Graduate School ETD Form 9
(Revised 12/07)

PURDUE UNIVERSITY
GRADUATE SCHOOL
Thesis/Dissertation Acceptance

This is to certify that the thesis/dissertation prepared

By Xiaojing Gong

Entitled

QUERY SEGMENTATION FOR E-COMMERCE SITES

For the degree of Master of Science

Is approved by the final examining committee:

Dr. Mohammed Al Hasan
Chair

Dr. Shiaofen Fang

Dr. Rajeev Raje

To the best of my knowledge and as understood by the student in the *Research Integrity and Copyright Disclaimer (Graduate School Form 20)*, this thesis/dissertation adheres to the provisions of Purdue University's "Policy on Integrity in Research" and the use of copyrighted material.

Approved by Major Professor(s): Dr. Mohammed Al Hasan

Approved by: Dr. Shiaofen Fang 07/12/2012
Head of the Graduate Program Date

Graduate School Form 20
(Revised 9/10)

PURDUE UNIVERSITY
GRADUATE SCHOOL

Research Integrity and Copyright Disclaimer

Title of Thesis/Dissertation:

QUERY SEGMENTATION FOR E-COMMERCE SITES

For the degree of Master of Science

I certify that in the preparation of this thesis, I have observed the provisions of *Purdue University Executive Memorandum No. C-22,* September 6, 1991, *Policy on Integrity in Research.**

Further, I certify that this work is free of plagiarism and all materials appearing in this thesis/dissertation have been properly quoted and attributed.

I certify that all copyrighted material incorporated into this thesis/dissertation is in compliance with the United States' copyright law and that I have received written permission from the copyright owners for my use of their work, which is beyond the scope of the law. I agree to indemnify and save harmless Purdue University from any and all claims that may be asserted or that may arise from any copyright violation.

Xiaojing Gong

Printed Name and Signature of Candidate

07/12/2012

Date (month/day/year)

*Located at http://www.purdue.edu/policies/pages/teach_res_outreach/c_22.html

QUERY SEGMENTATION FOR E-COMMERCE SITES

A Thesis

Submitted to the Faculty

of

Purdue University

by

Xiaojing Gong

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science

August 2012

Purdue University

Indianapolis, Indiana

This work is dedicated to my family and friends.

ACKNOWLEDGMENTS

I am heartily thankful to my supervisor, Dr. Mohammed Al Hasan, whose encouragement, guidance and support from the initial to the final level enabled me to develop an understanding of the subject.

I also want to thank Dr. Shiaofen Fang and Dr. Rajeev Raje for agreeing to be a part of my Thesis Committee.

Thank you to all my friends and well-wishers for their good wishes and support. And most importantly, I would like to thank my family for their unconditional love and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Gong, Xiaojing. M.S., Purdue University, August 2012. Query Segmentation For E-Commerce Sites. Major Professor: Dr. Mohammad Al Hasan.

Query segmentation module is an integral part of Natural Language Processing which analyzes users' query and divides them into separate phrases. Published works on the query segmentation focus on the web search using Google n-gram frequencies corpus or text retrieval from relational databases. However, this module is also useful in the domain of E-Commerce for product search. In this thesis, we will discuss query segmentation in the context of the E-Commerce area. We propose a hybrid unsupervised segmentation methodology which is based on prefix tree, mutual information and relative frequency count to compute the score of query pairs and involve Wikipedia for new words recognition. Furthermore, we use two unique E-Commerce evaluation methods to quantify the accuracy of our query segmentation method.

1. INTRODUCTION

The researchers have observed a widespread trend that the Internet search engine users increasingly use natural language text for retrieving meaningful results from web documents or online databases [1]. Although this requires the search engine to work harder for finding the desired search results, it provides an opportunity to the search engine vendors to apply advanced natural language processing (NLP) tools for understanding the user's search intent. *Query segmentation* is the first step along this process—it separates the words in a query text into various segments so that each segment maps to a distinct semantic component.

The interface of a modern web search engine is interactive. A user submits a search query by typing a text with several keywords in the search text box. The search engine removes the stopwords from the query to convert it into a processing format; occasionally, this step also includes the detection of phrases in the query. Then, the engine uses a word-based or a phrase-based inverse lookup table to retrieve the results which it presents to the user in the relevance order. Based on the quality of the search results, the user modifies the search query for expanding, narrowing, or re-ranking the search results. The process repeats until the user obtains her desired information or abandons the search out of the frustration caused from repeated failures.

Building a search index is a mature technology in search engine industry; however, detecting proper phrases is still not used actively by most of the search engines. For instance, not all search engines index the noun phrases, such as, a company name or a city name, in their inverted index. Nevertheless, they provide a partial solution for imposing phrase constraints in the query—a user can put double quotes around some query words to mandate that they be treated as a phrase; in that case the search

engine retrieved *only* those results in which the words in a phrase appear together. The task of query segmentation aims to shift this burden from the user to the search engine by automatically identifying phrases using the structural relationship among various words in a query text.

There have been significant research efforts in the field of query segmentation, however, the published works on query segmentation mainly focus on the web domain. For web queries, the segmentation mainly identifies the noun phrases that denote the name of a person, or a place in the query text. However, in the E-Commerce domain, the queries mainly represent a product that a shopper is interested to purchase from an online shop such as, eBay or Amazon. Product queries are different than the web queries on various aspects, but to the best of our knowledge, none of the existing works specifically address segmenting product queries.

The task of query segmentation is the same for both the web queries and the product queries. For both the cases, segmentation helps understanding user's search intent. However, the latter is significant for its potential usages in building various other applications. Typically, an E-Commerce query is in the form of free text, which does not specify the product in a well-structured form. A segment of a query text can be the name of the core product, while other segments can denote various other attributes of the product, such as, its model, color, or manufacturer; also, these segments can appear in the query in an arbitrary order. For an example, consider the query, `apple iPhone 4 white AT&T`. In this query, `iPhone 4` is the core product name, `apple` is the manufacturer, `white` is the color, and `AT&T` is the wireless service provider. Also for the query, `pottery barn shower curtain`, `pottery barn` is the manufacturer name, and `shower curtain` is the core product name. By segmenting a product query into various semantic units, an E-Commerce vendor can build various applications to benefit its customers—examples include query suggestion, automatic product catalogue generation, and attribute-based product indexing. Below

we discuss the benefits of a query segmentation task from the perspective of a online marketplace.

Improve the Precision of Search Result: Segmenting a query helps the marketplace to refine the search result by applying appropriate phrase constraints. Thus, the result set shrinks from the omission of the irrelevant products, and the precision improves.

Assist novice shoppers: Query segmentation is the first step for building applications such as *query suggestion*, and *query reformulation*, that are provided by the online marketplace to help unseasoned shoppers.

Build Product Catalog: Query segmentation helps converting unstructured text to structured data records with a well-defined schema. An E-Commerce catalog is comprised of specifications for millions of products. A comprehensive product catalog is a prerequisite for the effectiveness of an E-Commerce search service. Query segmentation helps in entity resolution which targets at structured and properly segmented phrases [2].

Recent research works suggest a variety of approaches to perform the query segmentation; we summarize those in the following paragraphs.

The first approach is based on mutual information between a pairs of query words [1, 3, 6]. If the mutual information value between two adjacent words is below some specific threshold (normally is 0), a segment boundary is inserted at that position. This approach has some limitations: first, MI approach cannot work beyond a specific length, so for long queries they are not applicable; second, MI relies heavily on the frequency statistic, so large training set is required so that the frequency statistic is reasonably accurate. Also, in some cases, frequency value is misleading, because

there are highly frequent patterns that are semantically meaningless (for example, the phrase `is a`). Nonetheless, in many query segmentation studies, mutual information based segmentation is used as a baseline for performance evaluation.

The second approach uses supervised learning [1, 4, 5]. Bergsma and Wang [1], one of the first works is this direction, establish the first standard corpus of 500 queries for supervised training; three human annotators segment each of the queries in the corpus. They use SVM (support vector machines) classification model for the supervised classification. Yu and Shi [4] provide a principled probabilistic model based on conditional random field (CRF); the CRF in this model is trained from the past search history and is adapted to user feedback. However, the limitation of this work is that methods based on CRF need large training data, which may be hard to obtain. Also, this work focus on query segmentation in the context of text stored in relational database; for this, it uses some database specific features which cannot be easily applied to unstructured text data.

The third approach is unsupervised method [8,12,13]. Tan and Peng [12] suggest unsupervised method based on expectation maximization. Their methods use n-gram frequency from web corpus and use Wikipedia as external knowledge to improve the query segmentation result, however, their work only considers web queries, whereas in this work, we considers E-commerce queries. We also use the Wikipedia to improve the segmentation accuracy for unknown word detection.

For the segmentation of Chinese queries, dictionary-based methods [9,17,23] are utilized recently. Such a method mainly employs a predefined dictionary and some rules for segmenting input sequence. These rules can be classified based on the scanning direction and the prior matching length. Using the Forward Matching Method (FMM) and the Reverse Matching Method (RMM), a dictionary-based method scan the input string from both the directions. The main disadvantage of dictionary-based method

is that its performance depends on the coverage of the lexicon, which may never be complete because new words appear constantly.

1.1 Contribution of this Thesis

In this thesis, we consider the task of query segmentation for segmenting E-Commerce queries. We adopt an unsupervised approach, which uses the normalized frequencies of queries for computing mutual information (MI) statistics. For the fast computation of MI statistics, we use a novel prefix tree like data structure. Similar to some of the existing works [12], we also use Wikipedia to recognize words for which no frequency statistics is available in the training data. We call our method a *hybrid method* for query segmentation.

Computation of MI requires the knowledge of frequencies for various queries. Typically, This information is available from the query log of an E-Commerce marketplace; besides query frequency, this log also stores a comprehensive search behavior of its visitors. Unfortunately, this data is not available to public, which is a significant bottleneck. In our work, we discover a proxy for query frequency, which is the number of items returned by a query; our experience shows that the above proxy also works well in practice.

The main contribution of our work is summarized as below:

- We propose a hybrid method for segmenting E-Commerce queries using an unsupervised approach. The hybrid method computes MI statistics from query frequency and use it for detecting the query segments; in case, the query contains words which no frequency information, the hybrid method uses Wikipedia.

Experiments show that the hybrid method performs better than other competing methods.

- We invent a prefix-tree like data structure for processing the frequency data effectively. It also work as an index for retrieving the frequency data of a query word. This data structure improves the execution time of the segmentation task substantially.

- We invent a proxy for the query frequency, which is the average number of listings that a query returns on an E-Commerce marketplace. Query frequency data is private, and the number of listings is public; so the proxy that we develop allows other researchers to work on query segmentation, even though the researchers do not have access to the query frequency data.

- We propose two evaluation metrics for query segmentation; these metrics are useful because the evaluation of E-Commerce queries is difficult, due to the lack off labeled corpora for such queries.

This thesis is useful to two groups: first, third party users who are interested in the segmentation of E-Commerce queries; second, an E-Commerce marketplace who considers applying segmentation to their queries for building tools such as query suggestion, and automatic catalogue generator. For the third party, the proxy to the query frequency should be interesting, as it would allow them to obtain training data for the segmentation task. On the other hand, the marketplace may find the comparative study among various segmentation methods, that we present in this thesis, useful. Further, they can try to adopt the hybrid method that we propose, which is better than the existing methods.

The rest of the thesis is organized as follows. In Chapter 2, we review the related works in query segmentation. In Chapter 3, we describe our proposed algorithm based on prefix tree, mutual information, relative frequency count and Wikipedia.

Chapter 4 introduces two evaluation metrics and reports the results. In Chapter 5, we conclude our study with a discussion and suggestions for future research.

2. PREVIOUS WORKS

In natural language processing, there has been a significant amount of research on text segmentation; examples include conditional random fields (CRF) based methods [4,5,11,22], mutual information (MI) based method using query frequency from query log [6], unsupervised methods using expectation maximization(EM) algorithm [12,13] and Chinese word segmentation [14,16]. Query segmentation in E-Commerce domain is similar to these works in the way that they all try to identify meaningful semantic units from the input.

The baseline approach for query segmentation that has been studied in previous work is based on mutual information (MI) between pairs of query words. Some researchers have considered using mutual information and context information to build a dictionary based on the statistics directly obtained from the training corpus. By contrast, we are using mutual information to prune a given dictionary. That is, instead of building a dictionary from scratch, we first populate the dictionary using all possible words in the training set and then use mutual information to prune away irrelevant words. Hence the statistics we use for calculating mutual information are more reliable than those directly obtained from corpus by frequency count [15]. For query segmentation for web search, [6] is one of the earliest approaches that works with web query. It segments queries by computing the so-called connexity score for a query segment by measuring the mutual information statistics among the adjacent terms. The limitation of connexity score is that it fails to consider the query length in account; also note that mutual information cannot by applied to more than three words [26]. Another problem with this approach is that it relies heavy on the frequency data. Consequently, it generates many non-sense but highly frequent phrases. In our approach, we introduce a weighting function to normalize the n-gram frequencies; we

also include relative frequent count to consider only the frequent words to calculate the mutual information. Note that mutual information segmentation often performs worse than the more involved methods.

One of the earliest methods that do not rely on mutual information is the supervised learning approach by Bergsma and Wang [1]. Bergsma and Wang propose a data-driven, machine learning approach to query segmentation. In their approach the decision to segment or not-segment between each pair of token is a supervised learning task. To facilitate this learning, they have created and made available a set of manually-segmented user queries; to build statistical features for the supervised classification, they used the phrase frequency data; they also created dependent features that are built on noun phrase queries. Yu and Shi [4] provide a principled probabilistic model based on conditional random field (CRF) that can be learned from past search history. They also show how a CRF model can be adapted by using user feedback. However, supervised approach requires large training data. Yu and Shi use the data stored in relational database and employ database-specific features to implement query segmentation. Bergsma and Wang use the dataset from AOL search query database which consists of 500 queries; they take queries that are of length 4 words or greater and contain only adjectives and nouns. The query sample of their corpus is not representative, because of the small number of queries and constraint bias.

Instead of supervised approach that requires training data, Tan and Peng [12] suggest unsupervised method. Tan and Peng's method utilize Google's n-gram frequency, a well-known web corpus and also Wikipedia. They setup a language model from the n-gram frequencies using expectation maximization (EM) method. The EM based method has also been used for Chinese word segmentation, where EM algorithm is applied to the whole corpus. To avoid this costly procedure, Tan and Peng run an EM algorithm on the fly over the affected sub-corpus. In their method, a segment's score

which is derived from the language model is increased by using external knowledge from Wikipedia. In our work, we also use Wikipedia or new word (unknown word) identification.

Hagen et. al. [19] score all segmentation for a given query by the weighted sum of the frequencies of contained n-grams which is obtained from Google web corpus. The Google n-gram corpus contains n-grams of length 1 to 5 along with their frequencies which is built from a 2006 Google index. Their algorithm derives a score for a valid segmentation. First, the n-gram frequency count of each of the potential segments is retrieved. Then all valid segmentations are enumerated and their frequency is normalized. The objective of normalization is to reduce the score gap so that longer segments have a chance to achieve a higher score than the shorter ones. For example, `iPhone 4s` has a much larger frequency count than `apple iPhone 4s`, the length-based frequency normalization avoids segmentation like `apple | iPhone 4s`, by assigning reasonably high score to the phrase `apple iPhone 4s`, so that the entire string can be treated as one segment. Hagen et. al.'s approach achieves good runtime performance. However, no explanation is given why the exponential normalization schema of nave query segmentation performs so well.

In Chinese word segmentation, dictionary-based method mainly employs a predefined dictionary for segmenting input sequence. One popular dictionary-based segmentation approach is the maximum matching method. The basic idea behind this method is that an input sentence should be segmented in such a way that the number of words produced should be the minimum [23]. The algorithm starts from the beginning of a query, finds the longest matching word and then repeats the process until it reaches the end of the sentence. The coverage of a dictionary is essential to the quality of segmented text. If a dictionary contains only a small portion of the words in the corpus to be segmented, many words are treated as unknown. The handling of unknown words in the process of segmentation is a difficult task. This method cannot deal

with the unknown words identification and may result in wrong segmentation. The maximum matching method matches query either from the beginning to the end or from the end to the beginning. The Forward Matching Method (FMM) groups the longest initial sequence of characters that match a dictionary entry as a word, then starts at the next character after the most recently found word and repeats the process until the end of the input sentence. The Backward Maximum Matching (BMM) works from the end of a sentence toward the beginning. This matching approach is fast, so it is good for those tasks where speed is the primary concern.

To summarize, all methods rely on frequency statistics (word by word information indicated by the co-occurrence probability or conditional probability) or machine learning method or dictionary-based method. It is difficult for statistical methods to segment words when sufficient information is not available. The hybrid unsupervised segmentation methodology proposed in this thesis combine the merits of existing approaches. We collect the frequent user queries from an E-Commerce website to setup a predefined dictionary. Then we perform the segmentation is the following three steps: First, we apply a dictionary-based method to the input text in order to divide the text into as many recognized segments (words) as possible, resulting in a partially segmented text. Next, using mutual information which is the best method for measuring words association, we prune away illegal words. The remaining undecided words of the text are then submitted to Wikipedia to detect unknown words. From our experiment, using both dictionary-based method and statistical approach improve the segmentation accuracy. The result of hybrid methodology is better than any one of the approach used alone—we will validate this claim in the evaluation section.

3. METHODOLOGY

Query segmentation in E-Commerce is defined as follows. Given a query from users, we group the words and help the users to better retrieve product information. Table 3.1 shows some examples of query segmentation. For instance, if the query typed by the user is `iPhone 3g external battery`, it is likely that a lot of relevant product documents will be retrieved. Because many search engines incorporate an inverted index to quickly locate documents containing the words in a query. The inverted index stores a list of the documents containing each word (example shown in Table 3.2). The return documents are applied an intersection algorithm which is to retrieve those documents where these words are appearing without sequence requirement. However, users want to get product documents where these words are appearing in the same order as in the phrase they put. So It is better to group the phrase `iPhone 3g` together by inserting double quotes around a phrase which telsl the search engine that the words should be present in the same sequence as the search phrase.

Table 3.1
Examples of Query Segmentation

Original Query	Segmented Query
white pearl earrings	white “pearl earrings”
apple ipad 2 smart cover	apple “ipad 2” “smart cover”
ipod touch 4th generation case	“ipod touch” “4th generation” case
princess diamond engagement ringr	princess diamond “engagement ring”
white gold wedding ring sets	“white gold” “wedding ring” sets

Table 3.2

Inverted Index

Word	Documents
iPhone	Document1, Document3, Document4, Document5
3g	Document2, Document3, Document5
external	Document3,Document5
battery	Document1, Document2, Document3, Document4, Document5

Query segmentation is by nature a structured prediction task. Specifically, given a sequence of query words, we predict association words. This thesis uses prefix tree, mutual information (MI), relative frequent count and Wikipedia to perform E-Commerce query segmentation. First we collect the frequent user queries from E-Commerce website to setup a predefined dictionary. Then segmentation is performed in three stages: First, a dictionary-based method is applied to the input text in order to divide the text into as many recognized segments (phrases) as possible, resulting in a partially segmented text. Next, mutual information prunes away illegal words (lower mutual information value and smaller relative frequeny count). Third, the remaining undecided words of the text are then submitted to Wikipedia to detect unknown words.

3.1 Query Segmentation: Problem Formulation

In this section, we formally define query segmentation.

DEFINITION 1 (TOKENS AND PHRASE). Tokens are strings which are considered as indivisible units. A phrase is a sequence of tokens.

DEFINITION 2 (INPUT QUERY) An input query Q is a pair (t_Q, p_Q) where $t_Q = \langle\ t_Q(1), t_Q(2), ..., t_Q(n)\ \rangle$ is a sequence of tokens, and $p_Q = \langle\ p_Q(1), p_Q(2), ..., p_Q(n)$

$\rangle$ is a sequence of increasing integers. The value $p_Q(i)$ is the position of the token $t_Q(i)$ in query Q. The number of tokens in Q is its length $|Q|$.

For example Consider the query: Q = `apple iPhone 4 leather case`. The tokens and position values are as follows.

apple	iPhone	4	leather	case
1	2	3	4	5

DEFINITION 3: (SEGMENTS AND SEGMENTATION). A segmentation is a sequence of segments $\mathcal{S}$ = $\langle$ S_1, S_2, ..., S_K $\rangle$ where for all k <= K, end(S_k) +1 = start(S_{k+1}). Namely, the segments are continuous and non-overlapping. We define the start and end of a segmentation as start($\mathcal{S}$) = start(S_1) and end(($\mathcal{S}$) = end(S_k)).

For example, continue with the previous example, segment S_1 = $\langle$ (2,3) $\rangle$ corresponds to the term `iPhone 4` and segment S_2 = $\langle$ (4,5) $\rangle$ corresponds to the term `leather case`. The valid segmentation should not have overlapping token in phrases. The following are two valid segmentations:

$\mathcal{S}_1 = \langle(1,1), (2,3), (4,5)\rangle$
$\mathcal{S}_2 = \langle(1,3), (4,5)\rangle$

3.2 Data

Currently, most of query segmentations focus on the web domain or some text data retrieved from relational databases. Three main datasets for segmentation algorithms are RDBMS, web search logs, and Google n-gram corpus [21] , the last one contains n-gram of length 1 to 5 from the 2006 Google index along with occurrence frequencies

extracted from trillion words of web pages. Although the n-gram corpus is easy to be applied in an application, its resources lack of other linguistic information.

Analysis of the commercial web search logs and user activity records have been proved to be a valuable resource for the researchers in the field of information retrieval, data mining, machine learning, and natural language processing. Large volumes of user queries were successfully leveraged in query segmentations and term associations [33]. Search logs provide an insight into the searcher behavior. Downey et al. [34] investigated the influence of query frequency on user behavior, and found that the rarer queries result in less clicks and fewer page visits. They concluded that users tend to be more satisfied with the results of the more popular queries. They also stated that "query frequency is more important than query length indicating that web search engines are optimized to handle common requests". This result is relevant to our work, since we will use query frequency to calculate the phrase statistical data.

Web query logs are the best source of information for building query segmentation algorithms. We could use queries in user session from historical logs as training data to build the data dictionary. The log comprises of a set of user sessions on the E-Commerce website. Each session stores date, time, customID, and a set of user activities. Some example events include purchase of an item, search some queries, and click on related search suggestions. However, the web log is highly confidential data for E-Commerce web sites like eBay or Amazon whose logs could disclose a lot of significant information to the competitors. So we propose one workaround solution for this data collection problem that does not enterprise confidential logs. We extract keywords list for all the categories from eBay lab website which is shown in 3.1 and send keywords to eBay search engine to retrieve searching result number which could represent user query frequency.

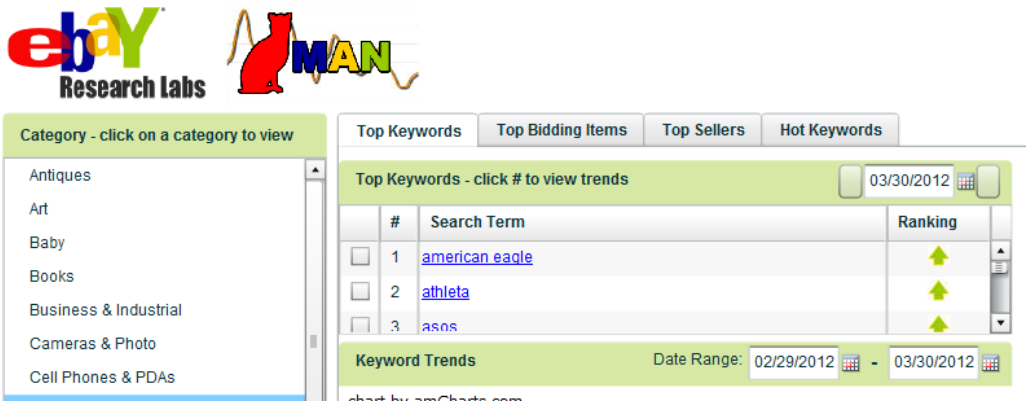

Figure 3.1. eBay Lab Data for Top Keywords Per-Category

CATMAN software on the eBay lab website keeps the top frequent queries by previous query logs. We could regard that top frequent queries are good and valid phrases as well as are the keywords for the search engine. Now we have the frequent queries, the next step is to find the queries frequency (number of times a query is searched in the web). We will send every keyword we extract from CATMAN software to eBay search engine. The search result count has the same approximation number as the query frequency. As we all know, user queries represent the demand side of the marketplace and the size of the retrieved item-set represents the supply counterpart. In eBay marketplace, the demand and supply correlates nicely as shown in 3.2 [10]. Because of its correlation, we try to use the search result number from supply side shown in 3.3, to represent the query frequency from the demand side.

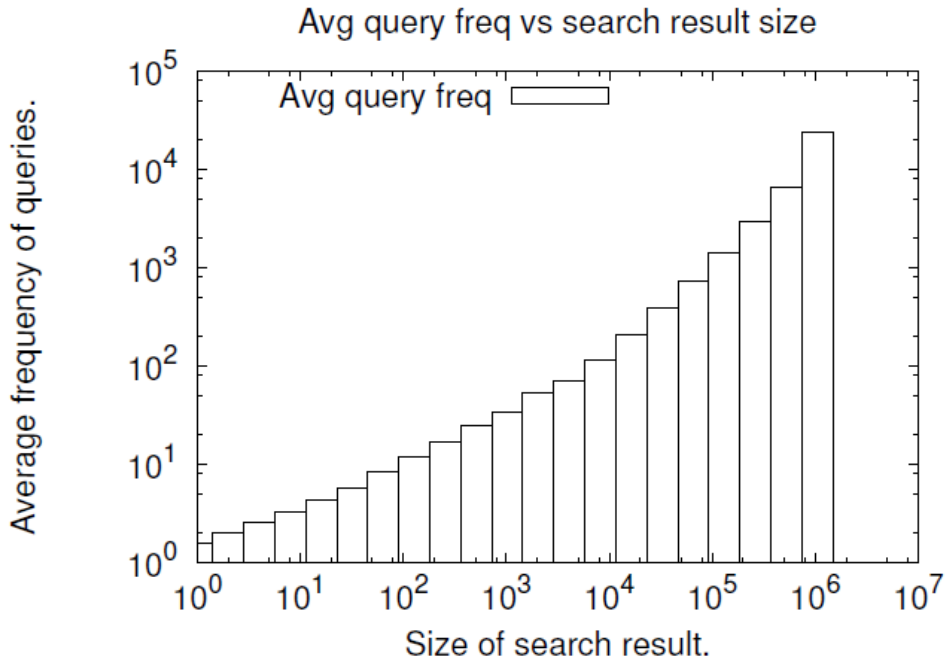

Figure 3.2. eBay marketplace Demand and Supply Correlation

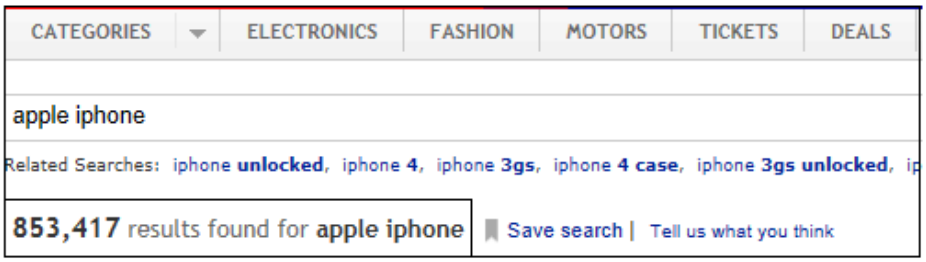

Figure 3.3. eBay Web Search Result Count from Supply Side

Figure 3.4 shows the distribution of queries by length. Query lengths demonstrate a power-law distribution, with the long queries in the tail. Most queries in the search log are short. Queries with $|q| \leq 4$ account for 90% of the total queries. Due to our

focus on Keywords from eBay lab web, our main queries are queries for which $|q| \leq 5$. We divide the queries into two main types: short and long. The division is based on query length. Short queries are queries for which $|q| \leq 3$ and long queries are queries for which $4 \leq |q| \leq 5$.

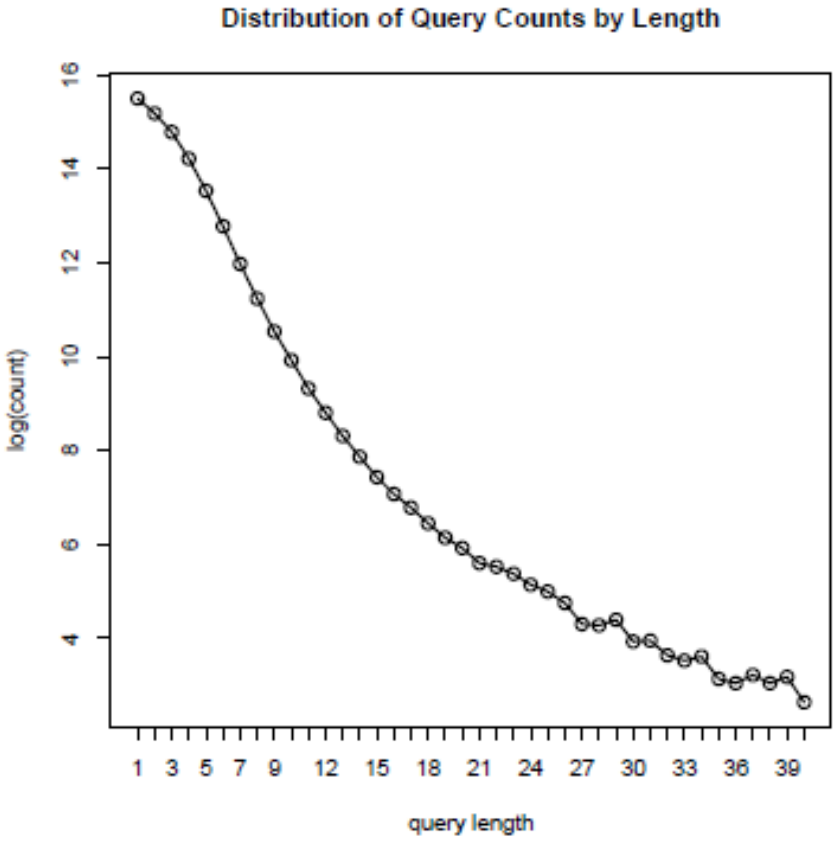

Figure 3.4. Distribution of Query Counts by Length

3.3 Prefix Tree Basics

The prefix tree is a data structure for storing strings or other sequences in a way that allows for a fast look-up [27]. It consists of one root labeled as "NULL", a set of item

prefix sub-trees as the children of the root, and a keyword token header table. Each node in the prefix sub-tree consists of three fields: token, count and node-link: the token represents a word from the query set, the count field indicates the number of frequency of query, and node-link connects to children in the prefix tree, or is null if there is no child. Head table records each token appeared in the query, token frequency and pointer linked to the next node which has the same token in the prefix tree.

The basic idea behind our prefix tree is that each successive word is stored as a separate node. All the descendants of a node have a common prefix of the string associated with that node, and the root is associated with the empty string. For instance, we have two queries: `apple iphone 4s` and `apple ipad 2`. We construct a null node as root followed with `apple` node which has two children: `iPhone` and `ipad`. Each of children has its own child `4s` with `iPhone` and `2` with `iPad`.

Let us illustrate by an example for this tree construction. Suppose we have query set shown in Table 3.3.

Table 3.3

Data Set with Five Queries

Query Data Set		
QID	Query	Frequency
100	Apple iPhone white	100
200	Apple iPhone 4s	200
300	Apple ipad 2	500
400	iPhone car holder	500
500	iPad white	1000

By scanning the dataset, we get the pairs (token: frequency): (apple: 800), (iPad: 1500), (2: 500), (iPhone: 800), (4s: 200), (white; 1100), (car; 500) and (holder; 500).

We use the tree construction algorithm to build the prefix tree. We scan each transaction and insert the tokens into the tree. First, we insert `Apple iPhone white 100` to the empty tree. This results in a single path:

Root(NULL) → (Apple:100) → (iPhone:100) → (white:100)

Then, we insert `Apple iPhone 4s 200`. This leads to two paths with `Apple` and `iPhone` is being the common prefixes:

Root(Null) → (Apple:300) → (iPhone:300) → (white:100)

And

Root(Null) → (Apple:300) → (iPhone:300) → (4s:200)

Third, we insert `iPhone car holder 500`, we get a new path:

Root(Null) → (iPhone:500) → (car:500) → (holder:500)

In a similar way, we can insert `Apple ipad 2` and `iPad white` to the tree and get the resulting tree as shown in Figure 3.5, which also shows the horizontal links for each token in dotted-line arrows. Table 3.4 is a keyword token header table.

Table 3.4

Token Header Table

Word	Frequency
Apple	800
iPad	1500
2	500
iPhone	800
4s	200
white	1100
car	500
holder	500

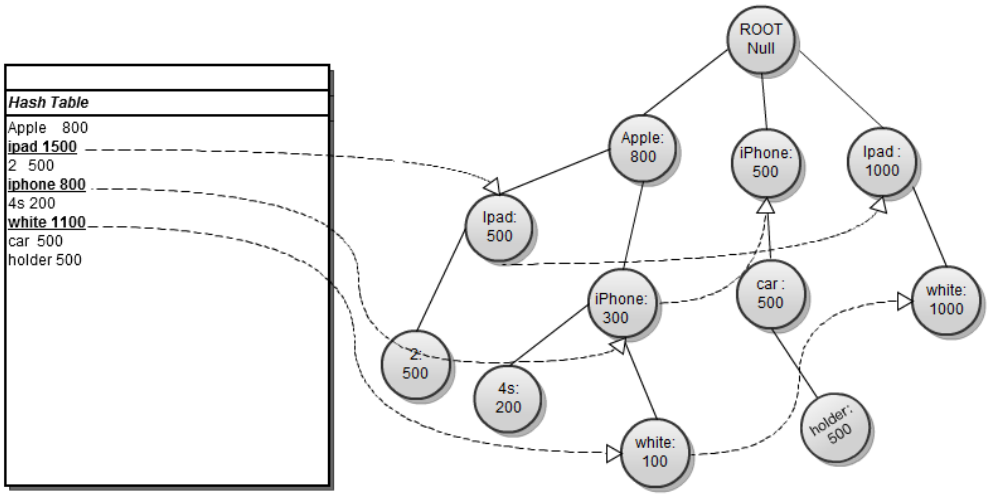

Figure 3.5. Results Header Table and Prefix tree in the example

3.4 Statistic Method

3.4.1 Mutual Information

Mutual information is a measure of association among words in a query. It compares the probability of a group of words to occur together to their probabilities of occurring independently. The two words mutual information is known as [28]:

$$I(x,y) = log_2 \frac{P(x,y)}{P(x) * P(y)} \tag{3.1}$$

Where $P(x)$ is the probability of observing token x, $P(y)$ the probability of observing token y, and $P(x,y)$ the probability of observing this phrase. If two tokens occur independently, the joint probability $P(x,y)$ should be close to the product of $P(x)$ and $P(y)$, thus the mutual information would be close to zero. On the other hand, if two tokens are strongly related and could construct to a phrase, the joint probability $P(x,y)$ would have a much larger value than the product of $P(x)$ and $P(y)$, so $I(x,y)$ would be much bigger than zero; if two events occur complementarily, the mu-

tual information value would be negative. The Table 3.5 shows the frequency values and mutual information values for two bigrams in our previous data collection. In Table 3.5, column $f(t_1)$ is the frequency value of the first token t_1 and column $f(t_2)$ is the frequency value of second token t_2; column $f(t_1\ t_2)$ is the occurrence frequency value of bigram phrase; and the last column, $I(t_1\ t_2)$ is the mutual information.

Table 3.5
Mutual Information of Bigrams

Bigrams	$f(t_1)$	$f(t_2)$	$f(t_1t_2)$	$I(t_1t_2)$
iPhone 4s	800	200	200	1.52
Car holder	500	500	500	2.2

The $P(t_1)$ in the mutual information definition is estimated by $f(t_1)/N$, $P(t_2)$ is estimated by $f(t_2)/N$, and probability of observing two tokens t_1t_2 occurring in the collection together in fixed order $t_1\ t_2$ is estimated by $f(t_1t_2)/N$.

However, it is only for capturing correlations among two words not more than two words. The mutual information of a trigram (three words) is defined as [29]:

$$I(x, y, z) = log_2 \frac{P_D(x, y, z)}{P_I(x, y, z)} \tag{3.2}$$

Where $P_D(x, y, z)$ is the probability for x, y and z to occur jointly, and $P_I(x, y, z)$ is the probability for x, y and z to occur independently. So in this situation $P_I(x, y, z) = P(x) * P(y) * P(z) + P(x) * P(y, z) + P(x, y) * P(z)$. In general case, the result for mutual information > 0 means the words are strongly connected. In my java coding, there is one class named Dictionary in which we have two methods to calculate the mutual information for bigram and trigram (MutualInformationBigram and MutualInformationTrigram).

Other researchers have considered using mutual information to build dictionary. By contrast, we are using mutual information to prune a given dictionary (prefix tree). We first add all possible words and then use mutual information to prune away illegal phrases (MI<0). Hence the statistics we use for calculating mutual information are more reliable than those directly obtained from corpus by frequency counting.

The drawback of this approach is that using this approach is difficult for capturing correlation among more than three words, thus it cannot handle long entities like sentence.

3.4.2 Relative Frequency Count

The relative frequency count for the n-gram is defined as [29]:

$$r_i = \frac{f_i}{K} \tag{3.3}$$

Where f_i is the total number of occurrences of the n-gram in the dataset, and K is the average number of occurrence of all the entries. In other words, f_i is normalized with respect to K to get the relative frequency. It may not worth the cost of entering the compound into the dictionary if it occurs every few times. The relative frequency count is therefore used as a feature for compound extraction. So, using both mutual information and relative frequency count as the extraction features are desirable since using either of these two features alone cannot provide enough information for compound finding. By using relative frequency count alone, it is likely to choose the n-gram with high relative frequency count but low mutual information among the words. For example, if $P(x)$ and $P(y)$ are very large, it may cause a large $P(x, y)$ even though they are not related in the context. Mutual informaiton formulation $\frac{P(x,y)}{P(x)*P(y)}$ would be very small for this case.

On the other hand, using mutual information alone may be highly unreliable if $P(x)$ and $P(y)$ are too small. An n-gram may have high mutual information not because the words with it are highly correlated but due to a large estimation error. Actually, the relative frequency count and mutual information supplement each other. A group of words of both high relative frequency and mutual information is most likely to be composed of words which are highly correlated, and very commonly used.

3.4.3 Maximum Matching

Maximum matching [30] is one of the most popular structural segmentation algorithms for Chinese text. This method favors long words and is a greedy algorithm by design, hence, sub-optimal. The matching direction can be forward or backward. The forward matching starts from the beginning of a sentence, finding the longest words among all the possible phrases and then repeating the process until it reaches the end of the sentence. The backward matching starts from the end of a sentence then works toward the beginning of the phrase.

The proposed algorithm of this paper makes use of a forward maximum matching strategy to identify and calculate good phases. We use MI and relative frequency count to evaluate the tokens' binding force which is a measure of how strongly are associated with another token. So, in this respect, we use both structural algorithm as well as a statistical approach.

For example, we process the user input query to calculate MI and relative frequency number based on prefix tree dictionary. The input query will be scanned and taken n tokens from the beginning as matching fields. The n here is 3 because of calculating trigram and bigram mutual information which will be used in the next scoring of the queries. This algorithm starts at the first token in a query and use a phase list

to keep all sub queries (phrases) statistical information. If phrases are found, max matching algorithm saves the sub queries as well as their statistical data calculated from the dictionary. Then remove the last token from matching field which becomes the bigram calculation and repeat to match dictionary again until one token left in matching fields. If match failure, save the queries, give the statistical values zero and save it to the list. Reoperation of the above until end of the user query. The Figure 3.6 will display the steps for max matching method.

After processing the matching algorithm, we get the phrase list which has all combinations of the trigram and bigram phrases with mutual information and relative frequency number information. We apply the filter logic on this list to find the good and valid phrases. From the statistical methods, we only consider a potentially meaningful phrases with two conditions:

- The phrase is significantly frequent in the data dictionary
- The phrase has good mutual information.

So we filter the phrases list by extracting only phrases whose mutual information is more than 0 as well as relative frequency numbe is bigger than 1.

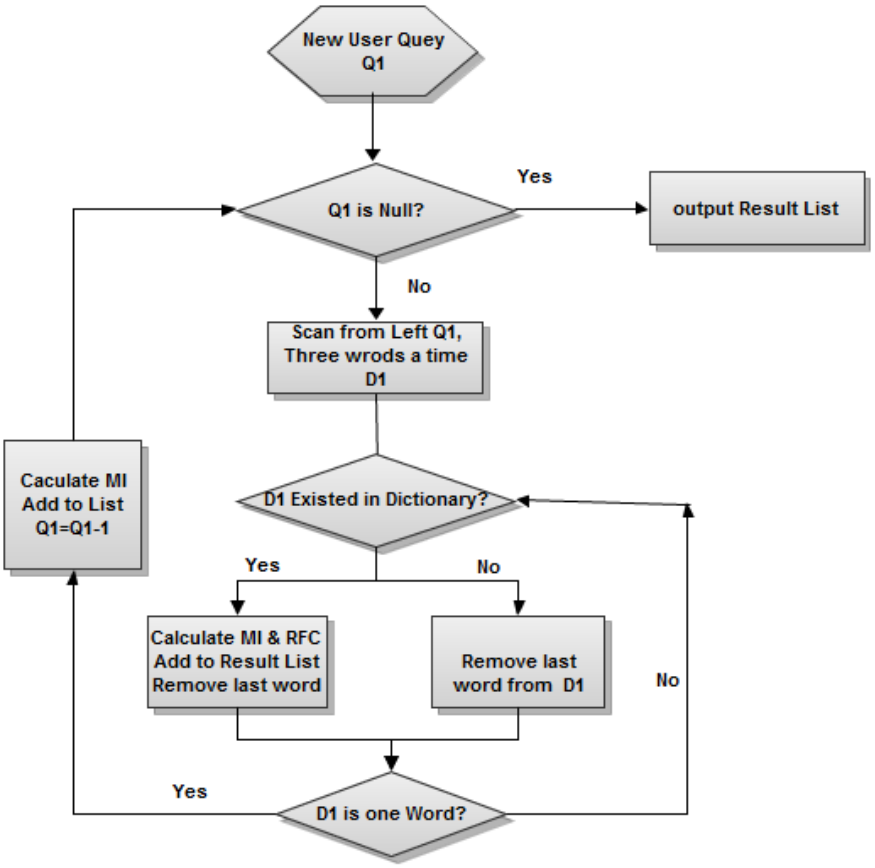

Figure 3.6. The Workflow for Maximum Matching Method

3.5 Use of Wikipedia

Since no dictionary could ever be complete, new word (unknown word) identification is an important issue in the query segmentation. Unrecognized words cause segmentation errors in that these out of vocabulary words in input text are often incorrectly segmented into single-character or other overly-short words [31]. So new word detection has been considered as an important process. We regard the new word detection as an integral part of the segmentation task, aiming to improve both segmentation and new word detection [32].

Wikipedia is the largest encyclopedia on the Internet and well known for its high-quality collaboratively edited page contents. We find that the Wikipedia article titles are particularly suitable to our needs: the articles span a huge number of topics with extremely wide coverage, ranging from persons to locations, from movies to scientific theorems, able to match the great diversity of the web search queries. Then, most articles are about well-establish topics which are known to a reasonably-sized community, thus we can avoid dealing with large numbers of infrequently used concepts. Furthermore, the articles are updated very fast, thus one can keep the concept dictionary up with the latest trend [12].

In our work, segmentation is performed in three steps: First, a dictionary-based method is applied to the input text in order to divide the text into as many recognized segments (words) as possible, resulting in a partially segmented text. Next, mutual information is best method for measuring words association and relative frequency count prune away illegal words. The remaining undecided words of the text are then submitted to Wikipedia to detect unknown words. Because after the first two steps, we still miss two kinds of scenarios of the good query segmentation. First one is the new word recognition (relative frequency number is 0), and the second one is good query but with limited frequency count in our dictionary (relative frequency number < 1 and mutual information >0). So in these two scenarios, we will send the query to Wikipedia website to check whether they are a valid article title. If they are article titles, we regard these queries as good queries, and save them in the phrase list.

3.6 System Architecture

We use both structural algorithm and statistical approach to filter the queries, and then send the undecided queries to the Wikipedia to check their accuracy. Figure 3.7 illustrates the architecture of our system. We have 500 thousand user queries and

build the prefix tree dictionary in memory by an offline processing. We provide the option to save the tree structure on the local disk after the first timeit is constructed for the first time so that we can import the serialized tree from the disk to memory which saves tree build time. When the dictionary is setup, we request the user to input query which needs to be segmented online.

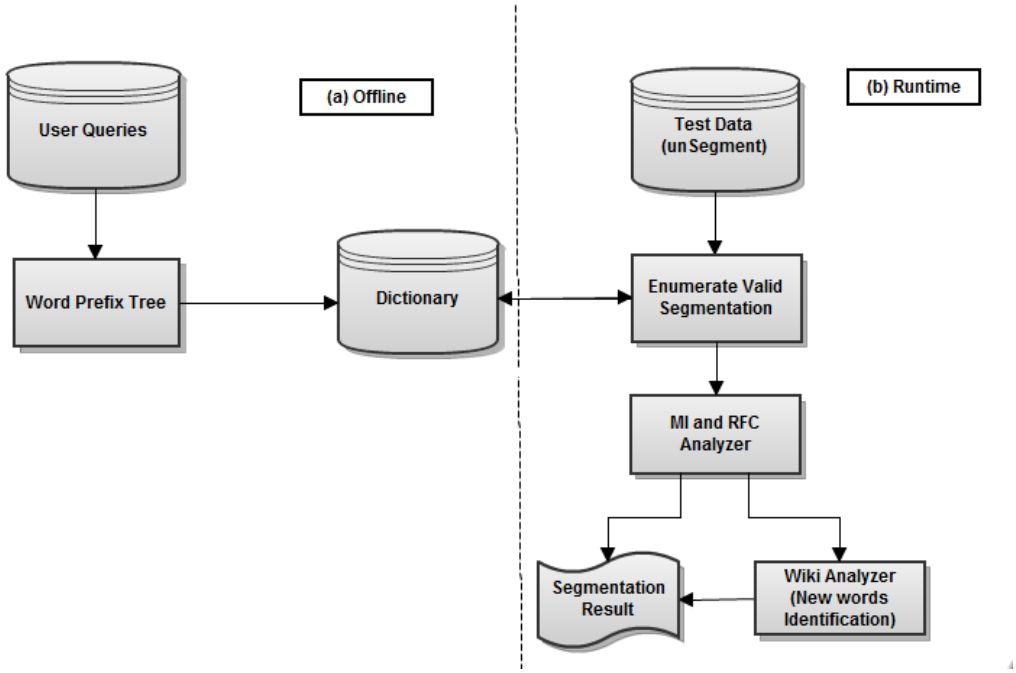

Figure 3.7. System Architecture

The unsegemented query will be processed twice to two Analyzer: MI and RFC Analyzer and Wikipedia Analyzer. In this step, We compute scores corresponding to a query and order the pair of queries. The scores are composed of three factors: mutual information, relevant frequent number and length of queries. All the valid segmentations are enumerated using maximum matching method, and, for each segmentation S, a score is computed according to the following function:

$$score(s) = 1000 * RFC(s) * MI * |s| \tag{3.4}$$

Because of the relative frequency count (RFC) value is small , the factor 1000 is given to avoid precision loss. This measurement gives us the degree of strength by which two words or three words are connected. If we only calculate the mutual information and relative frequency number, there will be a problem for treating different length of segments. For example, frequency of `apple iPhone` is always bigger than `apple iPhone 4s`, but obviously `apple iPhone 4s` is a better phrase which have more accurate description for a product. So we involve the $|s|$ which will give weight to long segments compared to shorter ones. Furthermore, the good queries from Wikipedia are given the highest score because of the fact that a Wikipedia's title is always the best phrase. The following table shows the score of phrases `apple iPhone 4s car holder`.

Table 3.6

Score of Query

Score of Query	
2393	iPhone 4s
662	car holder
321	Apple iPhone 4s
70	Apple iPhone

The optimal segmentation is based on the score of query. The segmenation takes good phrases and cuts the token as boundary. For example, a segmentation for a query: `Apple iPhone 4s car holder`

The single phrase segmentations ($\mathcal{S}_i$) are as follows :

$\mathcal{S}_1$ = Apple "iPhone 4s" car holder

$\mathcal{S}_2$ = Apple iPhone 4s "car holder"

$\mathcal{S}_3$ = "Apple iPhone 4s" car holder

$\mathcal{S}_4$ = "Apple iPhone" 4s car holder

3.7 GUI of Query Segmentation

We provide a GUI for the above system for query segmentation in this way we could easily choose the dictionary building flat files, construct the dictionary, serialize the Prefix tree on the local disk, reload the dictionary and get the recommended query segment result. Figure 3.8 shows the GUI of this query segmentation system.

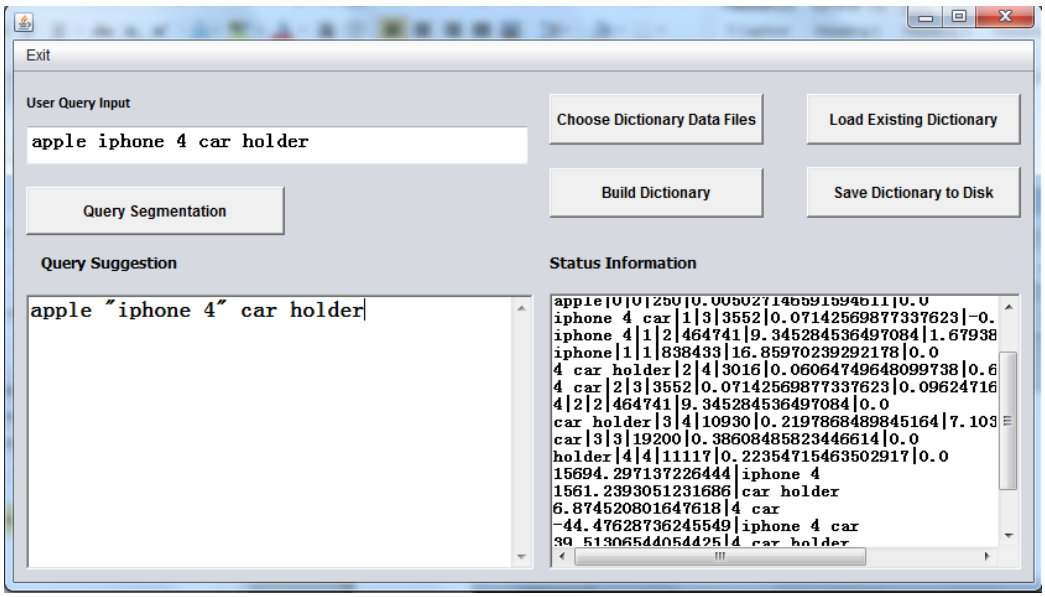

Figure 3.8. GUI of Query Segmentation

4. RESULTS

Evaluation of query segmentation is difficult, because the ground truth is not known. To the best of our knowledge, for E-commerce queries, no public corpus exists with manually labeled segmentations. Even though, we find experts who can segment queries for our evaluation, there is a high likelihood that there will be disagreement among various experts on the segmentation of a large number of queries. This is because, for many queries, there are multiple segmentations, all of which are perfectly valid. Consider the query, `lenovo t60 laptop`, both `lenovo t60|laptop` and `lenovo|t60 laptop` are valid segmentations, and it is hard to choose one over the other. So, we need evaluation metrics that are free from personal biases.

In this work, we propose two evaluation methods for segmentation that do not need manually labeled segmentation, hence they are free from personal biases. These methods use an E-commerce platform, such as eBay, or Amazon to collect statistics which they convert to a score for comparison. We name these methods as *phrase retrieval count*, and *phrase replaceability*, respectively. The first uses the number of product listing retrieved by the phrase-query built from the segmented form of the query, and the second uses query suggestion list to find whether the segmented phrase is a common part between a query and some of its top suggestion. For both the evaluation method, *accuracy* is our evaluation metrics; this is simply the ratio of correctly segmented queries for a given corpus. Let, $\hat{S}$ denotes the set of queries that are segmented correctly by a query segmentation method and S is the entire set of queries; then,

$$accuracy = \frac{|\hat{S}|}{|S|} \tag{4.1}$$

4.1 Evaluation based on phrase retrieval count

Modern search engines, both in web and in E-commerce, allow users to enter a phrase-query where multiple consecutive words are enclosed in a double quote. For example, instead of searching for `michael jackson poster`, a user can search for `"michael jackson" poster`, the latter enforces that the search engine considers the phrase "michael jackson" as a unit token and retrieve only those results in which they appear together in the correct order. If we obtain a multi-word phrase by segmenting a query, we can convert the query into a phrase-query by double quoting around the multi-word phrase. For a given query, the phrase-query obtained from a correct segmentation will retrieve more search result than that of a wrongly segmented query. For example, the phrase-query `"michael jackson" poster` will fetch many more results than the phrase-query, `michael "jackson poster"`, because the former is a correct segmentation, where the name "michael jackson" is correctly identified as a phrase, whereas in the latter it is not the case. Thus, our phrase retrieval count based evaluation is based on the following observation: *the larger the retrieval count of a phrase-query from a search engine, the better the quality of the corresponding query segmentation.*

To compute the accuracy using the above evaluation method, we segment each of the queries in our test dataset using our segmentation algorithm. The algorithm breaks a query into various segments—each of these segments has an associated score that defines the strength of that segment. For each query, we consider only the highest scoring segment as a phrase in that query, and put quotation around that segment to build a phrase-query. Then, we execute the phrase-query in eBay search engine and record its retrieval count. For the same query, we also build various other dummy

phrase-queries by ensuring that the number of words in the double quoted phrase of these dummy queries is the same as the number of words in the phrase-query that we have obtained earlier using our segmentation algorithm. Then we execute each of the dummy phrase-queries in eBay search engine and record their retrieval counts. If the phrase-query from our segmentation retrieves higher or equal number of search results than the search results of all the dummy phrase-queries, we consider our segmentation as the correct segmentation. Then we compute the accuracy of the segmentation algorithm using Equation 4.1.

Table 4.1
Example for Phrase Retrieval Count Evaluation

Segmentation	Search Results
“apple iphone” 4s car holder	717
apple “iphone 4s” car holder	322
apple iphone “4s car” holder	5
apple iphone 4s “car holder”	243

In Table 4.1, we show an example to evaluate a segmentation algorithm using phrase retrieval count. Consider the query, `apple iphone 4s car holder`, and the best-scoring segment of this query using a segmentation algorithm is `apple iphone`, so the corresponding phrase query for this segmentation is `"apple iphone" 4s car holder` which we show in the first column of the first row of the above table. The retrieval count for this query using eBay search is 717, which is shown in the second column of the same row. In the remaining rows, we show other dummy phrase-queries along with their corresponding retrieval counts. Clearly, the retrieval count of the algorithm’s phrase-query (fist row) is higher than the retrieval counts of all the dummy phrase-queries; so for this example, we will consider the algorithm’s segmentation to be correct. However, if the highest scoring segment from the algorithm is anything

else other than the `apple iphone`, then we will consider the algorithm's segmentation to be incorrect. Also note that, in all the dummy phrase-queries, the number of words in the quoted phrase is 2 which is exactly equal to the number of words in the phrase `apple iphone`, which is the highest-scoring segment obtained from the segmentation algorithm.

The phrase retrieval count based evaluation has some limitations. For instance using this method, only the highest scoring segment of a query plays a role in defining the accuracy of a segmentation method. For example, the query `iphone 4 white case` probably has two phrases in it, like, `iphone 4` and `white case`, but this evaluation will only consider the `iphone 4`' phrase for the evaluation. Also note that, the fact that the dummy phrase-queries should have the same length double-quoted segment as the original phrase-query is also a restriction, because it can happen that the best segmentation may be one that considers a different-length segment as the best-scoring segment. For instance, one can argue that for the query `apply iphone 4s car holder`, the best segment is neither `apple iphone`, nor `iphone 4s`, rather `apple iphone 4s`. However, we cannot simply compare the retrieval counts of phrase-queries with various length quoted phrases, because the retrieval count of a phrase-query containing a quoted sub-phrase of another phrase-query is equal or larger. For example, the number of results of `"apple iphone 4s" cup holder` is always smaller or at most equal to than the number of results of `"apple iphone" 4s cup holder`.

4.2 Query Suggestion Evaluation Method

After an end-user has input a query, intelligent search engines can suggest selectable suggestions for query terms to help end-users express their information needs. In E-Commerce area, the query suggestion system will help customers choose relevant and

popular products according to their input queries. All major web search engines and most proposed methods that suggest queries rely on search engine query logs to determine possible query suggestions. For eBay's query suggestion system, the relevancy of a key word is determined by query popularity and purchased-efficiency. You may notice that on occasion a term has a higher relevance than the term you entered. This is because it appears more often in eBay user session historical logs than your input term does. When a user types a query `iphone 4s` to the eBay search engine, he will be provided with quite a few alternative potential queries which are `iphone 4s case`, `iphone 4s accessories`, `iphone 4s charger`, and `iphone 4s unlocked`. The system provides those selectable suggestions to either replace or augment the current query because suggestions are most popular queries and have more money impact. Based on the above fact, our Query Suggestion based evaluation is based on the following observation: *if the good phrase detected by our system appears in the suggest query terms, we regard that this good phrase comes from a correct query segmentation.* Generally speaking, If end user inputs query q could be segmented as (s_1, s_2) and the suggestion queries from the E-Commerce platform are $q(s_1, s_3)$ and $q(s_1, s_4)$, then s_1 is a good phrase, because all the suggestion queries have the common phrases: s_1, as shown schematically in Figure 4.1.

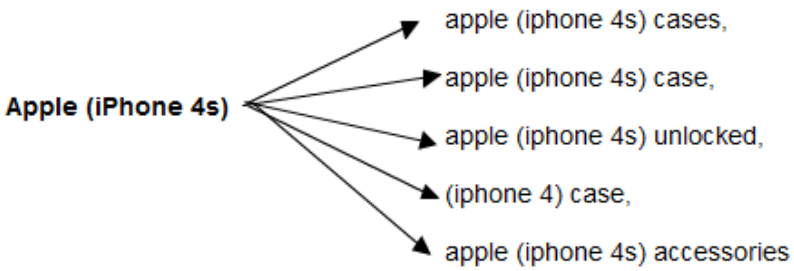

Figure 4.1. Example for Query Suggestion Evaluation

To compute the accuracy using query suggestion evaluation method, we execute each of the queries in our test dataset in eBay search engine and retrieve the first three suggestion queries to compare with our good phrase detected by our segmentation algorithm. If the good phrase from our segmentation appears in first three suggestion queries, we consider our segmentation as the correct segmentation. Then we compute the accuracy of segmentation algorithms using Equation 4.1. For another example, we have `new ipad 2` as the user input query, the first three suggestions of this query are: `ipad 2 new 16gb`, `ipad 2 32gb new black`, and `apple ipad`. The good phrase generated from our algorithm is `ipad 2` which does not appear in the third suggestion query `apple ipad`. In this case, we regard this segmentation is incorrect.

The query suggestion based evaluation also has some limitations. For instance using this method, only short queries (three or four tokens in one query) are selected as test data set. Query suggestion is a proven solution for short, general, and ambiguous queries. The benefits of query suggestion are limited by difficulties in presentation of long query suggestion list. In E-Commerce suggestion system, we observe that the system doesn't provide the suggestion queries if user input text contain more than 5 words. Also note that, the number of suggestion queries provided by suggestion system is various by queries. For query `iphone 4`, the system lists 5 most related queries. But for `ipad 2`, 6 alternatives are suggested by system. To ease the complexity of computation, we extract the first three suggest queries to compare with our detected good phrase, which doesn't influence the precision of evaluation accuracy.

4.3 Experiments

4.3.1 Data

Our testing queries are taken from eBay CATMAN software on the eBay research lab website which is the same place we fetch our training queries. The eBay CATMAN is a list of highly popular terms that people search for on eBay and each eBay Keyword is shown in sections by eBay category and date. Our original training data including the ones from CATMAN and their related suggestions from eBay suggestion system have almost half million queries. Now, we randomly select 5000 short queries (three tokens in one query) and 5000 long queries (more than three tokens). Then, we execute the queries in eBay search engine and record its retrieval count as query frequency. The query frequency value used for training is based on historical product titles matching result. Now, we fetch the search count again to match recent month product titles. For an example, the frequency of phrase `iphone 4` in our training data set is 384274 only half of current search count 810243. Supply agents update the products title frequently to meet the user demands. We mainly assume that the demand from user is the major source of information to product supply agents. This assumption holds well for E-Commerce domain.

4.3.2 Results

In the experiments, we use two classes of queries: short queries, and long queries. Short queries are up to 3 tokens per query. The long queries have more than 3 tokens per query. Figure 4.2 illustrates the segmentation accuracy for the two data sets we experimented with. The overall performance of short queries is better than that of long queries and evaluation based on phrase retrieval count has comparable accuracy for both classes of queries.

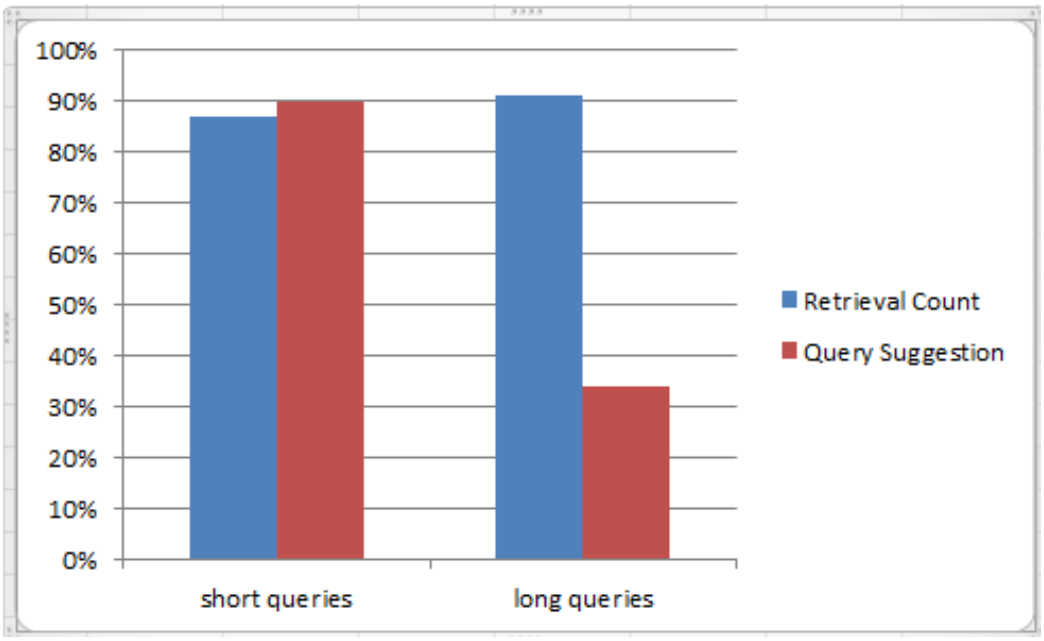

Figure 4.2. Segmentation Accuracy for Different Data Sets And Methods

Through error analysis, we found that the training data has an influence on the accuracy of segmentation when we use phrase retrieval count evaluation method. Our training data is from eBay Catman, a list of highly popular terms that people search for on eBay. Different people have their own naming habit when searching the same product. For example, customer wants to search `iphone 4 case`. Majority of the users would like to search `iphone 4 case` instead of `apple iphone 4 case`, which causes mutual information for `iphone 4` much bigger than `apple iphone` calculated from our data dictionary. However, in our retrieval count evaluation method, `apple iphone` phrase-query will retrieve more product results because of more related products matched in the supply repository. `apple iphone` could contain all generations of `apple iphone` products and its related accessories. There are two reasons for the segmentation failure using query suggestion evaluation method. First, as we discussed in query suggestion evaluation method, query suggestion is a proven solution for short, general queries, which explains the bad performance for long class queries. We also note that most short queries are general, ambiguous queries whose suggestions don't

have semantic consistency with the original queries. For example for query `iphone 4 unlocked`, one of its top suggestions is `iphone 3gs unlocked`, but the customer wants to explore all unlocked items with iphone 4 product. In the following, we will mainly show the experimental results for the shorter class queries using evaluation based on phrase retrieval count to compare the performance of different segmentation algorithms.

The result in Figure 4.3 shows that our hybrid segmentation approach has much higher query accuracy than Wiki segmentation using alone (increases from 34% to 91%). This is because a lot of user phrases are not regarded as a valid Wikipedia article title, which are used only for product search. Also note that, our hybrid segmentation has better performance than mutual Information method combined with relative frequent count.

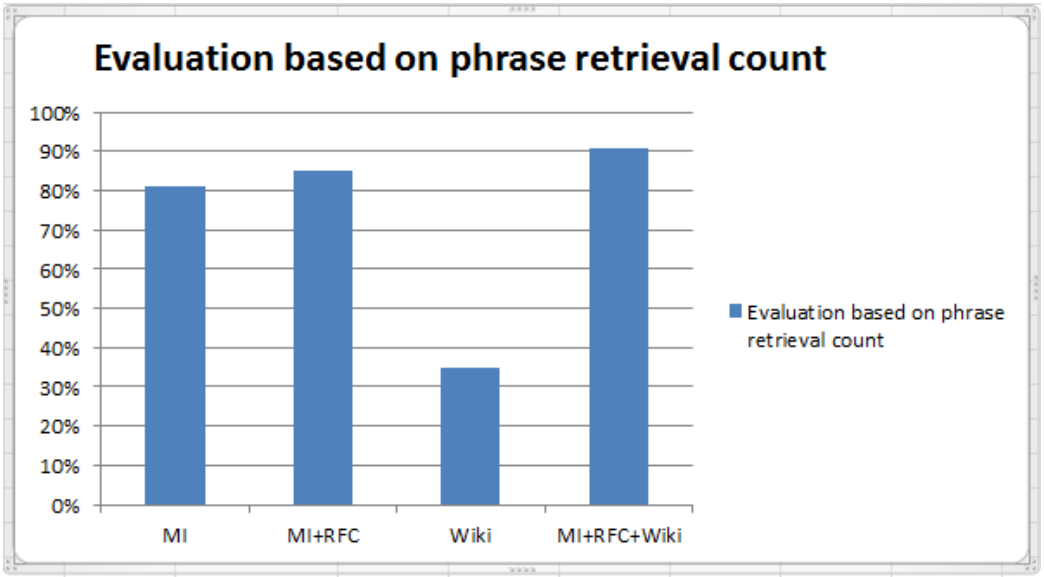

Figure 4.3. Segmentation Accuracy for Different Algorithms

We also found that different segmentation standards assumed by the data set will greatly affect the segmentation results. For example, it is not easy for human to judge

whether the phase `apple laptop computers` should be treated as `"apple laptop" computers` or `apple "laptop computers"`. Such different standards are totally depend on what user's intention is. It is impossible to have a perfect segment without knowing the sematic background.

5. SUMMARY

Query segmentation system is an important part of any E-Commerce site which helps query substitution and query suggestion work. In this thesis, we propose a hybrid method which combines mutual information and relative frequency number as statistic method, prefix tree to build dictionary, maximum matching to enumerate phrases, and Wikipedia titles to solve new words problem in E-Commerce business. Our application use three-stage word segmentation. First, a dictionary-based method is applied to the input text in order to divide the text into as many recognized segments (words) as possible, resulting in a partially segmented text. Next, mutual information which is best method for measuring words association and relative frequency count prune away illegal words. The remaining undecided words of the text are then submitted to Wikipedia to detect unknown words.

From the experiment, we found that there are some points we could improve for the future work.

- Time to build and search the dictionary
- Data clean and query moralization for the training data
- Index all the Wikipedia titles in memory

Currently, much time is spent on offline processing to build the prefix tree dictionary. Nowadays, processing large volume data requires a scalable distributed environment and we could use hadoop based distributed cluster environment such as hadoop distributed file system (HDFS) and mapReduce functions. Majority of the tasks for building the data dictionary and mutual information calculation could be push to the Hadoop cluster.

Currently our training data has not been cleaned or normalized. For example, the phrase ("iphone 4 new" →"new iphone 4" and "man shirts"→"men shirts"). Spelling corrections and synonym mapping ("super man"→ "superman") need to transformed before building the tree [10]. Because the constructer of the tree is highly depend on the words sequence and expression.

In our implementation, we send the new words and undecided word to Wikipedia website which step need networking access. We could consider indexing all the Wikipedia title words in to memory which, I believe, could greatly improve the whole segmentation time.

LIST OF REFERENCES

LIST OF REFERENCES

[1] S. Bergsma and Q. Wang. Learning Noun Phrase Query Segmentation. In Processings of EMNLP-CoNLL 2007, pages 819-826,2007.

[2] Anitha Kannan, Inmar E. Givoni, Rakesh Agrawal, Ariel Fuxman. Microsoft Research Technical Report, August 2010.

[3] J. Huang, J. Gao, J. Miao, X. Li, K. Wang, and F. Behr. Exploring Web Scale Language Models for Search Query Processing. In M. Rappa, P. Jones, J. Freire, and S.Chakrabarti, editors, Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, pages 451-460, 2010.

[4] X. Yu and H. Shi. Query segmentation using conditional random fields. In Proc. of KEYS, pages 21-26, 2009.

[5] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In ICML, pages 282-289, 2001.

[6] K. M. Risvik, T. Mikolajewski, and P. Boros. Query segmentation for web search. In Proc. of WWW 2003(Poster).

[7] David Pinto, Andrew McCallum, Xing Wei, and W. Bruce Croft. Table extraction using conditional random fields. In SIGIR, pages 235-242, 2003.

[8] C. Zhang, N. Sun, X. Hu, T. Huang, and T.-S. Chua. Query Segmentation based on Eigenspace Similarity. In Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the Fourth International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, pages 185-188, 2009.

[9] Wong, Pak-kwong, Chan. Chinese word segmentation based on maximum mathching and word binding force. Proceedings of the 16th International Conference on Computational Linguistics, 1996.

[10] Mohammad Al Hasan, Nish Parikh, Gyanit Singh, and Neel Sundaresan, Query Suggeston for E-commerce Sites, Hong Kong,2011.

[11] J. Guo, G. Xu, H. Li, and X. Cheng. A Unified and Discriminative Model for Query Refinement. In S. Myaeng, D. Oard, F. Sebastiani, T. Chua, and M. Leong, editors, Proceedings of the 31st Anuual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2008, Singapore, July 20-24, pages 379-386, 2008.

[12] B. Tan and F. Peng. Unsupervised query segmentation using generative language models and Wikipedia. In Processing of the 17th International Conference on World Wide Web, pages 347-356, 2008.

[13] R. Jones, B. Rey, O. Madani, and W. Greiner. Generating query substitutions. In proceedings of 15th International World Wide Web Conference, pages 387-396, 2006.

[14] M. R. Brent and T. A. Cartwright. Distributional regularity and phonotactic constrains and useful for segmentation, 1996.

[15] Fuchun Peng, Dale Schuurmans. Self-Supervised Chinese Word Segmentation, IDA, pages 238-247, 2001.

[16] R. Sproat, W. Gale, C. Shih, and N. Chang. Astochastic Finite-state Word Segmentation Algorithm for Chinese. Computational Linguistics, 1996.

[17] Zimin Wu, Gwyneth Tseng. Chinese text segmentation for text retrieval achievements and problems, 1993

[18] P. Ken Q, and Y. Xiaohui. Keyword query cleaning. Proc. VLDB Endow., 2008.

[19] M. Hagen, M. Potthast, B. Stein, and C. Brautigam. The power of Nave Query segmentation. In F. Crestani, S. Marchand-Maillet, H. -H. Chen, E. N. Efthimiadis, and J. Savoy, editors Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, Geneva, Switzerland, July 19-23, pages 797-798, 2010.

[20] X. Li, Y. Wang, and A. Acero. Extracting structured information from user queries with semi-supervised conditional random fields. In Proc. of SIGIR, 2009.

[21] T. Brants and A. Franz.Web 1T 5-gram Version 1.Linguistic Data Consortium LDC2006T13, 2006.

[22] J. Kiseleva, Q. Guo, E. Agichtein, D. Billsus and W. Chai. Unsupervised Query Segmentation Using Click Data: Preliminary Results. In Proc of the 19th International Conference on World Wide Web, pages 1131-1132,2010.

[23] Aitao Chen et al. Chinese text retrieval without using a dictionary. In proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval, pages 42-49, 1997.

[24] R. Baeza-Yates, C.Hurtado, and M. Mendoza. Query recommendation using query logs in search engine. In Proc. of EDBT Workshops, 2004.

[25] P. Boldi, F. Bonchi, C. Castillo, D. Donato, and S. Vigna. Query suggestions using query-flow graphs. In WSCD 09: Processing of the 2009 workshop on Web Search Click Data, pages 56-65, 2009.

[26] K. Y. Su, M. W. Wu, and J. S. Chang. "A Corpus-based Approach to Automatic Compound Extraction", Proc. ACL, New Mexico, USA, June 27-30, pages 242-247, 1994.

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

[28] K. W. Church and P. Hanks. Word Association Norms, Mutual Information, and Lexicography, Computational Linguistic, pp. 22-29 Mar. 1990.

[29] K. Y. Su, Y. L. Hsu and C. Saillard. Constructing a phase Structure Grammar by Incorporating Linguistic Knowledge and Satistical Log-Likelihood Ratio in processing of ROCLING IV, Kentin, Taiwan, pp. 257-275, Aug. 18-20, 1991.

[30] Aitao Chen, Jianzhang He,Liangjie Xu, Fredric C. Gey, and Jason Meggs. Chinese Text Retrieval Without Using a Dictionary. In Proceedings of the 20th Annual International ACM SIGIR Conference, pages 42-49, 1997.

[31] K. J. Chen and M. H. Bai. Unknown word Detection for Chinese by a Corpus-based Learning Method. Computational Linguistic and Chinese Language Process, 3(1):27-44, 1998.

[32] Fuchun Peng, Fangfang Feng, and A. McCallum, Chinese Segmentation and new word detection using Conditional Random Fields, Proceeding of the 20th International Conference on Computational Linguistic, University of Geneva, Switzerland, pp.562-268, 2004.

[33] M. Bendersky and W. B. Croft. Analysis of long queries in a large scale search log. In workshop of WSDM09, pages 8-14, Barcelona, Spain, 2009.

[34] D. Downey, S. Dumais, D. Liebling, and E. Horvitz. Understanding the relationship between searchers' queries and information goals. In Proceedings of CIKM, pages 449-458, 2008.