.39
Mic	94.72	95.7	95.7	95.11	95.31	96.29	93.94	93.76	95.51	95.9	92.96
MicR2C	92.32	93.55	93.81	93.42	94.09	94.33	91.58	90.99	93.19	93.96	90.47

Figure 5-16 Micro/Macro averaged F_1 with/without R^2Cut smoothing strategy of AdaBoost.MHR algorithm with information gain feature selection algorithm (ρ = 7.6)

information gain, 96.29% is the best micro-averaged F_1 obtained (Figure 5-16). For mutual information, micro-averaged F_1 reaches its highest level 44.14% (Figure 5-18). When using Odds Ratio selects 50 features, AdaBoost.MHR performs best with micro-averaged F_1 as high as 99.81% Figure 5-17.

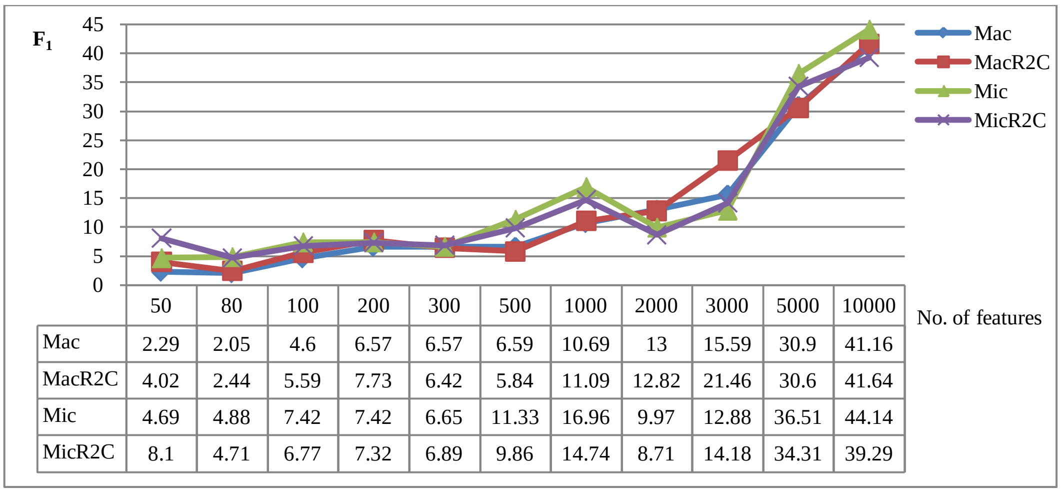
2) As in the previous cases, odd ratio is once again the best feature selection algorithm for AdaBoost.MHR in that it yields highest F_1 values of 99.81% and 99.78% for micro-averaging and micro-averaging measures respectively (Figure 5-17)

3) As in the previous cases, odd ration is once again the best feature selection algorithm for AdaBoost.MHR in that it yields highest F_1 values of 99.81% and 99.78% for micro-



	50	80	100	200	300	500	1000	2000	3000	5000	10000
Mac	99.78	99.15	98.83	89.68	93.25	87.27	85.29	89.88	91.07	89.86	91.74
MacR2C	99.2	98.77	98.12	92.19	93.67	88.69	88.36	88.94	90.36	88.53	91.75
Mic	99.81	99.22	99.22	88.06	93.56	87.49	85.72	89.67	91.6	92.39	92.78
MicR2C	0	0	0	0	0	0	0	0	0	0	0

Figure 5-17 Micro/Macro averaged F_1 with/without R^2Cut smoothing strategy of AdaBoost.MHR algorithm with odds ratio feature selection algorithm (ρ = 7.6)



	50	80	100	200	300	500	1000	2000	3000	5000	10000
Mac	2.29	2.05	4.6	6.57	6.57	6.59	10.69	13	15.59	30.9	41.16
MacR2C	4.02	2.44	5.59	7.73	6.42	5.84	11.09	12.82	21.46	30.6	41.64
Mic	4.69	4.88	7.42	7.42	6.65	11.33	16.96	9.97	12.88	36.51	44.14
MicR2C	8.1	4.71	6.77	7.32	6.89	9.86	14.74	8.71	14.18	34.31	39.29

Figure 5-18 Micro/Macro averaged F_1 with/without R^2Cut smoothing strategy of AdaBoost.MHR algorithm with mutual information feature selection algorithm

averaging and micro-averaging measures respectively (Figure 5-17).

4) The R^2Cut smoothing strategy marginally degrades the performance of AdaBoost.MHR in terms of F_1. As illustrated in Table 5-4, when 1000 or 200 features are selected by Odds Ratio, R^2Cut can improve macro-averaged F_1 by 3.07% and 2.51% respectively; in other case, it either only slightly improves, or slightly negatively affects the performance of AdaBoost.MHR.

5) The overall performance of AdaBoost.MHR is not as effective as Naïve Bayes and kNN. One of the possible reasons is that the parameter T is too small. To verify this assumption, T = 10, 100, 1000, and 10000 are tested against mutual information feature selection algorithm with only 100 features are chosen. In this scenario, the micro-averaged F_1 for different training rounds are 7.42%, 8.2%, 5.66% and 5.66%. In this case, the F_1 for Naïve Bayes and kNN are 21.11% and 6.25% respectively.

6) This reveals that AdaBoost.MHR outperforms kNN but is inferior to Naïve Bayes when using mutual information to select 100 features.

7) Another reason for Naïve Bayes and kNN outperforms AdaBoost.MHR might be the

Table 5-4 The improvement of F_1 of AdaBoost.MHR with / without R^2Cut smoothing strategy (%).

Features		50	80	100	200	300	500	1000	2000	3000	5000	10000	AVE
Info. Gain	Mac	0.04	-0.63	0.19	-0.07	-0.18	-0.71	-0.71	-0.8	-1.19	-0.77	0.3	-0.41
	Mic	-	-	-	-	-	-	-	-	-	-	-	-
	μ	-	-	-	-	-	-	-	-	-	-	-	-
Odd ratio	Mac	-0.58	-0.38	-0.71	2.51	0.42	1.42	3.07	-0.94	-0.71	-1.33	0.01	0.25
	Mic	-	-	-	-	-	-	-	-	-	-	-	-
	μ	-	-	-	-	-	-	-	-	-	-	-	-
AVG	Mac	-0.27	-0.5	-0.26	1.22	0.12	0.36	1.32	-0.87	-0.99	-1.05	0.17	-0.07
	Mic	-	-	-	-	-	-	-	-	-	-	-	-

ODP data set is more preferable for the former two text categorization algorithms. However, compared with the results provided by Schapire and Singer (2000) in there the data set Reuters-21578, AP Title and UseNet Data are tested against, Naïve Bayes and kNN can produce higher F_1 results by using ODP *categoryDocuments*.

5.2.4 Efficiency of the Feature Selection Algorithms / Classifiers

Table 5-5 presents the time taken for different feature selection algorithms when selecting 50, 80, 100, 200, 300, 500, 1000, 2000, 3000, 5000, and 10000 features.

As can be seen from the table, for the four different feature selection algorithms,

1) Mutual information is the most efficient feature selection algorithm, however, if this algorithm is employed to select features, text categorization algorithms will yield poor effectiveness in terms of F_1 as discussed in previous sections;
2) Odds ratio is the second most efficiency feature selection algorithm and it provided the best performance for the text categorization algorithms evaluated in previous sections, it is almost as efficient as mutual information.
3) Chi-square is computationally expensive to pick up features compared with mutual information and Odds Ratio; for example, when 1000 features are selected, Odds Ratio is nearly three time fast than chi-square $((19*60+17)/(6*60+28) \approx 2.98)$
4) Information gain is the most inefficiency feature selection algorithm. It takes more than 13 times longer than Odds Ratio $((1*60*60+27*60+53) / (6*60+28) \approx 13.59)$ when 1000 features are selected.

In summary, it is now clear that Odds Ratio is not only the most effective feature selection method for text categorization algorithms estimated so far, it is also one of the most efficient feature selection algorithms. It is only slightly slower than the most efficient nevertheless also the most ineffective mutual information feature selection strategy.

Table 5-5 Time taken by the feature selection algorithms (hh:mm:ss)

#features	**Chi-square**	**Information gain**	**Mutual information**	**Odds ratio**
50	00:00:34	00:02:59	00:00:27	00:02:34
80	00:00:58	00:05:39	00:00:34	00:02:43
100	00:01:10	00:06:38	00:00:39	00:02:47
200	00:02:31	00:15:09	00:01:01	00:03:21
300	00:04:04	00:25:25	00:01:25	00:03:43
500	00:06:40	00:38:55	00:02:15	00:04:28
1000	00:19:17	01:27:53	00:04:15	00:06:28
2000	00:47:08	03:39:47	00:08:14	00:10:29
3000	00:56:47	05:41:24	00:12:04	00:14:37
5000	01:57:51	09:24:18	00:19:48	00:23:11
10000	05:26:41	16:11:31	00:42:10	00:46:55

5.3 Experimental Results Based on S-Set

There are two reasons to present experimental results based on S-Set. The first one is that some algorithms such as AdaBoost and SVMs are computationally expensive to train and may take months to evaluate if the generated ODP data set is employed as a benchmark document collection using a desktop computer. It is very hard and not necessary to adjust all possible parameters to obtain an encyclopedia-like experimental results set in this research. The second reason is to verify how the performance of different text categorization algorithms are data set dependent. That is, to check if one text classifier and a feature selection algorithm which performs better when using ODP *categoryDocuments* as a data set can also yield consistently good results if the data set is S-Set.

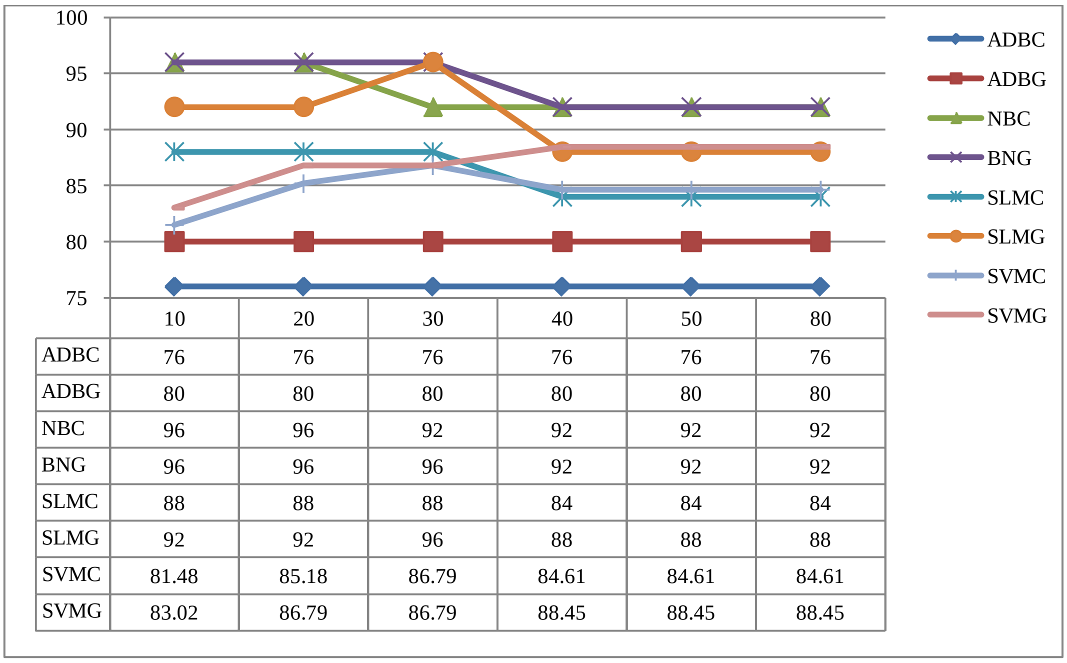
The S-Set contains 77 terms, the features selected by the feature selection algorithms are subsequently 80, 50, 40, 30, 20, and 10 because it can never select more than 77 features, especially when considering that a training data set is only part of the S-Set and should contain no more than 77 features.

In the following subsections, experimental results of AdaBoost.MHR, SVMLight and Naïve Bayes are provided at first; then, the performance of the seven feature selection algorithms are evaluated against the Naïve Bayes and kNN text categorization algorithms; the effectiveness of the proposed R^2Cut smoothing strategy is estimated subsequently; and finally, the experimental results of the Z-tfidf feature weighting approach are presented.

5.3.1 AdaBoost, Naïve Bayes, SVMLight, and Statistical Language Model

The experimental results of Adaboost.MHR, Naïve Bayes, SVMLight, and statistical language model are provided when chi-square and GSS coefficient are used to select features. Experimental results of Naïve Bayes and chi-square are taken as base line to estimate the performance of text classifiers and feature selection algorithms. Figure 5-19 demonstrates micro-averaged F_1 when 100, 50, 40, 30, 20, and 10 features are selected using the above three feature selection algorithms.

The figure reveals that Naïve Bayes is the most effectiveness text categorization algorithm compared with AdaBoost.HMR and SVMLight based on S-Set; although it is the most simple and most efficient one. This results support the observation in the previous section when testing document collection is the generated second level ODP *categoryDocument* set.



	10	20	30	40	50	80
ADBC	76	76	76	76	76	76
ADBG	80	80	80	80	80	80
NBC	96	96	92	92	92	92
BNG	96	96	96	92	92	92
SLMC	88	88	88	84	84	84
SLMG	92	92	96	88	88	88
SVMC	81.48	85.18	86.79	84.61	84.61	84.61
SVMG	83.02	86.79	86.79	88.45	88.45	88.45

Figure 5-19 Micro-averaged F_1 of Naïve Bayes, SVMLight, and AdaBoost.MHR with chi-square and GSS coefficient feature selection algorithms

ADBC/ADBG: AdaBoost.MHR + chi-square/GSS; NBC/NBG: Naïve Bayes + chi-square/GSS; SLMC/SLMG: Statistical Language Model + chi-square/GSS; SVMC/SVMG: SVMLight + chi-square/GSS

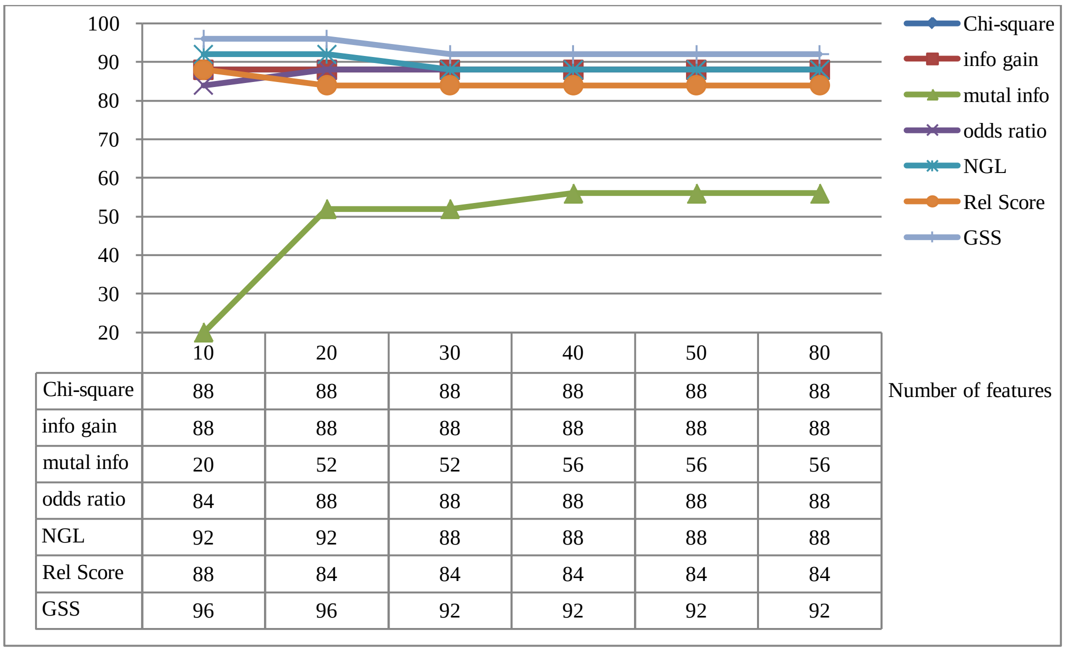
AdaBoost.MHR does not perform as well as expected primarily because of the limited number of documents and terms in S-Set, whereas a larger document set is more suitable for the algorithm. The performance of SVMLight is not as well as that of the Naïve Bayes, but better than the AdaBoost.

The performance of statistics language model is only slightly inferior to Naïve Bayes but outperforms the other two text categorization algorithms when S-Set is employed. Further research is planned to compare the performance of the model with different smoothing strategy and R^2Cut thresholding algorithm when ODP *categoryDocuments* are taken as document set.

5.3.2 Feature Selection Algorithms

Chapter 2 discussed seven feature selection algorithms; in this section, all of the algorithms are to be evaluated based on S-Set. Naïve Bayes and kNN are taken as the text classifiers, and the experimental results are presented in Figure 5-20 and Figure 5-21.

Figure 5-20 presents the micro-averaged F_1 for the seven feature selection algorithms when kNN classifier is evaluated. The micro-averaged F_1 over the six different numbers of features in descending order are 93.33% for GSS coefficient; 89.33% for NGL coefficient; 88.00% for chi-square and information gain; 87.33% for Odds Ratio; and once again, mutual

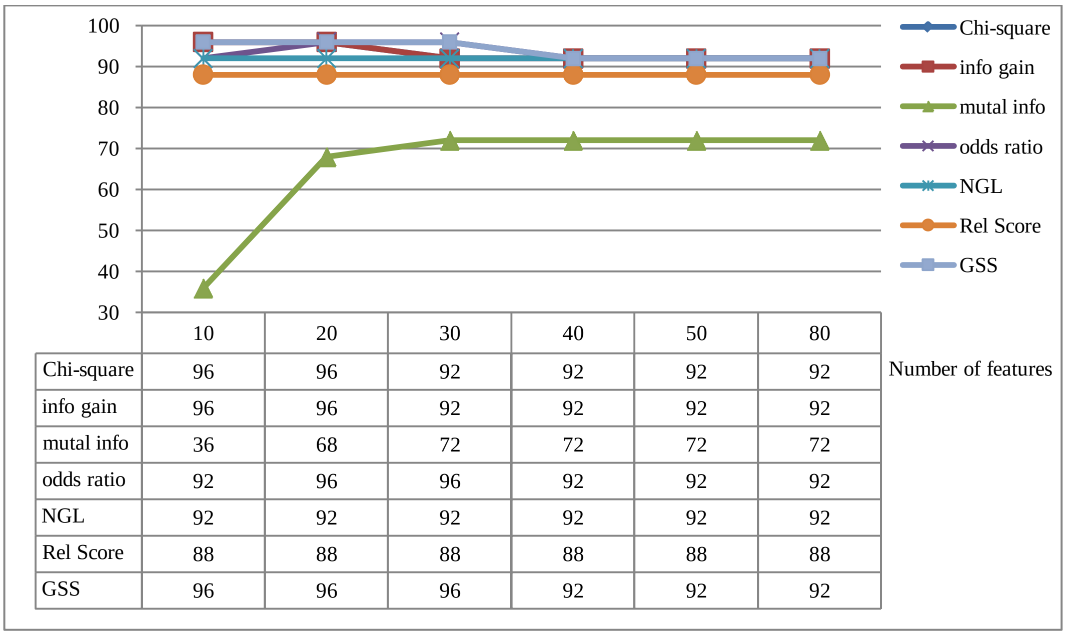


	10	20	30	40	50	80
Chi-square	88	88	88	88	88	88
info gain	88	88	88	88	88	88
mutal info	20	52	52	56	56	56
odds ratio	84	88	88	88	88	88
NGL	92	92	88	88	88	88
Rel Score	88	84	84	84	84	84
GSS	96	96	92	92	92	92

Figure 5-20 Micro-averaged F_1 of kNN (tfidf + cosine similarity) with chi-square, information gain, mutual information, NGL coefficient, Relevancy score, and GSS coefficient.

information is ranked at last with a low averaged F_1 value, 48.67%. This ranked list is almost the same as the one when Naïve Bayes is evaluated.

Experimental results of Naïve Bayes as shown in Figure 5-21 illustrates that GSS coefficient is the best feature selection algorithm and outperforms the other six approaches. The micro-



	10	20	30	40	50	80
Chi-square	96	96	92	92	92	92
info gain	96	96	92	92	92	92
mutal info	36	68	72	72	72	72
odds ratio	92	96	96	92	92	92
NGL	92	92	92	92	92	92
Rel Score	88	88	88	88	88	88
GSS	96	96	96	92	92	92

Figure 5-21 Micro-averaged F_1 of Naïve Bayes with chi-square, information gain, mutual information, NGL coefficient, Relevancy score, and GSS coefficient.

averaged F_1 value over the six different numbers of features chosen for GSS is 94.00%. The averaged values in descending order for the other feature selection algorithms are 93.33% for chi-square, information gain, and Odds Ratio; 92.00% for NGL coefficient; 88.00% for Relevancy score; and lastly 65.33% for mutual information.

5.3.3 The Effectiveness of R^2Cut Smoothing Strategy

Table 5-6 presents the improvements of F_1 over the six different numbers of features (10, 20, 30, 40, 50, and 80) picked up by the seven different feature selection algorithms used by Naïve Bayes with ρ is set to 0.4. As can be seen from the table, R ^{2}Cut strategy can marginally enhance the performance of feature selection algorithms in terms of F_1 in most cases.

The averaged improvement over the seven different feature selection algorithms for six different numbers of features chosen is 1.56% in terms of F_1 when Naïve Bayes text classifier is used as base line classifier. Recall that when second level ODP *categoryDocuments* are used as benchmark data set, R^2Cut strategy has a negative influence on micro-averaged F_1. How to use R^2Cut to boost micro-averaged F_1 is to be addressed in the future.

It can be found from the Table 5-6 that the suggested R^2Cut smoothing strategy may occasionally impose negative effective to some of the feature selection algorithms. For

Table 5-6 The improvement of macro / micro-averaged F_1 of Naïve Bayes with / without R^2Cut smoothing strategy (%) over seven different feature selection algorithms.

Features		**10**	**20**	**30**	**40**	**50**	**80**	**AVG**
Chi-square	Macro-averaged	1.66	0.81	1.59	1.59	1.59	1.59	1.47
	Micro-averaged	0.67	-2.67	0.67	0.67	0.67	0.67	0.11
	μ	1.17	-0.93	1.13	1.13	1.13	1.13	0.79
Information. Gain	Macro-averaged	1.66	0.8	1.59	1.59	1.59	1.59	1.47
	Micro-averaged	0.67	-2.67	0.67	0.67	0.67	0.67	0.11
	μ	1.17	-0.94	1.13	1.13	1.13	1.13	0.79
Mutual Information.	Macro-averaged	11.42	13.66	10.35	9.15	9.15	9.15	10.48
	Micro-averaged	-0.1	1.52	-1.05	-2	-2	-2	-0.94
	μ	5.66	7.59	4.65	3.58	3.58	3.58	4.77
Odd ratio	Macro-averaged	3.25	0.77	1.64	1.58	1.58	1.58	1.73
	Micro-averaged	1.33	-2.67	0.67	0.67	0.67	0.67	0.22
	μ	2.29	-0.95	1.16	1.13	1.13	1.13	0.98
NGL coefficient	Macro-averaged	1.59	1.59	1.59	1.59	1.59	1.59	1.59
	Micro-averaged	0.67	0.67	0.67	0.67	0.67	0.67	0.67
	μ	1.13	1.13	1.13	1.13	1.13	1.13	1.13
Relevancy score	Macro-averaged	3.17	1.59	1.59	1.59	1.59	1.59	1.85
	Micro-averaged	1.33	0.67	0.67	0.67	0.67	0.67	0.78
	μ	2.25	1.13	1.13	1.13	1.13	1.13	1.32
GSS coefficient	Macro-averaged	1.66	1.63	1.63	1.58	1.58	1.58	1.61
	Micro-averaged	0.67	0.67	0.67	0.67	0.67	0.67	0.67
	μ	1.17	1.15	1.15	1.13	1.13	1.13	1.14
AVG		2.12	1.17	1.64	1.48	1.48	1.48	1.56

example, when information gain is used to estimate micro-averaged F_1, the R^2Cut strategy is obviously not suitable because it produced a -0.94% decrease of F_1. However, it is interesting to note that for macro-averaged F_1, the improvement is significant, as high as 10.48%, and the averaged improvement over micro-averaged and macro-averaged is 4.77%, the highest average improvement in terms of F_1 over all feature selection algorithms.

The experimental results presented in this section demonstrate more improvements are obtained compared with the improvement presented in section 5.2, Table 5-2. The verification of the effectiveness of R^2Cut smoothing strategy in both small and large data sets makes it clear that the strategy is applicable in a wide range of data set with different sizes. However, once again, further research on how to use R^2Cut to improve micro-averaged F_1 is needed.

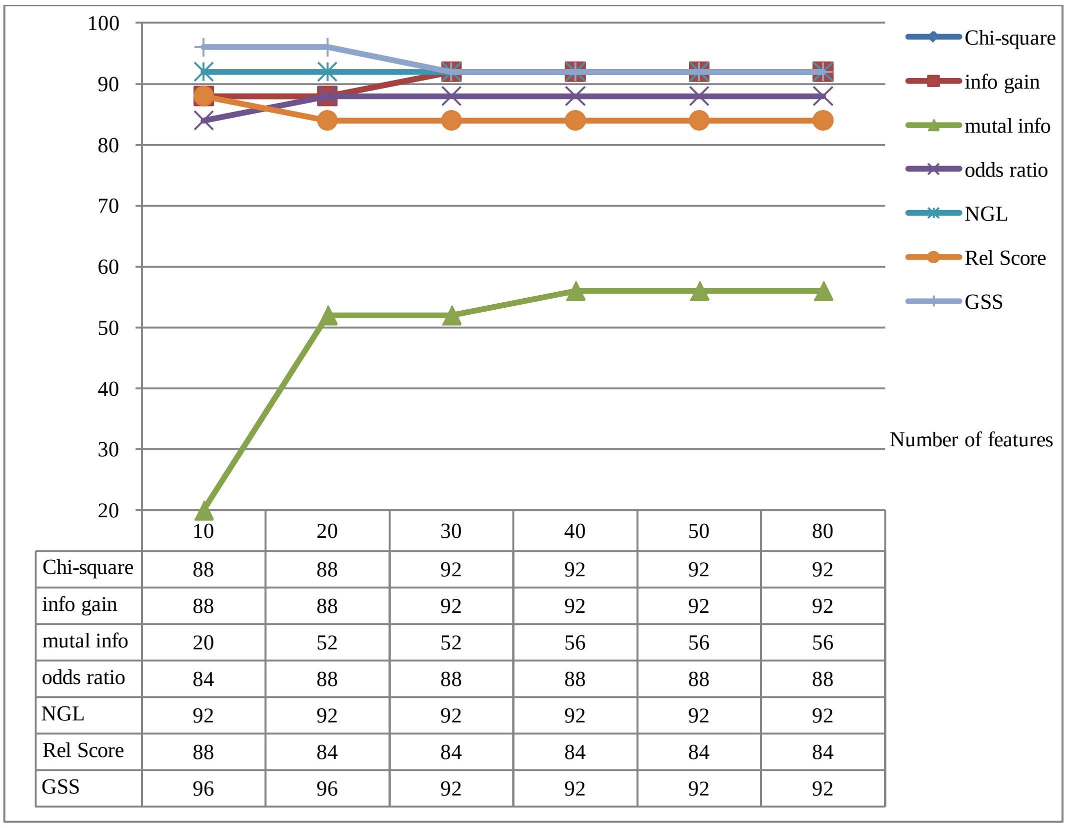
An attractive aspect of R^2Cut strategy is that it can improve the performance of a text categorization algorithm even if the algorithm itself can produce very good results. For example, with GSS coefficient feature selection algorithm, the macro-average and micro-averaged F_1 on average over the six different numbers of features are as high as 92.58% and 94.0% respectively. However, with the R^2Cut strategy, the corresponding F_1 are 94.19% and 94.67%. It is easy to understand R^2Cut strategy can improve the performance of a text categorization algorithm when F_1 is low because there is enough space to improve. The ability of R^2Cut strategy to improve a categorization algorithm stems from the mechanism that R^2Cut adjust true positive, false positive, and false negative only in case the algorithm is not very confident to arrange the top two ranked categories. For a given document **d**, Naïve Bayes estimates a ranked list of probabilities of **d** to be correctly assigned to categories $\{c_i| \ i = 1, 2, \ldots |C|\}$. If the probability of the top ranked category is bigger enough than the probability of the second ranked category, Naïve Bayes in this case is quite confident to make the assignment decision, and R^2Cut agrees with the decision as well. On the other hand, if the difference of the probabilities of the two top-ranked categories is very small, it implies in this scenario that Naïve Bayes classifier is not confident to predict the rank of the two categories. Defining the confidence of a decision made by a categorization algorithm as the relative difference of values assigned to the two top ranked categories, R^2Cut smoothing strategy takes this factor into account and modifies the final category prediction if the confidence of the classifier is relatively low to make a proper decision. Since the confidence is a relative concept, the R^2Cut strategy consequently has the ability to enhance a classifier even if it can produce satisfactory results, and as can be expected, the improvement is only marginally in this circumstance.

To obtain the best performance when using R^2Cut thresholding strategy, the parameter ρ has to be adjusted. If ρ is too small, it imposes little impact on the performance of a text categorization algorithm; on the other hand, if ρ is too large, it may impose negative effect and even deteriorate the performance of a classifier.

5.3.4 The Effectiveness of Z-tfidf Term-weighting Approach

Figure 5-22 presents the micro-averaged F_1 of kNN when using Z-tfidf term-weighting strategy and cosine value to estimate the similarity between two documents. Compared with the results presented in Figure 5-21, the data in the figure reveal that Z-tfidf term-weighting approach improves the performance of kNN when features are selected by chi-square, information gain, and NGL coefficient; whereas yields the identical results when mutual information, Odds Ratio, relevancy score, and GSS coefficient feature selection algorithms are employed. The improvements based on the three feature selection algorithms over the six different numbers of features chosen are the same for all 2.67%. The overall averaged F_1 over all different numbers of features is 82.76% if tf-idf is used, and 83.9% when Z-tfidf is employed. The improvement is on average 1.14%.

If the R^2Cut smoothing strategy is employed and parameter ρ is properly adjusted, another marginal improvement can be expected.



	10	20	30	40	50	80
Chi-square	88	88	92	92	92	92
info gain	88	88	92	92	92	92
mutal info	20	52	52	56	56	56
odds ratio	84	88	88	88	88	88
NGL	92	92	92	92	92	92
Rel Score	88	84	84	84	84	84
GSS	96	96	92	92	92	92

Figure 5-22 Micro-averaged F_1 of kNN (Z-tfidf + cosine similarity) with chi-square, information gain, mutual information, NGL coefficient, Relevancy score, and GSS coefficient.

5.4 Categorization of Web Snippets

To verify the applicability of the generated ODP *categoryDocuments* in the real world, in this section, 50 Web snippets obtained as the search results from *Google Directory* (http://directory.google.com/) are taken as the test data set. *Google Directory* has an identical Web directory to ODP, and each of the results from Google Directory has an extra line indicates which category the Web snippet belongs to.

The category given by *Google Directory* is to be compared with the category assigned by AdaBoost.MHR, kNN, and Naïve Bayes classifiers that are trained by the second level ODP *categoryDocument* set. Supposed that the category given by *Google Directory* is the true directory, the performance of the three classifiers trained by the generated second level ODP *categoryDocuments* set is evaluated and presented. It is worthy to note that assigning a Web snippet to a category is a subjective matter rather than an objective matter; it is not appropriate to take the assigned category of *Google directory* as the true, unique correct assignment, and to conclude that the category assigned by the above text classifiers trained by the second level ODP *categoryDocuments*, or other text classifiers, is not appropriate.

5.4.1 Testing Data Set Selection

Fifty Web snippets are selected from the search results of *Google Directory* when “jaguar” is used as a search term. The home page of *Google Directory* shows it has an identical Web directory as that of the ODP. However, if a Web snippet is assigned to category “World”, “Adult” or “Regional”, the snippet is not considered because the generated OPD *categoryDocument* set takes account of only the 14 top ODP categories as described in the previous section. It can be found that some of the categories assigned by *Google Directory* to the Web snippets are beyond the range of the 14 top-level ODP categories, for example, the category *Jeux>Jeux vidéo>Consoles* (French) which is assigned to one of the returned Web snippets is not included in the 14 top-level ODP directory.

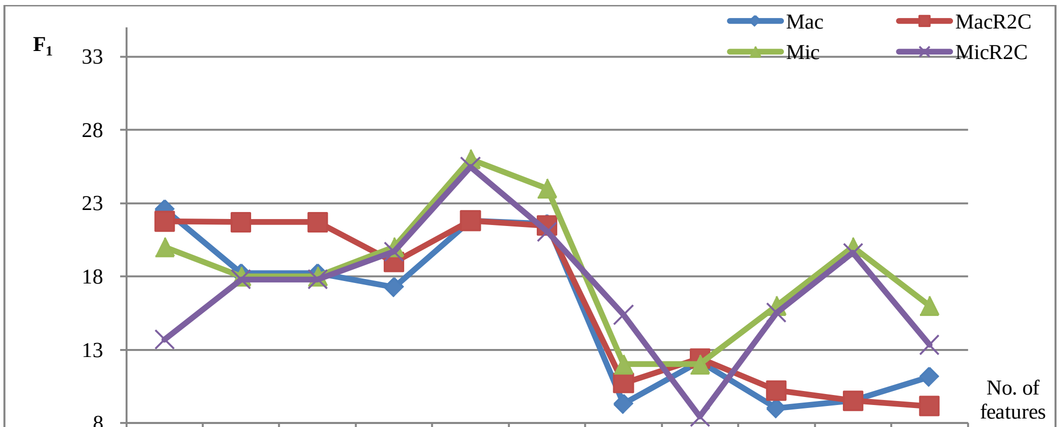
Google Directory is powered by ODP category; nevertheless, the search results from *Google Directory* and ODP are quite different. For example, when “jaguar” is used as a search term, Google has potentially 60 million search results available, and actually can return 1000 items, whereas ODP returns only 671 items (retrieved on January 21, 2010). Compared the results of first page of the two search engines, there are only two results that refer to the same Web site (Appendix 2). Another difference between the two sets of results is that all the returned items of ODP contain the search term “jaguar”, whereas the results of *Google Directory* may not contain the search term. In fact, for the 50 Web snippets selected in this research, none of them contains the search term “jaguar”.

The 50 search results of *Google Directory* are selected by following a reverse rank order. The 1000-th Web snippet is checked first to verify if the category assigned to the snippet is included in the top 14 ODP categories; if it is, the item is chosen, else the item is ignored. Then the 999-th is checked, and so on and so forth, until 50 items are picked up. The selected 50 items are presented in Appendix 2. Using the reverse order selection approach to select the results is to alleviate the relatedness of the returned items with the search-term “jaguar” and hope the results are independently distributed.

5.4.2 Experimental Results

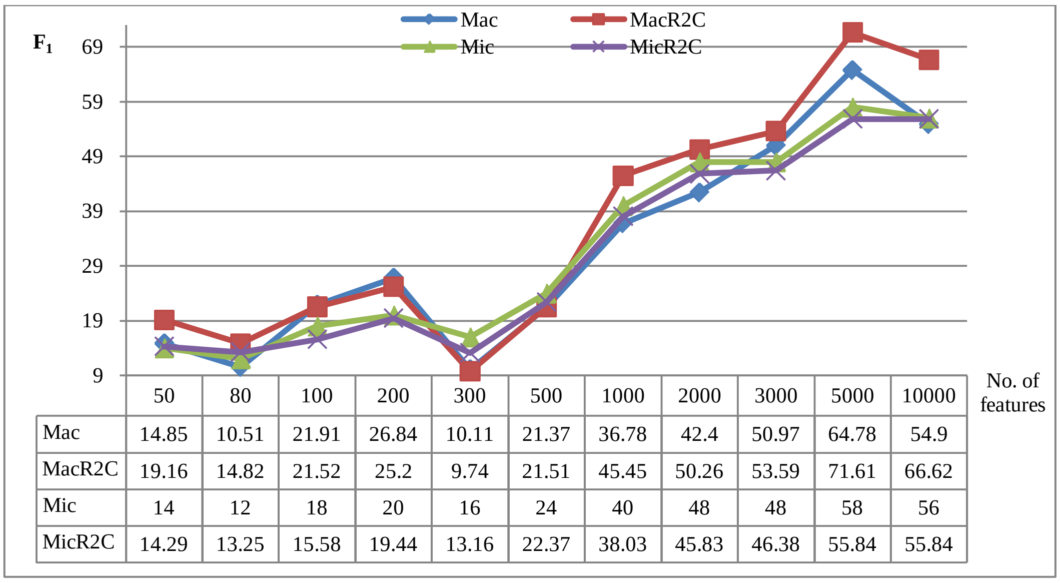
Experimental results are presented in Figure 5-23 to Figure 5-24 for AdaBoost.MHR, kNN, and Naïve Bayes classifiers. For each of the three text classifiers, different feature selection algorithms are evaluated, and the one which produces the highest averaged F_1 over micro-averaged, macro-averaged, with/without R^2Cut thresholding strategy are selected. Experimental results in Figure 5-23 are produced by using Odds Ratio feature selection algorithm; results in Figure 5-24 are obtained when relevancy score feature selection algorithm is used; and results in Figure 5-25 is generated by using GSS Coefficient feature selection algorithm.

The parameter ρ as presented in the figures for R^2Cut smoothing strategy has been adjusted for different categorization algorithms, and the one that gives best results are presented.



	50	80	100	200	300	500	1000	2000	3000	5000	10000
Mac	22.57	18.2	18.2	17.27	21.8	21.57	9.31	12.21	9.01	9.52	11.16
MacR2C	21.76	21.71	21.71	18.98	21.8	21.47	10.72	12.4	10.21	9.52	9.15
Mic	20	18	18	20	26	24	12	12	16	20	16
MicR2C	13.7	17.8	17.8	19.7	25.49	21.05	15.38	8.43	15.52	19.6	13.33

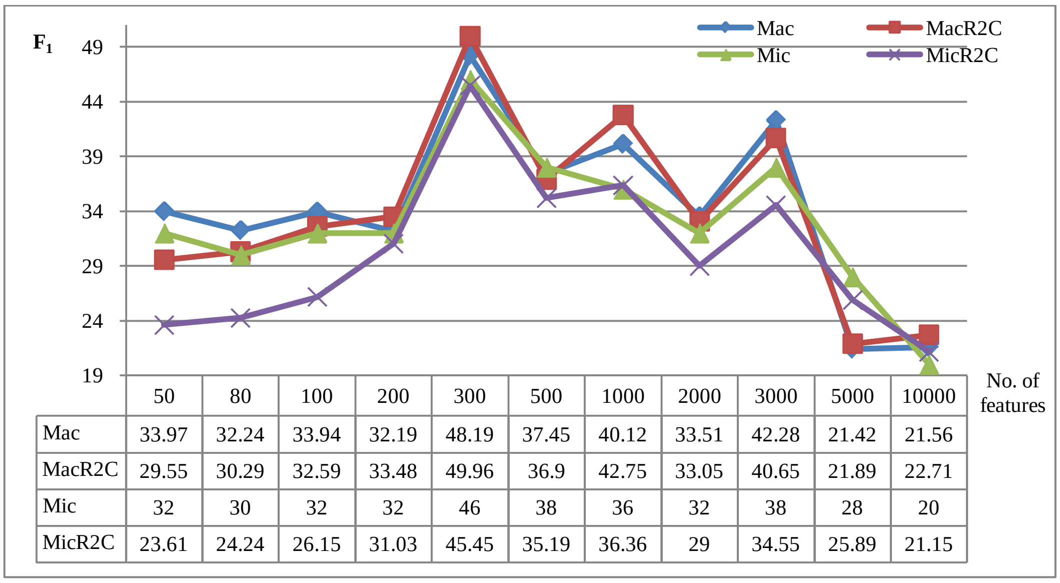
Figure 5-23 Micro/Macro averaged F_1 with/without R^2Cut smoothing strategy of AdaBoost.MHR algorithm with Odds Ratio feature selection algorithm ρ = 14.0



	50	80	100	200	300	500	1000	2000	3000	5000	10000
Mac	14.85	10.51	21.91	26.84	10.11	21.37	36.78	42.4	50.97	64.78	54.9
MacR2C	19.16	14.82	21.52	25.2	9.74	21.51	45.45	50.26	53.59	71.61	66.62
Mic	14	12	18	20	16	24	40	48	48	58	56
MicR2C	14.29	13.25	15.58	19.44	13.16	22.37	38.03	45.83	46.38	55.84	55.84

Figure 5-25 Micro/Macro averaged F_1 with/without R^2Cut smoothing strategy of kNN algorithm with Relevancy Score feature selection algorithm $\rho = 5.6$

Figure 5-23 illuminates that when 300 features are selected by Odds Ratio, micro-averaged F_1 reaches its climax at 26.00% without R^2Cut strategy. For kNN text classifier, employing relevancy score feature selection algorithm and R^2Cut thresholding strategy, macro-averaged F_1 can be as high as 71.61% when 5000 features are selected (Figure 5-25). Using R^2Cut strategy and GSS coefficient to select 300 features, the macro-averaged F_1 reaches its apex at Figure 5-24.



	50	80	100	200	300	500	1000	2000	3000	5000	10000
Mac	33.97	32.24	33.94	32.19	48.19	37.45	40.12	33.51	42.28	21.42	21.56
MacR2C	29.55	30.29	32.59	33.48	49.96	36.9	42.75	33.05	40.65	21.89	22.71
Mic	32	30	32	32	46	38	36	32	38	28	20
MicR2C	23.61	24.24	26.15	31.03	45.45	35.19	36.36	29	34.55	25.89	21.15

Figure 5-24 Micro/Macro averaged F_1 with/without R^2Cut smoothing strategy of Naïve Bayes algorithm with GSS coefficient feature selection algorithm $\rho = 0.08$

Once again, Figure 5-23 and Figure 5-25 demonstrate that R^2Cut thresholding strategy can improve macro-averaged F_1. The averaged improvements of macro-averaged F_1 over the 11 different numbers of features for Adaboost.MHR and kNN are 0.78% and 4.01% respectively as illustrated in Table 5-7. When macro-averaged F_1 reaches it summits, the improvements of R^2Cut strategy for Adaboost.MHR, kNN, and Naïve Bayes are 0.81%, 6.83%, and 1.77% respectively (Figure 5-23 to Figure 5-24). The figures also reveal that if the same value of parameter ρ in R^2Cut algorithm is used to evaluate both macro-averaged and micro-averaged F_1, the micro-averaged F_1 might be degraded. As suggested in the previous sections, further experiments are needed to approach the issue. This is mainly because training and testing a text classifier is very computational expensive. The time spent on obtaining the data as shown in Figure 5-23 to Figure 5-24 for Adaboost.MHR, kNN, and Naïve Bayes are 21 days (30173 minutes), 16 days (23092 minutes) and 6 days (8467 minutes). Time limitation of this project makes it impossible to conduct a large-scale adjustment of ρ to ensure R^2Cut producing best results for both macro-averaged and micro-averaged F_1.

5.5 Summary

As discussed in Chapter 1, human labeled documents are scarce because generate such a document set involves much human labour. In the previous chapter, a series of ODP *categoryDocument* sets were generated automatically. This chapter used the generated *categoryDocument* set as a benchmark document collection evaluated the performance of text categorization algorithms of Adaboost.MHR, Naïve Bayes, kNN when combined with feature selection algorithms such as chi-square, information gain, mutual information, and Odds Ratio to select 50, 80, 100, 200, 300, 500, 1000, 2000, 3000, 5000, and 10000 features. Experimental data demonstrate that the performance of the categorization algorithms in terms of F_1, a widely employed and accepted performance metric, is very satisfactory by using ODP second level *categoryDocument* set as a benchmark data set. The experimental results provided evidence that the **RO3** of this research is achieved successfully, and consequently enriched the existing benchmark document collection such as Reuters-21578,

Table 5-7 The improvement of macro-averaged F_1 of AdaBoost.MHR and kNN with / without R^2Cut smoothing strategy (%) over 11 different features selected (%).

Features	50	80	100	200	300	500	1000	2000	3000	5000	10000	μ
AdaBoost.MHR + Odds ratio	-0.81	3.51	3.51	1.71	0	-0.1	1.41	0.19	1.2	0	-2.01	0.78
kNN + Relevancy score	4.31	4.31	-0.39	-1.64	-0.37	0.14	8.67	7.86	2.62	6.83	11.72	4.01

RCV1, and OHSUMED that were however labeled by human experts (Chapter 2, Section 2.1.1.9).

The experimental data at the same time demonstrated that feature selection algorithms and the number of features are two essential factors that influence the performance of the text categorization algorithms, and the proposed R^2Cut thresholding strategy can marginally improve the macro-averaged F_1 of the above text classifiers.

The chapter also evaluated the performance of SVMLight, Naïve Bayes, kNN, and Statistical Language Model by using a small data set, the S-Set that simulates a set of Web snippets. The experimental results showed that statistical learning algorithms outperform the state of the art SVMs when only a piece of information is available.

Finally, a set of 50 labeled Web snippets from *Google Directory* was collected and used as testing data set to evaluate the performance of Adaboost, kNN, and Naïve Bayes classifier trained by the second level ODP *categoryDocument* set. Experimental results demonstrated that the best result obtained by kNN in terms of macro-averaged F_1 is 71.61% by R^2Cut thresholding strategy and relevancy score algorithm to pick up 5000 features. The best macro-averaged F_1 for Adaboost.MHR and Naïve Bayes are 26.00% and 49.96% respectively, by using the proposed R^2Cut thresholding strategy.

Experimental results in this research by using three different kind of labeled data sets (ODP *categoryDocument*, S-Set, and Web snippets from *Google Directory*) revealed that simple text categorization algorithms such Naïve Bayes and kNN performs consistently at least as better as the state of the art algorithms such as SVMs and Adaboost.

It can be found that the performance of the text categorization algorithms is not very encouraging when used to categorize Web snippets. The next chapter will discuss how to use *boostingUp* algorithm suggested in Chapter 3 to improve the performance of text categorization algorithms to categorize Web snippets, and at the same time to improve the performance of text clustering algorithms.

6 Evaluation of *BoostingUp* Algorithm

This chapter concentrates on evaluating the novel *boostingUp* algorithm. Section 6.1 introduces the implementation details of K-Means and Hierarchical Agglomerative Clustering (HAC) algorithm. The section also describes how to calculate Rand Index and Adjusted Rand Index, two evaluation metrics of clustering. The implementation of *boostingUp* algorithm is discussed in detail in Section 6.2. This is followed by Section 6.3 that presents the evaluation results of the two clustering algorithms based on three labeled sets of Web snippets as the search results of Google Directory and ODP. Section 6.4 is to provide the experimental results of *boostingUp* and discuss related issues; an example is provided to illustrate how *boostingUp* improve both categorization and clustering. Finally, Section 6.5 summarizes this chapter.

6.1 Implementation of Text Clustering Algorithms: K-Mean, HAC, and LSA

In this research, two text clustering algorithms, K-Means and Hierarchical Agglomerative Clustering (HAC) are implemented. K-Means is selected because it is one of the most widely and popular partitional clustering algorithms with acceptable performance and simple to implement. HAC is implemented for the reason that it is a representative of hierarchical clustering algorithms that arrange the document collection hierarchically. For K-Means, two concomitant centroids initialization algorithms, MaxMin and Anomalous Pattern are implemented to generate the number of clusters K, with a list of initial centroids for the K clusters. For HAC, single-link and complete link (Chapter 2, Section 2.2.3 Table 2-4) are implemented as cluster merging approach. In addition to taking cosine value as a similarity measure to produce a proximity matrix, Latent Semantic Analysis (LSA) is also employed to calculate a proximity matrix for HAC. LSA is chosen because it is one of the most important and widely used document similarity measure algorithm in the field of semantic analysis, and it is reported superior to other similarity measure approaches such as cosine similarity (Deerwester et al. 1990), although it is computational expensive.

For each of the clustering algorithms, the following five factors need to be taken into account:

1) How to represent a document;
2) How to reduce the high dimensionality if documents are represented by vectors;
3) How to measure the performance of a text clustering algorithm;
4) How to select the number K of clusters to be generated and the initial centroids;

5) How to measure the similarities between documents.

For factors 1) and 2), as in the case of text categorization, documents are represented in a high dimensional vector space, and the high dimensionality of the vector space will be reduced by feature selection algorithms as discussed in Chapter 2, Section 2.3. For the factor 3), Rand Index, Adjusted Rand Index (ARI), and F_β are to be used to measure the performance of the algorithms. Considering factor 5), since cosine value has been proven an effective similarity measure for comparing textual documents (Baeza-Yates and Ribeiro-Neto 1999; Manning, Raghavan, and Schütze 2008), it is also applied for K-Means and HAC. Further, for LSA, the inner product of the approximated matrix is the measure of similarities between documents (Appendix 5), thus the factor 5) is addressed. With regarding to the factor 4), how to generate the cluster number K is to be discussed in the following sections with the introduction of the implementation of the different clustering algorithms.

6.1.1 K-Means

The implemented K-Means algorithm in RIB takes as input the following parameters: 1) a HashMap <String, String> structure that passes in the document set in the format of <document ID, content> tuple; 2) the number of cluster K; 3) the type of similarity measure, such as cosine value, Euclidean distance, or others; 4) the centroid initialization approach: MaxMin or Anomalous Pattern; 5) whether a similarity-based ranked list of centroids for each of the documents is provided for the purpose of *boostingUp* algorithm.

The K-Means algorithm is illustrated in Appendix 5, Appendix Figure 5-8. In that figure, the first step is to assign the cluster number K and a list of initial tentative centroids. The number of clusters K can be decided by 1) Max-Min or Anomalous Pattern algorithm as discussed in Appendix 5. Note that the two algorithms can also produce an initial list of centroids; 2) assigning directly an arbitrary number and a list of randomly selected centroids, and then try other numbers until satisfactory results are obtained; and 3) based on the results of categorization where the number of categories involved are of interest. RIB implements the three approaches that decide the number of clusters to be generated.

A final cluster-merging step is carried out to eliminate clusters that contain only one document to avoid a strange case that for tens of generated clusters, only one of them contains more than one elements while all the others include only one document. Let $C = \{C_1, C_2, \ldots C_K\}$ be the generated K clusters , and $\mathbf{d} \in C_j$ is the only document in the generated cluster C_j with centroid $\mathbf{c}_j$ ($\mathbf{c}_j$ is the same as $\mathbf{d}$ under such circumstances), calculate the similarities $sim(\mathbf{d}, \mathbf{c}_i)$ between document $\mathbf{d}$ and the centroid $\mathbf{c}_i$ of cluster C_i, $i = 1, 2, \ldots K$ and $j \neq i$. Document $\mathbf{d}$ is to be merged to the cluster C_p if $sim(\mathbf{d}, \mathbf{c}_p) = \max \{sim(\mathbf{d}, \mathbf{c}_i) | i = 1, 2, \ldots K, j \neq i \}$.

6.1.2 Hierarchical Agglomerative Clustering (HAC)

The implemented HAC algorithm includes four classes: *HAC*, *PMProcess*, *hacTree* and *hacTreeNode*. *HAC* constructs an instance of the class and returns a given number of generated clusters according to parameters passed in. *hacTree* and *hacTreeNode* classes are designed to maintain the generated the clusters in a tree-like structure. Class *PMProcess* is instantiated with four parameters: 1) the type of measurement of similarities: cosine similarity, Euclidean distance (Chapter 2, Table 2-1), or others; 2) an initial proximity matrix (or similarity matrix (Manning, Raghavan, and Schütze 2008)) for which element (i, j) is the similarity score of document i and document j, which needs to be calculated before HAC is instantiated; 3) the type of similarity, single-link or complete-link; and 4) the dimensionality of the proximity matrix.

The algorithm is demonstrated in Figure 6-1. It first initializes a proximity matrix, and then repeatedly merges clusters until there are only two clusters left. Inside each loop, the algorithm estimates either single-link similarity or complete-link similarity (Chapter 2, Table 2-4) of the generated clusters, updates proximity matrix based on the estimated similarities, and then merges the clusters stored in the *htree* which is an instance of *hacTree* structure.

To return a given number of K clusters, the algorithm compares the clusters in *htree* with K, at the very beginning, *htree* contains only two clusters. If the size of the *htree* equals K, it returns clusters in *htree*; if K bigger than the size of *htree*, the algorithm splits a non-leaf

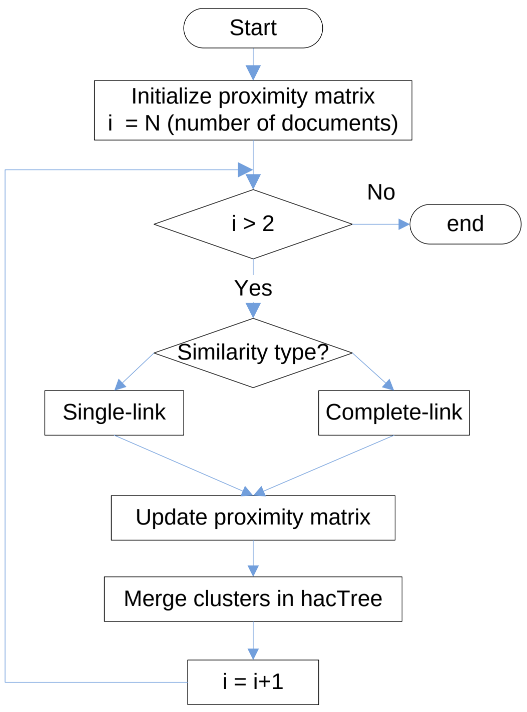


Figure 6-1 Flowchart of HAC algorithm

cluster to produce a new *htree*, and then compares if the size of *htree* equals K, if it is, it returns *htree*, otherwise, repeats the process until the size of *htree* is the same as K.

6.1.3 Latent Semantic Analysis (LSA)

In this project, JAMA (http://math.nist.gov/javanumerics/jama/), a basic linear algebra library for java is employed to perform singular value decomposition which is the core of LSA. A class *LSA* is designed to re-represent documents and queries in a reduced feature space. Methods in the class *LSA* include *SingularValueDecomposition()*, *getReducedSVD()*, *getDocumentProximityMatrix()*, *getTermProximityMatrix()*, and *getSVDQuery()*. For text categorization, *LSA* returns a documents proximity matrix that is then taken as input by HAC algorithm.

Dumais (2004) indicates that the number *n* of dimensions used in the reduced space dramatically affects the performance of LSA because it actually specifies the dimension of concept space in which terms and documents are represented. It is found that the performance of LSA is satisfactory when n ranges from 50 to 200, and the performance reaches its apex when dimension is assigned to 90. In this research, for efficiency purpose, the dimension in the reduced space is 50.

6.1.4 Evaluation Criteria – Adjusted Rand Index and F_β

There are a number of different criteria to evaluate the performance of text clustering algorithms, these include the purity and Rand Index introduced in Chapter 2. In this research, Rand Index is first introduced because it measures the agreement between a set of generated clusters and a set of external clusters which are taken as external criteria (Santos and Embrechts 2009).

Using the notion similar to that of Santos and Embrechts (2009), let S = {O_1, O_2, …, O_N} is a set of N objects; X = {x_1, x_2, … x_A}, Y = {y_1, y_2, …, y_B} are two different set of clusters of object in S, and $\cup_{i=1}^{A} x_i = S = \cup_{j=1}^{B} y_j$ and $x_i \cap x_{i'} = \phi = y_j \cap y_{j'}$ for $1 \le i \ne i' \le A$, and $1 \le j \ne j' \le B$. Supposed that n_{ij} be the number of objects in cluster $x_i \cap y_j$, and $n_{i.}$ and $n_{.j}$ be the number of objects in cluster x_i and y_j respectively, a table (Table 3-2) can be generated to illustrate the cluster overlap between X and Y.

To calculate the Rand Index, there are four possible decisions for each of the documents pairs need to be considered. The four decisions are similar to that defined in a contingency table for evaluating text categorization algorithms. Here the cluster X is taken as the external criteria or the true clusters to be referred to.

True Positive (TP): two similar documents are assigned in a same cluster;

True Negative (TN): two different documents are assigned to different clusters;

False Positive (FP): two dissimilar documents are assigned into a same cluster;

False Negative (FN): two similar documents are assigned to different clusters.

The four parameters can be calculated by using the vales in Chapter 3 Table 3-1.

$$TP = \sum_{i=1}^{A}\sum_{j=1}^{B}\binom{n_{ij}}{2} = (\sum_{i=1}^{A}\sum_{j=1}^{B} n_{ij}^2 - n)/2$$

$$FN = \sum_{i=1}^{A}\binom{n_{i.}}{2} - TP = \left(\sum_{i=1}^{A} n_{i.}^2 - \sum_{i=1}^{A}\sum_{j=1}^{B} n_{ij}^2\right)/2$$

$$FP = \sum_{j=1}^{B}\binom{n_{.j}}{2} - TP = \left(\sum_{j=1}^{B} n_{.j}^2 - \sum_{i=1}^{A}\sum_{j=1}^{B} n_{ij}^2\right)/2$$

$$TN = \binom{N}{2} - TP - FN - FP$$

$$= \binom{N}{2} - \sum_{i=1}^{A}\binom{n_{i.}}{2} - \sum_{j=1}^{B}\binom{n_{.j}}{2} + TP$$

$$= \left(\sum_{i=1}^{A}\sum_{j=1}^{B} n_{ij}^2 + n^2 - \sum_{i=1}^{A}\binom{n_{i.}}{2} - \sum_{j=1}^{B}\binom{n_{.j}}{2}\right)$$

The Rand Index is calculated

$$RandIndex(RI) = \frac{TP + TN}{TP + FP + FN + TN}$$

The measure F_β is estimated by

$$F_\beta = \frac{(\beta^2 + 1) \times P \times R}{\beta^2 \times P + R}$$

Table 6-1 Notion of the contingency table for comparing two clusters

Clusters	**y_1**	**y_2**	**…**	**y_B**	**Sums**
x_1	n_{11}	n_{12}	…	n_{1B}	$n_{1.}$
x_2	n_{21}	n_{22}	…	n_{2B}	$n_{2.}$
⋮	⋮	⋮		⋮	⋮
x_A	n_{A1}	n_{A2}	…	n_{AB}	$n_{A.}$
Sums	$n_{.1}$	$n_{.2}$	…	$n_{.B}$	$n_{..} = n$

Where P and R are precision and recall which are similar to the two measurements in information retrieval. Since most of research in the field of text categorization and clustering employs F_1 as a measure, it is also used in this research.

$$P = \frac{TP}{TP + FP}$$

$$R = \frac{TP}{TP + FN}$$

However, as indicated by Milligan and Cooper (1986), and Santos and Embrechts (2009), Rand Index suffers from known problems such as for two random partitions, the value of RI is not a constant (for example, say should be zero), or the values of RI approaches its upper boundary of unity with the number of clusters increasing. With the intention to address the issues, some other measures are proposed, and as pointed out by Milligan and Cooper (1986) and Santos and Embrechts (2009), Adjusted Rand Index (ARD) is a successful one which is recommended by the researchers. Therefore, in this research, ARI with F_1 are accepted as the measurements to evaluate the performance of text clustering algorithms.

Adjusted Rand Index is calculated by

$$ARI = \frac{\binom{n}{2} \times (TP + TN) - [(TP + FP) \times (TP + FN) + (TN + FN) \times (TN + FP)]}{\binom{n}{2}^2 - [(TP + FP) \times (TP + FN) + (TN + FN) \times (TN + FP)]}$$

6.2 Implementation of *BoostingUp*

This section introduces the implementation of *boostingUp* in Java programming language. The implemented algorithm is introduced and discussed in Chapter 3, Figure 3-3. A class *boostingUp* is designed to fulfill the functions of *boostingUp* algorithm.

Class *boostingUp* takes two input parameters *catClusters* and *clsClusters* that have the type of HashMap <String, ArrayList<String>>. The hashmap *catClusters* presents the categorization results and *clsClusters* is used to store the results of clustering.

The identifiers of documents in the categorization results are appended with a suffix that is a list of ranked categories returned by a text classifier, such as Naïve Bayes; similarly, identifiers of documents in the results of clustering are also expanded by a list of labels of clusters generated by a clustering algorithm such as K-Means.

Class *boostingUp* includes the following methods: *getCategories(), getContingentTable(), adjustCategoryResutls(),* and *adjustClusteringResults().* Method *getCategories()* extracts categories from *catClusters* and arranges the categories in a sorted ArrayList structure for quick retrieving purpose. The method also extracts and stores labels of clusters. Method *getContingentTable()* produces a two dimensional array with a variable name *contingentTable* where the row of the table corresponds to the categories of a given classifier and the column corresponds to one of the generated clusters. Each element of the table represents the number of documents that belong to a given category and a given cluster. Method *adjustCategoryResults()* and *adjustClusteringResults()* re-arrange the elements in the structures of *catClusters* and *clsClusters* according to *boostingUp* algorithm discussed in Chapter 3. The flow chart of *boostingUp* algorithm is demonstrated in Figure 6-2. In this figure, the variables *cellsWithValueOne* and *cellsWithValueBiggerThanOne* with the type of ArrayList <Integer> are used to keep a list of cells in the contingent table with value equals to one, and a list of cells for which their values are bigger than one. For a cell in the *row*th row and *col*th column, the value of the cell is decided by (with the assumption that the number of categories and clusters are less than 1000)

$$key = (row + 1)\times 1000 + (col + 1)$$

In the method of *adjustCategoryResutls()* and *adjustClusteringResults()*, the row and column of an element involved is obtained by a reversing process,

$$row = key / 1000 - 1$$

$$col = key \% (1000) - 1$$

where "%" is a java operator represents mod operation and "/" is integer division.

Note that function *adjustClusteringResults()* is not illustrated in Figure 6-2 for the purpose of conciseness. It is very similar to *adjustCategoryResutls()* except that *row* and *col* are exchanged when *max* and *mc* are obtained and element in *clsClusters* is moved.

6.3 Evaluation of Clustering Algorithms

This section first presents the evaluation results in terms of ARI and F_1 of K-Means and HAC text clustering algorithms when the ODP *categoryDocuments* are used as data set. It then provides the evaluation results for the two algorithms by using a collection of Web search results from Google when "jaguar" is the search-term. The evaluation results are taken as criteria to choose a clustering algorithm to be employed by the proposed *boostingUp* approach.

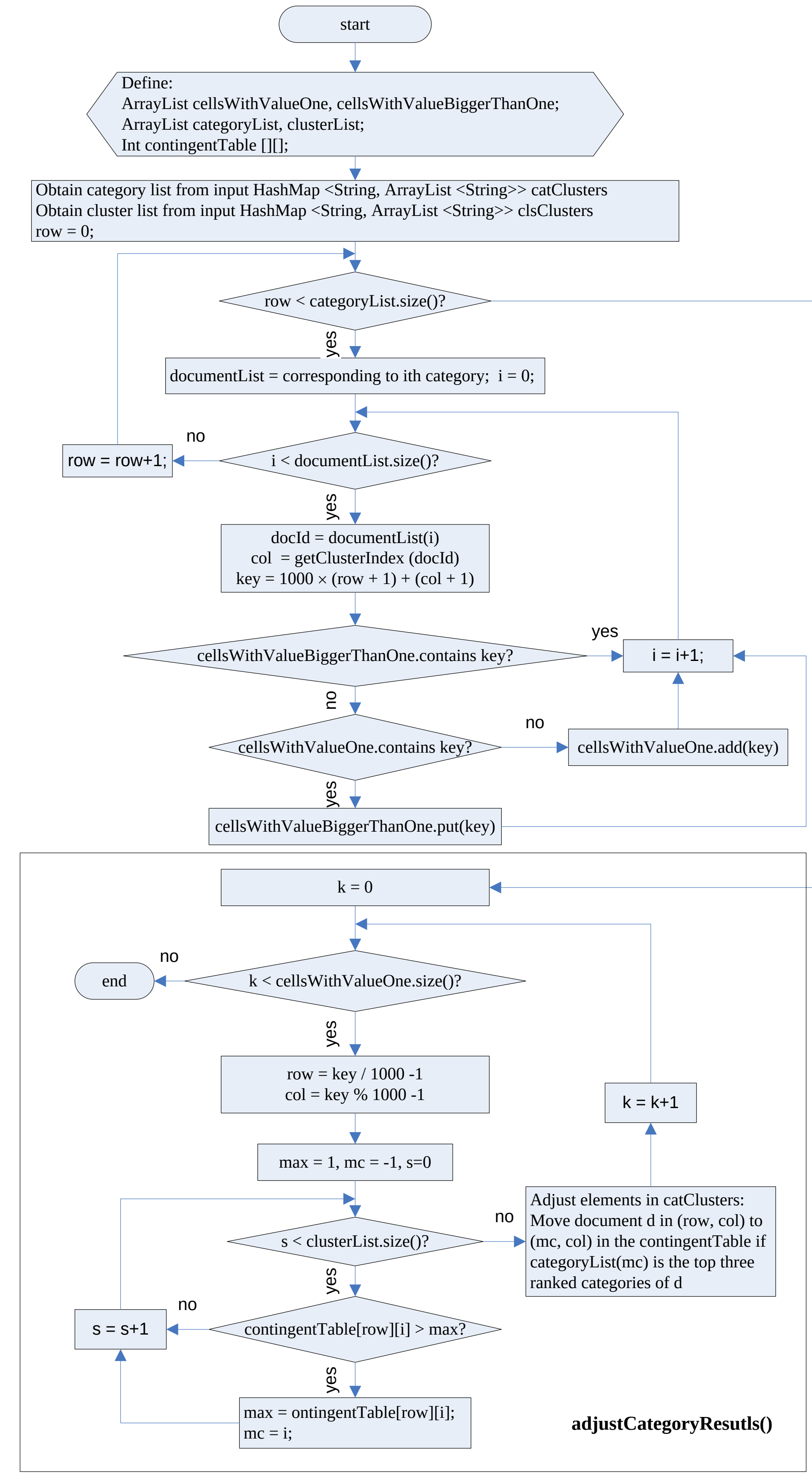

start
Define:
ArrayList cellsWithValueOne, cellsWithValueBiggerThanOne;
ArrayList categoryList, clusterList;
Int contingentTable [][];
Obtain category list from input HashMap <String, ArrayList <String>> catClusters
Obtain cluster list from input HashMap <String, ArrayList <String>> clsClusters
row = 0;
row < categoryList.size()?
yes
documentList = corresponding to ith category; i = 0;
no
row = row+1;
i < documentList.size()?
yes
docId = documentList(i)
col = getClusterIndex (docId)
key = 1000 × (row + 1) + (col + 1)
no
yes
cellsWithValueBiggerThanOne.contains key?
i = i+1;
no
no
cellsWithValueOne.contains key?
cellsWithValueOne.add(key)
yes
cellsWithValueBiggerThanOne.put(key)
k = 0
no
end
k < cellsWithValueOne.size()?
yes
row = key / 1000 -1
col = key % 1000 -1
k = k+1
max = 1, mc = -1, s=0
no
Adjust elements in catClusters:
Move document d in (row, col) to (mc, col) in the contingentTable if categoryList(mc) is the top three ranked categories of d
s < clusterList.size()?
yes
no
s = s+1
contingentTable[row][i] > max?
yes
max = ontingentTable[row][i];
mc = i;
adjustCategoryResutls()

Figure 6-2 Flowchart of *boostingUp* algorithm

6.3.1 Using ODP *CategoryDocuments* to Evaluate Clustering Algorithms

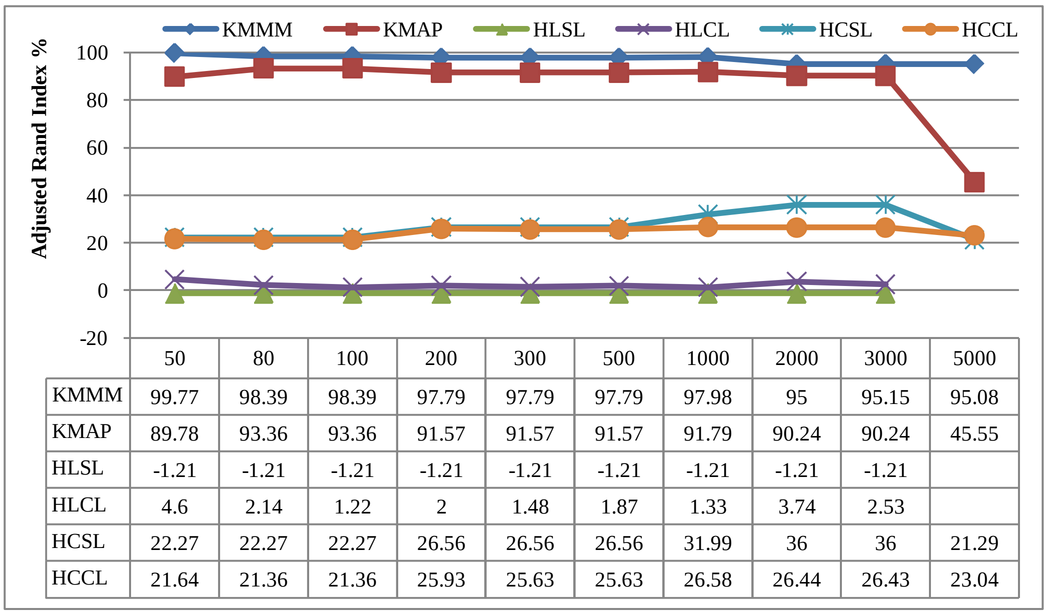
This section provides evaluation results for the text clustering algorithms when utilizing the generated ODP *categoryDocuments* as data sets. The clustering algorithms to be evaluated include:

1) K-Means + MaxMin centroids initialization algorithm (KMMM)
2) K-Means + Anomalous Pattern centroids initialization algorithm (KMAP)
3) HAC + LSA + Single-link (HLSL)
4) HAC + LSA + Complete-link (HLCL)
5) HAC + Cosine similarity + Single-link (HCSL)
6) HAC + Cosine similarity + Complete-link (HCCL)

Figure 6-3 and Figure 6-4 illustrate ARI and F_1 scores for the above six clustering algorithms when employing the second level ODP *categoryDocuments* as data set. Similar to the settings of experiments in Chapter 5, a list of 50, 80, 100, 200, 300, 500, 1000, 2000, 3000, and 5000 features are selected by Information Gain and Odds Ratio feature selection algorithms. Since the evaluation results of Information Gain are very similar to that of Odds Ratio, the results are not presented here. Since LSA is computationally intensive, when feature number is bigger than 3000, the results of HAC+LSA are also not given.

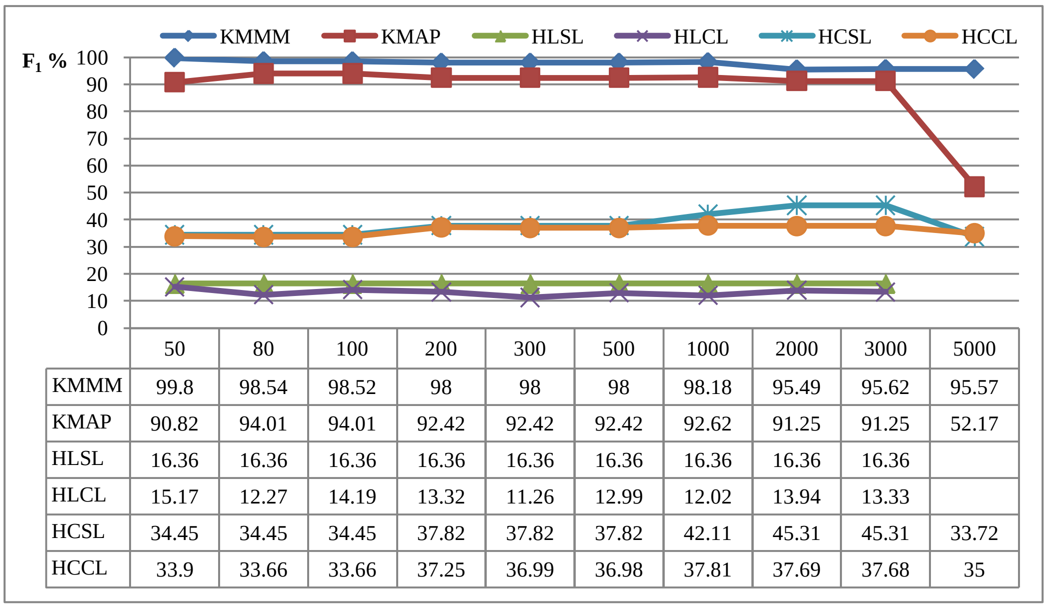
As can observed from the two figures,

1) In terms of ARI and F_1, using second level ODP *categoryDocument* set and Odds Ratio feature selection algorithm to select features from 50 to 5000, K-Means with MaxMin



	50	80	100	200	300	500	1000	2000	3000	5000
KMMM	99.77	98.39	98.39	97.79	97.79	97.79	97.98	95	95.15	95.08
KMAP	89.78	93.36	93.36	91.57	91.57	91.57	91.79	90.24	90.24	45.55
HLSL	-1.21	-1.21	-1.21	-1.21	-1.21	-1.21	-1.21	-1.21	-1.21	
HLCL	4.6	2.14	1.22	2	1.48	1.87	1.33	3.74	2.53	
HCSL	22.27	22.27	22.27	26.56	26.56	26.56	31.99	36	36	21.29
HCCL	21.64	21.36	21.36	25.93	25.63	25.63	26.58	26.44	26.43	23.04

Figure 6-3 ARI of clustering algorithms evaluated by 2nd level ODP *categoryDocument* set



	50	80	100	200	300	500	1000	2000	3000	5000
KMMM	99.8	98.54	98.52	98	98	98	98.18	95.49	95.62	95.57
KMAP	90.82	94.01	94.01	92.42	92.42	92.42	92.62	91.25	91.25	52.17
HLSL	16.36	16.36	16.36	16.36	16.36	16.36	16.36	16.36	16.36	
HLCL	15.17	12.27	14.19	13.32	11.26	12.99	12.02	13.94	13.33	
HCSL	34.45	34.45	34.45	37.82	37.82	37.82	42.11	45.31	45.31	33.72
HCCL	33.9	33.66	33.66	37.25	36.99	36.98	37.81	37.69	37.68	35

Figure 6-4 F_1 scores of clustering algorithms evaluated by 2nd level *categoryDocument* set

centroid initialization algorithm outperforms all the other five text clustering algorithms evaluated in this experiment.

2) K-Means with Anomalous Pattern centroid initialization algorithm is only (but always) slightly inferior to K-Means with MaxMin. It however, obviously produces much better results than HAC algorithms.
3) In addition to the well-known drawback of LSA for its computational cost, in this experiment, the combination of HAC and LSA is shown not to be effective as well.
4) HAC with cosine similarity measurement, although performing better than HAC with LSA, is not comparable with K-Means.
5) Experimental data of HAC with cosine similarity demonstrate that single-link approach is marginally superior to complete-link approach; no matter the measurement is ARI or F_1.
6) Increasing the number of features provides no evidence on improving the performance of the text clustering algorithms significantly, especially when the feature number is bigger than 3000.

6.3.2 Using Labeled Web Search Results to Evaluate Clustering Algorithms

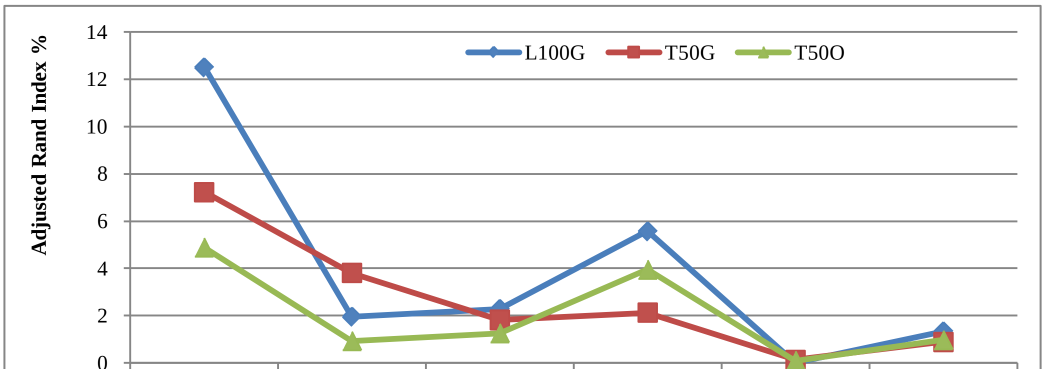
Labeled search results can be obtained by submitting queries to ODP or Google Directory Web search services, as discussed in Chapter 5, Section 5.4. In this section, three data sets are to be generated and used as a basis to evaluate the performance of the six clustering algorithms listed in Section 6.3.1. The purpose of the evaluation is to check which algorithm is more suitable for Web snippets clustering based on the three data sets.

The first set is similar to the data set generated in Chapter 5, Section 5.4, except that the new data set contains the last 100 but not 50 search results from Google Directory. The reason for selecting 100 Web snippets is that if there are not enough Web snippets that are less informative than full-length documents, there is hardly any meaningful clusters exist among the snippet collection. The second data set includes top 50 Web snippets of Google Directory when “jaguar” is the search term. A third labeled Web snippet collection is composed of top 50 results from ODP by using “jaguar” as a search-term as well. The first 10 hits of Google Directory and ODP for search-term “jaguar” are presented in Appendix 2 for reference.

Figure 6-5 and Figure 6-6 illustrate the Adjusted Rand Index and F_1 values of the different clustering algorithms evaluated based on the three sets of Web snippets. In these figures, L100G is the abbreviation of Last 100 Web snippets from Google Directory. Similarly, T50G refers to Top 50 Web snippets from Google Directory, and T50O represents the set of Top 50 Web snippets of ODP. All the Web snippets are the results of using the search-term “jaguar”. Acronyms for the algorithms were provided in the previous section.

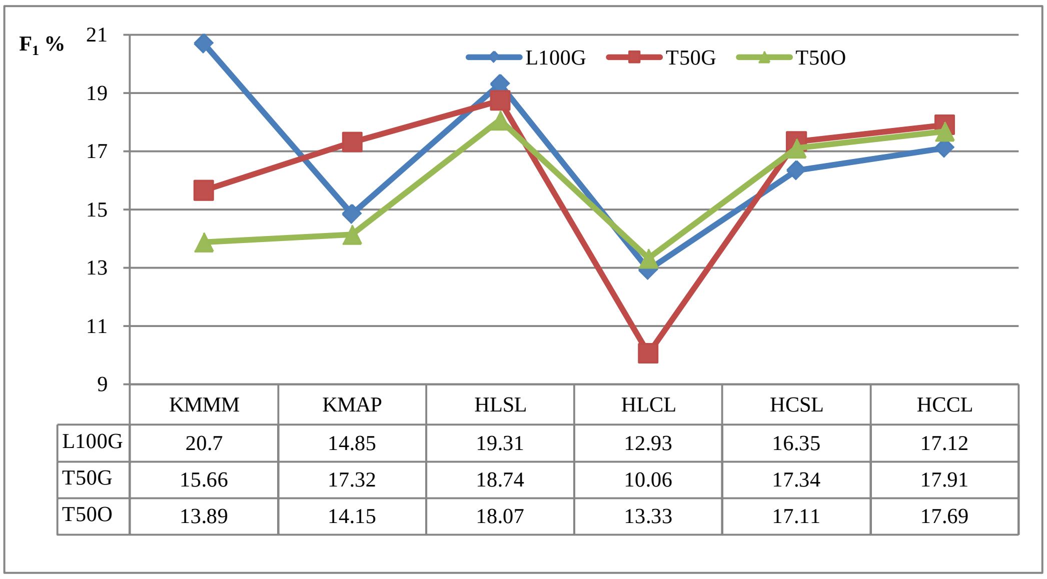
The figures demonstrate that

1) K-Means with MaxMin centroid initialization algorithm outperforms the other evaluated algorithms in terms of Adjusted Rand Index and F_1, no matter which Web snippet data set is employed. This results support the observation in the previous section when ODP *categoryDocuments* are utilized as the data sets.
2) HAC algorithm with cosine similarity and single-link approach performs worst in terms of ARI. When evaluated by F_1 score, HAC with LSA and complete-link algorithm performs badly. However, if measured by ARI, it is the second best algorithm and is only inferior to KMMM.



	KMMM	KMAP	HLSL	HLCL	HCSL	HCCL
L100G	12.51	1.96	2.29	5.57	0	1.34
T50G	7.23	3.81	1.82	2.13	0.13	0.89
T50O	4.89	0.93	1.26	3.95	0.1	0.97

Figure 6-5 ARI of clustering algorithms evaluated by different sets of Web snippet



	KMMM	KMAP	HLSL	HLCL	HCSL	HCCL
L100G	20.7	14.85	19.31	12.93	16.35	17.12
T50G	15.66	17.32	18.74	10.06	17.34	17.91
T50O	13.89	14.15	18.07	13.33	17.11	17.69

Figure 6-6 F_1 scores of clustering algorithms evaluated by different sets of Web snippet

3) Compared with the results presented in previously section when ODP *categoryDocuments* are employed as data set, it can be found that clustering Web snippets is a challenging task. When Web snippets are clustered, the highest ARI and F_1 are only 12.51% and 20.7% respectively, whereas these scores can be as high as 99.77 and 99.8 when using second level ODP *categoryDocuments* set.

Based on the above experimental results and observations, K-Means with MaxMin centroids initialization algorithm is chosen as the clustering algorithm to be applied in the proposed *boostingUp* method to be evaluated in the following section.

6.4 Evaluation of *BoostingUp*

This section presents the evaluation results for the proposed *boostingUp* algorithm in terms of ARI and F_1, and to estimate to what degree *boostingUp* can improve the performance of text categorization. Meanwhile, attention also needs to be paid to examine if *boostingUp* can improve performance of categorization and clustering at the same time.

6.4.1 Experiments Settings

The settings of the experiments are as follows.

1) Categorization and clustering algorithms

According to the experimental results presented in the previous section, clustering algorithm employed by *boostingUp* is K-Means with MaxMin centroid initiation. Clustered results are also merged to eliminate clusters that contain only one document. Text classifier employed in *boostingUp* algorithm is Naïve Bayes.

2) Feature selection algorithms and number of features selected

50, 80, 100, 200, 300, 500, 1000, 2000, 3000, 5000, and 10000 features are selected by using Information Gain, Odds Ratio and Relevancy Score feature selection algorithms for Naïve Bayes text classifier.

3) Evaluation measurements

For categorization results, macro-averaged and micro-averaged F_1 results of NB and *boostingUp* are presented. For clustered results, ARI and F_1 scores of K-Means and *boostingUp* are provided.

4) Data set employed to train a classifier

The fourth level ODP *categoryDocument* set is selected as the training data set because the average length of documents in this set is closer to the average length of Web snippets, compared with the average length of documents in the second and third level *categoryDocument* sets.

5) Test data set

The test data set for categorization and clustering is L100G, the last 100 Web snippets of Google Directory when “jaguar” is the search-term as described in Chapter 5, Section 5.4 and previous section, Section 6.3.

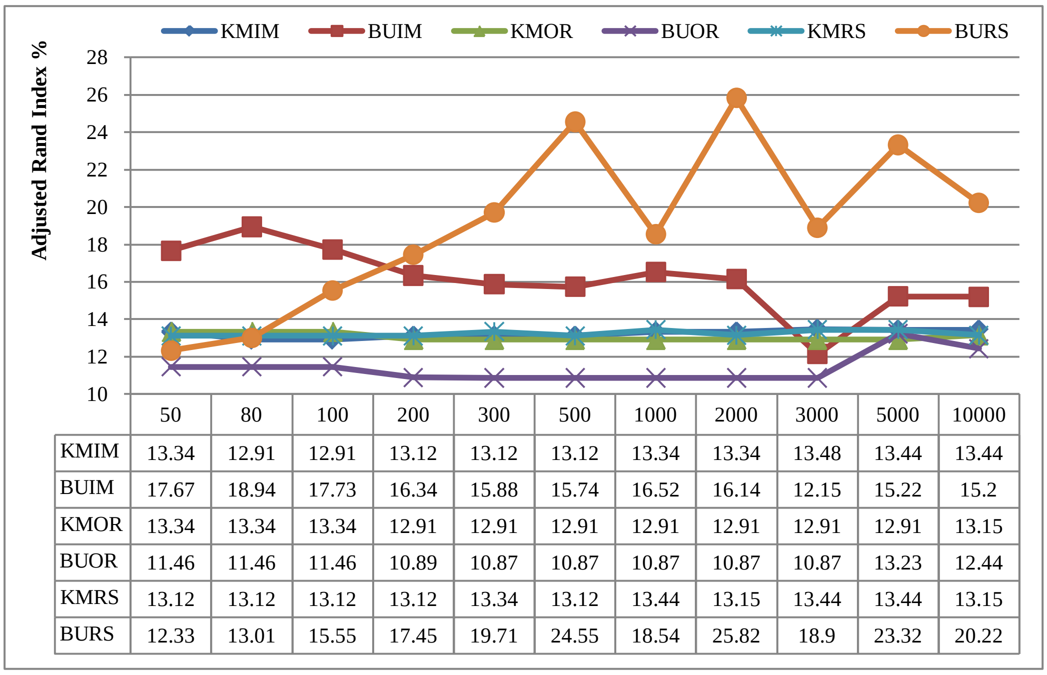
6.4.2 Experimental Results of *BoostingUp*

Experimental results are presented in Figure 6-7 to Figure 6-10. The abbreviations used in the figures are KM – K-Means clustering algorithm, BU – *boostingUp* algorithm, NB – Naïve Bayes classifier, IM – Information Gain, OR – Odds Ratio, RS – Relevancy Score. For example, BUOR refers to *boostingUp* with Odds Ratio feature selection algorithm.

Figure 6-7 illustrates the ARI scores of K-Means and *boostingUp* when 50, 80, 100, 200, 300, 500, 1000, 2000, 3000, 5000, and 10000 features are selected by Information Gain, Odds Ratio, and Relevancy Score feature selection algorithms.

Figure 6-8 presents the F_1 scores of K-Means and *boostingUp*.

Figure 6-9 and Figure 6-10 demonstrate the macro-averaged and micro-averaged F_1 scores of Naïve Bayes classifier and *boostingUp* algorithm respectively. Different numbers of features are picked up by Information Gain, Odds Ratio and Relevancy Score feature selection algorithms. Table 6-2 summarizes the improvements of *boostingUp* over K-Means and Naïve Bayes.



	50	80	100	200	300	500	1000	2000	3000	5000	10000
KMIM	13.34	12.91	12.91	13.12	13.12	13.12	13.34	13.34	13.48	13.44	13.44
BUIM	17.67	18.94	17.73	16.34	15.88	15.74	16.52	16.14	12.15	15.22	15.2
KMOR	13.34	13.34	13.34	12.91	12.91	12.91	12.91	12.91	12.91	12.91	13.15
BUOR	11.46	11.46	11.46	10.89	10.87	10.87	10.87	10.87	10.87	13.23	12.44
KMRS	13.12	13.12	13.12	13.12	13.34	13.12	13.44	13.15	13.44	13.44	13.15
BURS	12.33	13.01	15.55	17.45	19.71	24.55	18.54	25.82	18.9	23.32	20.22

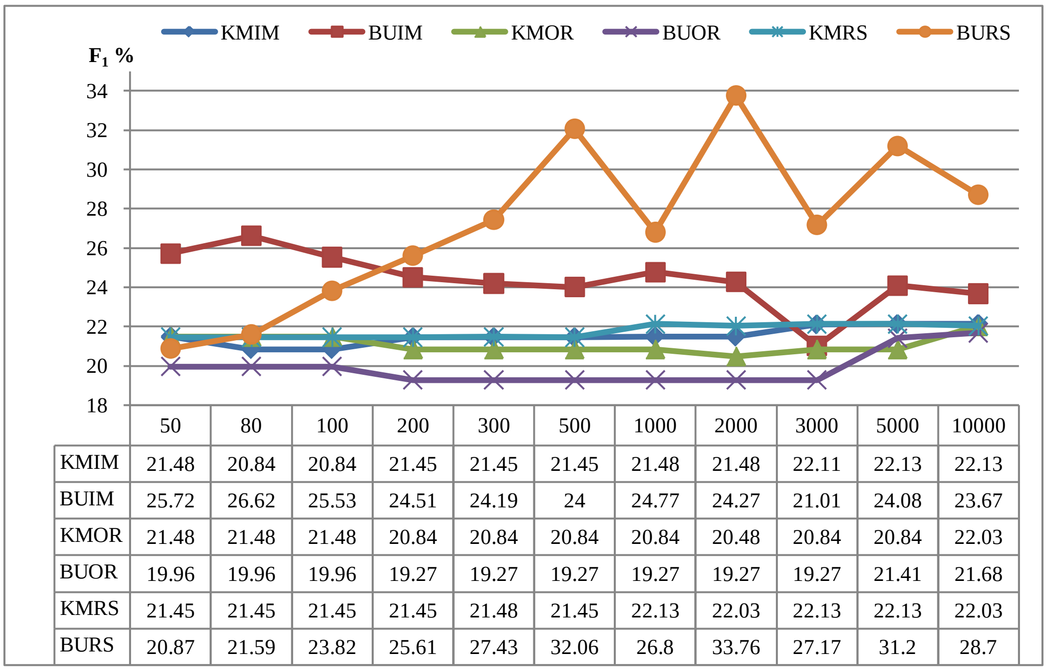
Figure 6-7 ARI of K-Means and *boostingUp* evaluated by T100G

KM – K-Means clustering algorithm, BU – *boostingUp* algorithm, NB – Naïve Bayes classifier, IM – Information Gain, OR – Odds Ratio, RS – Relevancy Score. For example, BUOR refers to *boostingUp* with Odds Ratio feature selection algorithm

Note that the ARI and F_1 scores presented in Figure 6-5 and Figure 6-6 of K-Means with L100G in Section 6.3 differ from that provided in Figure 6-7 and Figure 6-8. This is because the initial number K of clusters for K-Means in Section 6.3 is decided by MaxMin algorithm, whereas in Section 6.4, the initial K is decided by categorization algorithm, that is, the number of categories under which test documents are arranged. As different number of features are selected by different feature selection algorithms, different number of categories will be generated by Naïve Bayes classifier, and the number will consequently influence the evaluation results of K-Means which takes in the number as input.

Figure 6-7 shows that when 2000 features are selected by Relevancy Score, *boostingUp* reaches the apex of ARI 25.82%, and it always performs better than K-Means only except when 50 features are selected. The figure also demonstrates that *boostingUp* can always improve ARI when Information Gain is employed to choose features, with the only exception when 3000 features are picked up. However, if features are selected by Odds Ratio, the ARI of *boostingUp* is always lower than that of K-Means.

Figure 6-8 provides similar results with Figure 6-7 when the two algorithms are measured by F_1 scores. The best F_1 score is reached by *boostingUp* when 2000 features are chosen by Relevancy Score. Using Relevancy Score feature selection algorithm, *boostingUp* always outperforms K-Means except 50 features are chosen. Information Gain feature selection

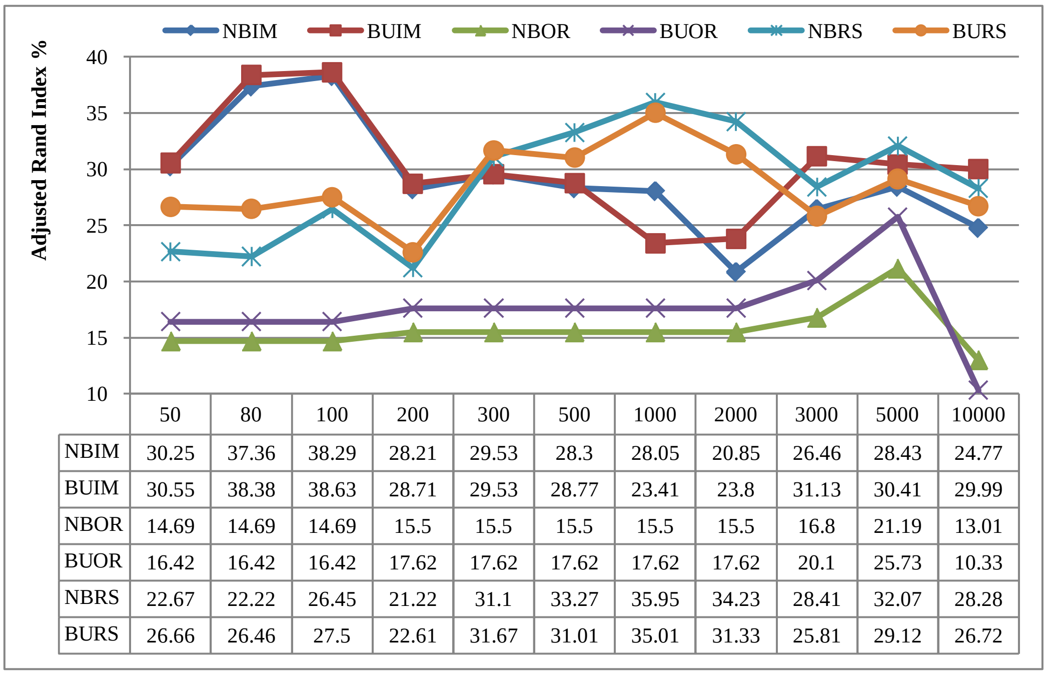


	50	80	100	200	300	500	1000	2000	3000	5000	10000
KMIM	21.48	20.84	20.84	21.45	21.45	21.45	21.48	21.48	22.11	22.13	22.13
BUIM	25.72	26.62	25.53	24.51	24.19	24	24.77	24.27	21.01	24.08	23.67
KMOR	21.48	21.48	21.48	20.84	20.84	20.84	20.84	20.48	20.84	20.84	22.03
BUOR	19.96	19.96	19.96	19.27	19.27	19.27	19.27	19.27	19.27	21.41	21.68
KMRS	21.45	21.45	21.45	21.45	21.48	21.45	22.13	22.03	22.13	22.13	22.03
BURS	20.87	21.59	23.82	25.61	27.43	32.06	26.8	33.76	27.17	31.2	28.7

Figure 6-8 F_1 scores of K-Means and *boostingUp* evaluated by L100G

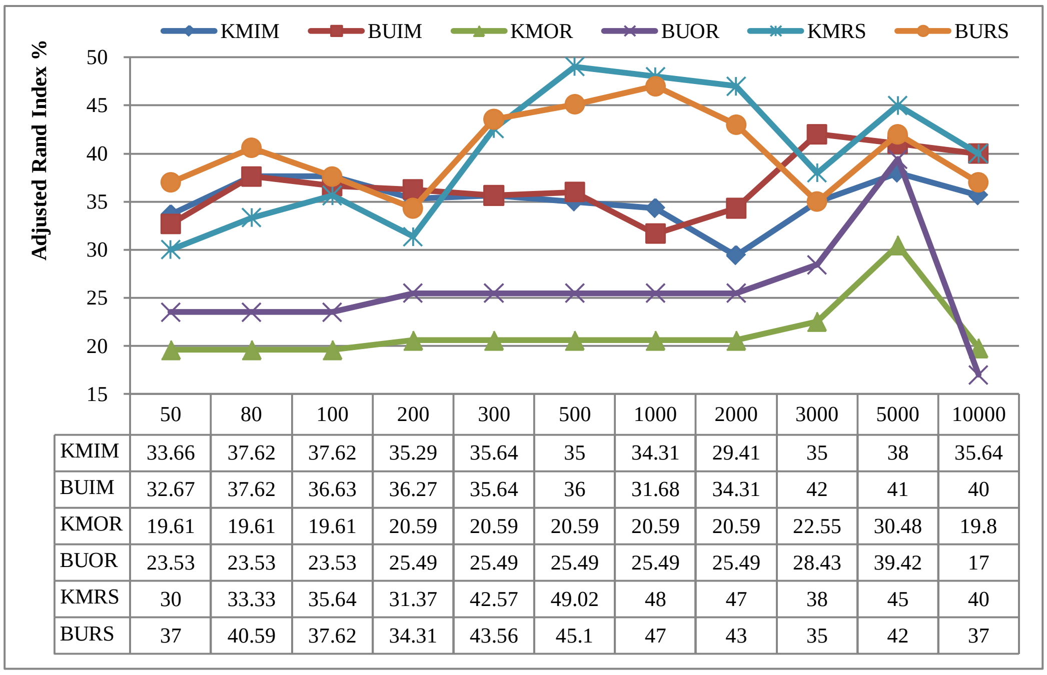
algorithm make *boostingUp* almost always produce better results than K-Means. Nevertheless, *boostingUp* is slightly inferior to K-Means if features are selected by Odds Ratio.

Figure 6-9 provides the macro-averaged F_1 scores of Naïve Bayes and *boostingUp*. It shows that the best macro-averaged F_1 score is 38.63% achieved by *boostingUp* when 100 features



	50	80	100	200	300	500	1000	2000	3000	5000	10000
NBIM	30.25	37.36	38.29	28.21	29.53	28.3	28.05	20.85	26.46	28.43	24.77
BUIM	30.55	38.38	38.63	28.71	29.53	28.77	23.41	23.8	31.13	30.41	29.99
NBOR	14.69	14.69	14.69	15.5	15.5	15.5	15.5	15.5	16.8	21.19	13.01
BUOR	16.42	16.42	16.42	17.62	17.62	17.62	17.62	17.62	20.1	25.73	10.33
NBRS	22.67	22.22	26.45	21.22	31.1	33.27	35.95	34.23	28.41	32.07	28.28
BURS	26.66	26.46	27.5	22.61	31.67	31.01	35.01	31.33	25.81	29.12	26.72

Figure 6-9 Macro-averaged F_1 scores of NB and *boostingUp* evaluated by T100G



	50	80	100	200	300	500	1000	2000	3000	5000	10000
KMIM	33.66	37.62	37.62	35.29	35.64	35	34.31	29.41	35	38	35.64
BUIM	32.67	37.62	36.63	36.27	35.64	36	31.68	34.31	42	41	40
KMOR	19.61	19.61	19.61	20.59	20.59	20.59	20.59	20.59	22.55	30.48	19.8
BUOR	23.53	23.53	23.53	25.49	25.49	25.49	25.49	25.49	28.43	39.42	17
KMRS	30	33.33	35.64	31.37	42.57	49.02	48	47	38	45	40
BURS	37	40.59	37.62	34.31	43.56	45.1	47	43	35	42	37

Figure 6-10 Micro-averaged F_1 scores of NB and *boostingUp* evaluated by T100G

are selected by Information Gain. The figure illustrates that *boostingUp* can almost always improve Naïve Bayes when features are chosen by Information Gain and Odds Ratio. When features are picked up by Relevancy Score, *boostingUp* can boost Naïve Bayes if feature number is less than 300.

Figure 6-10 presents the micro-averaged F_1 scores for Naïve Bayes and *boostingUp*. The results provided in this figure are similar to that of in Figure 6-9. The figure illustrates that when Odds Ratio is the feature selection algorithm, Naïve Bayes can always be boosted by *boostingUp* except when 10000 features are chosen. If Information Gain is employed to select features and when the feature number is bigger than 2000, *boostingUp* outperforms Naïve Bayes. Using Relevancy Score to select less than 300 features enable the performance of Naïve Bayes can be improved by *boostingUp*. The best micro-averaged F_1 is very close to 50% obtained by Naïve Bayes when 500 features are selected Relevancy Score algorithm.

The improvements of *boostingUp* over K-Means and Naïve Bayes are provided in Table 6-2. The data in the table shows that a maximum 12.67% increase of ARI (from 13.15% to 25.82%) over K-Means is achieved by *boostingUp* when 2000 features are selected by Relevancy Score, with an average 5.79% improvement of ARI (from 13.23% to 19.03%) over all the results when 50, 80, 100, 200, 300, 500, 1000, 2000, 3000, 5000, and 10000 features are chosen. The biggest improvement of Naïve Bayes in terms of micro-averaged F_1 is 8.94% (from 30.48% to 39.42%), achieved by *boostingUp* when 5000 features are chosen by Odds Ratio, with an average boosting of 4.39% over all the different numbers of features

Table 6-2 Improvements of *boostingUp* for K-Means and NB (%)

	#Features	CF_1	ARI	MAF_1	MIF_1
IG	50	4.24	4.33	0.3	-0.99
	80	5.78	6.03	1.02	0
	100	4.69	4.82	0.34	-0.99
	200	3.06	3.22	0.5	0.98
	300	2.74	2.76	0	0
	500	2.55	2.62	0.47	1
	1000	3.29	3.19	-4.64	-2.63
	2000	2.79	2.8	2.95	4.9
	3000	-1.1	-1.33	4.67	7
	5000	1.95	1.78	1.98	3
	10000	1.54	1.76	**5.22**	4.36
	AVG	2.87	2.91	**1.16**	1.51
	MAX	5.78	6.03	5.22	7.0
	MIN	-1.1	-1.33	-4.64	-2.63
OR	50	-1.52	-1.88	1.73	3.92
	80	-1.52	-1.88	1.73	3.92
	100	-1.52	-1.88	1.73	3.92
	200	-1.57	-2.04	2.12	4.9
	300	-1.57	-2.04	2.12	4.9
	500	-1.57	-2.04	2.12	4.9
	1000	-1.57	-2.04	2.12	4.9
	2000	-1.21	-2.04	2.12	4.9
	3000	-1.57	-2.04	3.3	5.88
	5000	0.57	0.32	4.54	**8.94**
	10000	-0.35	-0.71	-2.68	-2.8
	AVG	-1.22	-1.66	1.9	**4.39**
	MAX	0.57	0.32	4.54	8.94
	MIN	-1.57	-2.04	-2.68	-2.8
RS	50	-0.58	-0.89	3.99	7
	80	0.14	-0.11	4.24	7.26
	100	2.37	2.43	1.05	1.98
	200	4.16	4.33	1.39	2.94
	300	5.95	6.37	0.57	0.99
	500	10.61	11.43	-2.26	-3.92
	1000	4.67	5.1	-0.94	-1
	2000	11.73	**12.67**	-2.9	-4
	3000	5.04	5.46	-2.6	-3
	5000	9.07	9.88	-2.95	-3
	10000	6.67	7.07	-1.56	-3
	AVG	5.44	**5.8**	-0.18	0.2
	MAX	11.73	12.67	4.24	7.26
	MIN	-0.58	-0.89	-2.95	-4
Summary	AVG	**2.36**	**2.35**	**0.98**	**2.03**
	MAX	11.73	12.67	5.22	8.94
	MIN	-1.57	-2.04	-4.64	-4

Abbreviation in this table: CF_1: F_1 of K-Means, ARI: Adjusted Rand Index, MAF_1: macro-averaged F_1: MIF_1: micro-averaged F_1, IM: Information Gain, OR: Odds Ratio, RS: Relevancy Score

picked up by Odds Ratio. For macro-averaged F_1, the maximum enhancement is 5.22% (from 24.77% to 29.99%) when 10000 features are chosen by Information Gain feature selection algorithm.

6.4.3 Discussion

Based on the experimental data presented in Figure 6-7 to Figure 6-10 and Table 6-2, it can be concluded that the proposed *boostingUp* algorithm is able to improve the performance of both text categorization and clustering algorithms if the proper feature selection algorithm is utilized. A significant 12.67% ARI (from 13.15% to 25.82%) and 8.94% micro-averaged F_1 improvements (from 30.48% to 39.42%) of *boostingUp* over K-Means and NB are encouraging. On average, the improvements for K-Means of *boostingUp* over the three different feature selection algorithms are 2.35% (from 13.17% to 15.52%) and 2.37% (from 21.45% to 23.82%) respectively with respect to ARI and F_1. The average improvements for Naïve Bayes by *boostingUp* over the three feature selection algorithms are 0.97% (from 24.51% to 25.48%) and 2.04% (from 32.17% to 34.21%) respectively with regard to macro-averaged/micro-averaged F_1 value (Table 6-2).

Using Relevancy Score to select more than 100 features, *boostingUp* can consistently improve the performance of K-Means satisfactorily.

Utilizing Odds Ratio to choose less than 5000 features, *boostingUp* provides a more than 5% improvement of micro-averaged F_1 for Naïve Bayes.

Some issues related to the proposed *boostingUp* algorithm. The first one is the labels of the 100 test Web snippets provided by Google Directory are taken as the unique correct assignments. This assumption involves another two issues, the subjectiveness of labeling a Web snippet, and the multi-labeling of a Web snippet. Subjective characteristic of Web snippet labeling, or the relevance of a Web snippet to a given label, results in one document one label strategy is arguable. Therefore, multi-labeling strategy may alleviate this issue, however, multi-labeling may face the same issue of the subjectiveness of relevance judgment. Detailed discussion of the two issues is out of the scope of this research, and leads to an interesting further research direction i.e. the evaluation metrics of multi-labeling strategy.

The second issue of *boostingUp* is that it sometimes impairs the performance of clustering while boosting categorization, and vice versa. The understanding of this phenomenon is that if both clustering and categorization algorithms perform relatively well, the performances of both are to be enhanced by *boostingUp*. The reason is that *boostingUp* adjusts results of categorization and clustering based on the assumption that in an ideal scenario, categorization and clustering ought to generate identical results that are the true groups of a document collection. Therefore, the better the clustering and categorization algorithm performs, the more improvements *boostingUp* should produce. If both perform badly, it is uncertain how *boostingUp* affects their performance. Provided one of categorization or clustering performs well and the other poorly, the poorly performed algorithm may be enhanced by *boostingUp*, nevertheless, the performance of the well-performed algorithm might be negatively affected by *boostingUp*.

This intuitive understanding of *boostingUp* is verified by the experimental results presented. First of all, two observations can be obtained from Table 6-7 and Table 6-8. One observation is that for the three different feature selection algorithms, the performance of K-Means is relative stable in terms of ARI (Table 6-7, KMIR, KMOR, KMRS) and F_1 (Table 6-8). The second observation is that the performance of Naïve Bayes changes dramatically for different feature selection algorithms. In this experiment, for Web snippet categorization, generally speaking, experimental data in the two figures demonstrate the following performance order RS > IG > OR in terms of both macro-averaged and micro-averaged F_1 holds. According to the two observations, the *boostingUp* is expected to enhance the performance of clustering in a similar order. Experimental data in Table 6-2 reveal that the average improvements of K-Means by *boostingUp* for RS, IG and OR are 5.79%, 2.91% and 0.32% in terms of ARI, and 5.44%, 2.87% and 0.57% in terms of F_1, all in the same order as expected.

6.4.4 A Sample Run of *boostingUp*

Figure 6-11 presents a snapshot of output log file of *boostingUp* algorithm. For the sake of explanation, line number is added.

In the line 1 of this figure, ODPDataHashmapL4 indicates the training data set is the fourth level ODP *categoryDocuments*, this line is used to record feature selection algorithm, number of features, and the text clustering algorithm utilized by *boostingUp*.

The line 2 and 3 present the results of precision, recall, Rand Index, Adjusted Rand Index, and F_1 scores of K-Means clustering algorithm. Line 4, 5, and line 6, 7 show the macro-averaged and micro-averaged precision (Pr), recall (Re), and F_1 results of Naïve Bayes classifier. Line 8 to line 21 demonstrates how *boostingUp* adjusts categorized results based on clustered results; and Line 22 to line 26 is the traces of the adjustment of clustered results according to categorized the results of Naïve Bayes. Line 28 and 29 provide similar results as line 2 and 3, the evaluated clustering results of *boostingUp*, and line 30 to 33 are the evaluated categorization results of *boostingUp*.

Each of the 100 Web snippet has an unique identifier (ID) which is composed of a label (only considering the top 14 ODP categories) given by Google Directory, appended by a sequence number from 1 to 100. For example, the ID “sports_9” indicates the Web snippet is assigned the label “Sport” by Google Directory and it is the ninth result in the reversed order (refer to Appendix 2).

```
======================================================
1. Using ODPDataHashmapL4, information gain  algorithm, feature number: 2000 clsters ititiallized by adjusted max-min

2. Clustering: precision, recall, Rand index, Adjusted rand index, and F1
3. 0.2096436    0.22026432    0.85232323  0.13337472    0.21482277

4. Macaveraged Pr, Re, F1:
5. 0.24017572    0.21822226    0.20852344

6. Micaveraged Pr, Re, F1:
7. 0.29411766    0.29411766    0.29411766

8. Categorization: Moving arts_30 from shopping to games
9. Categorization: Moving science_44 from shopping to society
10. Categorization: Moving business_17 from recreation to society
11. Categorization: Moving shopping_15 from business to computers
12. Categorization: Moving reference_22 from science to reference
13. Categorization: Moving computers_40 from home to computers
14. Categorization: Moving computers_36 from science to computers
15. Categorization: Moving computers_42 from society to computers
16. Categorization: Moving arts_80 from society to recreation
17. Categorization: Moving science_8 from arts to society
18. Categorization: Moving computers_76 from reference to society
19. Categorization: Moving business_51 from society to reference
20. Categorization: Moving arts_26 from society to recreation
21. Categorization: Moving games_45 from kids and teens to games

22. Clustering: Moving sports_61 from sports_9 to games_79
23. Clustering: Moving games_3 from society_21 to games_79
24. Clustering: Moving arts_80 from games_87 to science_44
25. Clustering: Moving science_6 from society_32 to computers_12
26. Clustering: Moving computers_97 from society_93 to computers_12
------------------------------------------------

27. The new results
28. Clustering: precision, recall, Rand index, Adjusted rand index, and F1
29. 0.22941177    0.25770926    0.85252523  0.16135186    0.24273859

30. Macaveraged Pr, Re, F1:
31. 0.25749478    0.2640098    0.23800074

32. Micaveraged Pr, Re, F1:
33. 0.34313726    0.34313726    0.34313726
```

Figure 6-11 A record from the log file of *boostingUp*

Inspecting line 8 to 26, it can be found that line 12 to 15 and line 21 demonstrates that *boostingUp* correctly adjusted categorization results. Similarly, line 23 and 26 demonstrate clustering results are adjusted properly by *boostingUp*. However, note also that line 22 illustrates *boostingUp* sometimes make an inappropriate adjusting decision. Other adjustment decisions of *boostingUp* impose no impacts on the performance of categorization or clustering.

Since some of the categorization results are properly adjusted, the performance of categorization is improved from 20.82% and 29.41% in terms of macro-averaged / micro-averaged F_1 value to 23.8% and 34.31%. With two correct and one wrong adjustment decision of clustering results, a slightly improvement of clustering is achieved, 2.8% (16.14 % to 13.34%) and 2.78% (24.27% to 21.48%) in terms of ARI and F_1.

6.5 Summary

This section provided implementation details of clustering algorithms such as K-Means, HAC, and the proposed novel algorithm – *boostingUp*. The implementation of centroid initialization approaches such as Max-Min, Anomalous Pattern, and using LSA to calculate a proximity matrix for HAC are also introduced. The implemented clustering algorithms were evaluated by using the ODP *categoryDocument* sets generated in Chapter 4, and L100G, T50G, and T50O, three other sets of labeled Web snippets obtained as the search results of Google Directory and ODP. Based on the experimental results, Naïve Bayes and K-Means with MaxMin are selected as the categorization and clustering algorithm for *boostingUp*. Using the fourth level ODP *categoryDocuments* as training set and L100G as test set, the proposed *boostingUp* algorithm was able to improve the performance of both text categorization and clustering. Evaluation results based on three different feature selection algorithms to select a list number of features demonstrated that *boostingUp* improves on average Web snippet categorization by 0.97% and 2.04% in terms of macro-averaged and micro-averaged F_1, and enhances Web snippets clustering by an averaged 2.35% and 2.37% in terms of ARI and F_1.

In next Chapter, the *boostingUp* algorithm is to be employed to categorize the 1250 unlabeled search results (of 25 ambiguous search-terms) obtained by utilizing *Yahoo! Search Web Services API*. According to user relevance judgments of the results and a stipulated ranked list of user search preferences, the performance of RIB as judged by its recommended results is to be evaluated and compared with the results directly obtained from Yahoo!.

7 Evaluation of RIB

This chapter focuses on evaluating the performance of Recommender Intelligent Browser (RIB) in terms of precision and recall to verify whether one of the designed research objective of this project, **RO1** as described in Chapter 3, is achieved. Section 7.1 explains how the information stored in a user personal computer is collected and how user profile is created based on the collected information. Section 7.2 explains how evaluation criteria, precision and recall, are calculated in the experiments to be undertaken. In Section 7.3, RIB is evaluated by using two sets of Web search results from Google Directory and ODP respectively. Section 7.4 firstly introduces how the 25 search-terms are selected, and then discusses the concept of relevance judgment. This is followed by an explanation of how the categorized results are re-organized when calculating precision and recall, and in the end of this section, experimental results are presented. Section 7.5 discusses some limitations related in the research project and experiments, and Section 7.6 summarizes the chapter.

7.1 User Profiling

The purpose of user profiling in the field of Web information retrieval is to discover and capture user search preferences and usage patterns, and subsequently to utilize the collected information as implicit user feedback to improve the relevance of Web search results by means of query (search-term) expansion, search results re-ordering, or search results categorization/clustering.

This research concentrates on improving relevance of Web searching by recommending the user search results categorized under the 14 top-level ODP categories. A user profile is first created which is then used as a reference of user search preferences based on which Web hits are re-organized and recommended. To achieve the target, the user profile should also utilize the 14 top-level ODP categories to represent user search preferences by designing a weighting strategy that assigns/valuates the ODP categories. The main idea is to utilize the accumulated categorization status value of a classifier weight the 14 ODP categories, as described in detail in the following subsections.

7.1.1 User Profile Creation

User profile creation in RIB is composed of two steps. The first step is an initialization step that collects information stored in user personal computer. The second step is a continuous procedure that involves continuously collecting and recording Web snippets visited by the user, obtaining a list of categorization status values given by a trained classifier, and consequently updating the created user profile based on the approach proposed in Chapter 3, Section 3.2.

7.1.1.1 Personal Computer Information Collection

Figure 7.1 illustrates the procedure of personal computer information collection. A folder named as *cpath* in a personal computer is taken as input by the *getPersonalInfo()* process. Variable *fileType* contains the types of file, such as DOC, PDF, or TXT to be processed. The files under the given folder *cpath* are all extracted and put in an array named as *sa*. The

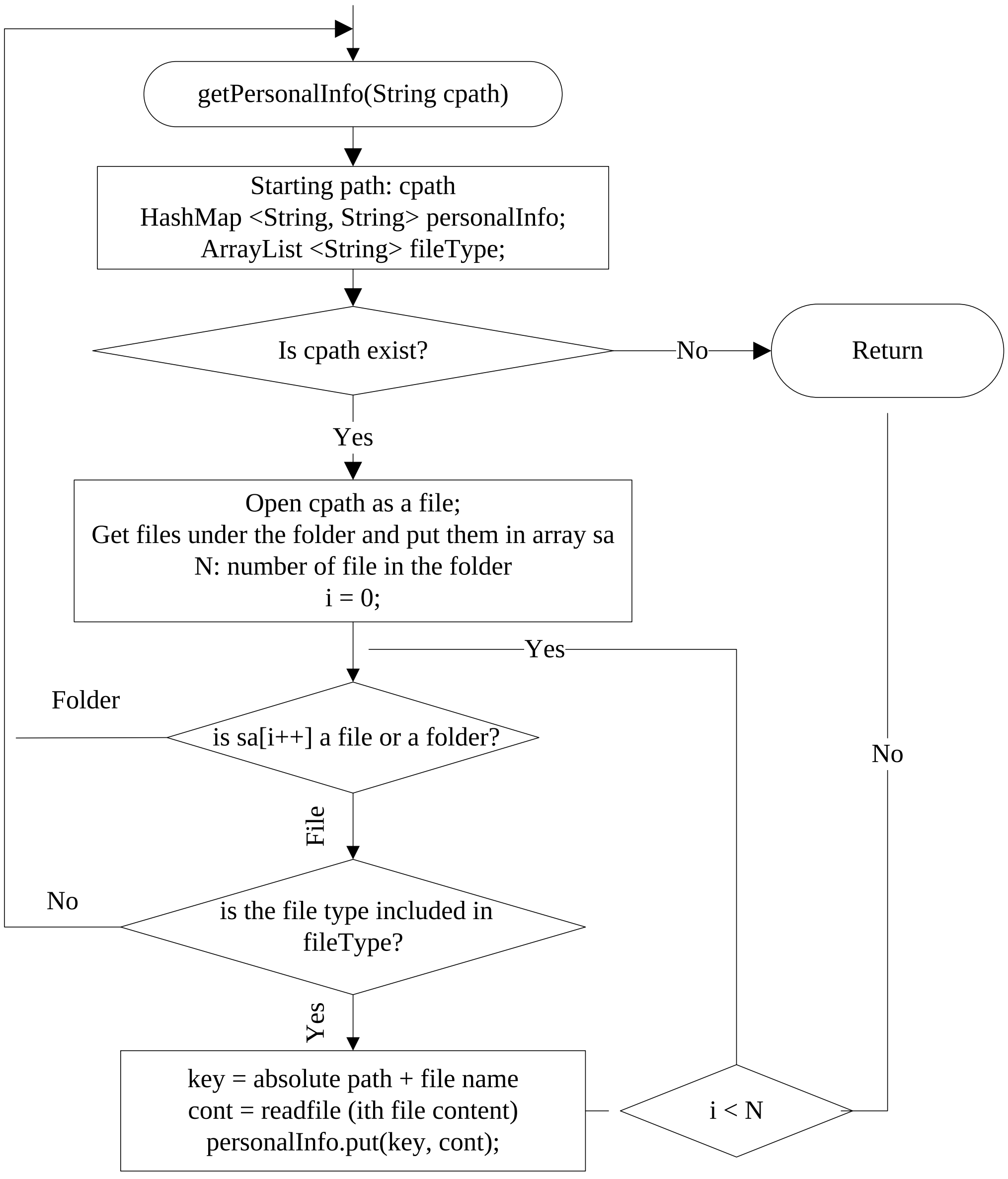

getPersonalInfo(String cpath)
Starting path: cpath
HashMap <String, String> personalInfo;
ArrayList <String> fileType;
Is cpath exist?
No
Return
Yes
Open cpath as a file;
Get files under the folder and put them in array sa
N: number of file in the folder
i = 0;
Yes
Folder
is sa[i++] a file or a folder?
No
File
No
is the file type included in fileType?
Yes
key = absolute path + file name
cont = readfile (ith file content)
personalInfo.put(key, cont);
i < N

Figure 7-1 Flowchart of get information from personal computer

elements in the array are analyzed one by one. If an element is a folder, the *getPersonalInfo()* recursively calls itself with the folder's name as an input parameter; if the element is a file, then a check is made if the type of the file is included in *filtType*, if it is, the absolute path and the name of the file are combined to form a unique identifier *key* of the file, the content of the file *cont* is read in, and then, the <*key*, *cont*> tuple is inserted into the *PersonalInfo* HashMap <String, String> structure. The i++ operator indicates variable *i* incremented by one. In the end, all files under the folder *cpath* and with their file types contained in *fileType* will be processed and stored in *PersonalInfo*.

The set *cpath* contains a list of hard driver names, such as "c:" or "d:" and several other folders that belong to a personal computer. This list is obtained by simply initialize *cpath* with, take Microsoft Windows XP Operating System as an example, "My Computer". All the information under the list will be collected and analyzed. The method illustrated in Figure 7-1 can also deal with only user created folders, or a combination of user created folders and a predefined list of folders such as "My Documents" created by an operating system.

7.1.1.2 User Profile Initialization and Updating

With the collected personal computer information, a user profile is initialized. The process is illustrated in Figure 7-2.

A Naïve Bayes classifier is trained by the fourth level ODP *categoryDocuments* with 100 features as selected by Relevancy Score feature selection algorithm. Naïve Bayes and Relevancy Score are selected as the classifier and feature selection algorithm because experimental results presented in Chapter 6 demonstrate that it performs relative well in terms of effectiveness and efficiency.

An array *preferenceArray* with dimensionality of 14 is defined to represent users search

```
1. Using fourth level ODP categoryDocuments to a train NB classifier,
            100 features are selected by Relevancy Score algorithm;

2. Initialize the value of the elements in preferenceArray to zero;

3. For each document collected from personal computer,
            using the NB classifier to get a ranked list of probabilities
            that the document be assigned to different ODP categories;

            Add the probabilities in the ranked list to their corresponding
                        categories in preferenceArray

4. Return preferenceArray as an initialized user profile
```

Figure 7-2 User profile initialization procedure

preferences under the 14 top-level ODP categories. It can be easily expanded into a matrix and thus can represent search preferences by not only the top-level ODP categories, but also plus the second, or third, or even the fourth level of ODP categories. The elements in the array are initially set to zero.

To initialize the *preferenceArray*, for each of the documents in *PersonalInfo*, using the trained NB classifier produces a ranked list of probability that the document is to be assigned to an ODP category. Accumulating the probabilities into *preferenceArray* until all the documents in *PersonalInfo* is processed. Then, the values in the *preferenceArray* can be taken as the measures or weights of user search preferences represented by ODP categories.

To update the *preferenceArray* dynamically, an approach described in Chapter 3 is utilized. Refer to Chapter 3, Section 3.2 for details.

7.2 Effectiveness Measurement

7.2.1 Precision, Recall and F_1

The most widely used effectiveness measurements of an information retrieval system are precision, recall and F_1. Precision is the fraction of retrieved documents that are relevant, recall is the fraction of relevant documents retrieved over all relevant documents, and F_1 = 2×precision×recall/(precision + recall) is the weighted harmonic measure of precision and recall (Manning, Raghavan, and Schütze 2008). Precision, recall and F_1 are collection based measures because to calculate the three measures, all the relevant documents in an information retrieval system for a given query have to be known a priori.

In the scenario of Web information retrieval, when a user submits a search-term, it is impossible to check all the indexed documents[30] and judge whether each of them is relevant or not to the information need expressed by the search-term, which themselves are usually ambiguous. Therefore, search engines return a ranked list of results that is calculated by some information retrieval models and algorithms that intend to estimate the relevant score of a document with regards to the given search-term.

7.2.2 Precision-recall Curve

The ranked list of search results is to be evaluated by an interpolated precision-recall curve which is now a standard measure of search engine effectiveness (Manning, Raghavan, and Schütze 2008). Considering the top T ranked results with R out of T of them are relevant,

[30] According to WorldWideWebSize.com (www.worldwidewebsize.com) Yahoo indexed about 56 billion webpages and Google indexed about 30 billion webpages. Retrieved on August 19, 2010.

and supposed that a user follows the ranked list to check the results one by one. If the i-th document in the ranked list is relevant and this is j-th relevant document so far encountered, then the recall at this moment is j/R, and the corresponding precision is j/i. If the i-th document is irrelevant, recall is kept unchanged and the precision is dropping down. Repeating this process produces R precision at recall level 1/R, 2/R, …, R/R.

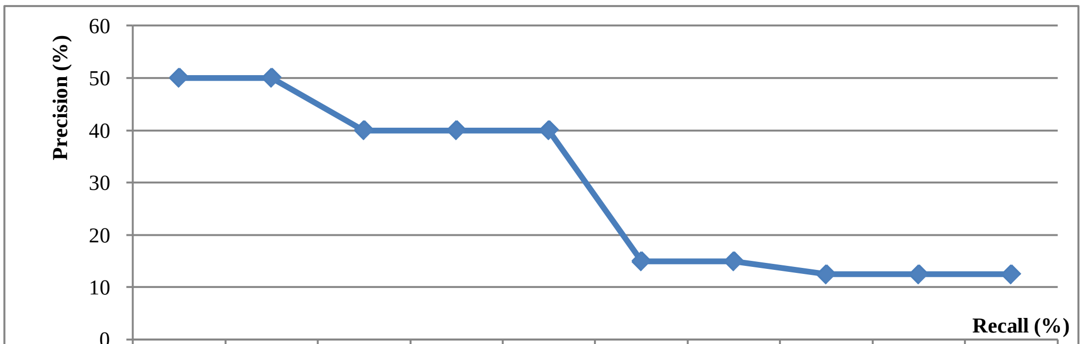
Precision-recall curve is usually drawn by using a standard interpolated precision at recall levels 0.1, 0.2, …, 1.0. The interpolated precision p_{ipx} at recall level $x \in \{0.1, 0.2, \ldots, 1.0\}$ is defined as precision found for recall level $r \geq x$, $r \in \{1/R, 2/R, \ldots, R/R\}$

(Equation 7-1) $p_{ipx} = \max_{r \geq x} p(r)$

For example, supposed that T = 50 and R = 4, that is, there are 50 returned results, four out of 50 are relevant, and the four relevant documents are ranked at 2nd, 5th, 20th, and 32th. For the first relevant document, the recall level is ¼ = 0.25, or more commonly, 25%, the precision for this document is ½ = 0.5, or 50%. When the second relevant document is met, the recall level is 2/4 = 50%, and the corresponding precision is 2/5 = 40%, similarly, when the third relevant document is met, recall is ¾ = 75%, and precision is 3/20 = 15%. For the last relevant document, the recall and precision is 4/4 = 100% and 4/32 = 12.5% respectively.

To interpolate the precision at different recall levels, such as recall level 60%, check all calculated precisions at their corresponding recall levels that are no less the 60%. In this case, there are two precisions, 15% at recall level 75%, and 12.5% at recall level 100%. According to Equation 7-1, precision at recall level 60% is 15%. The precision at other different recall levels can be interpolated the similar way, and finally, a precision-recall curve for the ten recall levels can be drawn, as illustrated in Figure 7-3.

If there are N different search-terms, each of them has their own interpolated precision-recall



Recall (%)	10	20	30	40	50	60	70	80	90	100
Precision	50	50	40	40	40	15	15	12.5	12.5	12.5

Figure 7-3 An example of precision-recall curve

curve. An averaged interpolated precision p_{ipj} at a given recall level j over the N different result sets is

$$p_{ipj} = \frac{1}{N}\sum_{k=1}^{N} p_{ipjk}$$

A mean average precision (*map*) over the ten recall levels is

(Equation 7-2) $$map = \frac{1}{10}\sum_{j=1}^{10}\frac{1}{N}\sum_{k=1}^{N} p_{ipjk}$$

In the following experiments, interpolated precision-recall curves are provided and compared against different search result sets, and mean averaged precisions are calculated as well.

7.2.3 Precision at Cut-off Level λ (P@ λ)

Another effectiveness measure of Web search results is precision at cut-off level λ, or simply P@ λ where λ are usually takes value 5, 10, 20, or 30. A cut-off level, according to Buckley and Voorhees (2000), “is a rank that defines the retrieved set”. A cut-off level of 20 indicates a set of top 20 retrieved results at the ranked list.

7.3 Recommendation of RIB – A Case Study

To evaluate the effectiveness of the proposed personalization approach, an imaginary user is created. Supposed that the user search preferences are represented by the weighted 14 top-level ODP categories, and the stipulated search interest of the imaginary user is “Computers” and “Sports”. Further, supposed that the search interest is represented by all of the c*ategoryDocuments* under the folder “Computers : Open source” and “Sports : Badminton” plus “Sports : Rodeo”. Then, these documents will be analyzed by *getPersonalInfo()* process. The number of *categoryDocuments* contained in the first folder is 53, and the number of documents in the second folder is 65 in which 34 of them belong to “Sports : Badminton” and 31 of them belong to “Sports : Rodeo”. The reason to include Badminton plus Rodeo is to make the number of documents contained in this folder is as close as to the number of documents in the folder “Computer : Open Source”.

With the above documents collected by *getPersonalInfo()* process, a user search preference profile is to be initialized. That is, based on the collected documents to assign weights to the 14 top-level ODP categories. To assign weights to the categories, a Naïve Bayes classifier is trained by using the fourth level ODP *categoryDocument* set with 100 features selected by Relevancy Score algorithm (Chapter 2, Section 2.3.1). The classifier predicts a ranked list of categories for each of the documents under the above two folders, and the corresponding probabilities estimated by the classifier for the top-level 14 ODP categories (category status value) are accumulated to the *preferenceArray*. The accumulated values for the 14 different categories are assumed can represent search preferences of the imaginary user. Appendix 6 part A is the output of the user profile. It shows that the initialized *preferenceArray* has a ranked preference list of Sports > Computer > Business > Arts > Society > Science > Recreation > Health > Shopping > Games > Kids and Teens > Home > Reference > News. This is in line with the two supposed user search interest: “Computers” and “Sports”, which are ranked the top two categories in the list.

Then, T50G and T50O Web snippet collections are employed as the test data sets for which the Web snippets are to be categorized by using the trained Naïve Bayes classifier to group them under the different categories of the 14 top-level ODP categories. The categorized results of T50G (refer to Appendix 6) are grouped into Arts (14), Business (18), Recreation (1)[31], Society (1), Sports (16). For T50O, the results are grouped into Arts (20), Business (12), Games (1), Shopping (1), Society (2), and Sports (14). The number in the brackets is the number of documents in the category.

RIB presents personalized results by re-ranking the grouped results according to the search preference order in the created *preferenceArray*. That is, documents grouped into “Sports” and “Computers” are presented first, and the documents grouped under the other ODP categories are presented according to the order provided in *preferenceArray*.

To evaluate the effectiveness of the personalized results, the relevance of the Web snippets need to be judged. To simplify the relevance judgment decision, assume that any Web snippets have the label starting with “Sports” or “Computers” are relevant, and the rest are not relevant. This assumption is based on the fact that the results of ODP are human edited and consequently if a result has the label “Sports”, the result is first judged sports related by the writer of the Web site when it is submitted to ODP, and then a human editor also agree the result is relevant to sports and thus it is classified to the category “Sports”. Based on this assumption, the relevance of the Web snippets can easily be determined.

The personalized results recommended by RIB based on the generated ranked preferences in *preferenceArray* are attached in Appendix 6. Part B is the personalized results of T50G, and Part C is the personalized results of T50O. Figure 7-4 and Figure 7-5 demonstrate the recall precision curves of the personalized and non-personalized results based on the two test data sets T50G and T50O.

---

[31] The categorized cluster that contains only one document is not merged with other clusters in this experiment because except categories “Sports” and “Computers” which are assumed the favourable search topics of the user, all other categories are merged to form a unique cluster.

7.3.1 Experimental Results based on T50G

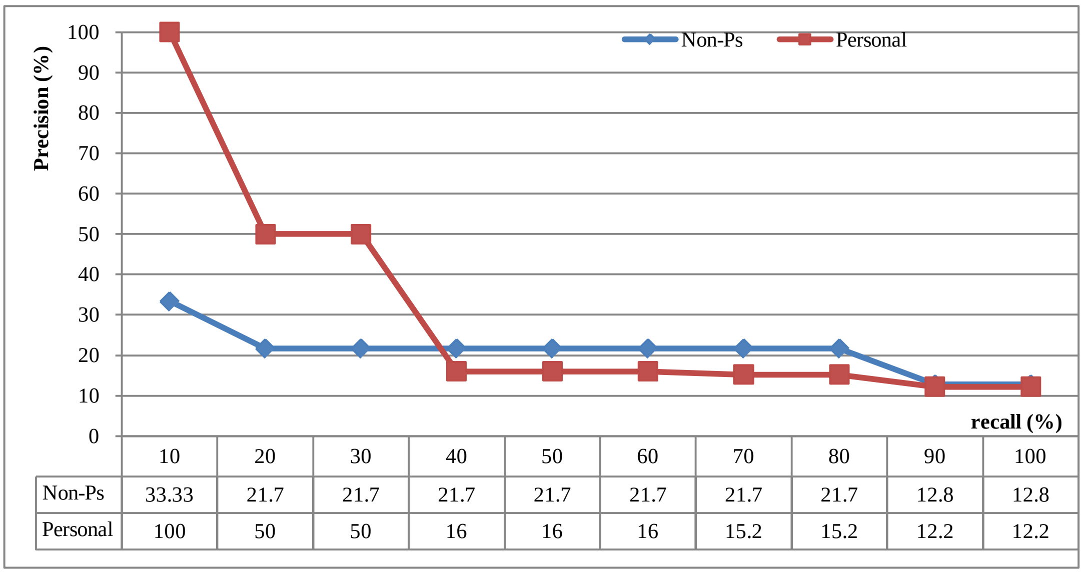
The recall-precision curve in Figure 7-4 illustrates that the personalized results recommended by RIB are more relevant than the results directly retrieved from Google Directory The averaged precision of personalized results is 30.28%, 9.2% improvement over the 20.08% of Google Directory.

If compared with P@10 (Section 7.2.3), the results of Google Directory and RIB are both 20% because among the top 10 returned results, both contain two relevant results. RIB ranks the results at the first and fourth position, whereas Google Directory arranged two relevant results at the third and tenth position.

7.3.2 Experimental Results based on T50O

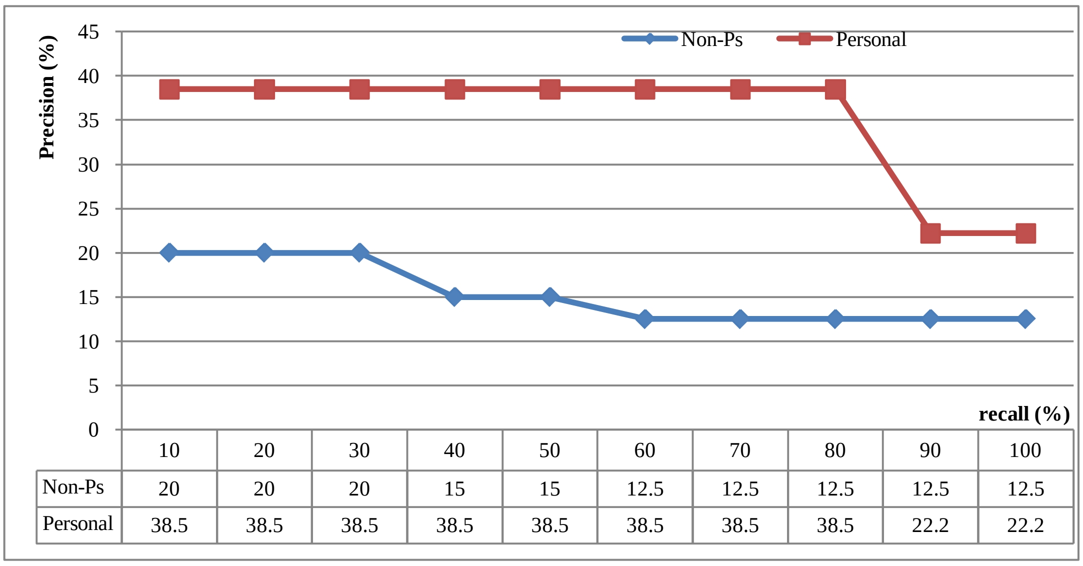
Figure 7-5 presents the personalized and non-personalized results of the top 50 Web snippets returned by ODP when "jaguar" is the search-term. As can be seen from this figure, personalized result of RIB once again outperforms the non-personalized results of ODP. The averaged precision over the ten recall levels of the personalized and non-personalized results are 35.24% and 15.25% respectively, the improvement is 19.99%.

When estimated by P@10 as in the previous section, the personalized results recommended by RIB is 30%, compared with 20% of the non-personalized results, a 10% performance improvement is achieved. Since a majority number (no less than 50%) of Web users browse only the first page of Web search results (Jansen and Spink 2006), - a higher score of P@10



	10	20	30	40	50	60	70	80	90	100
Non-Ps	33.33	21.7	21.7	21.7	21.7	21.7	21.7	21.7	12.8	12.8
Personal	100	50	50	16	16	16	15.2	15.2	12.2	12.2

Figure 7-4 Recall-precision curve of personalized results of RIB and non-personalized results of top 50 results of Google Directory



	10	20	30	40	50	60	70	80	90	100
Non-Ps	20	20	20	15	15	12.5	12.5	12.5	12.5	12.5
Personal	38.5	38.5	38.5	38.5	38.5	38.5	38.5	38.5	22.2	22.2

Figure 7-5 Recall-precision curve of personalized results of RIB and non-personalized results of top 50 results of Open Directory Project

is more attractive to users.

According to the two sets of experimental results, using the automatically created user profile based on the 14 top-level categories, RIB can improve the precision of Web search results by an average 14.59% ((9.2 + 19.99) / 2).

7.4 Comparison between Yahoo! and RIB

This section compares the relevance of Web search results of Yahoo! and RIB. The main focuses of this section includes ambiguous search-term selection, relevance judgment, Web snippet re-organization, and experimental results presentation.

7.4.1 Ambiguous Search-term Selection

Search-term selection plays a critical role when an information retrieval system is evaluated because the search-terms represent user information needs the information retrieval system intends to satisfy (Mizzaro 1998). Three principles suggested by Zhu (2007) are followed in this research when search-terms are selected. First, the search-terms should be real terms of real users; second, the search-terms are usually very short one or two words terms; third, the search-terms should cover a wide range of topics, and subsequently, the quantity of the search-terms should no less than 20 (refer to Section 7.5.1.4). Obviously, a large number of search-terms is more preferable to ensure a sound conclusion. However, since each search-term will produce 50 search results that need five human judges to do relevance judgments, too many search-terms will results in unaffordable human labour in this project.

Based on the above principles, 25 search-terms provide by Zeng et al. (2004) as show in Appendix 3, Appendix Table 3-1 are utilized in this research[32]. Checking for the three principles, first, these are search-terms submitted to Microsoft MSN search engine by real users. Second, 21 out of 25 of them are one word search-term, three out of 25 are two words search-terms, and one out of 25 is a three-word search-term. Third, the 25 search-terms are categorized into three groups including ambiguous terms, entity names and general terms that cover people, place, things, travel, computer technology, entertainment, and education. All of these are frequently searched topics as suggested by Jansen et al. (2005).

All of the search-terms have more than one meaning, and one and only one information need is to be stipulated for each of them. Because more and more people are using search engines to obtain information from the Web, search engines have to accommodate a wide range of user types with different search skill levels (Smyth 2007), and it is not uncommon that users have difficulties to express their information needs to fit search engines (Baeza-Yates and Ribeiro-Neto 1999). Taking into account this observation, therefore, for each search-term, a simple possible information need is specified for each of the 25 search-terms. The existence of other information needs, and which of them is more appropriate is ignored with the purpose to make the stipulated information needs as close to real user situation as possible. The stipulated information needs for the 25 search-terms are presented in Appendix 4.

7.4.2 Relevance Judgment

Relevance refers to "the ability (as of an information retrieval system) to retrieve material that satisfies the needs of the user" (Saracevic 2007a). Relevance has been recognized as an essential concept in the field of information retrieval since late 1950s, and two dominant contradictory perspectives of relevance, system-oriented, and user-oriented, with others are developed based on different assumptions made (Hjørland 2010; Saracevic 1975, 2007a, 2007b; Borlund 2003; Xu and Chen 2006; Saracevic 2008). According to Mizzaro (1997, 1998), relevance judgment is inherently subjective and involves psychological factors (Harter 1992) and no consensus exists on the relevance. Hjørland (2010) argues that the "subject knowledge view" suggested by Saracevic (1975) can be extended to a socio-cognitive paradigm of relevance for which the consensus or paradigm is in a given field, or it is domain oriented, and a currently irrelevant document may in the future become relevant. Delving into details of the perspectives of relevance is out the scope of this research. For evaluation purpose, a subjective knowledge view is taken in this research. However, this

[32] In Zeng et al.'s research, 30 search-terms are provided. In the previous study of this project, five of them have been used to evaluate the performance of a Special Search Browser. Consequently, the remaining 25 features are selected in this research.

view differs from the perspective of Hjørland (2010) that a relevant document may incorrectly judged as irrelevant, it is assumed that a summary of relevance judgment decisions from a group of judges is a neutrally relevance judgment decision which is accepted as the "gold standard for performance evaluation" (Saracevic 2008). This method corresponds to method four as suggested by Saracevic (Saracevic 2008).

Based on this assumption, in this research, for each of the returned Web snippets and a stipulated information need, five human judges are asked to provide their decisions on whether a Web snippet is relevant (R), partial relevant (P), irrelevant (I), or the snippet is not informative enough to make a decision (N).

There are 25 human judges were employed in this research. The judges were staff from School of Information Systems (Curtin University), as well as High Degree by Research (HDR) students in the field of Accounting, Economics and Finance, Management, Marketing, and Information Systems. The 25 judges are divided into five groups evenly (G1 to G5). Each group is provided with 250 different Web search results from five different search-terms. It takes about 20 minutes to two hours for a judge to finish the relevance judgment of the 250 Web snippets.

To scale the relevance judgment, a score is assigned to each of the relevance judgment decision. R is assigned positive three (3), P is assigned positive one (1), I is assigned negative three (-3), and N is assigned a score zero (0).

Since each of the returned Web snippets is to be judged by five judges, a final decision score is obtained by adding up the scores from the five judges. If the summed up value is bigger than zero, the Web snippet is judged as relevant; if the value is negative, the Web snippet is classified as irrelevant; if the summed up value equals zero, the website linked by the Web snippet is visited and a final judgment decision is made after the content of the website is checked.

7.4.3 Search Results Re-organization

Referring to Figure 3-1 in Chapter 3, RIB re-organizes Web search results in the following five steps.

Step 1: For each of the 25 search-terms, 50 Web snippets are obtained by using *Yahoo! Search Web Services API*.

Step 2: A Naïve Bayes classifier is trained by using the fourth level ODP *categoryDocument* set. 100 features are selected by using Relevancy Score feature selection algorithm.

Step 3: 50 search results for each of the 25 search-terms are categorized by the trained Naïve Bayes classifier; the 50 search results are also grouped by using K-Means with MaxMin centroid initialization algorithm.

Step 4: The categorized and clustered results are taken as input by *boostingUp* algorithm that then outputs a new set of categorized results.

Step 5: Grouped results from Step 4 are re-arranged and presented to user.

The last step involves how are the categorized results re-ranked, and this subsequently affects the evaluation results of RIB. One approach is to rank the results based on user profile as discussed in the previous section. Another method is to allow user to dynamically select interest topics, and present only results categorized under the selected topics. The intuition behind this approach is that users themselves personalize how they search. The advantage of this method is it provides user with the autonomy to manipulate categorized results; the disadvantage is that user has to make a decision as to which topic is relevant (although this could be optional). However, as the user is getting familiar with the categories in the ODP, selecting an appropriate topic should not be difficult. Therefore, in addition to using user profile to re-organize the grouped results as described in the previous section, RIB provides options for user to choose whether the search results are personalized according to the created user profile, or the results are re-organized according to the topics selected by user. In the following experiments, search results are personalized according to the topics selected by user.

Asking users to pick up interest topics once again involves human relevance judgments. To simplify the experiments and avoid the involvement of human judges, the final categorized results are re-organized by using the following rules. First, the two categories that contain the most relevant results are selected; second, the rest of the results, no matter how many categories they are arranged, are combined together as the third category. The interpolated precision-recall curve of RIB is drawn according to this re-ranked order.

7.4.4 Results of RIB and *Yahoo! Search Web Services API*

7.4.4.1 Relevance Judgment Results for the 25 Search-terms

The detailed relevance judgment results of the 25 search-terms with 1250 Web snippets are presented in Appendix 7, which includes not only the relevance judgment results, but also precision, recall for Yahoo search results, and for the re-ranked results of RIB as discussed in Section 7.4.3. The interpolated precision-recall curves at ten different recall levels for all of the 25 search-terms are also presented in this appendix.

In each of the table in Appendix 7, column W-S is the list of the 50 search results returned from Yahoo. The relevance judgment decisions of five judges are presented in column R(3), P(1), I(-3) and N(0), representing Relevant, Partial Relevant, Irrelevant, and Not sufficient information to made a decision. Each of the five judges is assigned a number one (1) to five (5) and their decisions are represented by the numbers assigned to them. For example, in the first table in Appendix 7, at the cross of line 6 and column R(3), the value "1235" indicates there are four judges numbered 1, 2, 3 and 5 believe the Web snippet (in line 6) is relevant, and value "4" in column I(-3) at the same line indicates that the judge numbered 4 thinks the Web snippet is irrelevant. Column SC is the summed up score based on the judgment decisions of the five judges. The SC score for line 6 in the same table is $9 = 4\times3 + 1\times(-3)$ because a relevant judgment is assigned score 3 (four judges made this decision), and an irrelevant judgment has a score -3 (one judge made this decision).

Column JG is the final binary relevance judgment decision based on the value in SC as described in Section 7.4.2. Number "1" indicates the Web snippet is relevant and "0" otherwise. The value in column RL is the number of relevant documents at a given row so far encountered from the first line. For example, in the same table at line 22, the number "2" indicates that this Web snippet is the second relevant document encountered if the results are checked from the line 1 sequentially. Note that if a Web snippet is irrelevant, the value of the column is set to zero (0). Column Rc and Pr represent Recall and Precision respectively.

Column NR (New Rank) in the tables in Appendix 7 is the new re-ranked results of RIB. For example, number "6" at the first line in this column in the first table indicates that the sixth Web snippet of Yahoo is the top ranked results of RIB, and the second line of NR indicates that the result ranked 22nd by Yahoo is ranked 2nd by RIB. Similar to RL, Rc and Pr, RL', Rc' and Pr' are the number of relevant results so far encountered, Recall and Precision of RIB.

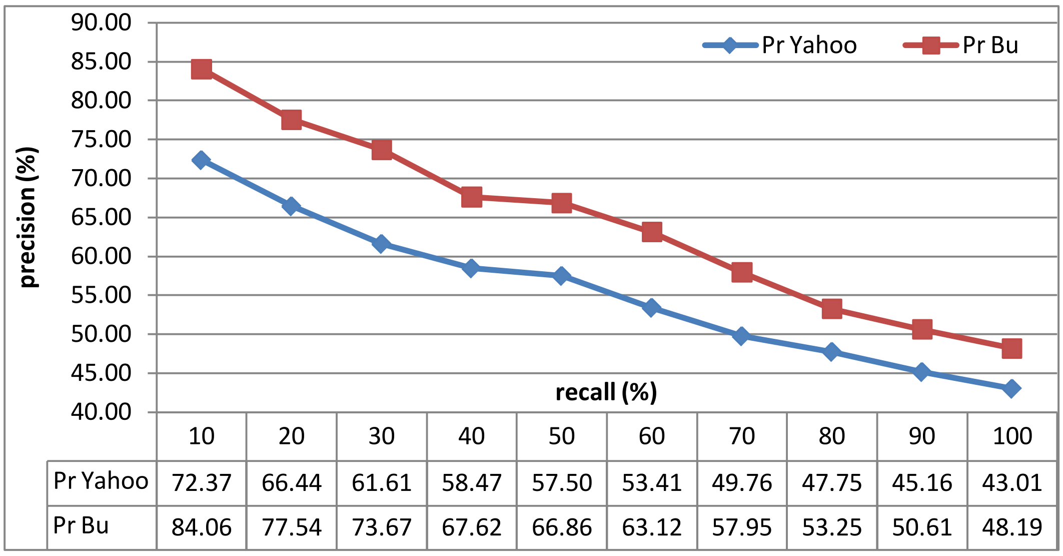
7.4.4.2 Precision-recall Curve of Yahoo and RIB over the Results of 25 Search-terms

Figure 7-6 illustrates the interpolated precision-recall curve of the search results of *Yahoo! Search Web Services API* and RIB based on the averaged results over the 25 search-terms and the relevance judgment results, which are the summed up results of five judges.

The improvement in terms of precision of RIB over Yahoo is shown in Figure 7-7.

The following observations can be obtained from the two figures.

1) Figure 7-6 demonstrates that the categorized and personalized results of RIB outperform the un-personalized results directly from *Yahoo! Search Web Services API* at all recall



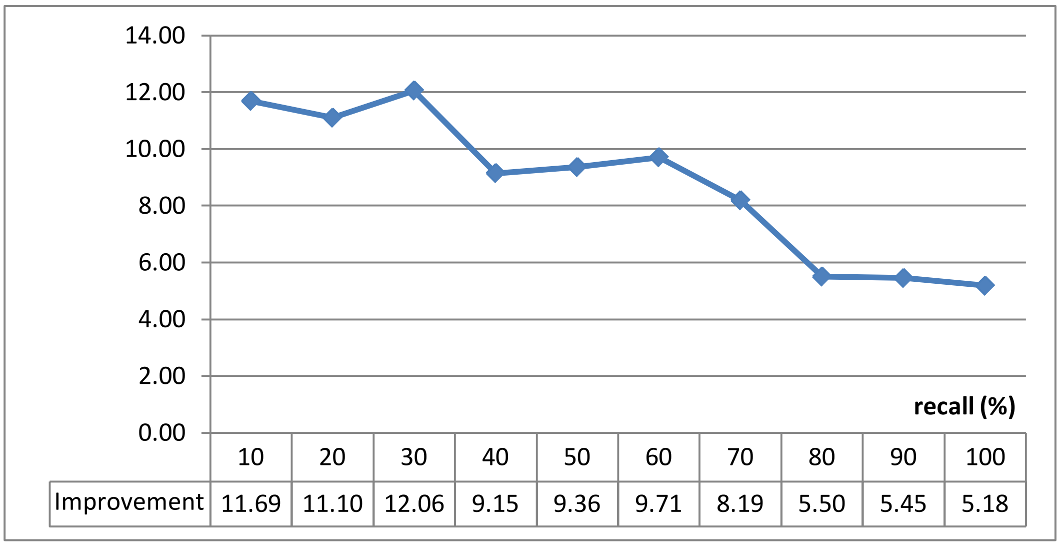
	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	72.37	66.44	61.61	58.47	57.50	53.41	49.76	47.75	45.16	43.01
Pr Bu	84.06	77.54	73.67	67.62	66.86	63.12	57.95	53.25	50.61	48.19

Figure 7-6 Recall-precision curve of Yahoo and RIB based on the averaged results over 25 search-terms

levels. The maximum improvement is 12.06% at the recall level 30%, and the minimum improvement is 5.18% at the recall level 100%.

2) The overall mean averaged precisions (map) of Yahoo and RIB over the results of 25 search-terms are 55.55% and 64.29% respectively; this indicates an 8.74% precision improvement of RIB over Yahoo.

3) Figure 7-7 shows that at the lower recall levels, such as those at level 10%, 20%, and 30%, the precision improvement is bigger than 10%; and the improvement in general decreases gradually and finally leads to a minimum of 5.18% improvement at the recall



	10	20	30	40	50	60	70	80	90	100
Improvement	11.69	11.10	12.06	9.15	9.36	9.71	8.19	5.50	5.45	5.18

Figure 7-7 Precision improvement of RIB over Yahoo at different recall levels based on the averaged results over 25 search-terms

level 100%. Higher precision at lower recall level is preferable according to the observation that users usually browse only a few pages of Web hits (Jansen and Spink 2006).

7.5 Discussion and Limitations

7.5.1 Personalization

7.5.1.1 Personal Information Collection

Creating a user profile which can not only represent user current search preferences but also deal with search concept drift is still an open research question, and a large number of techniques have been proposed to personalize Web search results as discussed in Chapter 2, Section 2.4. In this research, a predefined Web directory, the Open Directory Project, is employed to represent user search preferences. To initialize a user profile, with the assumption that the user agrees that the information can be collected and used for personalization purpose, information stored in a personal computer of the user is collected and categorized into the different ODP categories. This involves how to locate the information and how to process the stored information in different formats such as Microsoft Word document, PDF document, or HTML document.

To locate the information in a personal computer, one simple approach is to use the default folders such as “My Documents” created by an operating system. Information stored in a user created folder can also be collected by analyzing files in this kind of folders. With regard to format, one method is for each process circle, only one kind of format is dealt with, that is, the first circle processes only PDF format files, and the second circle process Microsoft Word documents. Although this can be implemented by using a specially developed recursive algorithm in RIB, the effectiveness of this approach needs to be verified by carrying out an experiment with large population. This kind of large-scale experiment is very hard to be carried out in this research project because personalization involves privacy issues and thus ethics issues which may prevent users from participating. Using an imaginary user to evaluate the effectiveness of the proposed user profiling approach is an obvious weakness of this research.

7.5.1.2 User Profile Initialization

The research uses a Naïve Bayes classifier, which is trained by the fourth level ODP *categoryDocuments*, to categorize the documents stored in a personal computer into the 14 top-level ODP categories. The scores assigned to each of the ODP categories by Naïve Bayes are accumulated as the weights of preferences for the categories. Because the training data set is totally independent from testing data set, the accuracy of the classifier may not be very high, and the ranked search preference list in terms of ODP categories may not exactly the same as the list arranged by users if users are asked to express their search preferences explicitly based on the 14 top-level OTP categories. Furthermore, the performance of Naïve Bayes is also affected by feature selection algorithms employed.

Other search preference weighting strategies also need to be evaluated. One such strategy is instead of using the real scores returned by the classifier to consider only the top ranked category for a test document, and increase the weight of the corresponding category in user profile. Another strategy is to consider the first three top ranked categories and added different values, such as, three (3), two (2), and one (1) to their corresponding categories. Experiments compare the different strategies are needed.

Despite the issues, it can be found that the precision improvements in the two case studies in Section 7.3 are encouraging, and this indicates that the user profile initialization approach utilized in this research is feasible and effective.

7.5.1.3 Search Concept Drift

The suggested approach in Chapter 3, Section 3.2 to address the search concept drift issue is not verified by experimental data in this research project. This is also because a large-scale experiment is too expensive to carry out with the limitations of timeframe and funding support. In addition, as pointed by Cornuéjols (2009), concept drift is still very much an open research issue and lack of a satisfying theoretical framework in this research field. Cornuéjols (2009) further suggests that it seems that research which focuses on the changes themselves rather than only on the drift might produce more fruitful results.

7.5.1.4 Number of Search-terms

To evaluate the effectiveness of an information retrieval system, the more search-terms used to evaluate the system, the more sound the conclusions are (Carterette et al. 2008). However, since the evaluation procedure involves human relevance judgment, which is inherently subjective (Hjørland 2010; Mizzaro 1997; Saracevic 2008) and human labour is expensive, large-scale human relevance judgment based evaluation is empirically difficult. Manning et al. (2008) suggest that 50 queries is the minimum number to evaluate an information retrieval system, Zeng et al. (2004) use 30 queries to evaluate their search results clustering algorithm, Leighton and Srivastava (1999) use 20 queries to compare the performance of five most popular search engines based on top 20 search results of the search engines, and Carterette et al. (Carterette et al. 2008) find that information retrieval systems are typically evaluated by using several dozen queries. Buckley and Voorhees (2000) argue that for a well designed experiment to obtain a desired level of confidence of experimental results, 25 queries is the minimum and 50 is better. Consequently, it is reasonable to select 25 search-terms in this search project.

7.5.1.5 Inconsistence Analysis of RIB

Checking Appendix 7, it can be found that for nine out of the 25 search-terms, the search results of Yahoo outperform the results of RIB. The nine search-terms are maps, music, jokes, games, disney, resume, susan dumais, graphic design, and saturn.

This phenomenon can be explained by the concept of relevance *Judgment Convergence Degree* (JCD) which is defined as the ratio of *Agreement Number* (AN) and *Judgment Number* (JN) proposed by Zhu (2007). That is:

JCD = AN/JN

For a given search-term, let N be the number of search results returned by an information retrieval system, m be the number of human judges, and k be the number of options for a human judge to select. Then, JN is defined as the total number of choices made by the m judges over all N×k possible choices; and AN is defined as the total number the sort of judgments for that all m judges make the same relevance judgment decision. For example, in this research, N is 50 which corresponding to the 50 search results for each of the 25 search-term, m is five (5) which corresponding to the five judges, and k is four (4) which corresponding to the four possible choices for a search result (P, R, I, or N).

Figure 7-8 illustrates the relationship between JCD and the precision improvement. Since no results are relevant to search-term “jobs”, so the search-term is excluded from the figure.

The figure demonstrates that as the improvement of precision increasing, in general, the JCD is increasing simultaneously; and when improvement is less than zero, the corresponding JCD is usually very small. The average JCD for the nine search-terms for which the precision improvement less than zero is 5.5%, whereas the averaged JCD for the search-terms for which the precision improvement is positive is about 19.88%, and the averaged JCD for those search-terms for which the precision improvement is greater than 5% is as high as almost 30% (29.18%).

The relationship between JCD and the precision improvement can be further verified by considering the correlation coefficient between precision improvement and JCD. Correlation coefficient is defined as for two random variables X and Y (Montgomery and Runger 2003):

$$\rho = \frac{\mathrm{cov}(X, y)}{\sqrt{V(X)V(Y)}} = \frac{\sigma_{XY}}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}$$

where cov(X,Y), also denoted as σ_{XY}, is covariance of X and Y, μ_X is the mean value of X, μ_Y is the mean value of Y, V(X), V(Y) are variance of X and Y, which are denoted as σ_X^2 and σ_Y^2 and defined as

$$\sigma_X^2 = V(X) = E(X-\mu)^2$$

$$\sigma_Y^2 = V(Y) = E(Y-\mu)^2$$

Using the data provided in Appendix 7, the correlation coefficient between JCD and precision improvement is 0.695, an indicator that the two variables are somewhat related.

Based on the above analysis, it is suggested that JCD may be used as a criterion to measure the clarity of the expression of information need and its corresponding search-term submitted to an information retrieval system, such as a search engine when human judges are employed to carry out relevance judgments. If the JCD of the search results of a search-term is lower than 5%, it indicates that judges have a very low agreement on their relevance judgments. Lower JCD may be caused by the expression of the information need is not clear enough to

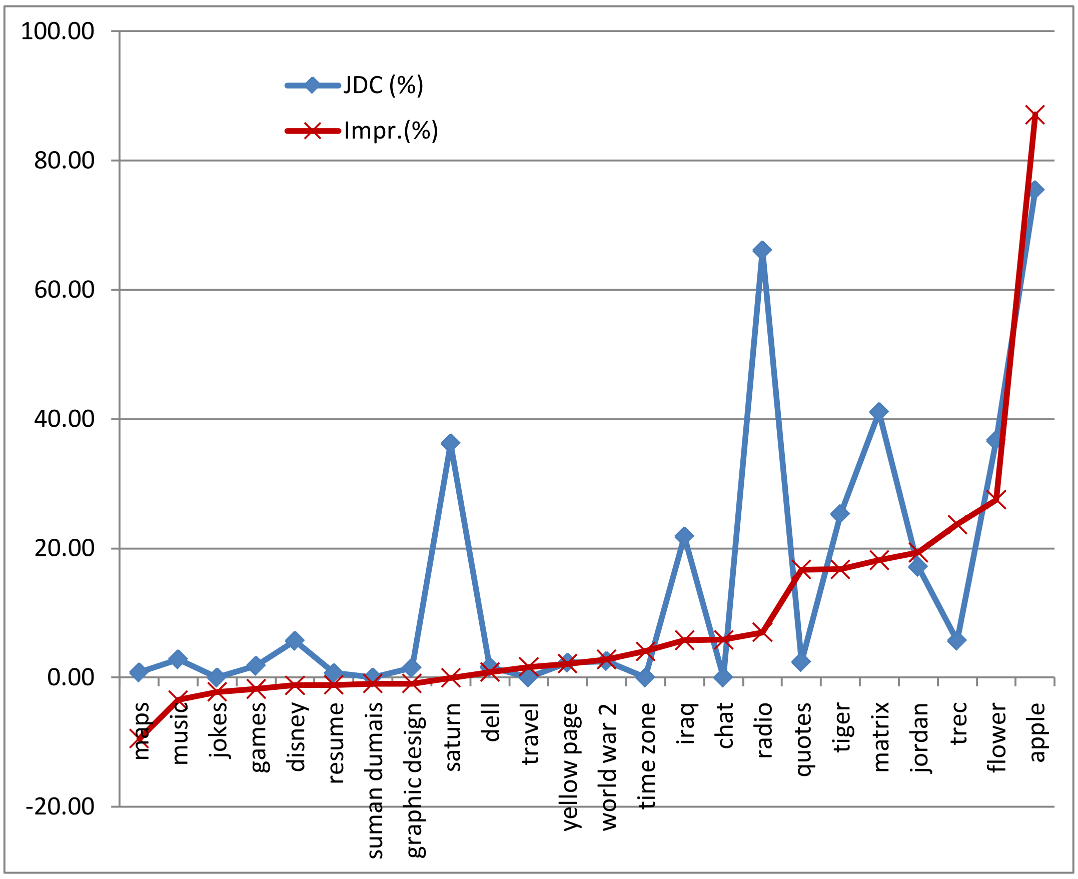


Figure 7-8 Relationship between precision improvement of RIB over Yahoo and relevance Judgment Convergence Degree (JCD)

make a correct judgment, or the Web snippets are too less-informative to make a correct judgment, or the judgment of some judges are suspicious. Therefore, the search results of this search-term should be excluded from the experiments that evaluate the performance of an information retrieval system when human relevance judgment is taken as the gold standard for performance evaluation.

7.6 Summary

This chapter presented the evaluation results of Yahoo and the recommended results of RIB either based on an automatically created user profile, which extracts user search preferences by analyzing information stored in the user's personal computer; or based on re-ranked categorized results of RIB. Two case studies showed that using automatically created user search preference profile, RIB could boost the precision of search results from 15.25% of Yahoo to 35.24% of RIB by using T50O test data set, and from 20.08% of Yahoo to 30.28% of RIB by using T50G test data set; the improvements are 19.99% and 9.2% respectively. Then, RIB was evaluated by using 25 search-terms to obtain 50 results for each of them, and for each of the 25×50 = 1250 returned results, relevance was judged by five judges and final relevance judgment decision was made by summing the decisions of the five judges. Precision-recall curves of Yahoo and RIB demonstrated that RIB outperforms Yahoo with a maximum precision improvement of 12.06% at recall level 30%, and a minimum precision improvement of 5.18% at recall level 100%. The averaged improvement of precision over the 25 search-terms was 8.74%, from 55.55% of Yahoo to 64.29% of RIB.

The ultimate objective of this research is to improve the relevance of Web search results, experimental data in this chapter clearly demonstrate that the objective of this project is achieved satisfactorily. In next chapter, future work is to be discussed and conclusions are to be drawn.

8 Conclusion and Future Work

Centred on the four objectives of this research project, experiment results related to the achievement of the objectives are provided in the precious four chapters. This chapter is to summarize the findings of this study in Section 8.1 based on the experimental data of the previous chapters; Section 8.2 is to present several possible future research directions, and finally, Section 8.3 summarizes this chapter.

8.1 Summary of the Findings of this Study

8.1.1 RIB can on Average Deliver More Relevance Web Hits

The ultimate objective of this study (**RO1**) as discussed in Chapter 3 is to improve the relevance of Web search results by means of text categorization, clustering and personalization to address the issue of low quality of Web search results stemmed from polysemous and synonymous characteristics of natural languages and search engines "one size fits all" strategy.

Personalized search results are produced either by using automatically created user profile or allowing user to select interest topics. Experimental results based on two case studies reveal that when using user profile to personalize Web search results, the averaged improvement of precision of RIB over Yahoo is 13.5%. If personalized results are obtained according to the selection of two interest topics which contains most relevant results, the averaged improvement of precision of RIB over Yahoo is 8.75%, from 53.92% of Yahoo to 62.67% of RIB, based on 25 search-terms which generate 1250 returned Web snippets and the relevance of the Web snippets are judged by five human judges. In the above experiments, RIB utilized a novel *BoostingUp* algorithm to categorize the Web snippets into the 14 top-level ODP categories. Text classifier employed by *BoostingUp* is Naïve Bayes which is trained by the fourth level ODP *categoryDocument* set with 100 features selected by Relevancy Score feature selection algorithm; the clustering algorithm used by *BoostingUp* is K-Means with MaxMin centroids initialization strategy. Based on the experimental results described above, the first conclusion is reached that

The developed Recommender Intelligence Browser (RIB) in this research, which combines Web snippets categorization, clustering and personalization, can on average improve the relevance of Web search results by 8.75%, from 53.92% of Yahoo to 62.67 of RIB, in terms of mean averaged precision (map).

8.1.2 *BoostingUp* can Marginally Improve the Performance of both Text Categorization and Clustering

The second objective of this research (**RO2**) is to improve the performance of text categorization by a novel *BoostingUp* algorithm. *BoostingUp* leverages the ability of clustering algorithms which groups a set of test documents by comparing their intra-similarity, and the power of text categorization which groups the same set of documents by comparing the inter similarity between the documents in the test set and the documents in a pre-labeled training document set. Using a set of 100 labeled Web snippets obtained from Google Directory, *BoostingUp* can improve the performance of Naïve Bayes in term of micro-averaged F_1 from 32.17% to 34.21%, the increase is 2.04%; and in terms of macro-averaged F_1, from 24.51% to 25. 47%, the improvement is 0.96%.

At the same time, the performance of clustering algorithm, here is K-Means, is also enhanced in terms of Adjusted Rand Index from 13.17% to 15.52%, the improvement is 2.35%;and in terms of F_1, from 21.45% to 23.82%, the increment is 2.37%. Based on the experimental data, the second conclusion is

BoostingUp algorithm can marginally improve the performance of both text categorization and clustering algorithms, here the Naïve Bayes and K-Means, in terms of F_1 and Adjusted Rand Index.

8.1.3 ODP Data can Produce Labeled Data Sets to Train a Classifier to Categorize Web Search Results

The third and fourth objectives of this study (**RO3** and **RO4**) involve three steps, the first is to extract semantic characteristics of ODP categories, and use the extracted semantic characteristics to generate a series of labeled data sets. The second step is to utilize the labelled data sets to evaluate text categorization/clustering algorithms such as Naïve Bayes, kNN, Adaboost, K-Means, HAC when combined with a variety of feature selection algorithms including Chi-square, Information Gain, Mutual Information, Relevancy Score, Odds Ratio, GSS Coefficient, and NGL coefficient as presented in Chapter 2 and Appendix 5. The third step is to use a text classifier such as Naïve Bayes to predict an ODP category for each returned Web snippets.

In this research, a series of labeled ODP *categoryDocument* sets are generated as described in Chapter 4. Text categorization algorithms are evaluated by using the generated labeled *categoryDocument* sets and experimental results are presented in Chapter 5 (**RO3**). Experimental data in Chapter 5 demonstrate that the performance of text categorization algorithms are influenced dramatically by feature selection algorithms and the number of features selected. The best performance in terms of F_1 is produced by Naïve Bayes when features are selected by Odds Ratio, this experimental result is in accordance with that presented by Mladenic and Grobelnik (2003).

When a text classifier trained by ODP *categoryDocuments* is utilized to predict labels for Web snippets, the best performance, 71.6% in terms of F_1 comes from kNN with Relevancy Score when 5000 features are selected, and for Naïve Bayes, the best performance in terms of F_1 is about 50% with GSS Coefficient is used to select 300 features. The results are satisfactory when considering that one Web snippet is assigned only one category and Web snippets are usually less informative than full-length text documents.

At the same time, the detailed structural information of ODP – the world's most comprehensive human edited Web directory, is also uncovered (**RO4**).

According to the experimental data provided in Chapter 4 and Chapter 5, the following conclusion can be drawn,

Semantic characteristics of ODP categories can be extracted and used to generate a series of labeled data sets. The generated labeled ODP categoryDocument sets enriched the collection of the existing labeled data sets that are served as benchmark document collection for the purpose of evaluation of text categorization algorithms. Specifically, the generated ODP categoryDocument sets can also be used to train a classifier to categorize Web snippets into the World's most comprehensive Web directory, the Open Directory Project. Meanwhile, the detailed structural information of ODP is uncovered as well.

8.1.4 The R^2Cut Thresholding Strategy is Able to Slightly Improve Text Categorization

Experimental data provided in Chapter 5 illustrate that the R^2Cut thresholding strategy proposed in this research is able to improve the performance of text classifier by more than 1% in terms of F_1 when parameter ρ is carefully adjusted. However, the performance of R^2Cut is sensitive to text categorization algorithms, feature selection algorithms, number of features, and the parameter ρ as well.

8.1.5 Z-tfidf Term-weighting Approach can Produce at Least as Good Results as tf-idf for kNN

Compared with the popular tf-idf term-weighting strategy (Salton and Buckley 1988), the suggested Z-tfidf term-weighting algorithm is able to improve the performance of kNN in terms of F_1 from 82.76% to 83.90% based on an experiment using a small text data set, S-Set; an averaged 1.14% F_1 improvement is achieved.

8.1.6 Relevance Judgment Convergence Degree (JCD) is in some way Related to the Precision Improvement of RIB

Last but not least, the study also discovers that there is a relationship between the improvement of precision of RIB and the relevance judgment results over 1250 Web snippets of 25 search-terms from 25 human judges; the correlation coefficient between the JCD and precision improvement is 0.695, an indication that precision improvement and JCD are somewhat related. Based on this observation, it is suggested JCD can be used as a criterion to evaluate to what degree the human judges agree with each other, and whether the results of a search-term are suitable to be employed as a text data set for the purpose of evaluating the effectiveness of an information retrieval system.

8.2 Future Research Directions

Recommended results and categorized results of RIB show significant improvement of relevance, and thus can provide a better search experience for Web users. Still, there is always room for further improvements. The following are several possible future research directions.

8.2.1 Re-organizing Web Snippets based on Resource Space Model (RSM)

Instead of categorizing Web snippets into the 14 top-level ODP categories, another promising method is to re-organize the snippets based on Resource Space Model (RSM) which can retrieve and organize heterogeneous sources of huge volumes of information (Zhuge, Xing, and Shi 2008). RSM is able to alleviate the subjective characteristic of text categorization and provide users with interactive semantic imagines, and finally to achieve the goal of a “semantic lens” to zoom “with diverse semantic link networks and multi-dimensional classification space through time” (Zhuge 2010). The main concern of this research may involve how to collect sufficient meta data to support the RSM to enable it to accomplish semantic zooming if the data sources are to extend from structured databases to the whole content of the Web.

8.2.2 Further Verification of the Findings by Using Different Labeled Data Sets

The findings presented in the previous section need to be further verified by employing not only the generated ODP *categoryDocument* sets, but also some of the most popularly employed labeled data sets such as RCV1 (Lewis et al. 2004) and OHSUMED. For instance, the effectiveness of *boostingUp* algorithm is to be examined by using RCV1 and compared with results of other text categorization algorithms using the same data set.

Some experiments (experiments in Chapter 5) used only the second level ODP *categoryDocuments*, while others (experiments in Chapter 6 and Chapter 7) used the fourth level *categoryDocument* set due to the limitation of timeframe of this study. These experiments could be carried out on different *categoryDocument* sets in the future.

The *boostingUp* algorithm is evaluated by combining Naïve Bayes and K-Means only in this study. The effectiveness of *boostingUp* when combining other more complex text categorization and clustering algorithms such as SVMs plus EM needs to be clarified as well.

8.2.3 Comparing the Results of Web Snippets Categorization and Clustering

One drawback of using a predefined set of categories is that the predefined topics are static and cannot reflex the dynamic changing of the topics in the Web. Web snippet clustering which discovers patterns within a collection of Web snippets is an effective way to approach this issue. Categorized results and clustered results can be compared by means of precision-recall curve and mean average precision to check whether clustering is able to provide more relevant results than categorizing.

8.2.4 Improvement of *BoostingUp*

Experimental results demonstrate that the proposed *boostingUp* is able to slightly improve the performance of both categorization and clustering. Nevertheless, a more effective *boostingUp* algorithm can be expected if the algorithm is improved, and the limitations and conditions of the usage of the algorithm are to be clarified via large-scale experiments with different data sets and diversity combinations of text categorization and clustering algorithms.

8.2.5 The Evolution of ODP

Despite this being the first research that uncovers the hierarchy of the ODP in terms of numbers of categories at the different levels of the ODP taxonomy tree, how the world's most comprehensive Web directory evolves and grows still have not seen by publications. A study aims at discovering the evolution of ODP in terms of the growth of its hierarchy, the number of categories at the different levels of the hierarchy, and the volume of the meta data provided by ODP will to some extent help to understand the evolution of the Web at the same period of time.

8.3 Summary

Semantic characteristics of the ODP categories were extracted to generate a series sets of *categoryDocuments*, which not only enrich the existing labeled document collections served as benchmark collections to evaluate text categorization and clustering algorithms, but also can be employed to train a classifier to assign a Web snippet an ODP category, and thus group Web search results under the different ODP categories. Recommended personalized results of RIB based on Web snippets categorization are able to significantly improve the precision compared with the un-categorized, un-personalized results directly obtained by using *Yahoo! Search Web Services API*.

The proposed *boostingUp* algorithm, which leverages the power of categorization and clustering, is shown to improve the performance of both categorization and clustering marginally in terms of F_1 and Adjusted Rand Index.

Another two novel algorithms suggested in this research, R^2Cut and Z-tfidf, can also slightly improve the performance of some kinds of text categorization algorithms in terms of F_1.

The study also discovered that there is a close relationship between JCD and the precision improvement of RIB, and the correlation coefficient between the two variables clearly demonstrated the existence of the relationship. JCD is suggested as a metric to estimate to what degree human judges agree with each other.

Future research directions include large-scale experiments using different labeled data sets and other existing benchmark document collection to further verify the findings in this research, re-organizing Web snippets based on Resource Space Model, improving the suggested *boostingUp*, *R^2Cut* and Z-tfidf algorithms, and the evolution of ODP.

Appendix 1 A Small Data Set – S-Set

The example document set includes 25 documents, 77 words, and the words count is 169. The statistical information about the documents is presented in the following table 1.

A: arts_design_c1ssub0, design interactive design design gallery design decorative art design 20th century;

B: arts_design_c1ssub1, collaborative art design decorative design art;

C: arts_design_c1ssub2, model information art design information 20th century design art;

D: arts_design_c1ssub3, directory select chic fashion art luxury shop work site;

E: arts_design_c1ssub4, diploma fashion design field business;

F: home_gardening_c2ssub0, garden plant vegetable plant variety garden topic;

G: home_gardening_c2ssub1, collection garden plant gallery garden news;

H: home_gardening_c2ssub2, general garden garden information garden garden topic vegetable plant;

I: home_gardening_c2ssub3, garden garden garden plant variety;

J: home_gardening_c2ssub4, class model algebra vegetable plant science;

K: home_gardening_c2ssub5, wall model fresh curtain art science;

L: home_gardening_c2ssub6, public private garden open visit;

M: home_gardening_c2ssub7, american garden education enjoy overview project;

N: home_gardening_c2ssub8, information public private community garden;

O: home_gardening_c2ssub9, plant encyclopedia feature photo short description;

P: science_math_c3ssub0, collection theory mathematics general abstract algebra student abstract algebra class;

Q: science_math_c3ssub1, topic theory science variety algebra linear algebra;

R: science_math_c3ssub2, universal algebra universal algebra mathematics algebra model theory variety;

S: science_math_c3ssub3, algebra class mathematics algebra variety;

T: science_math_c3ssub4, algebra mathematics mathematics algebra information algebra linear;

U: science_math_c3ssub5, add triangle plant algebra probability bayes;

V: science_math_c3ssub6, number theory scimath faq list;

W: science_math_c3ssub7, give integer sequence find name formula;

X: science_math_c3ssub8, online course 3d vector material difficult level;

Y: science_math_c3ssub9, course class public encyclopedia probability bayes;

Appendix Table 1-1 Statistical information of S-Set

terms	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Σ**
20th	1		1																							2
3d																								1		1
Abstract																2										2
Add																					1					1
Algebra										1						2	2	3	2	3	1					14
American													1													1
Art	1	2	2	1							1															7
Bayes																					1				1	2
Business					1																					1
Century	1		1																							2
Chic				1																						1
Class										1						1			1						1	4
Collaborative		1																								1
Collection							1									1										2
Community														1												1
Course																								1	1	2
Curtain											1															1
Decorative	1	1																								2
Description															1											1
Design	5	2	2		1																					10
Difficult																								1		1
Diploma					1																					1
Directory				1																						1
Education													1													1
Enjoy													1													1
Encyclopedia															1										1	2
Faq																						1				1
Fashion				1	1																					2
Feature															1											1
Field					1																					1
Find																							1			1
Formula																							1			1
Fresh											1															1
Gallery	1						1																			2
Garden						2	2	4	3			1	1	1												14
General								1								1										2
Give																							1			1
Information			2					1						1						1						5
Integer																							1			1
Interactive	1																									1
Level																								1		1
Linear																	1			1						2
List																						1				1
Luxury				1																						1
Material																								1		1
Mathematics																1		1	1	2						5
Model			1							1	1							1								4
Name																							1			1
News							1																			1
Number																						1				1
Online																								1		1
Open												1														1
Overview													1													1
Photo															1											1
Plant						2	1	1	1	1					1						1					8
Private												1		1												2
Probability																					1				1	2
Project													1													1
Public												1		1											1	3
Science										1	1						1									3
Scimath																						1				1
Select				1																						1
Sequence																							1			1
Shop				1																						1
Short															1											1
Site				1																						1
Student																1										1
Theory																1	1	1				1				4
Topic						1		1									1									3
Triangle																					1					1
Universal																		2								2
Variety						1			1								1	1	1							5
Vector																								1		1
Vegetable						1		1		1																3
Visit												1														1
Wall											1															1
Work				1																						1
Σ	11	6	9	9	5	7	6	9	5	6	6	5	6	5	6	10	7	9	5	7	6	5	6	7	6	169
terms	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	

Appendix 2 Web Snippets of Google Directory and ODP

A. Fifty Web Snippets of Google Directory when “jaguar” is the Search-term – T50G

No	Web snippet
1	Livingston Weekly Category: News > Newspapers > Regional > United States > Montana Weekly news for Livingston and Park County. Offers articles about local history, politics, and artists.
2	angelaid.com Blog Category: Society > Issues > ... > Service Organizations > Local A foundation for children with life-threatening diseases or situations (homeless, abused, etc.). Serving tri-county area around Jacksonville, Florida.
3	http://FirstPersonShooters.net Category: Games > Video Games > Shooter An encyclopedia of the genre, with information on games over various platforms, including walkthroughs, screenshots, and reviews.
4	Cosmic Groove : funk, soul, rare groove, jazz, latin , vinyl - Cd ... Category: Shopping > Entertainment > Recordings > Audio > Music > Specialty French record shop specialized in soul, funk, jazz, lounge, exotica, Latin, blaxploitation, and Afrobeat.
5	ACC Club Directory Collecting Club Directory, Featuring 6145 Groups! Category: Recreation > Collecting > Organizations Searchable database of collecting organizations around the world.
6	The Everett Interpretation Category: Science > Physics > Quantum Mechanics > Interpretations A set of frequently asked questions on Everett's many worlds approach to quantum mechanics.
7	DICTIONARIES Maya Category: Science > Social Sciences > ... > Languages > Natural > Mayan > Itzaj Online vocabulary of the language. PDF format.
8	MAMMFAUN, by Charles H. Smith Category: Science > Biology > ... > Animalia > Chordata > Mammalia A bibliography of publications concerning the geographical distribution of mammals.
9	Wizbang Sports Category: Sports > Resources > News and Media Includes news and commentary on football, basketball, hockey, and baseball. Features separate sections for NCAA sports.
10	Welcome - HOL Amiga database Category: Games > Video Games > Computer Platforms > Amiga Title database.
11	Hočąk Encyclopedia — Table of Contents Category: Arts > Literature > Myths and Folktales > Myths > Native American Articles, stories, and histories, edited and compiled by Richard L. Dieterle, with genealogies, bibliography, and links.
12	Used Macintosh - Apple Macintosh computers Category: Computers > Hardware > Retailers > Macintosh > Used Apple Macintosh computer supplies and accessories. Custom configured and upgraded Mac systems.
13	MyAviation.net - Aviation Photo Gallery Category: Recreation > Aviation > Multimedia > Photography Database hosting photos submitted from aviation photographers for free.
14	Scuba Diving Spots - Scuba Diving Directory! Category: Recreation > Outdoors > Scuba Diving > Guides and Directories A thorough directory of scuba resources from around the Internet.
15	UCABLES - Mobile phone unlocking cables, cellphone accessories ... Category: Shopping > Consumer Electronics > ... > Accessories > U Provides unlock and repair solutions for mobile phones. Provides accessories for Nokia, Sony, Ericsson, Motorola and Siemens.
16	CarGurus.com: reviews, specifications, photos, videos, prices and ... Category: Shopping > Vehicles > Directories Broad spectrum automotive portal.
17	Amusement Today Category: Business > Arts and Entertainment > ... > News and Media Monthly magazine serving the amusement industry, includes park attendance trends.
18	Rainforest Facts Category: Kids and Teens > School Time > ... > Biodiversity > Deforestation Explains the problems and solutions of rainforest deforestation. Included are facts and figures, which describe the important uses of this ecosystem and **...**
19	The Housing Bubble Blog Category: Home > Personal Finance > Money Management > Loans > Home Author Ben Jones examines the home price boom and its effect on owners, lenders, regulators, realtors and the economy as a whole typically cover a **...**
20	Self-Organizing Systems FAQ for Usenet newsgroup comp.theory.self **...** Category: Science > Math > Applications > Information Theory Includes a very extensive catalog of links on SOS.
21	Donna Haraway The Promises of Monsters Category: Society > Philosophy > Philosophers > H > Haraway, Donna > Works An essay by Haraway, first published in 1992. Considers the nature of "nature" in various contexts.
22	Museum Management Bibliography (NPS) Category: Reference > Bibliography Features links to books, manuals, technical leaflets and articles. From the National Park Service.
23	Famous Catholics Category: Arts > Performing Arts > ... > Articles and Interviews Actor Gerard Butler appears on a list of famous Catholics.

24	Epi- Olmec Hieroglyphic Writing and Texts Category: Science > Social Sciences > ... > Mesoamerican > Epigraphy > Olmec Paper by Terrence Kaufman and John Justeson.
25	acronyms finder dictionary and abreviations finder dictionary **...** Category: Reference > Dictionaries > Acronyms Contains terms with origins in the military, medical, technological, and business and training fields.
26	Flood Stories from Around the World Category: Arts > Literature > Myths and Folktales > Myths Brief description of flood myths from cultures all over the globe.
27	Blogs: Environment, climate change and green living blogs **...** Category: Science > Environment > Climate Change > News and Media Includes news articles covering climate change issues
28	ii.com · All About PINE: POP, IMAP, NNTP, & ESMPT Client for Unix **...** Category: Computers > Software > Internet > Clients > Mail > Pine Information about configuring Pine. Includes a list of links to resources about Pine.
29	Domestic Abuse and Sexual Assault related songs - Compiled by **...** Category: Society > People > ... > Violence and Abuse > Domestic Violence A listing of more than 100 songs in every genre, from traditional ballads to punk, many with links to online lyrics.
30	Brainwashed - Home Category: Arts > Music > ... > E > Electronica > IDM > Magazines and E-zines Weekly webzine with reviews, MP3s, and a list of new releases.
31	Don Markstein's Toonopedia Category: Arts > Animation > Cartoons On-line hypertext encyclopedia of comics, animation and other forms of cartoonery. "A vast repository of toonological knowledge."
32	WWF Passport - Take action online for the environment Category: Society > Issues > Environment > Activism Provides tools to campaign on critical environment issues all over the world from climate change to construction projects in protected areas.
33	Welcome to CrossFit: Forging Elite Fitness Category: Sports > People > Training Strength and conditioning program for police, military, and athletes. Includes message board, exercise videos, seminars and certifications offered, **...**
34	The PowerPoint FAQ Category: Computers > Software > Presentation > Microsoft PowerPoint Frequently asked questions including techniques, how-to information, troubleshooting, and links to other PPT related sites.
35	Jim Breen's Japanese Page Category: Science > Social Sciences > ... > Natural > Japanese > Resources Home of the free EDICT Japanese-English dictionary. Also links to other sites about Japanese.
36	Linux4Chemistry - Linux software for chemistry: molecular modeling **...** Category: Computers > Software > ... > Linux > Projects > Scientific An up-to-date website with the links to Linux software for chemistry including computational, visualization, graphic,
37	A Dictionary of Scientific Quotations Category: Reference > Quotations > Scientific A collection of quotations about science by scientists and other writers both ancient and modern.
38	Seattle News, Events, Restaurants, Music Category: News > Newspapers > Regional > United States > Washington Local information for Seattle area. Classifieds, film and restaurant reviews, event listings, feature and arts articles, travel.
39	Airliners.net | Airplanes - Aviation - Aircraft- Aircraft Photos **...** Category: Recreation > Aviation > Multimedia > Photography Photo gallery containing mostly commercial airliners but also military planes and helicopters. Also hosts a discussion forum and an aircraft data and history **...**
40	Apple, Macintosh, iPod and iPhone news | MacNN Category: Computers > Systems > Apple > Macintosh > News and Media Offers news, reviews, discussion, tips, troubleshooting, links, and reviews.
41	IGN Cheats: Game Cheats & Codes Category: Kids and Teens > Games > Computer and Video > Cheats and Hints Offers a FAQ section, tips and tricks, and a buyers' guide on a variety of games.
42	Macintosh Security Site - Security for Mac Platform MacOS X **...** Category: Computers > Software > Operating Systems > Mac OS > Security A site dedicated to security-related issues and software on the Macintosh.
43	Dallas News, Events, Restaurants, Music Category: News > Newspapers > Regional > United States > Texas Alternative newspaper with news, blogs, music, movies restaurants, and the arts.
44	COLOMBIAN ROCK ART MOTIFS: SOME IDEAS FOR INTERPRETATION Category: Science > Social Sciences > ... > Regional > South America > Colombia This article posits that religious and political leaders probably painted or engraved rocks that are scattered throughout Colombia today.
45	TruthBook Religious News Blog Category: Society > Religion and Spirituality > Computers > Weblogs Polls and surveys. Articles about religion and belief.
46	Atari: Legends of the Fall - Vox Category: Games > Video Games > ... > Black Ice White Noise Audiocast about life working at Atari on the Black Ice White Noise project. Includes audio and text of audio. akelatalamasca.vox.com/library/post/atari-legends-of-the-fall.html
47	Open Source vs. Mac vs. Windows Category: Computers > Systems > Apple > Macintosh > Advocacy and Evangelism The fear of being locked in by a vendor has pushed some users to embrace all open-source solutions... But is it really an advantage ?
48	Timelapse Review and Walkthrough Category: Games > Video Games > ... Review, walkthrough, and puzzle solutions.
49	Used Cars for sale in Canada Category: Shopping > Classifieds > Automotive Automotive classifieds for Canada. Latest gas prices along with buying and selling tips.
50	Category: Business > Arts and Entertainment > Music > Promotion > N Publicity for many groups such as Beastie Boys, Radiohead, Foo Fighters, and Sonic Youth.

B. Fifty Web Snippets of ODP when “jaguar” is the Search-term – T50O

No	Web snippet
1	Jaguar - Official worldwide web site of Jaguar Cars. Directs users to pages tailored to country-specific markets and model-specific Websites. -- *http://www.jaguar.com/ Recreation: Autos: Makes and Models: Jaguar*
2	angelaid.com Blog Category: Society > Issues > ... > Service Organizations > Local A foundation for children with life-threatening diseases or situations (homeless, abused, etc.). Serving tri-county area around Jacksonville, Florida.
3	http://FirstPersonShooters.net Category: Games > Video Games > Shooter An encyclopedia of the genre, with information on games over various platforms, including walkthroughs, screenshots, and reviews.
4	Cosmic Groove : funk, soul, rare groove, jazz, latin , vinyl - Cd ... Category: Shopping > Entertainment > Recordings > Audio > Music > Specialty French record shop specialized in soul, funk, jazz, lounge, exotica, Latin, blaxploitation, and Afrobeat.
5	ACC Club Directory Collecting Club Directory, Featuring 6145 Groups! Category: Recreation > Collecting > Organizations Searchable database of collecting organizations around the world.
6	The Everett Interpretation Category: Science > Physics > Quantum Mechanics > Interpretations A set of frequently asked questions on Everett's many worlds approach to quantum mechanics.
7	DICTIONARIES Maya Category: Science > Social Sciences > ... > Languages > Natural > Mayan > Itzaj Online vocabulary of the language. PDF format.
8	MAMMFAUN, by Charles H. Smith Category: Science > Biology > ... > Animalia > Chordata > Mammalia A bibliography of publications concerning the geographical distribution of mammals.
9	Wizbang Sports Category: Sports > Resources > News and Media Includes news and commentary on football, basketball, hockey, and baseball. Features separate sections for NCAA sports.
10	Welcome - HOL Amiga database Category: Games > Video Games > Computer Platforms > Amiga Title database.
11	Hočąk Encyclopedia — Table of Contents Category: Arts > Literature > Myths and Folktales > Myths > Native American Articles, stories, and histories, e□ited and compiled by Richard L. Dieterle, with genealogies, bibliography, and links.
12	Used Macintosh - Apple Macintosh computers Category: Computers > Hardware > Retailers > Macintosh > Used Apple Macintosh computer supplies and accessories. Custom configured and upgraded Mac systems.
13	MyAviation.net - Aviation Photo Gallery Category: Recreation > Aviation > Multimedia > Photography Database hosting photos submitted from aviation photographers for free.
14	Scuba Diving Spots - Scuba Diving Directory! Category: Recreation > Outdoors > Scuba Diving > Guides and Directories A thorough directory of scuba resources from around the Internet.
15	UCABLES - Mobile phone unlocking cables, cellphone accessories ... Category: Shopping > Consumer Electronics > ... > Accessories > U Provides unlock and repair solutions for mobile phones. Provides accessories for Nokia, Sony, Ericsson, Motorola and Siemens.
16	CarGurus.com: reviews, specifications, photos, videos, prices and ... Category: Shopping > Vehicles > Directories Broad spectrum automotive portal.
17	Amusement Today Category: Business > Arts and Entertainment > ... > News and Media Monthly magazine serving the amusement industry, includes park attendance trends.
18	Rainforest Facts Category: Kids and Teens > School Time > ... > Biodiversity > Deforestation Explains the problems and solutions of rainforest deforestation. Included are facts and figures, which describe the important uses of this type of ecosystem and ...
19	The Housing Bubble Blog Category: Home > Personal Finance > Money Management > Loans > Home Author Ben Jones examines the home price boom and its effect on owners, lenders, regulators, realtors and the economy as a whole. Entries typically cover a ...
20	Self-Organizing Systems FAQ for Usenet newsgroup comp.theory.self ... Category: Science > Math > Applications > Information Theory Includes a very extensive catalog of links on SOS.
21	Donna Haraway_The Promises of Monsters Category: Society > Philosophy > Philosophers > H > Haraway, Donna > Works An essay by Haraway, first published in 1992. Considers the nature of "nature" in various contexts.
22	Museum Management Bibliography (NPS) Category: Reference > Bibliography Features links to books, manuals, technical leaflets and articles. From the National Park Service.
23	Famous Catholics Category: Arts > Performing Arts > ... > Articles and Interviews Actor Gerard Butler appears on a list of famous Catholics.
24	Epi- Olmec Hieroglyphic Writing and Texts Category: Science > Social Sciences > ... > Mesoamerican > Epigraphy > Olmec Paper by Terrence Kaufman and John Justeson.
25	acronyms finder dictionary and abreviations finder dictionary **...** Category: Reference > Dictionaries > Acronyms Contains terms with origins in the military, medical, technological, and business and training fields.

26	Flood Stories from Around the World Category: Arts > Literature > Myths and Folktales > Myths Brief description of flood myths from cultures all over the globe.
27	Blogs: Environment, climate change and green living blogs **...** Category: Science > Environment > Climate Change > News and Media Includes news articles covering climate change issues
28	ii.com · All About PINE: POP, IMAP, NNTP, & ESMPT Client for Unix **...** Category: Computers > Software > Internet > Clients > Mail > Pine Information about configuring Pine. Includes a list of links to resources about Pine.
29	Domestic Abuse and Sexual Assault related songs - Compiled by **...** Category: Society > People > ... > Violence and Abuse > Domestic Violence A listing of more than 100 songs in every genre, from traditional ballads to punk, many with links to online lyrics.
30	Brainwashed - Home Category: Arts > Music > ... > E > Electronica > IDM > Magazines and E-zines Weekly webzine with reviews, MP3s, and a list of new releases.
30	Don Markstein's Toonopedia Category: Arts > Animation > Cartoons On-line hypertext encyclopedia of comics, animation and other forms of cartoonery. "A vast repository of toonological knowledge."
32	WWF Passport - Take action online for the environment Category: Society > Issues > Environment > Activism Provides tools to campaign on critical environment issues all over the world from climate change to construction projects in protected areas.
33	Welcome to CrossFit: Forging Elite Fitness Category: Sports > People > Training Strength and conditioning program for police, military, and athletes. Includes message board, exercise videos, seminars and certifications offered, **...**
34	The PowerPoint FAQ Category: Computers > Software > Presentation > Microsoft PowerPoint Frequently asked questions including techniques, how-to information, troubleshooting, and links to other PPT related sites.
35	Jim Breen's Japanese Page Category: Science > Social Sciences > ... > Natural > Japanese > Resources Home of the free EDICT Japanese-English dictionary. Also links to other sites about Japanese.
36	Linux4Chemistry - Linux software for chemistry: molecular modeling **...** Category: Computers > Software > ... > Linux > Projects > Scientific An up-to-date website with the links to Linux software for chemistry including computational, visualization, graphic,
37	A Dictionary of Scientific Quotations Category: Reference > Quotations > Scientific A collection of quotations about science by scientists and other writers both ancient and modern.
38	Seattle News, Events, Restaurants, Music Category: News > Newspapers > Regional > United States > Washington Local information for Seattle area. Classifieds, film and restaurant reviews, event listings, feature and arts articles, travel.
39	Airliners.net | Airplanes - Aviation - Aircraft- Aircraft Photos **...** Category: Recreation > Aviation > Multimedia > Photography Photo gallery containing mostly commercial airliners but also military planes and helicopters. Also hosts a discussion forum and an aircraft data and history **...**
40	Apple, Macintosh, iPod and iPhone news | MacNN Category: Computers > Systems > Apple > Macintosh > News and Media Offers news, reviews, discussion, tips, troubleshooting, links, and reviews.
41	IGN Cheats: Game Cheats & Codes Category: Kids and Teens > Games > Computer and Video > Cheats and Hints Offers a FAQ section, tips and tricks, and a buyers' guide on a variety of games.
42	Macintosh Security Site - Security for Mac Platform MacOS X **...** Category: Computers > Software > Operating Systems > Mac OS > Security A site dedicated to security-related issues and software on the Macintosh.
43	Dallas News, Events, Restaurants, Music Category: News > Newspapers > Regional > United States > Texas Alternative newspaper with news, blogs, music, movies restaurants, and the arts.
44	COLOMBIAN ROCK ART MOTIFS: SOME IDEAS FOR INTERPRETATION Category: Science > Social Sciences > ... > Regional > South America > Colombia This article posits that religious and political leaders probably painted or engraved rocks that are scattered throughout Colombia today.
45	TruthBook Religious News Blog Category: Society > Religion and Spirituality > Computers > Weblogs Polls and surveys. Articles about religion and belief.
46	Atari: Legends of the Fall - Vox Category: Games > Video Games > ... > Black Ice White Noise Audiocast about life working at Atari on the Black Ice White Noise project. Includes audio and text of audio. akelatalamasca.vox.com/library/post/atari-legends-of-the-fall.html
47	Open Source vs. Mac vs. Windows Category: Computers > Systems > Apple > Macintosh > Advocacy and Evangelism The fear of being locked in by a vendor has pushed some users to embrace all open-source solutions... But is it really an advantage ?
48	Timelapse Review and Walkthrough Category: Games > Video Games > ... Review, walkthrough, and puzzle solutions.
49	Used Cars for sale in Canada Category: Shopping > Classifieds > Automotive Automotive classifieds for Canada. Latest gas prices along with buying and selling tips.
50	Category: Business > Arts and Entertainment > Music > Promotion > N Publicity for many groups such as Beastie Boys, Radiohead, Foo Fighters, and Sonic Youth.

C. First Page of Google Directory and ODP when “jaguar” is the Search-term

First Page of Google Directory when jaguar is the Search-term

1.1 Jaguar - Jaguar International

Category: Recreation > Autos > Makes and Models > Jaguar
Official worldwide web site of ***Jaguar*** Cars. Directs users to pages tailored to country-specific markets and model-specific websites.
www.jaguar.com/

1.2 Jaguar Facts - Panthera onca - Defenders of Wildlife - Defenders ...

Category: Kids and Teens > School Time > ... > Animals > Mammals > Jaguar
Includes color photographs and information about the size, appearance, life span, habitat, and diet of this endangered feline.
www.defenders.org/wildlife_and_habitat/wildlife/jaguar.php

1.3 The Official Website of the Jacksonville Jaguars - jaguars.com

Category: Sports > Football > American > NFL > Jacksonville Jaguars
The official team site with scores, news items, game schedule, and roster.
www.jaguars.com/

1.4 Jaguar

Category: Science > Biology > ... > Felidae > Panthera > Panthera onca
General information and facts from Big Cats Online.
dspace.dial.pipex.com/agarman/jaguar.htm

1.5 Jaguar Models - Main Page (resin model kits)

Category: Shopping > Recreation > Models
Plastic and resin military model kits from Japan, Europe, USA, and Asia.
www.jaguarmodels.com/

1.6 Jag-lovers - the most visited Jaguar enthusiast site on the Planet!

Category: Recreation > Autos > Makes and Models > Jaguar
Large and well-known resource. Includes an collection of brochures ranging from 1930's to present, mailing list, photo albums and other features.
www.jag-lovers.org/

1.7 Apple - Mac OS X Snow Leopard - The world's most advanced OS

Category: Computers > Software > ... > Mac OS > System Software > Mac OS X
The Apple Mac OS X product page. Describes features in the current version of Mac OS X, a screenshot gallery, latest software downloads, and a directory of...
www.apple.com/macosx/

1.8 Jaguar **Overview**

Category: Home > Consumer Information > ... > By Make > Jaguar > New
Detailed ***Jaguar*** pricing, reviews, comparisons, specifications and information.
www.edmunds.com/jaguar/index.html

1.9 Atari Jaguar - Wikipedia, the free encyclopedia

Category: Games > Video Games > Console Platforms > Atari > Jaguar 64
Offers summary information on the system's history and specifications.
en.wikipedia.org/wiki/Atari_Jaguar

1.10 Schrödinger -> Home

Category: Science > Chemistry > Software > Companies
Producer of the ***Jaguar*** quantum chemistry package and the MacroModel molecular mechanics package.
www.schrodinger.com/

First Page of Open Directory Project when jaguar is the Search-term

Jaguar - Official worldwide web site of **Jaguar** Cars. Directs users to pages tailored to country-specific markets and model-specific websites.
-- *http://www.**jaguar**.com/* *Recreation: Autos: Makes and Models: Jaguar* *(64)*

Wikipedia: Atari **Jaguar** - Offers summary information on the system's history and specifications.
-- *http://en.wikipedia.org/wiki/Atari_**Jaguar*** *Games: Video Games: Console Platforms: Atari: Jaguar 64* *(26)*

Jaguar Parts USA by Bergen **Jaguar** - **Jaguar** parts and accessories specialist. Located in New Jersey.
-- *http://www.jaguarpartsusa.com/* *Shopping: Vehicles: Parts and Accessories: Makes and Models: European: British: Jaguar* *(27)*

Jaguar (Panthera Onca) - Fact sheet describes this feline's appearance, habitat, reproduction, and social system. Includes photos.
-- *http://www.bigcatrescue.org/**jaguar**.htm* *Kids and Teens: School Time: Science: Living Things: Animals: Mammals: Jaguar* *(9)*

Jacksonville Jaguars Stats & History - Statistics, records, and history of the Jacksonville Jaguars team and players.
-- *http://www.jagsstats.com* *Sports: Football: American: NFL: Jacksonville Jaguars* *(5)*

Jaguar XJR-S - This car was the pre-production test vehicle. It is "one-of-one" and the site uniquely covers a historical footnote.
-- *http://www.jaguarxjr-s.com/* *Recreation: Autos: Makes and Models: Jaguar* *(64)*

MobyGames: **Jaguar** - Offers a searchable list of game titles. Includes overviews, publisher information, and cover scans.
-- *http://www.mobygames.com/browse/games/**jaguar**/* *Games: Video Games: Console Platforms: Atari: Jaguar 64* *(26)*

Classic **Jaguar** - **Jaguar** performance parts and restoration specialist based in Texas. Technical information, online secure catalog with photos.
-- *http://www.classicjaguar.com/* *Shopping: Vehicles: Parts and Accessories: Makes and Models: European: British: Jaguar* *(27)*

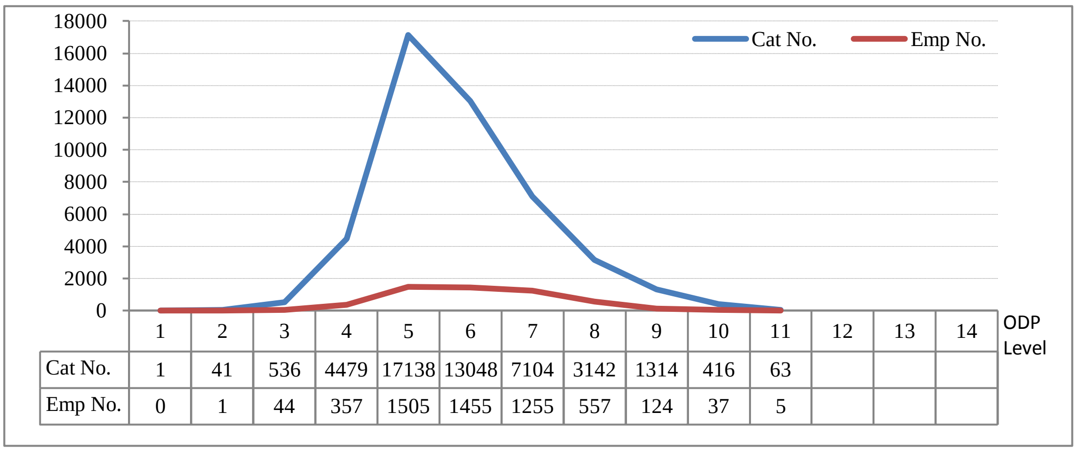
Jaguar - Includes information on their habitat, range, diet, size, reproduction, and life cycle.
-- *http://www.zoo.org/factsheets/**jaguar**/**jaguar**.html* *Kids and Teens: School Time: Science: Living Things: Animals: Mammals: Jaguar* *(9)*

Jacksonville Jaguars News - Jaguars news feeds from newspapers, sports sites and blogs.
-- http://www.mysportsscoop.com/jaguars/ Sports: Football: American: NFL: Jacksonville Jaguars: News and Media (3)

Retrieved 2010-01-29. Note that there are only two items are the same in the two first pages. Item 1 and 2 in ODP's first page, corresponding to the first and ninth item in Google Directory results.

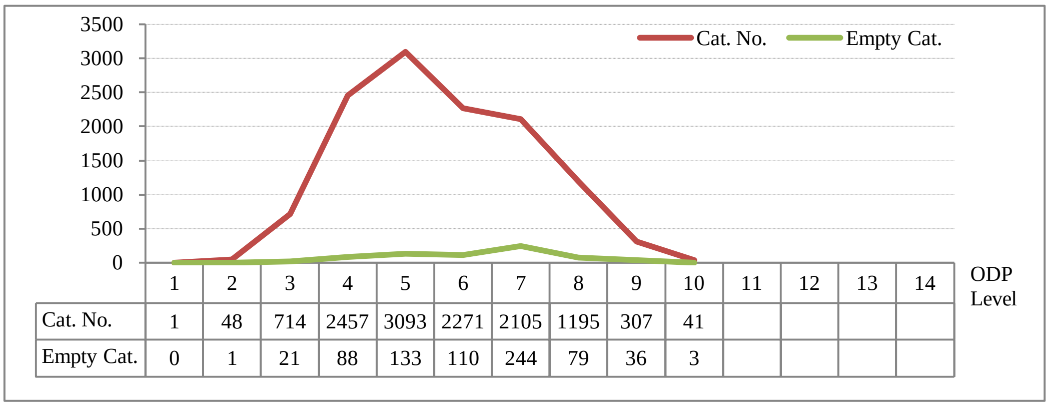
Appendix 3 Categories and Empty Categories of ODP

This appendix includes 17 figures which demonstrate the number of subcategories and empty categories for each of the 17 top-level ODP categories. Figure 4-5 in Chapter 4 summarizes the 17 figures in a single figure.



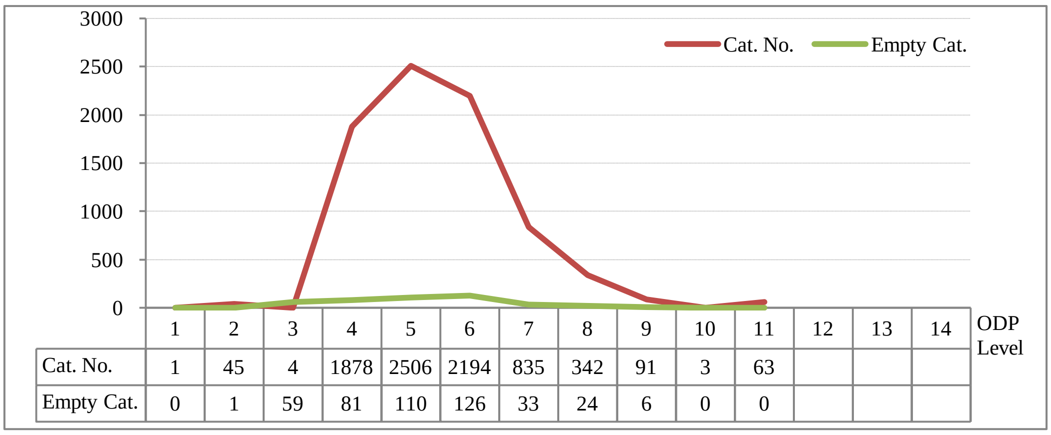
ODP Level	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Cat No.	1	41	536	4479	17138	13048	7104	3142	1314	416	63			
Emp No.	0	1	44	357	1505	1455	1255	557	124	37	5			

Appendix Figure 3-1 Category / empty category number of "Arts"



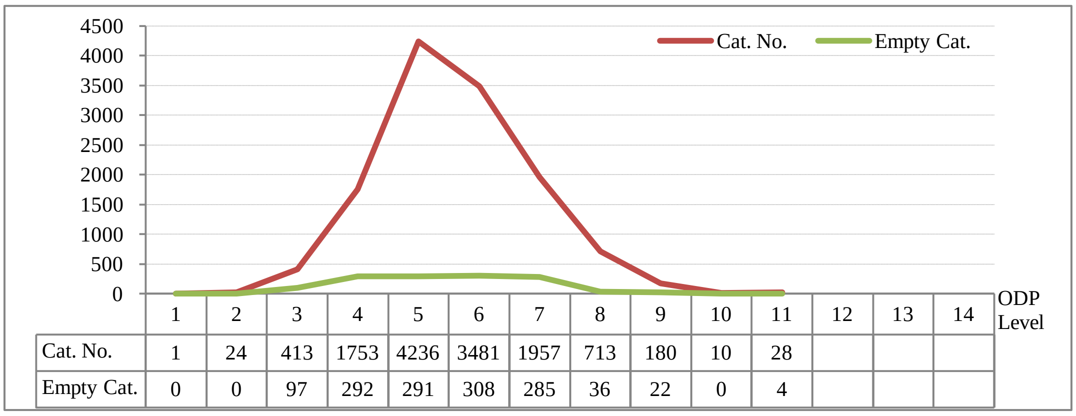
ODP Level	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Cat. No.	1	48	714	2457	3093	2271	2105	1195	307	41				
Empty Cat.	0	1	21	88	133	110	244	79	36	3				

Appendix Figure 3-2 Category / empty category number of "Business"



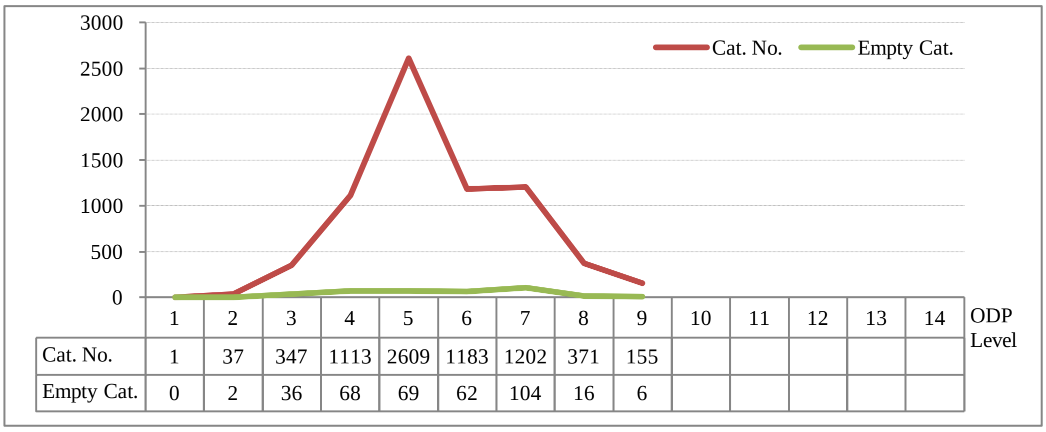
ODP Level	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Cat. No.	1	45	4	1878	2506	2194	835	342	91	3	63			
Empty Cat.	0	1	59	81	110	126	33	24	6	0	0			

Appendix Figure 3-3 Category / empty category number of “Computers”



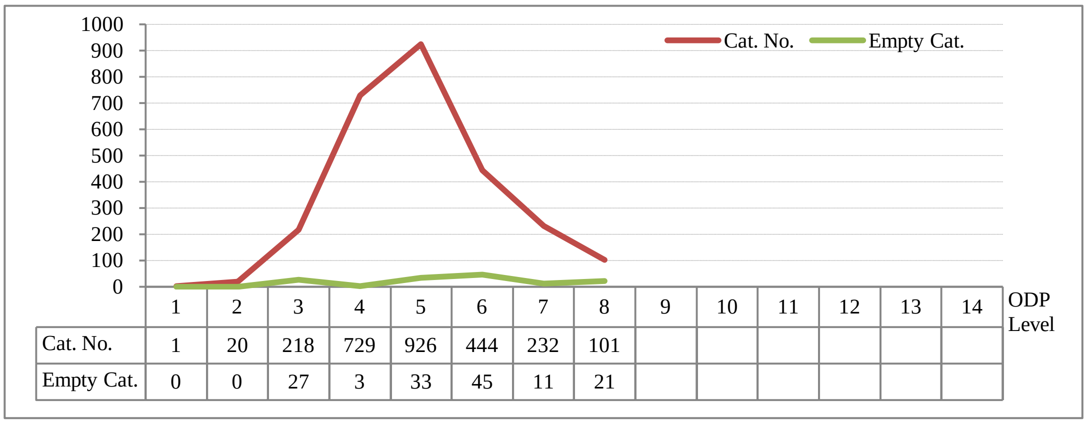
ODP Level	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Cat. No.	1	24	413	1753	4236	3481	1957	713	180	10	28			
Empty Cat.	0	0	97	292	291	308	285	36	22	0	4			

Appendix Figure 3-4 Category / empty category number of “Games”



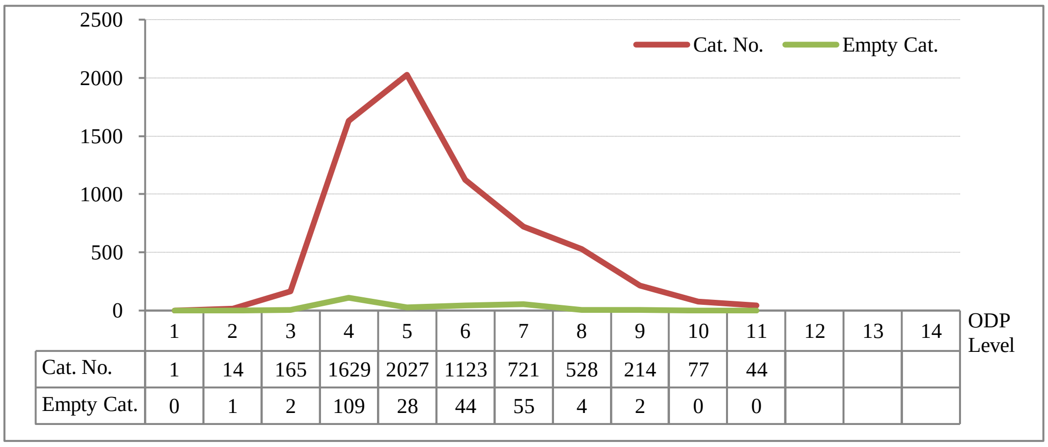
ODP Level	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Cat. No.	1	37	347	1113	2609	1183	1202	371	155					
Empty Cat.	0	2	36	68	69	62	104	16	6					

Appendix Figure 3-5 Category / empty category number of “Health”



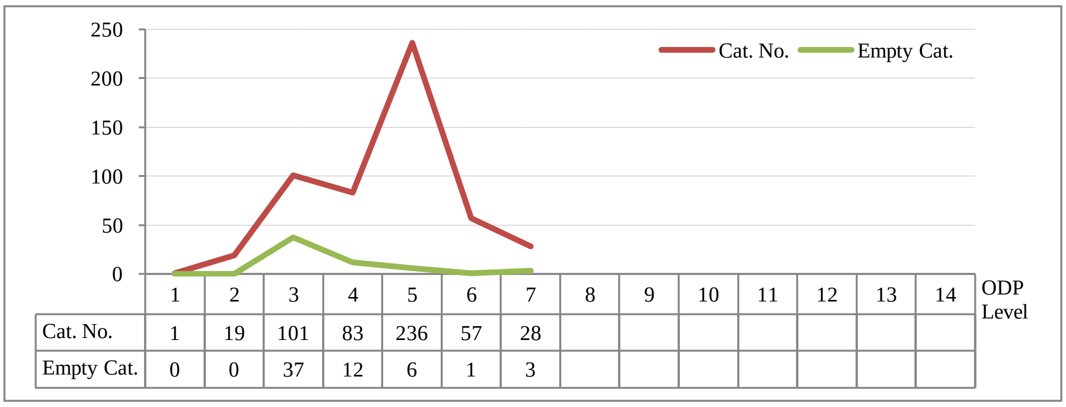
ODP Level	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Cat. No.	1	20	218	729	926	444	232	101						
Empty Cat.	0	0	27	3	33	45	11	21						

Appendix Figure 3-6 Category / empty category number of “Home”



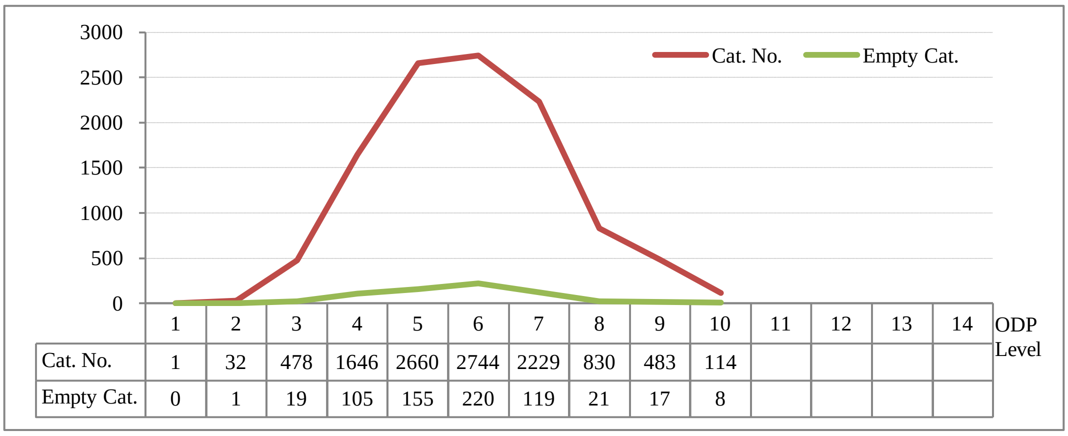
ODP Level	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Cat. No.	1	14	165	1629	2027	1123	721	528	214	77	44			
Empty Cat.	0	1	2	109	28	44	55	4	2	0	0			

Appendix Figure 3-7 Category / empty category number of category “Kids and Teens”



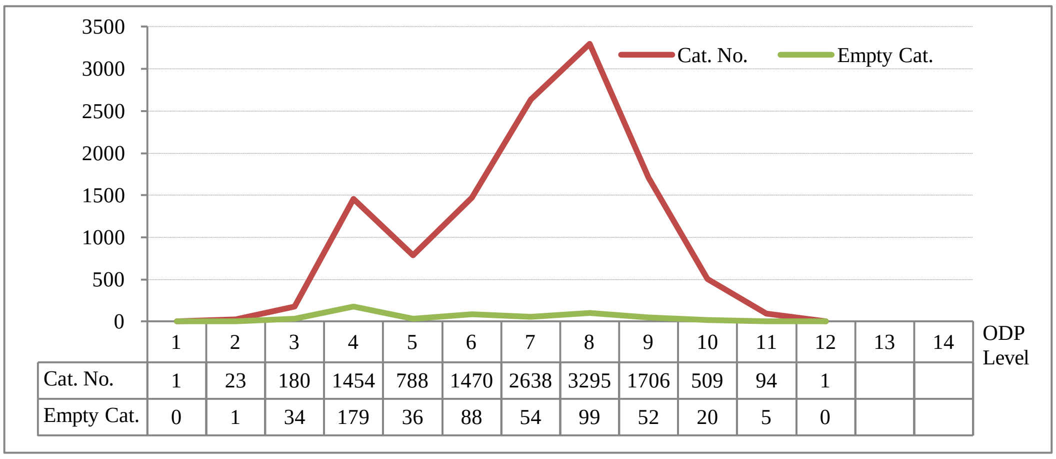
ODP Level	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Cat. No.	1	19	101	83	236	57	28							
Empty Cat.	0	0	37	12	6	1	3							

Appendix Figure 3-8 Category / empty category number of category “News”



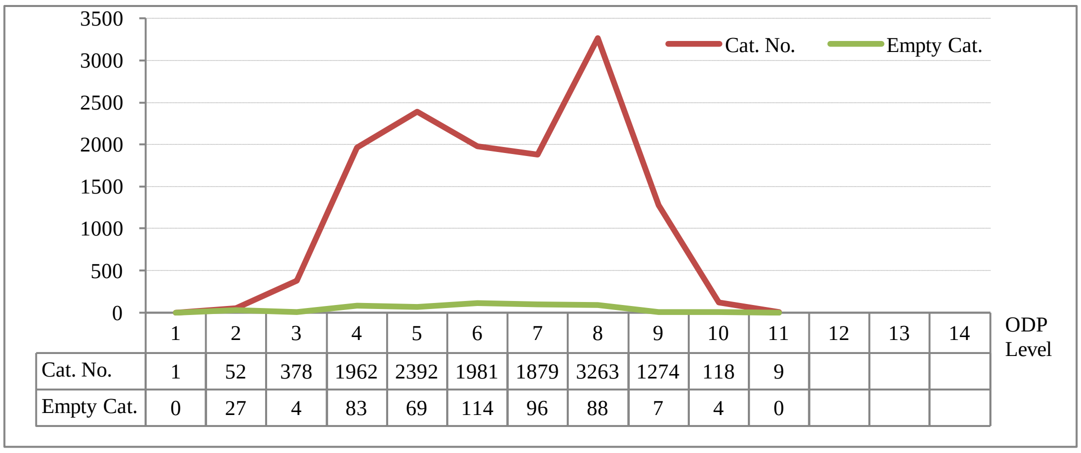
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Cat. No.	1	32	478	1646	2660	2744	2229	830	483	114				
Empty Cat.	0	1	19	105	155	220	119	21	17	8				

Appendix Figure 3-10 Category / empty category number of category "Recreation"



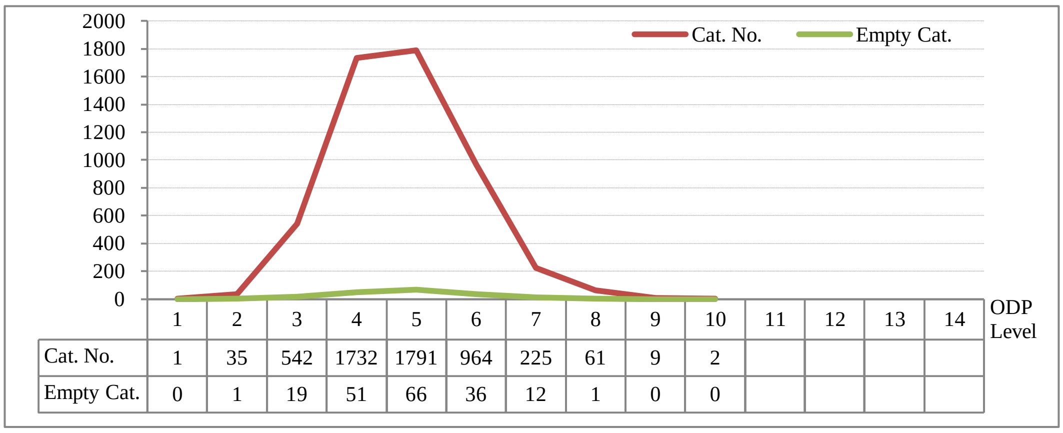
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Cat. No.	1	23	180	1454	788	1470	2638	3295	1706	509	94	1		
Empty Cat.	0	1	34	179	36	88	54	99	52	20	5	0		

Appendix Figure 3-11 Category / empty category number of category "Reference"



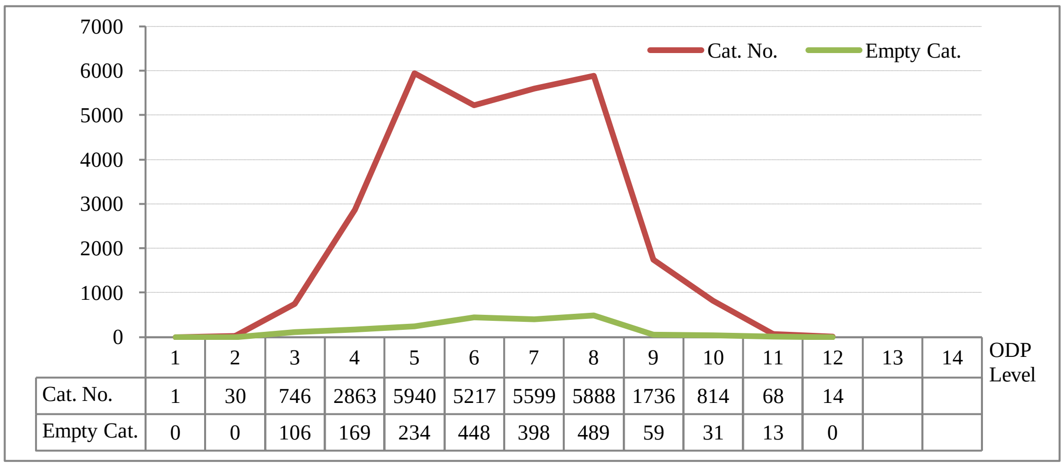
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Cat. No.	1	52	378	1962	2392	1981	1879	3263	1274	118	9			
Empty Cat.	0	27	4	83	69	114	96	88	7	4	0			

Appendix Figure 3-9 Category / empty category number of category "Science"



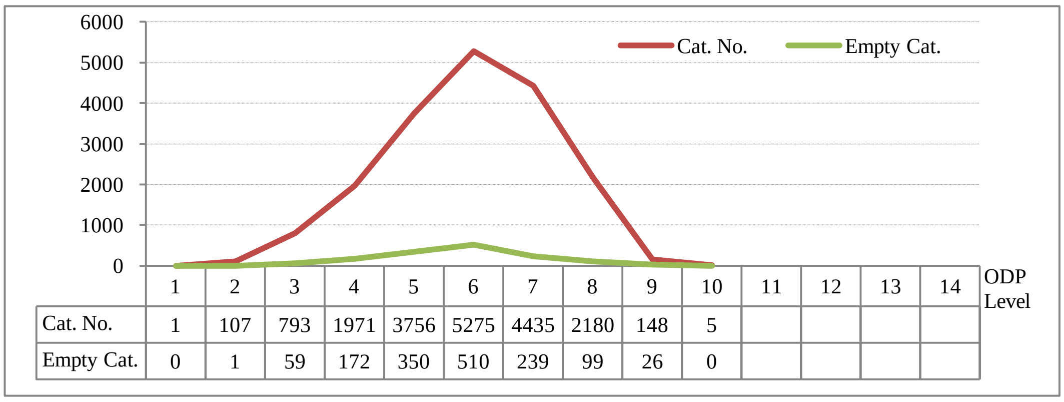
ODP Level	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Cat. No.	1	35	542	1732	1791	964	225	61	9	2				
Empty Cat.	0	1	19	51	66	36	12	1	0	0				

Appendix Figure 3-12 Category / empty category number of category “Shopping”



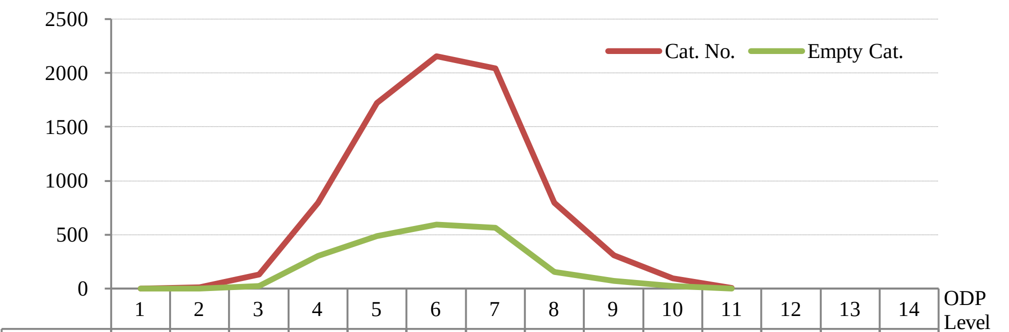
ODP Level	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Cat. No.	1	30	746	2863	5940	5217	5599	5888	1736	814	68	14		
Empty Cat.	0	0	106	169	234	448	398	489	59	31	13	0		

Appendix Figure 3-13 Category / empty category number of category “Society”



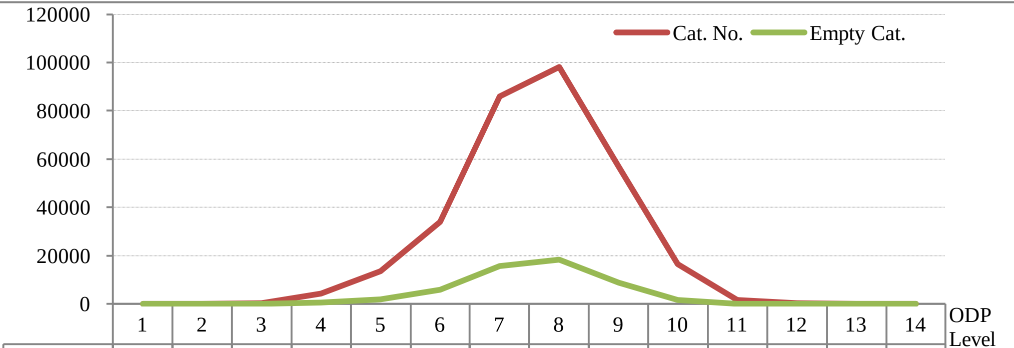
ODP Level	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Cat. No.	1	107	793	1971	3756	5275	4435	2180	148	5				
Empty Cat.	0	1	59	172	350	510	239	99	26	0				

Appendix Figure 3-14 Category / empty category number of category “Sports”



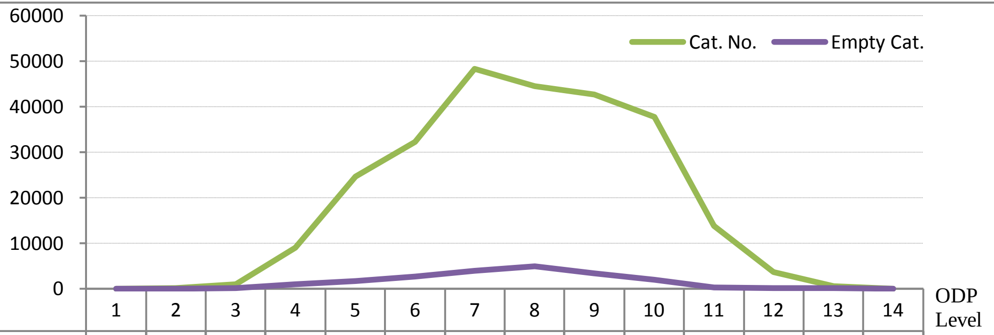
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Cat. No.	1	12	132	796	1722	2152	2044	793	309	98	6			
Empty Cat.	0	0	22	303	486	593	562	154	71	22	0			

Appendix Figure 3-17 Category / empty category number of category "Adult"



	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Cat. No.	1	36	393	4399	13514	33955	86032	98245	56816	16388	1738	181	11	3
Empty Cat.	0	26	43	522	1939	5735	15662	18288	8897	1573	71	4	0	0

Appendix Figure 3-16 Category / empty category number of category "Regional"



	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Cat. No.	1	81	1054	9002	24600	32288	48263	44513	42665	37670	13853	3709	637	76
Empty Cat.	0	8	129	1006	1719	2712	3978	4884	3374	1980	312	88	104	42

Appendix Figure 3-15 Category / empty category number of category "World"

Appendix 4 30 Ambiguous Search-terms and Information Needs

Appendix Table 3-1 30 search-terms and information needs (Zeng et al. 2004)

Search-term		Your information need
Ambiguous terms	jaguar	Animal jaguar
	apple	The fruit apple
	saturn	the planet Saturn
	jobs	the person Steve Jobs
	jordan	the Hashemite kingdom Jordan
	tiger	the animal tiger
	trec	Text REtrieval Conference
	ups	The Uninterrupted Power Supply
	quotes	how to correctly use quotes in writing
	matrix	the film matrix
Entity names	susan dumais	the researcher Susan Dumais
	clinton	the US ex-president, Bill Clinton
	iraq	geographic and demographical information about iraq
	dell	the dell computer company
	disney	Information about Disney empire
	world war 2	history related to world war 2
	ford	Henry Ford, the founder of the Ford Motor Company
General terms	health	how to keep healthy
	yellow pages	the origin of yellow pages
	maps	how to read maps
	flower	wild flower
	music	music classification by Genre
	chat	computer-mediated chat systems
	games	history of games
	radio	history of radio
	jokes	the most funny jokes
	graphic design	the art and practice of graphical design
	resume	how to write a resume
	time zones	time zones of the world
	travel	travel planning and preparation

In this research, 25 search-terms from the table below are employed. Five of them, jaguar, ups, clinton, ford, and health have been studied and thus are excluded.

Appendix 5 Categorization, Clustering and Feature Selection Algorithms

This Appendix presents the details of text categorization and clustering algorithms implemented and utilized in this research. Since there are a range of variants of most of the algorithms, it is necessary to clearly describe the algorithms that are involved in this research project when the algorithms are evaluated and utilized in RIB.

The algorithms discussed in this appendix are listed in Appendix Table 5-1. All the algorithms have been programmed according to these specifications except for SVM for which SVM^{Light} is employed, and EM for which a further study is scheduled.

A. Text Categorization Algorithms

1) Naïve Bayes Classification

Naïve Bayes classifier is one of the widely employed, effective, and efficient text categorization algorithms. Suppose a document $\mathbf{d}_j$ is represented as a feature vector $\mathbf{d}_j$ = (t_1,

Appendix Table 5-1 Algorithms discussed in this appendix

type	algorithm	Java implementation
categorization	Multinomial Naïve Bayes	A. 1) Coded by Zhu
	k-nearest neighbours, including tf-idf/z-tfidf term-weighting strategy, cosine similarity and Euclidean distance	A. 2) Coded by Zhu
	Support Vector Machine	A. 3) SVMLight, coded by Thorsten Joachims
	AdaBoost	A. 4) AdaBoost.MH Coded by Zhu
	Statistical Language Model with Jelinek-Mercer & Bayesian smoothing.	A. 5) Coded by Zhu
clustering	K-Means, with MaxMin & Anomalous Pattern centrods initialization, Hierarchical Agglomerative Clustering	K-Means B. 1) Coded by Zhu HAC, Chapter 6, Section 6.1.2 coded by Zhu
	Expectation Maximization (EM)	B. 2) On programming
	Latent Semantic Analysis (LSA)	B. 3) Coded by Zhu
Feature selection	Term-frequency	C. 1) Coded by Zhu
	Mutual information	C. 2) Coded by Zhu
	Information gain	C. 3) Coded by Zhu
	Chi-square	C. 4) Coded by Zhu
	Odds ratio	C. 5) Coded by Zhu
	NGL coefficient	C. 6) Coded by Zhu
	GSS coefficient	C. 7) Coded by Zhu
	Relevancy score	C. 8) Coded by Zhu

t_2, …, t_V) where V is total number of features in a document set D = {$\mathbf{d}_j$ | $\mathbf{d}_j \in$ D, j = 1, …, N}, N is the total number of documents in the event space D. Let C = {c_i | $c_i \in$ C, i = 1, …, M} be a set of categories where M is the total number of categories in C. If a document is assigned one and only one category in C, the probability of $\mathbf{d}_j$ is assigned c_i, according to Bayes rule, is

$$P(C = c_i \mid D = \mathbf{d}_j) = P(C = c_i) \times \frac{P(D = \mathbf{d}_j \mid C = c_i)}{P(D = \mathbf{d}_j)}$$

where P(C=c_i|D=$\mathbf{d}_j$) is the conditional probability of C=c_i given $\mathbf{d}_j$, P(D=$\mathbf{d}_j$ | C=c_i) is the conditional probability of $\mathbf{d}_j$ appears given C=c_i, P(D=d_j) is the prior probability of observing document D=$\mathbf{d}_j$, and P(C=c_i) is the prior probability of document occurring in C=c_i.

In addition,

$$P(D = \mathbf{d}_j) = \sum_{i=1}^{M} P(D = \mathbf{d}_j \mid C = c_i) \times P(C = c_i)$$

This is because naïve Bayes rule supposes that the all possible events (documents) are independent with each other.

To find the most appropriate category for a document, Naïve Bayes classifier assigns the document to the most likely or maximum a posteriori class. Without misunderstanding, this can be expressed as

$$c_{map} = \arg\max_{c_i \in C} \hat{P}(c_i \mid \mathbf{d}_j) = \arg\max_{c_i \in C} \hat{P}(c_i) \times \frac{\hat{P}(\mathbf{d}_j \mid c_i)}{\hat{P}(\mathbf{d}_j)}$$

Further assuming that the occurrence of a particular value of feature t_k is statistically independent of the occurrence of any other features, given that the document is of category c_i, this leads to

$$c_{map} = \arg\max_{c_i \in C} \hat{P}(c_i) \times \frac{\hat{P}(\mathbf{d}_j \mid c_i)}{\hat{P}(\mathbf{d}_j)} = \arg\max_{c_i \in C} \hat{P}(c_i) \times \frac{\prod_{k=1}^{V} \hat{P}(t_{kj} \mid c_i)}{\hat{P}(\mathbf{d}_j)}$$

Note that when estimating the $\arg\max_{c_i \in C} \hat{P}(c_i \mid \mathbf{d}_j)$, $\hat{P}(\mathbf{d}_j)$ is the same for all categories and thus can be neglected, a final estimate of c_{map} is

$$c_{map} = \arg\max_{c_i \in C} \hat{P}(c_i \mid \mathbf{d}_j) \propto \arg\max_{c_i \in C} \hat{P}(c_i) \times \prod_{k=1}^{V} \hat{P}(t_{kj} \mid c_i)$$

V is the number of terms in document set D. $\hat{P}(t_{kj} \mid c_i)$ is the conditional probability of term t_{kj} occurring in a document of class c_i. Taking logarithms to avoid floating point underflow,

$$c_{map} \propto \arg\max_{c_i \in C}(\log \hat{P}(c_i) + \sum_{k=1}^{n_d} \log \hat{P}(t_{kj} \mid c_i)$$

The probability of $\hat{P}(c_i)$ and $\hat{P}(t_{kj} \mid c_i)$ can be estimated by the maximum likelihood estimate,

$$\hat{P}(c_i) = \frac{N_{c_i}}{N}$$

$$\hat{P}(t_{kj} \mid c_i) = \frac{T_{c_i t}}{\sum_{t' \in V} T_{c_i t'}}$$

Where N_{ci} is the number of documents in class c_i, N is total number of documents in document set D, T_{cit} is the frequency of t in training documents from class c_i. To avoid zero estimation, Laplace smoothing is used to estimate $\hat{P}(t_{kj} \mid c_i)$,

$$\hat{P}(t_{kj} \mid c_i) = \frac{T_{c_i t} + 1}{(\sum_{t' \in V} T_{c_i t'}) + |V|}$$

V is the number of terms in the vocabulary set composed of all features in a document collection (Manning, Raghavan, and Schütze 2008).

This model is referred to as multinomial Bayes model, or multinomial model. Another popular model is the binary independent model, or multinomial unigram language model if t_i is simply taken as an indicator of presence (takes value 1) or absence (takes value 0) of the term. The probability of $\hat{P}(t_{kj} \mid c_i)$ is then estimated as

$$\hat{P}(t_k \mid c_i) = \hat{p}_{ki}^{t_k}(1 - \hat{p}_{ki})^{1-t_k}$$
$$= (\frac{\hat{p}_{ki}}{1 - \hat{p}_{ki}})^{t_k}(1 - \hat{p}_{ki})$$

Where $\hat{p}_{ki} = \hat{P}(t_k = 1 \mid c_i)$ is the probability of term t_k appears given category c_i.

Naïve Bayes classifiers and their variants (Eyheramendy, Lewis, and Madigan 2003) have been well researched and applied in text categorization for more than fifty years (Lewis 1998) and usually taken as a base classifier when comparing different categorization algorithms (Yang 1999; Rennie et al. 2003; Kołcz and Yih 2007). However, the effectiveness of this simple text categorization algorithm presented in literatures is not consistent (Lewis et al. 2004; Sebastiani 2002b; Yang 1999; Yang and Liu 1999).

2) kNN Classifier

k-Nearest Neighbour (kNN) classification algorithm is one of the most often used instance-based learning. The *k* nearest neighbors are decided by measuring the distance between a test document and training documents. Measurements of distance are presented in Chapter 2, Section 2.1, Table 2-1.

Some of the measurements of distance involve the count of appearance of terms in a document or in a group of documents. Directly using the raw frequency schema prefers long documents to short ones. Max-Min and Z-score are two methods to address the issue (Larose 2005).

$$Max-Min: \quad x^* = \frac{x-\min(x)}{range(x)} = \frac{x-\min(x)}{\max(x)-\min(x)}$$

$$Z-score: \quad x^* = \frac{x-\mu}{\sigma}$$

Where μ is the mean and σ is standard deviation. To choose an appropriate number of k, one approach is to try some possible k with randomly selected training data sets, and then pick up the k which minimizes the classification error rate (Larose 2005).

With the k selected nearest neighbours, a final category to be assigned to the test document can be decided by the simple, un-weighted Majority Voting algorithm (Zhu 2007). Two variants of Majority Voting are proposed and discussed in detail in Chapter 5, Section 5.1.4. An alternative approach to decide the final category is weighted voting schema (Larose 2005), as introduced in Chapter 2.

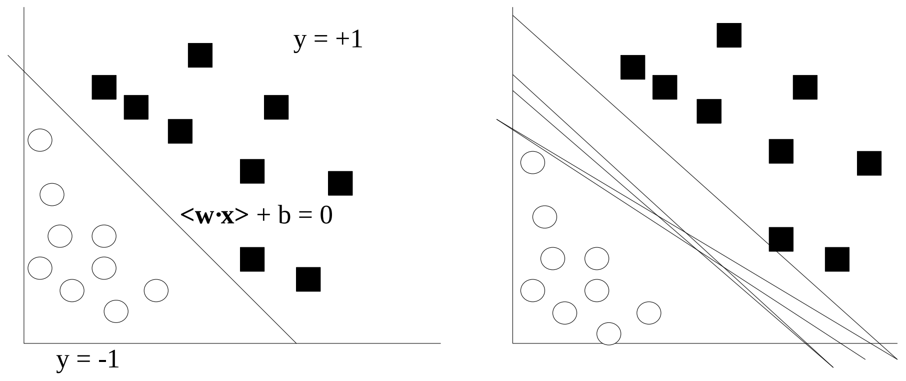
3) Support Vector Machines

Linear SVM for Linear Separable Case

In the simplest case, following the notation of (Burges 1998; Liu 2007), let the linearly separable training data set D = $\{\mathbf{x}_i, y_i | i = 1, 2, \dots n;\ y_i \in \{-1, +1\};\ \mathbf{x}_i \in X \subseteq \mathbf{R}^d\}$, $\mathbf{x}_i$ is a d-dimensional vector with its class label y_i, which is either positive (+1) or negative (-1). The training data can be linearly separated by a separating hyperplane (called the *decision boundary* or *decision surface*), as shown in the left of Appendix Figure 5-1,

(A5-1): $\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0$

where **w** is the normal vector of the hyperplane, and b is the bias. $\langle \mathbf{w} \cdot \mathbf{x} \rangle$ is the dot product of vector **w** and **x**. **w** is perpendicular to the hyperplane, and also called weight vector. Note that



Appendix Figure 5-1 Linear separable data set and possible decision boundaries

changing b will parallel move the hyperplane, the perpendicular distance from a hyperplane to the original is $|b|/||\mathbf{w}||$, and for any $\lambda \in \mathbf{R}^+$ (positive real value), $< \lambda\mathbf{w}\cdot\mathbf{x}> + \lambda b = \lambda(<\mathbf{w}\cdot\mathbf{x}> + b) = 0$. This means the hyperplane is re-scalable. There are many such hyperplanes that can separate the training data set D (right part of Appendix Figure 5-1).

SVMs intend to find a linear decision function

(A5-2): $f(\mathbf{x}) = <\mathbf{w}\cdot\mathbf{x}> + b$

so that if $f(\mathbf{x}_i) \geq 0$, $\mathbf{x}_i$ is assigned to positive class, and otherwise negative class. That is,

$$y_i = \begin{cases} +1 & if < \mathbf{w}\cdot\mathbf{x}_i > +b \geq 0 \\ -1 & if < \mathbf{w}\cdot\mathbf{x}_i > +b \leq 0 \end{cases}$$

Considering Appendix Figure 5-2, let d_+ (d_-) be the shortest distance from the separating hyperplane $<\mathbf{w}\cdot\mathbf{x}> + b = 0$ to the closest positive (negative) example, the margin is $d_+ + d_-$. Schölkopf and Smola (2002) indicate that there exists a “unique optimal hyperplane, distinguished by the maximum margin of separation between any training point and the hyperplane” (p11), and “there are theoretical arguments supporting the good generalization performance of the optimal hyperplane” (p11). Therefore, support vector algorithm intends to find the separating hyperplane H with largest margin.

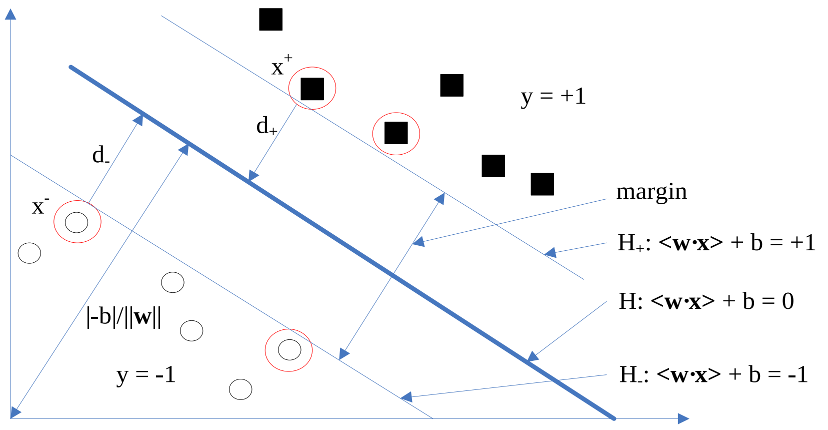
Suppose the training data satisfy the following constraints:

(A5-3): $<\mathbf{w}\cdot\mathbf{x}_i> + b \geq +1$ for $y_i = +1$

(A5-4): $<\mathbf{w}\cdot\mathbf{x}_i> + b \leq -1$ for $y_i = -1$

Or for any i,

(A5-5): $y_i <\mathbf{w}\cdot\mathbf{x}_i> + b -1 \geq 0$



Appendix Figure 5-2 Linear separating hyperplane for the separable case

By choosing a scale for **w** and b, parallel hyperplanes H$_+$ and H$_-$ can be constructed so as for data points (vectors) **x**$^+$ and **x**$^-$, the equality in (A5-3) and (A5-4) holds; these points are called *support vectors*. That is, for these points **x**$_+$ in H$_+$, <**w**·**x**$_+$> + b = +1, for **x**$_-$ in H$_-$, <**w**·**x**$_-$> + b = -1. The perpendicular distances from hyperplane H$_-$ and hyperplane H$_+$ are |1+b|/||**w**|| and |-1+b|/||**w**|| respectively. Because the distance from H to the origin is |b|/||**w**||, therefore, d$_+$ = d$_-$ = 1/||**w**||, the margin is 2/||**w**||, and H is half way between H$_-$ and H$_+$.

Maximize the margin between H$_+$ and H$_-$ is equivalent to minimize ||**w**^{2}||/2 subject to the constraints (A5-5). With the solving of the optimization problem, the hyperplane H can be determined.

Standard Lagrange multiplier method (Luenberger and Ye 2008) can be applied to this problem since the objective function ||**w**^{2}||/2 is a convex quadratic function (Cristianini and Shawe-Taylor 2000), and the constraints of (A5-5) are linear in parameters **w** and b . Instead of optimizing only the objective function, the constraints are also optimized, that is, the following Lagrangian is to be optimized (minimized)

(A5-6): $$L_p = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n}\alpha_i[y_i(<\mathbf{w}\cdot\mathbf{x}_i> + b) - 1]$$

Where α$_1$, α$_2$, …, α$_n$ are the *Lagrange multipliers.*

An optimal solution of (A5-6) should satisfy Kuhn-Tucker conditions (Burges 1998; Liu 2007; Luenberger and Ye 2008). This leads to

(A5-7): $$\frac{\partial L_p}{\partial w_j} = w_j - \sum_{i=1}^{n} y_i\alpha_i x_{i,j} = 0 \quad j = 1,2,\ldots,d$$

(A5-8): $$\frac{\partial L_p}{\partial b} = -\sum_{i=1}^{n} y_i\alpha_i = 0$$

(A5-9): y_i (<$\mathbf{w}\cdot\mathbf{x}_+$> + b) - 1 ≥ 0, i = 1, 2, …, n

(A5-10): $\alpha_i \geq 0$, i = 1, 2, …, n

(A5-11): $\alpha_i(y_i$ (<$\mathbf{w}\cdot\mathbf{x}_+$> + b) – 1) = 0, i = 1, 2, …, n

The above optimization problem is still hard to solve because of the inequality constraints. However, the dual formulation of the optimization problem may simplify things. In general, if the *primal* representation is to minimize $\mathbf{c}^T\mathbf{x}$ subject to $\mathbf{A}\mathbf{x} \geq$ b, x≥0, the dual is to maximize $\boldsymbol{\lambda}^T\mathbf{b}$ subject to $\boldsymbol{\lambda}^T\mathbf{A} \leq c^T$, λ≥0.

Regarding to the particular L_p optimization problem, the dual representation is to maximize L_p, subject to the constraints that the gradient of L_p with respect to set zero of **w** and b, and subject to the constraints that $\alpha_i \geq 0$ as well. Setting the gradient of L_p with respect to **w** and b vanish give the conditions:

$$w_j = \sum_{i=1}^{n} y_i \alpha_i x_{i,j}, \quad j = 1,2,...d$$

$$\sum_{i=1}^{n} y_i \alpha_i = 0$$

Substituting the conditions into equation (A5-6) to eliminate the primal variables gives the dual objective function L_D,

(A5-12):

$$\begin{aligned} L_D &= \frac{1}{2}\sum_{i=1}^{n} y_i\alpha_i x_{i,j} \sum_{j=1}^{n} y_{i,j}\alpha_j x_{i,j} - \sum_{i=1}^{n} y_i\alpha_i \sum_{j=1}^{n} y_j\alpha_j < x_i \cdot x_j > - b\sum_{i=1}^{n}\alpha_i y_i + \sum_{i=1}^{n}\alpha_i \\ &= -\frac{1}{2}\sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j < x_i \cdot x_j > - b \times 0 + \sum_{i=1}^{n}\alpha_i \\ &= \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j < x_i \cdot x_j > \end{aligned}$$

Now, the optimization problem of equation (A5-6) is to maximize its dual expression (A5-12) subject to:

(A5-13): $$\sum_{i=1}^{n} y_i \alpha_i = 0$$
$$\alpha_i \geq 0, \quad i = 1,2,...,n$$

Solution of (A5-12) gives the value of α_i, the support vectors corresponding to α_i, > 0. The α_i can be used to obtain $\hat{\mathbf{w}}$ and $\hat{b}$ by equation (A5-7) and (A5-11). Practically, when calculating $\hat{b}$, to alleviate numerical errors introduced during computing, all support vectors are used to get a value of $\hat{b}$, and the average of the values are the final value of $\hat{b}$. The final linear *decision boundary*, or the *maximal margin hyperplane* is

(A5-14): $$< \hat{\mathbf{w}} \cdot \mathbf{x} > + \hat{b} = \sum_{i \in sv} y_i \alpha_i < \mathbf{x}_i \cdot \mathbf{x} > + \hat{b} = 0$$

where sv indicates the set of support vectors.

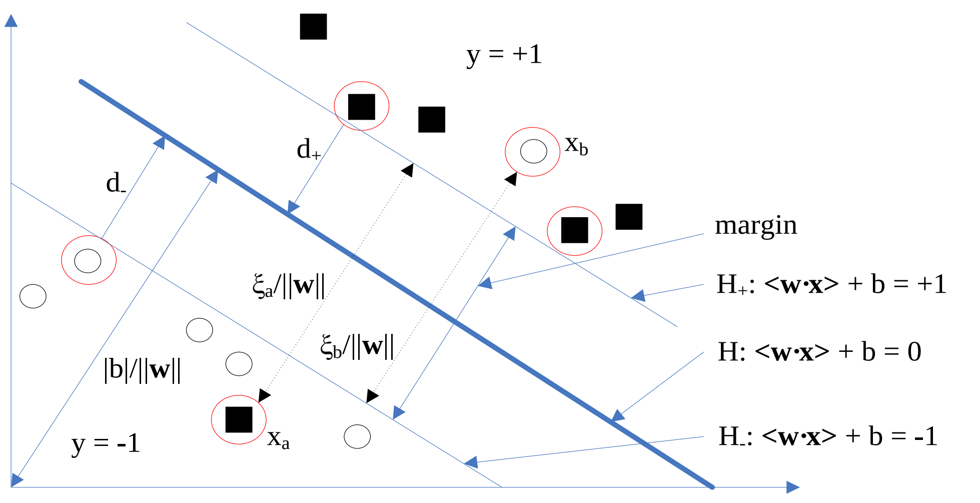
With the final decision boundary, for a given test vector **z**, estimating the following function

(A5-15): $$sign(< \hat{\mathbf{w}} \cdot \mathbf{z} > + \hat{b}) = sign\left(\sum_{i \in sv} y_i \alpha_i < \mathbf{x}_i \cdot \mathbf{z} > + \hat{b} \right)$$

If the function returns +1, the test vector **z** is to be classified as positive, and negative otherwise.

Linear SVM for Linear Non-separable Case

In practice, the existence of noises in training data results the data set is not linearly separable; and in fact, some problems are not linearly separable even in the absence of noise. Appendix Figure 5-3 demonstrates the linear non-separable case where training examples $\mathbf{x}_a$ and $\mathbf{x}_b$ cannot be separated by a linear hyperplane. Solution to this problem (Burges 1998; Cristianini and Shawe-Taylor 2000; Schölkopf and Smola 2002) is to introduce a further cost, called slack variable, to relax the constraints of (A5-3) and (A5-4) when necessary. Set the slack variable $\xi_i \geq 0$ for i = 1, 2, …, n, non-separable SVM is to minimize $||w||^2/2$ with the constraints:



Appendix Figure 5-3 Linear separating hyperplanes for non-linear separable data set. $\mathbf{x}_a$ and $\mathbf{x}_b$ are error examples

(A5-16): $<\mathbf{w}\cdot\mathbf{x}_i> + b \geq 1- \xi_i$ for $y_i = +1$

(A5-17): $<\mathbf{w}\cdot\mathbf{x}_i> + b \leq -1+ \xi_i$ for $y_i = -1$

If an example is not correctly labeled, its corresponding ξ_i must exceed unity, so $\sum_i \xi_i$ is an upper bound of the number of training errors. When an error occurs, an intuitive approach to penalize the error is to change the object function to be minimized from $||w||^2/2$ to

$$||\mathbf{w}||^2/2 + C\sum_i \xi_i$$

where C is a user chosen parameter to indicate how much penalty is to be imposed to errors.

The constraints with slack variables are called *soft-margin SVM*. The primal Lagrangian of this formation is

(A5-18): $$L_p = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i[y_i(<\mathbf{w}\cdot\mathbf{x}_i> + b) - 1 + \xi_i] - \sum_{i=1}^{n}\mu_i\xi_i$$

where α_i, $\mu_i \geq 0$ are the *Lagrange multipliers*. The *Kuhn-Tucker conditions* for optimality are

$$(A5\text{-}19): \quad \frac{\partial L_p}{\partial w_j} = w_j - \sum_{i=1}^{n} y_i\alpha_i x_{i,j} = 0, \quad j = 1,2,\ldots,d$$

$$(A5\text{-}20): \quad \frac{\partial L_p}{\partial b} = -\sum_{i=1}^{n} y_i\alpha_i = 0$$

$$(A5\text{-}21): \quad \frac{\partial L_p}{\partial \xi_i} = C - \alpha_i - \mu_i = 0, \quad i = 1,2,\ldots,n$$

$$\mathbf{y_i\,(<w\cdot x_i> + b) - 1 + \xi_i \geq 0, \quad i = 1, 2, \ldots, n}$$

$$\xi_i, \alpha_i, \mu_i \geq 0, \quad i = 1,2,\ldots,n$$

$$(A5\text{-}22): \quad \alpha_i[\mathbf{y_i\,(<w\cdot x_i> + b) - 1} + \xi_i] = \mathbf{0}, \quad \mathbf{i = 1, 2, \ldots, n}$$

$$(A5\text{-}23): \quad \mu_i\xi_i = 0, \quad \mathbf{i = 1, 2, \ldots, n}$$

As the simplest linear separable case, L_P can be transformed into its dual L_D by setting the gradient of L_P with respect to the primal variables, **w**, b, and ξ_i, to zero, and substituting the resulting relations back into (A5-18). From (A5-19) to (A5-21), there are

$$w_j = \sum_{i=1}^{n} y_i\alpha_i x_{i,j}, \quad j = 1,2,\ldots d$$

$$\sum_{i=1}^{n} y_i\alpha_i = 0$$

$$C - \alpha_i - \mu_i = 0$$

Substituting these equations back into (A5-18) can get L_D.

(A5-24):

$$L_D = \frac{1}{2}\sum_{i=1}^{n} y_i\alpha_i x_{i,j}\sum_{j=1}^{n} y_{i,j}\alpha_j x_{i,j} + (\alpha_i + \mu_i)\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n} y_i\alpha_i \sum_{j=1}^{n} y_j\alpha_j < x_i \cdot x_j >$$
$$- b\sum_{i=1}^{n}\alpha_i y_i + \sum_{i=1}^{n}\alpha_i - \sum_{i=1}^{n}\alpha_i\xi_i - \sum_{i=1}^{n}\mu_i\xi_i$$
$$= -\frac{1}{2}\sum_{i,j=1}^{n} y_i y_j \alpha_i\alpha_j < x_i \cdot x_j > - b \times 0 + \sum_{i=1}^{n}\alpha_i + (\alpha_i + \mu_i)\sum_{i=1}^{n}\xi_i - (\alpha_i + \mu_i)\sum_{i=1}^{n}\xi_i$$
$$= \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} y_i y_j \alpha_i\alpha_j < x_i \cdot x_j >$$

Because $\mu_i \geq 0$ and $C - \mu_i - \alpha_i = 0$, the dual of (A5-18) is

(A5-25):
$$\text{Maximize}: L_D = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} y_i y_j \alpha_i\alpha_j < x_i \cdot x_j >$$
$$\text{Subject to}: \sum_{i=1}^{n} y_i\alpha_i = 0 \quad and \quad 0 \leq \alpha_i \leq C$$

It can be found that the slack variables ξ_i and its Lagrange multipliers μ_i are not in the dual and the object function is identical to (A5-12). With the solution of α_i, $\hat{\mathbf{w}}$ can be estimated by (A5-19). To estimate $\hat{b}$ by (A5-22), α_i should not be zero; note that from (A5-21) and (A5-23), when $0 < \alpha_i \leq C$, μ_i must be zero, and $\hat{b}$ is thus calculated by

$$\hat{b} = \frac{1}{y_j} - \sum_{i=1}^{n} y_i\alpha_i < \mathbf{x}_i \cdot \mathbf{x}_j >, \quad \alpha_j \neq 0$$

The final $\hat{b}$ should be the average over all such training examples (Burges 1998), and the final decision boundary is

(A5-26): $$< \hat{\mathbf{w}} \cdot \mathbf{x} > + \hat{b} = \sum_{i=1}^{n} y_i\alpha_i < \mathbf{x}_i \cdot \mathbf{x} > + \hat{b} = 0$$

As the linear separable case, a test data point is classified by calculating

(A5-27): $$sign(< \hat{\mathbf{w}} \cdot \mathbf{z} > + \hat{b}) = sign\left(\sum_{i \in sv} y_i\alpha_i < \mathbf{x}_i \cdot \mathbf{z} > + \hat{b}\right)$$

One of the important features of SVMs is that its solution is sparse in α_i (Wen, Edelman, and Gorsich 2003). Only a small number of training examples with non-zero α_i, these examples include those lie on the margins ($y_i(<\mathbf{w}\cdot\mathbf{x}_i> + b) = 1$), inside the margin ($\alpha_i = C$ and $y_i(<\mathbf{w}\cdot\mathbf{x}_i> + b) < 1$), or errors. This property enables SVMs work well with large data set.

Another important property of SVMs is, in practical, there is no need to calculate $\hat{\mathbf{w}}$. This can be seen clearly from (A5-15) and (A5-27). This property is crucial for constructing nonlinear decision boundaries by kernel method (Schölkopf and Smola 2002).

Note that C is parameter decided by experiment on training data set, and then verified by cross-validation to give the best classification result.

Nonlinear SVMs

Nonlinear SVMs (Burges 1998; Cristianini and Shawe-Taylor 2000; Hearst et al. 1998; Schölkopf and Smola 2002; Vapnik 1999) deal with the case where the training data set is not linearly separable, as shown in Appendix Figure 5-4. Nonlinear SVMs employ kernel functions to map the original data space (input space) into a so-called feature space where examples are linear separable, and hence linear SVMs can be applied.

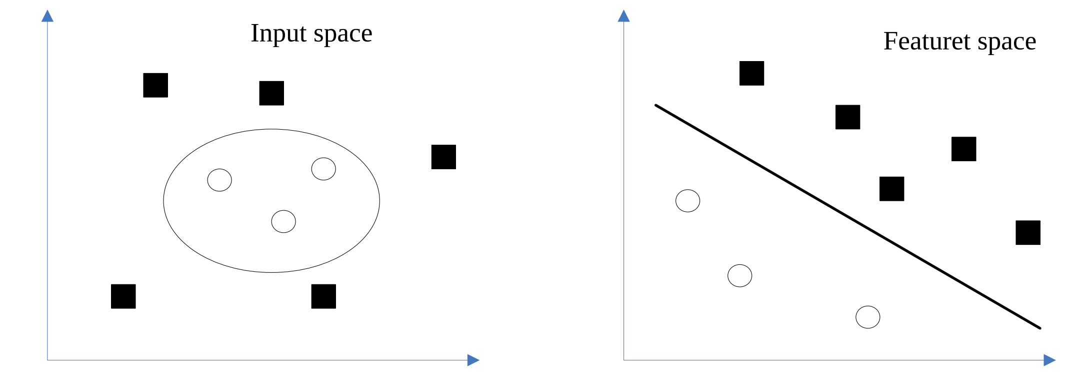
A fundamental concept in nonlinear SVMs is the *kernel function* which is defined as (Shawe-Taylor and Cristianini 2004):

> *A kernel is a function k that for all* $\mathbf{x}, \mathbf{z} \in X$ *satisfies* $k(\mathbf{x}, \mathbf{z}) = <\Phi(x) \cdot \Phi(\mathbf{y})>$ *where* Φ *is a mapping from X to an (inner product) feature space F.* $\Phi : \mathbf{x} \mapsto \Phi(\mathbf{x}) \in F$

A kernel function maps the input data set D = $\{\mathbf{x}_i, y_i | i = 1, 2, \dots n; y_i \in \{-1, +1\}; \mathbf{x}_i \in X \subseteq \mathbf{R}^d\}$ to the feature space H = $\{\Phi(\mathbf{x}_i), y_i | i = 1, 2, \dots n; y_i \in \{-1, +1\}; \{\Phi(\mathbf{x}_i) \in H \subseteq \mathbf{R}^h\}$ where h is dimension of H, and h > d. With this mapping, the optimization task is now to minimize:

$$\|\mathbf{w}\|^2 / 2 + C\sum_i \xi_i$$

Subject to:



Appendix Figure 5-4 Mapping non-linear separable data set by a kernel function to a linear separable space

$$y_i\,(<\mathbf{w}\cdot\Phi(\mathbf{x}_i)>+\,b)\geq 1-\xi_i,\qquad i=1,2,\ldots,n$$
$$\xi_i\geq 0,\quad i=1,2,\ldots,n$$

Or in the dual representation

$$Maximize: L_D=\sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i,j=1}^{n}y_iy_j\alpha_i\alpha_j<\Phi(x_i)\cdot\Phi(x_j)>$$
$$Subject\ to: \sum_{i=1}^{n}y_i\alpha_i=0\quad and\quad 0\leq\alpha_i\leq C,\quad i=1,2,...,n$$

The final decision boundary is

$$<\hat{\mathbf{w}}\cdot\mathbf{x}>+\hat{b}=\sum_{i=1}^{n}y_i\alpha_i<\Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x})>+\hat{b}=\sum_{i=1}^{n}y_i\alpha_i k<\mathbf{x}_i\cdot\mathbf{x}>+\hat{b}$$

Therefore, kernel function plays an essential role in nonlinear separable case.

In the case of text categorization, linear SVMs over perform complex nonlinear ones, and are fast to learn and apply (Dumais and Chen 2000). In this research, a linear SVM is utilized.

4) Boosting

This section presents in detail the Boostexter classifiers. Following Schapire and Singer's (2000) notation, let $\mathcal{X}$ denotes the example document set and $\mathcal{Y}$ denotes a finite set of possible labels or classes/categorizes, the size of $\mathcal{Y}$ is k = | $\mathcal{Y}$|. For each document x $\in \mathcal{X}$, and a set of labels Y $\subseteq \mathcal{Y}$, a tuple (x, Y) indicates a labeled example where Y, which can include multiple labels, is assigned to x. Single-label classification, contrast to multi-label classification for which one document may be assigned more than one labels, is a special case when | $\mathcal{Y}$| = 1 indicates one document is assigned one and only label. Further, define Y[ℓ] for $\ell \in \mathcal{Y}$ to be

$$\mathrm{Y}[\ell]=\begin{cases}+1 & if\ \ell\in Y\\ -1 & if\ \ell\notin Y\end{cases}$$

The BoosTexter intends to rank possible labels for a given example document, and places appropriate labels on the top of the ranking list. That is, for a given x $\in \mathcal{X}$ to find a function f : $\mathcal{X}\times\mathcal{Y}\rightarrow\Re$ so that $\ell\in\mathcal{Y}$ is ranked by f (x,·), and f (x, ℓ_1) > f (x, ℓ_2) indicates ℓ_1 is ranked higher than ℓ_2. Successful classifiers rank labels in Y higher than those not in Y.

AdaBoost is a simple single-label version of Boosting algorithm which weights over training examples according to their importance, and forces weak classifiers to pay more attention to the examples which are most difficult to learn. For multi-label scenario, a single weight for each training example is obviously not sufficient, and a set of weights over training examples and labels are needed to be maintained. The misclassified examples and their corresponding labels will thus receive higher weights, in contrasting with the examples whose weights are lowered for they are easier to be classified. Weighting training examples and their corresponding labels is to force a sequence of weak learners to focus on examples and labels that are hard to learn, and to achieve the final objective of training a highly accurate classifier.

AdaBoost.MH

The AdaBoost.MH algorithm is shown in Appendix Figure 5-5. Let S = <(x_1, Y_1), (x_2, Y_2), …, (x_m, Y_m)> be the training set in which $x_i \in \mathcal{X}$ and $Y_i \subseteq \mathcal{Y}$. The set of weights is taken as a distribution D_t over the examples and labels. D_t is initialized uniformly with 1/(mk). The algorithm repeats an arbitrary T times, and on each round t, the weak classifier takes as input the distribution D_t and training set S to produce a weak hypothesis $h_t(x, \ell) : \mathcal{X} \times \mathcal{Y} \rightarrow \Re$, which predicts whether the label ℓ is assigned to x. The absolute value of $|h_t(x, \ell)|$ can be used to estimate the *confidence* of how well the hypothesis is.

The next step is to calculate a parameter α_t, which is to be discussed later, and then updates the distribution D_t, which is increased to indicate the hypothesis $h_t(x, \ell)$ failed to predict the

```
Given: $(x_1, Y_1), (x_2, Y_2), \ldots, (x_m, Y_m)$ where $x_i \in \mathcal{X}, Y_i \subseteq \mathcal{Y}$
Initialize $D_1(i, \ell) = 1/(mk)$.
For $t = 1, \ldots, T$:
    1) Pass distribution $D_t$ to weak learner
    2) Get weak hypothesis $h_t : \mathcal{X} \times \mathcal{Y} \rightarrow \Re$
    3) Choose $\alpha_t \in \Re$.
    4) Update:
        $D_{t+1}(i, \ell) = \frac{D_t(i, \ell)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i, \ell) = Y_i[\ell] \\ e^{\alpha_t} & \text{if } h_t(x_i, \ell) \neq Y_i[\ell] \end{cases}$
        $= \frac{D_t(i, \ell)\exp(-\alpha_t Y_i[\ell] h_t(x_i, \ell))}{Z_t}$
    where $Z_t$ is a normalization factor (chosen so that $D_{t+1}$ will be a distribution).
        $Z_t = \sum_{i=1}^{m} \sum_{\ell \in \mathcal{Y}} D_t(i, \ell)\exp(-\alpha_t Y_i[\ell] h_t(x_i, \ell))$
Output the final hypothesis:
$f(x, \ell) = \sum_{t=1}^{T} \alpha_t h_t(x, \ell)$.
```

Appendix Figure 5-5 The algorithm AdaBoost.MH (Schapire and Singer 2000) correct label set $Y(\ell)$, that is, $h_t(x, \ell)$ and $Y(\ell)$ differ in sign. The final hypotheses sorts documents utilizing a weighted vote of the weak hypotheses.

Hypothesis produced by AdaBoost.MH intends to predict all and only all of the appropriate labels, thus $H : \mathcal{X} \rightarrow 2^{\mathcal{Y}}$. With regards to distribution D, the loss function is (Schapire and Singer 1999)

$$\begin{aligned} hloss_D(H) &= \frac{1}{k} E_{(x,Y)\sim D}[|h(x)\Delta Y|] \\ &= \frac{1}{k} E_{(x,Y)\sim D}[|h(x)-Y| \cup |Y-h(x)|] \end{aligned}$$

Where Δ denotes the symmetric difference[33] between h(x) and Y, k is a normalization factor to insure that $hloss_D(H) \in [0,1]$, and E is expectation of a random variable.

For multi-label categorization, multi-label issue is reduced to binary data in this algorithm. Each example (x, Y) is mapped to k binary-labeled examples of the form $((x, \ell), Y[\ell])$ for all $\ell \in \mathcal{Y}$. That is, each observed label set Y specifies k binary labels, and binary AdaBoost can thus be applied to the derived binary data. It has been proven (Schapire and Singer 1999) that the upper bound of $hloss_D(H)$ is $\prod_{t=1}^{T} Z_t$, where Zt is the normalization factor calculated on each round t. The bound is an important factor involved in choosing α_t and weak learner. Note that the space and time requirements per-round are O(mk), which does not include the call to the weak learner.

So far, there are still two problems need to be solved, one if how to choose α_t, the other is how to select the weak learner. The two problems will be discussed after the introduction of another version of the boosting algorithm, AdaBoost.MR.

AdaBoost.MR

While the main idea of AdaBoost.MH is to minimize $hloss_D(H)$, AdaBoost.MR intends to search hypothesis which places correct labels on the top of a ranked list. Supposed a labeled observation is (x,Y), two labels $\ell_0 \notin Y$ and $\ell_1 \in Y$, a desired function should minimize the misordering $f(x, \ell_1) \leq f(x, \ell_0)$, or the empirical *ranking loss*:

[33] The term Symmetric difference, which is usually denoted by $A \Delta B$, is the set of elements in one of the sets, but not in both. It can be expressed as $A \Delta B = (A - B) \cap (B - A)$, or equivalently, $A \Delta B = (A \cup B) - (A \cap B)$

$$\frac{1}{m}\sum_{i=1}^{m}\frac{1}{|Y_i||\mathrm{Y}-Y_i|}|\{(\ell_0,\ell_1)\in(\mathrm{Y}-Y_i)\times Y_i : f(x,\ell_1)\le f(x,\ell_0)\}|,$$

$$Y_i \neq \Phi, \mathrm{Y} \neq Y$$

Note that D_t is now a distribution over {1, 2, …, m}$\times \mathcal{Y} \times \mathcal{Y}$, and denote the weight for instance x_i and the pair ℓ_0, ℓ_1 by $D_t(i, \ell_0, \ell_1)$ which is zero except when ℓ_0, ℓ_1 is a crucial pair relative to (x_i, Y_i). The weak hypothesis $h_t : \mathcal{X} \times \mathcal{Y} \rightarrow \Re$ ranks the labels based on D_t which is updated by increasing the weight in case h_t correctly predicts $h_t(x_i, \ell_1) > h_t(x_i, \ell_0)$. Since the empirical ranking loss has been proven is no less than $\prod_{t=1}^{T} Z_t$ (Schapire and Singer 1999), the same as AdaBoost.MH.

α_t and h_t can be derived by minimizing

```
Given: $(x_1, Y_1), (x_2, Y_2), ..., (x_m, Y_m)$ where $x_i \in \mathcal{X}, Y_i \subseteq \mathcal{Y}$
Initialize $D_1(i,\ell_o,\ell_\tau) = \begin{cases} 1/(m \cdot |Y_i| \cdot |\mathcal{Y} - Y_i|) & \text{if } \ell_o \notin Y_i \text{ and } \ell_\tau \in Y_i \\ 0 & \text{else} \end{cases}$
For t = 1, ..., T:
    1) Train weak learner using distribution $D_t$
    2) Get weak hypothesis $h_t : \mathcal{X} \times \mathcal{Y} \rightarrow \Re$
    3) Choose $\alpha_t \in \Re$.
    4) Update:
        $D_{t+1}(i,\ell_o,\ell_\tau) = \frac{D_t(i,\ell)}{Z_t} \times \begin{cases} e^{-\alpha_t} & if\ h_t(x_i,\ell_\tau) > h_t(x_i,\ell_o) \\ e^{\alpha_t} & if\ h_t(x_i,\ell_\tau) < h_t(x_i,\ell_o) \end{cases}$
        $= \frac{D_t(i,\ell_o,\ell_\tau)\exp(\frac{1}{2}\alpha_t(h_t(x_i,\ell_o) - h_t(x_i,\ell_\tau)))}{Z_t}$
    where $Z_t$ is a normalization factor (chosen so that $D_{t+1}$ will be a distribution).
        $Z_t = \sum_{i,\ell_o,\ell_\tau} D_t(i,\ell_o,\ell_\tau)\exp(\frac{1}{2}\alpha_t(h_t(x_i,\ell_o) - h_t(x_i,\ell_\tau)))$
Output the final hypothesis:
$f(x,\ell) = \sum_{t=1}^{T} \alpha_t h_t(x,\ell)$.
```

Appendix Figure 5-6 AdaBoost.MR algorithm (Schapire and Singer 2000)

```
$\textit{Given}: (\mathbf{x}_1, Y_1), (x_2, Y_2), \ldots, (x_m, Y_m)$ where $x_i \in \mathcal{X}, Y_i \subseteq \mathcal{Y}$
$\textit{Initialize}\ v_1(i, \ell) = (m \cdot |Y_i| \cdot |\mathcal{Y} - Y_i|)^{-1/2}$.
For $t = 1, \ldots, T$:
    1) Train weak learner using distribution $D_t(i, \ell_0, \ell_1) = v_t(i, \ell_0) \cdot v_t(i, \ell_1)$
    2) Get weak hypothesis $h_t : \mathcal{X} \times \mathcal{Y} \to \Re$
    3) Choose $\alpha_t \in \Re$.
    4) Update:
        $v_{t+1}(i, \ell) = \frac{v_t(i,\ell)}{\sqrt{Z_t}} \times \begin{cases} e^{-\alpha_t/2} & \text{if } h_t(x_i,\ell) = Y_i[\ell] \\ e^{\alpha_t/2} & \text{if } h_t(x_i,\ell) \neq Y_i[\ell] \end{cases}$
        $= \frac{v_t(i,\ell)\exp(-\frac{1}{2}\alpha_t Y_i[\ell] h_t(x_i,\ell))}{\sqrt{Z_t}}$
    $\textit{where}\ Z_t$ is a normalization factor
        $Z_t = \sum_{i=1}^{m}\left[\left(\sum_{\ell \in Y_i} v_t(i,\ell)\exp(-\frac{1}{2}\alpha_t h_t(x_i,\ell))\right)\left(\sum_{\ell \notin Y_i} v_t(i,\ell)\exp(\frac{1}{2}\alpha_t h_t(x_i,\ell))\right)\right]$
Output the final hypothesis:
$f(x, \ell) = \sum_{t=1}^{T} \alpha_t h_t(x, \ell)$.
```

Appendix Figure 5-7 An efficient AdaBoost.MR.

For each example on each boosting round, the running time is linear in the number of labels (O(k)) (Schapire and Singer 2000)

$$Z_t = \sum_{i,\ell_0,\ell_1} D_t(i, \ell_0, \ell_1)\exp(\frac{1}{2}\alpha_t(h_t(x_i,\ell_0) - h_t(x_i,\ell_1)))$$

For each training example (x_i, Y_i), there are $|Y_i| \cdot |\mathcal{Y} - Y_i|$ weights need to be maintained, the space and time complexity for each round is $O(mk^2)$. Because $D_t(i, \ell_0, \ell_1)$ is always zero except ℓ_0, ℓ_1 is a crucial pair, D_t can be factorized into $D_t(i, \ell_0, \ell_1) = v_t(i, \ell_0) \cdot v_t(i, \ell_1)$, this makes the weights is over $\{1, 2, \ldots, m\} \times \mathcal{Y}$, and time complexity is consequently reduce to O(mk). The AdaBoost.MR algorithm is demonstrated in Appendix Figure 5-6. A more efficient AdaBoost.MR is shown in Appendix Figure 5-7.

Choosing α_t and Weak Hypothesis

Boosting algorithms are designed to combine with any classifiers to boost the predictability of the base or weak classifiers, in this sense, any classifier can be taken as the base classifier. In practice, according to Schapire and Singer (2000), the widely applied weak classifiers are decision trees and neural nets (Han and Kamber 2006; Mitchell 1997). In text categorization, base/weak classifiers employed in boosting are mainly decision trees or even simply decision stumps (Cai and Hofman 2003; Schapire and Singer 1999; Bloehdorn and Hotho 2004), although other classifiers, such as Naïve Bayes is also employed (Kim, Hahn, and Zhang 2000). The reason is that the performance of boosting is affected when base classifiers are too strong or too weak (Han and Kamber 2006; Schapire 2003).

A very simple one-level decision tree is utilized as weak learner in BoosTexter, and all words and pairs of adjacent words are used as possible terms. The one-level decision tree checks the presence and absence of a term in a document at the root of the tree. Outcome of the test is then used by the weak hypothesis to predict with a confidence whether each label is associated with the document (Schapire and Singer 2000). Weak learners first search all possible terms. For each term, the weak learner produces a score to indicate the prediction of a term in a document regarding to a given label. The weak hypothesis with the lowest score is selected to calculate the Z_t.

Let *w* be a possible term, $w \in \mathbf{x}$ indicate term *w* occurs in document **x**, and

$$h(\mathbf{x},l) = \begin{cases} c_{0l} & if \;\; w \notin \mathbf{x} \\ c_{1l} & if \;\; w \in \mathbf{x} \end{cases}$$

The weak learner of AdaBooster algorithms search all possible terms, and calculate values of $c_{j\ell}$ as described in the following four subsections. Based on the values of $c_{j\ell}$, a score is assigned for the resulting weak hypothesis. The weak learner subsequently returns the weak hypothesis that has the lowest score.

AdaBoost.MH with Real-valued Predictions

To calculate $c_{j\ell}$, for a given term *w* and each possible label ℓ, let $X_0 = \{\mathbf{x} : w \notin \mathbf{x}\}$, $X_1 = \{\mathbf{x} : w \in \mathbf{x}\}$, $j \in \{0,1\}$, $b \in \{-1, +1\}$, define

$$W_b^{j\ell} = \sum_{i=1}^{m} D_t(i,\ell)[[x_i \in X_j \wedge Y_i[\ell] = b]]$$

where $[[\,\pi\,]]$ be 1 if π holds and 0 otherwise. That is, $W_+^{j\ell}\,(W_-^{j\ell})$ is the weight of the documents in partition X_j which are (are not) labeled by ℓ. $W_+^{j\ell}\,(W_-^{j\ell})$ is the abbreviation of

$$W_{+1}^{j\ell}\,(W_{-1}^{j\ell})$$

Z_t is proven minimized for a given term by selecting (Schapire and Singer 1999)

$$c_{j\ell} = \frac{1}{2}\ln\left(\frac{W_+^{j\ell}}{W_-^{j\ell}}\right)$$

when $\alpha_t = 1$,

$$Z_t = 2 \sum_{j \in \{0,1\}} \sum_{\ell \in \mathcal{Y}} \sqrt{W_+^{j\ell} W_-^{j\ell}}$$

$W_+^{j\ell}$ *or* $W_-^{j\ell}$ may be zero in practice and causes numerical overflowing problem, $c_{j\ell}$ is smoothed

$$c_{j\ell} = \frac{1}{2} \ln \left(\frac{W_+^{j\ell} + \varepsilon}{W_-^{j\ell} + \varepsilon} \right)$$

When set $\varepsilon = 1/(mk)$, it bounds $| c_{j\ell} |$ by roughly $(\ln(1/\varepsilon))/2$.

The term *w* that minimizes Z_t is chosen as the weak hypothesis.

AdaBoost.MH with Real-valued Predictions and Abstaining

Let $W_0 = \sum_{i: x_i \in X_0} D_t(i, \ell)$ be the weight of all the documents where term *w* does not occur. In this case (Schapire and Singer 1999),

$$Z_t = W_0 + 2 \sum_{\ell \in \mathcal{Y}} \sqrt{W_+^{1\ell} W_-^{1\ell}}$$

As in the previous case, the weak learner selects the term that has the smallest Z_t for each round as the weak hypothesis. This learner considers only the documents that contain term w when calculating Z_t, it is not only more efficient, but also more intuitively reasonable in the sense that it ignores the term which does not occur in a document by forcing weak hypothesis to set confidence values between the term and the documents to zero.

AdaBoost.MH with Discrete Predictions

Let

$$c_{j\ell} = sign(W_+^{j\ell} - W_-^{j\ell})$$
$$r_t = \sum_{j \in \{0,1\}} \sum_{\ell \in \mathcal{Y}} \left| W_+^{j\ell} - W_-^{j\ell} \right|$$

To minimize Z_t, it has been proven (Schapire and Singer 1999) that the value of α_t should be set as

$$\alpha_t = \frac{1}{2} \ln \left(\frac{1 + r_t}{1 - r_t} \right)$$

Giving

$$Z_t = \sqrt{1 - r_t^2}$$

The criteria for selecting the weak hypothesis is the same as the previous algorithms, that is, select the term that minimizing Z_t.

AdaBoost.MR with Discrete Predictions

To approximate Z_t described in subsection AdaBoost.MR, for a given hypothesis h_t, let

$$r_t = \frac{1}{2}\sum_{i,\ell_0,\ell_1} \mathrm{D}_\mathrm{t}(\mathrm{i},\ell_0,\ell_1)(h(x_i,\ell_1) - h(x_i,\ell_0))$$

It can be shown that $Z_t \le \sqrt{1 - r_t^2}$ (Schapire and Singer 1999). Instead of minimizing Z_t, this weak learner tries to minimize its upper bound. Or to maximize r_t. Choose

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1 + r_t}{1 - r_t}\right)$$

Let

$$d_t(\mathrm{i},\ell) = \frac{1}{2} v_t(\mathrm{i},\ell) \sum_{\ell':Y_i[\ell'] \ne Y_i[\ell]} v_t(\mathrm{i},\ell')$$

It can be shown (Schapire and Singer 1999)

$$r_t = \sum_{i,\ell} \mathrm{d}_\mathrm{t}(\mathrm{i},\ell)\mathrm{Y}_\mathrm{i}[\ell]h(x_i,\ell)$$

For a given term *w*, choose

$$c_{j\ell} = sign\Big(\sum_{i:x_i \in X_j} d_t(i,\ell)Y_i[\ell]\Big)$$

giving

$$r_t = \sum_{j\in\{0,1\}} \sum_{\ell \in \mathrm{Y}} \left| \sum_{i:x_i \in X_j} d_t(i,\ell)Y_i[\ell] \right|$$

The term that maximize r_t is selected, and corresponding prediction is assigned.

To speed up the time consuming learning process, Schapire and Singer (2000) suggest to employ an inverted list (Manning, Raghavan, and Schütze 2008; Witten, Moffat, and Bell 1999) to store terms and the list of documents in which they occur. For AdaBoost.MH, it is also recommended to pre-calculate on each round t

$$W^{j\ell} = \sum_{i:x_i \in X_j} D_t(i,\ell)$$

By using inverted list, $W_+^{j\ell}$ is the summarization over documents that contain term w, and then $W_-^{j\ell} = W^{j\ell} - W_+^{j\ell}$.

5) Statistical Language Modelling

In the field of text categorization, based on training examples and given k classes, k different language models {P$_1$(d), P$_2$(d), …, P$_k$(d),} can be trained, applying Bayes's formula, for a given document d, the estimated category is

$$\hat{c} = \arg\max_c P(c|d) = \arg\max_c P(d|c) \times P(c)$$

where the language model P(c) is the estimated likelihood (Rosenfeld 2000).

In the field of information retrieval, according to Liu and Croft (2003), an n-gram language model is a Markov process which estimate

$$P_n(S) = \prod_{i=1}^{m} P(w_i \mid w_{w-1}, w_{i-2}, ..., w_{i-n+1})$$

where m is the number of words in a specified document or word sequence, n is the order of the Markov process, that is, the number of states upon which the next state depends. Further, a Markov process is "a stochastic process with the property that, given the value of X$_t$, the values of X$_s$ for s > t are not influenced by the values of X$_u$ for u < t" (Taylor and Karlin 1998), and a stochastic process is "a family of random variables X$_t$, where t is a parameter running over a suitable index set T" (Taylor and Karlin 1998). Ponte and Croft (1998) suggest an information retrieval model which estimates $\hat{p}(Q \mid M_d)$, the probability of a query Q given the language model M$_d$, by producing the probability of terms in Q, $\prod_{t \in Q} \hat{p}(t \mid M_d)$, and the probability of not producing other terms, $\prod_{t \notin Q} (1 - \hat{p}(t \mid M_d))$, that is,

$$\hat{p}(Q \mid M_d) = \prod_{t \in Q} \hat{p}(t \mid M_d) \times \prod_{t \notin Q} (1 - \hat{p}(t \mid M_d))$$

$\hat{p}(t \mid M_d)$ can simply be estimated by maximum likelihood estimation,

$$\hat{p}_{ml}(t \mid M_d) = \frac{tf(t,d)}{dl(d)}$$

where tf(t,d) is the raw term frequency of term t in document d, and dl(d) is the count of all terms in d. However, when a term does not appear in a document, $\hat{p}(t \mid M_d)$ will be zero, and it will propagate in case even if only one term in the query does not appear in the document, despite all the rest terms in the query are all informative. This is well known data zero-frequency problem (Liu and Croft 2003). To avoid this issue, the probability of the term does not present in a document is assigned to cf(t)/cs, where cf(t) is raw frequency of term t in the document collection, and cs is the total number of tokens in the collection.

Introducing

$$\hat{p}_{avg}(t) = \frac{\sum_{d_{t \in d}} p_{ml}(t \mid M_d)}{df(t)}$$

to alleviate potential issues caused by maximum likelihood estimate, where df(t) is the document frequency of term t. Further introducing

$$\hat{R}_{t,d} = \left(\frac{1.0}{1.0 + \bar{f}(t)} \right) \times \left(\frac{\bar{f}(t)}{1.0 + \bar{f}(t)} \right)^{tf(t,d)}$$

where $\bar{f}(t) = p_{avg}(t) \times dl(d)$ is the mean frequency of term t in documents where it appears.

$\hat{p}(t \mid M_d)$ is finally estimated by

$$\hat{p}(t \mid M_d) = \begin{cases} p_{ml}(t,d)^{(1.0 - \hat{R}_{t,d})} \times p_{avg}(t)^{\hat{R}_{t,d}} & if\ tf(t,d) > 0 \\ cf(t)/cs & otherwise \end{cases}$$

Experimental results on TREC (Text REtrieval Conference, http://trec.nist.gov/) demonstrate this conceptual simple model is superior to standard tf-idf weighting strategy (Ponte and Croft 1998).

Introducing an adjustable parameter to the maximum likelihood estimator to improve its accurate is referred to as *smoothing* technique, which is the research focus for many language models (Zhai and Lafferty 2001). Experimental outcomes reveal that the performance of language models is sensitive to the designed smoothing parameters, which serve not only as a mechanism to boost the effectiveness of language models, but also as a mean to generate common and non-informative words in a query (Zhai and Lafferty 2001).

A simple but practically effective smoothing approach (Song and Croft 1999) is to combine a document model $\hat{p}(t \mid M_d)$ with the collection model,

$$\hat{p}(t \mid M_d) = \lambda \hat{p}_{ml}(t \mid M_d) + (1 - \lambda)\hat{p}_{ml}(t \mid M_C)$$

where 0 < λ < 1, and the collection model $\hat{p}_{ml}(t \mid M_C)$, similar to document model, is estimated by

$$\hat{p}_{ml}(t \mid M_C) = \frac{tf(t,C)}{dl(C)}$$

Where C represent the document collection, tf(t,C) is the raw frequency of term t occurs in the collection, and dl(C) is the number of terms in the collection. This approach is referred to as *linear interpolation* language model (Manning, Raghavan, and Schütze 2008), or *Jelinek-Mercer* smoothing (Zhai and Lafferty 2001). A variant of this approach is to assign a different smoothing parameter λ_i for each query term q_i,

$$\hat{p}(q_i \mid M_d) = \lambda_i \hat{p}_{ml}(q_i \mid M_d) + (1-\lambda_i)\hat{p}_{ml}(q_i \mid M_C)$$

Another popular smoothing scheme, which is called *Bayesian smoothing using Dirichlet priors* (Zhai and Lafferty 2001), is to estimate a language model by utilizing collection model as a prior distribution in a Bayesian updating process,

$$\hat{p}(t \mid M_d) = \frac{tf(t,d) + \alpha \hat{p}_{ml}(t \mid M_C)}{dl(d) + \alpha}$$

Experimental results presented by Zhai and Lafferty (2001) suggest for Jelinek-Mercer smoothing model, maximum retrieval performance is reached for short queries, when $\lambda = 0.1$, and for long verbose queries when $\lambda = 0.7$. Bayesian smoothing is more suitable for short queries, and the optimal value of α is around 2000, or with the range from 500 to 10000.

B. Text Clustering Algorithms

1) K-Means

K-Means uses *minimum distance rule*, which is very popular in data analysis, to assign members into their nearest centroids (Jain, Murty, and Flynn 1999; Mirkin 2005). The most widely employed distance metric is *Euclidean distance* which is a special case of the more general Minkowski metric. A list of commonly utilized distance measurements is presented in Chapter 2, Figure 2-1.

In the field of text clustering, let V be the set of vocabulary in the document collection, a document d is represented by a V-dimensional vector, or a formed cluster S_i with centroid $\mathbf{c}_i$, define *within cluster error* as

$$E^2(S_i, \mathbf{c}_i) = \sum_{k=1}^{n_i} D(\mathbf{x}_k, \mathbf{c}_i) = \sum_{k=1}^{n_i}\sum_{j=1}^{V}(x_{k,j} - c_{i,j})^2$$

which is the summary of distances ($D(\mathbf{x}_k, \mathbf{c}_i)$) from the centroids **c**$_i$ to the document **x**$_i$, i = 1, 2, …, n$_i$, in the cluster, n$_i$ is the size of S$_i$. The square error of cluster S is the aggregate of $E^2(S_i, \mathbf{c}_i)$ over all clusters

$$E^2(S, \mathbf{c}) = \sum_{k=1}^{K} E^2(x_k, \mathbf{c}_k)$$

The K-Means algorithm is shown in Appendix Figure 5-8. It tells nothing about how to choose the number of K, and how to choose the initial K centroids. Different initialization leads to different clustering results. One approach is to assign an arbitrary K, with randomly selected K centroids as the initialization, at the high risk of failing to construct clusters that are meaningful to the specific application scenario. An alternative strategy is Incremental K-Means (Mirkin 2005), which also assigns an arbitrary K, and randomly chooses K tentative centroids. However, it then processes only one entity at each iteration, arranges the entity to a cluster based on minimum distance rule, updates the corresponding centroids and re-constructs the clusters if necessary.

Selecting an appropriate K is essential for K-Means algorithms. Many approaches have been proposed and studied (Chiang and Mirkin 2006). Among the algorithms, MaxMin and Anomalous Pattern (AP) are two approaches which are simple and effective (Mirkin 2005).

MaxMin for Initializing Centroids

Based on the clustering intuition that documents (or patterns) with any cluster must be close to each other and far away from documents in other clusters, *MaxMin* intends to form an initial set of centroids. The algorithm first choose two documents x_i and x_j meet $D(x_i^0, x_j^0) = \max_{x_i, x_j \in I} D(x_i, x_j)$ where I = {x$_i$| i = 1, 2, … N}, N is the total number of documents, x_i and x_j forms an initial seed set S = { x_i, x_j }. The second step is to calculate the minimum distance $D(x_i^k, x_j^k) = \max_{x_i^S \in S, x_j^I \in I-S} D(x_i^S, x_j^I)$, that is, for each $x_j \in$ I - S, calculate the

1. Choose the number K, and tentative centroids c1, c2, … Ck, assume the initial cluster list S$_i$ (i = 1, 2, …, K) is empty
2. Assign each document to the closest centroids
3. Recompute the cluster centroids using the current cluster members to obtain a new cluster set S$_i$’ (i = 1, 2, …, K)
4. If convergence criterion is not satisfied, go to step 2. otherwise, output the formed cluster. Convergence criterion: no (or minimal) reassignment of document to new clusters; or square error ceases to decrease significantly after some number of iterations

Appendix Figure 5-8 K-Means clustering algorithm (Mirkin 2005) distance between x_j and $x_i \in S$, let the one closest to $x_i \in S$ is $x_{i,j}$, $j = 1, \ldots|S|$, $|S|$ is the size of the S, and corresponding distance is $d_{min}(i, j)$. The last step check whether stop-condition is satisfied, if either or all of the following condition is satisfied, S is output as the initial centroids and K is assigned to $|S|$; otherwise, choose the maximum distance $d_{min}(i, j)^{max}$ from $d_{min}(i, j)$, $I = 1, 2, \ldots, |S|$, and let $x_{i,j}$ is the corresponding document in I - S, remove $x_{i,j}$ from I-S and add it to S as a new centroids. The stop-condition is: 1) $|S|$ is bigger than a predefined a threshold K_0; 2) the distance $d_{min}(i, j)^{max}$ is larger than a predefined threshold, such as one third of the distance between the initial two centroids; 3) the ratio of $d_{min}(i, j)^{max}$ in this iteration to the previous iteration is significant, for example, 35% (Mirkin 2005). The main drawbacks of the algorithm are it is computational expensive and the stop-condition needs to be decided experimentally.

Anomalous Pattern

To mitigate the computational intensive issue of K-Means for which the upper bound on running time is in general exponential in the number of points (Vattani 2009), one solution is to compare documents with a *reference document* rather than among documents. A typical such reference document is the mean or centroids of all documents in a cluster. Then, an *anomalous pattern* (document) which is far from the *reference pattern* is an available candidate seed. This Anomalous Pattern (AP) (Mirkin 1999, 2005) algorithm is illustrated in Appendix Figure 5-9.

With the formed clusters, some clusters have only one document, or less than a specified number of documents, should be removed from the cluster list, and a final cluster number K is then recalculated. A predefined threshold K_0 can also be a stop condition if the number formed clusters reached K_0. Standard K-Means algorithm can then be employed with the formed number of K and the centroids c_i, $i = 1, 2, \ldots, K$. This approach is called *Intelligent K-Means algorithm*, or *iK-Means* which works well on low dimensional space, and may fail when pattern are scattered too far away from each other. In other words, iK-Means is not

1. Preprocessing. Specify pattern means a as reference pattern, standardize all the patterns by
   $y_{i,a} = (x_i - a) / b_a$ where $x_i \in I$ is a pattern, b_a can be standard deviation or a value which quantifies the span of x_i.
2. Initialization. Calculate a tentative centroid c which is the farthest from the reference pattern
3. Cluster update. For all pattern y_j, if $D(y_j, c) < D(y_j, a)$, put y_j into the cluster S_k where c is the centroid
4. Centriod update. Calculate the new mean c' of S_k, if c' differs from the previous centroid c, let c = c', and then goto step 3. if c is equal to c', goto next step.
5. Update pattern set. Excluding S_k from I, that is, $I \leftarrow I - S_k$. If I is empty, end; otherwise go to step 2.

Appendix Figure 5-9 Anomalous Pattern clustering algorithm (Mirkin 2005, 1999) good at dealing with sparse data set (Mirkin 2005).[34]

2) Expectation-Maximization (EM) Algorithm

For text clustering, Let X = {$x_i \in \Re^T$; i = 1, …, N} is the observed document set, N is the number of documents and T is the dimension of documents in X; C = {$C_1, C_2, \ldots, C_M$} is the cluster set and M is the number of clusters, further supposed the probability of cluster C_j is α_j, thus $\alpha_1 + \alpha_2 +, \ldots + \alpha_M = 1$; Y = {$y_k$, k = 1, …, T'} is the set of missing data which specifies the hidden-state information; θ = { θ_m; m = 1, …, M} is the set of unknown parameters that define the density function for estimating the true density of X; $\theta^{(i)} = \{\pi^{(i)}, \varphi^{(i)}\}$ where $\pi^{(i)}$ is the prior probability of the i-th component density, $\varphi^{(i)}$ is the i-th component density. The complete data set is composed of X and Y.

Suppose the probabilistic distribution model of clusters is

$$p(x \mid \theta) = \sum \alpha_i p_i(x \mid \theta)$$

Where $\Theta = \{\alpha_1, \ldots, \alpha_M, \theta_1, \ldots \theta_M\}$, p_i is a density function parameterized by θ. The log likelihood functions for the incomplete data X and complete data set {X + Y} are

$$\log(L(\Theta \mid X)) = \log(P(X)) = \log \prod_{i=1}^{N} p(x_i \mid \Theta) = \sum_{i=1}^{N} \log\left(\sum_{j=1}^{M} \alpha_j p_j(x_i \mid \theta_j)\right)$$

$$\log(L(\Theta \mid X, Y)) = \log(P(X, Y \mid \Theta)) = \log \prod_{i=1}^{N} p(y_i \mid x_i, \theta_i) p(y_i \mid \theta_i)$$

Define

$$\begin{aligned}
Q(\theta \mid \theta^{(n)}) &= E_Y(\log L(\theta \mid X, Y) \mid X, \theta^{(n)}) \\
&= \sum_Y \log L(\theta \mid X, Y) P(Y \mid X, \theta^{(n)}) \\
&= \sum_Y \{P(Y \mid X, \theta^{(n)})\}\{\log P(X, Y \mid \theta)\} \\
&= \sum_Y \{P(Y \mid X, \theta^{(n)})\}\{\log P(Y \mid \theta) P(X \mid Y, \theta)\} \\
&= \sum_{y_{1=1}}^{M} \cdots \sum_{y_{1=1}}^{M} \left\{\prod_{j=1}^{N} P(y_j \mid x_j, \theta^{(n)})\right\}\left\{\sum_{i=1}^{N} \log P(y_i \mid \theta) P(x_i \mid y_i, \theta)\right\} \\
&= \sum_{y_{1=1}}^{M} \cdots \sum_{y_{N=1}}^{M} \left\{\prod_{j=1}^{N} P(y_j \mid x_j, \theta^{(n)})\right\}\left\{\sum_{i=1}^{N} \log(\alpha_i P(x_i \mid y_i, \theta))\right\}
\end{aligned}$$

[34] This observation is also verified by the experimental results provided in Chapter 6.

The last expression can further be simplified into (Bilmes 1998)

$$Q(\theta|\theta^{(n)})=\sum_{k=1}^{M}\sum_{i=1}^{N}\log(\alpha_k P_k(x_i|\theta_k))P(k|x_i,\theta^{(n)})$$
$$=\sum_{k=1}^{M}\sum_{i=1}^{N}\log(\alpha_k)P(k|x_i,\theta^{(n)})+\sum_{k=1}^{M}\sum_{i=1}^{N}\log(P_k(x_i|\theta_k))P(k|x_i,\theta^{(n)})$$

With the constraint $\sum_i \alpha_i = 1$, performing Lagrange optimization with a Lagrange multiplier λ,

$$\frac{\partial}{\partial\alpha_k}\left[\sum_{k=1}^{M}\sum_{i=1}^{N}\log(\alpha_k)P(k|x_i,\theta^{(n)})+\lambda\left(\sum_k\alpha_k-1\right)\right]=0$$
$$or$$
$$\sum_{i=1}^{N}\frac{1}{\alpha_k}P(k|x_i,\theta^{(n)})+\lambda=0$$

This will yields λ = –N and

$$\alpha_k=\frac{1}{N}\sum_{i=1}^{N}P(k|x_i,\theta^{(n)})$$

In case there exists an analytical expression for θ_k as a function, such as if a one parameter Poisson distribution is assumed for the clusters (Chakrabarti 2003), that is, $\theta_k \sim e^{-\mu}\frac{\mu^x}{x!}$ for x = 0, 1, …. Then, let

$$\frac{\partial}{\partial\mu_k}\left[\sum_{k=1}^{M}\sum_{i=1}^{N}P(k|x_i,\theta^{(n)})(-\mu_k+x_i\log\mu_k)\right]=0$$

This leads to

$$\sum_{i=1}^{N}(-1+\frac{x_i}{\mu_k})P(k|x_i,\theta^{(n)})=0$$
$$\mu_k=\frac{\sum_{i=1}^{N}x_iP(k|x_i,\theta^{(n)})}{\sum_{i=1}^{N}P(k|x_i,\theta^{(n)})}$$

Define $L(X|\theta^{(n)})\equiv\log p(X|\theta^{(n)})$ as the likelihood of X, where $\theta^{(n)}$ is the estimated parameters after nth iteration, and $L(X,Y|\theta^{(n)})\equiv\log p(X,Y|\theta^{(n)})$ is the complete data likelihood. EM algorithm intends looking for the expected value of $L(X,Y|\theta^{(n)})$ with respect to the unknown data Y given the observed X and the expected $\theta^{(n)}$, that is, define

$$Q(\theta|\theta^{(n)})\equiv E_Z\{\log p(X,Y|\theta)|X,\theta^{(n)}\}$$

Note that X and $\theta^{(n)}$ are constants, Y is unknown random variable governed by the distribution $p(y|X, \theta^{(n)})$.

3) Latent Semantic Analysis for Clustering

Singular Value Decomposition

Let A be a m×n (m≥ n) matrix, and rank(A) = r, the singular value decomposition SVD of A is:

$$A = USV^T$$

where U, V, referred to as matrix of left and right singular vectors, are orthogonal matrix their first r columns define the orthonormal eigenvectors associated with the r nonzero eigenvalues of AA^T and A^TA respectively. $U^TU = V^TV = I_n$, and $S = \mathrm{diag}(\sigma_1, \sigma_2, ..., \sigma_n)$ is the matrix of singular values, here $\sigma_i > 0$ for i = 1, ..., r, $\sigma_i = 0$ for $i \geq r+1$.

Let $A_k = \sum_{i=1}^{k} u_i \cdot \sigma_i \cdot v_i^T$ is formed from the k largest singular triplets of A, it is proven that A_k can best estimate A (Furnas et al. 1988).

$$\min_{rank(B)=k} = \| A - B \|_2 = \| A - A_K \|_2 = \sigma_{k+1}^2$$

$\|X\|_2$ is the Euclidean norm of matrix X, B is a m×n matrix of rank r which is the best approximation of A (Gentle 2007).

Latent Semantic Indexing

Let A is the terms by documents matrix, and a_{ij} is the weight of term i in document j. tf-idf (Salton and Buckley 1988) is a widely utilized schema for weighting a term in a document. Using SVD factorizes A into S, U, and V, keeps only the k largest singular values of S along with the corresponding rows and columns of U and V to approximate the original A (Berry, Dumais, and O'Brien 1995; Deerwester et al. 1990; Furnas et al. 1988),

$$\begin{aligned}\hat{A} &= U_{mk} S_{kk} V_{kn}^T \\ &\approx A\end{aligned}$$

Selection of k is essential to LSI. A bigger k may fit well to the real latent structure, and a smaller k may filter out more noises. It is suggested the selected k should maximize information retrieval performance (Furnas et al. 1988).

Geometrically, in the k-dimensional space, S is served "to stretch or shrink the orthogonal axes of this space", and "terms and documents become points" in this space (Furnas et al. 1988).

Document **d** is estimated in the LSI space is

$$\hat{\mathbf{d}} = d^T U_{mk} S_{kk}^{-1}$$

For a given query **q** which is represented as a vector the same way as a document, in the k-dimensional LSI space, is estimated as

$$\hat{\mathbf{q}} = q^T U_{mk} S_{kk}^{-1}$$

The similarity between a query and document can then be calculated by the Vector Space Model (Salton, Wong, and Yang 1975) which takes the cosine value of two vectors with angle θ as their similarity.

$$sim(\mathbf{d_j}, \mathbf{q}) = \cos(\theta) = \frac{\mathbf{d_j} \bullet \mathbf{q}}{|\mathbf{d_j}| \times |\mathbf{q}|} = \frac{\sum_{i=1...m} a_{i,j} \times q_i}{\sqrt{\sum_{i=1...m} a_{i,j}^2 \times \sum_{i\,=\,1...m} q_i^2}}$$

In a dynamic data set environment where new documents and terms are to be added in, some techniques are developed to update the estimation approach in order to save computation time (Berry, Dumais, and O'Brien 1995). As the improvement of the computation power of computers, it is appropriate to re-calculate the SVD to get better performance.

LSI can also be used to compare the similarity among terms and similarity among documents. Similarity between two terms can be compared by the dot product of the corresponding row vectors of $\hat{A}$, and the square matrix $\hat{A}\hat{A}^T$ contains all these term to term dot products. Notice that U is orthonormal and S is diagonal,

$$\hat{A}\hat{A}^T = U_k S_k V_k^T V_k S_k^T U_k^T = U_k S_k I_k S_k^T U_k^T = U_k S_k^2 U_k^T$$

Similarly the similarity among documents is calculated by following formula

$$\hat{A}^T \hat{A} = V_k S_k^T U_k^T U_k S_k V_k^T = V_k S_k I_k S_k^T V_k^T = V_k S_k^2 V_k^T$$

C. Feature Selection Algorithms

1) Frequency-based Approach

Frequency-based approaches select features that are most frequent in a class. Frequency can be document frequency, the number of documents in a class that contain a given feature; or collection frequency, the number of a feature occurs in documents in a class. Document frequency is the simplest approach for dimensionality reduction, and can easily scale to very large document collection. When a large number of features are selected, frequency-based approaches are comparable with other approaches in terms of effectiveness; however, when only a small portion of features are selected, frequency-based approaches perform worse.

2) Mutual Information

Mutual Information (Yang and Pedersen 1997; Church and Hanks 1990) tries to compare the probability of event x and event y together (the joint probability) with the probabilities of observing x and y independently (chance). That is,

$$MI(x,y) = \log\frac{P(x,y)}{P(x)P(y)}$$

If x and y are independent, P(x,y) = P(x)P(y) and MI(x,y)=0; when a genuine association exists between x and y, P(x,y) should larger than P(x)P(y), and thus MI(x,y) > 0. Substitute event x and y with feature t_k and category c_i,

$$MI(t_k,c_i) = \log\frac{P(t_k,c_i)}{P(t_k)P(c_i)}$$

considering a contingency table of feature t_k and category c_i, let A denotes the number of times t_k and c_i co-occur; B is the number of times t_k occurs without c_i; C is the number of times c_i occurs without t_k; and D is the number of times neither is occur (Appendix Table 5-2); N is the total number of documents, then IG can be estimated as

$$MI(t_k,c_i) \approx \log\frac{A\times N}{(A+C)\times(A+B)}$$

To estimate how goodness a feature for all categories, two alternative approaches are given as follows:

$$MI_{avg}(t) = \sum\nolimits_{i=1}^{m} P(c_i)MI(t,c_i)$$
$$MI_{\max}(t) = \max\nolimits_{i=1}^{m}\{MI(t,c_i)\}$$

Note that MI can also be expressed as:

Appendix Table 5-2 Contingency table of t_k and c_i

	c_i occurs	**c_i not occurs**
t_k occurs	A	B
t_k not occurs	C	D

$$\begin{aligned} MI(t_k,c_i) &= \log\frac{P(t_k,c_i)}{P(t_k)P(c_i)} \\ &= \log\frac{P(t_k\mid c_i)P(c_i)}{P(t_k)P(c_i)} \\ &= \log P(t_k\mid c_i) - \log P(t_k) \end{aligned}$$

This indicates MI is sensitive to the marginal probabilities of individual features; when two features have an identical conditional probability logP(t_k|c_i), MI assign a higher score to the rare feature (Yang and Pedersen 1997).

3) Information Gain

Information gain (Mitchell 1997; Yang and Pedersen 1997) "measures the number of bits of information obtained for categorization prediction by knowing the presence or absence of a term in a document."(Yang and Pedersen 1997) Let C = {c_i | i = 1, 2, …, m} is set of categories, then the information gain of feature t is defined as:

$$\begin{aligned} IG(t,c) &= \sum_{c\in\{c_i,\bar{c}_i\}}\sum_{t\in\{t_k,\bar{t}_k\}} P(t,c)\log\frac{P(t,c)}{P(t)P(c)} \\ &= -\sum_{i=1}^{m} P(c_i)\log P(c_i) \\ &\quad + P(t_k)\sum_{i=1}^{m} P(c_i\mid t_k)\log P(c_i\mid t_k) \\ &\quad + P(\bar{t}_k)\sum_{i=1}^{m} P(c_i\mid \bar{t}_k)\log P(c_i\mid \bar{t}_k) \end{aligned}$$

Using the contingency table (Table A5-2), IG can be expressed as:

$$\begin{aligned} IG(t,c) &= \frac{A}{N}\log\frac{N\times A}{(A+B)\times(A+C)} + \frac{C}{N}\log\frac{N\times C}{(C+D)\times(A+C)} \\ &\quad + \frac{B}{N}\log\frac{N\times B}{(A+B)\times(B+D)} + \frac{D}{N}\log\frac{N\times D}{(C+D)\times(B+D)} \end{aligned}$$

This is how the IG is estimated.

To reveal the relationship between IG and MI, note that IG can also be expressed as:

$$\begin{aligned} IG(t) &= \sum_{c\in\{c_i,\bar{c}_i\}}\sum_{t\in\{t_k,\bar{t}_k\}} P(t,c)\log\frac{P(t,c)}{P(t)P(c)} \\ &= \sum_{i=1}^{m} P(t,c_i)MI(t,c_i) + \sum_{i=1}^{m} P(\bar{t},c_i)MI(\bar{t},c_i) \end{aligned}$$

It shows that IG is the weighted average of the MI(t, c) and MI($\bar{t}$, c), where the weights are the joint probabilities P(t, c) and P($\bar{t}$, c) respectively. So, IG takes into consideration of feature absence in the form MI($\bar{t}$, c) which is ignored by MI; also, MI uses non-normalized scores and IG normalizes the MI scores by the joint probabilities (Yang and Pedersen 1997).

4) Chi-square χ^2

The chi-square χ^2 statistic (Yang and Pedersen 1997) estimates the independence of two events, the occurrence of a feature t and the occurrence of a class c. Using the contingency table (Table A5-2), the χ^2 is:

$$\chi^2(t,c) = \frac{N \times (AD - BC)^2}{(A+C)\times(B+D)\times(A+B)\times(C+D)}$$

Like mutual information evaluation, χ^2 statistic equals zero when t and c are independent. The higher the χ^2 value is, the less true the hypothesis that t and c independent is. The global χ^2 statistic can be estimated by taking the average or maximum χ^2 value as follows (Yang and Pedersen 1997):

$$\chi^2_{avg} = \sum_{i=1}^{m} P(c_i)\chi^2(t,c_i)$$
$$\chi^2_{\max} = \max_{i=1}^{m}\{\chi^2(t,c_i)\}$$

According to Yang and Pedersen (1997), χ^2 is a normalized value that differs from MI, χ^2 values are comparable across features for the same category. When any cell in the contingency table is lightly populated in case of low frequency feature, this normalization breaks down. Hence the χ^2 statistic is known not to be reliable for low frequency features.

5) Odds Ratio

Odds ratio was originally suggested for selecting terms for relevance feedback based on the assumption that the distribution of features on relevant documents is different from the distribution of features on irrelevant documents (Rijsbergen, Harper, and Porter 1981). Given the contingency table (Table A5-2), the Odds Ratio can be expressed as

$$\begin{aligned} OR &= \log\frac{odds(t_k \mid c)}{odds(t_k \mid \bar{c})} \\ &= \log\frac{P(t_k \mid c_i)(1-P(t_k \mid \bar{c}_i))}{P(t_k \mid \bar{c}_i)(1-P(t_k \mid c_i))} \\ &= \log\frac{A/C}{B/D} = \log\frac{A\times D}{B\times C} \end{aligned}$$

To handle the singularities, that is, when any of the A, B, C, or D is zero, following the approach proposed by Shaw JR (1995), the OR is replaced with:

$$OR = \begin{cases} \log \dfrac{D}{B \times (N^2 - 1)} & if\ A = 0 \\ \log \dfrac{D \times (N^2 - 1)}{B} & if\ C = 0 \\ \log \dfrac{A \times (N^2 - 1)}{C} & if\ B = 0 \\ \log \dfrac{A}{C \times (N^2 - 1)} & if\ D = 0 \end{cases}$$

Odds ratio can also be interpreted as the sum of the logarithm of the ratios of the distribution of the feature on the relevant document and on the irrelevant documents. If a feature appears in more than half of the relevant documents the logarithm of the ration on the relevant documents is positive. In contrast a feature is penalized if it appears in more than half of the irrelevant documents. That is, a feature that appears frequently in the relevant documents and infrequently in the irrelevant documents will have a high score (Ruiz and Srinivasan 2002).

6) NGL Coefficient

NGL coefficient (Ng, Goh, and Low 1997) is a variant of a "one sided" χ^2 statistic metric by taking one of the square root of χ^2. They argue that the rationale behind this is the emphasizing of the words that only come from the relevant texts of a category C, and de-emphasizing the words that come from the irrelevant texts or are highly indicative of non-membership in C. NGL coefficient selects exactly these indicative words of category C, while χ^2 selects not only these words set, but also those words that are indicative of non-membership in C.

7) GSS Coefficient

GSS coefficient (Galavotti, Sebastiani, and Simi 2000) is another variant of the χ^2 statistic metric by going a further step in the direction of NGL coefficient. In NGL coefficient, $\sqrt{N}$ is redundant because it is same for all pairs of (t_k, c_i), and can consequently be removed. For extremely rare features and categories, $\sqrt{P(t_k)P(\bar{t}_k)}$ and $\sqrt{P(c_k)P(\bar{c}_k)}$ are both very small. The two factors presented at the denominator of NGL coefficient will emphasize extremely rare features and extremely rare categories, and thus should also be removed. The final GSS coefficient, $P(t_k, c_i)P(\bar{t}_k, \bar{c}_i) - P(t_k, \bar{c}_i)P(\bar{t}_k, c_i)$, is a simplified χ^2 as shown in Chapter 2, Table 2-5.

8) Relevancy Score

Relevancy score is introduce by (Wiener, Pedersen, and Weigend 1995a) with the belief that relevancy score performs no difference with other feature selection algorithms such as information gain, or chi-square. Relevancy score is calculated based on Salton and Buckley's relevancy weight to measure how unbalance a term in across documents with and without a category, where 1/6 is a dampening factor.

$$\log \frac{P(t_k, c_i) + 1/6}{P(\bar{t}_k, \bar{c}_i) + 1/6}$$

they found when 20 features are selected, their classifier performs best.

Appendix 6 Case Study of Personalized Results of RIB

A. Output of User Profile after Initialization

```
Sports    -5345.038309054697
Computers    -5412.423555197079
Business    -5424.352131841959
Arts    -5427.4041545621685
Society    -5437.306500855186
Science    -5453.566123996552
Recreation    -5459.423570085095
Health    -5591.804845492097
Shopping    -5618.871690937458
Games    -5619.895131411285
Kids and Teens    -5661.5921294947075
Home    -5689.0415835673375
Reference    -5738.55817166406
News    -6023.204239757383
```

B. Personalized Results of T50G

Personalized Results of RIB based on created user profile for the 50 search results of Google when "jaguar" is the search term. The search results of Google are categorized into five different top-level ODP categories. The categories are re-arranged according to the order in the created user profile. Note that returned items about Sports and Computers are supposed to be of interest to user. The re-organized results are:

Sports (16):

```
sports_3^sports^arts^games
home_7^sports^computers^recreation
recreation_15^sports^recreation^business
sports_16^sports^arts^society
recreation_17^sports^recreation^shopping
recreation_18^sports^recreation^arts
recreation_19^sports^recreation^shopping
recreation_31^sports^recreation^arts
recreation_34^sports^recreation^society
recreation_35^arts^society^sports
reference_39^sports^recreation^kids and teens
recreation_42^sports^recreation^arts
recreation_43^sports^recreation^arts
recreation_44^science^sports^recreation
recreation_49^sports^recreation^society
home_50^sports^recreation^computers
```

Business (18):

```
science_4^business^arts^society
shopping_5^business^shopping^arts
science_6^business^arts^society
science_8^business^arts^shopping
recreation_9^business^science^computers
recreation_12^sports^recreation^business
shopping_20^business^society^arts
computers_22^business^science^computers
computers_23^business^computers^science
shopping_25^business^society^shopping
business_27^business^science^computers
shopping_29^business^shopping^society
games_30^business^computers^science
business_38^business^shopping^science
games_41^society^business^arts
recreation_46^computers^society^business
computers_47^business^computers^arts
business_48^business^arts^society
```

Arts (14):

recreation_1^arts^sports^recreation
kids and teens_2^arts^society^sports
kids and teens_11^arts^sports^games
recreation_13^arts^sports^recreation
shopping_21^arts^business^society
science_24^society^arts^sports
recreation_26^arts^society^sports
arts_28^arts^sports^home
kids and teens_32^arts^recreation^sports
science_33^arts^sports^recreation
home_36^shopping^business^arts
shopping_37^business^society^arts
reference_40^arts^sports^recreation
kids and teens_45^arts^sports^recreation

Society (1):

computers_10^society^health^arts

Recreation (1):

society_14^recreation^society^science

For each of the above items, for example, "society_14^recreation^society^science", society_14 indicates that the 14th ranked results of Google is assigned category "society" by Google. RIB categorized it into the category "Recreation", the number in the brackets indicates how many results are in this category. The three categories after the number 14 and starting with character '^' are the ranked list of categories for that the returned results 14 may be assigned to by RIB. In this case, the returned result is assigned category Recreation by RIB.

C. Personalized Results of T50O

Personalized Results of RIB based on created user profile for the 50 search results of ODP when "jaguar" is the search term. The search results of ODP are categorized into six different top-level ODP categories. The categories are re-arranged according to the order in the created user profile. The re-organized results are:

Sports (14):

recreation_1^sports^recreation^arts
shopping_3^shopping^business^sports
sports_10^sports^recreation^kids and teens
games_16^sports^recreation^games
games_18^sports^recreation^games
sports_20^sports^arts^society
arts_23^society^sports^recreation
recreation_25^recreation^sports^society
sports_36^sports^arts^business
arts_38^sports^recreation^society
recreation_40^computers^recreation^sports
sports_44^sports^arts^society
sports_48^sports^recreation^arts
recreation_50^sports^arts^recreation

Business (12):

games_2^business^computers^sports
kids and teens_4^business^arts^society
kids and teens_9^business^arts^society
society_14^business^sports^recreation
games_15^business^computers^society
business_19^business^shopping^computers

games_26^business^computers^shopping
shopping_27^business^society^arts
games_30^business^computers^science
science_33^business^arts^society
business_35^business^sports^shopping
shopping_42^business^shopping^arts

Arts (20):

sports_5^sports^society^arts
recreation_6^arts^business^society
shopping_8^arts^business^society
arts_11^arts^sports^society
home_13^arts^society^business
science_17^arts^society^sports
games_21^arts^sports^games
arts_22^arts^sports^recreation
games_24^arts^games^society
kids and teens_28^arts^society^business
arts_29^arts^sports^recreation
society_32^arts^science^society
games_34^arts^society^science
arts_37^arts^society^shopping
games_39^arts^games^computers
games_41^arts^games^sports
kids and teens_43^arts^sports^games
home_45^arts^society^science
science_46^arts^shopping^business
arts_49^recreation^sports^arts

Society (2):

games_12^society^arts^business
home_31^society^business^sports

Shopping (1):

business_47^shopping^business^arts

Games (1):

games_7^games^arts^sports

Appendix 7 Summary of Relevance Judgment

Each of the five judges are numbered from 1 to 5, their judgment for a given result is represented by their corresponding number in the column R, P, I, or N

Web snippet (W-S): the list of Web search results from Yahoo
R(3): the result is judged as relevant and it will contribute 3 scores
P(1): the result is judged as partially relevant and will contribute 1 score
I(-3): the results is judged as irrelevant and will contribute -3 scores
N(0): the result is not informative enough to make a decision, contribute score 0
Score (SC): a final relevance judgment score = 3*R + 1*P - 3*I, the score is calculated manually
auto score: the score is calculated based on the above formula
Judge (JG): a final binary relevant or irrelevant judgment decision. If score bigger than zero, it is relevant; a minus value indicates it is irrelevant; when score equals zero, human adjustment is needed after visit the website of the result.
RL: summarized number of relevant results at each of the relevant result
Rc: recall at each of the relevant results
pr: precision calculated at each of the relevant result
new rank (NR): the re-ranked results of RIB
Judge(JG): the relevance judgment results for re-ranked results of RIB
RL': summarized number of relevant results at each of the relevant result of RIB
Rc': recall at each of the relevant results of RIB
Pr': precision calculated at each of the relevant result of RIB

R-Lv: 10 recall level
Pr Yahoo: interpolated precision value at the different recall level for the results of Yahoo
Pr Bu: interpolated precision at the different recall level for results of RIB
JN: Judgment Number
AN: Agreement Number
JCD: Judgment Convergence Degree
Impr. Improvement of precision of RIB over Yahoo

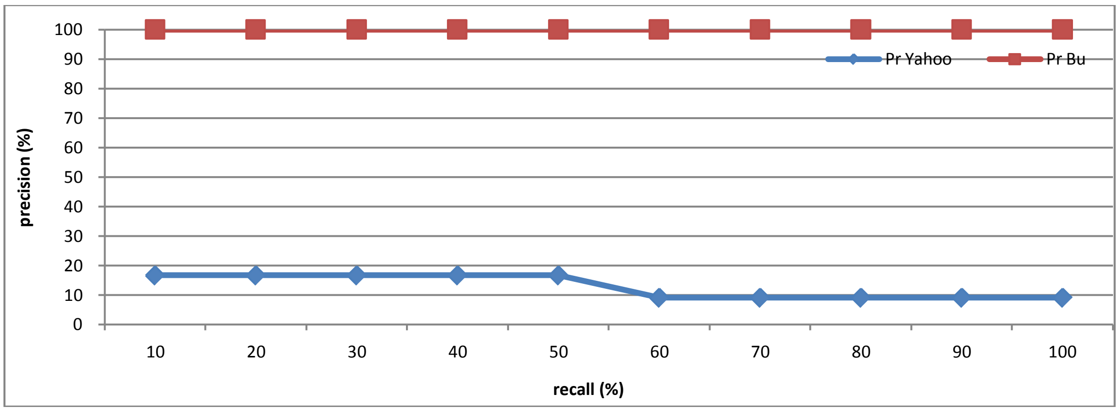
Search-term G1-1: apple; Info need: Information about fruit apple

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL’	Rc’	Pr’
1. Apple			12345		-15	0	0	0	0	6	1	1	0.5	1
2. Officia			12345		-15	0	0	0	0	22	1	2	1	1
3. Apple			12345		-15	0	0	0	0	1	0	0	0	0
4. AAPL		1	2345		-11	0	0	0	0	2	0	0	0	0
5. Apple			12345		-15	0	0	0	0	3	0	0	0	0
6. Apple	1235		4		9	1	1	0.5	0.1667	4	0	0	0	0
7. apple			12345		-15	0	0	0	0	5	0	0	0	0
8. TUA		3	1245		-11	0	0	0	0	7	0	0	0	0
9. AAPL			12345		-15	0	0	0	0	8	0	0	0	0
10. Apple			12345		-15	0	0	0	0	9	0	0	0	0
11. Apple			12345		-15	0	0	0	0	10	0	0	0	0
12. Enab			12345		-15	0	0	0	0	11	0	0	0	0
13 Apple			12345		-15	0	0	0	0	12	0	0	0	0
14. Apple			12345		-15	0	0	0	0	13	0	0	0	0
15. Newe			12345		-15	0	0	0	0	14	0	0	0	0
16. Apple			12345		-15	0	0	0	0	15	0	0	0	0
17. AAP			12345		-15	0	0	0	0	16	0	0	0	0
18. jobs			12345		-15	0	0	0	0	17	0	0	0	0
19. apple			12345		-15	0	0	0	0	18	0	0	0	0
20. Apple			12345		-15	0	0	0	0	19	0	0	0	0
21. Apple			12345		-15	0	0	0	0	20	0	0	0	0
22. Apple	1234		5		9	1	2	1	0.0909	21	0	0	0	0
23. Apple			12345		-15	0	0	0	0	23	0	0	0	0
24. Open			12345		-15	0	0	0	0	24	0	0	0	0
25. U.S			12345		-15	0	0	0	0	25	0	0	0	0
26. iPod			12345		-15	0	0	0	0	26	0	0	0	0
27. apple			12345		-15	0	0	0	0	27	0	0	0	0
28. Apple			12345		-15	0	0	0	0	28	0	0	0	0
29. Apple			12345		-15	0	0	0	0	29	0	0	0	0
30. Apple			12345		-15	0	0	0	0	30	0	0	0	0
31. Apple			12345		-15	0	0	0	0	31	0	0	0	0
32. Apple			12345		-15	0	0	0	0	32	0	0	0	0
33. Apple			12345		-15	0	0	0	0	33	0	0	0	0
34. Apple			12345		-15	0	0	0	0	34	0	0	0	0
35. Apple		3	1245		-11	0	0	0	0	35	0	0	0	0
36. Apple			12345		-15	0	0	0	0	36	0	0	0	0
37. Mac		2	1345		-11	0	0	0	0	37	0	0	0	0
38. AAP			12345		-15	0	0	0	0	38	0	0	0	0
39. Apple			12345		-15	0	0	0	0	39	0	0	0	0
40. Apple			12345		-15	0	0	0	0	40	0	0	0	0
41. Apple			12345		-15	0	0	0	0	41	0	0	0	0
42. Apple			12345		-15	0	0	0	0	42	0	0	0	0
43. Fiona			12345		-15	0	0	0	0	43	0	0	0	0
44. Apple			12345		-15	0	0	0	0	44	0	0	0	0
45. Apple			12345		-15	0	0	0	0	45	0	0	0	0
46. Apple			12345		-15	0	0	0	0	46	0	0	0	0
47. Apple			12345		-15	0	0	0	0	47	0	0	0	0
48. Text			12345		-15	0	0	0	0	48	0	0	0	0
49. Apple			12345		-15	0	0	0	0	49	0	0	0	0
50. Apple			1345	2	-12	0	0	0	0	50	0	0	0	0

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL’: number of relevant document in new ranked results, Rc’: recall of new results, Pr’: precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	16.67	16.67	16.67	16.67	16.67	9.1	9.1	9.1	9.1	9.1
Pr Bu	100	100	100	100	100	100	100	100	100	100

JN: 57; AN: 43; JCD: 0.7544; Impr: 87.115



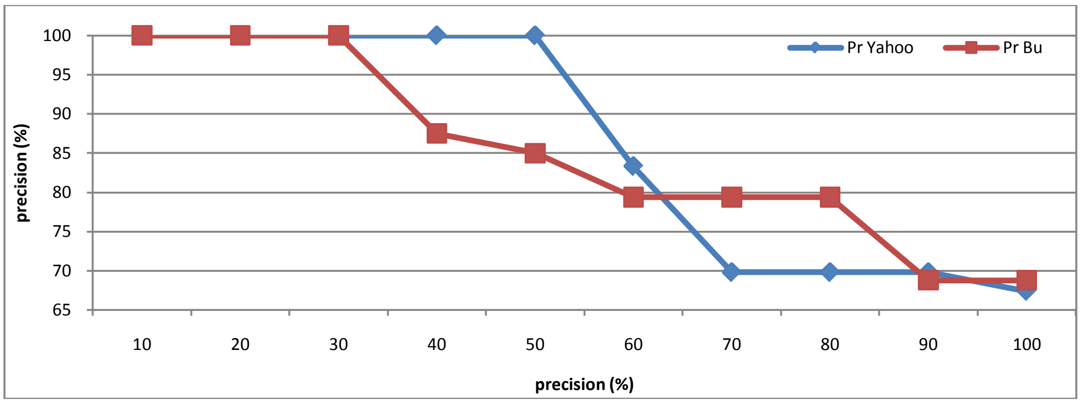
Search-term G1-2: Disney; Info need: the person water disney

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL’	Rc’	Pr’
1. Disney	12345				15	1	1	0.0303	1.0000	1	1	1	0.0303	1.0000
2. Disney	24	35	1		5	1	2	0.06061	1.0000	3	1	2	0.0606	1.0000
3. Disney	4	235	1		3	1	3	0.09091	1.0000	5	1	3	0.0909	1.0000
4. Disney	1234	5			13	1	4	0.12121	1.0000	6	1	4	0.1212	1.0000
5. Walt D	1234	5			13	1	5	0.15152	1.0000	7	1	5	0.1515	1.0000
6. Disney	12345				15	1	6	0.18182	1.0000	9	1	6	0.1818	1.0000
7. Walt D	234	5	1		7	1	7	0.21212	1.0000	13	1	7	0.2121	1.0000
8. The W	12345				15	1	8	0.24242	1.0000	15	1	8	0.2424	1.0000
9. Walt	12345				15	1	9	0.27273	1.0000	16	1	9	0.2727	1.0000
10. Disney	1245	3			13	1	10	0.30303	1.0000	22	1	10	0.303	1.0000
11. The Wal	124	35			11	1	11	0.33333	1.0000	23	0	0	0	0.0000
12. Mouse	1234		5		9	1	12	0.36364	1.0000	24	1	11	0.3333	0.9167
13. Laughin	124	3	5		7	1	13	0.39394	1.0000	30	0	0	0	0.0000
14. The Wa	124	3	5		7	1	14	0.42424	1.0000	35	1	12	0.3636	0.8571
15. JimHill	4	12	5	3	2	1	15	0.45455	1.0000	36	1	13	0.3939	0.8667
16. Playh	4	23	5	1	2	1	16	0.48485	1.0000	39	1	14	0.4242	0.8750
17. Bunny	2	35	4	1	2	1	17	0.51515	1.0000	44	0	0	0	0.0000
18. Disney	1345	2			13	1	18	0.54545	1.0000	49	1	15	0.4545	0.8333
19. Disney		125	4	3	-1	0	0	0	0.0000	4	1	16	0.4848	0.8421
20. Disney		235	14		-3	0	0	0	0.0000	17	1	17	0.5152	0.8500
21. Toon	23		145		-3	0	0	0	0.0000	25	1	18	0.5455	0.8571
22. DIS -	124	35			11	1	19	0.57576	0.8636	26	0	0	0	0.0000
23. Walt D	2	3	45	1	-2	0	0	0	0.0000	29	0	0	0	0.0000
24. Disney	125	34			11	1	20	0.60606	0.8333	31	0	0	0	0.0000
25. Disney	3	125	4		3	1	21	0.63636	0.8400	33	0	0	0	0.0000
26. Disney			12345		-15	0	0	0	0.0000	42	1	19	0.5758	0.7308
27. Disney		3	245	1	-8	0	0	0	0.0000	43	1	20	0.6061	0.7407
28. Disney	3	5	24	1	-2	0	0	0	0.0000	2	1	21	0.6364	0.7500
29. Year o		3	1245		-11	0	0	0	0.0000	8	1	22	0.6667	0.7586
30. Walt	4	3	25	1	-2	0	0	0	0.0000	10	1	23	0.697	0.7667
31. AllHe	3		24	15	-3	0	0	0	0.0000	11	1	24	0.7273	0.7742
32. Disney		3	124	5	-8	0	0	0	0.0000	12	1	25	0.7576	0.7813
33. Disney	23		145		-3	0	0	0	0.0000	14	1	26	0.7879	0.7879
34. Disney		23	145		-7	0	0	0	0.0000	18	1	27	0.8182	0.7941
35. Disney	12345				15	1	22	0.66667	0.6286	19	0	0	0	0.0000
36. Disney	23	1	45		1	1	23	0.69697	0.6389	20	0	0	0	0.0000
37. Disney	5	123	4		3	1	24	0.72727	0.6486	21	0	0	0	0.0000
38. Radio	234	15			11	1	25	0.75758	0.6579	27	0	0	0	0.0000
39. D23 |	12345				15	1	26	0.78788	0.6667	28	0	0	0	0.0000
40. Disney	23	45	1		5	1	27	0.81818	0.6750	32	0	0	0	0.0000
41. Amazo	123	45			11	1	28	0.84848	0.6829	34	0	0	0	0.0000
42. Disney	1235	4			13	1	29	0.87879	0.6905	37	1	28	0.8485	0.6667
43. Save u	1345	2			13	1	30	0.90909	0.6977	38	1	29	0.8788	0.6744
44. Disney		134	25		-3	0	0	0	0.0000	40	1	30	0.9091	0.6818
45. Disney	1	235	4		3	1	31	0.93939	0.6889	41	1	31	0.9394	0.6889
46. Disney	2		345	1	-6	0	0	0	0.0000	45	1	32	0.9697	0.6957
47. Radi	24	3	5	1	4	1	32	0.9697	0.6809	46	0	0	0	0
48. The W		23	145		-7	0	0	0	0.0000	47	1	33	1	0.6875
49. Disney	25	4	13		1	1	33	1	0.6735	48	0	0	0	0
50. Disney			1245	3	-12	0	0	0	0.0000	50	0	0	0	0

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL’: number of relevant document in new ranked results, Rc’: recall of new results, Pr’: precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	100	100	100	100	100	83.33	69.8	69.8	69.8	67.4
Pr Bu	100	100	100	87.5	85	79.4	79.4	79.4	68.8	68.8

JN: 123; AN: 7; JCD: 0.0569; Impr: -1.183



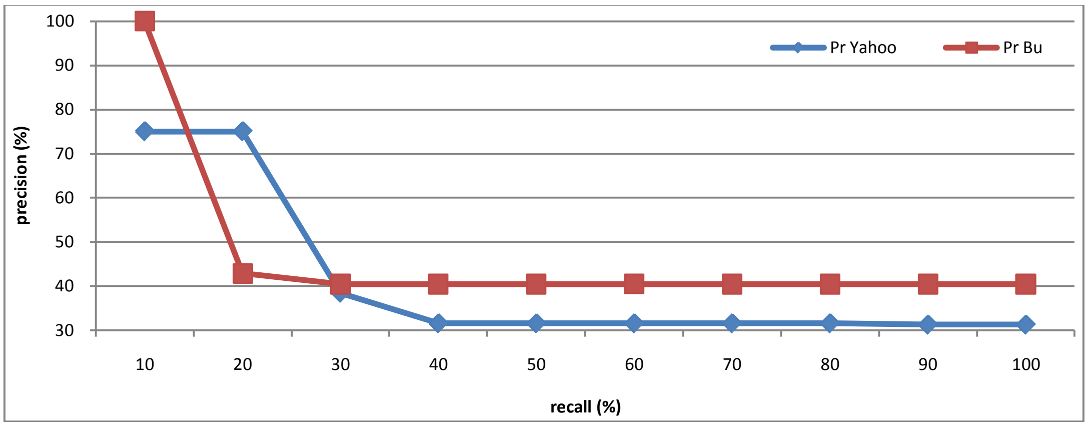
Search-term G1-3: iraq; Info need: geographic & demographical info of iraq

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL'	Rc'	Pr'
1. Iraqi	345		12		3	1	1	0.0667	1.0000	3	1	1	0.0667	1.0000
2. Iraq	3	1	245		-5	0	0	0.0000	0.0000	4	1	2	0.1333	1.0000
3. Iraq	1345	2			13	1	2	0.1333	0.6667	5	0	0	0.0000	0.0000
4. CIA	1345	2			13	1	3	0.2000	0.7500	7	0	0	0.0000	0.0000
5. Iraq		13	245		-7	0	0	0.0000	0.0000	8	0	0	0.0000	0.0000
6. Iraq	1345	2			13	1	4	0.2667	0.6667	9	0	0	0.0000	0.0000
7. Iraq N		345	12		-3	0	0	0.0000	0.0000	13	1	3	0.2000	0.4286
8. Iraq		3	1245		-11	0	0	0.0000	0.0000	14	0	0	0.0000	0.0000
9. The S			12345		-15	0	0	0.0000	0.0000	16	0	0	0.0000	0.0000
10. Iraq			12345		-15	0	0	0.0000	0.0000	17	0	0	0.0000	0.0000
11. Emba			12345		-15	0	0	0.0000	0.0000	18	0	0	0.0000	0.0000
12. Iraq			12345		-15	0	0	0.0000	0.0000	20	0	0	0.0000	0.0000
13. Iraq -	35	4	12		1	1	5	0.3333	0.3846	21	0	0	0.0000	0.0000
14. The Ir			12345		-15	0	0	0.0000	0.0000	22	0	0	0.0000	0.0000
15. Iraq			12345		-15	0	0	0.0000	0.0000	24	0	0	0.0000	0.0000
16. Iraq		5	1234		-11	0	0	0.0000	0.0000	26	0	0	0.0000	0.0000
17. Iraq		135	24		-3	0	0	0.0000	0.0000	27	1	4	0.2667	0.2353
18. Iraq	3		1245		-9	0	0	0.0000	0.0000	28	1	5	0.3333	0.2778
19. iraqi.	345		12		3	1	6	0.4000	0.3158	30	0	0	0.0000	0.0000
20. Iraq			12345		-15	0	0	0.0000	0.0000	34	0	0	0.0000	0.0000
21. War ry			12345		-15	0	0	0.0000	0.0000	35	0	0	0.0000	0.0000
22. Ame		1	2345		-11	0	0	0.0000	0.0000	36	1	6	0.4000	0.2727
23. Iraq		1	2345		-11	0	0	0.0000	0.0000	38	1	7	0.4667	0.3043
24. Iraq		3	1245		-11	0	0	0.0000	0.0000	43	0	0	0.0000	0.0000
25. www			12345		-15	0	0	0.0000	0.0000	48	1	8	0.5333	0.3200
26. Iraq			12345		-15	0	0	0.0000	0.0000	19	1	9	0.6000	0.3462
27. Iraq:	245		13		3	1	7	0.4667	0.2593	23	0	0	0.0000	0.0000
28. Iraq	345	12			11	1	8	0.5333	0.2857	25	0	0	0.0000	0.0000
29. Iraq	1245	3			13	1	9	0.6000	0.3103	29	1	10	0.6667	0.3448
30. Iraq	5	4	123		-5	0	0	0.0000	0.0000	32	0	0	0.0000	0.0000
31. Iraq:			1234	5	-12	0	0	0.0000	0.0000	33	1	11	0.7333	0.3548
32. FT.			12345		-15	0	0	0.0000	0.0000	39	1	12	0.8000	0.3750
33. Iraq	1245	3			13	1	10	0.6667	0.3030	46	0	0	0.0000	0.0000
34. Iraq			12345		-15	0	0	0.0000	0.0000	47	1	13	0.8667	0.3824
35.			12345		-15	0	0	0.0000	0.0000	1	1	14	0.9333	0.4000
36. Iraq	1345	2			13	1	11	0.7333	0.3056	2	0	0	0.0000	0.0000
37. Welcome		3	1245		-11	0	0	0.0000	0.0000	6	1	15	1.0000	0.4054
38. Iraq	13	245			9	1	12	0.8000	0.3158	10	0	0	0.0000	0.0000
39. Iraq -.	345	2	1		7	1	13	0.8667	0.3333	11	0	0	0.0000	0.0000
40. Iraq War:			12345		-15	0	0	0.0000	0.0000	12	0	0	0.0000	0.0000
41. IRAQ			1345	2	-12	0	0	0.0000	0.0000	15	0	0	0.0000	0.0000
42. Post-war			12345		-15	0	0	0.0000	0.0000	31	0	0	0.0000	0.0000
43. Iraq War:			12345		-15	0	0	0.0000	0.0000	37	0	0	0.0000	0.0000
44. Iraq	5	34	12		-1	0	0	0.0000	0.0000	40	0	0	0.0000	0.0000
45. Iraq			12345		-15	0	0	0.0000	0.0000	41	0	0	0.0000	0.0000
46. albawab			12345		-15	0	0	0.0000	0.0000	42	0	0	0.0000	0.0000
47. Map of	1345	2			13	1	14	0.9333	0.2979	44	0	0	0.0000	0.0000
48. Iraq -	345		1	2	6	1	15	1.0000	0.3125	45	0	0	0.0000	0.0000
49. Iraq			12345		-15	0	0	0.0000	0.0000	49	0	0	0.0000	0.0000
50. Iraq :		5	1234		-11	0	0	0.0000	0.0000	50	0	0	0.0000	0.0000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL': number of relevant document in new ranked results, Rc': recall of new results, Pr': precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	75	75	38.5	31.6	31.6	31.6	31.6	31.6	31.3	31.3
Pr Bu	100	42.9	40.5	40.5	40.5	40.55	40.5	40.5	40.5	40.5

JN: 87; AN: 19; JCD: 0.2184; Impr: 5.785



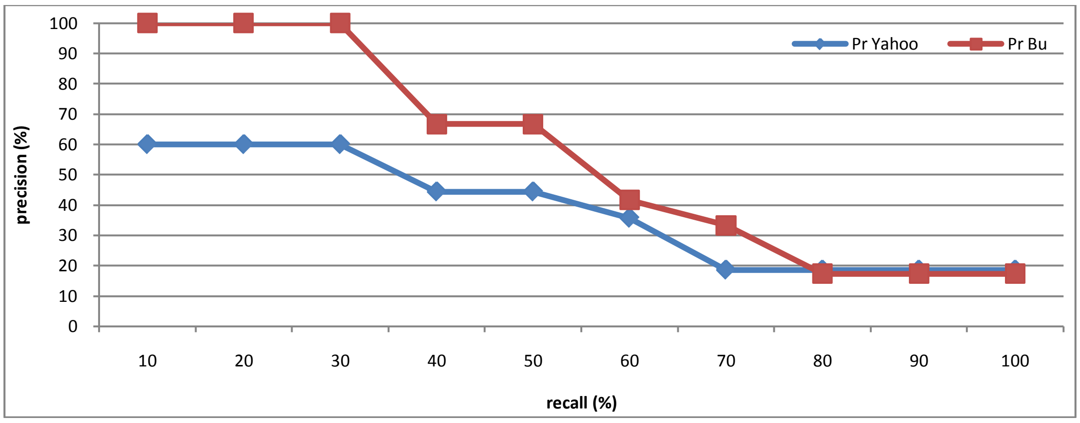
Search-term G1-4: matrix; Info need: the mathematics concept matrix

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL’	Rc’	Pr’
1. Matrix			12345		-15	0	0	0.0000	0.0000	3	1	1	0.1250	1.0000
2. The Matrix			12345		-15	0	0	0.0000	0.0000	4	1	2	0.2500	1.0000
3. The Matrix	12345				15	1	1	0.1250	0.3333	5	1	3	0.3750	1.0000
4. 2009	1345	2			13	1	2	0.2500	0.5000	10	0	0	0.0000	0.0000
5. The Matrix	12345				15	1	3	0.3750	0.6000	29	0	0	0.0000	0.0000
6. Matrix -			12345		-15	0	0	0.0000	0.0000	38	1	4	0.5000	0.6667
7. The Matrix			12345		-15	0	0	0.0000	0.0000	44	0	0	0.0000	0.0000
8. Matrix			1245	3	-12	0	0	0.0000	0.0000	47	0	0	0.0000	0.0000
9. The Matrix	1345	2			13	1	4	0.5000	0.4444	50	0	0	0.0000	0.0000
10. matrix:			4	1235	-3	0	0	0.0000	0.0000	6	0	0	0.0000	0.0000
11. MATRIX			12345		-15	0	0	0.0000	0.0000	8	0	0	0.0000	0.0000
12. Matrix		3	1245		-11	0	0	0.0000	0.0000	14	1	5	0.6250	0.4167
13. Used			12345		-15	0	0	0.0000	0.0000	25	0	0	0.0000	0.0000
14. Matrix	1234	5			13	1	5	0.6250	0.3571	32	0	0	0.0000	0.0000
15. Toyota			12345		-15	0	0	0.0000	0.0000	1	0	0	0.0000	0.0000
16. The Matri			12345		-15	0	0	0.0000	0.0000	2	0	0	0.0000	0.0000
17. MLS.			12345		-15	0	0	0.0000	0.0000	7	0	0	0.0000	0.0000
18. The			12345		-15	0	0	0.0000	0.0000	9	1	6	0.7500	0.3333
19. New			12345		-15	0	0	0.0000	0.0000	11	0	0	0.0000	0.0000
20. Matrix			12345		-15	0	0	0.0000	0.0000	12	0	0	0.0000	0.0000
21. Matrix			12345		-15	0	0	0.0000	0.0000	13	0	0	0.0000	0.0000
22. MATRIX			12345		-15	0	0	0.0000	0.0000	15	0	0	0.0000	0.0000
23. matrix -			12345		-15	0	0	0.0000	0.0000	16	0	0	0.0000	0.0000
24. Matrix:			1234	5	-12	0	0	0.0000	0.0000	17	0	0	0.0000	0.0000
25. Matrix			12345		-15	0	0	0.0000	0.0000	18	0	0	0.0000	0.0000
26. Kelley			12345		-15	0	0	0.0000	0.0000	19	0	0	0.0000	0.0000
27. matrix			12345		-15	0	0	0.0000	0.0000	20	0	0	0.0000	0.0000
28. Pantech			12345		-15	0	0	0.0000	0.0000	21	0	0	0.0000	0.0000
29. 2009	5		1234		-9	0	0	0.0000	0.0000	22	0	0	0.0000	0.0000
30. matrix -			12345		-15	0	0	0.0000	0.0000	23	0	0	0.0000	0.0000
31. Astrology		235	14		-3	0	0	0.0000	0.0000	24	0	0	0.0000	0.0000
32. Matrix			12345		-15	0	0	0.0000	0.0000	26	0	0	0.0000	0.0000
33. Matrix			12345		-15	0	0	0.0000	0.0000	27	0	0	0.0000	0.0000
34. The Matr			12345		-15	0	0	0.0000	0.0000	28	0	0	0.0000	0.0000
35. Harmonic			12345		-15	0	0	0.0000	0.0000	30	0	0	0.0000	0.0000
36. The			12345		-15	0	0	0.0000	0.0000	31	0	0	0.0000	0.0000
37. MatrixOn			12345		-15	0	0	0.0000	0.0000	33	0	0	0.0000	0.0000
38. Buy Toy	2345		1		9	1	6	0.7500	0.1579	34	0	0	0.0000	0.0000
39. The	1		2345		-9	0	0	0.0000	0.0000	35	0	0	0.0000	0.0000
40. matrix -	35		124		-3	0	0	0.0000	0.0000	36	0	0	0.0000	0.0000
41. ExamMat	1345	2			13	1	7	0.8750	0.1707	37	0	0	0.0000	0.0000
42. Matrix			12345		-15	0	0	0.0000	0.0000	39	0	0	0.0000	0.0000
43. splash	345	1	2		7	1	8	1.0000	0.1860	40	0	0	0.0000	0.0000
44. American			124	35	-9	0	0	0.0000	0.0000	41	1	7	0.8750	0.1591
45. Matrix	3		1245		-9	0	0	0.0000	0.0000	42	0	0	0.0000	0.0000
46. Matrix		13	24	5	-4	0	0	0.0000	0.0000	43	1	8	1.0000	0.1739
47. Welcome		5	234	1	-8	0	0	0.0000	0.0000	45	0	0	0.0000	0.0000
48. ITA.			24	135	-6	0	0	0.0000	0.0000	46	0	0	0.0000	0.0000
49. Toyota			12345		-15	0	0	0.0000	0.0000	48	0	0	0.0000	0.0000
50. Matrix		1	2345		-11	0	0	0.0000	0.0000	49	0	0	0.0000	0.0000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL’: number of relevant document in new ranked results, Rc’: recall of new results, Pr’: precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	60	60	60	44.4	44.4	35.7	18.6	18.6	18.6	18.6
Pr Bu	100	100	100	66.7	66.7	41.7	33.3	17.4	17.4	17.4

JN: 73; AN: 30; JCD: 0.411; Impr: 18.17



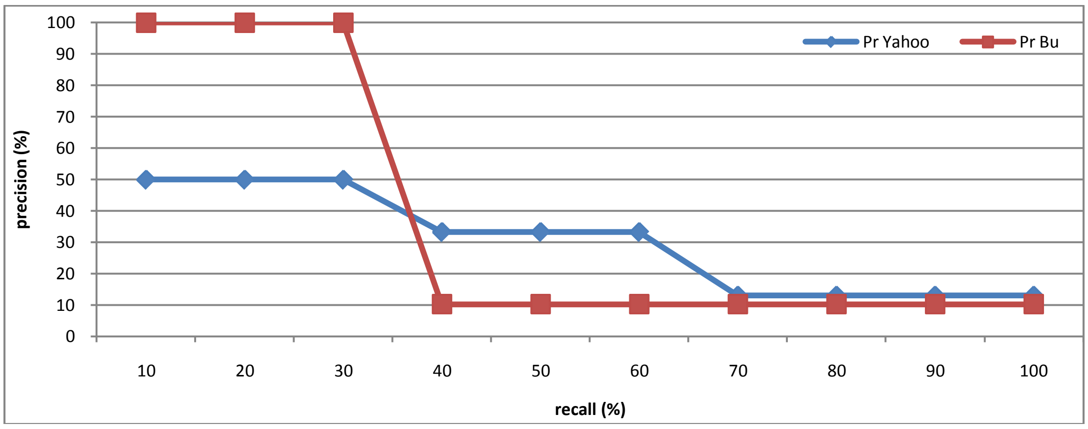
Search-term G1-5: radio; Info need: history of radio

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL'	Rc'	Pr'
1. Online			12345		-15	0	0	0.0000	0.0000	2	1	1	0.3333	1.0000
2. Radio -	12345				15	1	1	0.3333	0.5000	3	0	0	0.0000	0.0000
3. AOL			12345		-15	0	0	0.0000	0.0000	5	0	0	0.0000	0.0000
4. Radio-			12345		-15	0	0	0.0000	0.0000	8	0	0	0.0000	0.0000
5. Pandora			12345		-15	0	0	0.0000	0.0000	10	0	0	0.0000	0.0000
6. radio:	5	234	1		3	1	2	0.6667	0.3333	12	0	0	0.0000	0.0000
7. Radio (			134	25	-9	0	0	0.0000	0.0000	14	0	0	0.0000	0.0000
8.			12345		-15	0	0	0.0000	0.0000	16	0	0	0.0000	0.0000
9. Radio - C			12345		-15	0	0	0.0000	0.0000	22	0	0	0.0000	0.0000
10. Real			12345		-15	0	0	0.0000	0.0000	24	0	0	0.0000	0.0000
11. Radio -			12345		-15	0	0	0.0000	0.0000	26	0	0	0.0000	0.0000
12. Radio -			12345		-15	0	0	0.0000	0.0000	27	0	0	0.0000	0.0000
13. iRadio.			12345		-15	0	0	0.0000	0.0000	28	0	0	0.0000	0.0000
14. Radio			12345		-15	0	0	0.0000	0.0000	36	0	0	0.0000	0.0000
15. NPR :			12345		-15	0	0	0.0000	0.0000	39	0	0	0.0000	0.0000
16. Sirius://			12345		-15	0	0	0.0000	0.0000	41	0	0	0.0000	0.0000
17. FOX Ne.			12345		-15	0	0	0.0000	0.0000	44	0	0	0.0000	0.0000
18. SIRIUS			12345		-15	0	0	0.0000	0.0000	46	0	0	0.0000	0.0000
19. Web-			12345		-15	0	0	0.0000	0.0000	9	0	0	0.0000	0.0000
20.			12345		-15	0	0	0.0000	0.0000	15	0	0	0.0000	0.0000
21. XM			12345		-15	0	0	0.0000	0.0000	21	0	0	0.0000	0.0000
22. Live-.			12345		-15	0	0	0.0000	0.0000	23	1	2	0.6667	0.0909
23. Radio	245	3	1		7	1	3	1.0000	0.1304	25	0	0	0.0000	0.0000
24. Free		5	1234		-11	0	0	0.0000	0.0000	30	0	0	0.0000	0.0000
25.			1345	2	-12	0	0	0.0000	0.0000	34	0	0	0.0000	0.0000
26. XMRadio			1	2345	-3	0	0	0.0000	0.0000	37	0	0	0.0000	0.0000
27. Radio (	3	1	245		-5	0	0	0.0000	0.0000	1	0	0	0.0000	0.0000
28. Alice@			125	34	-9	0	0	0.0000	0.0000	4	0	0	0.0000	0.0000
29. RadioSha			12345		-15	0	0	0.0000	0.0000	6	1	3	1.0000	0.1034
30. AARP			12345		-15	0	0	0.0000	0.0000	7	0	0	0.0000	0.0000
31. wsRadio			12345		-15	0	0	0.0000	0.0000	11	0	0	0.0000	0.0000
32. ABC.			12345		-15	0	0	0.0000	0.0000	13	0	0	0.0000	0.0000
33. CBC.ca -			12345		-15	0	0	0.0000	0.0000	17	0	0	0.0000	0.0000
34. U.S. Radi			12345		-15	0	0	0.0000	0.0000	18	0	0	0.0000	0.0000
35. Radio			12345		-15	0	0	0.0000	0.0000	19	0	0	0.0000	0.0000
36. ESPNR			12345		-15	0	0	0.0000	0.0000	20	0	0	0.0000	0.0000
37. Talk			12345		-15	0	0	0.0000	0.0000	29	0	0	0.0000	0.0000
38. 97.1			12345		-15	0	0	0.0000	0.0000	31	0	0	0.0000	0.0000
39. CMT :			12345		-15	0	0	0.0000	0.0000	32	0	0	0.0000	0.0000
40. NPR: All			12345		-15	0	0	0.0000	0.0000	33	0	0	0.0000	0.0000
41. ksl.com -			12345		-15	0	0	0.0000	0.0000	35	0	0	0.0000	0.0000
42. Radio			12345		-15	0	0	0.0000	0.0000	38	0	0	0.0000	0.0000
43. Oprah			12345		-15	0	0	0.0000	0.0000	40	0	0	0.0000	0.0000
44. RadioJa			12345		-15	0	0	0.0000	0.0000	42	0	0	0.0000	0.0000
45. MSN.			12345		-15	0	0	0.0000	0.0000	43	0	0	0.0000	0.0000
46. Free			12345		-15	0	0	0.0000	0.0000	45	0	0	0.0000	0.0000
47. RadioSt			12345		-15	0	0	0.0000	0.0000	47	0	0	0.0000	0.0000
48. Web-			12345		-15	0	0	0.0000	0.0000	48	0	0	0.0000	0.0000
49. Howstuff			1345	2	-12	0	0	0.0000	0.0000	49	0	0	0.0000	0.0000
50. ABC.net.			12345		-15	0	0	0.0000	0.0000	50	0	0	0.0000	0.0000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL': number of relevant document in new ranked results, Rc': recall of new results, Pr': precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	50	50	50	33.3	33.3	33.3	13	13	13	13
Pr Bu	100	100	100	10.3	10.3	10.3	10.3	10.3	10.3	10.3

JN: 62; AN: 41; JCD: 0.6613; Impr: 7.02



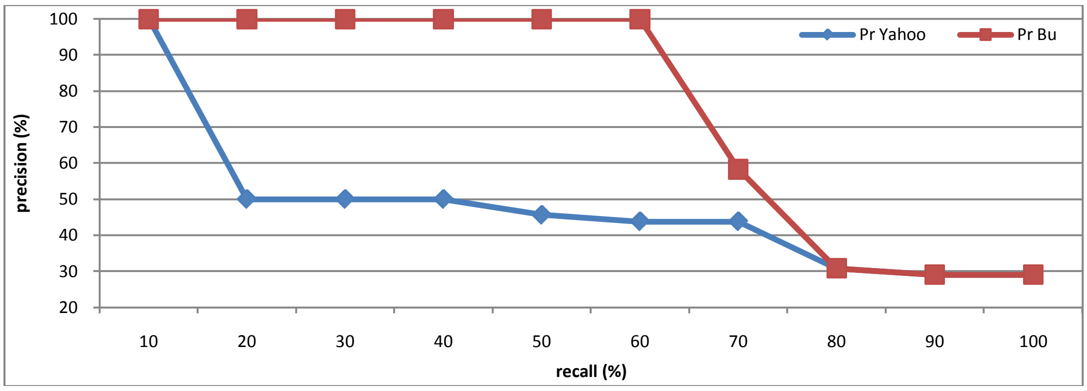
Search-term G2-1: flower; Info need: wild flower

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL'	Rc'	Pr'
1. Flower -	14	235			9	1	1	0.1111	1.0000	1	1	1	0.1111	1.0000
2. 1-800-			12345		-15	0	0	0.0000	0.0000	4	1	2	0.2222	1.0000
3. ProFlower.			12345		-15	0	0	0.0000	0.0000	7	1	3	0.3333	1.0000
4. flower:		12345			5	1	2	0.2222	0.5000	11	1	4	0.4444	1.0000
5. Flowers.vg		2	1345		-11	0	0	0.0000	0.0000	14	1	5	0.5556	1.0000
6. Flower	2	45	13		-1	0	0	0.0000	0.0000	16	1	6	0.6667	1.0000
7. Flower -		1345	2		1	1	3	0.3333	0.4286	19	0	0	0.0000	0.0000
8. Types Of	1245	3			13	1	4	0.4444	0.5000	2	0	0	0.0000	0.0000
9. Rhode		1	245	3	-8	0	0	0.0000	0.0000	3	0	0	0.0000	0.0000
10. Flowers.		2	1345		-11	0	0	0.0000	0.0000	5	0	0	0.0000	0.0000
11. FLOWER		135	4	2	1	1	5	0.5556	0.4545	6	0	0	0.0000	0.0000
12. PlaySta.			12345		-15	0	0	0.0000	0.0000	8	1	7	0.7778	0.5833
13. French			12345		-15	0	0	0.0000	0.0000	9	0	0	0.0000	0.0000
14 Flower |		123	4	5	1	1	6	0.6667	0.4286	10	0	0	0.0000	0.0000
15. Austin			12345		-15	0	0	0.0000	0.0000	12	0	0	0.0000	0.0000
16. Flower -		12345			5	1	7	0.7778	0.4375	13	0	0	0.0000	0.0000
17. Flickr:		2	1345		-11	0	0	0.0000	0.0000	15	0	0	0.0000	0.0000
18. Diane's		3	1245		-11	0	0	0.0000	0.0000	17	0	0	0.0000	0.0000
19. Flower	4	35	12		-1	0	0	0.0000	0.0000	18	0	0	0.0000	0.0000
20. Flower		24	13	5	-4	0	0	0.0000	0.0000	20	0	0	0.0000	0.0000
21. Sunflow			12345		-15	0	0	0.0000	0.0000	21	0	0	0.0000	0.0000
22. Cut			1234	5	-12	0	0	0.0000	0.0000	22	0	0	0.0000	0.0000
23. Martin's			12345		-15	0	0	0.0000	0.0000	23	0	0	0.0000	0.0000
24. FLOWER		5	1234		-11	0	0	0.0000	0.0000	24	0	0	0.0000	0.0000
25. Hialeah			12345		-15	0	0	0.0000	0.0000	25	0	0	0.0000	0.0000
26. Official	245	13			11	1	8	0.8889	0.3077	26	1	8	0.8889	0.3077
27. Fuquay-			12345		-15	0	0	0.0000	0.0000	27	0	0	0.0000	0.0000
28 Pictures of	4	25	13		-1	0	0	0.0000	0.0000	28	0	0	0.0000	0.0000
29. Flowerb			12345		-15	0	0	0.0000	0.0000	29	0	0	0.0000	0.0000
30. Find a		5	1234		-11	0	0	0.0000	0.0000	30	0	0	0.0000	0.0000
31. flower -		12345			5	1	9	1.0000	0.2903	31	1	9	1.0000	0.2903
32. . FTD.c			12345		-15	0	0	0.0000	0.0000	32	0	0	0.0000	0.0000
33. Find			12345		-15	0	0	0.0000	0.0000	33	0	0	0.0000	0.0000
34. . Dearbo			12345		-15	0	0	0.0000	0.0000	34	0	0	0.0000	0.0000
35. Flower			12345		-15	0	0	0.0000	0.0000	35	0	0	0.0000	0.0000
36. San			12345		-15	0	0	0.0000	0.0000	36	0	0	0.0000	0.0000
37. Florist in			12345		-15	0	0	0.0000	0.0000	37	0	0	0.0000	0.0000
38. Flowers			12345		-15	0	0	0.0000	0.0000	38	0	0	0.0000	0.0000
39. Baltimore			12345		-15	0	0	0.0000	0.0000	39	0	0	0.0000	0.0000
40. Los			12345		-15	0	0	0.0000	0.0000	40	0	0	0.0000	0.0000
41. flower -		5	124	3	-8	0	0	0.0000	0.0000	41	0	0	0.0000	0.0000
42. Flower		235	14		-3	0	0	0.0000	0.0000	42	0	0	0.0000	0.0000
43. Send			12345		-15	0	0	0.0000	0.0000	43	0	0	0.0000	0.0000
44. Flower			12345		-15	0	0	0.0000	0.0000	44	0	0	0.0000	0.0000
45. Flower			12345		-15	0	0	0.0000	0.0000	45	0	0	0.0000	0.0000
46. Milwauke			12345		-15	0	0	0.0000	0.0000	46	0	0	0.0000	0.0000
47. Singapore			12345		-15	0	0	0.0000	0.0000	47	0	0	0.0000	0.0000
48. MY FLO			12345		-15	0	0	0.0000	0.0000	48	0	0	0.0000	0.0000
49. Flower			12345		-15	0	0	0.0000	0.0000	49	0	0	0.0000	0.0000
50. Welcome		25	134		-7	0	0	0.0000	0.0000	50	0	0	0.0000	0.0000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL': number of relevant document in new ranked results, Rc': recall of new results, Pr': precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	100	50	50	50	45.6	43.8	43.8	30.8	29	29
Pr Bu	100	100	100	100	100	100	58.3	30.8	29	29

JN: 79; AN: 29; JCD: 0.3671; Impr: 27.51



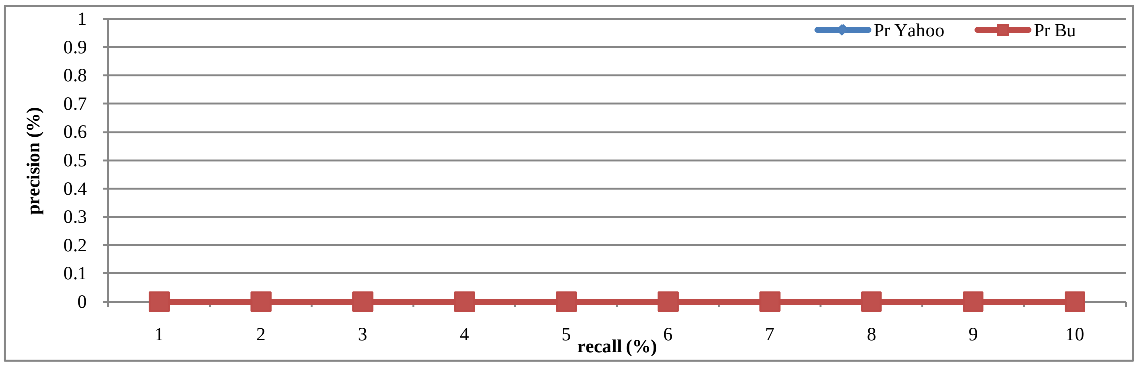
Search-term G2-2: jobs; Info need: the person Steve Jobs

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL’	Rc’	Pr’
1. Jobs.com.			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
2. I-JOBS -			1245	3	-12	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
3. CareerBui			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
4. Job Search		3	1245		-11	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
5. Monster.			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
6. SnagAJob			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
7. USAJOBS			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
8. Job Search			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
9. . Jobs			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
10. Job		3	1245		-11	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
11. SFGate			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
12. Jobs : Job			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
13. Alaska			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
14. Jobs | one			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
15. Search			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
16. FlipDog.			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
17. GO Jobs			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
18. NIH -			1245	3	-12	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
19. JobsInRI.			1245	3	-12	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
20. JobsIn			1245	3	-12	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
21. AJC Jobs			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
22. Guardian			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
23. USPS -			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
24. USAJOB			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
25. Shine.			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
26. City of P			1234	5	-12	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
27. Jobs at			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
28. Trades			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
29. Search			1245	3	-12	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
30. Jobs -			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
31. Australia			1245	3	-12	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
32. Search			1245	3	-12	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
33. Search			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
34. Bdjobs.co			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
35. SacJobs.c			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
36. Search			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
37. Minnesot			1245	3	-12	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
38. Search		3	1245		-11	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
39. South		3	1245		-11	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
40. FEMA -			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
41. FortWayn			1245	3	-12	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
42. Jobs |		5	124	3	-8	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
43. Search			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
44. Browse			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
45. Chicago			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
46. Beyond.c			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
47. Search			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
48 Raytheon			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
49. Jobs At			12345		-15	0	0	0.0000	0.0000	0	0	0	0.0000	0.0000
50. Monster.c			12345		-15	0	0		0.0000	0	0	0	0.0000	0.0000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL’: number of relevant document in new ranked results, Rc’: recall of new results, Pr’: precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	0	0	0	0	0	0	0	0	0	0
Pr Bu	0	0	0	0	0	0	0	0	0	0

JN: 66; AN: 35; JCD: 0.5303; Impr: 0



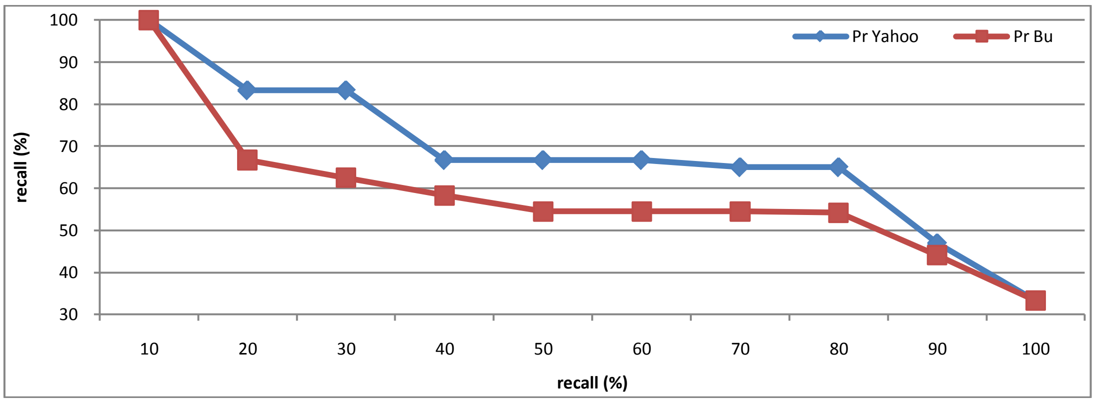
Search-term G2-3: maps; Info need: How to read maps?

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL’	Rc’	Pr’
1. Yahoo!	235		14		3	1	1	0.0625	1.0000	1	1	1	0.0625	1.0000
2. Google	235		14		3	1	2	0.1250	1.0000	3	1	2	0.1250	1.0000
3. MapQuest	235		14		3	1	3	0.1875	1.0000	4	0	0	0.0000	0.0000
4. Maps.com		235	14		-3	0	0	0.0000	0.0000	7	0	0	0.0000	0.0000
5. National	35	2	14		1	1	4	0.2500	0.8000	13	1	3	0.1875	0.6000
6. Map		1235	4		1	1	5	0.3125	0.8333	15	1	4	0.2500	0.6667
7. Maps.	5	23	14		-1	0	0	0.0000	0.0000	18	0	0	0.0000	0.0000
8. eMapsPlus	5	23	14		-1	0	0	0.0000	0.0000	20	1	5	0.3125	0.6250
9 Maps		235	14		-3	0	0	0.0000	0.0000	25	0	0	0.0000	0.0000
10. Multidisc		3	1245		-11	0	0	0.0000	0.0000	26	0	0	0.0000	0.0000
11. Map -	15	23	4		5	1	6	0.3750	0.5455	2	1	6	0.3750	0.5455
12. Bing	35	2	14		1	1	7	0.4375	0.5833	5	1	7	0.4375	0.5833
13 MSN	235		14		3	1	8	0.5000	0.6154	35	0	0	0.0000	0.0000
14. Maps	235		14		3	1	9	0.5625	0.6429	46	0	0	0.0000	0.0000
15. Expedia	35	2	14		1	1	10	0.6250	0.6667	6	1	8	0.5000	0.5333
16. Google	235		14		3	1	11	0.6875	0.6875	8	0	0	0.0000	0.0000
17. Virginia		3	1245		-11	0	0	0.0000	0.0000	9	0	0	0.0000	0.0000
18. Ask.com	3	2	145		-5	0	0	0.0000	0.0000	10	0	0	0.0000	0.0000
19. Maps	12345				15	1	12	0.7500	0.6316	11	1	9	0.5625	0.4737
20. Maps	35	2	14		1	1	13	0.8125	0.6500	12	1	10	0.6250	0.5000
21. Maps and		35	124		-7	0	0	0.0000	0.0000	14	1	11	0.6875	0.5238
22. Free	3	25	14		-1	0	0	0.0000	0.0000	16	1	12	0.7500	0.5455
23. Quick	35		124		-3	0	0	0.0000	0.0000	17	0	0	0.0000	0.0000
24. Google	3	25	14		-1	0	0	0.0000	0.0000	19	1	13	0.8125	0.5417
25. Maps and	3	25	14		-1	0	0	0.0000	0.0000	21	0	0	0.0000	0.0000
26. Rand		23	145		-7	0	0	0.0000	0.0000	22	0	0	0.0000	0.0000
27. Google		2	14	35	-5	0	0	0.0000	0.0000	23	0	0	0.0000	0.0000
28. MapQue		1		2345	1	1	14	0.8750	0.5000	24	0	0	0.0000	0.0000
29. Gmaps	1	35	24		-1	0	0	0.0000	0.0000	27	0	0	0.0000	0.0000
30. A-Maps		35	124		-7	0	0	0.0000	0.0000	28	1	14	0.8750	0.4667
31. Maps -		3	1245		-11	0	0	0.0000	0.0000	29	0	0	0.0000	0.0000
32. Explore	125	3	4		7	1	15	0.9375	0.4688	30	0	0	0.0000	0.0000
33. AAAma			14	235	-6	0	0	0.0000	0.0000	31	0	0	0.0000	0.0000
34. maps.go			14	235	-6	0	0	0.0000	0.0000	32	1	15	0.9375	0.4412
35. FEMA:	5	23	14		-1	0	0	0.0000	0.0000	33	0	0	0.0000	0.0000
36. MAPS -		3	1245		-11	0	0	0.0000	0.0000	34	0	0	0.0000	0.0000
37. Find ma		3	15	24	-5	0	0	0.0000	0.0000	36	0	0	0.0000	0.0000
38. Google	2	3	14	5	-2	0	0	0.0000	0.0000	37	0	0	0.0000	0.0000
39. BING		3	1245		-11	0	0	0.0000	0.0000	38	0	0	0.0000	0.0000
40. US Maps		235	14		-3	0	0	0.0000	0.0000	39	0	0	0.0000	0.0000
41. UCSD	3	2	145		-5	0	0	0.0000	0.0000	40	0	0	0.0000	0.0000
42. TomTom		35	14	2	-4	0	0	0.0000	0.0000	41	0	0	0.0000	0.0000
43. Your web		3	145	2	-8	0	0	0.0000	0.0000	42	0	0	0.0000	0.0000
44. Interactiv		35	124		-7	0	0	0.0000	0.0000	43	0	0	0.0000	0.0000
45. Maps	2		4	135	-1	0	0	0.0000	0.0000	44	0	0	0.0000	0.0000
46. National	3	25	14		-1	0	0	0.0000	0.0000	45	0	0	0.0000	0.0000
47. Maps of	3	25	14		-1	0	0	0.0000	0.0000	47	0	0	0.0000	0.0000
48. Open	23	5	14		1	1	16	1.0000	0.3333	48	1	16	1.0000	0.3333
49. BING	3	2	145		-5	0	0	0.0000	0.0000	49	0	0	0.0000	0.0000
50. Quick	3	25	14		-1	0	0	0.0000	0.0000	50	0	0	0.0000	0.0000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL’: number of relevant document in new ranked results, Rc’: recall of new results, Pr’: precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	100	83.3	83.3	66.7	66.7	66.7	65	65	46.9	33.3
Pr Bu	100	66.7	62.5	58.3	54.5	54.5	54.5	54.2	44.1	33.3

JN: 126; AN: 1; JCD: 0.0079; Impr: -9.43



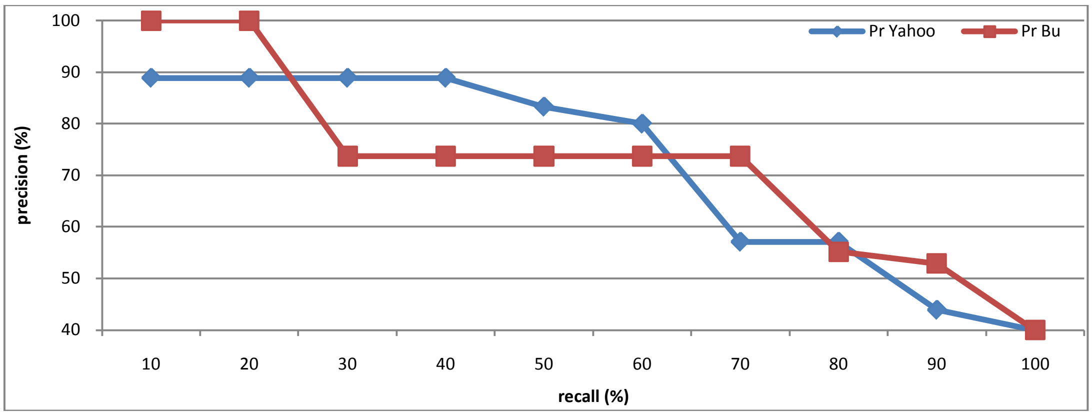
Search-term G2-4: Saturn; Info need: the planet Saturn

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL’	Rc’	Pr’
1. Saturn			12345		-15	0	0	0.0000	0.0000	6	1	1	0.0500	1.0000
2. Saturn -	12345				15	1	1	0.0500	0.5000	9	1	2	0.1000	1.0000
3. Saturn -	12345				15	1	2	0.1000	0.6667	12	1	3	0.1500	1.0000
4. The Saturn	2345		1		9	1	3	0.1500	0.7500	14	1	4	0.2000	1.0000
5. Saturn -	12345				15	1	4	0.2000	0.8000	18	0	0	0.0000	0.0000
6. Saturn's	12345				15	1	5	0.2500	0.8333	23	1	5	0.2500	0.8333
7. Saturn -	12345				15	1	6	0.3000	0.8571	36	0	0	0.0000	0.0000
8. Views of	12345				15	1	7	0.3500	0.8750	44	0	0	0.0000	0.0000
9. Saturn	35	12	4		5	1	8	0.4000	0.8889	8	1	6	0.3000	0.6667
10. Saturn			12345		-15	0	0	0.0000	0.0000	15	1	7	0.3500	0.7000
11 NASA -	1234	5			13	1	9	0.4500	0.8182	24	0	0	0.0000	0.0000
12. Solar	1234	5			13	1	10	0.5000	0.8333	27	1	8	0.4000	0.6667
13. Saturn |			12345		-15	0	0	0.0000	0.0000	49	1	9	0.4500	0.6923
14. Saturn:	245	13			11	1	11	0.5500	0.7857	1	0	0	0.0000	0.0000
15. Cassini	1245	3			13	1	12	0.6000	0.8000	2	1	10	0.5000	0.6667
16 Saturn			12345		-15	0	0	0.0000	0.0000	3	1	11	0.5500	0.6875
17. Cassini-	1	23	45		-1	0	0	0.0000	0.0000	4	1	12	0.6000	0.7059
18. Saturn:		3	1245		-11	0	0	0.0000	0.0000	5	1	13	0.6500	0.7222
19. Saturn			12345		-15	0	0	0.0000	0.0000	7	1	14	0.7000	0.7368
20. Saturn of			12345		-15	0	0	0.0000	0.0000	10	0	0	0.0000	0.0000
21. Saturn -			12345		-15	0	0	0.0000	0.0000	11	1	15	0.7500	0.7143
22. Saturn of			12345		-15	0	0	0.0000	0.0000	13	0	0	0.0000	0.0000
23. Explorat	24	13	5		5	1	13	0.6500	0.5652	16	0	0	0.0000	0.0000
24. Saturn		3	45	12	-5	0	0	0.0000	0.0000	17	0	0	0.0000	0.0000
25. Saturn -	24	13	5		5	1	14	0.7000	0.5600	19	0	0	0.0000	0.0000
26. . Saturn		3	45	12	-5	0	0	0.0000	0.0000	20	0	0	0.0000	0.0000
27. Saturn -	1245	3			13	1	15	0.7500	0.5556	21	0	0	0.0000	0.0000
28. Saturn -	12345				15	1	16	0.8000	0.5714	22	0	0	0.0000	0.0000
29. Saturn			12345		-15	0	0	0.0000	0.0000	25	1	16	0.8000	0.5517
30. Saturn of			12345		-15	0	0	0.0000	0.0000	26	0	0	0.0000	0.0000
31. Saturn -	12345				15	1	17	0.8500	0.5484	28	1	17	0.8500	0.5484
32. Saturn of			12345		-15	0	0	0.0000	0.0000	29	0	0	0.0000	0.0000
33. Saturn of			12345		-15	0	0	0.0000	0.0000	30	0	0	0.0000	0.0000
34. SaturnFL.			12345		-15	0	0	0.0000	0.0000	31	1	18	0.9000	0.5294
35. Saturn of			12345		-15	0	0	0.0000	0.0000	32	0	0	0.0000	0.0000
36. Saturn —			12345		-15	0	0	0.0000	0.0000	33	0	0	0.0000	0.0000
37. Saturn of			1234	5	-12	0	0	0.0000	0.0000	34	0	0	0.0000	0.0000
38. Saturn			12345		-15	0	0	0.0000	0.0000	35	0	0	0.0000	0.0000
39. Saturn V:	4	3	125		-5	0	0	0.0000	0.0000	37	0	0	0.0000	0.0000
40. Saturn of		3	1245		-11	0	0	0.0000	0.0000	38	0	0	0.0000	0.0000
41. Saturn	145	23			11	1	18	0.9000	0.4390	39	0	0	0.0000	0.0000
42. Saturn of		3	1245		-11	0	0	0.0000	0.0000	40	0	0	0.0000	0.0000
43. Saturn of			12345		-15	0	0	0.0000	0.0000	41	1	19	0.9500	0.4419
44. New			12345		-15	0	0	0.0000	0.0000	42	0	0	0.0000	0.0000
45 Faulkner			12345		-15	0	0	0.0000	0.0000	43	0	0	0.0000	0.0000
46. Saturn |	1		345	2	-6	0	0	0.0000	0.0000	45	0	0	0.0000	0.0000
47. Saturn of			12345		-15	0	0	0.0000	0.0000	46	0	0	0.0000	0.0000
48. Saturn of			12345		-15	0	0	0.0000	0.0000	47	0	0	0.0000	0.0000
49. Saturn |	1245	3			13	1	19	0.9500	0.3878	48	0	0	0.0000	0.0000
50. Saturn	134	2	5		7	1	20	1.0000	0.4000	50	1	20	1.0000	0.4000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL’: number of relevant document in new ranked results, Rc’: recall of new results, Pr’: precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	88.9	88.9	88.9	88.9	83.3	80	57.1	57.1	43.9	40
Pr Bu	100	100	73.7	73.7	73.7	73.7	73.7	55.2	52.9	40

JN: 80; AN: 29; JCD: 0.3625; Impr: -0.04



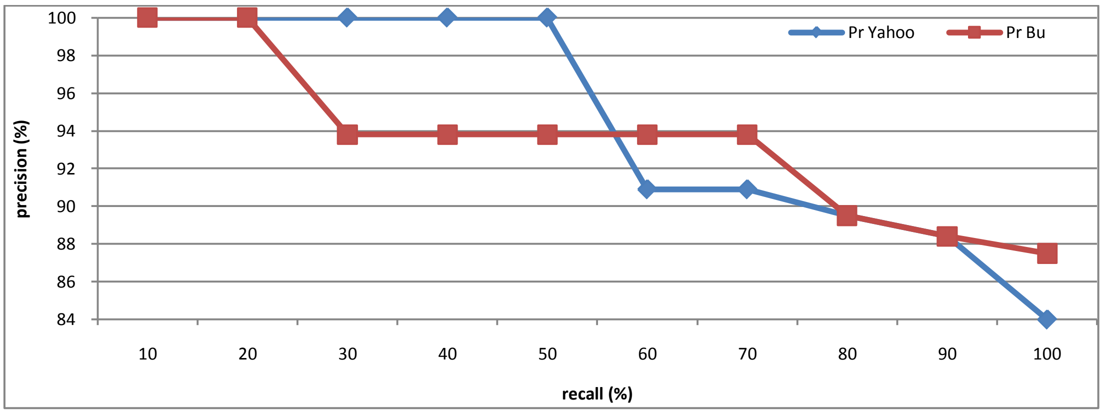
Search-term G2-5: susan dumais; Info need: the research Susan Dumais

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL'	Rc'	Pr'
1. Susan	1234			5	12	1	1	0.0238	1.0000	1	1	1	0.0238	1.0000
2. ACM	134	2		5	10	1	2	0.0476	1.0000	2	1	2	0.0476	1.0000
3. Susan s	1234			5	12	1	3	0.0714	1.0000	5	1	3	0.0714	1.0000
4. DBLP:	234	1		5	10	1	4	0.0952	1.0000	7	1	4	0.0952	1.0000
5. Susan	234	1		5	10	1	5	0.1190	1.0000	11	1	5	0.1190	1.0000
6. Susan	123			45	9	1	6	0.1429	1.0000	13	1	6	0.1429	1.0000
7. Susan	1234			5	12	1	7	0.1667	1.0000	15	1	7	0.1667	1.0000
8. Susan	4	123		5	6	1	8	0.1905	1.0000	16	1	8	0.1905	1.0000
9. Susan	24	3	1	5	4	1	9	0.2143	1.0000	23	1	9	0.2143	1.0000
10. Geeking		1234		5	4	1	10	0.2381	1.0000	26	1	10	0.2381	1.0000
11. Improved	124	3		5	10	1	11	0.2619	1.0000	32	1	11	0.2619	1.0000
12. Susan	14	3		25	7	1	12	0.2857	1.0000	39	1	12	0.2857	1.0000
13. Meeting	1	23	4	5	2	1	13	0.3095	1.0000	47	0	0	0.0000	0.0000
14. Susan	124		3	5	6	1	14	0.3333	1.0000	3	1	13	0.3095	0.9286
15. UIC -	24	13		5	8	1	15	0.3571	1.0000	9	1	14	0.3333	0.9333
16. From:	4	123		5	6	1	16	0.3810	1.0000	17	1	15	0.3571	0.9375
17. Interfaces	24	13		5	8	1	17	0.4048	1.0000	20	1	16	0.3810	0.9412
18. Susan	2	13		45	5	1	18	0.4286	1.0000	24	0	0	0.0000	0.0000
19. Susan	14	23		5	8	1	19	0.4524	1.0000	27	1	17	0.4048	0.8947
20. CiteSeer	24	13		5	8	1	20	0.4762	1.0000	33	1	18	0.4286	0.9000
21. Susan	124	3		5	10	1	21	0.5000	1.0000	34	1	19	0.4524	0.9048
22. Mehran			13	245	-6	0	0	0.0000	0.0000	37	1	20	0.4762	0.9091
23. Susan	124	3		5	10	1	22	0.5238	0.9565	43	1	21	0.5000	0.9130
24 LSU		3	14	25	-5	0	0	0.0000	0.0000	44	1	22	0.5238	0.9167
25. Susan	4		13	25	-3	0	0	0.0000	0.0000	50	1	23	0.5476	0.9200
26 Web	4	3	1	25	1	1	23	0.5476	0.8846	4	1	24	0.5714	0.9231
27. Browse	4	12		35	5	1	24	0.5714	0.8889	6	1	25	0.5952	0.9259
28. Congratu	124	3		5	10	1	25	0.5952	0.8929	8	1	26	0.6190	0.9286
29. Dumais	124		3	5	6	1	26	0.6190	0.8966	10	1	27	0.6429	0.9310
30. CiteSee		12		345	2	1	27	0.6429	0.9000	12	1	28	0.6667	0.9333
31. ANNUA	1	23		45	5	1	28	0.6667	0.9032	14	1	29	0.6905	0.9355
32. Susan	124	3		5	10	1	29	0.6905	0.9063	18	1	30	0.7143	0.9375
33. ACL-08:	124	3		5	10	1	30	0.7143	0.9091	19	1	31	0.7381	0.9394
34. Dumai	2	1	3	45	1	1	31	0.7381	0.9118	21	1	32	0.7619	0.9412
35. index.h	24		1	35	3	1	32	0.7619	0.9143	22	0	0	0.0000	0.0000
36. Susan	4		1	235	-1	0	0	0.0000	0.0000	25	0	0	0.0000	0.0000
37. What is a	4	12		35	5	1	33	0.7857	0.8919	28	1	33	0.7857	0.8919
38. ANNUA	24	1	3	5	4	1	34	0.8095	0.8947	29	1	34	0.8095	0.8947
39. Hierarch	24	13		5	8	1	35	0.8333	0.8974	30	1	35	0.8333	0.8974
40. Recent	124		3	5	6	1	36	0.8571	0.9000	31	1	36	0.8571	0.9000
41. langreiter.	4		1	235	-1	0	0	0.0000	0.0000	35	1	37	0.8810	0.9024
42. Informat	4	23	1	5	2	1	37	0.8810	0.8810	36	0	0	0.0000	0.0000
43. Faculty	14	2	3	5	4	1	38	0.9048	0.8837	38	1	38	0.9048	0.8837
44. CIIR	24	3	1	5	4	1	39	0.9286	0.8864	40	1	39	0.9286	0.8864
45. Susan	24	1		35	7	1	40	0.9524	0.8889	41	0	0	0.0000	0.0000
46. Digital	4	2	1	35	1	1	41	0.9762	0.8913	42	1	40	0.9524	0.8696
47. Latent	4	2	13	5	-2	0	0	0.0000	0.0000	45	1	41	0.9762	0.8723
48. Definitio		34	1	25	-1	0	0	0.0000	0.0000	46	1	42	1.0000	0.8750
49. Dumais	1		3	245	-1	0	0	0.0000	0.0000	48	0	0	0.0000	0.0000
50. Arts	24	1		35	7	1	42	1.0000	0.8400	49	0	0	0.0000	0.0000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL': number of relevant document in new ranked results, Rc': recall of new results, Pr': precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	100	100	100	100	100	90.9	90.9	89.5	88.4	84
Pr Bu	100	100	93.8	93.8	93.8	93.8	93.8	89.5	88.4	87.5

JN: 153; AN: 0; JCD: 0; Impr: -0.93



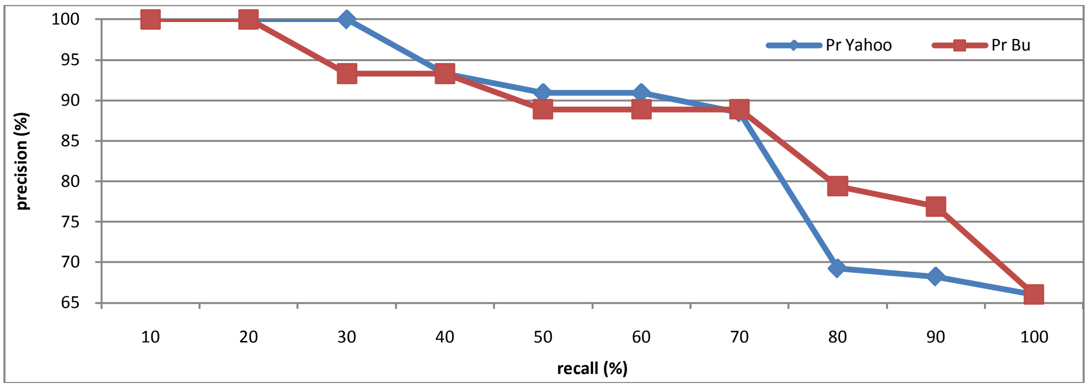
Search-term G3-1: dell; Info need: the Dell company

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL’	Rc’	Pr’
1. Dell	1345		2		9	1	1	0.0303	1.0000	2	1	1	0.0303	1.0000
2. Laptops	12345				15	1	2	0.0606	1.0000	3	1	2	0.0606	1.0000
3. Dell	12345				15	1	3	0.0909	1.0000	4	1	3	0.0909	1.0000
4. Dell	1345	2			13	1	4	0.1212	1.0000	5	1	4	0.1212	1.0000
5. Dell	1345		2		9	1	5	0.1515	1.0000	6	1	5	0.1515	1.0000
6. Dell	1345		2		9	1	6	0.1818	1.0000	10	1	6	0.1818	1.0000
7. Dell -	24	13	5		5	1	7	0.2121	1.0000	15	1	7	0.2121	1.0000
8. DELL:	1245	3			13	1	8	0.2424	1.0000	17	1	8	0.2424	1.0000
9. Dell:	135	4	2		7	1	9	0.2727	1.0000	18	0	0	0.0000	0.0000
10. Laptops	14	35	2		5	1	10	0.3030	1.0000	19	1	9	0.2727	0.9000
11. Encuentre	14	35	2		5	1	11	0.3333	1.0000	20	1	10	0.3030	0.9091
12. Printers -	345	1	2		7	1	12	0.3636	1.0000	21	1	11	0.3333	0.9167
13. Bac		5	24	13	-5	0	0	0.0000	0.0000	26	1	12	0.3636	0.9231
14 . Laptops	14	5	2	3	4	1	13	0.3939	0.9286	28	1	13	0.3939	0.9286
15. Linux	345	1	2		7	1	14	0.4242	0.9333	30	1	14	0.4242	0.9333
16. Sitemap |	13	45	2		5	1	15	0.4545	0.9375	31	1	15	0.4545	0.9375
17. Dell's	1345	2			13	1	16	0.4848	0.9412	34	0	0	0.0000	0.0000
18. Desktop		5	124	3	-8	0	0	0.0000	0.0000	37	0	0	0.0000	0.0000
19. Desktop	1345		2		9	1	17	0.5152	0.8947	40	1	16	0.4848	0.8421
20. Welcome	1345		2		9	1	18	0.5455	0.9000	43	1	17	0.5152	0.8500
21. Dell	134	25			11	1	19	0.5758	0.9048	48	1	18	0.5455	0.8571
22. Learn	34	1	25		1	1	20	0.6061	0.9091	50	1	19	0.5758	0.8636
23. Dell		1345	2		1	1	21	0.6364	0.9130	1	1	20	0.6061	0.8696
24. Dell |		3	1245		-11	0	0	0.0000	0.0000	8	1	21	0.6364	0.8750
25. Premier..		1345	2		1	1	22	0.6667	0.8800	14	1	22	0.6667	0.8800
26. Desktop	134	25			11	1	23	0.6970	0.8846	22	1	23	0.6970	0.8846
27. Nancy			2345	1	-12	0	0	0.0000	0.0000	23	1	24	0.7273	0.8889
28 Dell	13	245			9	1	24	0.7273	0.8571	24	0	0	0.0000	0.0000
29. Monitors:	3	15	24		-1	0	0	0.0000	0.0000	25	1	25	0.7576	0.8621
30. Desktop	134	25			11	1	25	0.7576	0.8333	36	0	0	0.0000	0.0000
31. Dell ISG	13	5	24		1	1	26	0.7879	0.8387	38	0	0	0.0000	0.0000
32. Forums -	5	13	24		-1	0	0	0.0000	0.0000	39	1	26	0.7879	0.8125
33. Home			2345	1	-12	0	0	0.0000	0.0000	41	0	0	0.0000	0.0000
34. Dell		345	12		-3	0	0	0.0000	0.0000	44	1	27	0.8182	0.7941
35. Direct2 -	4	35	12		-1	0	0	0.0000	0.0000	45	0	0	0.0000	0.0000
36. Citrix		35	124		-7	0	0	0.0000	0.0000	49	0	0	0.0000	0.0000
37. Desktop	1	5	24	3	-2	0	0	0.0000	0.0000	7	1	28	0.8485	0.7568
38. Dell -			2345	1	-12	0	0	0.0000	0.0000	9	1	29	0.8788	0.7632
39. Dell	4	135	2		3	1	27	0.8182	0.6923	11	1	30	0.9091	0.7692
40. Laptops	134	25			11	1	28	0.8485	0.7000	12	1	31	0.9394	0.7750
41. Dell	4		125	3	-6	0	0	0.0000	0.0000	13	0	0	0.0000	0.0000
42. Dell			2345	1	-12	0	0	0.0000	0.0000	16	1	32	0.9697	0.7619
43. Learn	34	125			9	1	29	0.8788	0.6744	27	0	0	0.0000	0.0000
44. Dell	14	235			9	1	30	0.9091	0.6818	29	0	0	0.0000	0.0000
45. Dell:	2	13	45		-1	0	0	0.0000	0.0000	32	0	0	0.0000	0.0000
46. 计算机	1	45	2	3	2	1	31	0.9394	0.6739	33	0	0	0.0000	0.0000
47. Amazon.	4	35	12		-1	0	0	0.0000	0.0000	35	0	0	0.0000	0.0000
48. Desktop	14	25		3	8	1	32	0.9697	0.6667	42	0	0	0.0000	0.0000
49. AAA.com/			1345	2	-12	0	0	0.0000	0.0000	47	0	0	0.0000	0.0000
50. Dell Bulg	134	25			11	1	33	1.0000	0.6600	46	1	33	1.0000	0.6600

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL’: number of relevant document in new ranked results, Rc’: recall of new results, Pr’: precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	100	100	100	93.3	90.9	90.9	88.5	69.2	68.2	66
Pr Bu	100	100	93.3	93.3	88.9	88.9	88.9	79.4	76.9	66

JN: 153; AN: 0; JCD: 0; Impr: -0.93



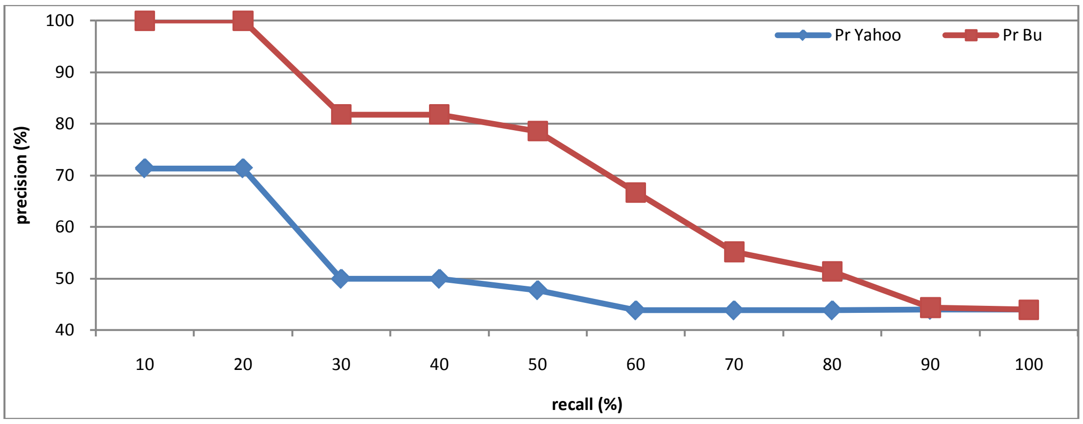
Search-term G3-2: Jordan; Info need: the Hashemite kingdom Jordan

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL’	Rc’	Pr’
1. Jumpman			12345		-15	0	0	0.0000	0.0000	2	1	1	0.0455	1.0000
2. Jordan -	12345				15	1	1	0.0455	0.5000	5	1	2	0.0909	1.0000
3. Jordan:	12	35	4		5	1	2	0.0909	0.6667	15	1	3	0.1364	1.0000
4. Michael			12345		-15	0	0	0.0000	0.0000	18	1	4	0.1818	1.0000
5. Jordan	134	5	2		7	1	3	0.1364	0.6000	22	1	5	0.2273	1.0000
6. Jordan -	1235	4			13	1	4	0.1818	0.6667	31	0	0	0.0000	0.0000
7. Jordan	34	125			9	1	5	0.2273	0.7143	33	0	0	0.0000	0.0000
8. Air	4		235	1	-6	0	0	0.0000	0.0000	46	1	6	0.2727	0.7500
9. Jordan's			12345		-15	0	0	0.0000	0.0000	50	1	7	0.3182	0.7778
10. Jordan	4	3	25	1	-2	0	0	0.0000	0.0000	3	1	8	0.3636	0.8000
11. Jordan		145	2	3	-1	0	0	0.0000	0.0000	24	1	9	0.4091	0.8182
12. Jordan			12345		-15	0	0	0.0000	0.0000	32	0	0	0.0000	0.0000
13. Jordan		13	5	24	-1	0	0	0.0000	0.0000	34	1	10	0.4545	0.7692
14. Jordan	34	5	12		1	1	6	0.2727	0.4286	36	1	11	0.5000	0.7857
15. Jordan	12345				15	1	7	0.3182	0.4667	38	1	12	0.5455	0.8000
16. City of	45	2	1	3	4	1	8	0.3636	0.5000	39	1	13	0.5909	0.8125
17. Jordan		4	235	1	-8	0	0	0.0000	0.0000	45	0	0	0.0000	0.0000
18. Jordan -	123	45			11	1	9	0.4091	0.5000	47	0	0	0.0000	0.0000
19. Charles			12345		-15	0	0	0.0000	0.0000	1	0	0	0.0000	0.0000
20. NBA.c			12345		-15	0	0	0.0000	0.0000	4	0	0	0.0000	0.0000
21. Jordan :	34		25	1	-1	0	0	0.0000	0.0000	6	1	14	0.6364	0.6667
22. Jordan:	1234	5			13	1	10	0.4545	0.4545	7	1	15	0.6818	0.6818
23. Jordan -	1234	5			13	1	11	0.5000	0.4783	8	0	0	0.0000	0.0000
24. Map of	1345	2			13	1	12	0.5455	0.5000	9	0	0	0.0000	0.0000
25. . Jordan		5	1234		-11	0	0	0.0000	0.0000	10	0	0	0.0000	0.0000
26. Stand-Out			12345		-15	0	0	0.0000	0.0000	11	0	0	0.0000	0.0000
27. Air			12345		-15	0	0	0.0000	0.0000	12	0	0	0.0000	0.0000
28. Fromm	34	15	2		5	1	13	0.5909	0.4643	13	0	0	0.0000	0.0000
29. NBA.			12345		-15	0	0	0.0000	0.0000	14	1	16	0.7273	0.5517
30. Online			12345		-15	0	0	0.0000	0.0000	16	1	17	0.7727	0.5667
31. Jordan :			12345		-15	0	0	0.0000	0.0000	17	0	0	0.0000	0.0000
32. Jordan		4	235	1	-8	0	0	0.0000	0.0000	19	0	0	0.0000	0.0000
33. Jordan		45	2	13	-1	0	0	0.0000	0.0000	20	0	0	0.0000	0.0000
34. YouTube	134	25			11	1	14	0.6364	0.4118	21	0	0	0.0000	0.0000
35. JORDAN	4		235	1	-6	0	0	0.0000	0.0000	23	1	18	0.8182	0.5143
36. Petra	34	125			9	1	15	0.6818	0.4167	25	0	0	0.0000	0.0000
37. 23Jordan			12345		-15	0	0	0.0000	0.0000	26	0	0	0.0000	0.0000
38. Jordan -	1234	5			13	1	16	0.7273	0.4211	27	0	0	0.0000	0.0000
39. Jordan	34	1	25		1	1	17	0.7727	0.4359	28	1	19	0.8636	0.4872
40. Jordan		45	2	13	-1	0	0	0.0000	0.0000	29	0	0	0.0000	0.0000
41. Jordan	4	15	2	3	2	1	18	0.8182	0.4390	30	0	0	0.0000	0.0000
42. Jordan	134	25			11	1	19	0.8636	0.4524	35	0	0	0.0000	0.0000
43. Jordan	4	13	25		-1	0	0	0.0000	0.0000	37	0	0	0.0000	0.0000
44. YouTube			12345		-15	0	0	0.0000	0.0000	40	0	0	0.0000	0.0000
45. Jordan			12345		-15	0	0	0.0000	0.0000	41	1	20	0.9091	0.4444
46. Jordan	4	135	2		3	1	20	0.9091	0.4348	42	1	21	0.9545	0.4565
47. Jordan's			12345		-15	0	0	0.0000	0.0000	43	0	0	0.0000	0.0000
48 Jordan			12345		-15	0	0	0.0000	0.0000	44	0	0	0.0000	0.0000
49. Jordan	13	45	2		5	1	21	0.9545	0.4286	48	0	0	0.0000	0.0000
50. Governm	1234	5			13	1	22	1.0000	0.4400	49	1	22	1.0000	0.4400

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL’: number of relevant document in new ranked results, Rc’: recall of new results, Pr’: precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	71.4	71.4	50	50	47.8	43.9	43.9	43.9	44	44
Pr Bu	100	100	81.8	81.8	78.6	66.7	55.2	51.4	44.4	44

JN: 105; AN: 18; JCD: 0; Impr: 19.36



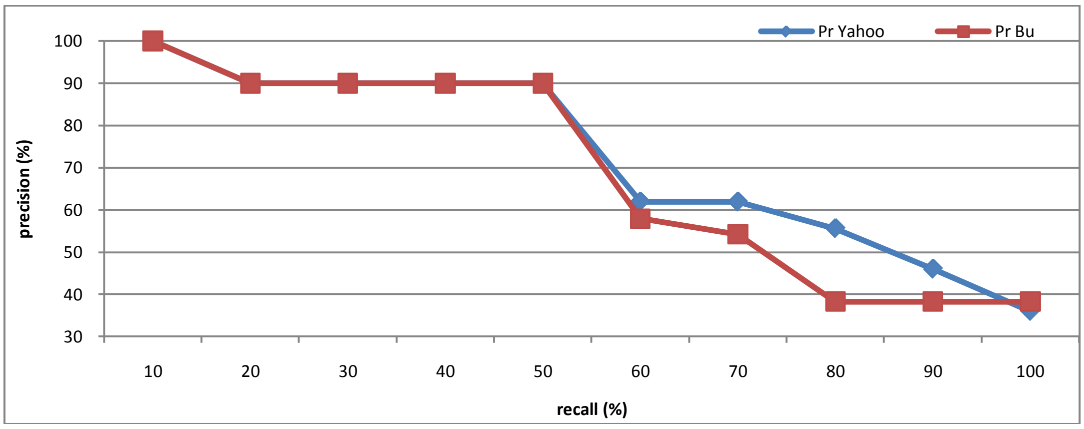
Search-term G3-3: music; Info need: music classification by Genre

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL’	Rc’	Pr’
1. Yahoo!	45	23	1		5	1	1	0.0556	1.0000	1	1	1	0.0556	1.0000
2. Music.com	45	3	12		1	1	2	0.1111	1.0000	2	1	2	0.1111	1.0000
3. AOL	45	3	12		1	1	3	0.1667	1.0000	3	1	3	0.1667	1.0000
4. Last.FM	45		12	3	-1	0	0	0.0000	0.0000	4	0	0	0.0000	0.0000
5. Music in	12345				15	1	4	0.2222	0.8000	5	1	4	0.2222	0.8000
6. Amazon.	345		12		3	1	5	0.2778	0.8333	6	1	5	0.2778	0.8333
7. allmusic	245	3	1		7	1	6	0.3333	0.8571	7	1	6	0.3333	0.8571
8. MySpace	14	35	2		5	1	7	0.3889	0.8750	8	1	7	0.3889	0.8750
9 Billboard	45	3	12		1	1	8	0.4444	0.8889	9	1	8	0.4444	0.8889
10. NPR	145	3	2		7	1	9	0.5000	0.9000	10	1	9	0.5000	0.9000
11. Yahoo!	1345		2		9	1	10	0.5556	0.9091	11	1	10	0.5556	0.9091
12. Music -	4	15	23		-1	0	0	0.0000	0.0000	12	0	0	0.0000	0.0000
13 G-Music	4	5	12	3	-2	0	0	0.0000	0.0000	13	0	0	0.0000	0.0000
14. New	4	5	12	3	-2	0	0	0.0000	0.0000	14	0	0	0.0000	0.0000
15. YouTube	4	1	25	3	-2	0	0	0.0000	0.0000	15	0	0	0.0000	0.0000
16. College	4	5	123		-5	0	0	0.0000	0.0000	16	0	0	0.0000	0.0000
17. Music	4		5	123	-1	0	0	0.0000	0.0000	17	0	0	0.0000	0.0000
18. EMusic.c	24	135			9	1	11	0.6111	0.6111	19	0	0	0.0000	0.0000
19. Piczo:	4	1	235		-5	0	0	0.0000	0.0000	21	1	11	0.6111	0.5789
20. MSN	45	1	2	3	4	1	12	0.6667	0.6000	22	1	12	0.6667	0.6000
21. Music	34	15	2		5	1	13	0.7222	0.6190	23	0	0	0.0000	0.0000
22. Music	1234		5		9	1	14	0.7778	0.6364	24	0	0	0.0000	0.0000
23. MySpace		1	245	3	-8	0	0	0.0000	0.0000	25	0	0	0.0000	0.0000
24. Music	5		1234		-9	0	0	0.0000	0.0000	27	1	13	0.7222	0.5417
25. Music -	5	4	23	1	-2	0	0	0.0000	0.0000	28	0	0	0.0000	0.0000
26. eBay –	4		1235		-9	0	0	0.0000	0.0000	29	0	0	0.0000	0.0000
27 BBC	1245	3			13	1	15	0.8333	0.5556	30	0	0	0.0000	0.0000
28. Yahoo!	4		125	3	-6	0	0	0.0000	0.0000	31	0	0	0.0000	0.0000
29. Music	4	5	123		-5	0	0	0.0000	0.0000	32	0	0	0.0000	0.0000
30. Music :	4	5	123		-5	0	0	0.0000	0.0000	33	0	0	0.0000	0.0000
31. Yahoo!7	45		12	3	-1	0	0	0.0000	0.0000	34	0	0	0.0000	0.0000
32. E-Music	4	5	123		-5	0	0	0.0000	0.0000	35	0	0	0.0000	0.0000
33. KCRW -	4	5	12	3	-2	0	0	0.0000	0.0000	36	1	14	0.7778	0.4242
34. Free	4	15	23		-1	0	0	0.0000	0.0000	40	0	0	0.0000	0.0000
35. Columbia	5		1234		-9	0	0	0.0000	0.0000	41	0	0	0.0000	0.0000
36. Music	45	1	23		1	1	16	0.8889	0.4444	42	0	0	0.0000	0.0000
37. Q-music	4	5	2	13	1	1	17	0.9444	0.4595	43	0	0	0.0000	0.0000
38. Florida		5	1234		-11	0	0	0.0000	0.0000	44	0	0	0.0000	0.0000
39. CDNOW	45		123		-3	0	0	0.0000	0.0000	45	0	0	0.0000	0.0000
40. Musicro	45		123		-3	0	0	0.0000	0.0000	46	0	0	0.0000	0.0000
41 Music |	4	5	23	1	-2	0	0	0.0000	0.0000	47	0	0	0.0000	0.0000
42. Home :	4	1	235		-5	0	0	0.0000	0.0000	48	0	0	0.0000	0.0000
43. Music	4	5	123		-5	0	0	0.0000	0.0000	49	0	0	0.0000	0.0000
44. Musicnot	4	5	12	3	-2	0	0	0.0000	0.0000	50	1	15	0.8333	0.3409
45. music:			12345		-15	0	0	0.0000	0.0000	20	1	16	0.8889	0.3556
46. Sheet	4	5	12	3	-2	0	0	0.0000	0.0000	37	1	17	0.9444	0.3696
47. MTV.	4	5	123		-5	0	0	0.0000	0.0000	18	1	18	1.0000	0.3830
48. Cornell			12345		-15	0	0	0.0000	0.0000	26	0	0	0.0000	0.0000
49. Virginia			12345		-15	0	0	0.0000	0.0000	38	0	0	0.0000	0.0000
50. Start	4	5	2	13	1	1	18	1.0000	0.3600	39	0	0	0.0000	0.0000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL’: number of relevant document in new ranked results, Rc’: recall of new results, Pr’: precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	100	90	90	90	90	61.9	61.9	55.6	46	36
Pr Bu	100	90	90	90	90	57.9	54.2	38.3	38.3	38.3

JN: 142; AN: 4; JCD: 0.0282; Impr: -3.44



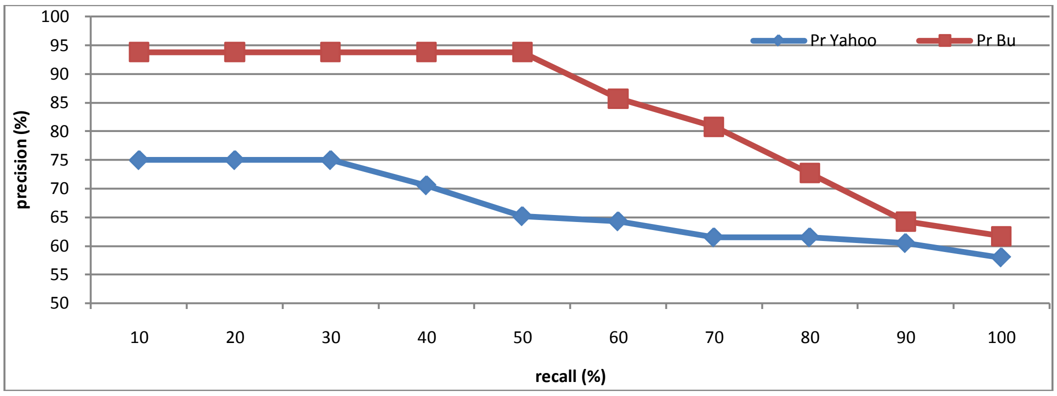
Search-term G3-4: tiger; Info need: the animal tiger

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL'	Rc'	Pr'
1. Tiger -	12345				15	1	1	0.0345	1.0000	2	1	1	0.0345	1.0000
2. Tiger	12345				15	1	2	0.0690	1.0000	6	1	2	0.0690	1.0000
3. TigerDir			12345		-15	0	0	0.0000	0.0000	8	0	0	0.0000	0.0000
4. tiger:	1235		4		9	1	3	0.1034	0.7500	9	1	3	0.1034	0.7500
5. Tiger			12345		-15	0	0	0.0000	0.0000	11	1	4	0.1379	0.8000
6. San Diego	124	35			11	1	4	0.1379	0.6667	25	1	5	0.1724	0.8333
7. Tiger -	12345				15	1	5	0.1724	0.7143	28	1	6	0.2069	0.8571
8. Tiger			12345		-15	0	0	0.0000	0.0000	35	1	7	0.2414	0.8750
9. Tiger Facts	123	5	4		7	1	6	0.2069	0.6667	43	1	8	0.2759	0.8889
10. Tiger	2	1345			7	1	7	0.2414	0.7000	45	1	9	0.3103	0.9000
11. Bengal	12345				15	1	8	0.2759	0.7273	50	1	10	0.3448	0.9091
12. Bengal	12345				15	1	9	0.3103	0.7500	12	1	11	0.3793	0.9167
13. Tiger	24	135			9	1	10	0.3448	0.7692	22	1	12	0.4138	0.9231
14. Tiger			12345		-15	0	0	0.0000	0.0000	37	1	13	0.4483	0.9286
15. Tiger			12345		-15	0	0	0.0000	0.0000	38	1	14	0.4828	0.9333
16. Tiger	1235	4			13	1	11	0.3793	0.6875	1	1	15	0.5172	0.9375
17. Save the	34	125			9	1	12	0.4138	0.7059	3	0	0	0.0000	0.0000
18. Tiger			12345		-15	0	0	0.0000	0.0000	4	1	16	0.5517	0.8889
19. Great	12345				15	1	13	0.4483	0.6842	5	0	0	0.0000	0.0000
20. Tiger			12345		-15	0	0	0.0000	0.0000	7	1	17	0.5862	0.8500
21. Tiger		5	1234		-11	0	0	0.0000	0.0000	10	1	18	0.6207	0.8571
22. tiger	1234	5			13	1	14	0.4828	0.6364	13	1	19	0.6552	0.8636
23. Tiger It is	1234	5			13	1	15	0.5172	0.6522	14	0	0	0.0000	0.0000
24. National	45		123		-3	0	0	0.0000	0.0000	15	0	0	0.0000	0.0000
25. Tiger :	1245	3			13	1	16	0.5517	0.6400	16	1	20	0.6897	0.8000
26. All for	25	4	1	3	4	1	17	0.5862	0.6538	17	1	21	0.7241	0.8077
27. TigerDir			12345		-15	0	0	0.0000	0.0000	18	0	0	0.0000	0.0000
28. TigerWil	12345				15	1	18	0.6207	0.6429	19	1	22	0.7586	0.7857
29. tiger -	1245	3			13	1	19	0.6552	0.6552	20	0	0	0.0000	0.0000
30. Tiger			12345		-15	0	0	0.0000	0.0000	21	0	0	0.0000	0.0000
31. Tigerpro			12345		-15	0	0	0.0000	0.0000	23	1	23	0.7931	0.7419
32. U.S.			1245	3	-12	0	0	0.0000	0.0000	24	0	0	0.0000	0.0000
33. Panthera	1234	5			13	1	20	0.6897	0.6061	26	1	24	0.8276	0.7273
34. ClassicTi			235	14	-9	0	0	0.0000	0.0000	27	0	0	0.0000	0.0000
35. Siberian	1234	5			13	1	21	0.7241	0.6000	29	1	25	0.8621	0.7143
36. Tiger	3		1245		-9	0	0	0.0000	0.0000	30	0	0	0.0000	0.0000
37. YouTube	1234	5			13	1	22	0.7586	0.5946	31	0	0	0.0000	0.0000
38. Great	124	35			11	1	23	0.7931	0.6053	32	0	0	0.0000	0.0000
39. Tigers -	1234	5			13	1	24	0.8276	0.6154	33	1	26	0.8966	0.6667
40. Tiger			1235	4	-12	0	0	0.0000	0.0000	34	0	0	0.0000	0.0000
41. YouTube	24	135			9	1	25	0.8621	0.6098	36	0	0	0.0000	0.0000
42. The Tiger		2	35	14	-5	0	0	0.0000	0.0000	39	1	27	0.9310	0.6429
43. Tiger	24	135			9	1	26	0.8966	0.6047	40	0	0	0.0000	0.0000
44. Page 1			12345		-15	0	0	0.0000	0.0000	41	1	28	0.9655	0.6364
45. BBC -	1234	5			13	1	27	0.9310	0.6000	42	0	0	0.0000	0.0000
46. Tiger	134	25			11	1	28	0.9655	0.6087	44	0	0	0.0000	0.0000
47. Tiger:			12345		-15	0	0	0.0000	0.0000	46	1	29	1.0000	0.6170
48. Events:			12345		-15	0	0	0.0000	0.0000	47	0	0	0.0000	0.0000
49. Tiger-Vac			12345		-15	0	0	0.0000	0.0000	48	0	0	0.0000	0.0000
50. White	1234	5			13	1	29	1.0000	0.5800	49	0	0	0.0000	0.0000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL': number of relevant document in new ranked results, Rc': recall of new results, Pr': precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	75	75	75	70.6	65.2	64.3	61.5	61.5	60.5	58
Pr Bu	93.8	93.8	93.8	93.8	93.8	85.7	80.8	72.7	64.3	61.7

JN: 83; AN: 21; JCD: 0.253; Impr: 16.76



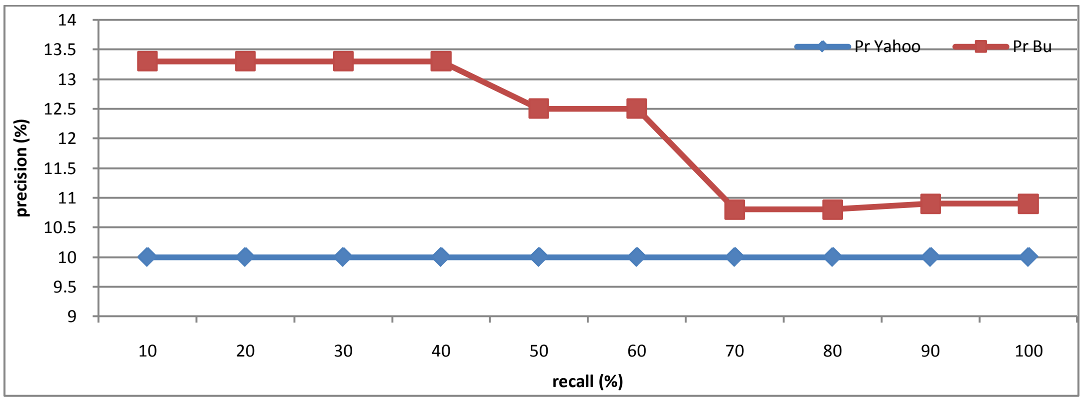
Search-term G3-5: yellow pages; Info need: the origin of yellow pages

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL’	Rc’	Pr’
1. Yahoo!	4	25	13		-1	0	0	0.0000	0.0000	2	0	0	0.0000	0.0000
2. Verizon.	4		1235		-9	0	0	0.0000	0.0000	5	0	0	0.0000	0.0000
3. Yellow	4	5	123		-5	0	0	0.0000	0.0000	13	0	0	0.0000	0.0000
4. YellowP.	4	5	123		-5	0	0	0.0000	0.0000	15	0	0	0.0000	0.0000
5. SMART	4	5	123		-5	0	0	0.0000	0.0000	16	0	0	0.0000	0.0000
6. MSN		5	1234		-11	0	0	0.0000	0.0000	17	0	0	0.0000	0.0000
7. Yellow	4	5	123		-5	0	0	0.0000	0.0000	19	0	0	0.0000	0.0000
8. Switchboa		1	2345		-11	0	0	0.0000	0.0000	23	0	0	0.0000	0.0000
9. Yellow	4	5	123		-5	0	0	0.0000	0.0000	26	0	0	0.0000	0.0000
10. AnyWho		4	1235		-11	0	0	0.0000	0.0000	31	0	0	0.0000	0.0000
11. Yellow	4	5	123		-5	0	0	0.0000	0.0000	39	0	0	0.0000	0.0000
12. ~ Advan	4	5	123		-5	0	0	0.0000	0.0000	40	0	0	0.0000	0.0000
13. Yellow	4		1235		-9	0	0	0.0000	0.0000	48	0	0	0.0000	0.0000
14. Yellow	4	5	123		-5	0	0	0.0000	0.0000	49	1	1	0.2000	0.0714
15. US Yells	4	5	123		-5	0	0	0.0000	0.0000	50	1	2	0.4000	0.1333
16. Yellow	4	15	23		-1	0	0	0.0000	0.0000	1	0	0	0.0000	0.0000
17. Yellow	4	5	123		-5	0	0	0.0000	0.0000	3	0	0	0.0000	0.0000
18. Yellow	2345	1			13	1	1	0.2000	0.0556	4	0	0	0.0000	0.0000
19. Yellow	4	5	123		-5	0	0	0.0000	0.0000	6	0	0	0.0000	0.0000
20. SuperPa		245	13		-3	0	0	0.0000	0.0000	7	0	0	0.0000	0.0000
21. Superps.	4	125	3		3	1	2	0.4000	0.0952	9	0	0	0.0000	0.0000
22. "florist" -			12345		-15	0	0	0.0000	0.0000	12	0	0	0.0000	0.0000
23. Online	4	5	123		-5	0	0	0.0000	0.0000	20	0	0	0.0000	0.0000
24. The	4	5	123		-5	0	0	0.0000	0.0000	21	1	3	0.6000	0.1250
25. IAF.net -		45	123		-7	0	0	0.0000	0.0000	24	0	0	0.0000	0.0000
26. Yellow	4	5	123		-5	0	0	0.0000	0.0000	25	0	0	0.0000	0.0000
27. Yellow	4		135	2	-6	0	0	0.0000	0.0000	28	0	0	0.0000	0.0000
28. White		4	1235		-11	0	0	0.0000	0.0000	37	0	0	0.0000	0.0000
29. DCG			12345		-15	0	0	0.0000	0.0000	38	0	0	0.0000	0.0000
30. Yellow/	4		1235		-9	0	0	0.0000	0.0000	42	0	0	0.0000	0.0000
31. belize	4	5	123		-5	0	0	0.0000	0.0000	44	0	0	0.0000	0.0000
32. Yahoo!	4	5	123		-5	0	0	0.0000	0.0000	45	0	0	0.0000	0.0000
33. Egypt	4		1235		-9	0	0	0.0000	0.0000	8	0	0	0.0000	0.0000
34. Yellow		5	1234		-11	0	0	0.0000	0.0000	10	0	0	0.0000	0.0000
35. Yellow	4		1235		-9	0	0	0.0000	0.0000	11	0	0	0.0000	0.0000
36. yellow	12	5	34		1	1	3	0.6000	0.0833	14	0	0	0.0000	0.0000
37. Search C	4	5	123		-5	0	0	0.0000	0.0000	18	1	4	0.8000	0.1081
38. UAE	4	25	13		-1	0	0	0.0000	0.0000	22	0	0	0.0000	0.0000
39. Internet	4	5	123		-5	0	0	0.0000	0.0000	27	0	0	0.0000	0.0000
40. UAE	4	5	123		-5	0	0	0.0000	0.0000	29	0	0	0.0000	0.0000
41. Mongolia	4	5	123		-5	0	0	0.0000	0.0000	30	0	0	0.0000	0.0000
42. Internat	4	5	123		-5	0	0	0.0000	0.0000	32	0	0	0.0000	0.0000
43. Hawaii	4	5	123		-5	0	0	0.0000	0.0000	33	0	0	0.0000	0.0000
44. Sulekha	4		1235		-9	0	0	0.0000	0.0000	34	0	0	0.0000	0.0000
45. United	4	5	123		-5	0	0	0.0000	0.0000	35	0	0	0.0000	0.0000
46. Australia	4	5	123		-5	0	0	0.0000	0.0000	36	1	5	1.0000	0.1087
47. Alberta			12345		-15	0	0	0.0000	0.0000	41	0	0	0.0000	0.0000
48. Rhode	4	5	123		-5	0	0	0.0000	0.0000	43	0	0	0.0000	0.0000
49. Why	45	1	23		1	1	4	0.8000	0.0816	46	0	0	0.0000	0.0000
50. Standard	34	15	2		5	1	5	1.0000	0.1000	47	0	0	0.0000	0.0000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL’: number of relevant document in new ranked results, Rc’: recall of new results, Pr’: precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	10	10	10	10	10	10	10	10	10	10
Pr Bu	13.3	13.3	13.3	13.3	12.5	12.5	10.8	10.8	10.9	10.9

JN: 130; AN: 3; JCD: 0.0231; Impr: 2.16



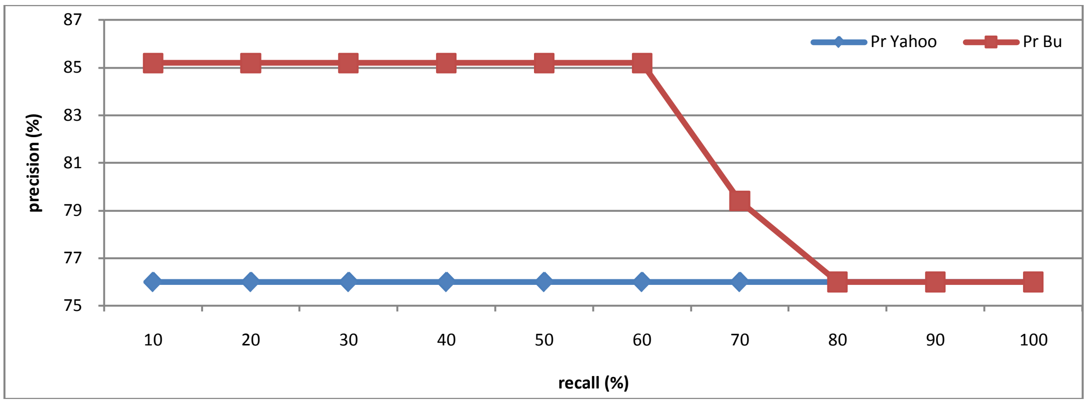
Search-term G4-1: chat; Info need: computer-mediated chat systems

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL'	Rc'	Pr'
1. Yahoo!	2	15	34		-1	0	0	0.0000	0.0000	2	1	1	0.0263	1.0000
2. Chat -	3	124	5		3	1	1	0.0263	0.5000	4	0	0	0.0000	0.0000
3. ICQ Chat	125	34			11	1	2	0.0526	0.6667	5	1	2	0.0526	0.6667
4. Online	3	1	245		-5	0	0	0.0000	0.0000	9	1	3	0.0789	0.7500
5. Yahoo!	23	4	15		1	1	3	0.0789	0.6000	11	1	4	0.1053	0.8000
6. Chat	23	1	45		1	1	4	0.1053	0.6667	13	0	0	0.0000	0.0000
7. Paltalk	2	4	35	1	-2	0	0	0.0000	0.0000	20	1	5	0.1316	0.7143
8. Talk City	23	1	45		1	1	5	0.1316	0.6250	21	1	6	0.1579	0.7500
9. MySpace	23		14	5	1	1	6	0.1579	0.6667	25	0	0	0.0000	0.0000
10. chat		4	5	123	-2	0	0	0.0000	0.0000	28	1	7	0.1842	0.7000
11. mIRC	1	245	3		3	1	7	0.1842	0.6364	30	1	8	0.2105	0.7273
12. ICQ.com	23	45	1		5	1	8	0.2105	0.6667	37	1	9	0.2368	0.7500
13. Chat -		5	234	1	-8	0	0	0.0000	0.0000	45	1	10	0.2632	0.7692
14. About	12	3	45		1	1	9	0.2368	0.6429	46	1	11	0.2895	0.7857
15. how do i	1	23	45		-1	0	0	0.0000	0.0000	1	0	0	0.0000	0.0000
16. Chat	23	5	14		1	1	10	0.2632	0.6250	3	1	12	0.3158	0.7500
17. Yaba	23	5	14		1	1	11	0.2895	0.6471	8	1	13	0.3421	0.7647
18. Live Chat	13	24	5		5	1	12	0.3158	0.6667	14	1	14	0.3684	0.7778
19. ParaChat	34	25	1		5	1	13	0.3421	0.6842	17	1	15	0.3947	0.7895
20. ParaChat	23	45	1		5	1	14	0.3684	0.7000	18	1	16	0.4211	0.8000
21. X-Chat	34	2	15		1	1	15	0.3947	0.7143	19	1	17	0.4474	0.8095
22. YouTube	1		2345		-9	0	0	0.0000	0.0000	36	1	18	0.4737	0.8182
23. Chatabla	23		145		-3	0	0	0.0000	0.0000	38	1	19	0.5000	0.8261
24. Browse	3	24	15		-1	0	0	0.0000	0.0000	39	1	20	0.5263	0.8333
25. Alamak	23		145		-3	0	0	0.0000	0.0000	40	1	21	0.5526	0.8400
26. Chat-Web	23	5	14		1	1	16	0.4211	0.6154	41	1	22	0.5789	0.8462
27. xat.com	2		1345		-9	0	0	0.0000	0.0000	44	1	23	0.6053	0.8519
28. WireClub	23	5	14		1	1	17	0.4474	0.6071	49	1	24	0.6316	0.8571
29. Flash	12	345			9	1	18	0.4737	0.6207	6	1	25	0.6579	0.8621
30. DigiChat	24	5	13		1	1	19	0.5000	0.6333	7	0	0	0.0000	0.0000
31. Chat.	1	4	5	23	1	1	20	0.5263	0.6452	10	0	0	0.0000	0.0000
32. chat	1		2345		-9	0	0	0.0000	0.0000	12	1	26	0.6842	0.8125
33. Google	235	4	1		7	1	21	0.5526	0.6364	15	0	0	0.0000	0.0000
34. YAYchat	235		14		3	1	22	0.5789	0.6471	16	1	27	0.7105	0.7941
35. Google	1235		4		9	1	23	0.6053	0.6571	22	0	0	0.0000	0.0000
36. FREE	124	3		5	10	1	24	0.6316	0.6667	23	0	0	0.0000	0.0000
37. Chat -	14	25	3		5	1	25	0.6579	0.6757	24	0	0	0.0000	0.0000
38. TeenCha	235		14		3	1	26	0.6842	0.6842	26	1	28	0.7368	0.7368
39. YapChat	235	4	1		7	1	27	0.7105	0.6923	27	0	0	0.0000	0.0000
40. ChatNZ -	235		14		3	1	28	0.7368	0.7000	29	1	29	0.7632	0.7250
41. ChatSpan	235		14		3	1	29	0.7632	0.7073	31	1	30	0.7895	0.7317
42. OohYa	235		14		3	1	30	0.7895	0.7143	32	0	0	0.0000	0.0000
43. People	1235		4		9	1	31	0.8158	0.7209	33	1	31	0.8158	0.7209
44. Chatzy -	1235	4			13	1	32	0.8421	0.7273	34	1	32	0.8421	0.7273
45. Chatho	1235		4		9	1	33	0.8684	0.7333	35	1	33	0.8684	0.7333
46. AddonC	1	345	2		3	1	34	0.8947	0.7391	42	1	34	0.8947	0.7391
47. Kewlchat	235		4	1	6	1	35	0.9211	0.7447	43	1	35	0.9211	0.7447
48. YouTube	15	3	24		1	1	36	0.9474	0.7500	47	1	36	0.9474	0.7500
49. Coolsmile	235		4	1	6	1	37	0.9737	0.7551	48	1	37	0.9737	0.7551
50. Spinchat	235		4	1	6	1	38	1.0000	0.7600	50	1	38	1.0000	0.7600

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL': number of relevant document in new ranked results, Rc': recall of new results, Pr': precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	76	76	76	76	76	76	76	76	76	76
Pr Bu	85.2	85.2	85.2	85.2	85.2	85.2	79.4	76	76	76

JN: 136; AN: 0; JCD: 0; Impr: 5.86



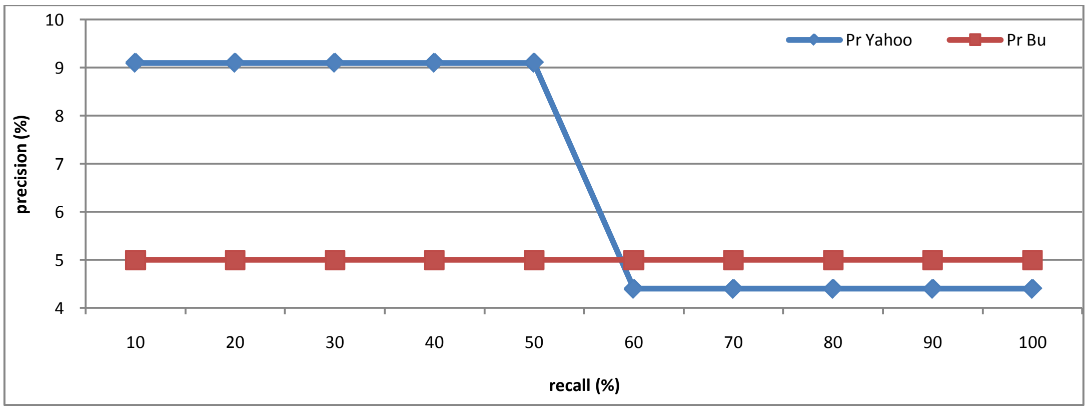
Search-term G4-2: games; Info need: history of games

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL'	Rc'	Pr'
1. Yahoo!	1	5	234		-5	0	0	0.0000	0.0000	1	0	0	0.0000	0.0000
2. Games			2345	1	-12	0	0	0.0000	0.0000	2	0	0	0.0000	0.0000
. Pogo.com			2345	1	-12	0	0	0.0000	0.0000	3	0	0	0.0000	0.0000
4. MSN	1		2345		-9	0	0	0.0000	0.0000	4	0	0	0.0000	0.0000
5. MiniCli			2345	1	-12	0	0	0.0000	0.0000	6	0	0	0.0000	0.0000
6. Addicting			2345	1	-12	0	0	0.0000	0.0000	7	0	0	0.0000	0.0000
7. Play Gam	1		2345		-9	0	0	0.0000	0.0000	8	0	0	0.0000	0.0000
8. PopCap		1	2345		-11	0	0	0.0000	0.0000	10	0	0	0.0000	0.0000
9. Candysta		2	134	5	-8	0	0	0.0000	0.0000	13	0	0	0.0000	0.0000
10. GameHou		1	2345		-11	0	0	0.0000	0.0000	16	0	0	0.0000	0.0000
11. X Games	2	145	3		3	1	1	0.5000	0.0909	18	0	0	0.0000	0.0000
12. Games -		15	234		-7	0	0	0.0000	0.0000	20	0	0	0.0000	0.0000
13. GameFoo		5	234	1	-8	0	0	0.0000	0.0000	21	0	0	0.0000	0.0000
14. Games |	1		2345		-9	0	0	0.0000	0.0000	22	0	0	0.0000	0.0000
15. Games |	1		2345		-9	0	0	0.0000	0.0000	23	0	0	0.0000	0.0000
16. All Gam		5	234	1	-8	0	0	0.0000	0.0000	24	0	0	0.0000	0.0000
17. GAMES |	1	5	234		-5	0	0	0.0000	0.0000	25	0	0	0.0000	0.0000
18. The		1	2345		-11	0	0	0.0000	0.0000	26	0	0	0.0000	0.0000
19. Atari	2	5	134		-5	0	0	0.0000	0.0000	28	0	0	0.0000	0.0000
20. FreeOn		5	234	1	-8	0	0	0.0000	0.0000	30	0	0	0.0000	0.0000
21. Bingo |	1		2345		-9	0	0	0.0000	0.0000	31	0	0	0.0000	0.0000
22. Flash	1		2345		-9	0	0	0.0000	0.0000	32	0	0	0.0000	0.0000
23. Armor			2345	1	-12	0	0	0.0000	0.0000	33	0	0	0.0000	0.0000
24. Miniclip			2345	1	-12	0	0	0.0000	0.0000	34	0	0	0.0000	0.0000
25. Flash	1		2345		-9	0	0	0.0000	0.0000	35	0	0	0.0000	0.0000
27. Down	1	5	234		-5	0	0	0.0000	0.0000	36	0	0	0.0000	0.0000
28. Gamesto			2345	1	-12	0	0	0.0000	0.0000	37	0	0	0.0000	0.0000
29. All (http:		5	234	1	-8	0	0	0.0000	0.0000	38	0	0	0.0000	0.0000
30. Game	1	35	24		-1	0	0	0.0000	0.0000	39	0	0	0.0000	0.0000
31. Daily	1	5	234		-5	0	0	0.0000	0.0000	40	0	0	0.0000	0.0000
32. AGAME	1		2345		-9	0	0	0.0000	0.0000	41	0	0	0.0000	0.0000
33. Downl	1		2345		-9	0	0	0.0000	0.0000	42	0	0	0.0000	0.0000
34. Big Fish			2345	1	-12	0	0	0.0000	0.0000	43	0	0	0.0000	0.0000
35. ArcadeTo			2345	1	-12	0	0	0.0000	0.0000	44	0	0	0.0000	0.0000
36. Neopets		5	234	1	-8	0	0	0.0000	0.0000	45	0	0	0.0000	0.0000
37. Multipla	1		2345		-9	0	0	0.0000	0.0000	47	1	1	0.5000	0.0278
38. Free	1		2345		-9	0	0	0.0000	0.0000	48	0	0	0.0000	0.0000
39. Andkon	1		2345		-9	0	0	0.0000	0.0000	49	0	0	0.0000	0.0000
40. Games.	2	15	34		-1	0	0	0.0000	0.0000	9	0	0	0.0000	0.0000
41. 2K Game			12345		-15	0	0	0.0000	0.0000	11	1	2	1.0000	0.0500
42. Action			12345		-15	0	0	0.0000	0.0000	5	0	0	0.0000	0.0000
43. Apple.co	1		2345		-9	0	0	0.0000	0.0000	12	0	0	0.0000	0.0000
44. iWon	1		2345		-9	0	0	0.0000	0.0000	14	0	0	0.0000	0.0000
45. Cartoon	1		2345		-9	0	0	0.0000	0.0000	15	0	0	0.0000	0.0000
46. Sandlot			2345	1	-12	0	0	0.0000	0.0000	17	0	0	0.0000	0.0000
47. Microsoft		2345		1	4	1	2	1.0000	0.0435	19	0	0	0.0000	0.0000
48. Y8.com -			2345	1	-12	0	0	0.0000	0.0000	27	0	0	0.0000	0.0000
49. Text			2345	1	-12	0	0	0.0000	0.0000	29	0	0	0.0000	0.0000
50. Play			2345	1	-12	0	0	0.0000	0.0000	46	0	0	0.0000	0.0000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL': number of relevant document in new ranked results, Rc': recall of new results, Pr': precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	9.1	9.1	9.1	9.1	9.1	4.4	4.4	4.4	4.4	4.4
Pr Bu	5	5	5	5	5	5	5	5	5	5

JN: 110; AN: 2; JCD: 0.0182; Impr: -1.75



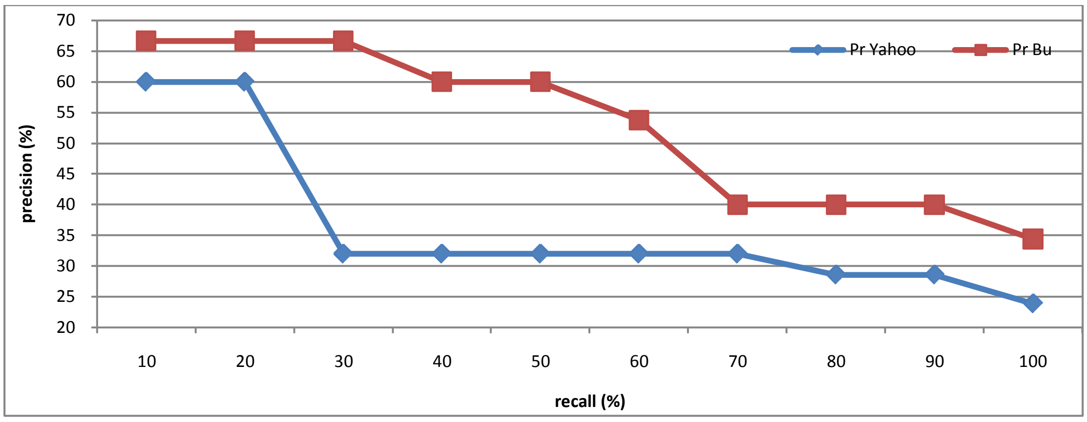
Search-term G4-3: quotes; Info need: How to use quotes in writing?

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL’	Rc’	Pr’
1. Quoteland		125	34		-3	0	0	0.0000	0.0000	15	0	0	0.0000	0.0000
2. Famous	12	5	34		1	1	1	0.0909	0.5000	17	1	1	0.0909	0.5000
3. Wisdom		124	35		-3	0	0	0.0000	0.0000	19	0	0	0.0000	0.0000
4. The Quote	12	45	3		5	1	2	0.1818	0.5000	21	1	2	0.1818	0.5000
5. Famous	15	2	34		1	1	3	0.2727	0.6000	22	1	3	0.2727	0.6000
6. QuoteGeek	1	25	34		-1	0	0	0.0000	0.0000	25	1	4	0.3636	0.6667
7. Quotes and	15		234		-3	0	0	0.0000	0.0000	28	0	0	0.0000	0.0000
8. Quotations	1	25	34		-1	0	0	0.0000	0.0000	32	0	0	0.0000	0.0000
9. Bartlett's		245	3	1	-1	0	0	0.0000	0.0000	34	1	5	0.4545	0.5556
10. Yahoo!			2345	1	-12	0	0	0.0000	0.0000	35	1	6	0.5455	0.6000
11. Quotes of		5	234	1	-8	0	0	0.0000	0.0000	45	0	0	0.0000	0.0000
12. Quote		4	235	1	-8	0	0	0.0000	0.0000	47	0	0	0.0000	0.0000
13. Quotes -		145	23		-3	0	0	0.0000	0.0000	2	1	7	0.6364	0.5385
14. Famous	1	245	3		3	1	4	0.3636	0.2857	6	0	0	0.0000	0.0000
15. Famous		145	23		-3	0	0	0.0000	0.0000	12	0	0	0.0000	0.0000
16. Quote.c		5	1234		-11	0	0	0.0000	0.0000	18	0	0	0.0000	0.0000
17. Great-	1	245	3		3	1	5	0.4545	0.2941	24	0	0	0.0000	0.0000
18. Photobuc	1	5	234		-5	0	0	0.0000	0.0000	41	0	0	0.0000	0.0000
19. Quotela		15	234		-7	0	0	0.0000	0.0000	42	0	0	0.0000	0.0000
20. Quotes.n		245	13		-3	0	0	0.0000	0.0000	46	1	8	0.7273	0.4000
21. Amusing		245	3	1	1	1	6	0.5455	0.2857	49	0	0	0.0000	0.0000
22. AllGreat	1	245	3		3	1	7	0.6364	0.3182	1	0	0	0.0000	0.0000
23. HolliesQu		124	35		-3	0	0	0.0000	0.0000	3	0	0	0.0000	0.0000
24. Quotes			12345		-15	0	0	0.0000	0.0000	4	1	9	0.8182	0.3750
25. Motivatio	1	245	3		3	1	8	0.7273	0.3200	5	1	10	0.9091	0.4000
26. Myspace	1	5	234		-5	0	0	0.0000	0.0000	7	0	0	0.0000	0.0000
27. twilight	1		2345		-9	0	0	0.0000	0.0000	8	0	0	0.0000	0.0000
28. Words of			12345		-15	0	0	0.0000	0.0000	9	0	0	0.0000	0.0000
29. GoodQu		25	34	1	-4	0	0	0.0000	0.0000	10	0	0	0.0000	0.0000
30. Inspiring	2		34	15	-3	0	0	0.0000	0.0000	11	0	0	0.0000	0.0000
31. Compare	1		2345		-9	0	0	0.0000	0.0000	13	0	0	0.0000	0.0000
32. BigCharts	1		2345		-9	0	0	0.0000	0.0000	14	1	11	1.0000	0.3438
33. Stock	1		2345		-9	0	0	0.0000	0.0000	16	0	0	0.0000	0.0000
34. World of	15	24	3		5	1	9	0.8182	0.2647	20	0	0	0.0000	0.0000
35. Quotatio		1245	3		1	1	10	0.9091	0.2857	23	0	0	0.0000	0.0000
36. Quotes		245	13		-3	0	0	0.0000	0.0000	26	0	0	0.0000	0.0000
37. Quotes -		5	234	1	-8	0	0	0.0000	0.0000	27	0	0	0.0000	0.0000
38. Funny	1		2345		-9	0	0	0.0000	0.0000	29	0	0	0.0000	0.0000
39. Famous		1	2345		-11	0	0	0.0000	0.0000	30	0	0	0.0000	0.0000
40. futureso			12345		-15	0	0	0.0000	0.0000	31	0	0	0.0000	0.0000
41. Fire Hot		2	345	1	-8	0	0	0.0000	0.0000	33	0	0	0.0000	0.0000
42. The Frie		15	234		-7	0	0	0.0000	0.0000	36	0	0	0.0000	0.0000
43. (RIO) -		15	234		-7	0	0	0.0000	0.0000	37	0	0	0.0000	0.0000
44. Quotes	5	1	234		-5	0	0	0.0000	0.0000	38	0	0	0.0000	0.0000
45. Bright	1	25	34		-1	0	0	0.0000	0.0000	39	0	0	0.0000	0.0000
46. Famous	1	245	3		3	1	11	1.0000	0.2391	40	0	0	0.0000	0.0000
47. Mot	1	2	34	5	-2	0	0	0.0000	0.0000	43	0	0	0.0000	0.0000
48. (MNC) -	1	5	234		-5	0	0	0.0000	0.0000	44	0	0	0.0000	0.0000
49. The	1	5	234		-5	0	0	0.0000	0.0000	48	0	0	0.0000	0.0000
50. Positive		2	34	15	-5	0	0	0.0000	0.0000	50	0	0	0.0000	0.0000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL’: number of relevant document in new ranked results, Rc’: recall of new results, Pr’: precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	60	60	32	32	32	32	32	28.6	28.6	23.9
Pr Bu	66.7	66.7	66.7	60	60	53.8	40	40	40	34.4

JN: 125; AN: 3; JCD: 0.024; Impr: 16.72



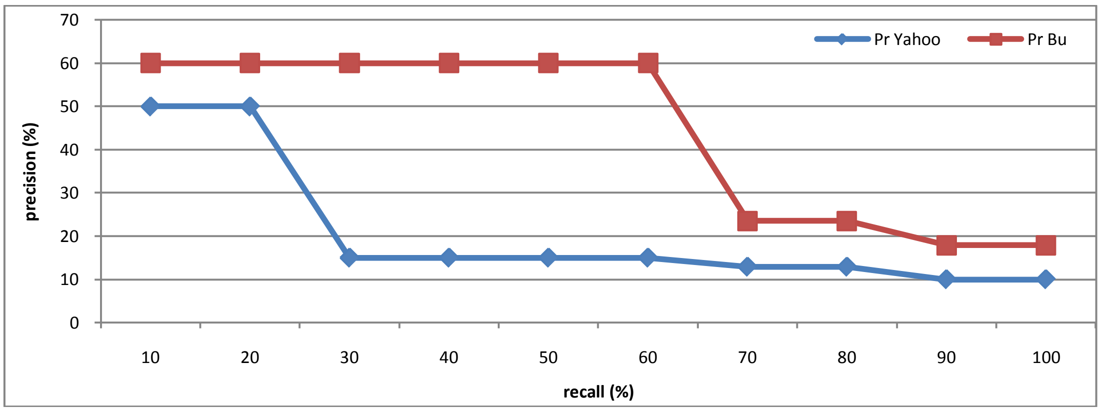
Search-term G4-4: trec; Info need: Text REtreivel Conference

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL'	Rc'	Pr'
1. Texas Real			2345	1	-12	0	0	0.0000	0.0000	1	0	0	0.0000	0.0000
2. Text	2345			1	12	1	1	0.2000	0.5000	13	0	0	0.0000	0.0000
3. TREC -	1		2345		-9	0	0	0.0000	0.0000	16	1	1	0.2000	0.3333
4. Forensic	1		2345		-9	0	0	0.0000	0.0000	50	1	2	0.4000	0.5000
5. Home:	12		345		-3	0	0	0.0000	0.0000	2	1	3	0.6000	0.6000
6. trec.nist.g	1		4	235	-1	0	0	0.0000	0.0000	5	0	0	0.0000	0.0000
7. TREC -	1		2345		-9	0	0	0.0000	0.0000	7	0	0	0.0000	0.0000
8. Teens of	1		2345		-9	0	0	0.0000	0.0000	10	0	0	0.0000	0.0000
9. TREC	2		1345		-9	0	0	0.0000	0.0000	12	0	0	0.0000	0.0000
10. Tropical		1	2345		-11	0	0	0.0000	0.0000	22	0	0	0.0000	0.0000
11. trec.coop		1	345	2	-8	0	0	0.0000	0.0000	30	0	0	0.0000	0.0000
12. homepage	1		2345		-9	0	0	0.0000	0.0000	33	0	0	0.0000	0.0000
13. TREC Inc		1	2345		-11	0	0	0.0000	0.0000	34	0	0	0.0000	0.0000
14. Trec			12345		-15	0	0	0.0000	0.0000	47	0	0	0.0000	0.0000
15. TREC-			2345	1	-12	0	0	0.0000	0.0000	49	0	0	0.0000	0.0000
16. Text	2345			1	12	1	2	0.4000	0.1250	6	0	0	0.0000	0.0000
17. Zahid			2345	1	-12	0	0	0.0000	0.0000	20	1	4	0.8000	0.2353
18. Center for	1		2345		-9	0	0	0.0000	0.0000	3	0	0	0.0000	0.0000
19. Trec	1		2345		-9	0	0	0.0000	0.0000	26	0	0	0.0000	0.0000
20. Text	1235	4			13	1	3	0.6000	0.1500	35	0	0	0.0000	0.0000
21. Real	1		2345		-9	0	0	0.0000	0.0000	36	0	0	0.0000	0.0000
22. Therap	1		2345		-9	0	0	0.0000	0.0000	38	0	0	0.0000	0.0000
23. TREC		1	345	2	-8	0	0	0.0000	0.0000	43	0	0	0.0000	0.0000
24. Las		1	2345		-11	0	0	0.0000	0.0000	44	0	0	0.0000	0.0000
25. Toronto	1		2345		-9	0	0	0.0000	0.0000	4	0	0	0.0000	0.0000
26. Home -	1		234	5	-6	0	0	0.0000	0.0000	8	0	0	0.0000	0.0000
27. TREC	1		2345		-9	0	0	0.0000	0.0000	23	0	0	0.0000	0.0000
28. Trec		1	234	5	-8	0	0	0.0000	0.0000	31	1	5	1.0000	0.1786
29. DES			1234	5	-12	0	0	0.0000	0.0000	39	0	0	0.0000	0.0000
30. Welcome		1	2345		-11	0	0	0.0000	0.0000	46	0	0	0.0000	0.0000
31. TREC	25	1	34		1	1	4	0.8000	0.1290	9	0	0	0.0000	0.0000
32. TREC-	1		34	25	-3	0	0	0.0000	0.0000	27	0	0	0.0000	0.0000
33. TREC		1	2345		-11	0	0	0.0000	0.0000	15	0	0	0.0000	0.0000
34. Tenne			12345		-15	0	0	0.0000	0.0000	45	0	0	0.0000	0.0000
35. TREC			12345		-15	0	0	0.0000	0.0000	18	0	0	0.0000	0.0000
36. :: ITRE ::	1		2345		-9	0	0	0.0000	0.0000	40	0	0	0.0000	0.0000
37. Teachers		1	234	5	-8	0	0	0.0000	0.0000	11	0	0	0.0000	0.0000
38. South			12345		-15	0	0	0.0000	0.0000	14	0	0	0.0000	0.0000
39. UMM |	12		345		-3	0	0	0.0000	0.0000	17	0	0	0.0000	0.0000
40. TREC	1		2345		-9	0	0	0.0000	0.0000	19	0	0	0.0000	0.0000
41. BM TRE	1		2345		-9	0	0	0.0000	0.0000	21	0	0	0.0000	0.0000
42. First		1	345	2	-8	0	0	0.0000	0.0000	24	0	0	0.0000	0.0000
43. TREC			12345		-15	0	0	0.0000	0.0000	25	0	0	0.0000	0.0000
44. trec-			12345		-15	0	0	0.0000	0.0000	28	0	0	0.0000	0.0000
45. TREC -		1	2345		-11	0	0	0.0000	0.0000	29	0	0	0.0000	0.0000
46. A Renow	1		2345		-9	0	0	0.0000	0.0000	32	0	0	0.0000	0.0000
47. TREC -	2	1	345		-5	0	0	0.0000	0.0000	37	0	0	0.0000	0.0000
48. Registr		2	1345		-11	0	0	0.0000	0.0000	41	0	0	0.0000	0.0000
49. Charter l			2345	1	-12	0	0	0.0000	0.0000	42	0	0	0.0000	0.0000
50. Spoken	235		4	1	6	1	5	1.0000	0.1000	48	0	0	0.0000	0.0000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL': number of relevant document in new ranked results, Rc': recall of new results, Pr': precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	50	50	15	15	15	15	12.9	12.9	10	10
Pr Bu	60	60	60	60	60	60	23.5	23.5	17.9	17.9

JN: 105; AN: 6; JCD: 0.0571; Impr: 23.7



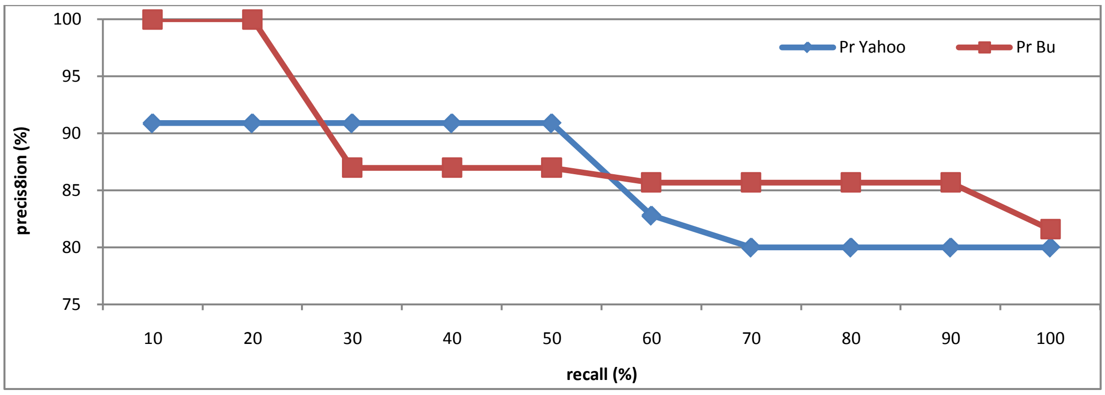
Search-term G4-5: world war 2; Info need: history related to world war 2

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL’	Rc’	Pr’
1. World War	12345				15	1	1	0.0250	1.0000	1	1	1	0.0250	1.0000
2. World War	2345	1			13	1	2	0.0500	1.0000	5	1	2	0.0500	1.0000
3. WORLD	345	12			11	1	3	0.0750	1.0000	11	1	3	0.0750	1.0000
4. World War		123	45		-3	0	0	0.0000	0.0000	14	1	4	0.1000	1.0000
5. World War	2345	1			13	1	4	0.1000	0.8000	16	1	5	0.1250	1.0000
6. World War	2345	1			13	1	5	0.1250	0.8333	20	1	6	0.1500	1.0000
7. EyeWitnes	235	14			11	1	6	0.1500	0.8571	21	1	7	0.1750	1.0000
8. National	25	13	4		5	1	7	0.1750	0.8750	22	1	8	0.2000	1.0000
9. World War	2	1345			9	1	8	0.2000	0.8889	26	1	9	0.2250	1.0000
10. National	2	1	34	5	-2	0	0	0.0000	0.0000	28	0	0	0.0000	0.0000
11. World	24	13		5	8	1	9	0.2250	0.8182	30	0	0	0.0000	0.0000
12. World	235	14			11	1	10	0.2500	0.8333	33	1	10	0.2500	0.8333
13. World	123	4	5		7	1	11	0.2750	0.8462	34	1	11	0.2750	0.8462
14. Military	25	134			9	1	12	0.3000	0.8571	36	0	0	0.0000	0.0000
15. World	123	45			11	1	13	0.3250	0.8667	37	1	12	0.3000	0.8000
16. BBC -	1235	4			13	1	14	0.3500	0.8750	39	1	13	0.3250	0.8125
17. World	1235	4			13	1	15	0.3750	0.8824	41	1	14	0.3500	0.8235
18. World	2345			1	12	1	16	0.4000	0.8889	42	1	15	0.3750	0.8333
19. World	2	345		1	6	1	17	0.4250	0.8947	43	1	16	0.4000	0.8421
20. World	15	234			9	1	18	0.4500	0.9000	44	1	17	0.4250	0.8500
21. World	15	234			9	1	19	0.4750	0.9048	45	1	18	0.4500	0.8571
22. World	25	134			9	1	20	0.5000	0.9091	46	1	19	0.4750	0.8636
23. World	245	3	1		7	1	21	0.5250	0.9130	48	1	20	0.5000	0.8696
24. World	2	1	34	5	-2	0	0	0.0000	0.0000	2	1	21	0.5250	0.8750
25. WW2 -	123	4	5		7	1	22	0.5500	0.8800	4	0	0	0.0000	0.0000
26. Italy in	25	13	4		5	1	23	0.5750	0.8846	6	1	22	0.5500	0.8462
27. World		3	245	1	-8	0	0	0.0000	0.0000	8	1	23	0.5750	0.8519
28. World		135	24		-3	0	0	0.0000	0.0000	10	0	0	0.0000	0.0000
29. World	24	135			9	1	24	0.6000	0.8276	13	1	24	0.6000	0.8276
30. The		124	35		-3	0	0	0.0000	0.0000	17	1	25	0.6250	0.8333
31. World		23	14	5	-4	0	0	0.0000	0.0000	18	1	26	0.6500	0.8387
32. BrainPOP	4	1	235		-5	0	0	0.0000	0.0000	19	1	27	0.6750	0.8438
33. World	12345				15	1	25	0.6250	0.7576	24	0	0	0.0000	0.0000
34. 2 World	35	24	1		5	1	26	0.6500	0.7647	25	1	28	0.7000	0.8235
35. World	125	34			11	1	27	0.6750	0.7714	29	1	29	0.7250	0.8286
36. World		4	3	125	-2	0	0	0.0000	0.0000	40	1	30	0.7500	0.8333
37. World	12	345			9	1	28	0.7000	0.7568	50	1	31	0.7750	0.8378
38. World	235	14			11	1	29	0.7250	0.7632	3	1	32	0.8000	0.8421
39. World	25	34	1		5	1	30	0.7500	0.7692	7	1	33	0.8250	0.8462
40. Causes of	1235	4			13	1	31	0.7750	0.7750	9	1	34	0.8500	0.8500
41. Hyper	125	3	4		7	1	32	0.8000	0.7805	12	1	35	0.8750	0.8537
42. BBC -	12345				15	1	33	0.8250	0.7857	15	1	36	0.9000	0.8571
43. Books	15	3	24		1	1	34	0.8500	0.7907	23	1	37	0.9250	0.8605
44. World	1235	4			13	1	35	0.8750	0.7955	27	0	0	0.0000	0.0000
45. World	125	34			11	1	36	0.9000	0.8000	31	0	0	0.0000	0.0000
46. World	12	345			9	1	37	0.9250	0.8043	32	0	0	0.0000	0.0000
47. World	1235	4			13	1	38	0.9500	0.8085	35	1	38	0.9500	0.8085
48. World	25	134			9	1	39	0.9750	0.8125	38	1	39	0.9750	0.8125
49. The	5	2	134		-5	0	0	0.0000	0.0000	47	1	40	1.0000	0.8163
50. BIGpedia	1235	4			13	1	40	1.0000	0.8000	49	0	0	0.0000	0.0000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL’: number of relevant document in new ranked results, Rc’: recall of new results, Pr’: precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	90.9	90.9	90.9	90.9	90.9	82.8	80	80	80	80
Pr Bu	100	100	87	87	87	85.7	85.7	85.7	85.7	81.6

JN: 105; AN: 6; JCD: 0.0571; Impr: 23.7



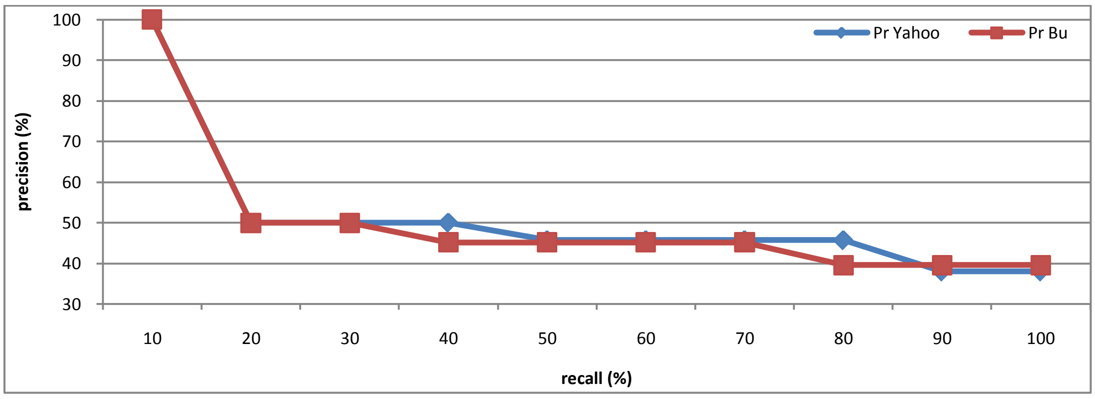
Search-term G5-1: graphic design; Info need: the art & practice of graphical design

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL'	Rc'	Pr'
1. Graphic	2345		1		9	1	1	0.0526	1.0000	1	1	1	0.0526	1.0000
2. Graphic	1345	2			13	1	2	0.1053	1.0000	2	1	2	0.1053	1.0000
3. Add		2	134	5	-8	0	0	0.0000	0.0000	3	0	0	0.0000	0.0000
4. Breez Gra	5	24	13		-1	0	0	0.0000	0.0000	4	0	0	0.0000	0.0000
5. Design	25	1	34		1	1	3	0.1579	0.6000	6	0	0	0.0000	0.0000
6. All Gra	4	25	13		-1	0	0	0.0000	0.0000	7	0	0	0.0000	0.0000
7. Fanshawe		2	1345		-11	0	0	0.0000	0.0000	11	1	3	0.1579	0.4286
8. Graphic	245	1	3		7	1	4	0.2105	0.5000	12	1	4	0.2105	0.5000
9. U.S. Depa		2	134	5	-8	0	0	0.0000	0.0000	13	1	5	0.2632	0.5556
10. Graphic	2		134	5	-6	0	0	0.0000	0.0000	14	0	0	0.0000	0.0000
11. What is	14	25	3		5	1	5	0.2632	0.4545	15	0	0	0.0000	0.0000
12. Graphic	124	35			11	1	6	0.3158	0.5000	16	1	6	0.3158	0.5000
13. SCAD	124	35			11	1	7	0.3684	0.5385	17	1	7	0.3684	0.5385
14. RP	2	34	15		-1	0	0	0.0000	0.0000	18	0	0	0.0000	0.0000
15. Virginia	2	14	35		-1	0	0	0.0000	0.0000	19	0	0	0.0000	0.0000
16. Graphic	25	4	13		1	1	8	0.4211	0.5000	20	0	0	0.0000	0.0000
17. Renaissa	12	5	34		1	1	9	0.4737	0.5294	21	0	0	0.0000	0.0000
18. Ablaze		2	1345		-11	0	0	0.0000	0.0000	22	0	0	0.0000	0.0000
19. Profess		123	45		-3	0	0	0.0000	0.0000	23	1	8	0.4211	0.4211
20. Miami		24	13	5	-4	0	0	0.0000	0.0000	24	0	0	0.0000	0.0000
21. Claudia		2	14	35	-5	0	0	0.0000	0.0000	25	0	0	0.0000	0.0000
22. Graphic	2	45	13		-1	0	0	0.0000	0.0000	26	1	9	0.4737	0.4091
23. 51 New	24	5	13		1	1	10	0.5263	0.4348	27	1	10	0.5263	0.4348
24. Design 26		235	14		-3	0	0	0.0000	0.0000	28	1	11	0.5789	0.4583
25. Graphic		24	135		-7	0	0	0.0000	0.0000	29	0	0	0.0000	0.0000
26. Graphic	24	5	13		1	1	11	0.5789	0.4231	30	0	0	0.0000	0.0000
27. Graphic	124	35			11	1	12	0.6316	0.4444	31	0	0	0.0000	0.0000
28. HW		2345	1		1	1	13	0.6842	0.4643	32	0	0	0.0000	0.0000
29. UArts		24	13	5	-4	0	0	0.0000	0.0000	33	1	12	0.6316	0.4138
30. KJ	25		134		-3	0	0	0.0000	0.0000	34	1	13	0.6842	0.4333
31. Creative	2	4	13	5	-2	0	0	0.0000	0.0000	35	1	14	0.7368	0.4516
32. PLP		23	14	5	-4	0	0	0.0000	0.0000	36	1	15	0.7895	0.4688
33. Graphic	4	235	1		3	1	14	0.7368	0.4242	37	0	0	0.0000	0.0000
34. Graphic	24	35	1		5	1	15	0.7895	0.4412	38	0	0	0.0000	0.0000
35. Graphic	24	3	1	5	4	1	16	0.8421	0.4571	39	0	0	0.0000	0.0000
36. Daniel	12		4	35	3	1	17	0.8947	0.4722	40	0	0	0.0000	0.0000
37. AB				12345	0	0	0	0.0000	0.0000	41	0	0	0.0000	0.0000
38. SC		2	134	5	-8	0	0	0.0000	0.0000	43	0	0	0.0000	0.0000
39. Graphic		24	13	5	-4	0	0	0.0000	0.0000	44	0	0	0.0000	0.0000
40. kb		2	134	5	-8	0	0	0.0000	0.0000	45	0	0	0.0000	0.0000
41. KP				12345	0	0	0	0.0000	0.0000	46	0	0	0.0000	0.0000
42. CP		2	134	5	-8	0	0	0.0000	0.0000	47	0	0	0.0000	0.0000
43. JLM		2	134	5	-8	0	0	0.0000	0.0000	48	1	16	0.8421	0.3721
44. Eating		123	4	5	-1	0	0	0.0000	0.0000	49	0	0	0.0000	0.0000
45. GittyUP		2	134	5	-8	0	0	0.0000	0.0000	50	1	17	0.8947	0.3778
46. YouTube	5	4	123		-5	0	0	0.0000	0.0000	5	1	18	0.9474	0.3913
47. RTE			145	23	-9	0	0	0.0000	0.0000	10	0	0	0.0000	0.0000
48. Designsc	1245	3			13	1	18	0.9474	0.3750	8	1	19	1.0000	0.3958
49. Blue Iris	12		34	5	-1	0	0	0.0000	0.0000	9	0	0	0.0000	0.0000
50. Amazon.	234	1		5	10	1	19	1.0000	0.3800	42	0	0	0.0000	0.0000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL': number of relevant document in new ranked results, Rc': recall of new results, Pr': precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	100	50	50	50	45.7	45.7	45.7	45.7	38	38
Pr Bu	100	50	50	45.2	45.2	45.2	45.2	39.6	39.6	39.6

JN: 134; AN: 2; JCD: 0.0149; Impr: -0.92



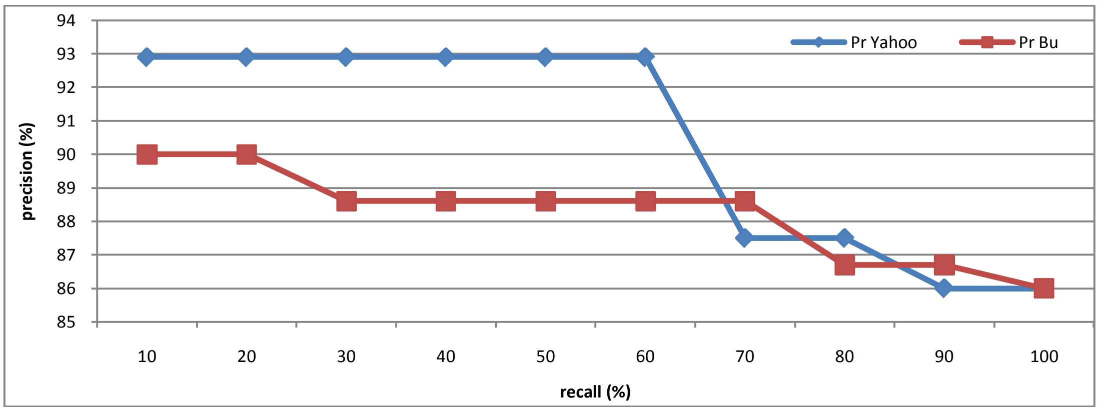
Search-term G5-2: jokes; Info need: the most funny jokes

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL'	Rc'	Pr'
1. Stand Up	345		12		3	1	1	0.0233	1.0000	3	1	1	0.0233	1.0000
2. texte		34	125		-7	0	0	0.0000	0.0000	6	1	2	0.0465	1.0000
3. Aha! Jokes	345		1	2	6	1	2	0.0465	0.6667	8	1	3	0.0698	1.0000
4. Jackass	345		12		3	1	3	0.0698	0.7500	11	1	4	0.0930	1.0000
5. 101 Fun	1345		2		9	1	4	0.0930	0.8000	17	0	0	0.0000	0.0000
6. Jokes	345		12		3	1	5	0.1163	0.8333	18	1	5	0.1163	0.8333
7. Jokes.Net:	345		12		3	1	6	0.1395	0.8571	19	1	6	0.1395	0.8571
8. Funny and	345	1		2	10	1	7	0.1628	0.8750	23	1	7	0.1628	0.8750
9. Jokes	1345		2		9	1	8	0.1860	0.8889	24	1	8	0.1860	0.8889
10. Funny	34	15	2		5	1	9	0.2093	0.9000	25	1	9	0.2093	0.9000
11. The-	45	1	2	3	4	1	10	0.2326	0.9091	26	1	10	0.2326	0.9091
12. Clean	345	1	2		7	1	11	0.2558	0.9167	27	1	11	0.2558	0.9167
13. Jokes -	345		12		3	1	12	0.2791	0.9231	29	0	0	0.0000	0.0000
14. Jokes	345		12		3	1	13	0.3023	0.9286	32	1	12	0.2791	0.8571
15. Best	1345		2		9	1	14	0.3256	0.9333	41	1	13	0.3023	0.8667
16. Jokes in	45	13	2		5	1	15	0.3488	0.9375	43	0	0	0.0000	0.0000
17. Jokes2Go		345	12		-3	0	0	0.0000	0.0000	45	1	14	0.3256	0.8235
18. Funny	345		1	2	6	1	16	0.3721	0.8889	46	1	15	0.3488	0.8333
19. Jokes For	45	13	2		5	1	17	0.3953	0.8947	47	1	16	0.3721	0.8421
20. Funny	345		12		3	1	18	0.4186	0.9000	49	1	17	0.3953	0.8500
21. AhaJokes	45	3	12		1	1	19	0.4419	0.9048	1	1	18	0.4186	0.8571
22. Funny	145	3	2		7	1	20	0.4651	0.9091	7	1	19	0.4419	0.8636
23. Funny	345	1	2		7	1	21	0.4884	0.9130	10	1	20	0.4651	0.8696
24. Jokes and	1345			2	12	1	22	0.5116	0.9167	14	1	21	0.4884	0.8750
25. HQ	1345			2	12	1	23	0.5349	0.9200	20	1	22	0.5116	0.8800
26. Jokes -	145	3		2	10	1	24	0.5581	0.9231	35	1	23	0.5349	0.8846
27. e-Jokes -	45	3	1	2	4	1	25	0.5814	0.9259	48	1	24	0.5581	0.8889
28. Funny	345		12		3	1	26	0.6047	0.9286	2	0	0	0.0000	0.0000
29. Jokes4us	45		123		-3	0	0	0.0000	0.0000	4	1	25	0.5814	0.8621
30. Pirate	5	34	12		-1	0	0	0.0000	0.0000	5	1	26	0.6047	0.8667
31. Joke -	5	3	124		-5	0	0	0.0000	0.0000	9	1	27	0.6279	0.8710
32. Asian	45	13		2	8	1	27	0.6279	0.8438	12	1	28	0.6512	0.8750
33. Amazing	345		12		3	1	28	0.6512	0.8485	13	1	29	0.6744	0.8788
34. Italian	345		12		3	1	29	0.6744	0.8529	15	1	30	0.6977	0.8824
35. emailAJo	35	4	12		1	1	30	0.6977	0.8571	16	1	31	0.7209	0.8857
36. Mighty	345		1	2	6	1	31	0.7209	0.8611	21	1	32	0.7442	0.8889
37. Hilarious	345		12		3	1	32	0.7442	0.8649	22	1	33	0.7674	0.8919
38. TooFunn	1345		2		9	1	33	0.7674	0.8684	28	1	34	0.7907	0.8947
39. bdjokes ::	35	4	12		1	1	34	0.7907	0.8718	30	0	0	0.0000	0.0000
40. Jokes and	345		12		3	1	35	0.8140	0.8750	31	0	0	0.0000	0.0000
41. HAWW	45	3	12		1	1	36	0.8372	0.8780	33	1	35	0.8140	0.8537
42. The	345		12		3	1	37	0.8605	0.8810	34	1	36	0.8372	0.8571
43. Jokestan	5	34	12		-1	0	0	0.0000	0.0000	36	1	37	0.8605	0.8605
44. Clean	5	34	12		-1	0	0	0.0000	0.0000	37	1	38	0.8837	0.8636
45. MyJokes	1345		2		9	1	38	0.8837	0.8444	38	1	39	0.9070	0.8667
46. My jokes	345		12		3	1	39	0.9070	0.8478	39	1	40	0.9302	0.8696
47. Jokes and	1345		2		9	1	40	0.9302	0.8511	40	1	41	0.9535	0.8723
48. Funny	1345		2		9	1	41	0.9535	0.8542	42	1	42	0.9767	0.8750
49. Crazy	145	3	2		7	1	42	0.9767	0.8571	44	0	0	0.0000	0.0000
50. Funny	345	1		2	10	1	43	1.0000	0.8600	50	1	43	1.0000	0.8600

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL': number of relevant document in new ranked results, Rc': recall of new results, Pr': precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	92.9	92.9	92.9	92.9	92.9	92.9	87.5	87.5	86	86
Pr Bu	90	90	88.6	88.6	88.6	88.6	88.6	86.7	86.7	86

JN: 126; AN: 0; JCD: 0; Impr: -2.2



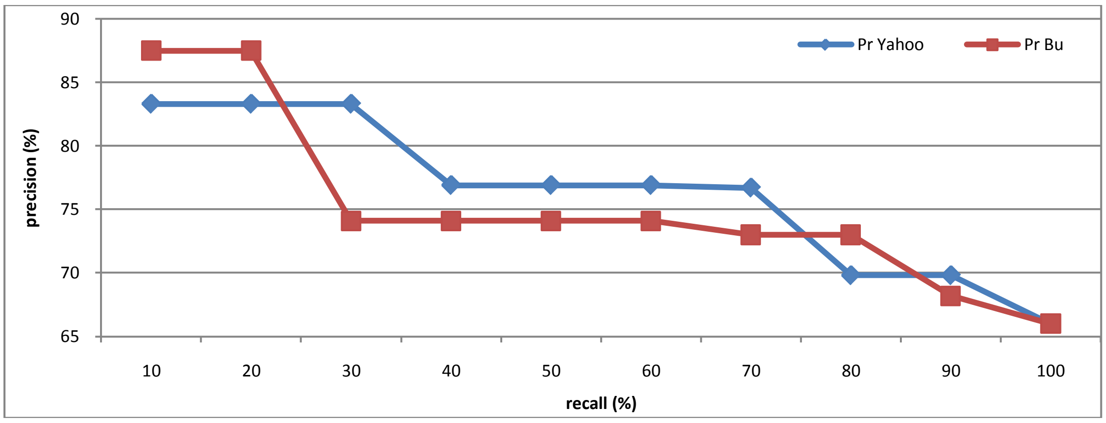
Search-term G5-3: resume; Info need: How to write a resume

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL'	Rc'	Pr'
1. Resumes -	1345			2	12	1	1	0.0303	1.0000	2	0	0	0.0000	0.0000
2. Résumé -	5	4	13	2	-2	0	0	0.0000	0.0000	4	1	1	0.0303	0.5000
3. Get.	5	4	13	2	-2	0	0	0.0000	0.0000	5	1	2	0.0606	0.6667
4. Resume	345	1		2	10	1	2	0.0606	0.5000	7	1	3	0.0909	0.7500
5. Resume	1345			2	12	1	3	0.0909	0.6000	8	1	4	0.1212	0.8000
6. Entry	1345			2	12	1	4	0.1212	0.6667	11	1	5	0.1515	0.8333
7. Free	345	1		2	10	1	5	0.1515	0.7143	12	1	6	0.1818	0.8571
8. Resume -	1345			2	12	1	6	0.1818	0.7500	13	1	7	0.2121	0.8750
9. JobStar:	345	1		2	10	1	7	0.2121	0.7778	22	1	8	0.2424	0.8889
10. Free	1345			2	12	1	8	0.2424	0.8000	29	0	0	0.0000	0.0000
11. e-Resume	5	34	1	2	2	1	9	0.2727	0.8182	38	0	0	0.0000	0.0000
12. e-resume.	5	34	1	2	2	1	10	0.3030	0.8333	40	1	9	0.2727	0.7500
13. Professio	35	4	1	2	4	1	11	0.3333	0.8462	14	0	0	0.0000	0.0000
14. resume:		5	134	2	-8	0	0	0.0000	0.0000	17	0	0	0.0000	0.0000
15. Resume	145		3	2	6	1	12	0.3636	0.8000	19	1	10	0.3030	0.6667
16. e-Resume		45	13	2	-4	0	0	0.0000	0.0000	20	1	11	0.3333	0.6875
17. FaxRe		5	1234		-11	0	0	0.0000	0.0000	25	1	12	0.3636	0.7059
18. Post your			12345		-15	0	0	0.0000	0.0000	26	1	13	0.3939	0.7222
19. CV Res	145		3	2	6	1	13	0.3939	0.6842	27	1	14	0.4242	0.7368
20. Resume	5	14	3	2	2	1	14	0.4242	0.7000	42	1	15	0.4545	0.7500
21. Resume	35	4	1	2	4	1	15	0.4545	0.7143	49	0	0	0.0000	0.0000
22. Resume	1	345		2	6	1	16	0.4848	0.7273	1	1	16	0.4848	0.7273
23. Freshers	345		1	2	6	1	17	0.5152	0.7391	3	0	0	0.0000	0.0000
24. eResum	345		1	2	6	1	18	0.5455	0.7500	6	1	17	0.5152	0.7083
25. Resumes	5	34	1	2	2	1	19	0.5758	0.7600	9	1	18	0.5455	0.7200
26. Sample	135	4		2	10	1	20	0.6061	0.7692	10	1	19	0.5758	0.7308
27. Resume	145	3		2	10	1	21	0.6364	0.7778	15	1	20	0.6061	0.7407
28. Careers	5	14	3	2	2	1	22	0.6667	0.7857	16	0	0	0.0000	0.0000
29. Profes	3		145	2	-6	0	0	0.0000	0.0000	18	0	0	0.0000	0.0000
30. Free	1345			2	12	1	23	0.6970	0.7667	21	1	21	0.6364	0.7000
31. Resume		35	14	2	-4	0	0	0.0000	0.0000	23	1	22	0.6667	0.7097
32. Resume |	145	3		2	10	1	24	0.7273	0.7500	24	1	23	0.6970	0.7188
33. Resumes	5	34	1	2	2	1	25	0.7576	0.7576	28	1	24	0.7273	0.7273
34. Profess		5	134	2	-8	0	0	0.0000	0.0000	30	1	25	0.7576	0.7353
35. Basic -		45	123		-7	0	0	0.0000	0.0000	31	0	0	0.0000	0.0000
36. Best		34	125		-7	0	0	0.0000	0.0000	32	1	26	0.7879	0.7222
37. Introducti	15	34		2	8	1	26	0.7879	0.7027	33	1	27	0.8182	0.7297
38. RESUME			1345	2	-12	0	0	0.0000	0.0000	34	0	0	0.0000	0.0000
39. The Write	4		135	2	-6	0	0	0.0000	0.0000	35	0	0	0.0000	0.0000
40. What's	135	4		2	10	1	27	0.8182	0.6750	36	0	0	0.0000	0.0000
41. Resume	1345		2		9	1	28	0.8485	0.6829	37	1	28	0.8485	0.6829
42. How to	14	35		2	8	1	29	0.8788	0.6905	39	0	0	0.0000	0.0000
43. Resume	1345			2	12	1	30	0.9091	0.6977	41	1	29	0.8788	0.6744
44. Professio		3	145	2	-8	0	0	0.0000	0.0000	43	1	30	0.9091	0.6818
45. Resume		14	3	25	-1	0	0	0.0000	0.0000	44	0	0	0.0000	0.0000
46. Careers		345	1	2	1	1	31	0.9394	0.6739	45	0	0	0.0000	0.0000
47. resume		4	13	25	-5	0	0	0.0000	0.0000	46	1	31	0.9394	0.6596
48. Resume	1345			2	12	1	32	0.9697	0.6667	47	0	0	0.0000	0.0000
49. Create a		4	13	25	-5	0	0	0.0000	0.0000	48	1	32	0.9697	0.6531
50. How To	1345			2	12	1	33	1.0000	0.6600	50	1	33	1.0000	0.6600

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL': number of relevant document in new ranked results, Rc': recall of new results, Pr': precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	83.3	83.3	83.3	76.9	76.9	76.9	76.7	69.8	69.8	66
Pr Bu	87.5	87.5	74.1	74.1	74.1	74.1	73	73	68.2	66

JN: 144; AN: 1; JCD: 0.0069; Impr: -1.13



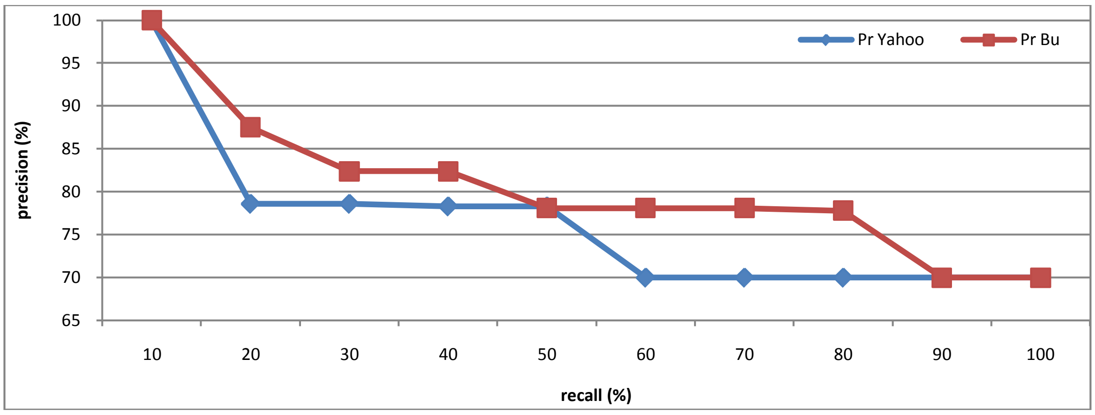
Search-term G5-4: time zone; Info need: time zone of the world

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL’	Rc’	Pr’
1. Official	5	234	1		3	1	1	0.0286	1.0000	2	1	1	0.0286	1.0000
2. The World	1345		2		9	1	2	0.0571	1.0000	18	1	2	0.0571	1.0000
3. Time zone	145	3	2		7	1	3	0.0857	1.0000	21	1	3	0.0857	1.0000
4. WorldTi	1345		2		9	1	4	0.1143	1.0000	38	1	4	0.1143	1.0000
5. TimeAn	1345		2		9	1	5	0.1429	1.0000	41	1	5	0.1429	1.0000
6. TimeZone	5	3	124		-5	0	0	0.0000	0.0000	43	1	6	0.1714	1.0000
7. time zone:	15	4	23		1	1	6	0.1714	0.8571	44	0	0	0.0000	0.0000
8. List of	5	34	12		-1	0	0	0.0000	0.0000	46	1	7	0.2000	0.8750
9. Time Zone	345		12		3	1	7	0.2000	0.7778	1	1	8	0.2286	0.8889
10. U.S. Ti		345	12		-3	0	0	0.0000	0.0000	3	1	9	0.2571	0.9000
11. Time	345		12		3	1	8	0.2286	0.7273	7	1	10	0.2857	0.9091
12. Downl	1345		2		9	1	9	0.2571	0.7500	8	0	0	0.0000	0.0000
13. Europe	45	3	12		1	1	10	0.2857	0.7692	10	0	0	0.0000	0.0000
14. World	1345		2		9	1	11	0.3143	0.7857	11	1	11	0.3143	0.7857
15. Time	45	1	2	3	4	1	12	0.3429	0.8000	13	1	12	0.3429	0.8000
16. current		45	12	3	-4	0	0	0.0000	0.0000	14	1	13	0.3714	0.8125
17. World-.	1345		2		9	1	13	0.3714	0.7647	17	1	14	0.4000	0.8235
18. World	134	5	2		7	1	14	0.4000	0.7778	19	1	15	0.4286	0.8333
19. Time z	1345		2		9	1	15	0.4286	0.7895	22	0	0	0.0000	0.0000
20. Online	1345		2		9	1	16	0.4571	0.8000	26	0	0	0.0000	0.0000
21. WorldTi	1345		2		9	1	17	0.4857	0.8095	28	0	0	0.0000	0.0000
22. King	3	45	12		-1	0	0	0.0000	0.0000	29	1	16	0.4571	0.7273
23. Microsoft		1345	2		1	1	18	0.5143	0.7826	30	1	17	0.4857	0.7391
24. The.	5	4	12	3	-2	0	0	0.0000	0.0000	34	1	18	0.5143	0.7500
25. Downl		13	24	5	-4	0	0	0.0000	0.0000	37	0	0	0.0000	0.0000
26. Understa	1	4	235		-5	0	0	0.0000	0.0000	40	1	19	0.5429	0.7308
27. Time		14	23	5	-4	0	0	0.0000	0.0000	42	1	20	0.5714	0.7407
28. Time	1	4	235		-5	0	0	0.0000	0.0000	45	1	21	0.6000	0.7500
29. Time	14	5	23		1	1	19	0.5429	0.6552	47	1	22	0.6286	0.7586
30. Current	345	1	2		7	1	20	0.5714	0.6667	48	1	23	0.6571	0.7667
31. Time		45	123		-7	0	0	0.0000	0.0000	4	1	24	0.6857	0.7742
32. Time		4	125	3	-8	0	0	0.0000	0.0000	5	1	25	0.7143	0.7813
33. NASA -	1345		2		9	1	21	0.6000	0.6364	6	0	0	0.0000	0.0000
34. Time	14		23	5	1	1	22	0.6286	0.6471	9	1	26	0.7429	0.7647
35. Time	14		23	5	1	1	23	0.6571	0.6571	12	1	27	0.7714	0.7714
36. Time	1	4	23	5	-2	0	0	0.0000	0.0000	15	1	28	0.8000	0.7778
37. Time	5	4	123		-5	0	0	0.0000	0.0000	16	0	0	0.0000	0.0000
38. Time	1345		2		9	1	24	0.6857	0.6316	20	1	29	0.8286	0.7632
39. TimeTi	345		12		3	1	25	0.7143	0.6410	23	1	30	0.8571	0.7692
40. TimeZo (	145		23		3	1	26	0.7429	0.6500	24	0	0	0.0000	0.0000
41. Taeger	5	14	2	3	2	1	27	0.7714	0.6585	25	0	0	0.0000	0.0000
42. World	1345		2		9	1	28	0.8000	0.6667	27	0	0	0.0000	0.0000
43. - World	1345		2		9	1	29	0.8286	0.6744	31	0	0	0.0000	0.0000
44. List of	1	4	23	5	-2	0	0	0.0000	0.0000	32	0	0	0.0000	0.0000
45. Time	1	4	2	35	1	1	30	0.8571	0.6667	33	1	31	0.8857	0.6889
46. Time and	1345		2		9	1	31	0.8857	0.6739	35	1	32	0.9143	0.6957
47. Time	145		23		3	1	32	0.9143	0.6809	36	0	0	0.0000	0.0000
48. thetime	1345		2		9	1	33	0.9429	0.6875	39	1	33	0.9429	0.6875
49. Time	145		23		3	1	34	0.9714	0.6939	49	1	34	0.9714	0.6939
50. TIME	145	3	2		7	1	35	1.0000	0.7000	50	1	35	1.0000	0.7000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL’: number of relevant document in new ranked results, Rc’: recall of new results, Pr’: precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	100	78.6	78.6	78.3	78.3	70	70	70	70	70
Pr Bu	100	87.5	82.4	82.4	78.1	78.1	78.1	77.8	70	70

JN: 132; AN: 0; JCD: 0; Impr: 4.06



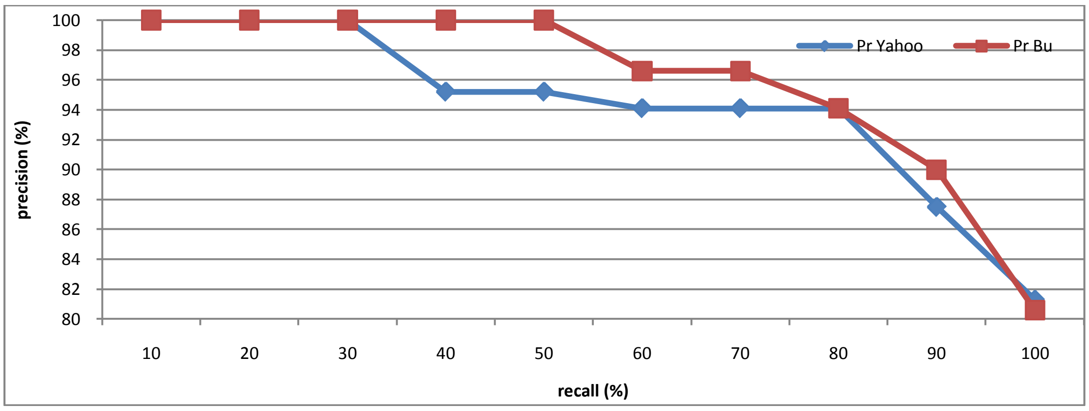
Search-term G5-5: travel; Info need: travel planning and preparation

W-S	R(3)	P(1)	I(-3)	N(0)	SC	JG	RL	Rc	Pr	NR	JG	RL’	Rc’	Pr’
1. Yahoo!	1345	2			13	1	1	0.0256	1.0000	17	1	1	0.0256	1.0000
2. Travelocity	34	25	1		5	1	2	0.0513	1.0000	21	1	2	0.0513	1.0000
3. AOL	1345	2			13	1	3	0.0769	1.0000	25	1	3	0.0769	1.0000
4. Cheap	345	2	1		7	1	4	0.1026	1.0000	33	1	4	0.1026	1.0000
5. Expedia	345	2	1		7	1	5	0.1282	1.0000	41	1	5	0.1282	1.0000
6. Travel	145	23			11	1	6	0.1538	1.0000	45	1	6	0.1538	1.0000
7. Orbitz -	45	23	1		5	1	7	0.1795	1.0000	1	1	7	0.1795	1.0000
8. Travel		23		145	2	1	8	0.2051	1.0000	6	1	8	0.2051	1.0000
9. Hotwire	4	235	1		3	1	9	0.2308	1.0000	8	1	9	0.2308	1.0000
10. Travel	345	12			11	1	10	0.2564	1.0000	11	1	10	0.2564	1.0000
11. AAA.co		23		145	2	1	11	0.2821	1.0000	22	1	11	0.2821	1.0000
12. STA	4	235	1		3	1	12	0.3077	1.0000	27	1	12	0.3077	1.0000
13. Travel	4	235	1		3	1	13	0.3333	1.0000	31	1	13	0.3333	1.0000
14. Travelzoo	4	25	13		-1	0	0	0.0000	0.0000	40	1	14	0.3590	1.0000
15. Priceline.	45	2	13		1	1	14	0.3590	0.9333	2	1	15	0.3846	1.0000
16. Travel	345	12			11	1	15	0.3846	0.9375	3	1	16	0.4103	1.0000
17. Travel -	345	2	1		7	1	16	0.4103	0.9412	4	1	17	0.4359	1.0000
18. SmarterT	345	2	1		7	1	17	0.4359	0.9444	5	1	18	0.4615	1.0000
19. Home |	345	2	1		7	1	18	0.4615	0.9474	7	1	19	0.4872	1.0000
20. New	35	24	1		5	1	19	0.4872	0.9500	9	1	20	0.5128	1.0000
21. Travel s	45	23	1		5	1	20	0.5128	0.9524	10	1	21	0.5385	1.0000
22. Travel	4	23	1	5	2	1	21	0.5385	0.9545	13	1	22	0.5641	1.0000
23. Liberty	4	235	1		3	1	22	0.5641	0.9565	14	0	0	0.0000	0.0000
24. American	4	25	13		-1	0	0	0.0000	0.0000	15	1	23	0.5897	0.9583
25. Oregon	15	24	3		5	1	23	0.5897	0.9200	16	1	24	0.6154	0.9600
26. Yahoo!	35	24	1		5	1	24	0.6154	0.9231	18	1	25	0.6410	0.9615
27. Fodor's.	345	2	1		7	1	25	0.6410	0.9259	19	1	26	0.6667	0.9630
28. CheapTi	4	235	1		3	1	26	0.6667	0.9286	20	1	27	0.6923	0.9643
29. iExplore	35	24	1		5	1	27	0.6923	0.9310	23	1	28	0.7179	0.9655
30. Overseas	5	234	1		3	1	28	0.7179	0.9333	24	0	0	0.0000	0.0000
31. Yahoo!	35	24	1		5	1	29	0.7436	0.9355	26	1	29	0.7436	0.9355
32. TripAd	345	2	1		7	1	30	0.7692	0.9375	28	1	30	0.7692	0.9375
33. eTravel:	45	23	1		5	1	31	0.7949	0.9394	29	1	31	0.7949	0.9394
34. U.S. and	45	23	1		5	1	32	0.8205	0.9412	30	1	32	0.8205	0.9412
35. Travel |	1345	2			13	1	33	0.8462	0.9429	32	1	33	0.8462	0.9429
36. Amadeus	5	2	134		-5	0	0	0.0000	0.0000	34	1	34	0.8718	0.9444
37. Costco		2	14	35	-5	0	0	0.0000	0.0000	35	1	35	0.8974	0.9459
38. Arizona	35	24	1		5	1	34	0.8718	0.8947	36	0	0	0.0000	0.0000
39. Find	4	25	13		-1	0	0	0.0000	0.0000	37	0	0	0.0000	0.0000
40. About.c	4	235	1		3	1	35	0.8974	0.8750	38	1	36	0.9231	0.9000
41. Travel |	15	2	34		1	1	36	0.9231	0.8780	39	0	0	0.0000	0.0000
42. Travel.		24	13	5	-4	0	0	0.0000	0.0000	42	0	0	0.0000	0.0000
43. Travelocit	45	2	13		1	1	37	0.9487	0.8605	43	1	37	0.9487	0.8605
44. Great		2	1	345	-2	0	0	0.0000	0.0000	44	0	0	0.0000	0.0000
45. PlacesOn	345	12			11	1	38	0.9744	0.8444	46	0	0	0.0000	0.0000
46. All	5	24	13		-1	0	0	0.0000	0.0000	47	0	0	0.0000	0.0000
47. Travel -		2	14	35	-5	0	0	0.0000	0.0000	48	1	38	0.9744	0.8085
48. Expedi	4	235	1		3	1	39	1.0000	0.8125	49	0	0	0.0000	0.0000
49. Wilcox		245	13		-3	0	0	0.0000	0.0000	50	0	0	0.0000	0.0000
50. Travel	4	12	35		-1	0	0	0.0000	0.0000	50	0	0	0.0000	0.0000

W-S: Web Snippet, R(3): Relevance with score 3, P(1): Partial relevant, with score 1, I(-3): Irrelevant, with score -3, N(0): no sufficient information to support a decision, with score 0. SC: score, JG: judgment, RL: number of relevant document, Rc: recall of Yahoo, Pr: precision of Yahoo, NR: new ranked results of RIB, JG: judgment of the new results, RL’: number of relevant document in new ranked results, Rc’: recall of new results, Pr’: precision for new results.

R-Lv	10	20	30	40	50	60	70	80	90	100
Pr Yahoo	100	100	100	95.2	95.2	94.1	94.1	94.1	87.5	81.3
Pr Bu	100	100	100	100	100	96.6	96.6	94.1	90	80.6

JN: 141; AN: 0; JCD: 0; Impr: 1.64



9 References

Abdi, H., and L. J. Williams. 2010. Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics* 2 (4): 433-459.

Aggarwal, C. C., A. Hinneburg, and D. A. Keim. 2001. *The Eighth International Conference on Database Theory (ICDT 2001), January 4-6, 2001: On the Surprising Behavior of Distance Metrics in High Dimensional Space*. London, UK: Springer Berlin

AI-Mubaid, H., and S. A. Umair. 2006. A New Text Categorization Technique Using Distributional Clustering and Learning Logic. *IEEE Transactions on Knowledge and Data Engineering* 18 (9): 1156-1165.

Albertoni, R., E. Camossi, M. D. Martino, F. Giannini, and M. Monti. 2006. Semantic Granularity for the Semantic Web. In *On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops*, ed. R. Meersman, Z. Tari and P. Herrero, 1863-1872. Heidelberg: Springer Berlin.

Apté, C., F. Damerau, and S. M. Weiss. 1994. Automated Learning of Decision Rules for Text Categorization. *ACM Transactions on Information Systems* 12 (3): 233-251.

Avison, D., and M. Myers. 2005. Qualitative Research. In *Research in Information Systems: A Handbook for Research Supervisors and Their Students,*, ed. D. Avison and J. Pries-Heje, 239-253. Burlington: Elsevier Butterworth-Heinemann.

Baeza-Yates, R., and B. Ribeiro-Neto. 1999. *Modern Information Retrieval*. Harlow: Addison Wesley.

Barrett, R., P. P. Maglio, and D. C. Kellem. 1997. *Proceedings of the ACM CHI 97 Human Factors in Computing Systems Conference, How to Personalize the Web*. Atlanta, Georgia: ACM Press

Bekkerman, R., R. El-Yaniv, N. Tishby, and Y. Winter. 2003. Distributional Word Clusters vs. Words for Text Categorization. *Journal of Machine Learning Research* 3 (March): 1183-1208.

Berry, M. W., S. T. Dumais, and G. W. O'Brien. 1995. Using linear algebra for intelligent information retrieval. *Society for Industrial and Applied Mathematics (SIAM) Review* 37 (4): 573-595.

Beyer, K., J. Goldstein, R. Ramakrishnan, and U. Shaft. 1999. *Proceedings of the 7th International Conference on Database Theory (ICDT 1999), January 10-12, 1999: When Is "Nearest Neighbor" Meaningful?* Jerusalem, Israel: Springer-Verlag

Bilmes, J. A. 1998. *A Gentle Tutorial of the EM Algorithms and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models.* http://crow.ee.washington.edu/people/bulyko/papers/em.pdf (accessed September 1st, 2009).

Bloehdorn, S., and A. Hotho. 2004. Boosting for Text Classification with Semantic Features. In *LNCS 3932, Knowledge Discovery and Data Mining, The 10th ACM SIGKDD Conference on*, ed. L. Getoor, A. Hotho and Y. Sure, 70-87. Berlin, Heidelberg: Springer Verlag.

Boley, D., M. Gini, R. Gross, E.-H. Han, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, and J. Moore. 1999. Partitioning-based clustering for Web document categorization. *Decision Support Systems* 27 (3): 329-341.

Borlund, P. 2003. The Concept of Relevance in IR. *Journal of the American Society for Information Science and Technology* 54 (10): 913-925.

Buckley, C., and E. M. Voorhees. 2000. *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, July 24-28, 2000: Evaluating Evaluation Measure Stability*. Athens Grace: ACM Press

Burges, C. J. C. 1998. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery* 2 (2): 121-167.

Cai, L., and T. Hofman. 2003. *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003), July 28 - August 1, 2003: Text Categorization by Boosting Automatically Extracted Concepts*. Toronto, Canada: ACM Press

Campos, L. M., and A. E. Romero. 2009. Bayesian network models for hierarchical text classification from a thesaurus. *International Journal of Approximate Reasoning* 50 (7): 932-944.

Candan, Ç. a., M. A. Kutay, and H. Ozaktas. 2000. The Discrete Fractional Fourier Transform. *IEEE Transactions on Singal Processing* 48 (5): 1329-1337.

Cardoso-Cachopo, A., and A. L. Oliveira. 2003. An Empirical Comparison of Text Categorization Methods. In *Proceedings of the 10th International Symposium on String Processing and Information Retrieval*, ed. M. A. Nascimento, E. S. d. Moura and A. L. Oliveira, 183-196. Berlin: Springer.

Caropreso, M. F., S. Matwin, and F. Febastiani. 2001. A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. In *Text databases and document management: theory and practice*, ed. A. G. Chin, 78-102. Hershey: IGI Publishing.

Carpineto, C., and O. Stanisław. 2009. A Survey of Web Clustering Engines. *ACM Computing Surveys* 41 (2): Article 17.

Carterette, B., V. Pavlu, E. Kanoulas, J. A. Aslam, and J. Allan. 2008. *Proceedings of the 31st Annual International SIGIR Conference on Research and Development in Information Retrieval (SIGIR 08), July 20-24, 2008, : Evaluation Over Thousands of Queries*. Singapore:

Castellano, G., A. M. Fanelli, M. A. Torsello, and L. C. Jain. 2009. Innovations in Web Personalization. In *Web Personalization in Intelligent Environments*, ed. G. Castellano, L. C. Jain and A. M. Ganelli, 1-26. Berlin: Springer-Verlag.

Ch, V. S. R., and B. D. Chaudhary. 2006. *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence, December 18-22, 2006: A Hierarchy of Search Engines based on ODP Concepts*. HongKong, China: IEEE Computer Society

Chakrabarti, S. 2003. *Mining the Web: Discovering Knowledge from Hypertext Data*. Edited by J. Gray, *Data Management Systems*. San Francisco: Morgan Kaufmann.

Chan, P. K. 1999. *Proceedings of Workshop on Web Usage Analysis and User Profiling WEBKDD, August 15, 1999: A non-invasive learning approach to building web user profiles*. San Diego CA.: ACM Press

Chapelle, O., B. Scholkopf, and A. Zien. 2006. *Semi-Supervised Learning*. Edited by T. Dietterich, *Adaptive Computation and Machine Learning*. Cambridge, Massachusetts: The MIT Press.

Chen, J., H. Huang, S. Tian, and Y. Qu. 2009. Feature selection for text classification with Naïve Bayes *Expert Systems with Applications* 36 (3): 5432-5435.

Chen, L., and K. Sycara. 1998. *Proceedings of the Second International Conference on Autonomous Agents, May 9-13, 1998: WebMate: A Personal Agent for Browsing and Searching*. Minneapolis, St. Paul: ACM Press

Chiang, M. M.-T., and B. Mirkin. 2006. *Proceedings of the UK Workshop on Computational Intelligence 2006, Determining the number of clusters in the Straight K-means: Experimental comparison of eight options*. Leeds, UK:

Chirita, P.-A., W. Nejdl, R. Paiu, and C. Kohlschütter. 2005. *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 15-19, 2005: Using ODP Metadata to Personalize Search*. Salvador, Brazil: ACM Press. http://citeseer.ist.psu.edu/cutting92scattergather.html (accessed August 15-19, 2005).

Chowdhury, A., and I. Soboroff. 2002. *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 11-15, 2002: Automatic Evaluation of World Wide Web Search Services*. Tampere, Finland: ACM Press

Church, K. W., and P. Hanks. 1990. Word Association Norms, Mutual Information, and Lexicography. *Computational Linguistics* 16 (1): 22-29.

Cilibrasi, R. L., and P. M. Vitányi. 2007. The Google Similarity Distance. *IEEE Transactions on Knowledge and Data Engineering* 19 (3): 370-383.

Cios, K. J., W. Pedrycz, R. W. Swiniarski, and L. A. Kurgan. 2007. *Data Mining A Knowledge Discovery Approach*. New York: Springer.

Cohen, W. W., and Y. Singer. 1999. Context-Sensitive Learning Methods for Text Categorization. *ACM Transactions on Information Systems* 17 (2): 141-173.

Cohn, M., and R. Herring. 2005. *Yahoo Claims Edge Over Google*. http://www.basex.com (accessed July 7, 2006).

Comité, F. D., R. Gilleron, and M. Tommasi. 2003. Learning Multi-label Alternating Decision Trees from Texts and Data. In *Machine Learning and Data Mining in Pattern Recognition*, ed. P. Perner and A. Rosenfeld, 251-274. Berlin/Heidelberg: Springer.

Cornuéjols, A. 2009. On-line learning: Where are we so far? In *Blueprint on Ubiquitous Knowledge Discovery*, ed. M. May and L. SaittaBerlin: Springer.

Cristianini, N., and J. Shawe-Taylor. 2000. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge: Cambridge University Press.

Cutting, D. R., D. R. Karger, J. O. Pedersen, and J. W. Tukey. 1992. *Proceedings of the 15th annual international ACM/SIGIR conference on Research and development in information retrieval, June 21-24, 1992: Scatter/gather: A Cluster-based Approach to Browsing Large Document Collections*. Copenhagen, Denmark: ACM Press. http://citeseer.ist.psu.edu/cutting92scattergather.html (accessed May 10, 2007).

Dagan, I., Y. Karov, and D. Roth. 1997. *Proceedings of EMNLP-97, 2nd Conference on Empirical Methods in Natural Language Processing, August 1-2, 1997: Mistake-Driven Learning in Text Categorization*. Providence, RI.:

Davidov, D., E. Gabrilovich, and S. Markovitch. 2004. *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 25-29, 2004: Parameterized Generation of Labeled Datasets for Text Categorization Based on a Hierarchical Directory*. Sheffield, UK: ACM Press

Debole, F., and F. Sebastiani. 2005. An Analysis of the Relative Hardness of Reuters-21578 Subsets. *Journal of the American Society for Information Science and Technology* 56 (6): 584-596.

Deerwester, S., S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science* 41 (6): 391-407.

Dempster, A. P., N. M. Laird, and D. B. Rubin. 1977. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)* 39 (1): 1-38.

Dhillion, I. S., S. Mallela, and R. Kumar. 2003. A Divisive Information-Theoretic Feature Clustering Algorithm for Text Classification. *Journal of Machine Learning Research* 3 (March): 1265-1287.

Dietterich, T. G. 2000. An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization. *Machine Learning* 40 (2): 139-157.

Dou, Z., R. Song, and J.-R. Wen. 2007. *Proceedings of the 16th international conference on World Wide Web, WWW 2007, May 8-12, 2007: A Large scale Evaluation and Analysis of Personalized Search Strategies*. Banff, Alberta, Canada: ACM Press

Dumais, S., E. Cutrell, J. Cadiz, G. Jancke, R. Sarin, and D. C. Robbins. 2003. *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003), July 28 - August 1, 2003: Stuff I've Seen: A System for Personal Information Retrieval and Re-Use*. Toronto, Canada: ACM Press

Dumais, S., J. Platt, D. Heckerman, and M. Sahami. 1998. *Proceedings of the 1998 ACM CIKM International Conference on Information and Knowledge Management*

(CIKM'98), November 3-7, 1998: Inductive Learning Algorithms and Representatoins for Text Categorization. Bethesda, Maryland, USA: ACM Press

Dumais, S. T. 2004. Latent Semantic Analysis. *Annual Review of Information Science and Technology (ARIST)* 38: 189-230.

Dumais, S. T., and H. Chen. 2000. *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, July 24-28, 2000: Hierarchical Classification of Web Content*. Athens Grace: ACM Press

Dupret, G. 2003. *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003), July 28 - August 1, 2003: Latent Concepts and teh Number Orthogonal Factors in Latent Semantic Analysis*. Toronto, Canada: ACM Press

Escudero, G., L. Màrquez, and G. Rigau. 2000. Boosting Applied to Word Sense Disambiguation. In *Machine Learning: ECML 2000: 11th European Conference on Machine Learning*, ed. R. L. Màntaras and E. Plaza, 665-677. Berlin/Heidelberg: Springer.

Eyheramendy, S., D. D. Lewis, and D. Madigan. 2003. *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics, January 3-6, 2003: On the Naive Bayes Model for Text Categorization*. Key West, Florida USA:

Fan, R.-E., and C.-J. Lin. 2007. *A Study on Threshold Selection for Multi-label Classification*. Taipei: National Taiwan University.

Farach, M. 1997. *Proceedings of the 38th Annual Symposium on Foundations of Computer Science, October 20-22, 1997: Optimal Suffix Tree Construction with Large Alphabets*. Miami Beach, FL: IEEE Computer Society

Feldman, R., and J. Sanger. 2006. *The Text Mining Handbook - Advanced Approaches in Anlyzing Unstructured Data*. Cambridge: Cambridge University Press.

Ferragina, P., and A. Gulli. 2005. *Proceedings of the Special interest tracks and posters of the 14th international conference on World Wide Web, May 10-14, 2005: A Personalized Search Engine Based on Web-Snippet Hierarchical Clustering*. Chiba, Japan: ACM Press

Fonseca, F., M. Egenhofer, C. Davis, and G. Câmara. 2002. Semantic Granularity in Ontology-Driven Geographic Information Systems. *Annals of Mathematics and Artificial Intelligence - Special Issue on Spatial and Temporal Granularity* 36 (1-2): 121-151.

Forman, G. 2003. An Extensive Empirical Study of Feature Selection Metrics for Text Classification. *Journal of Machine Learning Research* 3: 1289-1305. http://jmlr.csail.mit.edu/papers/volume3/forman03a/forman03a_full.pdf (accessed 2008/2/28).

Freund, Y., and R. E. Schapire. 1999. A Short Introduction to Boosting. *Journal of Japaness Society for Artificial Intelligence* 14 (5): 771-780.

--------. August 1997. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boostin. *Journal of Computer and System Science* 55 (1): 119-139.

Furnas, G. W., S. Deerwester, S. T. Dumais, T. K. Landauer, R. A. Harshman, L. A. Streeter, and K. E. Lochbaum. 1988. *Proceedings of the 11th annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR-98), June 13-15, 1988: Information retrieval using a singular value decomposition model of latent semantic structure*. Grenoble, France: ACM Press

Gabrilovich, E. 2006. Feature Generation for Textual Information Retrieval Using World Knowledge. PhD diss., Department of Computer Science, Israel Institute of Technology, Haifa. http://www.cs.technion.ac.il/~gabr/pubs.html (accessed March 8, 2008).

Galavotti, L., F. Sebastiani, and M. Simi. 2000. Experiments on the Use of Feature Selection and Negative Evidence in Automated Text Categorization. In *Proceedings of the 4th European Conference on Research and Advanced Technology for Digital Libraries*, ed. J. L. Borbinha and T. Baker, 59-68. London, UK: Springer-Verlag.

Gauch, S., J. Chaffee, and A. Pretschner. 2003. Ontology-based personalized search and browsing. *Web intelligence and Agent System* 1 (3-4): 219-234.

Gauch, S., M. Speretta, A. Chandramouli, and A. Micarelli. 2007. Personalized Search on the World Wide Web. In *The Adaptive Web*, ed. P. Brusilovsky, A. Kobsa and W. Nejdl, 54-89. Berlin, Heidelberg: Springer-Verlag.

Geman, S., E. Bienenstock, and R. Doursat. 1992. Neural Networks and the Bias/Variance Dilemma. *Neural Computation* 4 (1): 1-58.

Gentili, G., A. Micarelli, and F. Sciarrone. 2003. INFOWEB: An Adaptive Information Filtering System for the Cultural Heritage Domain. *Applied Artificial Intelligence* 17 (8-9): 715-744.

Gentle, J. E. 2007. *Matrix Algebra - Theory, Computations, and Applications in Statistics*. New York: Springer.

Geraci, F., M. Pellegrini, P. Pisati, and F. Sebastiani. 2006. *Proceedings of the 21st Annual ACM Symposium on Applied Computing, April 23-27, 2006: A Scalable Algorithm for High-Quality Clustering of Web Snippets*. Dijon, France: ACM Press

Giles, J. 2005. Internet encyclopaedias go head to head. *Nature* 438 (7070): 900-901.

Glover, E. J., S. Lawrence, M. D. Gordon, W. P. Birmingham, and L. Giles. 2001. Improving Web searching with user preferences. Web Search – Your Way. *Communications of the ACM* 44 (12): 97-102.

Godoy, D., and A. Amandi. 2006. Modeling user interests by conceptual clustering. *Information Systems* 31: 247-265.

Golub, G. H., and C. F. v. Loan. 1996. *Matrix Computations*. 3rd ed. Baltimore: The Johns Hopkins University Press.

Gospodnetić, O., and E. Hatcher. 2005. *Lucene In Action*. Greenwich: Manning Publications.

Gruber, T. R. 1993. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. *International Journal of Human-Computer Studies* 43 (5-6): 907-928.

Guyon, I., S. Gunn, M. Nikravesh, and L. A. Zadeh. 2006. *Feature Extraction: Foundations and Applications*. Edited by J. Kacprzyk, *Studies in Fuzziness and Soft Computing*. Berlin: Springer.

Hamerly, G., and C. Elkan. 2003. *Proceedings of the seventeenth annual conference on neural information processing systems (NIPS), December 8-13, 2003: Learning the k in k-means*. Vancouver, Canada: MIT Press

Han, E.-H., and G. Karypis. 2000. Centroid-Based Document Classification: Analysis and Experimental Results. In *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery* ed. D. A. Zighed, J. Komorowski and J. Zytkow, 424-431. London: Springer.

Han, J., and M. Kamber. 2006. *Data Mining: Concepts and Techniques*. Edited by J. Gray. 2nd ed, *DataManagement Systems*. San Francisco: Morgan Kaufmann.

Hand, D. J. 2006. Classifier Technology and the Illusion of Progress. *Statistical Science* 21 (1): 1-15.

Hand, D. J., and K. Yu. 2001. Idiot's Bayes - Not So Stupid After All? *International Statistical Review* 69 (3): 385-398.

Harter, S. P. 1992. Psychological Relevance and Information Science. *Journal of the American Society for Information Science* 43 (9): 602-615.

Hasan, M. M., and Y. Matsumoto. 1999. Document Clustering: Before and After the Singular Value Decomposition. *Joho Shori Gakkai Kenkyu Hokoku* 99 (95): 47-54.

Haveliwala, T. H. 2002. *Proceedings of the 11th international conference on World Wide Web (WWW'02), May 7-11, 2002: Topic-sensitive PageRank*. Honolulu, Hawaii: ACM Press

Hearst, M. A., S. T. Dumais, E. Osuna, J. Platt, and B. Schölkopf. 1998. Support vector machines. *IEEE Intelligent Systems* 13 (4): 18-28.

Hearst, M. A., and J. O. Pedersen. 1996. *Proceedings of the 19th annual international ACM/SIGIR conference on Research and development in information retrieval, August 18-22, 1996: Reexamining the Cluster Hypothesis: Scatter/Gather on Retrieval Results*. Zurich, Switzerland: ACM Press

Hersh, W., C. Buckley, T. Leone, and D. Hickam. 1994. *Proceedings of SIGIR-94, the 19th Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval, July 3-6, 1994: OHSUMED: An interactive retrieval evaluation and new large test collection for research*. Dublin, Ireland: ACM/Springer

Hjørland, B. 2010. The Foundation of the Concept of Relevance. *Journal of the American Society for Information Science and Technology* 61 (2): 217-223.

Hull, D. A. 1994. *Proceedings of SIGIR-94, the 19th Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval, July 3-6, 1994: Improving Text Retrieval for the Routing Problem Using Latent Semantic Indexing*. Dublin, Ireland: ACM/Springer

Húsek, D., J. Pokorný, H. Řezanková, and V. Snášel. 2007. Data Clustering: From Documents to the Web. In *Web Data Management Practices: Emerging Techniques and Technologies*, ed. A. Vakali and G. Pallis, 1-33. Hershey: Idea Group Publishing.

Hyvärinen, A., and E. Oja. 2000. Independent component analysis: algorithms and applications. *Neural Networks* 13 (4-5): 411-430.

Ifrim, G., G. Bakir, and G. Weikum. 2008. *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, Fast Logistic Regression for Text Categorization with Variable-Length N-grams*. Las Vegas, Nevada, USA: ACM press

Introduction to MeSH. 2007. http://www.nlm.nih.gov/mesh/introduction2008.html (accessed August 12, 2008).

Iyer, R. D., D. D. Lewis, R. E. Schapire, Y. Singer, and A. Singhal. 2000. *Proceedings of the 2000 ACM CIKM International Conference on Information and Knowledge Management, November 6-11, 2000: Boosting for Document Routing*. McLean, VA. USA.: ACM Press

Jacsó, P. 2007a. Clustering search results. Part 1: Web-wide search engines. *Online Information Review* 31 (1): 85-91.

--------. 2007b. Clustering search results. Part 2: search engine for highly structured databases. *Online Information Review* 31 (2): 234-241.

Jain, A., M. Murty, and P. Flynn. 1999. Data Clustering: A Review. *ACM Computing Surveys* 31 (3): 264-323.

Jain, A. K. 2009. Data Clustering: 50 Years Beyond K-Means. *Pattern Recognition Letters*:

Jain, A. K., and R. C. Dubes. 1988. *Algorithms for clustering data*. Englewood Cliffs: Prentice-Hall Inc.

Jansen, B. J., and A. Spink. 2006. How are we searching the World Wide Web? A Comparison of Nine Search Engine Transaction Logs. *Information Processing and Management* 42 (1): 248-263.

Jansen, B. J., A. Spink, and J. Pedersen. 2005. A Temporal Comparison of AltaVista Web Searching. *Journal of the American Society for Information Science and Technology* 56 (6): 559-570.

Jeh, G., and J. Widon. 2003. *Proceedings of the 12th International World Wide Web Conference, May 20-24, 2003: Scaling Personalized Web Search*. Budapest, Hungary: ACM Press

Joachims, T. 1997. *Proceedings of ICML-97, 14th International Conference on Machine Learning, July 8 - 12, 1997: A Probability Analysis of the Rocchio Algorithm with TFIDF for Text Categorization*. Nashville, TN.: Morgan Kaufmann

--------. 1998. *Proceedings of the 10th European Conference on Machine Learning (ECML-98), April 21-23, 1998: Text Categorization with Support Vector Machines: Learning with Many Relevant Features*. Chemnitz, Germany: Springer-Verlag

--------. 1999. Making large-scale support vector machine learning practical. In *Advances in Kernel Methods - Support Vector Learning*, ed. B. Schölkopf, C. Burges and A. Smola, 169-184. Cambridge, MA: MIT Press.

--------. 2008. *SVM^{light} Support Vector Machine*. http://svmlight.joachims.org/ (accessed June 20, 2009).

Jones, K. S. 2003. Document Retrieval: Shall Data, Deep Theories; Historical Reflections, Potential Directions. In *Advances in Information Retrieval, 25th European Conference on IR Research, ECIR 2003*, ed. F. Sebastiani, 1-11. Berlin Heidelberg: Springer-Verlag.

Kanungo, T., D. M. Mount, N. S. Netanyahu, C. Piatko, R. Silverman, and A. Y. Wu. 2000. *Proceedings of the sixteenth annual symposium on Computational geometry, June 12-14, 2000: The Analysis of a Simple k-Means Clustering Algorithm*. Hong Kong: ACM Press

Kearns, M. J., and U. V. Vazirani. 1994. *An introduction to computational learning theory*. Cambridge, Mass.: MIT Press.

Kim, Y.-H., S.-Y. Hahn, and B.-T. Zhang. 2000. *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, July 24-28, 2000: Text Filtering by Boosting Naive Bayes Classifiers*. Athens Grace: ACM Press

Klas, C.-P., and N. Fuhr. 2000. *Proceedings of the 22nd Annual Colloquium of the British Computer Society Information Retrieval Specialist Group (BCSIGSG-00), A New Effective Approach for Categorizing Web Documents*. Cambridge, UK: Unknown

Kobsa, A., and J. Schreck. 2003. Privacy Through Pseudonymity in User-Adaptive Systems. *ACM Transactions on Internet Technology* 3 (2): 149-183.

Kołcz, A., and W.-t. Yih. 2007. *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, August 12-15, 2007: Raising the Baseline for High-Precision Text Classifiers*. San Jose, California, USA: ACM Press

Kontostathis, A., and W. M. Pottenger. 2006. A Framework for understanding Latent Semantic Indexing (LSI) performance. *Information Processing and Management* 42: 56-73.

Kotsiantis, S. 2007. Supervised Machine Learning: A Review of Classification Techniques. *Informatica* 31 (3): 249-268.

Koychev, I. 2001. *Proceedings of the UM 2001 Workshop on Machine Learning for User Modeling, 2001: Learning about User in the Presence of Hidden Context*. Sonthofen, Germany:

Kung, S. Y., M. W. Mak, and S. H. Lin. 2004. *Biometric Authentication: A Machine Learning Approach*. Upper Saddle Rive: Prentice Hall PTR.

Kurland, O., and L. Lee. 2009. Clusters, Language Models, and ad hoc Information Retrieval. *ACM Transactions on Information Systems* 27 (3): 13:1-39.

Kyriakopoulou, A. 2008. Text Classification Aided by Clustering: a Literature Review. In *Tools in Artificial Intelligence*, ed. P. Fritzsche, 233-252. Vienna: InTech Education and Publishing.

Lam, S. L. Y., and D. L. Lee. 1999. *Proceedings of DASFAA-99, 6th IEEE International Conference on Database Advanced Systems for Advanced Applications, April 19-22, 1999: Feature Reduction for Neural Network Based Text Categorization*. Hsinchu, Taiwan: IEEE Computer Society

Lam, W., M. Ruiz, and P. Srinivasan. 1999. Automatic Text Categorization and Its Application to Text Retrieval. *IEEE Transactions on Knowledge and Data Engineering* 11 (6): 865-978.

Landauer, T. K., P. W. Foltz, and D. Laham. 1998. An Introduction to Latent Semantic Analysis. *Discourse Processes* 25: 259-284.

Lang, K. 1995. *Proceedings of the 12th International Machine Learning Conference (ML95), July 9-12, 1995: NewsWeeder: Learning to Filter Netnews*. Tahoe City, California: Morgan Kaufmann

Larose, D. T. 2005. *Discovering knowledge in data : an introduction to data mining*. Hoboken, New Jersey: John Wiley & Sons.

Leighton, H. V., and J. Srivastava. 1999. First 20 Precision among World Wide Web Search Services (Search Engines). *Journal of the American Society for Information Science* 50 (1): 870-881.

Lertnattee, V., and T. Theeramunkong. 2004. Effect of term distributions on centroid-based text categorization. *Information Sciences* 158 (1): 89-115.

Lewis, D. 1998. Naive (Bayes) at Forty: The Independence Assumption in Information Retrieval. In *Proceedings of the 10th European Conference on Machine Learning* ed. C. Nedellec and C. Rouveirol, 4-15. London: Springer-Verlag.

Lewis, D. D. 1992. *Proceedings of the 15th annual international ACM/SIGIR conference on Research and development in information retrieval, June 21-24, 1992: An Evaluation of Phrasal and Clustered Representation on a Text Categorization Task*. Copenhagen, Denmark: ACM PressMay 10, 2007).

Lewis, D. D., and W. A. Gale. 1994. *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval, A Sequential Algorithm for Training Text Classifiers*. Dublin, Ireland: Springer-Verlag London

Lewis, D. D., Y. Yang, T. G. Rose, and F. Li. 2004. RCV1: A New Benchmark Collection for Text Categorization Research. *Journal of Machine Learning Research* 5: 361-397.

Li, B., Q. Lu, and S. Yu. 2004. An Adaptive k-Nearest Neighbor Text Categorization Strategy. *ACM Transactions on Asian Language Information Processing* 3 (4): 215-226.

Li, Y. H., and A. K. Jain. 1998. Classification of Text Documents. *The Computer Journal* 41 (8): 537-546.

Limbu, D. K., A. M. Connor, and S. G. MacDonell. 2005. *Proceedings of the 14th International Conference on Adaptive Systems and Software Engineering (IASSE05), July 20-22, 2005: A Framework for Contextual Information Retrieval from the WWW*. Toronto, Canada: ISCA

Lin, T. Y., and I.-J. Chiang. 2006. *Proceedings of the 10th Conference on Artificial Intelligence and Applications, December 2-3, 2005: Granulate and Conquer: Clustering Web Pages Semantically using Combinatorial Topology*. Kaohsiung, Taiwan:

Liu, B. 2007. *Web Data Mining - Exploring Hyperlinks, Contents, and Usage Data*. Berlin: Springer-Verlag.

Liu, F., C. Yu, and W. Meng. 2002. *Proceedings of the Eleventh International Conference on Information and Knowledge Management, November 4-9, 2002: Personalized Web Search by Mapping User Queries to Categories*. McLean, Virginia: ACM Press

--------. 2004. Personalized Web Search for Improving Retrieval Effectiveness. *IEEE Transactions on Knowledge and Data Engineering* 16 (1): 28-40.

Liu, J., and L. Birnbaum. 2007. *Proceedings of the 2007 IEEE/WIC/ACM International Conference on Web Intelligence Measuring Semantic Similarity between Named Entities by Searching the Web Directory*. Silicon Vally, CA, USA: IEEE Computer Society

Liu, X., and W. B. Croft. 2003. Statistical Language Modeling for Information Retrieval. *Annual Review of Information Science and Technology* 39: 3-31.

Lu, X., B. Zheng, A. Velivelli, and C. Zhai. 2006. Enhancing Text Categorization with Semantic-enriched Representation and Training Data Augmentation. *Journal of the American Medical Informatics Association* 13 (5): 526-535.

Luenberger, D. G., and Y. Ye. 2008. *Linear and Nonlinear Programming*. 3rd ed. New York: Springer.

Ma, Z., G. Pant, and O. R. L. Sheng. 2007. Interest-Based Personalized Search. *ACM Transactions on Information Systems* 25 (1, Article 5.): 1-38.

MacQueen, J. 1967. *Proceedings of 5th Berkley Symposium on Mathematical Statistics and Probability, June 21 - July 18, 1967: Some methods for classification and analysis of multivariate observations*. Berkeley, California: University of California Press

Maguitman, A. G., F. Menczer, F. Erdinc, H. Roinestad, and A. Vespignani. 2006. Algorithmic Computation and Approximation of Semantic Similarity. *World Wide Web* 9 (4): 431-456.

Manning, C. D., P. Raghavan, and H. Schütze. 2008. *Introduction to Information Retrieval*. Cambridge: Cambridge University Press.

Meng, W., C. Yu, and K.-L. Liu. 2002. Building Efficient and Effective Metasearch Engines. *ACM Computing Surveys* 34 (1): 48-89.

Micarelli, A., F. Gasparetti, F. Sciarrone, and S. Gauch. 2007. Personalized Search on the World Wide Web. In *The Adaptive Web*, ed. P. Brusilovsky, A. Kobsa and W. Nejdl, 195-230. Berlin, Heidelberg: Springer-Verlag.

Micarelli, A., and F. Sciarrone. 2004. Anatomy and Empirical Evaluation of an Adaptive Web-Based Information Filtering System. *User Modeling and User-Adapted Interaction* 14 (2-3):

Middleton, S. E., D. C. De Roure, and N. R. Shadbolt. 2001. *Proceedings of the 1st international conference on Knowledge capture, October 21-23, 2001: Capturing knowledge of user preferences: ontologies in recommender systems*. Victoria, British Columbia, Canada: ACM Press

Mihalkova, L., and R. Mooney. 2009. Learning to Disambiguate Search Queries from Short Sessions. In *Machine Learning and Knowledge Discovery in Databases*, ed. W. Buntine, M. Grobelnik and D. Mladenic, 111-117. Heidelberg: Springer Berlin.

Miller, D. R. H., T. Leek, and R. M. Schwartz. 1999. *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 15-19, 1999: A Hidden Markov Model Information Retrieval System*. Berkley, UAS: ACM Press

Milligan, G. W., and M. C. Cooper. 1986. A Study of the Comparability of External Criteria for Hierarchical Cluster Analysis. *Multivariate Behavioral Research* 21 (4): 441-458.

Mirkin, B. 1999. Concept Learning and Feature Selection Based on Square-Error Clustering. *Machine Learning* 35 (1): 25-39.

--------. 2005. *Clustering for data mining : a data recovery approach*. Edited by J. Lafferty, D. Madigan, F. Murtagh and P. Smyth, *Computer Science and Data Analysis Series*. Boca Raton: Chapman & Hall/CRC.

Mitchell, T. M. 1997. *Machine Learning*. New York: McGraw-Hill Companies.

Mizzaro, S. 1997. Relevance: The Whole History. *Journal of the American Society for Information Science* 48 (9): 810-832.

--------. 1998. How many relevances in information retrieval. *Interacting with Computers* 10 (3): 303-320.

Mladenic, D., and M. Grobelnik. 2003. Feature selection on hierarchy of web documents. *Decision Support Systems* 35: 45-87.

Mobasher, B. 2007. Data Mining for Web Personalization. In *The Adaptive Web*, ed. P. Brusilovsky, A. Kobsa and W. Nejdl, 90-135. Berlin, Heidelberg: Springer-Verlag.

Montebello, M. 1998. *Proceedings of String Processing and Information Retrieval: A South American Symposium, September 9-11, 1998: Information Overload--An IR Problem?* Santa Cruz de la Sieerra, Bolivia: IEEE Computer Society

Montgomery, D. C., and G. C. Runger. 2003. *Applied Statistics and Probability for Engineers*. 3rd ed. New York: John Wiley & Sons.

Monti, S., P. Tamayo, J. Mesirov, and T. Golub. 2003. Consensus Clustering: A Resampling-Based Method for Class Discovery and Visualization of Gene Expression Microarray Data. *Machine Learning* 52 (1-2): 91-118.

Moon, T. K. 1996. The Expectation-Maximization Algorithm. *IEEE Signal Processing Magazine* 13 (6): 47-60.

Nelles, O. 2001. *Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy Models*. Berlin: Springer.

Ng, H. T., W. B. Goh, and K. L. Low. 1997. *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 27-31, 1997: Feature selection, perceptron learning, and a usability case study for text categorization*. Philadelphia, PA: ACM Press

Notes, G. 1998. Northern Light: New Search Engine for the Web and Full-Text Articles. *Database Magazine* 21 (1): 32-37.

Osiński, S., J. Stefanowski, and D. Weiss. 2004. *Proceedings of the International IIS: IIPWM'04 Conference, May 17-20, 2004: Lingo: Search Results Clustering Algorithm Based on Singular Value Decomposition*. Zakopane, Poland: Springer

Osiński, S., and D. Weiss. 2005. A Concept-Driven Algorithm for Clustering Search Results. *IEEE Intelligent Systems* May/June: 48-94.

Owen, C. L. 1998. Design Research: Building the knowledge base. *Design Studies* 19 (1): 9-20.

Panzzani, M., J. Muramatsu, and D. Billsus. 1996. *Proceedings of the Thirteenth National Conference On Artificial Intelligence (AAAI '96), August 4-8, 1996: Syskill & Webert: Idenfitying interesting web sites*. Portlan, Oregon: AAAI Press

Perugini, S. 2008. Symbolic links in the Open Directory Project. *Information Processing and Management* 44 (2): 910-930.

Pierralos, D., G. Paliouras, C. Papatheodorou, and C. D. Spyropoulos. 2003. Web Usage Mining as a Tool for Personalization: a Survey. *User Modeling and User – Adapted Interaction* 13 (4): 311-372.

Pitkow, J., H. Schütze, T. Cass, R. Cooley, D. Turnbull, A. Edmonds, E. Adar, and T. Breuel. 2002. Personalized Search: A contextual computing approach may prove a breakthrough in personalized search efficiency. *Communications of the ACM* 45 (9): 50-55.

Ponte, J., and W. B. Croft. 1998. *Proceedings of SIGIR-98, the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 24-28, 1998: A Language Modeling Approach to Information Retrieval*. Melbourne, Australia: ACM Press

Porter, M. 1980. An algorithm for suffix stripping. *Program* 14 (3): 130-137.

Prescher, D. 2003. *Proceedings of the 15th European Summer School in Logic, Language and Information (ESSLLI-03), August 18-29, 2003: A Tutorial on the Expectation-Maximization Algorithm Including Maximum-Likelihood Estimation and EM Training of Probabilistic Context-Free Grammars*. Vienna, Austria:

Pressman, R. S. 2001. *Software Engineering: A Practitioner's Approach*. fifth ed. New York: McGraw Hill High Education.

Qi, X., and B. D. Davison. 2009. Web Page Classification: Features and Algorithms. *ACM Computing Surveys* 41 (2): Article 12.

Qiu, F., and J. Cho. 2006. *Proceedings of the 15th international conference on World Wide Web, WWW 2006, May 23-26, 2006: Automatic Identification of Uer Interest for Personalized Search*. Edinburgh, Scotland: ACM Press

Qiu, G., K. Liu, J. Bu, C. Chen, and Z. Kang. 2007. *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 23-27, 2007: Quantify Query Ambiguity Using ODP Metadata*. Amsterdam, The Netherlands: ACM Press

Quinlan, J. R. 1993. *C4.5 Programs for Machine Learning*. San Mateo: Morgan Kaufmann.

Randall, R. A. 1976. How Tall Is a Taxonomic Tree? Some Evidence for Dwarfism. *American Ethnologist* 3 (3): 543-553.

Rennie, J. D. M., L. Shih, J. Teevan, and D. R. Karger. 2003. *Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003: Tackling the Poor Assumptions of Naive Bayes Text Classifiers*. Washington DC: AAAI Press

Rijsbergen, C. J. v. 1979. *Information Retrieval*. 2nd ed. London: Butterworths.

Rijsbergen, C. J. v., D. J. Harper, and M. F. Porter. 1981. The selection of Good Search Terms. *Information Processing & Management* 17 (2): 77-91.

Rokach, L., and O. Maimon. 2005. Clustering Methods. In *Data Mining and Knowledge Discovery Handbook*, ed. O. Maimon and L. Rokach: Springer.

Rosenfeld, R. 2000. Two decades of statistical language modeling: where do we go from here? *Proceedings of the IEEE* 88 (8): 1270-1278.

Ruiz, M. E., and P. Srinivasan. 2002. Hierarchical Text Categorization Using Neural Networks. *Information Retrieval* 5 (1): 87-118.

Rumpler, B. 2001. A study of the impact of the user profile in documentary systems. *Online Information Review* 25 (6): 359-364.

Sahami, M., V. Mittal, S. Baluja, and H. Rowley. 2004. The Happy Searcher: Challenges in Web Information Retrieval. In *Lecture Notes in Computer Science, Trends in Artifical Intelligence 2004*, ed. C. Zhang, H. W. Guesgen and W. K. Yeap, 3-12. Berlin Heidelberg: Springer-Berlag.

Salton, G., and C. Buckley. 1988. Term-Weighting Approaches in Automatic Text Retrieval. *Information Processing & Management* 24 (5): 513-523.

Salton, G., A. Wong, and C. S. Yang. 1975. A Vector Space Model for Automatic Indexing. *Communication of the ACM* 18 (11): 613-620.

Santamaria, C., J. Gonzalo, and F. Verdejo. 2003. Automatic Association of Web Directory with Word Senses. *Computational Linguistics* 29 (3): 485-502.

Santos, J. M., and M. Embrechts. 2009. On the Use of the Adjusted Rand Index as a Metric for Evaluating Supervised Classification. In *Proceedings of the 19th International Conference on Artificial Neural Networks: Part II* ed. C. Alippi, M. Polycarpou, C. Panayiotou and G. Ellinas, 175-186. Berlin, Heidelberg: Springer-Verlag.

Saracevic, T. 1975. Relevance: A Review of and a Framework for the Thinking on the Notion of Information Science. *Journal of the American Society for Information Science* 26 (6): 321-343.

--------. 2007a. Relevance: A Review of the Literature and a Framework for the Thinking on the Notion of Information Science. Part II: Nature and Manifestations of Relevance. *Journal of the American Society for Information Science and Technology* 58 (13): 1915-1933.

--------. 2007b. Relevance: A Review of the Literature and a Framework for the Thinking on the Notion of Information Science. Part III: Behavior and Effects of Relevance. *Journal of the American Society for Information Science and Technology* 58 (13): 2126-2144.

--------. 2008. Effects of Inconsistent Relevance Judgments on Information Retrieval Test Results: A Historical Perspective. *Library Trends* 56 (4): 763-783.

Schapire, R. E. 2003. The Boosting Approach to Machine Learning An Overview. In *Nonlinear Estimation and Classification*, ed. D. D. Denison, M. H. Hansen, C. C. Holmes, B. Mallick and B. Yu, 149-171. New York: Springer Verlag.

Schapire, R. E., and Y. Singer. 1999. Improved Boosting Algorithms Using Confidence-rated Predictions. *Machine Learning* 37 (3): 297-336.

--------. 2000. Boostexter: A Boosting-based System for Text Categorization. *Machine Learning* 39 (2-3): 135-168.

Schapire, R. E., Y. Singer, and A. Singhal. 1998. *Proceedings of SIGIR-98, the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 24-28, 1998: Boosting and Rocchio Applioced to Text Filtering.* Melbourne, Australia: ACM Press

Schölkopf, B., and A. J. Smola. 2002. *Learning with Kernels - Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, Massachusetts: The MIT Press.

Schütze, H., D. A. Hull, and J. O. Pedersen. 1995. *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, A Comparison of Classifiers and Document Representations for the Routing Problem*. Seattle, Washington: ACM Press

Sebastiani, F. 2002a. Machine Learning in Automated Text Categorization. *ACM Computing Surveys* 34 (1): 1-47.

--------. 2002b. Machine Learning in Automated Text Categorization. *ACM Computing Surveys* 34 (1): 1-47.

--------. 2005. Text Categorization. In *Text Mining and its Applications to Intelligence, CRM and Knowledge Management*, ed. A. Zanasi, 109-129. Southampton: MIT Press.

Sebastiani, F., A. Sperduti, and N. Valdambrini. 2000. *Proceedings of the 2000 ACM CIKM International Conference on Information and Knowledge Management (CIKM'00)*

November 6-11, 2000: An Improved Boosting Algorithm and its Application to Text Categorization. McLean, VA. USA.: ACM Press

Seher, I. 2007. A Personalised Query Expansion Approach Using Context, Computer Science, University of Western Sydney, Sydney

Shavlik, J., S. Calcari, T. Eliassi-Rad, and J. Solock. 1999. *Proceedings of the 1999 International Conference on Intelligent User Interfaces, January 5-8, 1999: An Instructable, Adaptive Interface for Discovering and Monitoring Information on the World-Wide Web*. Redondo Beach, Los Angeles.: ACM Press

Shaw JR, W. M. 1995. Term-Relevance Computations and Perfect Retrieval Performance. *Information Processing & Management* 31 (4): 491-498.

Shawe-Taylor, J., and N. Cristianini. 2004. *Kernel Methods for Pattern Analysis*. New York: Cambridge University Press.

Shen, D., R. Pan, J.-T. Sun, J. J. Pan, K. Wu, J. Yin, and Q. Yang. 2006. Query enrichment for web-query classification. *ACM Transactions on Information Systems* 24 (3): 320-352.

Shen, X., B. Tan, and C. Zhai. 2007. Privacy Protection in Personalization Search. *ACM SIGIR Forum* 41 (1): 4-17.

Sherman, C. 2000. Humans Do It Better: Inside the Open Directory Project. *Online*, July, 2000.

Sieg, A., B. Mobasher, and R. Burke. 2004. *Proceedings of the 2004 Meeting of the International Federation of Classification Societies, Inferring User's Information Context: Integrating User Profiles and Concept Hierarchies*. Chicago:

Slonim, N., and N. Tishby. 2001. *Proceedings of the 23rd European Colloquium on Information Retrieval Research (ECIR-01), April 4-6, 2001: The Power of Word Clusters for Text Classification*. Darmstadt, DE: British Computer Society

Smith, B. 2004. Ontology. In *Blackwell Guide to the Philosophy of Computing and Information*, ed. L. Floridi, 155-166. Oxford: Blackwell.

Smyth, B. 2007. A Community-Based Approach to Personalizing Web Search. *Computer* 40 (8): 42-50.

Song, F., and W. B. Croft. 1999. *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 15-19, 1999: A General Language Model for Information Retrieval*. Berkley, UAS: ACM Press

Speretta, M., and S. Gauch. 2005. *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence, September 19-22, 2005: Personalized Search based on User Search Histories*. Compiegne University of Technology, France: IEEE Computer Society

Stamou, S., and A. Ntoulas. 2009. Search personalization through query and page topical analysis. *User Modeling and User-Adapted Interaction* 19 (1-2): 5-33.

Steinbach, M., G. Karypis, and V. Kumar. 2000. *Proceedings of Workshop on Text Mining, Sixth ACMKDD International Conference on Knowledge Discovery and Data Mining (KDD'00), August 20, 2000: A Comparison of Document Clustering Techniques*. Boston, UAS: ACM Press

Stermsek, G., M. Strembeck, and G. Neumann. 2007. *Proceedings of the International Conference on Information and Knowledge Engineering (IKE), A User Profile Derivation Approach based on Log-File Analysis*. Las Vegas, Nevada: CSREA Press

Straub, D., D. Gefen, and M.-C. Boudreau. 2005. Quantitative Research. In *Research in Information Systems: A Handbook for Research Supervisors and Their Students*, ed. D. Avison and J. Pries-Heje, 221-238. Burlington: Elsevier Butterworth-Heinemann.

Sugar, C. A., and G. M. James. 2003. Finding the Number of Clusters in a Dataset: An Information-Theoretic Approach. *Journal of American Statistical Association* 98 (463): 750-763.

Sun, A., E.-P. Lim, and W.-K. Ng. 2002. *Proceedings of the 4th International Workshop on Web Information and Data Management (WIDM 2002), November 4-9, 2002: Web Classification Using Support Vector Machine*. Virginia, USA: ACM Press

Tanudjaja, F., and L. Mui. 2002. *Proceedings of the 35th Hawaii International Conference on System Sciences (HICSS-35'02), January 7-10, 2002: Persona: A Contextualized and Personalized Web Search*. Big Island, Hawaii: IEEE Computer Society

Taylor, H. M., and S. Karlin. 1998. *An Introduction to Stochastic Modeling*. 3rd ed. San Diego: Academic Press.

Toutanova, K., F. Chen, K. Popat, and T. Hofmann. 2001. *Proceedings of the 2001 ACM CIKM International Conference on Information and Knowledge Management (CIKM'01), November 5-10, 2001: Text Classification in a Hierarchical Mixture Model for Small Training Sets*. Atlanta, Georgia, USA: AMC Press

Trajkova, J., and S. Gauch. 2004. *Proceedings of RIAO 2004, April 26-28, 2004: Improving Ontology-Based User Profiles*. Avignon, France:

Treeratpituk, P., and J. Callan. 2006. *Proceedings of the 2006 international conference on Digital government research, May 21-24, 2006: Automatically labeling hierarchical clusters*. San Diego, California: ACM Press

Tsymbal, A. 2004. *The problem of concept drift: definitions and related work*. Trinity College Dublin. https://www.cs.tcd.ie/publications/tech-reports/reports.04/TCD-CS-2004-15.pdf (accessed October 12, 2007).

Vaishnavi, V., and W. L. Kuechler. 2007. *Design Research in Information Systems*. http://www.isworld.org/Researchdesign/drisISworld.htm (accessed August 29, 2007).

Valiant, L. G. November 1984. A Theory of the Learnable. *Communications of the ACM* 27 (11): 1134-1142.

Vapnik, V. N. 1999. *The Nature of Statistical Learning Theory*. 2nd ed. New York: Springer-Verlag.

Varma, V. 2002. *the Proceedings of the Language Engineering Conference (LEC 02), December 13-15, 2002: Building large scale ontology networks*. Hyderabad, India: IEEE Computer Society

Vattani, A. 2009. *Proceedings of the 25th Annual Symposium on Computational Geometry, June 8-10, 2009: k-means Requires Exponentially Many Iterations Even in the Plane*. Aarhus, Denmark: ACM Press

Venable, J. R. 2006a. *Proceedings of the 2006 Information Resource Management Association Conference, May 21-24, 2006: A Framework for Design Science Research Activities*. Washington: Idea Group Publishing

--------. 2006b. *Proceedings of the first International Conference on Design Science Research in Information Systems and Technology, February 24-25, 2006: The Role of Theory and Theorising in Design Science Research*. Claremont, CA.: CGU

Voorhees, E. M. 1999. *Proceedings of the 8th Text REtrieval Conference, November 17-19, 1999: The TREC-8 question answering track report*. Gaithersburg, Maryland USA: Department of Commerce, NIST

--------. 2005a. *Proceedings of the Fourteenth Text Retrieval Conference (TREC 2005), November 15-18, 2005: Common Evaluation Measures*. Gaithersburg, Maryland. http://trec.nist.gov/pubs/trec14/t14_proceedings.html (accessed August 25, 2006).

--------. 2005b. *Proceedings of the Fourteenth Text Retrieval Conference (TREC 2005), November 15-18, 2005: Overview of TREC 2005*. Gaithersburg, Maryland. http://trec.nist.gov/pubs/trec14/t14_proceedings.html (accessed August 25, 2006).

Wang, T.-Y., and H.-M. Chiang. 2007. Fuzzy support vector machine for multi-class text categorization. *information Processing and Management* 43 (4): 914-929.

Wang, Y., and J. Hodges. 2005. *Proceedings of the 2005 International Conference on Artificial Intelligence, ICAI 2005, June 27-30, 2005: Document Clustering using Compound Words*. Las Vegas, Nevada, USA: CSREA Press

Webb, G. I., M. J. Pazzani, and D. Billsus. 2001. Machine Learning for User Modeling. *User Modeling and User-Adapted Interaction* 11 (1-2): 19-29.

Wei, C.-P., C. C. Yang, and C.-M. Lin. 2008. A Latent Semantic Indexing-based approach to multilingual document clustering. *Decision Support Systems* 45 (3): 606-620.

Weigend, A. S., and J. O. Pedersen. Oct 1999. Exploiting Hierarchy in Text Categorization. *Information Retrieval* 1 (3): 193-216.

Wen, T., A. Edelman, and D. Gorsich. 2003. A Fast Projected Conjugate Gradient Algorithm for Training Support Vector Machines. In *Fast Algorithms for Structured Matrices: Theory and Applications*, ed. O. Vadim, 245-263. Boston: American Mathematical Society.

Wiener, E., J. O. Pedersen, and A. S. Weigend. 1995a. *Proceedings of 4th Annual Symposium on Document Analysis and Information Information Retrieval (SDAIR-95), A Neural Network Approach to Topic Spotting*. Las Vegas, NV:

--------. 1995b. *Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval, April 24-26, 1995: A Neural Network Approach to Topic Spotting*. Las Vegas, NV.:

Witten, I. H., A. Moffat, and T. C. Bell. 1999. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Edited by E. Fox, *Multimedia Information and Systems*. San Diego: Morgan Kaufmann.

Wu, B., and B. D. Davison. 2006. *Proceedings of the 15th international conference on World Wide Web (WWW 2006), May 23-26, 2006: Detecting Semantic Cloaking on the Web*. Edinburgh, Scotland: ACM Press

Xu, G. 2008. Web Mining for Recommendation and Personalization, The School of Computer Science & Mathematics, Victoria University, Australia, Melbourne

Xu, Y., and Z. Chen. 2006. Relevance Judgment: What Do Information Users Consider Beyond Topicality? *journal of the American Society for Information Science and Technology* 57 (7): 961-973.

Yang, Y. 1999. An Evaluation of Statistical Approaches to Text Categorization. *Information Retrieval* 1 (2): 69-90.

--------. 2001. *Proceedings of the 24th ACM International Conference on Research and Development in Information Retrieval (SIGIR '01), September 9-13, 2001: A Study on Thresholding Strategies for Text Categorization*. New Orleans, Louisiana, USA: ACM Press

Yang, Y., C. G. Chute, and M. Clinic. July 1994. An Example-Based Mapping Method for Text Categorization and Retrieval. *ACM Transactions on Information Systems* 12 (3): 252-277.

Yang, Y., and X. Liu. 1999. *Proceedings of the 22nd ACM International Conference on Research and Development in Information Retrieval (SIGIR '99), August 15-19, 1999: A Re-examination of Text Categorization Methods*. Berkeley, CA: ACM Press

Yang, Y., and J. O. Pedersen. 1997. *Proceedings of the 14th International Conference on Machine Learning (ICML-97), July 8-12, 1997: A Comparative Study on Feature Selection in Text Categorization*. San Francisco: Morgan Kaufmann Publishers

Yang, Y., S. Slattery, and R. Ghani. 2002. A Study of Approaches to Hypertext Categorizatio. *Journal of Intelligent Information Systems* 18 (2-3): 219-241.

Yang, Y., J. Zhang, and B. Kisiel. 2003. *Proceedings of the 26th Annual International ACM Conference on Research and Development in Information Retrieval, July 28 - August 1, 2003: A Scalablity Analysis of Classifiers in Text Categorization*. Toronto, Canada: ACM Press

Ye, H., and B. W. N. Lo. 2000. Feature Competitive Algorithm for Dimension Reduction of the Self-Organizing Map Input Space. *Applied Intelligence* 13 (3): 215-230.

Yu, L., K. K. Lai, S. Wang, and W. Huang. 2006. A Bias-Variance-Complexity Trade-Off Framework for Complex System Modeling. In *Computational Science and Its Applications - ICCSA 2006: International Conference*, ed. M. L. Gavrilova, O. Gervasi, V. Kumar, C. J. K. Tan, D. Taniar, A. Laganà, Y. Mun and H. Choo, 518-527. Berlin Heidelberg: Springer-Verlag.

Zamir, O., and O. Etzioni. 1998. *Proceedings of the 19th ACM International Conference on Research and Development in Information Retrieval (SIGIR-98), August 24-28, 1998:*

Web Document Clustering: A Feasibility Demonstration. Melbourne, Australia: ACM Press

--------. 1999. *Proceedings of the Eighth International World Wide Web Conference (WWW8), May 11-14, 1999: Grouper: A Dynamic Clustering Interface to Web Search Results*. Toronto, Canada: Elsevier

Zeng, H.-J., Q.-C. He, Z. Chen, W.-Y. Ma, and J. Ma. 2004. *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 25-29, 2004: Learning to Cluster Web Search Results*. Sheffield, UK: ACM Press

Zhai, C. 2008. Statistical Language Models for Information Retrieval A Critical Review. *Foundations and Trends in Information Retrieval* 2 (3): 137-213.

--------. 2009. *Statistical Language Models for Information Retrieval*. Edited by G. Hirst, *Synthesis Lectures on Human LanguageTechnologies*. San Rafael: Morgan & Claypool.

Zhai, C., and J. Lafferty. 2001. *Proceedings of the 24th ACM International Conference on Research and Development in Information Retrieval (SIGIR '01), September 9-13, 2001: A Study of Smoothing Methods for Language Models Applied to Ad Hoc Information Retrieval*. New Orleans, Louisiana, USA: ACM Press

Zhao, Y., and G. Karypis. 2002. *Proceedings of the eleventh international conference on Information and knowledge management (CIKM'02), November 4-9, 2002: Evaluation of Hierarchical Clustering Algorithms for Document Datasets*. McLean, Virginia, USA.: ACM Press

Zhu, D. 2007. Improving the Relevance of Search Results via Search-term Disambiguation and Ontological Filtering. Master diss., School of Information Systems, Curtin Business School, Curtin University of Technology, Perth. http://adt.curtin.edu.au/theses/available/adt-WCU20080331.163016 (accessed December 20, 2007).

--------. 2008a. *Proceedings of the 2008 Curtin Business School Doctoral Colloquium, October 2-3, 2008: Using the Open Directory Project for Web Information Retrieval: A Survey*. Perth, Australia: Curtin University of Technology

--------. 2009. *Improving the Relevance of Search Results: Search-term Disambiguation and Ontological Filtering*. Saarbrucken: VDM Verlag.

Zhu, D., and H. Dreher. 2007. *Proceedings of the Inaugural IEEE Digital Ecosystems and Technologies Conference, February 21-23, 2007: Determining and Satisfying Search Users Real Needs via Socially Constructed Search Concept Classification*. Cairns, Australia: IEEE

--------. 2008a. Improving Web Search by Categorization, Clustering, and Personalization. In *LNAI 5139, Advanced Data Mining and Applications, The Fourth International Conference ADMA*, ed. C. Tang, 659-666. Berlin, Heidelberg: Springer Verlag.

--------. 2008b. *Proceedings of the Second IEEE Digital Ecosystems and Technologies Conference, February 26-29, 2008: IR Issues for Digital Ecosystems Users*. Phitsanulok, Thailand: IEEE

--------. 2008c. *Proceedings of the Second IEEE Digital Ecosystems and Technologies Conference, February 26-29, 2008: Personalized Information Retrieval in Digital Ecosystems*. Phitsanulok, Thailand: IEEE

--------. 2009. Discovering Semantic Aspects of Socially Constructed Knowledge Hierarchy to Boost the Relevance of Web Searching. *Journal of Univeral Computer Science* 15 (8): 1685-1710. http://www.jucs.org/jucs_15_8/discovering_semantic_aspects_of (accessed October 23, 2009).

--------. 2010. Exploring semantic characteristics of socially constructed knowledge repository to optimize Web search. In *Ontologies Driven Web Mining: Concepts and Techniques*, ed. H. O. Nigro and S. E. G. CisaroHershey: IGI Global.

Zhu, X. 2008b. *Semi-Supervised Learning Literature Survey*. University of Wisconsin - Madison.

Zhuge, H. 2010. Interactive semantics. *Artificial Intelligence* 174 (2): 190-204.

Zhuge, H., Y. Xing, and P. Shi. 2008. Resource Space Model, OWL and Database: Mapping and Ingegration. *ACM Transactions on Internet Technology (TOIT)* 8 (4): Article 20.