A Query Categorization Technique Using

Categories Descriptor Files

Prepared By

Ulfat Mohammed Abdel Qader

Supervised By

Professor Walid Salameh

Submitted to the Department of Computer Science at
Amman Arab University of Graduate Studies
in partial fulfillment of the requirements for the degree of
MASTER OF COMPUTER SCIENCE

Graduate Collage of Computing Studies

Amman Arab University for Graduate Studies

September 2006

DELEGATION

I am, Ulfat Mohammed Abdel Qader ,

delegate Amman Arab University for Graduate Studies to supply libraries, organizations or individuals with copies of this thesis upon their request.

Name: Ulfat Mohammed Abdel Qader

Signature: ________________________

Date: ____________________________

DECISION OF THE EXAMINING COMMITTEE

This Thesis, titled **(A Query Categorization Technique Using Categories Descriptor File**s) was successfully defended and approved on 16.September 2006.

Committee Members:		Signature
______________________________	Chair	______________
______________________________	Member	______________
______________________________	Member	______________
______________________________	Supervisor & Member	______________

ACKNOWLEDGMENT

There are so many who have made this work possible that they cannot be listed in one page.

First, I would like to thank my advisor Dr. Walid Salameh who supported me during a whole year of work on my thesis. He has given me good advice and directions while still allowing me to go on my own.

Second, I should never forget thanking my family for their loving support, and to my dear friends, who had a deep belief in my ability to fulfill this work, and supported me during the many hard times I had passed through, and never hesitated to give any advice or help.

Special thanks go to Rawan Khatieb, who was my best colleague during my master degree and with whom I have always shared fears and hopes.

Also my thanks to my teachers, especially Dr. Hussein Al-Bahadili, who created the desire in me to be distinguished and to look forward to contributing to life and never be a mere viewer.

Special thanks are reserved for Dina Baker and Reham Saleh who helped me collect data, and they never complained about the long times they spent at the desktop just to help me. They always tried their best to accommodate me and make me feel better

Special thanks to Mr Waseem Swedan, who always found a good and fast solution for every problem and supported me with his wide experience. Lastly, I can't forget thanking my manager Mr Hassan al Zoubi who helped me during the last two years by creating such a wonderful accommodating working environment that enabled me to balance between my work and study.

DEDICATION

To the soul of my Mother

To my soul Mate

To all my Teachers

To my Friends who left an impact on me

and supported me through my life

One crowded hour of glorious life is worth an age without a name

TABLE OF CONTENTS

LIST OF TABLES

List of Figures

FIGURES:

EQUATIONS:

LIST OF APPENDICES

NOMENCLATURE

Ω	Document sets consist of (training set),(validation set) and (test set).
a_{ij}	Weight of the word *i* in document *j*
C4.5	Decision Tree algorithm
CB	Concept Vector-Based
CSV	Categorization Status Value
DBMS	Database Management System
e-mail	Electronic mail
HAC	Hierarchical Agglomerative Clustering
H-CB	Hierarchical Concept Vector Based
H-NB	Hierarchical Naive Bayesian
IDF	Inverse Document Frequency
IR	Information Retrieval
KDD	Knowledge Discovery & Data Mining
KE	Knowledge Engineering
kNN	k-Nearest Neighbor
LLSF	Linear Least Squares Fit
ML	Machine Learning
NB	Naive Bayesian
RAM	Random Access Memory
SVD	Singular Value Decomposition

SVM – Support Vector Machine

Tf – Term frequency

Tf-idf – Term frequency inverse document frequency – term weighing scheme

الملخــص

تقســيم أوامر البحث بإستخدام ملفــــات لوصف التقســيمات

إعداد

الفت محمد احمد عبد القادر

إشراف

الاستاذ الدكتور وليد سلامه

Arabic Summary

يتناول هذا البحث تقديم طريقة جديدة لتصـنيف أوامر البحث على محركات البحث بهدف تسـريع و تسـهيل عملية البحث التي يقوم بهــا الكثير من مستخدمي الانترنت على اختلاف مستوياتهم من مبتدئ الى متخصص.و يعتمد البحث على ربط أوامر البحث بمجموعات من التصنيفات التي تشكل كل منها موضوعا من موضوعـــات البحث المختلفة. يقـــــوم البحث على إنشاء مجموعة من الملفات الواصفة لمجموعة التقسيمات المستخدمة، حيث تم الاعتماد على مصـــادر المعلومات المتواجدة على الانترنت لجمع المعلومات المتعلقة بكل بــــاب من أبواب التصـنيف في ملف منفصـل ثم تطبيق إحدى أسـاليب التصنيف المعروفة لقياس درجة الارتباط بين أوامر البحث و الملفات الوصفية. و عليه يمكن تحديد مجموعة التصنيفات التي ينتمي إليها أمر البحث المستخدم.

ان تصـنيف اوامر البحث ساهم حسب نتائج البحث في ازالة الغموض عن اوامر البحث لتحديد المعنى الصحيح لهـا من خلال ربطها بمجموعة من التصــنيفات التي من شـــأنها تحديد المعاني المختلفة التي ممكن ان تحملها أوامر البحث

A Query Categorization Technique Using

Prepared By

Ulfat Mohammed Abdel Qader

Supervised By

Professor Walid Salameh

Abstract

In this thesis, an approach for query-categorization is proposed in order to facilitate the process of Web taxonomy constructing. We propose an approach to categorize the user’s Web query terms into a pre-defined category names based on query terms relevant to these categorizes that reflect the user’s search interests. The Approach is mainly based on constructing a descriptor file for each category name using the available knowledge resources over the Web, and then applying text classification techniques to measure the relevance between query terms and the whole set of descriptor files.

The obtained experimental results show the effectiveness of the approach in reducing the human effort during the searching process by proposing the best categories for the user’s input query term and in capturing the intended meaning of query terms.

CHAPTER 1

INTRODUCTION

1.1 Historical Background

During the last 10 years, information Retrieval (IR) has occupied a prominent status in the information system field. There was a need for some techniques to accomplish the task of accessing the increased amount of digital documents. Text categorization - also known as text classification or topic spotting - is the process of labeling natural language texts with some categories names of a predefined sets (Sebastiani, 2002). Text categorization was as such a suitable process to perform that task.

Text categorization dates back to the early 60's, but it started to become a major subfield of IR in the early 90's. That transformation can be explained by the increased applicative interest and the availability of more powerful hardware. Text categorization is now being applied in many contexts, ranging from document indexing based on a controlled vocabulary, to document filtering, automated metadata generation, word sense disambiguation, population of hierarchical catalogues of Web resources, and in general any application requiring document organization or elective and adaptive document dispatching (Sebastiani, 2002).

The most popular text categorization approach appeared in the late ’80s. It was called knowledge engineering (KE), KE consisting of manually defined set of rules encoding expert knowledge on how to classify documents under the given categories. However, the popularity of that approach was lost in the ’90s because of the machine learning (ML) paradigm, which is the process that automatically builds an automatic text classifier by learning - using a set of pre-classified documents - the characteristics of the categories of interest (Sebastiani, 2002). ML has advantages over the human experts in terms of: (i) delivering better accuracy in comparison to that achieved by human experts; (ii) considerable savings in terms of expert labor power; and (iii) its porting to a different set of categories.

Current-day text categorization is thus a discipline at the crossroads of ML and IR, and as such, it shares a number of characteristics with other tasks such as information/knowledge extraction from texts and text mining. There is still considerable debate about where the exact border between these disciplines lies, and the terminology is still evolving.

The term “Text mining” is used to denote all the tasks which by analyzing large quantities of text and detecting usage patterns, try to extract potentially useful information. According to this view, Text Categorization is an instance of text mining.

As a note, we should clarify the term “automatic text classification” since it tends to confuse the reader most of the time. Aside from (i) the automatic assignment of documents to a predefined set of categories, which is the intended meaning used in this research. The term has also been used to mean (ii) the automatic identification of such a set of categories or (iii) the automatic identification of such a set of categories and the grouping of documents under them, a task usually called text clustering, or (iv) any activity of placing text items into groups.

1.2 Query Categorization

Most of web search queries submitted by user space contain only two or three terms, thus they provide very limited information about the actual meaning of the user's requests. Studying the query data by trying to find search patterns and understanding user intents, utilizing this information is a key factor to construct an effective web search engines. One way of approaching this problem computationally is to approximate the intended meaning of a query by a node, or a set of nodes in a given subject taxonomy.

For instance, a query “the raven” can indicate that a user searches for information on entertainment/movies or on zoology. Thus, the intuitive problem of capturing the intended meaning of a query is reduced to the computational problem of mapping the query string to a set of nodes in a given fixed, but arbitrary subject taxonomy.

Query Categorization problem can be stated as follows: How can we use a set of existing, pre-defined subject taxonomy to place a given query terms into proper categories. For example, the appropriate category for the query term "Real Player" may be Computer / Company or Computer / Software Download.

Text categorization is a well-known topic in information retrieval and text mining fields. Most work in this area has been focused on categorizing web pages or longer text or corpus. However, search query categorization is very different in the sense that query terms are usually very short and with implicit and subjective user intents. Therefore, how to understand automatically user search intents given the search query terms would be very interesting to IR and text mining researches.

In this work, we intend to propose an approach to organize query terms[1] into a subject taxonomy, where deeper analysis of domain-specific terminology and further discovery of term relationships can be performed under each subject domain.

1.3 Statement of Problem

Commonly, users attempt to use too short query terms to describe the field of their search interests. Since too short query terms do not contain enough information to target the most related documents relevant for user query, an attempt to categorize query terms into a set of related categories can save the effort of browsing a large amount of retrieved documents. This automatic categorization of user query terms could ease the process of search through the Internet, contributes to the query clustering research field and most importantly helps the user to hit the documents that he/she searches for in a reasonable time by increasing the degree of precision and recall of his search process.

Query Categorization is defined as the problem of automatically assigning pre-defined categories to query terms according to the terms' supposed search interests or information needs. More formally, let C be a pre-defined

[1] Query Terms we mention are the complete query strings submitted by users to search services. In the rest of this document, the terms “query” and “terms” refer to query terms when they are used without any additional explanation.

taxonomy, and let $V = \{w_1; w_2; \ldots; w_n\}$ be the set of all query terms that have been categorized into C properly. Query categorization is used to determine a category set $C_{(t)} \subseteq C$ for a given unknown query term $t \notin V$, so each $c \notin C_{(t)}$ is considered as a possible category t is related to (Chuang and Chien, 2003 (a)).

The solution for this query categorization problem is to build an automatic approach that will enable user query terms to be mapped to its matched categories among set of categories classified into a pre-defined taxonomy. To accomplish this solution we had to use a classification method based on a term feature vector and an inner-product similarity measure is applied to this query terms categorization task.

1.4 Stated Theory

The introduced theory in this thesis concentrates on ensuring that the performance of our proposed approach used to categorize user query terms is effective in terms of top *n* inclusion rate measurement. This latter measurement is defined as “the rate of instances whose highly ranked *n* candidate classes contain the correct class (es)" (Chuang and Chien, 2003 (b)).Hypothesis state that: *the proposed query categorization technique that* *rely on using descriptor files for category classes has an effective performance in terms of top n inclusion rate measurement.*

1.5 Research Goal

This research aims to develop an automatic approach for categorizing user's search query terms. The proposed approach aims in the first place to ease the search process over the Web by mapping the user's query terms to a set of categories. This mapping can contribute to revealing the ambiguity that usually search query terms suffer from and guide the user to capture the intended meaning of his/her search request. On the other hand, the proposed query categorization approach aims to introduce a new technique for categorizing query terms using descriptor files.

1.6 Related Work

Few reports in the literature have focused on the Web query terms categorization problem. Recently, some document-based approaches to acquiring domain-specific semantic lexicons or to enrich the existing ontology's have been proposed (Riloff and Shepherd, 1997; Chien, 1999 ; Avancini, Rauber and Sebastiani, 2002)

Capturing the intended meaning of a search query term with the help of word sense disambiguation has been heavily investigated. Word sense disambiguation is often resolved implicitly when query terms are sufficiently long; since web query consists of two or three words in average; many studies conducted to demonstrate that word sense disambiguation can improve information retrieval. (Schutze and Pederson, 1994 ; Gonzalo, Verdejo, Chugur and etal ., 1998; Stokoe, Oakes and Tait, 2003)

Other researchers have focused on query expansion or reformulation. Query expansion is defined as the process of “rewriting the query Q posed to a global schema G into a new query Q`, in such a way that all the knowledge about the constraints in G has been “compiled” into Q` (Zoran, 2003)". Bruza and Dennis developed a hyper-index to provide the user with query reformulation recommendations (Bruze and Dennis, 1997; Dennis, Bruza and MacArthur, 2002), and that differs from this work, which attempts to reformulate the category names instead of query terms by using descriptor file to expand the intended meaning of each category name.

Agrawal and Srikant have developed an enhanced Naive Bayes algorithm that is based on the intuition that if two documents share their category in one taxonomy, they are likely also to share their category in another taxonomy (Agrawal and Srikant, 2001). Different authors have studied integrating objects from different taxonomies. David Vogel, Steffen Bickel, and Peter Haider have conducted a study to find out a solution for this problem based on a taxonomic mapping between a web directory and the subject taxonomy while this thesis aimed to map query terms to a category name that represent a subject taxonomy (Vogal, Bickel and Haider, 2005).

Several research studies have been conducted on term clustering, such as the works on latent semantics, Singular Value Decomposition **(**SVD) algorithm, term relationship analysis etc (Deerwester, Dumais, Furnas and etal., 1990; Mandala, Tokunaga and Tanaka, 1999; Sanderson and Croft, 1999). Most of them have dealt with the automatic clustering of con-trolled indexed terms into clusters and with using them to assist IR systems in the query expansion process so as to improve the precision and recall ratios. On the other hand, this thesis uses a descriptor file to expand the meaning of each category name instead of query terms and without using any cluster of indexed terms.

In addition, research has been conducted on web query terms to characterize them according to their geographical locality. Anther study conducted to introduce several alternatives for automatically and efficiently categorizing query terms using variety of state-of-the-art machine learning tools (Gravano, Hatzivassiloglou and Lichtenstien, 2003).

There is a number of published works on terminology analysis for Web information retrieval (Anick and Tipirneni, 1999 ; Bruze, MacArthur and Dennis, 2000) and a similar work on the clustering of Web query terms (Beeferman and Berger, 2000) in which a collection of user transactions from an Internet search engine are mined to discover clusters of similar query terms. The clustered queries could then be used for term suggestion to help users form their own search requests. This research differs from this work in that it tries to structure each unknown query term into appropriate and meaningful subject categories, and provide possible corresponding subject domains for each unknown term. That is to a certain degree similar to the study conducted by Shui-Lung Chuang and Lee-Feng Chien, where in their study the authors attempted to categorize query terms by adopting the query logs as the terminology source, thus avoiding term segmentation and extraction problems commonly faced in document-based approaches (Chuang and Chien, 2003 (b)).

Chuang and Chien study has been used as a comparative study to the proposed query categorization approach in this thesis. The results have been used as a comparative base for the results gained from applying the proposed approach to a set of descriptor files for each category in order to resolve query ambiguity.

1.7 Contribution

This research contributes by proposing a new approach of categorizing user query terms. To categorize query terms that contain usually two or three words would not be an effective approach without additional information that describe these query terms.

Most query categorizing researches used extra information to help in categorizing query terms in a set of subject taxonomy. Query terms would be less ambiguous and meaningful if they are quite long, that is why extra information about query terms is essential to provide sufficient context to resolve the intended meaning.

Different researchers have used different sources of information to expand query terms context, some used background knowledge (Vogal and etal., 2005) to help in query reformulation, another researcher categorized query terms according to their geographical locality (Chuang and Chien, 2003 (b)). Also, search engines logs have been used to categories query terms as they can provide an extra knowledge about query terms.

This research contributes by introducing a new approach to help in query reformulation in order to achieve a better mapping between query terms and arbitrary set of categories. This approach is mainly based on using a descriptor set of files collected from Wikipedia[٢] encyclopedia, those descriptor files are used to provide extra information about each category and they are used by IR text classification techniques to map between set of categories and the target user query terms.

1.8 Road Map

The roadmap for the remainder of this thesis is as follows:

This chapter defined text categorization and outlined the contributions of this work along with a literature review of the most popular text categorization approaches and ends with stating the theory the thesis is based on.

- Chapter 2 introduces an overview of some query categorization approaches and their techniques and the evaluation criteria used to asses different query categorization approaches.
- Chapter 3 describes the case study that was used to compare this work with. The constraints of this case study, the data sets and experiments the case study conducted, and it ends with a discussion of the results the researchers have achieved.
- Chapter 4 describes the proposed approach of this thesis, the algorithms used to develop the simulator, the experiments conducted, the text categorization methods applied in order to implement the

[٢] Wikipedia : http://www.wikipedia.org/

- simulator and finally ends with a discussion of the achieved results compared with those gained by the case study.

- Chapter 5 draws conclusions and describes possible areas of future work

CHAPTER 2

THEORY AND BACKGROUND

2.1 Text Categorization

Categorization is a problem that cognitive psychologists have dealt with for many years. In the process of categorizing electronic documents, categories are typically used as a means of organizing and getting an overview of the information in a collection of several documents. Folders in electronic mail (e-mail) and topics in Usenet News are a couple of concrete examples of categories in computer-mediated communication. Furthermore, the information within a document is normally organized by some kind of structure, either physical (layout) or logical (context), or (at the best) both.

Text categorization is defined in this thesis as an information retrieval task in which one or more category labels are assigned to a document. Mathematically, Text categorization is the “Task of approximating the unknown target function Φ: $D \times C \rightarrow \{T, F\}$ (that describes how documents ought to be classified) by means of a function Φ: $D \times C \rightarrow \{T, F\}$ called the classifier, where $C = \{c_1, . . . , c_{|C|}\}$ is a predefined set of categories and D is a domain of documents. If $\Phi(d_j, c_i) = T$, then d_j is called a positive example (or a member) of c_i, while if Φ (d_j, c_i) = F it is called a negative example of c_i. (Avancini and etal., 2002)".

Text categorization is a subjective task: when two experts (human or artificial) decide whether to classify document d_j under category c_i they may disagree, in fact, this happens with relatively high frequency.

Depending on the application, text categorization may be either single-label where exactly one $ci \in C$ must be assigned to each $dj \in D$, or multi-label where any number $0 \le n_j \le |C|$ of categories may be assigned to a document $dj \in D$. A special case of single-label text categorization is binary text categorization, in which, given a category c_i, each $dj \in D$ must be assigned either to c_i or to its complement c_i. Multi-label text categorization under $C = \{c_1,\ldots,c_{|C|}\}$ is usually tackled as |C| independent binary classification problems under $\{c_i, c_i\}$, for $i = 1,..,|C|$. A classifier for c_i is then a function Φ_i: $D \rightarrow \{T, F\}$ that approximates the unknown target function Φ_i: $D \rightarrow \{T,F\}$ (Avancini and etal., 2002).

Text categorization system consists of three different phases: document indexing, classifier learning, and classifier evaluation. The three following paragraphs are devoted to these three phases, respectively; for a more detailed treatment.

2.1.1 Document Indexing

"Document indexing denotes the activity of mapping a document d_j into a compact representation of its content that can be directly interpreted (i) by a classifier-building algorithm and (ii) by a classifier, once it has been built. (Avancini and etal., 2002)".

Text Categorization usually employs document indexing methods borrowed from IR, where a text is represented as a vector of term weights $d_j = (w_{1j},\ldots, w_{|T|j})$. Here, T is the dictionary, i.e. the set of terms (also known as features) that occur at least once in at least k documents, and $0 \le w_{kj} \le 1$ quantifies the importance of t_k in characterizing the semantics of d_j. Typical values of k are between 1 and 5.

An indexing method is characterized by firstly defining what a term is, the most frequent choice is to identify terms either with the word occurring in the document, or with their stems (i.e their morphological roots); and secondly, a method to compute the words weights. This function can be accomplished either by using statistical or probabilistic techniques. Statistical techniques are the most popular option. One popular class of statistical term weighting functions is tf×idf ,where two intuitions are at play: (a) the more frequently t_k occurs in d_j, the more important t_k for d_j (the term frequency intuition); (b) the more documents t_k occurs in, the less discriminating it is, i.e. the smaller its contribution is in characterizing the semantics of a document in which it occurs (the inverse document frequency intuition). Weights computed by tf×idf techniques are often normalized so as to contrast the tendency of tf×idf to emphasize long documents (Avancini and etal., 2002).

2.1.2 Classifier Learning

A text classifier for c_i is automatically generated by a general inductive process (the learner), which by observing the characteristics of a set of documents pre-classified under c_i or c_i, captures the characteristics that a new unseen document should have in order to belong to c_i.

Different learners have been applied in the text categorization literature, including probabilistic methods, regression methods, decision tree and decision rule learners, neural networks, batch and incremental learners of linear classifiers, example-based methods, support vector machines, genetic algorithms, hidden Markov model that is a statistical model where the system being modeled is assumed to be a Markov process with unknown parameters, and the challenge is to determine the hidden parameters from the observable parameters, and classifier committees. Some of these methods generate binary-valued classifiers of the required form $\Phi: D \times C \rightarrow \{T, F\}$, but some others generate real-valued functions of the form (Categorization Status Value) CSV: $D \times C \rightarrow [0, 1]$. For the latter, a set of thresholds τ_i needs to be determined (typically, by experimentation on a validation set) allowing to turn real-valued CSVs into the final binary decisions (Avancini and etal., 2002).

2.1.3 Classifier Evaluation

Both training efficiency (i.e. average time required to build a classifier Φ_i from a given corpus Ω), classification efficiency (i.e. average time required to classify a document by means of Φ_i), and effectiveness (i.e. average correctness of Φ_i's classification behavior) are measures of success for a learner. However, effectiveness is usually considered the most important criterion, since in most applications one is willing to trade training time and classification time for correct decisions.

In text categorization, effectiveness is often measured by a combination of precision (π), the percentage of positive categorization decisions that are correct, and recall (ρ), the percentage of positive, correct categorization decisions that are actually taken. Since a classifier can be tuned to emphasize one at the expense of the other, only combinations of the two are significant, the most popular combination between them is defined by Equation (2.1) (Avancini etal,2002).

$$F1 = \frac{2\pi\rho}{\pi + \rho} \qquad (2.1)$$

2.2 Text Categorization Methods

Classification methods in general aim to map between a set of different objects and their best match class/es. The input to these methods is a set of objects (i.e., training data), and the output is the classes which these objects belong to (i.e., dependent variables) and a set of variables describing different characteristics of the objects (i.e., independent variables).

Classifiers in general fall into two classes: "generative models which initially estimate class conditional densities P (document/class) and discriminant models which directly estimate the posterior probabilities P (class/document) (Menon, 2004)."

Once such a predictive model is built, it can be used to predict the class of the objects for which class information is not known as a prior. The key advantage of supervised learning methods over unsupervised methods (for example, clustering) is that by having an explicit knowledge of the classes the different objects belong to, these algorithms can perform an effective feature selection if that leads to better prediction accuracy.

Following is a brief overview of some classification algorithms that have been used in data mining and machine learning fields.

2.2.1 Vector Space Model

Vector spaces are commonly used in existing methods of retrieval and categorization for handling word distribution over documents or categories. We use vector spaces for a different purpose; that is, we use two vector spaces to define and to solve a transformation between two vocabularies. For document categorization, we define a source space using document words as the dimensions and a target space using category identifiers as the dimensions; also, we define the transformation from the source space to the target space. We represent the training documents as source vectors, and their categories as target vectors. We give the source-vector/target-vector pairs as the input to an Linear Least Squares Fit (LLSF) (Yang and Chute, 1994) algorithm that computes a likely transformation from a source vector to a target vector, that is, from a document representation using its own words to a representation using categories.

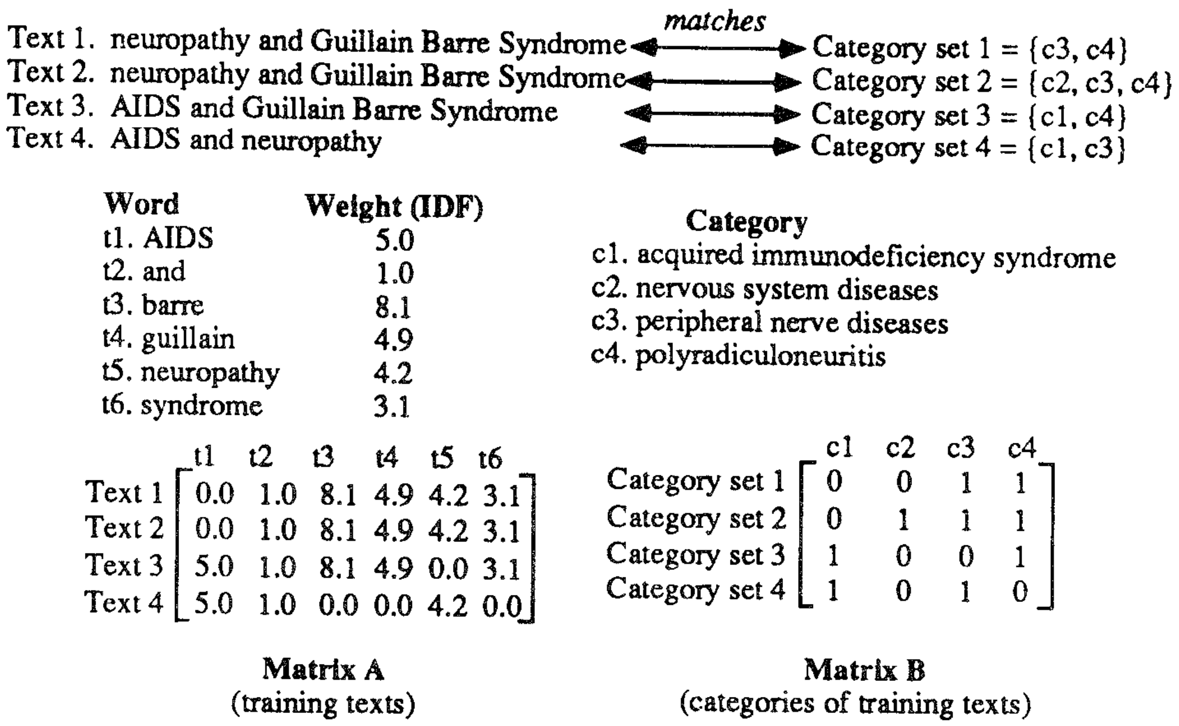
Figure 1 shows an example of the matrix representation of a training set (Yang and Chute, 1994). There are four simplified documents (texts), each assigned to a set of categories. We use two matrices, A and B to represent the training set. A row in matrix A represents a document, and the corresponding row in matrix B represents the category set of the document. The columns of matrix A are words, and columns of matrix B are categories. Element a_{ij} of matrix A is the weight of word w_j in the i^{th} document. Equation (2.2) represents the formula used to calculate the value of a_{ij} (Yang and Chute, 1994). We adopted the following word-weighting schemes as options of a_{ij}.

Option 1: A binary weight, i.e., $a_{ij} = 1$ if word w_j is present in the i^{th} document, and $a_{ij} = 0$ otherwise.

Option 2: The within-document word frequency (TF), i.e., $a_{ij} = TF_{ij}$ = the number of times word w_j occurs in the i^{th} document,

Option 3: The Inverse Document Frequency (IDF), i.e,

$$a_{ij} = \mathrm{IDF}_j = \log\left(\frac{\text{number of documents in the entire collection}}{\text{number of documents with word } w_j}\right) + 1. \quad (2.2)$$



		matches	
Text 1.	neuropathy and Guillain Barre Syndrome	⟷	Category set 1 = {c3, c4}
Text 2.	neuropathy and Guillain Barre Syndrome	⟷	Category set 2 = {c2, c3, c4}
Text 3.	AIDS and Guillain Barre Syndrome	⟷	Category set 3 = {c1, c4}
Text 4.	AIDS and neuropathy	⟷	Category set 4 = {c1, c3}

Word	Weight (IDF)
t1. AIDS	5.0
t2. and	1.0
t3. barre	8.1
t4. guillain	4.9
t5. neuropathy	4.2
t6. syndrome	3.1

Category
c1. acquired immunodeficiency syndrome
c2. nervous system diseases
c3. peripheral nerve diseases
c4. polyradiculoneuritis

$$\begin{array}{c|cccccc} & t1 & t2 & t3 & t4 & t5 & t6 \\ \hline \text{Text 1} & 0.0 & 1.0 & 8.1 & 4.9 & 4.2 & 3.1 \\ \text{Text 2} & 0.0 & 1.0 & 8.1 & 4.9 & 4.2 & 3.1 \\ \text{Text 3} & 5.0 & 1.0 & 8.1 & 4.9 & 0.0 & 3.1 \\ \text{Text 4} & 5.0 & 1.0 & 0.0 & 0.0 & 4.2 & 0.0 \end{array}$$

Matrix A
(training texts)

$$\begin{array}{c|cccc} & c1 & c2 & c3 & c4 \\ \hline \text{Category set 1} & 0 & 0 & 1 & 1 \\ \text{Category set 2} & 0 & 1 & 1 & 1 \\ \text{Category set 3} & 1 & 0 & 0 & 1 \\ \text{Category set 4} & 1 & 0 & 1 & 0 \end{array}$$

Matrix B
(categories of training texts)

Figure 1: The matrix representation of matched text\category-set pairs.

Option 4: The combination (TFIDF) of the above two, i.e., $a_{ij} = TF_{ij} \times IDF_j$

For example, word $\mathbf{a_{14}}$, which is **Syndrom** in this case, has a value **1** for TF_{ij} weighting schemas since it appears for only once in Text 1, and it has the value **3.1** for the IDF_j weighting term schemas. As a result, the weight of word **Syndrom** in document a =1 × **3.1**. That is listed in matrix A.

Similarly, elements of matrix B are category weights in the corresponding category set. Category weighting is the same as word weighting described above (except that “word is replaced by “category”). The matrices in Figure 1 use TFIDF weights for words, and binary weights for categories.

2.2.2 k-Nearest Neighbor (KNN) Algorithm

KNN classifier is an instance-based learning algorithm that is based on a distance function for pairs of observations, such as the Euclidean distance or Cosine. In this classification paradigm, k nearest neighbors of a training data is computed first. Then the similarities of one sample from testing data to the k nearest neighbors are aggregated according to the class of the neighbors, and the testing sample is assigned to the most similar class. One of the advantages of KNN is that it is well suited for multi-modal classes - where an object can belong to more than on class - as its classification decision is based on a small neighborhood of similar objects (i.e., the major class). So, even if the target class is multi-modal (i.e., consists of objects whose independent variables have different characteristics for different subsets), it can still lead to good accuracy. A major drawback of the similarity measure used in KNN is that it uses all features equally in computing similarities. This can lead to poor similarity measures and classification errors, when only a small subset of the features is useful for classification (Menon, 2004).

2.2.3 Naive Bayesian (NB) Algorithm

NB algorithm has been widely used for document classification, and it has shown to produce a very good performance. The basic idea is to use the joint probabilities of words and categories to estimate the probabilities of categories given a document. This approach is based on the Naïve Bayes Theorem, which can be expressed by Equation (2.3) (Menon, 2004);

$$P(c_j \mid d) = \frac{P(c_j)\, P(d \mid c_j)}{P(d)} \qquad (2.3)$$

Where,

$P(c_j \mid d)$ is the

posterior probability of observing class c_j given document vector, d,

$P(d \mid c_j)$ is the prior probability of observing document vector d given occurrence of class c_j,

$P(c_j)$ is the probability that a randomly picked document belongs to class c_j,

P(d) is the probability that a randomly picked document has vector d as its representation.

The denominator in the above equation does not differentiate between categories and

can be left out. The estimation of $P(d \mid c_j)$ in Equation (2.2) can be quite problematic. Hence in order to alleviate this problem, it is common to make the assumption that any two coordinates of the document vector, when viewed as random variables, are statistically independent of each other. This independence assumption allows us to compute the required probability by using Equation (2.4) (Menon, 2004) as:

$$P(d \mid c_j) = \prod_{i=k}^{m} \mathrm{P}(w_k \mid c_j) \qquad (2.4)$$

Where *m* is the number of distinct words and *w* is the document collection.

NB algorithm computes the posterior probability that the document belongs to different classes and assigns it to the class with the highest posterior probability. The posterior probability of class is computed using Bayes rule and the testing sample is assigned to the class with the highest posterior probability. The naive part of NB algorithm is the assumption of word independence that the conditional probability of a word given a category is assumed to be independent from the conditional probabilities of other words given that category. There are two versions of NB algorithm. One is the multi-variant Bernoulli event model that only takes into account the presence or absence of a particular term, so it does not capture the number of occurrence of each word. Another model is the multinomial model that captures the word frequency information in documents (Menon, 2004).

2.2.4 Decision Tree (C4.5) Algorithm

C4.5 algorithm work based on generating a classifier in a form of decision tree. Decision tree is either a structure that consists of a leaf representing a class or a decision node that specifies some tests to be performed on a singular attribute value with one branch and sub-tree for each possible outcome of the test (Menon, 2004). The decision tree is generated by a systematic choice of which attribute to use as a splitting criterion. At every stage of the tree, the attribute that provides the most information after a split is chosen to be a decision node. This information measure is computed using an attribute-specific quantity called the Gain Ratio Criterion. The Equations below (2.5), (2.6),(2.7) and (2.8) (Menon, 2004) show how it is derived. Entropy, the expected information needed to classify a given document set, *D*, of interest, is given as,

$$Entropy(D) = -\sum_{j=1}^{C} p(D,j) \times \log_2\left(p(D,j)\right) \tag{2.5}$$

where *C* is the total number of classes, j the jth class and p(*D*,j) the proportion of documents in D belonging to the jth class.

The information gain of an attribute is the expected reduction in entropy caused by the partitioning on this attribute and is given as:

$$Information\,Gain(D,T) = Entropy(D) - \sum_{i=1}^{k} \frac{|D_i|}{|D|} \times Entropy(D_i) \tag{2.6}$$

Where *T* refers to an attribute of interest, i refers to the ith value of the attribute *T*, k refers to the total possible values of attribute *T*, |*D*| is the number of documents in *D* and $|D_i|$ is the number of documents in *D* that has the ith value for the attribute *T*

The entropy of the split for a particular attribute *T* is given as:

$$Split(D,T) = -\sum_{i=1}^{d} \frac{|D_i|}{|D|} \times \log_2 \frac{|D_i|}{|D|} \qquad (2.7)$$

Where d is the number of daughter nodes and $|D_i|$ is the number of documents in the ith daughter node. This takes into account the number and size of daughter nodes into which an attribute splits the dataset, disregarding any information about the class. In addition, the gain ratio is given as:

$$Gain\ Ratio = \frac{Gain(D,T)}{Split(D,T)} \qquad (2.8)$$

Gain ratio is used to measure the desirability of an attribute *T* within a dataset *D* at every stage of decision tree build process. Attributes with the highest gain ratio will be targeted for one more split process, the whole process will be repeated at each new branches at this new decision node. The building process at Decision tree will stop when all tests on a sub-dataset have zero gain ratios. However, if this condition was not fulfilled then the process will stop when the classification error within each leaf node is minimized.

Noisy data can lead to over fitting, in this case the classifier generated becomes complex. In addition, generalisability is lost leading to poor accuracy when the model that was built is being tested on unseen data. In order to solve this case, C4.5 attempt to prune the decision tree either by sub-tree replacement, which investigates whether replacing a sub-tree with a leaf node will reduce the error rate, or sub-tree rising, which attempts to raise a decision node one level up the hierarchy of the tree, also with the aim of reducing the error rate. Pruning decision tree aims to reduce the number of misclassified records by reducing the complexity of the generated tree (Menon, 2004).

2.3 Text Categorization Application

Text categorization has been used for a variety of different applications; in here, we review briefly the most important ones.

- **Automatic Indexing for Boolean Information Retrieval Systems**

Each document is assigned with one or more key words or key phrases describing its content, where these key words and key phrases belong to a finite set called controlled dictionary (Sebastiani, 2002) often consisting of a thematic hierarchical thesaurus (e.g.,the NASA thesaurus for the aerospace discipline, or the MESH thesaurus for medicine). Usually, this assignment is done by trained human indexers, and is thus a costly activity.

Automatic indexing with controlled dictionaries is closely related to automated metadata generation. In digital libraries, one is usually interested in tagging documents by metadata that describes them under a variety of aspects (e.g., creation date, document type or format, availability,etc.). Some of this metadata is thematic, that is, its role is to describe the semantics of the document by means of bibliographic codes, key words or key phrases. The generation of this metadata may thus be viewed as a problem of document indexing with controlled dictionary, and thus tackled by means of text categorization techniques (Sebastiani, 2002).

Document Organization

Indexing with a controlled vocabulary is an instance of the general problem of document base organization. In general, many other issues concerning document organization and filing can be addressed by text categorization techniques. For example, at the offices of a newspaper incoming “classified” ads must be, prior to publication, categorized under categories such as Personal, Cars for Sale, Real Estate, etc.

Organizing Documents into categories will facilitate the process of searching. For example, the automatic filing of newspaper articles under the appropriate sections (e.g., Politics, Home News, Lifestyles, etc.), or the automatic grouping of conference papers into sessions (Sebastiani, 2002).

Text Filtering

“Text filtering is the activity of classifying a stream of incoming documents dispatched in an asynchronous way by an information producer to an information consumer (Sebastiani, 2002)”. An example of this application is a news feed, where the producer is a news agency and the consumer is a newspaper.

Filtering can be seen as a case of single-label text categorization, that is, the classification of incoming documents into two disjoint categories, the relevant and the irrelevant (Sebastiani, 2002).

- **Hierarchical Categorization of Web Pages**

Text categorization has many applications to automatically classified Web pages, or sites, under the hierarchical catalogues hosted by popular Internet portals. Providing such a hierarchical categorization of web pages makes it easier for the researcher to navigate in this hierarchy instead of issuing a query terms to search engines.

Classifying Web pages automatically has obvious advantages, since the manual categorization of a large; enough subset of the Web is infeasible.

Automatic Web page categorization has two essential peculiarities:

1. *The hypertextual nature of the documents:* Links are a rich source of information, as they may be understood as stating the relevance of the linked page to the linking page.
2. *The hierarchical structure of the category set:* This may be used, for example, by decomposing the classification problem into a number of smaller classification problems, each corresponding to a branching decision at an internal node (Sebastiani, 2002).

- **Word Sense Disambiguation**

“Word sense disambiguation is the activity of finding- given the occurrence in a text of an ambiguous (i.e., polysemous or homonymous) word- the sense of this particular word occurrence (Sebastiani, 2002)”. For instance, bank may have (at least) two different senses in English, as in the Bank of England (a financial institution) or the bank of river Thames (a hydraulic engineering artifact). It is a word sense disambiguation responsibility to decide which of the above senses the occurrence of bank in "last week I borrowed some money from the bank" has.

Word sense disambiguation is just an example of the more general issue of resolving natural language ambiguities, which is one of the most important problems in computational linguistics.

Word sense disambiguation is very important for many applications, including natural language processing, and indexing documents by word senses rather than by words for IR purposes. Word sense disambiguation may be seen as a text categorization task once we view word occurrence contexts as documents and word senses as categories (Sebastiani, 2002).

2.4 Query Categorization Techniques

There are very good and interesting techniques for query categorization problem solving. These techniques were proposed to (Knowledge Discovery and Data Mining) KDD CUP - 2005 Award (ACM SIGKDD, 2005). However, they differ mostly in the approaches for designing the solution and the learning methods used. Next is a list of the major query categorization techniques adopted by most participants.

2.4.1.Common Approaches for Categorization

Query categorization techniques are different and in general, no two approaches are the same. However, the components that most query categorization approaches mainly consist of are three high level components:

1. Preprocessing.
2. Gathering extra information to augment the query term.
3. Queries modeling.

- **Preprocessing**

There are several big benefits to preprocessing the queries:

- Reduce the impact of noise in queries.
- Reduce the workload on noisy queries.
- Benefit feature extraction.
- Improve accuracy.

To remove an obvious inappropriate content from the raw queries, some preliminary filtering methods need to be performed. In addition, the noise and the versatile nature of queries such as misspelling, foreign language words, and acronyms, must be handled using the advantages of data mining and information retrieval to apply existing text mining preprocessing techniques and create new methodologies for processing search query specifically.

The most commonly used methods are standard text mining techniques like stop words filtering, stemming, and term frequency filtering. Some other approaches uses more advanced techniques, such as spelling correction, compound word breaking, abbreviation expansion, and named entity detection (Li, Zheng and Dai, 2005).

Gathering Extra Information

Search query terms usually contain few words leading to an implicit meaning. Sometime it is hard to learn the meaning of a query solely based on the query itself, that is why many query categorization approaches to find another way of gathering extra information to augment the query term.

Most approaches utilized the knowledge available on World Wide Web and search engines. They used the following resources to build the knowledge base: search engines/Internet, search result snippets - those are quick links to other pages or sub domains within the main site that is listed at number one for the search term bag of words - search engine directories, search result titles, or search result web pages. Some of them also adopted tools like WordNet[٣] to expand queries (Li and etal., 2005).

- **Modeling**

The query categorization approaches submitted to KDD Cup 2005 provided three major dimensions: search query terms, words or phrases, and categories. They used the following two major modeling approaches to building the models based on these dimensions.

[٣] WordNet: http://wordnet.princeton.edu/

Mapping pre-defined/existing directory structure to KDD Cup categories.

2. Constructing the mappings between KDD categories and words (descriptions), and then using the mappings to answer the categories of search query terms that were treated as a bag of words (Li and etal., 2005).

2.4.2 Frequently Used Learning Methods

There are many machine-learning methods adopted in query categorization approaches, either singular methods or a combination of multiple methods together. The most popular used methods are for example; Naïve Bayesian classifiers, (Support Vector Machine) SVM, KNN, Neural Network that is a system of interconnecting neurons in a network working together to produce an output function (Wikipedia), and Logistic Regression - a statistical regression model for binary dependent variables. It can be considered as a generalized linear model that utilizes the logarithm as its link function, and has binomially distributed errors (Wikipedia) - are the popular ones (Li and etal., 2005).

Some query categorization approaches combined multiple methods together in order to achieve a higher performance. To drive an example about that, some query categorization approaches used distance/probability as criteria to combine predictions. Some applied ad- hoc rules in combining predictions. Others adopted some various learning methods like ensemble learning (Zouari, Heutte and Lecourtier, 2005) (e.g. Boosting).

More advanced methods were adopted to tune model parameters, such as using manually tagged examples to tune model parameters or using reward/penalty factors in model tuning/training.

2.4.3 Other Techniques

As we mentioned earlier, a search query cannot express a clear and explicit meaning but it tends to be ambiguous with implicit meaning. To solve this case, simple reprocessing techniques such as stemming and stop word filtering are not sufficient in capturing the meaning of a search query. This problem guides the efforts toward introducing new techniques such as:

- Query / term clustering: Query clustering is a class of techniques aiming at grouping users' semantically related, not syntactically related, and queries in a query repository, which were accumulated with the interactions between users and the system. The driving force of the development of query clustering techniques comes recently from the requirements of modern web searching (Radev, Jing, Stys and etal.,2003).
- Detecting centroid (key) words in a query: A centroid is a set of words that are statistically important to a cluster of documents. As such

- , centroids could be used both to classify relevant documents and to identify salient sentences in a cluster. Detecting centriod words in user query terms form one of the techniques query preprocessing (Wen and Zhang, 2002).
- Soundex for query mapping: Soundex is a phonetic algorithm for indexing names by their sound when pronounced in English. The basic aim is for names with the same pronunciation to be encoded to the same string so that matching can occur despite minor differences in spelling. Soundex algorithm can be applied to map the queries of the same pronunciation (wikipedia).
- Identifying specific groups for queries (e.g. URL, email, trash, and name entities): identifying the queries which belong to the same group such as, URLs, or email addresses can help in detecting the actual meaning of query terms.

- Substring/partial matching: trying to match query terms either totally or partially with other substrings, which have already formed query terms, is used as a query processing technique. Substrings which have already preceded can be used to process new query terms never processed before because of the substrings and queries.

Knowledge representation is also an important aspect. The way queries or other knowledge are represented could dramatically affect the classification performance. Here are a few popular representation techniques adopted in some approaches:

- N-gram (e.g. bi-gram and tri-gram)

An n-gram is a sub-sequence of *n* items from a given sequence, an n-gram of size 2 is a "bigram", size 3 is a "trigram"; and size 4 or more is simply called an "n-gram". They are a commonly used techniques to design kernels that allow machine-learning algorithms such as support vector machines to learn from string data. N-grams can also be used to find likely candidates for the correct spelling of a misspelled word. N-grams are often used in pattern recognition systems in order to assess the probability of a given word sequence appearing in text of the language of interest (Wikipedia).

- Feature-value vector representation

Feature-value vector is a largely used model for describing instance characteristics. It suggests an observation space in which dimensions represent features of the object we want to classify and dimension values are the values of the features as observed in the object. Each instance object is then a point in the feature space, i.e. if the feature space is (F_1,..., F_n) an instance I is:

$$I = (f_1, \ldots, f_n) \tag{2.9}$$

Where each f_i is respectively the value of the feature F_i for I.

Exploiting the feature-value vector model and the related learning algorithms in the context processing natural language may then be a problem especially when the successful bag-of-word abstraction is abandoned for deeper language interpretation models. The prior independence among features, the flatness of the values, and the certainty of the observations are not very well suited for syntactical and semantic models. On the one side, syntactical models would require the possibility of defining relations among features in order to represent either constituents or dependencies among words (Basili, Pazienza and Zanzotto, No year).

- Graphic representation (for word relationship in queries)

Although the Semantic Web is meant to help machines interpret data on the Web, end-users still need to view and query the data. Ideally, also non-experts should be empowered to formulate queries. State of the art retrieval systems for semistructured data such as (Ding, Finin, Joshi and etal., 2004) that are targeted towards end-users still use keyword-based search interfaces. There is a need for graphical interfaces that enable non-expert users to formulate complex queries over semistructured data (Harth, Ryszard and Decker, 2005).

2.5 Evaluation Criteria

Query categorization approaches seek to map search query terms into a meaningful set of related categories to help in either query clustering or to enhance searching process that is widely used by large number of World Wide Web users. That way, there is a need to measure the performance of each approach. To fulfill this, there is a need for some evaluation criteria to use.

Precision Criterionconcerns the percentage of query terms categorization decisions that are correct, it is measured by calculating the rate of the total query terms correctly mapped for category c_i to the total of query terms assigned to category c_i, where c_i is a one category name that represents a subject domain. Equation (2.10) represents the formula used to calculate precision criteria (Li and etal., 2005):

$$\text{Precision} = \frac{\sum_i \# \textit{ of queries are correctly tagged as } c_i}{\sum_i \# \textit{ of queries are tagged as } c_i} \qquad (2.10)$$

Recall Criterion concerns the percentage of query terms categorization decisions that are actually taken, it is measured by calculating the rate of the total query terms correctly mapped for category c_i to the total of query terms manually assigned to category c_i. Equation (2.11) represents the formula used to calculate precision criteria (Li and etal., 2005):

$$\text{Recall} = \frac{\sum_i \# \textit{ of queries are correctly tagged as } c_i}{\sum_i \# \textit{ of queries whose category is labeled as } c_i \textit{ by human labeler}} \qquad (2.11)$$

F1 Criterion is the most known measurement that combine between precision and recall and it is defined in Equation (2.12) (Li and etal., 2005):

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \qquad (2.12)$$

CHAPTER 3

DISCUSSION OF THE CASE STUDY

This chapter starts by introducing the case study the thesis utilized for results comparison. In addition, it is discussing the techniques used during the selected case study, the set of constraints which took place during the study experiment, the applied text categorization techniques the researchers used, and finally the results of the proposed query categorization technique introduced by the case study.

3.1 The Comparable Case Study

Chuang and Chien had conducted a study under the title "Automatic Query Taxonomy Generation for Information Retrieval Application" (Chuang and Chien, 2003 (b)). They proposed an approach that consists of two computational processes: "Hierarchical query clustering to generate query taxonomy from scratch, and query categorization to place new-arrived queries into taxonomy" (Chuang and Chien, 2003 (b)).

In this Study, the second component of the case study approach had been used, that is query categorization, to compare it with the thesis proposed approach.

Chuang and Chien approach based on the mining of search results returned from the real world Web search engines to construct the taxonomies for queries. Figure 2 shows an abstract diagram of the proposed approach that is mainly composed of three computational modules: Context extraction, Query clustering and Query categorization (Chuang and Chien, 2003 (b)).

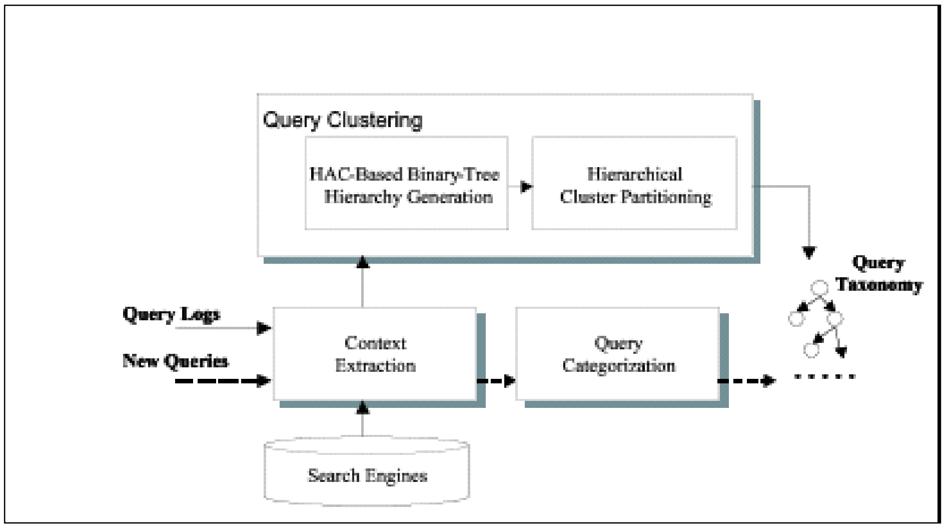


Figure 2: An abstract diagram showing the concept of the proposed approach.

A Query clustering module organizes queries of different forms such as single term, Boolean expression and natural sentences. In order to cluster queries of the same information needs, the proposed approach uses the highly ranked search-result snippets retrieved by search engines as a descriptor features for queries. The Middle ware module is context extraction; it was developed to perform the task of exploitations of queries features. Query clustering module uses a proposed clustering technique called HAC+P (Chuang and Chien, 2003 (b)), which is an extension of the Hierarchical Agglomerative Clustering (Beeferman and Berger, 2000) algorithm (HAC). It has been designed to generate a natural and comprehensive multi-way tree cluster hierarchy by combining it to a cluster partition technique. On the other hand, the query categorization module was developed to categorize any new-arrived query in order to enrich the existing taxonomy. The next section will concentrate on describing the query categorization module that was used to compare its result with the proposed approach using the same data set.

3.2 Case Study Constraints

- Query pattern is much shorter and usually contains insufficient feature information in itself for judging its relevance with other target instances in an automatic categorization task. The contexts of a query are obtained from highly ranked search-result snippets returned by Web search engines or IR systems, e.g., the titles and descriptions of search-result entries, and texts surrounding matched terms. The features for a query are then extracted from the returned snippets (Chuang and Chien, 2003 (b)). On the other hand, this thesis attempts to determine the query's context information by Introducing descriptor files for the intended mapping category names, those files aim to

- cover the most context information for each category name in order to capture the sufficient meaning for query terms.

- Both the case study and this thesis had adopted the vector-space model for data representation. For each query *q*, each query can then be converted into a bag of feature terms by applying normal text processing techniques. Let T be the feature term vocabulary, and let t_i be the ith term in T. With simple processing, a query *q* can be represented as a term vector v_q in a T-dimensional space, where $v_{q,i}$ is the weight t_i in v_q. The term weights in this work are determined according to one of the conventional tf-idf term weighting schemes, in which each term weight $v_{q,i}$ is defined in Equation (3.1):

$$\mathbf{v_{q,i} = (1 + \log_2 f_{q,i}) \times \log_2(n / n_i)} \qquad (3.1)$$

where $f_{q,i}$ - The frequency t_i occurring in $v_{q's}$ corresponding feature term bag.

n - The total number of queries.

n_i - The number of queries that contain t_i in their corresponding bags of feature terms.

The similarity between a pair of queries is computed as the cosine of the angle between the corresponding Vectors as it is in Equation (3.2) (Chuang and Chien, 2003 (b)):

$$\mathbf{Sim\ (v_a\ ,\ v_b) = Cos\ (v_a\ ,\ v_b)} \qquad (3.2)$$

$$\mathbf{Cos(v_a,v_b) = \frac{v_a \times v_b}{|v_a|\times|v_b|}}$$

3.3 Query Categorization Module

As mentioned before, query categorization modules proposed in the study of Chung and Chien (Chuang and Chien, 2003 (b)) study aimed to place any new-arrived queries into the existent taxonomy; in summary, this will lead to reduce the heavy load of reconstructing the whole taxonomy. Given a new query *q*, query categorization is to determine a set of clusters that are considered as *q*'s related clusters. This candidate query *q* is represented as a feature vector v_q. To categorize this query two approaches were proposed by this study.

Mean. "Each cluster C_i is represented as its centroid c_i whose jth feature weight is defined as the average of all its contained instances' jth feature weights. If a cluster is an interior cluster, its contained instances are those instances contained in itself and its composed clusters. The relevance score between v_q and C_i is defined by the Equation (3.3)" (Chuang and Chien, 2003 (b)).

$$r_{Mean}(q, C_i) = sim(v_q, c_i) \qquad (3.3)$$

k-Nearest Neighbor (kNN). “kNN has been an effective classification approach to a broad range of pattern recognition and text classification problem (Dasarathy.1991). In the kNN approach, the relevance between v_q and the candidate cluster C_i is defined by the Equation (3.4)” (Chuang and Chien, 2003 (b)).

$$r_{kNN}(q, C_i) = \sum_{vj \in Rk(q) \cap Ci} sim(v_q, v_j) \qquad (3.4)$$

Where Rk(q) are q’s k most-similar instances, measured by *sim* function, in the whole collection.

3.4 Experiments and Results

This section represents the conducted the experiment hold by Chung and Chien (Chuang and Chien, 2003 (b)) in term of the used data set and the results the had achieved. The section starts by describing the data set that also had been used by this thesis. Then, it represents the results of the proposed query categorization approach introduced in Chung and Chien study.

3.4.1 Experiment Data Set

To have a standard basis for performance evaluation, we used the Yahoo! Directory as our major experimental data set (Chuang and Chien, 2003 (b)), which is the same set used by Chuang and Chien (Chuang and Chien, 2003 (b)) in their study. The category names in the top three levels of Yahoo! Computer Science directory were collected. There were 36, 177, and 278 category names in the first, second, and the third levels, respectively. These category names were short in length and specifically expressed some subject domain requests; therefore, they could play the role of queries that users might submit. Some category names could be placed in multiple categories; i.e. they have multi-class information.

The study used the first and the second-level category names as the training instances with the first-level categories of Yahoo!'s CS directory as the target classes. The third-level category names were used for testing. Table 1 represents the first-level category names used as the main categories to collect a descriptor file for each one of them. On the other hand, Table 2 represents the second and the third level of category names in Yahoo! Directory to use them as a query terms to be mapped to the first level of Yahoo! Directory categories.

Table 1: **First-level category names in Yahoo! directory**

CompINET:CS	CompINET:CS
Algorithms	Library_and_Information_Science
Architecture	Linguistics
Artificial_Intelligence	Logic_Programming
Compression	Mobile_Computing
Computational_Learning_Theory_(COLT)	Modeling
Computational_Sciences	Networks
Computer_Vision	Neural_Networks
Databases	Object-Oriented_Programming
Distributed_Computing	Operating_Systems
DNA-Based_Computing	Quantum_Computing
Electronic_Computer_Aided_Design_(ECAD)	Real-Time_Computing
End_User_Programming	Robotics
Finite_Model_Theory	Security_and_Encryption
Formal_Methods	Software_Engineering
Graphics	Supercomputing_and_Parallel_Computing
Handwriting_Recognition	Symbolic_Computation
Human-Computer_Interaction_(HCI)	User_Interface
Knowledge_Sciences	Virtual_Reality

Table 2: Selected sample for the second-level category names used as training set for the first-level category names

Query Term	Match Category
LessTif	CS: Operating_Systems
Linux	CS: Operating_Systems
Machinima	CS: Graphics
Mathematics	CS: Logic_Programming
Mirages	CS: Computer_Vision
Miva Script	CS: End_User_Programming
Mnemonics	CS: Linguistics
Quantum Computing	CS: Quantum_Computing
Quantum Cryptography	CS: Quantum_Computing CS: Security_and_Encryption

3.4.2 Results Discussion

kNN and Mean similarity methods have been applied to the first level of category names which represents the target labels, and the second - third level labels which represent the query terms. The result of each method is summarized as shown in Table 3 that lists the result of top 1-5 inclusion rates, where top *n* inclusion rate is defined as "the rate of instances whose highly ranked *n* candidate classes contain the correct class(es) (Chuang and Chien, 2003 (b))".

Table 3: Top 1-5 inclusion rates for query categorization in Chung and Chien study

Yahoo!	Top-1	2	3	4	5
Mean	.7219	.8609	.8940	.9073	.9305
kNN	.7185	.8841	.9238	.9437	.9636

The table shows that top-*n* inclusion rate increases as long as the value of *n* increases and that after all ended up with a good performance of the proposed query categorization approach. The good performance can be returned to the usage of Web search results returned by real-world search engines as the information source for the target queries. This gives several advantages: First, huge amounts of context information to characterize a query. Second, the Web, whose content is always refreshing and enlarging, offers the most up-to-date information. Third, there is no need to prepare the information of queries laboriously; existing search engines reduce this load.

CHAPTER 4

DISSCUSION OF THE PROPOSED APPROACH

4.1 Background of the Proposed Approach

Most of the proposed studies about query categorization problems work in three different tracks; each of these tracks is considered a field of research and study. As mentioned in the previous chapter, most query categorization approaches consist of three main components; preprocessing, gathering information to augment queries, and on modeling.

Studies had been conducted by many researchers, which mainly concentrated on one component than another and had proposed solutions for each component at a time.

This thesis concentrates on proposing a new way for gathering extra information to reveal the ambiguity combined with a user search query. The approach proposes collecting a set of descriptor files that describe categories' names, the descriptor files are gathered from the knowledge available on World Wide Web resources, and have mainly been collected from Wikipedia encyclopedia that contains sets of scientific articles. An example of these collected files is provided in Appendix A.

The proposed query categorization approach is based on reducing the ambiguity of a user search query that usually contains few words .There are many proposed approaches to categorize a query work in the same area of interest. This work attempts to categorize any search query into a set of categories by trying to enhance the meaning of the search query. The meaning of a search query can be enhanced by either query expansion or query reformulation. This work attempts to enhance the query meaning by proposing a descriptor file for each category name. Each category represents a subject domain (i.e. algorithm, entertainment, etc) that can be defined and described from different aspects. For example, the descriptor file may contain the definition of the subject domain that represents the category name, popular keywords in that subject, algorithms related to that category, famous people working in that field of science, and much more aspects that reflect the uniqueness of each category.

In contrast to the case study discussed above, the descriptor files have been collected manually without any updating mechanism, or any indexing controllers, but its coverage is more accurate than the result returned by search engines used in Chuang and Chien (Chuang and Chien, 2003 (b)) study to reformulate the query terms.

The collection of the descriptor files are treated as a document set, where each file within is represented by a vector Vc_i that describes category c_i. The descriptor files are collected using the knowledge available on the World Wide Web resources; for our work, we collected the information about each category c_i from Wikipedia encyclopedia that contains a set of scientific articles for every field of science. Using this resource to construct our descriptor document set helped us to increase query terms meaning by mapping user query search *q* to one or more of the best match documents within the set using one of text classification algorithms.

To categorize any user search query, it is represented as a feature vector v_q. We used the vector space model approach to map it to the best match category/ies. The relevance score between v_q and Vc_i is given by Equation (4.1) where similarity between vectors is calculated as follow:

$$\mathbf{r}\ (q, \mathbf{V}c_i) = sim\ (q, \mathbf{V}c_i) \qquad (4.1)$$

$$sim(q, \mathbf{V}c_i) = \frac{q \times \mathbf{V}c_i}{|q| \times |\mathbf{V}c_i|}$$

4.2 Experiments and Results

To assess the performance of the proposed query categorization approach, two types of experiments have been conducted. The first one is to measure the top *n* inclusion rate for the proposed approach using the same data set Chung and Chien (Chuang and Chien, 2003 (b)) had used. Another experiment conducted to assess the performance of the simulator in terms of execution time was based upon two criteria. Firstly, the length of query terms and. Secondly, in terms of *n* value in top *n* inclusion rate.

The first experiment used Space Vector Model to measure the similarity between a given query vector V_q and the document set. The return result would be the top five ranked categories which mapped the query vector V_q.

In order to conduct this experiment, a simulator had been manipulated according to the algorithm explained in Figure 3 that describes the procedure followed to describe the categorization process.

QueryCategorise (*q, C, F, n*)
q: the unknown query term
C: the set of category labels
F: the set of descriptor files
Vc_i: vector represent category i
n: the number of target categories
DB: database
a_{ij}: term *i* in document *j*

```
1.  for all $f \in F$ do
2.    for all $a_{ij} \in f$ do
3.      $a_{ij} \rightarrow$ DB
4.    end for
5.  end for
6.  for all $a_{ij} \in$ DB
7.    if ($a_{ij}$) is a stop word
8.        delete $a_{ij}$
9.    else
10.           remove_special_characters ($a_{ij}$)
11.           $a_{ij} \rightarrow Vc_i$
```

```
12.
13.       end if
14.     end for
15.     for all c Є  C do
16.       for all $a_i$ Є $Vc_i$
17.         calculate TF, IDF weight schemas
18.          TFIDF = TF × IDF
19.      end for
20.     for all $a_i$ Є q do
21.   $a_i$ → DB
22.   remove_special_characters  ($a_i$)
23.  calculate TF,IDF,TFIDF
24. end for
25. for all c Є  C do
26.  sim (q, $Vc_i$) = q × $Vc_i$  /  | q | × |$Vc_i$|
27. end for
28. return top-ranked n categories in C
```

Figure 3: An algorithm procedure describes the categorization process.

The next driving example will clarify the use of the explained algorithm to return the target top ranked n categories among a list of category names represented as descriptor files.

Example:

To categorize the query terms **(Universal Mobile Telecommunications System (UMTS)**) , which is a category in the second level of Yahoo! Directory used as a query terms to map it to the top *n* ranked categories among those existing in the first level of Yahoo! Directory. The simulator converts the query terms into a vector by calculating tf-idf value for every term in query terms. The similarity between the query vector and each file in the set of descriptor files is calculated according to Equation (4.1). According

Table 4: An example of the top five category names mapped to a query vector

to this example the simulator returns the following top 5 categories listed below in Table 4.

Query Terms	Top 5 ranked categories
Universal Mobile Telecommunications System (UMTS)	Mobile_Computing Modeling Linguistics Virtual_Reality Operating_Systems

The simulator is a software application based on Space Vector Model method. For the purpose of implementation, we elected VB.net as an implementation language. Our choice for VB.net was mainly for two main reasons; the first one is providing the user with a friendly user interface in order to facilitate the application and help to display the result in an organized way. The second is for the limitation of Random Access Memory (Ram) usage faced by using any Consol Application project; this will lead to many problems during the test part of the conducted experiment. Moreover, there is another indirect supportive reason behind that choice, which is the ability to use any suitable DataBase Management System (DBMS) to store the vectors built for every category name. This facility helps avoiding the usage of large amounts of Ram and gives an advantage of using this application on any Personal Computer with a reasonable amount of memory.

After conducting the experiment and calculating the top *n* inclusion rate, we reached the following results. Table 5 lists the top 1-5 inclusion rate fulfilled by the proposed approach of this thesis.

Table 5: Top 1-5 inclusion rates for query categorization in our proposed approach

Yahoo!	**Top-1**	**2**	**3**	**4**	**5**
Space Vector Model	0.4976	0.6493	0.6825	0.7251	0.7583

The result shows that top *n* inclusion rate increases as the value of *n* increases. However, we can see the performance of our proposed approach is poorer than Chuang and Chien study. The difference in the obtained results is due to two main reasons. Firstly, the way each approach reformulate its query terms. Chunag and Chien study used the Web search engines results as the information source for the target queries. This technique has an advantage of automatic up-to date mechanism, which is absent in the proposed approach of this thesis. However, this work used descriptor files to category names as a source of information for query terms. The coverage of these files has a deep impact on the obtained results. The more it describes the category the more it is mapped correctly. Secondly, the applied text categorization algorithm may affect the results, Chung and Chien got a good performance using kNN algorithm while this thesis applied a similarity measurement using space vector model method.

Figure 4 represents the results in a line graph to view the degree of accuracy of the proposed categorization technique

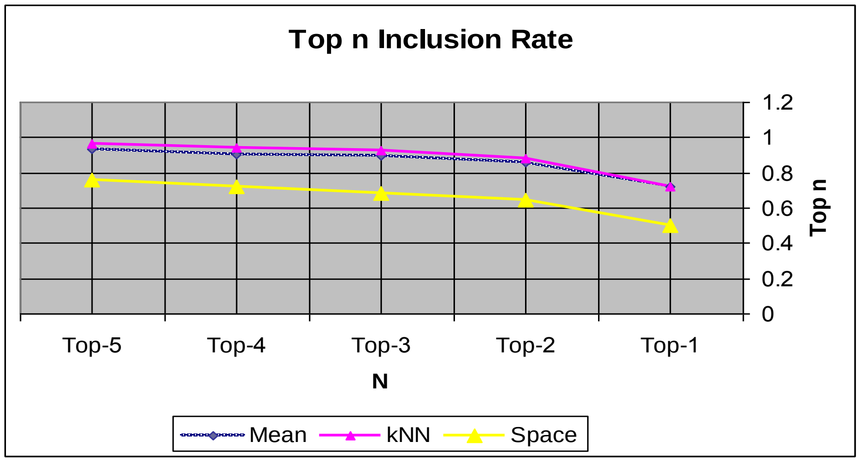


Figure 4: Line Graph representing the results of the proposed approach compared to the results of the case study

In order to measure the performance of the proposed approach in terms of execution time, two different experiments have been conducted. The first one was aimed at calculating the average execution time per top *n* inclusion rate. The experiment used a random sample of five query terms of the same length. Each set of query terms was tested using the simulator for five times ,where each time the simulator returned different value of top *n* ranked categories ranging from 1- 5. Each time a set of query terms was tested, the execution time was measured using a built-in method in VB.net libraries designed to measure the time spent by a program to finish its execution. The average of execution time was measured for all query terms each time the trial was conducted for a different value of top *n*.

Figure 5 summarizes the relation between the average execution time and the top *n* inclusion rate.

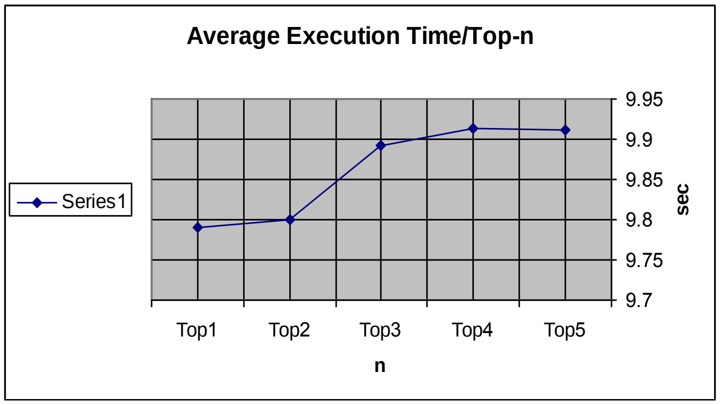


Figure 5: Line Graph representing the relation between average execution time / top-n inclusion

The Figure shows that the change of *n* value does not have a radical effect on the execution time which leads to the conclusion that the proposed approach is stable in terms of top *n* inclusion rate.

The second experiment used other five different query terms of various lengths ranging from one word to five in order to figure out the relationship between the average execution time and the query terms length. The experiment was based on selecting a clustered sample of query terms where each cluster consisted of five different queries of the same length. Table 6 represents the selected query terms of different lengths used in this experiment.

Table 6: Clustered sample of query terms of different lengths used to measure the average execution time

Query Length	Query terms
1	Solaris ,Wireless ,Telecommuting, Telecommunications ,Robots
2	Specific Languages , Web Bugs, Windows 2000,Virtual Reality , Virtual Pets
3	Web Site Templates ,Web Site Donations,Web Site Templates,Privacy Seal Programs ,Online Virus Scanners
4	Virus Protection Software Reviews ,Open Shortest Path First (OSPF),NTP (Network Time Protocol), (MUDs , MUSHes, MOOs), POP (Post Office Protocol)
5	VRML (Virtual Reality Modeling Language) ,NNTP (Network News Transfer Protocol),SMTP (Simple Mail Transfer Protocol) ,XFML (eXchangeable Faceted Metadata Language) ,VRML (Virtual Reality Modeling Language)

Each one of these query terms was tested using the implemented simulator; the execution time of the simulator was measured and the average execution time was calculated for each cluster within the sample .The result of the conducted experiment is presented in Figure (6) below.

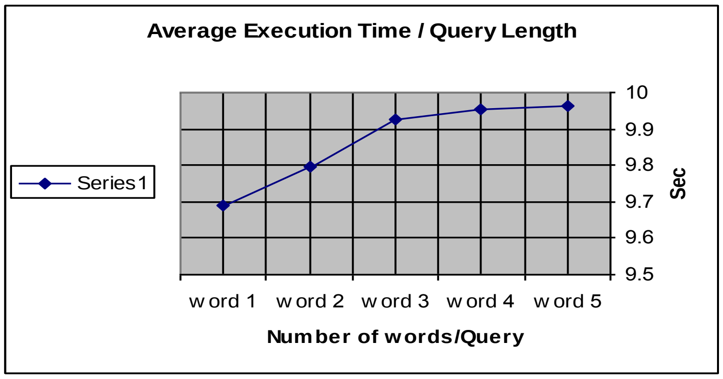


Figure 6: Line Graph representing the relation between average execution time / query terms length

Figure 6 shows the observed relationship in a line graph. It has been concluded that the average execution time increases as the query terms length increases. However, the percentage of the increase is relatively small. Furthermore, since all query terms have a small number of words, we can claim that query terms length does not have a great effect on the performance of the proposed approach in terms of execution time.

As mentioned earlier in chapter two, most query categorization approaches used evaluation criteria to measure their accuracy. Precision, recall and F1 criteria are used to perform this task. To assess the accuracy of our proposed approach we have applied these criteria to the obtained results of top-*n* inclusion rate experiment. Table 7 represents the result of precision, recall and F1 criteria for the top-5 inclusion rate.

Table 7: Summary of Precision, recall and F1 criteria results for Top-*5* inclusion rate

Yahoo!	**Top-5**
Precision	0.1603
Recall	0.535117
F1	0.246758

The table shows that we have a poor performance in terms of precision. This thesis proposed a new way of supporting source information to increase query terms meaning by building descriptor files for each category names. As mentioned earlier, the documents set of files is built manually and it has no controlling index with search engines results as it is in Chuang and Chien study (Chuang and Chien, 2003 (b)), which in turn provides updating for the search results used as a source of information for query terms. Based on these constraints, the precision of the proposed approach was poor, as the coverage of the files is not enough to retrieve all related files for a given query term. On the other hand, 50 % of the query terms was mapped leading to an acceptable recall performance but still need to be enhanced.

4.3 The Pros and Cons of the Proposed Approach

This approach simply requires a personal computer that has a reasonable amount of RAM, about 250 MB. However, the process of collecting and analyzing results consumes much more time, since we have to calculate more top *n* inclusion rate for almost 299 query terms. The proposed approach has some pros and cons that can be summarized as follows:

- **Pros:**
  - The proposed approach adds a new assistance in enhancing the searching process over the World Wide Web by suggesting a list of possible category names that may reveal potential ambiguity in the search term. Mapping query terms to a list of category names helps the user to minimize the time he/she usually spends doing a search.

- Descriptors Files play a significant role in providing additional information about category names in order to match query terms with their possible categories.

Cons:

- It has a slow speed in case of large collection of document sets. Actually, this might be due to the nature of the classification methods, which require a lot of calculating tasks.
- The proposed approach does not have a mechanism for updating descriptor files contents.

CHAPTER 5

CONCLUSIONS AND POSSIBLE WORKS

This chapter summarizes the query categorization approach that has been implemented using Space Vector Model similarity concept method. The conclusion as well as some possible works are suggested.

5.1 Conclusions

This research has proposed a new approach to map query terms into a set of category names. This approach has the advantage of enriching the widely used process of searching over the web and it can be of benefit in the area of query clustering problems.

The experimental results have shown the feasibility of our approach. Using descriptor files for each category name relying on the knowledge available on the World Wide Web has a positive effect in the case of gathering extra information to reveal the implicitly of search query terms. It has been proven that the top *n* inclusion rate increases as the value of *n* increases. Although, the results show that the proposed approach achieved a reasonable top *n* inclusion rate, nevertheless they were not as high as the reported results of the approach proposed in the case study. This variation is due to many reasons. First, the by the accuracy and coverage of the descriptor files that have been built to describe each category name, have a well defined descriptor file for each category name which leads to a better mapping of query terms and category names. Also, the kNN and Mean similarity methods used by the case study were worked over clusters of queries, which extracted their features from the retrieved highly ranked search-result snippets for each query. However, this approach worked directly on the descriptor files to extract the features of each query and try to map it to the target category/ies. In addition, the absence of up-to date mechanism for descriptor files content will affect the performance of the proposed approach since the content of these files is the major factor the proposed approach relies on.

By assessing the performance of the proposed approach, we observe that the precision rate is poor. This is in part due to the limitations imposed by the adopted classification method and the accuracy of the descriptor files. On the other hand, we have achieved a reasonable recall percentage (50%) of the query terms that have been mapped.

To sum up, the proposed approach demonstrated a good performance in case of top *n* inclusion rate but still needs to be enhanced. The approach mapped 50% of the query terms, which is a reasonable recall percentage but, in terms of precision, the percent of correctly mapped query terms is low which leads to weak performance in terms of precision criterion.

5.2 Possible Future Works

From the research undertaken, many questions and possible extensions have emerged.

These are outlined as follows:

- It was shown that the accuracy of results depends in the first place on the proposed descriptor set that was built to describe the category names. One way to overcome this problem is either by carefully selecting the training set examples, so that they better reflect the descriptor set, or trying to work on a structuring mechanism for the descriptors files. Rather than relying on random information that may differ from one file to another, putting a structure for these descriptor files will enhance the quality of the information they contain and thus lead to better results.
- Linking the proposed approach with an updating mechanism to keep an up-to-date version of these file may enhance the coverage of these

- files and thus lead to a better precision percentage.
- The adopted classification method had a great impact on the overall results in term of precision and recall. To solve this issue we recommend the adoption of other similarity measurements to map the results. Such approach implies the use of Naive Bayesian which utilizes the joint probabilities of words and categories to estimate the probabilities of relevant categories. This can lead to better results.

REFERENCES

ACM SIGKDD 2005. http://www.acm.org/sigs/sigkdd/kdd2005

Agrawal, R. and Srikant, R ,2001. On Integrating Catalogs. "In proceedings of the conference on the World Wide Web."

Anick, P. G. and Tipirneni, S.(1999). The paraphrase search assistant: Terminological feedback for interactive information seeking . "In Proceedings of the 22th ACM International Conference on Research and Development in Information Retrieval (SIGIR'99)."

Avancini, H., Rauber, A. and Sebastiani, F. (2002). Organizing Digital Libraries by Automated Text Categorization (on line).Available: http://info.acm.org/class/1998/ccs98.html

Basili, R., Pazienza, M.P and, Zanzotto, F.M.(no year). Exploiting the feature vector model for learning linguistic representations of relational concepts (on line).Available: http://www.dcs.shef.ac.uk/~fabio/ATEM03/basili-ecml03-atem.pdf

Beeferman, D. and Berger, A. (2000). Agglomerative clustering of a search engine query log. "In Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining."

Bruza, P.D and Dennis, S. (1997). Query reformulation on the internet: Empirical data and the hyperindex search engine." In Proceedings of the

Conference on Computer-Assisted Information Searching on Internet."

Bruza, P.D, McArthur, R. and Dennis, S. (2000). Interactive internet search: keyword, directory and query reformulation mechanisms compared. "In Proceedings of the 23rd ACM International Conference on Research and Development in Information Retrieval (SIGIR'2000)."

Chien, L.F.(1999). Pat-tree-based adaptive keyphrase extraction for intelligent Chinese information retrieval. Information Processing and Management Volume: 35, pp 501-521.

A- Chuang, S.L. and Chien, L.F. (2003). Enriching Web taxonomies through subject categorization of query terms from search engine logs. Decision Support Systems, Volume: 35, Issue 1, April 2003, pp. 113-127.

B- Chuang, S.L. and Chien, L.F. (2003). Automatic Query Taxonomy Generation for Information Retrieval Applications. MCB UP Ltd , Volume: 27 Issue: 4, pp. 243 – 255

Deerwester, S., Dumais, S.T., Furnas, G. W. and etal. (1990) . Indexing by latent semantic analysis. Journal of the American Society for Infor-mation Science, Volume: 41 (6) (1990), pp. 391 - 407.

Dennis, S. , Bruza, P. and McArthur, R. (2002). Web searching: a process-oriented experimental study of three interactive search paradigms. American Society for Information Science and Technology, Volume: 53. pp, 120–133.

Ding, L., Finin, T.,Joshi, A. and etal .(2004). Swoogle: A search and metadata engine for the semantic web. "In proceedings of the Thirteenth ACM Conference on Inf. and Knowledge Mgmt."

Gonzalo, J., Verdejo, F., Chugur, I and etal. (1998). Indexing with wordnet synsets can improve text retrieval. "In Proceedings of the Annual Meeting of the Association for Computational Linguistics and International Conference on Computational Linguistics."

Gravano, L., Hatzivassiloglou, V. and Lichtenstien, R.(2003). Categorizing Web Queries According to Geographical Locality. ACM 1-58113-723-0/03/0011,pp 325-333.

Harth, A., Ryszard, S. R. and Decker, S. (2005). Graphical Representation of RDF Queries (on line). Availabe: http://www2006.org/programme/files/pdf/p149.pdf

Li, Y., Zheng, Z. and Dai, H.(2005). KDD CUP-2005 Report: Facing a Great Challenge. SIGKDD Explorations, Volume 7, Issue 2, pp. 91-99.

Mandala, R. Tokunaga, T. and Tanaka, H. (1999). Combining multiple evidence from different types of thesaurus for query expansion. "In Proceedings of the 22nd ACM International Conference on Research and Development in Information Retrieval (SIGIR'99)."

Menon, R. (2004). mining of textual databases within the product development process. Published doctoral dissertation, University of Singapore.

Radev, D.R., Jing, H. , Stys, M., and etal (2003). Centroid-based summarization of multiple documents. Unpublished doctoral dissertation, University of Michigan, Ann Arbor.

Riloff, E. and Shepherd, J. (1997). A corpus-based approach for building semantic lexicons." In Proceedings of the Second Conference on Empirical Methods in Natural Language Processing."

Sanderson, M., Croft, B. (1999). Deriving concept hierarchies from text. "In Proceedings of the 22nd ACM International Conference on Research and Development in Information Retrieval (SIGIR'99)."

Sebastiani, F. (2002). Machine Learning in Automated Text Categorization. ACM Computing Surveys, Vol. 34, No. 1, pp. 1–47.

Schutze, H. and Pederson, J.O. (1994). Information retrieval based on word senses. "In Proceedings of the Annual Symposium on Document Analysis and Information Retrieval."

Stokoe, C., Oakes, P. O. and Tait, J. (2003).Word sense disambiguation in information retrieval revisited. In Proceedings of the International ACM SIG Conference on Research and Development in Information Retrieval.

Vogal, D., Bickel, S. and Haider, P. (2005). Classifying Search Engine Queries Using the Web as Background Knowledge, SIGKDD Explorations, Volume: 7, Issue 2, pp. 117-122.

Wen, J. and Zhang, H. (2002). Query clustering in the Web Context (on line).Available:

http://research.microsoft.com/~jrwen/jrwen_files/publications/QC-CIR.pdf

Wikipedia. (free encyclopiedia) Available:http://www.wikipedia.org/

Wordnet. (free encyclopiedia) Available: http://wordnet.princeton.edu/

Yang, Y. and Chute, C.G. (1994). An Example-based Mapping Method for Text Categorization and Retrieval. ACM Transactions on Information Systems, Volume: 12, No 3, pp. 252-277

Zoran, M. (2003). Techniques for query reformulation, query merging, and information reconciliation (on line). Available: http://www.cs.umd.edu

Zouari, H., Heutte, L. and Lecourtier, Y. (2005). Controlling the diversity in classifier ensembles through a measure of agreement. Pattern Recognition, Volume: 38, pp. 2195-2199 .

Appendices

APPENDIX A

Category name: Visual programming language

Visual programming language (VPL) is any programming language that lets users specify programs in a two-(or more)-dimensional way. Conventional textual languages are not considered two-dimensional
since the compiler or interpreter processes them as one-dimensional streams of characters. A VPL
allows programming with visual expressions, spatial arrangements of text and graphic symbols. Most
VPLs are based on the idea of "boxes and arrows," that is, boxes or circles or bubbles, treated as
screen objects, connected by arrows, lines or arcs. Non Visual Programming languages are completely based on text, when using non VPL applications it is impossible to see images until the program is run. This is a disadvantage to programmers as the user is unable to see the results of their coding unless the program is run.

VPLs may be further classified, according to the type and extent of visual expression used, into icon-based languages, form-based languages, and diagram languages. Visual programming environments
provide graphical or iconic elements which can be manipulated by users in an interactive way according to some specific spatial grammar for program construction.

A visually transformed language is a non-visual language with a superimposed visual representation.
Naturally visual languages have an inherent visual expression for which there is no obvious textual equivalent.

Current developments try to integrate the visual programming approach with dataflow languages to
either have immediate access to the program state resulting in online debugging (i.e. LabVIEW) or
automatic program generation and documentation (i.e. visual paradigm). Dataflow languages also allow automatic parallelisation, which is likely to become one of the greatest programming challenges of the future.

Known keywords:
Visual Knowledge Representation Languages
KRS
Conceptual Graphs
declaritive compositional modeling framework
KSM knowledge modeling system
Hypertropes
Informal concept
CoMo-Kit