2017

Smart Image Search System Using Personalized Semantic Search Method

Fangyu Zhang
South Dakota State University

SMART IMAGE SEARCH SYSTEM USING PERSONALIZED SEMANTIC SEARCH METHOD

BY

FANGYU ZHANG

A thesis submitted in partial fulfillment of the requirements for the

Master of Science

Major in Computer Science

South Dakota State University

2017

SMART IMAGE SEARCH SYSTEM USING PERSONALIZED SEMANTIC SEARCH METHOD

This thesis is approved as a creditable and independent investigation by a candidate for the Master of Science in Computer Science degree and is acceptable for meeting the thesis requirements for this degree. Acceptance of this does not imply that the conclusions reached by the candidate are necessarily the conclusions of the major department.

Yi Liu, PhD
Thesis Advisor Date

Sung Y. Shin, PhD
Major Advisor Date

Steven Hietpas, PhD
Head, Department of EECS Date

Kinchel C. Doerner, PhD
Dean, Graduate School Date

ACKNOWLEDGEMENTS

I would like to express my special appreciation and thanks to my advisor Dr. Yi Liu, you have been a tremendous mentor for me. Without your incredible patience and timely wisdom and counsel, my thesis work would have been a frustrating and overwhelming pursuit. Your advisor on both research as well as on my career has been priceless.

I would also like to thank Dr. Shin as my major advisor. Your encouragement, insightful comments, and hard questions improve a lot to my research. Thanks for your guidance on my research and career.

Thanks also go to Dr. Min. Thanks for helping me with my coursework and academic research during my graduate years at South Dakota State University.

Also, I would like to thank Dr. Vance Ovens for agreeing to be on my thesis committee.

A special thanks to my parents, my mother, Xinxia Li, my father, Xiaopeng Zhang. Words cannot express how grateful I am to my parents. Your great love and support mean everything in the moments when there was no one to answer my queries.

CONTENTS

LIST OF FIGURES

ABSTRACT

SMART IMAGE SEARCH SYSTEM USING PERSONALIZED SEMANTIC SEARCH METHOD

FANGYU ZHANG

2017

Due to the emerge in huge numbers of information on the internet nowadays, search technologies are widely used in various fields. Achieving the most relevant search result for the users becomes a big challenge now. While the traditional semantic search technologies seem to achieve the most relevant search result, however, it faces two problems: one is the one-size-fits-all problem, and another is low efficiency. The purpose of this research is to build a Smart Image Search System by using the personalized semantic search method to solve those problems. The personalized semantic search method makes the search system avoids the one-size-fits-all issue, and increase the efficiency.

In the Smart Image Search System, the personalized semantic search method provides users three options to search. They are non-option search, general-option search, and private-option search. Each option search has its specific user needs to achieve the most relevant results. Those options are adopted to solve the one-size-fits-all problem. Also, based on the idea of semantic context concept, the personalized semantic method uses two approaches to increase the search efficiency. First, it applies Apache OpenNLP Library to avoid useless words. Second, it considers the searchers' actions such as click and feedbacks to affect the associated words and associated weight. The Smart Image Search System uses the associated words and associated weight to calculate the relativity

for the search results. This approach makes the Smart Image Search System becomes a self-improved system.

Smart Image Search System is implemented based on the presented methodology and design. As a result of current research on semantic search technologies, we conclude that the Smart Image Search System can avoid useless words, fix the one-size-fits-all problem, and self-improve its relevancy.

Chapter 1. Introduction

1. 1 Motivation

Nowadays, an enormous amount of information is available online. Thus, searching has emerged as a key technology to facilitate users to access information. The traditional search techniques, such as keyword-based search [WEN 2005], do not achieve the most relevant results quickly because of the following two problems.

Problem 1. Low efficiency

Keyword search is one of the most important and widely used technologies in the search field. Because some keywords play a major role in describing users' needs and some of them are irrelevant in searching, there is an enormous challenge of how to better rank the search results. By using the keyword search, there are 17.7 common steps of a search task session [White 2007], which means that a user needs to browse 17.7 web pages before finishing his search task. Keyword search returns results that include many non-relevant items [Wikipedia 2016_1], which means users have to spend a lot of time and effort finding their target results.

Problem 2. One-size-fits-all problem

Different users may have different priorities with the same search queries due to various personal interests. For example, if several users search for "malaria" on a keyword-based search engine, the keyword search engine will show the same sequential results about malaria. However, the users may focuses on different perspectives of

“malaria”, for example, some people may be interested in the malaria distribution while some people may wish to find out about malaria prevention. But keyword search technology won’t be able to differentiate different users’ interests based on the same phrase. Thus, keyword search technology adopts a "one-size-fits-all" strategy and does not consider the user’s personal interests.

Due to these two problems, it is necessary to have a search system not only considers the relevance of search result but also the user’s personal interests. Such search system can help users save time and effort to complete search tasks.

Semantic search technology [Dong 2008] is adopted in this research to address the two problems with search queries. It seeks to improve search accuracy by analyzing the user’s intent and analyzing their interests to find the most relevant results.

1.2 Goal

The purpose of this research is to build a Smart Image Search System by using the personalized semantic search method. The Smart Image Search System achieves the following objectives:

Objective 1. Consider the user's personal interests to avoid the one-size-fits-all problem.

Objective 2. Avoid stop words [Wikipedia 2017_1] not only in search queries, but also in snippets; this will save a considerable amount of time without dealing with nonrelative search queries.

Objective 3. The Smart Image Search System can self-improve relevancy.

To achieve the aforesaid objectives, we design a Smart Image Search System that follows the procedures in Figure 1.1.

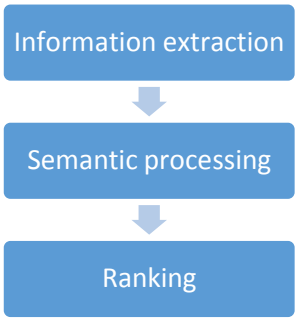


Figure 1.1 Semantic search procedures

Information extraction

Information extraction is for filtering stop words. This process helps increase search efficiency.

Semantic processing

Semantic processing focuses on evaluating the keywords with weights. This process helps obtain the most relevant results to reduce the searcher's searching effort. Thus, the searcher can easily find the target result without browsing several web pages.

Ranking

Ranking sorts the relevant results from high to low relativity. The relativity is calculated by the weight of each useful attribute.

1.3 Thesis Organization

The thesis presents the search system with the personalized semantic method to handle the two problems with searching. The personalized semantic search method focuses on user's personal interest to achieve the most relevant result. The rest of the thesis is organized as follows.

Chapter 2 presents the background information used in the development of the semantic search engine. It includes the definition of semantic search and the methodological concepts that are utilized in the latest semantic search technology.

Chapter 3 presents the key methodology that is designed in this research. It focuses on improving the traditional semantic search method.

Chapter 4 uses a Smart Image Search System as a case study to demonstrate the presented design and methodology. This chapter includes the design and implementation of the Smart Image Search System.

Chapter 5 compares the personalized semantic search method used in this research with the traditional semantic search method and evaluates the semantic search method by using the Smart Image Search System based on the objectives stated in Chapter 1.

Chapter 6 concludes the work and proposes further possible enhancement to the Smart Image Search System.

Chapter 2. Background

Semantic search technology is adopted in this research to build a system that can efficiently achieve the most relevant results.

2.1 Semantic Search

Unlike common search algorithms, such as keyword search [Rahman 2013], semantic search is a context-based search technology. It uses the principles of natural language to determine what users search for. Usually, location, the synonyms of a term, current trends, word variations, and other natural language elements as part of the search are important aspects to be incorporated in semantic search [Techopedia 2017].

The semantic search method contains a natural language processing, programs for semantic analysis, and a general problem-solving system [Winograd 1972].

2.1.1 Definition

Semantic search aims to generate more relevant results by understanding users' search intents and the meaning of search queries [Wikipedia 2017_2]. Thus, semantic search is a technology that can evaluate and understand search queries to find the most relevant results. The relevant results can be consulted on a website, database, or any other information repository [Wikipedia 2017_2].

2.1.2 Approaches to semantic search

Recently, many semantic search approaches have been published. Four approaches to semantic search that are widely used in the search field are shown in Figure 2.1 [HLWIKI 2016].

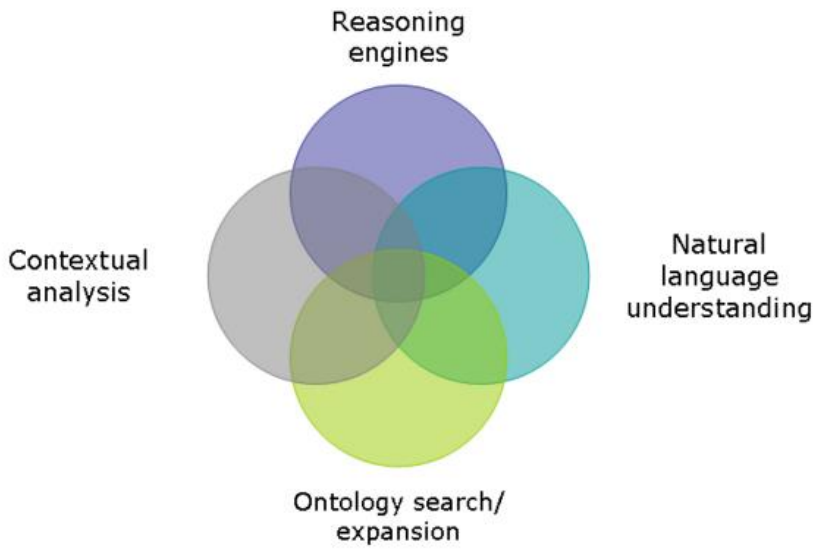


Figure 2.1 Approaches to semantic search

Natural language understanding

The core part of natural language understanding is natural language processing (NLP). NLP is related to the area of human–computer interaction and deals with the application of computational models to processing text or speech data [Wikipedia 2017_3]. NLP will be introduced in detail in Chapter 3.

Ontology search/expansion

To retrieve more relevant documents, search queries need to have an unambiguous context. Query expansion techniques range from relevance feedback mechanisms to the use of knowledge models, such as ontologies, to resolve ambiguities [Bhogal 2007].

The search queries entered by the user matches the ontology classes, and fetches the new related concepts from the ontology. An ontology together with a set of

individual instances of classes constitutes a knowledge base. Ontology classes describe most ontologies in the domain [Zanasi 2007]. For example, “apple” is ambiguous without any information related to its domain. If a user searches for “delicious apple,” then the ontology class is a fruit; if a user searches for “Apple application,” then the ontology class is mobile applications on the iOS platform. The new information is added to a user query to provide better results when searching is performed.

Contextual analysis

Contextual analysis is a method for analyzing the environment in which a business operates [Wikipedia 2017_4]. The contextual analysis helps assess the text in its cultural, historical, or social context. It may also characterize the textuality of the text. Textuality represents a text in a specific way. For example, the physicality of a book is a case of textuality in the print medium. The contextual analysis considers all the circumstances of the production of the text.

Reasoning engines

The reasoning engine is a piece of software that can infer logical consequences from a set of asserted facts or axioms [Wikipedia 2017_5].

2.2 Related work

To create semantic search technology that can self-improve relevancy, we concentrate on the main concepts and algorithms of semantic search nowadays. In this research, the

semantic context [Xu 2014] is the core concept we use. The basic formulation of the semantic context is adopted to improve and implement our semantic search algorithm.

2.2.1 Semantic Context

Considering that semantic search is a data-searching technique in which a search query aims to understand the intent and contextual meaning of the user's search queries, we can understand this technology from its semantic context definition and problem formulation.

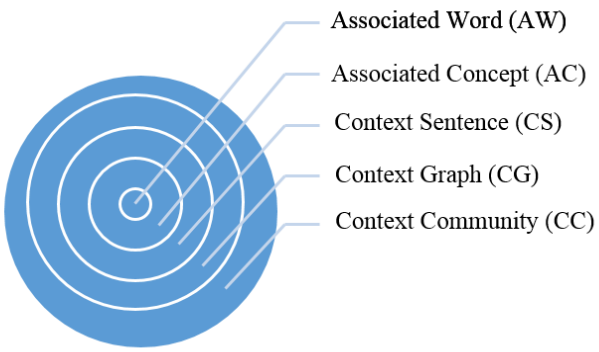


Figure 2.2 Semantic context

The basic definition of semantic context [Xu 2014] is given below:

Let C = $\{W_1, W_2, ..., W_n\}$ be a concept containing a set of words, which can be a query to the web search engine.

Let S = $\{S_1, S_2, ..., S_m\}$ be a set of documents related to the concept.

Associated word (AW)

An associated word [Xu 2014] frequently co-occurs with the concept C, with frequently appears in the document set S. The set of associated words of concepts of concept C is denoted as

$$AW_c = \{aw_1, aw_2, ..., aw_{AWc}\} \quad (1)$$

Associated Concept (AC)

An associated concept [Xu 2014] frequently co-occurs with concept C, which frequently appears in the document set S. The set of associated concepts of concept C is denoted as

$$AC_c = \{ac_c, ac_c, \ldots, ac_{ACc}\} \quad (2)$$

Associated Sentence (AS)

An associated sentence [Xu 2014] contains concept C and a sequence of words, which appears in the document set S. The associated concepts are composed of some associated words. The set of associated concepts of concept C is denoted as

$$CS_c = \{ cs_1, cs_2, \ldots, cs_{CSc} \} \quad (3)$$

Associated Graph (AG)

An associated graph [Xu 2014] is a data structure of the associated words, which reflects an associated relation between associated words.

$$CG_c = \{ N, E \} \quad (4)$$

$$N = AW_c$$

$$E = \{ e_1, e_2, \ldots, e_E \}$$

The edge e_k can be denoted as $< aw_i, aw_j, \lambda >$, which means the edge e_k is from node aw_i to aw_j with weight λ. The computation method of λ will be explained in detail in Section 2.2.2.

Context Community (CC)

A context community [Xu 2014] is a subgraph of the context graph, which reflects part of the context of concept C. The set of context communities of concept C is denoted as

$$CC_c = \{cc_1, cc_2, \ldots, cc_{cc}\} \quad (5)$$

where $\forall aw_i \in \forall cc_i \wedge \forall aw_j \in \forall cc_j \text{ -> } aw_i \neq aw_j$

To understand these contexts much better, an example [Xu 2014] is given below.

Consider a semantic context of the concept "apple", by integrating the annotation features from the dictionary and Google, we have the following associated words with "apple"

AW = {iPhone, iPod, iPad, records, fruit, …}.

Based on the associated words, we can extend the AW to have the following associated concepts:

AC = {iPhone 6s, iPad Air, apple tree, …}

By understanding the apple's AW and AC, we can obtain the following context sentences:

CS_1 = {Apple designs and creates iPhone}

CS_2 = {The apple is a common fruit all over the world}

After classifying each pair of associate words with an edge, we can draw the context graph of "apple" with the following graph information.

N = {iPhone, iPad, Mac, tree, fruit, …}

E= {<iPhone, iPad, λ_1>, <tree, fruit, λ_2 >, …}

By observing and analyzing the context graph of "apple" in Figure 2.3 [Xu 2014], the following three examples of the context communities of "apple" are obtained:

CC_1 = {iPhone, iPad, Mac, iPod, computer, …}

CC_2 = {fruit, tree, rose, …}

CC_3 = {sound, creation, records, …}

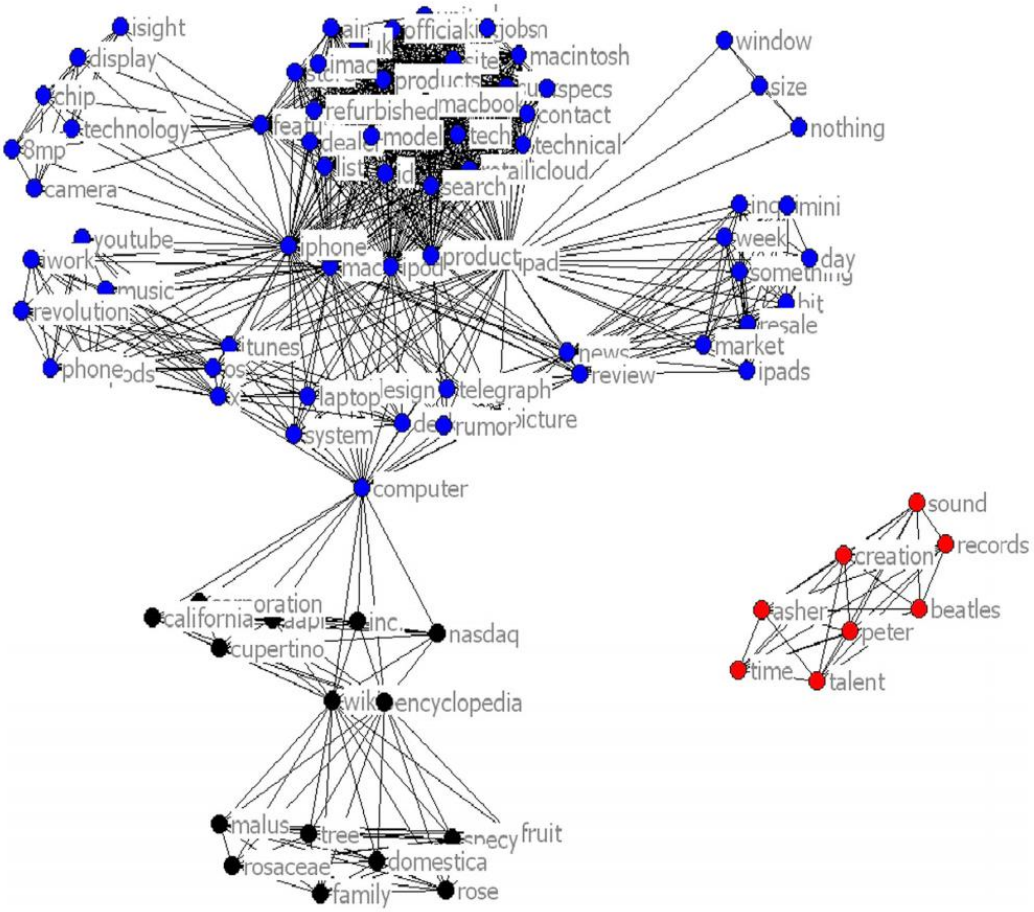


Figure 2.3 Context graph of "apple"

2.2.2 Formulation

A snippet is a brief window of text extracted by a search engine around the query words in a web page [Bollegala 2011]. Suppose the associated words extracted from the snippets set S is

$$\text{AW} = \{aw_1,\ aw_2, \ldots,\ aw_m\}$$

where m is the number of associated words. The appearance of the associated words in the number of n snippets can be represented by a 1-0 matrix. a_{nm} is 1 means the m_{th} word appears in nth snippets.

$$AWS = \begin{pmatrix} a_{11} & \dots & a_{1m} \\ \dots & \dots & \dots \\ a_{n1} & \dots & a_{nm} \end{pmatrix} \tag{6}$$

Some existing research, such as term-frequency-inverse document frequency (tf-idf), computed as the logarithm of the number of the documents in the corpus divided by the number of documents where the specific term appears [Tf-Idf 2017]. Currently, some semantic search methods, such as the temporal semantic context (TSC) method, use snippet frequency to weight a word. The snippet frequency means the appearance frequency of each word in snippets. Thus, the weight of each word is computed as

$$\mathrm{W}(\mathrm{A_i}) = \sum_{j}^{n} a_{ji} / n, a_{ji} \in AWS \tag{7}$$

The snippet frequency of each word is used as the ranking schema to obtain the search result. The higher the snippet frequency is, the more relevant the result is.

Chapter 3. Methodology

This chapter focuses on the methodology that the Smart Image Search System uses to achieve the proposed objectives. In this research, the traditional semantic search is redesigned with the following three improvements.

i. First improvement

The Smart Image Search System provides three options to deal with users' search queries semantically. Each option deals with specific historical data to satisfy users.

ii. Second improvement

Compared with the typical semantic search method, the personalized semantic search method in this research simplifies the process procedures without orderly processing of the semantic context from associated words to the context community. The personalized semantic search method only uses associated words and context sentences to determine relevant results.

iii. Third improvement

Based on the user’s feedback, the Smart Image Search System can self-improve its relevancy of search results. The more feedback is given, the more relevant is achieved.

3.1 Historical data

The Smart Image Search System in the research uses users’ historical search queries in the semantic search. As shown in Figure 3.1, there are three types of historical data for

search queries: non-option historical data, private-option historical data, and general-option historical data. Each option needs to be semantically processed with historical data.

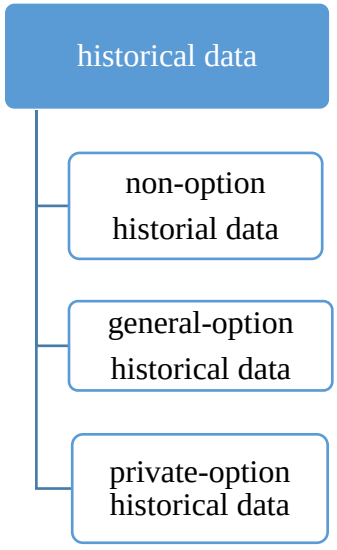


Figure 3.1 Historical data types

3.1.1 Data process options

Non-option historical data is semantically processed without considering the associated attributes. It only uses the extracted attributes from search queries to obtain relevant results. For example, the search queries are “malaria”, thus the relativity for the search result is only calculated by “malaria”.

General-option historical data is used for all users who aim to obtain the most popular search results. The general search results indicate search trends and are considered as a good reference with high relevancy for users. For example, the search queries are “malaria”, thus the relativity for the search result is calculated not only by “malaria” but

also by the relevant words of “malaria” such as “Africa” and “map”. The relevant words, “malaria” and “map”, are obtained by all the users’ search behaviors.

Private-option historical data is only considered when a user logs into the Smart Image Search System and selects the private option. The Smart Image Search System processes the user’s current search queries semantically by referencing the user’s historical search queries. For example, the search queries are “malaria”, thus the relativity for the search result is calculated not only by “malaria” but also by the relevant words of “malaria” such as “Asia” and “map” for the private user. The relevant words, “Asia” and “map”, are obtained by the private user’s search behaviors. In this example, even most of users are interested in “Africa”, the private user is more interested in “Asia”.

The Smart Image Search System uses the three types of historical data for search queries to address problem 2 (“one-size-fits-all”) stated in Chapter 1. The users have three options in searching: (1) using the non-option historical data of search queries as a reference to obtain the most relevant results; (2) using general-option historical data to generate the most popular results by using all users’ historical data as a reference; and (3) using the user’s private-option historical data to find the user’s specific interest results.

However, this strategy faces an issue when the user uses the Smart Image Search System for the first time since there is no historical data in the database for the reference. This issue makes the Smart Image Search System the same as the keyword search engine at the very beginning. Thus, we need to make a base database to store the most relevant snippet attributes to enable the Smart Image Search System to search semantically, even in the first-time usage.

Google Trends [Wikipedia 2016_2] is a powerful tool for analyzing all the searches from Google and shows a list of the most popular search queries related to the search item. The Smart Image Search System adopts Google Trends to build the base database.

Google Trends shows the frequency at which "a search term is searched for relative to the total number of searches". For instance, the results for searching for "malaria" using Google Trends are shown in Figure 3.2.

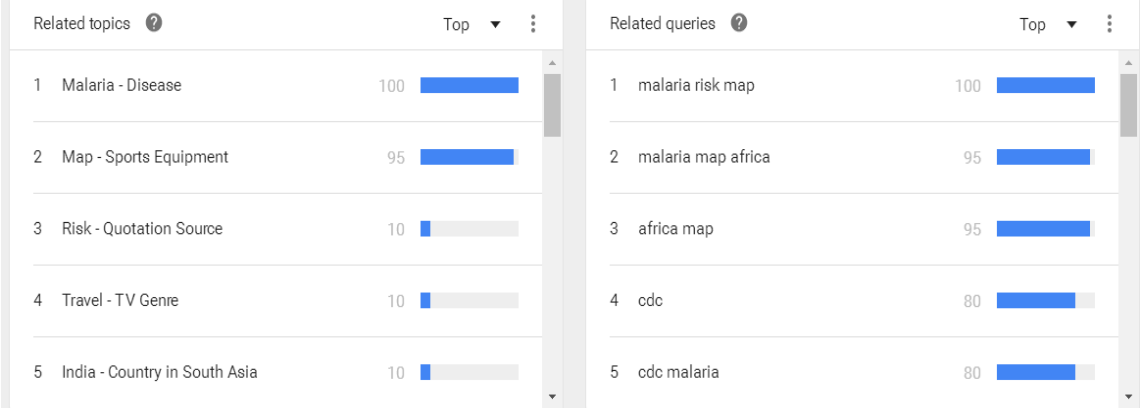


Figure 3.2 Top snippets of malaria-related searches

As shown in the figure, the results of searching for "malaria map" using Google Trends show highly relevant searches and "malaria risk map" is the most relevant topic for malaria map.

After searching for a search term, such as "malaria map," the Smart Image Search System collects the information as primary attributes in this research to build the base database.

3.2 Semantic search

The Smart Image Search System has three core components, Information extraction, Semantic processing, and Ranking, as shown in Figure 3.3.

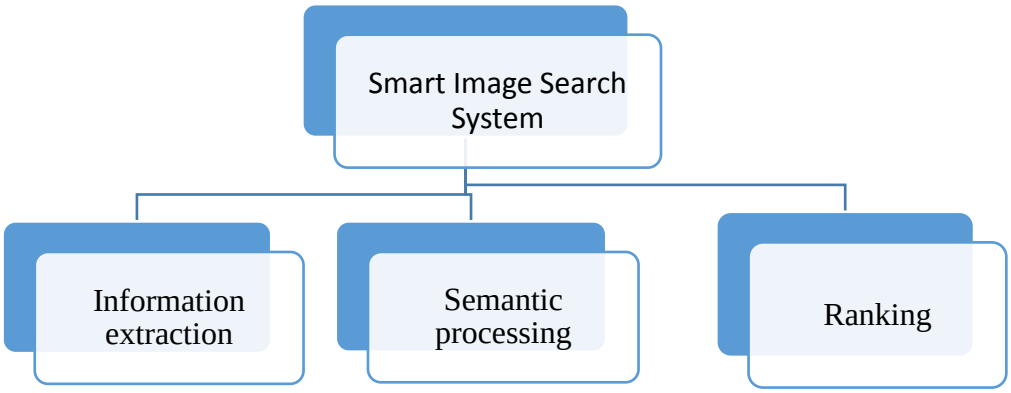


Figure 3.3 Three components of the Smart Image Search System

3.2.1 Information Extraction

Information Extraction is the first component for processing the user's search queries. Each query is processed with the two procedure: Natural language parsing, and Syntax extraction

3.2.1.1 Natural language processing

Natural language processing is a method to associate language with computational linguistics. It processes sentences in a natural language, such as English, to obtain a better understanding of the user's search queries.

In this research, the Smart Image Search System uses the Apache OpenNLP library to perform natural language processing. The Apache OpenNLP library is a machine

learning-based toolkit for processing natural language text. It includes the following six modules for processing:

Sentence Detector

Sentence Detector is used for detecting sentence boundaries such as period sign, and it returns an array of strings.

Tokenizer

Tokenizer is used to recognize each token. The token is usually a word that is separated by a whitespace.

Name Finder

Name Finder can recognize names from the context. For example, “Mary got involved in malaria work”. In the example, the Name Finder returns “Mary” as the result.

POS Tagger

The process of assigning one of the parts of speech to the given word is called Parts of Speech (POS) Tagging [Wikipedia 2017_6]. POS Tagger is a module for handling POS tagging. POS include nouns, pronouns, verbs, adverbs, adjectives, conjunctions, and their sub-categories. Figure 3.4 [Upenn 2017] shows all of the POS tags in detail.

Number	Tag	Description
1.	CC	Coordinating conjunction
2.	CD	Cardinal number
3.	DT	Determiner
4.	EX	Existential *there*
5.	FW	Foreign word
6.	IN	Preposition or subordinating conjunction
7.	JJ	Adjective
8.	JJR	Adjective, comparative
9.	JJS	Adjective, superlative
10.	LS	List item marker
11.	MD	Modal
12.	NN	Noun, singular or mass
13.	NNS	Noun, plural
14.	NNP	Proper noun, singular
15.	NNPS	Proper noun, plural
16.	PDT	Predeterminer
17.	POS	Possessive ending
18.	PRP	Personal pronoun
19.	PRP$	Possessive pronoun
20.	RB	Adverb
21.	RBR	Adverb, comparative
22.	RBS	Adverb, superlative
23.	RP	Particle
24.	SYM	Symbol
25.	TO	*to*
26.	UH	Interjection
27.	VB	Verb, base form
28.	VBD	Verb, past tense
29.	VBG	Verb, gerund or present participle
30.	VBN	Verb, past participle
31.	VBP	Verb, non-3rd person singular present
32.	VBZ	Verb, 3rd person singular present
33.	WDT	Wh-determiner
34.	WP	Wh-pronoun
35.	WP$	Possessive wh-pronoun
36.	WRB	Wh-adverb

Figure 3.4 POS tag

Chunker

Chunker is used to partition a sentence into a set of chunks by using the tokens generated by the Tokenizer.

Parser

Parser is a program, working as a simple compiler, that receives input in the form of sequential source program instructions, interactive online commands, and markup tags and breaks them into parts that can then be managed by another program.

The Smart Image Search System uses Sentence Detector, Tokenizer, and POS Tagger to achieve partially natural language processing. It imports the Apache OpenNLP library [Opennlp 2017], which supports tokenization, sentence segmentation, and POS, to extract attributes in Information Extraction component. The following example shows how Apache OpenNLP library works in the Smart Image Search System:

User input	Malaria is a very common and serious malady.
Sentence Detector output	Malaria is a very common and serious malady.
Tokenizer output	Malaria is a very common and serious malady.
POS Tagger output	Malaria_ NN is_ VBZ a_ DT very_ RB common_ JJ and_ CC serious_ JJ malady. _NNP

Table 3.1 OpenNLP example

3.2.1.2 Syntax Extraction

Syntax Extraction plays a major role in avoiding some stop words during searching to help the Smart Image Search System save time and storage space. Thus, it improves the efficiency of the Smart Image Search System. After the partial natural language processing in the previous step, the Smart Image Search System uses the Stanford Log-Linear Part-Of-Speech Tagger [Stanford 2017] to preserve the noun words as the attributes. The extracted noun words are stored in attributes database as useful attributes, each useful attribute includes the information as shown in table 3.2.

Variable	Data type	Description
AttributeName	String	The attribute name
RelatedAttributesName	String	The related attributes' name
Weight	Float	The ratio of the appearance of attributes
Tag	String	The POS tag for managing the attributes

Table 3.2 Attribute information

Syntax extraction uses the POS tag to filter some stop words, such as "is," "very," "serious," and "at." Figure 3.5 shows an example of information extraction.

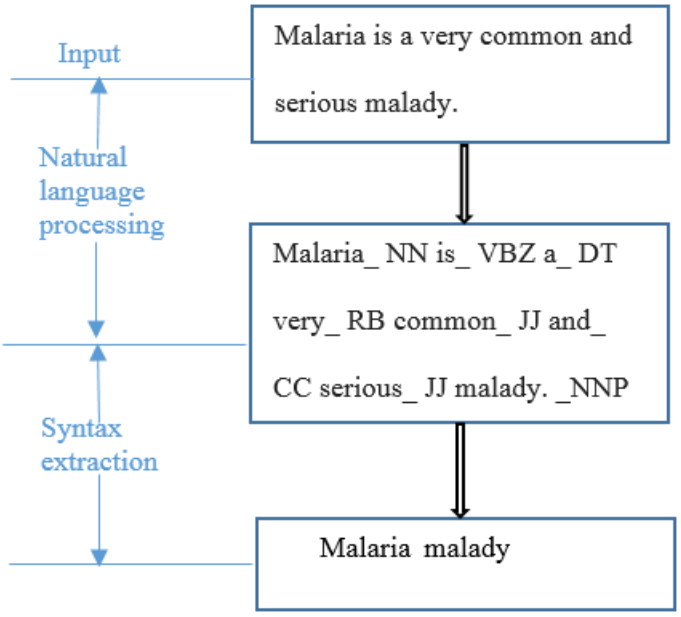


Figure 3.5 Example of information extraction

After searching for "Malaria is a very common and serious malady," the Smart Image Search System applies natural language processing and then uses syntax extraction to extract attributes. The extracted attributes are "malaria" and "malady," as shown in Figure 3.5.

3.2.2 Semantic Processing

A new semantic search method is proposed in this research to make the system to be self-improving.

The semantic processing is applied to attributes to compute their relativity. The relativity indicates the relevance between search queries and snippets. In the research, a new semantic method named personalized semantic search improves the search accuracy

by analyzing users' feedback to impact the associated weight of associated attributes, that is, the personalized semantic search method uses not only the weight of each attribute, but also the associated weight for a pair of associated attributes to calculate the relevancy.

3.2.2.1 Weight

The personalized semantic search method classifies the attributes into two types, extracted attributes and associated attributes, which are used to improve the traditional search technology. Each type has a specific method to compute the weight.

i. Extracted attribute

An attribute that is extracted from the user's search queries is called an extracted attribute. The weight of an extracted attribute should be given to indicate what is the most relevant to the concept. In this thesis, the formula (7) in Chapter 2 is used to compute the weight of each extracted attribute.

ii. Associated attribute

An attribute that is relevant to the extracted attribute is called an associated attribute. The Smart Image Search System uses the co-occurrence ratio of the associated words to compute the associated weight. It applies three weight formulas to calculate the associated weight of the associated attribute based on users' feedback. The three formulas to calculate the associated weight are given below:

The first weight:

$$\text{Old weight} = \frac{\text{Total number of co-occurrences in a pair of associated attributes}}{\text{Total number of occurrences of the associated attributes}}$$

$$\text{Update weight} = \frac{\text{Total number of co-occurrences in a pair of associated attributes+1}}{\text{Total number of occurrences for the associated attributes+1}} \quad (8)$$

The first weight works for the search queries.

The *second weight* has the same weight formula as the first weight. Compare to the first weight, the second weight works for selected search results once the user click the search results. The personalized semantic search method uses second weight to better understand the search queries.

The third weight:

Once the user is satisfied with the search results, the Smart Image Search System will apply the third weight to combine the search queries and snippet queries together for the weight. Figure 3.6 shows how the combination of weights between search attributes and snippet attributes works.

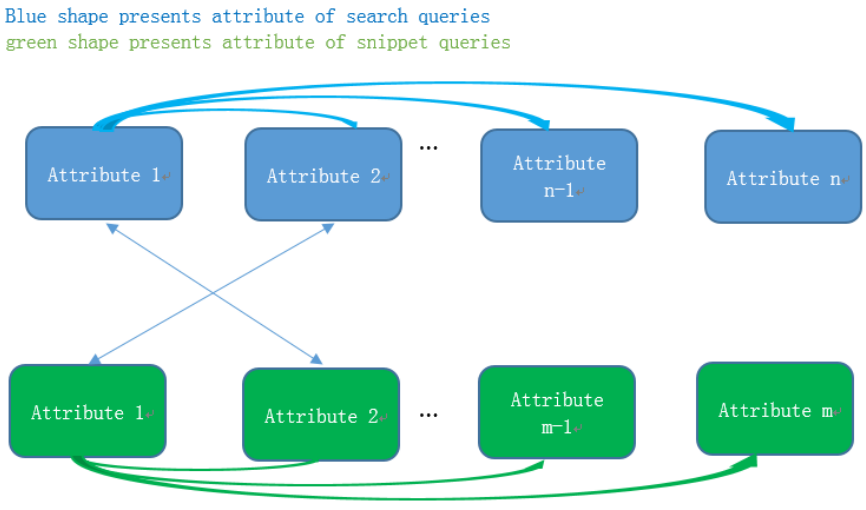


Figure 3.6 Combination of weights

$$\text{Weight update} = \frac{\text{Total number of co-occurrences in a pair of associated attributes+x}}{\text{Total number of occurrences for the associated attributes+x}} \quad (9)$$

Where x = 1 for different attributes,

x = 2 for the same attributes

3.2.2.2 Relativity computation

In this research, each extracted attribute's weight and the associated attribute's associated weight are used to set up a formula to compute the relativity for each search result. Figure 3.7 and Figure 3.8 show the associated weight for each pair of associated attributes is computed.

Figure 3.7 Associated attributes

Assume S1 is a set of weight terms that contains the extracted attribute A_i and S2 is a set of weight terms that contains the associated attribute A_k, the weight term can be either search queries or result queries. Thus, the associated weight between A_i and A_k is denoted as

$$W < A_i, A_k > = \frac{S1 \cap S2}{S1+S2-S1 \cap S2} \quad (10)$$

To compute the associated weight of associated attributes, we build the following associated attribute map:

	A_1	A_2	⋯	A_k	⋯	A_n
A_1	O_{11}	O_{12}	⋯	O_{1k}	⋯	O_{1n}
A_2	O_{21}	O_{22}	⋯	O_{2k}	⋯	O_{2n}
⋮	⋮	⋮	⋱	⋮	⋱	⋮
A_i	O_{i1}	O_{i2}	⋯	O_{ik}	⋯	O_{in}
⋮	⋮	⋮	⋱	⋮	⋱	⋮
A_n	O_{n1}	O_{n2}	⋯	O_{nk}	⋯	O_{nn}

Figure 3.8 Associated attributes map

In Figure 3.8, O_{ik} indicates the co-occurrence times of A_i and A_k. Based on the formula (10) and the associated attribute map, the associated weight is computed as

$$W < A_i, A_k > = \frac{O_{ik}}{O_{ii}+O_{kk}-O_{ik}} \tag{11}$$

In this research, the extracted attribute and its associated attributes are used to set up the following formula to compute the relativity for each search result.

$$\begin{aligned}\text{Relativity} = &\sum_{i}^{n} [(\text{ith} - \text{extracted attribute}).\text{extracted}\, weight^2 * a_{ij} \\ &+ \sum_{k}^{m} (\text{kth} - \text{associated attribute}).\, associated\ \ weight * \text{extracted}\, weight^2 * a_{kj}] \\ = &\sum_{i}^{n} (W(A_i)^2 * a_{ij} + \sum_{k}^{m} W < A_i, A_k > * W(A_k)^2 * a_{kj})\end{aligned} \tag{12}$$

where n is the number of extracted attributes in the search query and a_{kj} shows whether the kth attribute appears in the j_{th} snippets.

If a_{kj}= 0, then the k_{th} attribute never appears in j_{th} snippets.

If a_{kj}=1, then the k_{th} attribute appears in j_{th} snippets.

3.2.3 Ranking

The Smart Image Search System uses the relativity of each search result as the ranking schema. In this research, the new semantic search method simplifies the process for the semantic context. Compared with the traditional semantic method, which adopts the spinning tree [Seppänen 1970] to determine the most relevant result, the personalized semantic search method uses the formula (12) to calculate the relativity, stores the relativity in the snippet database, and uses the database in-order clause as shown in Figure 3.9 to sort the results by the value of relativity.

```
SELECT * FROM t1
  ORDER BY key_part1 DESC, key_part2 DESC;
```

Figure 3.9 Database sort clause

By using the in-order clause, the Smart Image Search System ranks the results from high to low relativity with a lower complexity than the traditional semantic search.

Chapter 4. Case study

The personalized semantic search method is applied to the Smart Image Search System for the case study. Smart Image Search System adopts the personalized semantic search method to find the most relevant Image. In this case study, the Smart Image Search System represents the characteristics of the Smart Image Search System, such as avoiding stop words, fixing the one-size-fits-all problem, and self-improving the relevance of the search results.

4.1 The Design of the Smart Image Search System

The Smart Image Search System uses the presented semantic search method in Chapter 3 to achieve the most relevant image. It semantically processes the users' search queries with three options: the non-option, the general option, and the private option.

4.1.1 Design Decisions

The following design decisions are made based on the requirements of making the Smart Image Search System achieve the three objectives stated in Chapter 1. The Smart Image Search System is designed and developed based on the following design decisions.

Design decision 1: All the searchers' inputs will be processed with three major processing procedures: the information extraction subsystem, the semantic processing subsystem, and the ranking subsystem.

Design decision 2: The user's search queries can be processed with three options in searching: using the non-option historical data of search queries as a reference to obtain the most relevant results; using the user's private-option historical data to find the user's specific interest results; and using the general-option historical data to generate the most popular results by using all users' historical data as a reference.

Design decision 3: The Smart Image Search System is used for searching images, and it supports the JEPG, PDF, and PNG image types. Each image is stored in the database named image with its title, description, image Uniform Resource Locator (URL), and relativity.

4.1.2 The System Design of the Smart Image Search System

As shown in Figure 4.1, the system is composed of three components: Information Extraction, Semantic Processing, and Ranking.

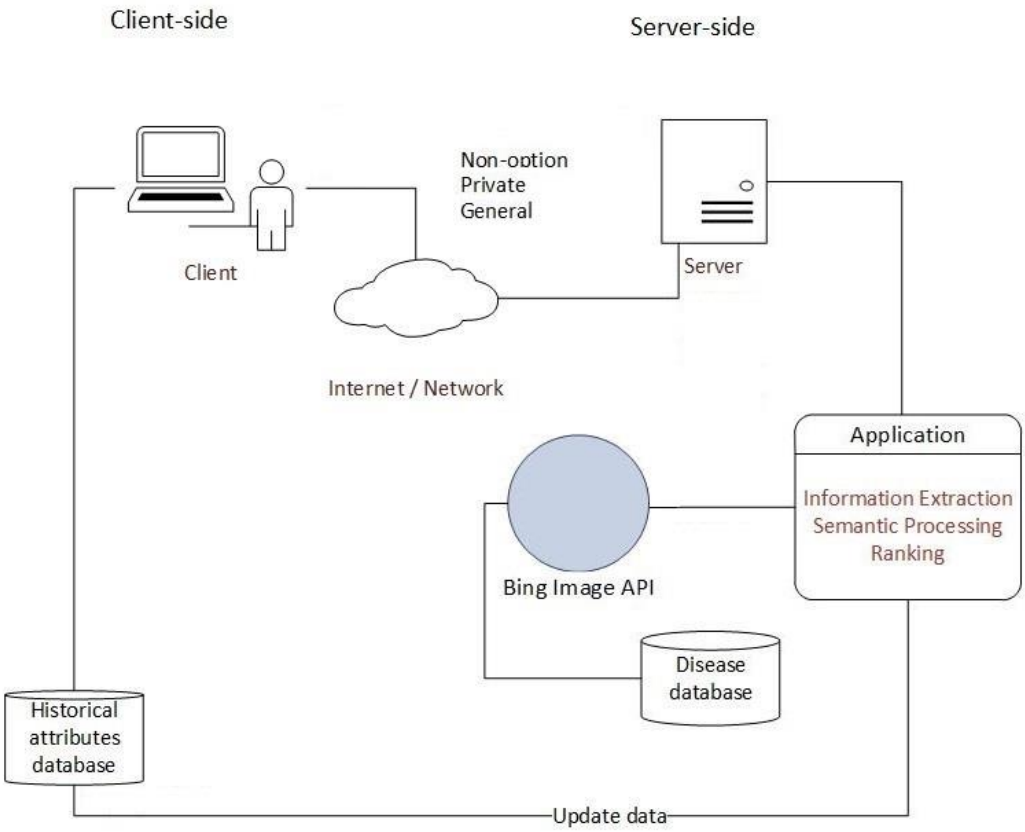


Figure 4.1 Smart Image Search System overview

The Smart Image Search System addresses design decision 1 by mapping the three major processing procedures in three components, as shown in application in Figure 4.1.

To address design decision 2, the Smart Image Search System provides three options to process historical data, as shown in Figure 4.1, which are the non-option, private historical data option, and general historical data option.

4.1.2.1 The Design of the Information Extraction Component

Information Extraction is used for extracting useful attributes from the user's input. The Information Extraction component is composed of three models of the Apache

OpenNLP library and one syntax extraction module. Figure 4.2 shows the architecture of this component.

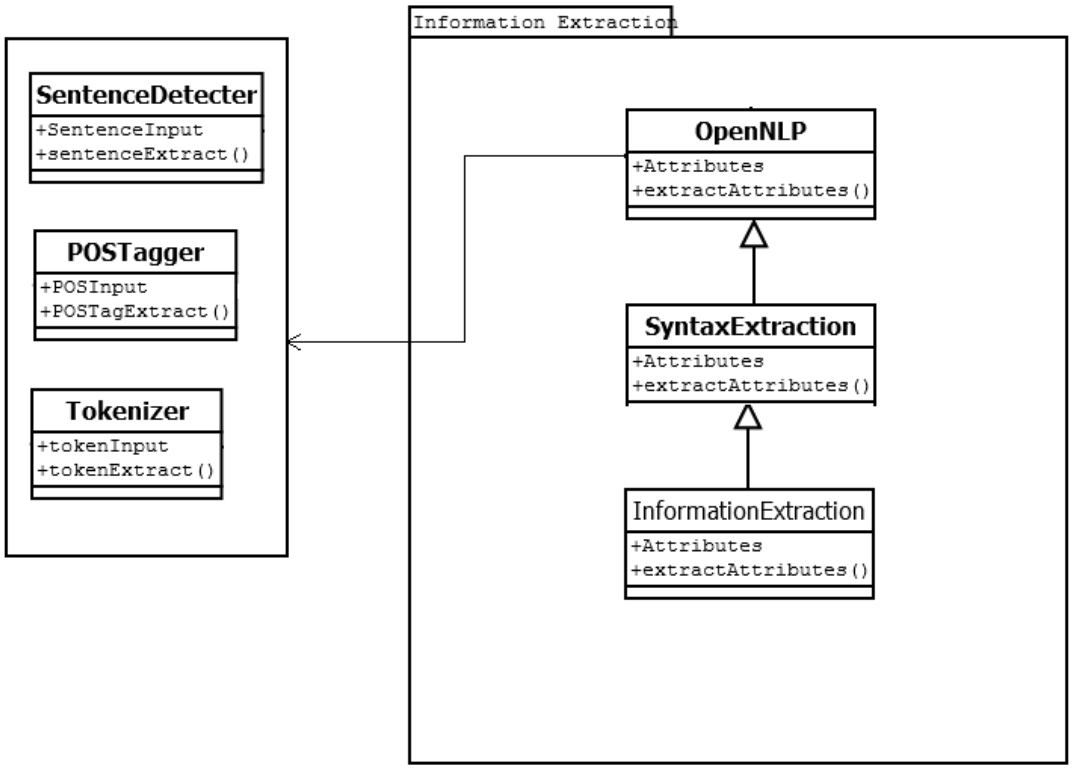


Figure 4.2. Information Extraction

The proposed methodologies in Section 3.2.1 on extracting useful information as attributes are applied to this component. The operation extractAttributes() in the class OpenNLP is used for parsing each word and classifies them with its tag. The Smart Image Search System uses the operation extractAttributes() in the class SyntaxExtraction to extract noun words as a useful attribute.

4.1.2.2 Design of the Semantic Processing Component

Semantic Processing plays a core role in the Smart Image Search System. The component Semantic Processing is responsible for processing the associated attributes with their associated weight. To avoid the one-size-fits-all problem, the Smart Image Search System takes the searcher's private feedback in the Semantic Processing component to obtain the most relevant search results for the searcher. Figure 4.3 shows the workflow of Semantic Processing.

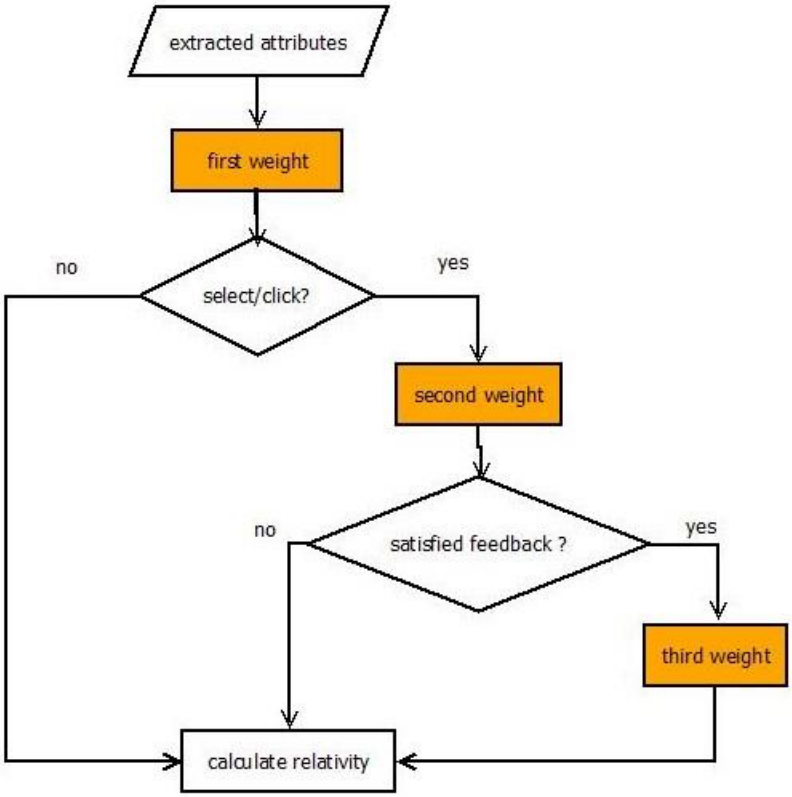


Figure 4.3 Workflow of Semantic Processing

As the workflow shows, the attributes are weighted at least once and at most three times. The component uses the value of the associated weight to calculate the relativity,

which takes the relevant attributes into consideration. For example, if a user searches for "malaria," as the word "Africa" is relevant to malaria, the relativity is calculated by not only the weight of "malaria," but also the associated weight between "Africa" and "malaria." Figure 4.4 shows the main methods of the component Semantic Processing.

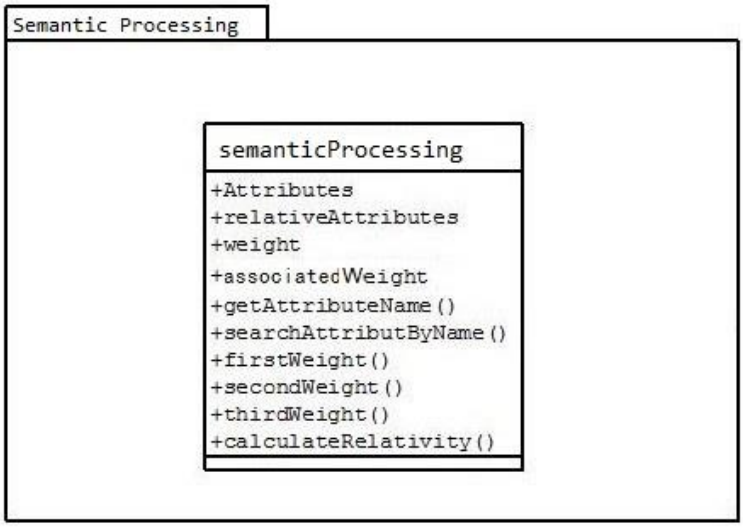


Figure 4.4 Semantic Processing

Semantic Processing provides three operations for weighting: Operation firstWeight(), which evaluates the weight for all the extracted attributes in search queries; operation secondWeight(), which is only applied when the user selects specific results; and operation thirdWeight(), which is only applied when the searcher marks that they are satisfied via the feedback. The operations secondWeight() and thirdWeight() make the Semantic Processing achieve the objectives not only by considering the user's personal interests to avoid the one-size-fits-all problem, but also by calculating the relativity based on the personalized semantic search method. The operations getAttributeName() and

searchAttributesByName() help obtain the relative attributes of the current attribute. Operation calculateRelativity() uses the current attributes' weight and the relevant attributes' associated weight to calculate the relativity.

4.1.2.3 Design of the Ranking Component

The component Ranking is responsible for displaying the search results from high to low relativity. Figure 4.5 shows the classes in the component.

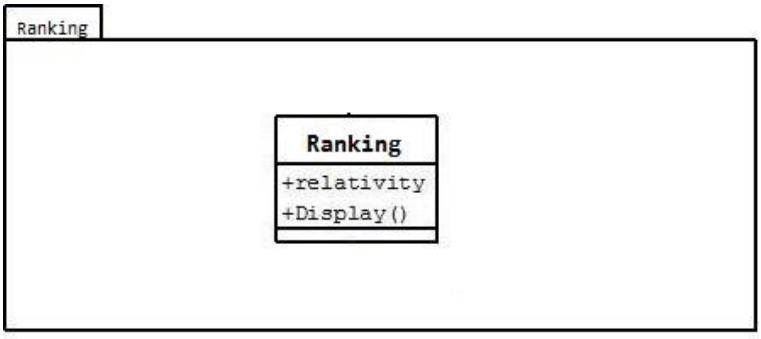


Figure 4.5 Ranking

The class Ranking uses the operation display() to show the search results with the sorted relativity. The Smart Image Search System stores the relativity in the snippets database and directly sorts the snippets with the database in-order clause.

4.2 Implementation of the Smart Image Search System

The Smart Image Search System is implemented based on the design in Figure 4.6. This section introduces each component of the Smart Image Search System in detail with its programming techniques.

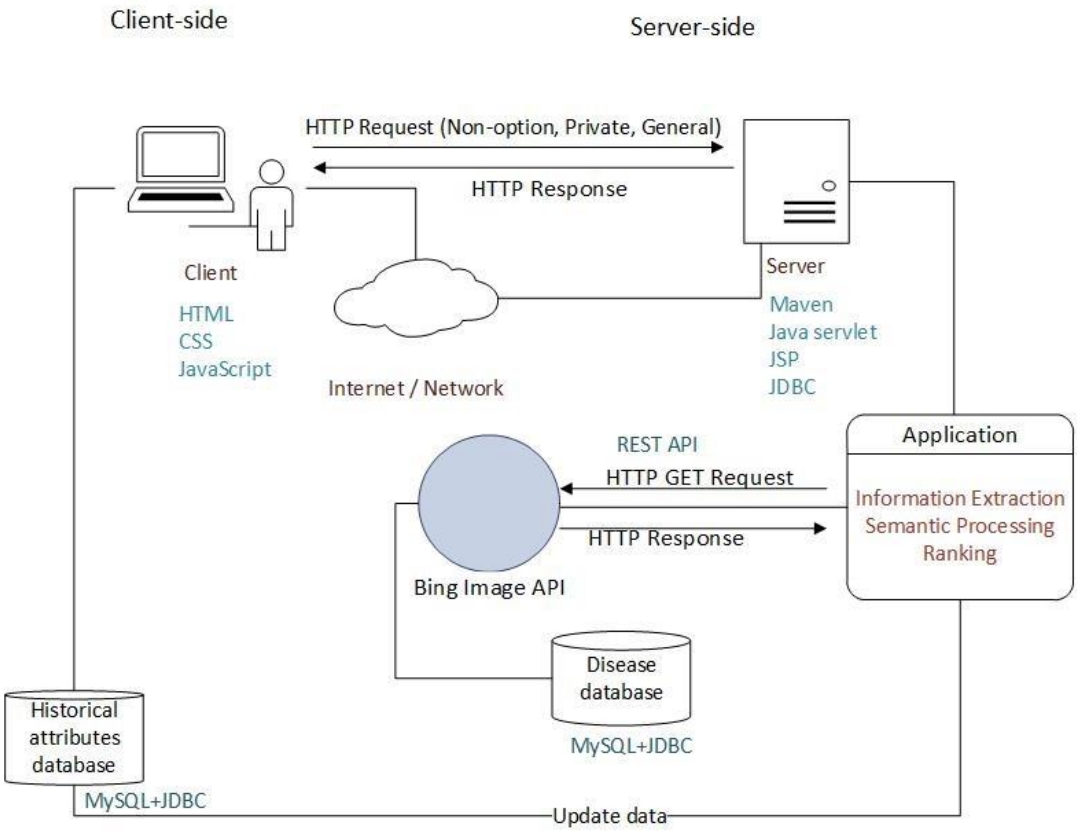


Figure 4.6 Implementation Overview

In Figure 4.6, the Smart Image Search System implementation includes two parts: the client part and the server part. The client part is not only used to make the input request, but also to make the feedback response to interact with the server. The server part processes the requests from the client part and also evaluates the client's feedback by weighting the associated attributes. The details on the implementation are introduced below.

4.2.1 Server Side

The server side of the Smart Image Search System is implemented by using JavaServer Pages (JSP) [Teodorovici 2013], Extensible Markup Language (XML) [W3schools 2017_1], and Java Servlet [Pursnani 2001] in Maven [Maven 2008]. JSP is a technology that helps software developers create dynamically generated web pages based on HTML [W3schools 2017_2], XML, or other document types. XML is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The servlet receives a client's request and generates a response based on that request. A Java servlet extends the capabilities of a server.

The implementation of each component of the Smart Image Search System is illustrated by converting the design into codes.

4.2.1.1 The Implementation of Information Extraction

The Information Extraction component of the Smart Image Search System is used for extracting useful attributes from the user's search queries. Information Extraction is implemented by its presented methodologies in Chapter 3. By importing the Apache OpenNLP library, the operation extractAttributes() in the class InformationExtraction parses each word with its POS tag. The operation extractAttributes() in the class SyntaxExtraction extracts the noun attributes as useful attributes. Figure 4.7 shows the code of POM.XML, which imports the Apache OpenNLP library. Figure 4.8 shows the code of the class OpenNLP. Figure 4.9 shows the code of the class SyntaxExtraction, and Figure 4.10 shows the code of the class InformationExtraction.

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>Engine</groupId>
  <artifactId>Engine</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>war</packaging>
  <build>
    <sourceDirectory>src</sourceDirectory>
    <plugins>
      <plugin>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>3.3</version>
        <configuration>
          <source>1.7</source>
          <target>1.7</target>
        </configuration>
      </plugin>
      <plugin>
        <artifactId>maven-war-plugin</artifactId>
        <version>2.6</version>
        <configuration>
          <warSourceDirectory>WebContent</warSourceDirectory>
          <failOnMissingWebXml>false</failOnMissingWebXml>
        </configuration>
      </plugin>
    </plugins>
  </build>
  <dependencies>
    <dependency>
        <groupId>org.apache.opennlp</groupId>
        <artifactId>opennlp-tools</artifactId>
        <version>1.5.3</version>
    </dependency>
  </dependencies>
</project>
```

Figure 4.7 Code of pom.xml

```
public class OpenNLP{

    Attributes[] attributes;
    String[] attributeName;
    String[] attributeTag;
    int attributesNum;
    public OpenNLP(String input) throws IOException{
        attributesNum=0;
        POSModel model = new POSModelLoader().load(new File("en-pos-maxent.bin"));
        PerformanceMonitor perfMon = new PerformanceMonitor(System.err, "sent");
        POSTaggerME tagger = new POSTaggerME(model);

        ObjectStream<String> lineStream =
                new PlainTextByLineStream(new StringReader(input));

        perfMon.start();
        String line;

        while ((line = lineStream.read()) != null) {
            attributeName = WhitespaceTokenizer.INSTANCE.tokenize(line);
            attributeTag = tagger.tag(attributeName);
            attributesNum++;
            perfMon.incrementCounter();
        }
        perfMon.stopAndPrintFinalResult();
    }

    public Attributes[] extractAttributes(){
        for(int i=0;i<attributesNum;i++)
        {
            attributes[i].attributeName=attributeName[i];
            attributes[i].attributeTag=attributeTag[i];
        }
        return attributes;
    }

}
```

Figure 4.8 Code of class OpenNLP

```
public class SyntaxExtraction extends OpenNLP {
    Attributes[] extractedAttributes;
    int extractedAttibutesNum;
    public SyntaxExtraction(String input) throws IOException {
        super(input);
        // TODO Auto-generated constructor stub
    }

    public Attributes[] extractAttributes(){
        extractedAttibutesNum=0;
        for(int i=0;i<attributesNum;i++)
        {
            if(usefulTag(attributes[i].attributeTag)){
                extractedAttributes[extractedAttibutesNum].attributeName=attributeName[i];
                extractedAttributes[extractedAttibutesNum].attributeTag=attributeTag[i];
                extractedAttibutesNum++;
            }

        }
        return extractedAttributes;
    }

    public boolean usefulTag(String attributeTag){
        String[] usefulTag={"NN","NNS","NNP","NNPS"};
        boolean isUseful=false;

        for(int i=0;i<usefulTag.length;i++){
            if(usefulTag[i].equalsIgnoreCase(attributeTag)){
                return true;
            }
        }
        return isUseful;
    }
}
```

Figure 4.9 Code of class SyntaxExtraction

```
public class InformationExtraction extends SyntaxExtraction{

    public InformationExtraction(String input) throws IOException {
        super(input);
        // TODO Auto-generated constructor stub
    }

    public Attributes[] extractAttributes(){
        return super.extractAttributes();
    }
}
```

Figure 4.10 Code of class InformationExtraction

4.2.2 The Implementation of Semantic Processing

The Semantic Processing component shown in Figure 4.4 is responsible for evaluating each pair of associated attributes by the three types of weight. The options for weight are stated in section 4.1.1.2. The Semantic Processing component, the class SemanticProcessing, is implemented based on the methodologies presented in Chapter 3. Figure 4.11 shows the partial code of the class SemanticProcessing.

```
public class SemanticProcessing {
    Attributes[] searchAttribute;
    int searchAttributesNum;
    Attributes[] resultAttribute;
    int resultAttributesNum;
    Attributes relativeAttributes;
    float weight;
    String[][] associatedAttributes;
    float assocatedWeight;
    double relativity;
    public SemanticProcessing (String input,String result) throws IOException{
        InformationExtraction ieSearch=new InformationExtraction(input);
        InformationExtraction ieResult=new InformationExtraction(result);
        searchAttribute=ieSearch.attributes;
        resultAttribute=ieResult.attributes;
        searchAttributesNum=ieSearch.attributesNum;
        resultAttributesNum=ieResult.attributesNum;
    }
    //first weight
    public void firstWeight(){
        firstUpdate(searchAttribute);
    }
    //second weight
    public void secondWeight(){
         firstUpdate(resultAttribute);
    }
    //third weight
    public void thirdweight(){
        thirdUpdate(searchAttribute,resultAttribute);
    }
    //calculate the relativity for each snippets
    public void calculateRelativity(){
        for(int i=0;i<searchAttributesNum;i++)
        {
            for(int j=0;j< associatedAttributes[].length;j++)
            {
                calculate(searchAttribute[i], associatedAttributes[i][j]);
            }
        }
    }
                .
                .
}
```

Figure 4.11 partial code of class SemanticProcessing

4.2.3 Implementation of Ranking

The Ranking component in Figure 4.5 is responsible for sorting the search results order by the relativity. It is implemented in Class Rank. The operation display () presents the search results in order. Figure 4.12 shows the code of Class Rank.

```
public class Ranking{
    int id;
    String title;
    String description;
    String imageURL;

    public Ranking() {
        // TODO Auto-generated constructor stub
    }

    public void display () throws SQLException{
        DBConnect conn =new DBConnect();
        String sql = "SELECT iddisease, title, description FROM disease" +
                " ORDER BY relativity DECS";
        conn.rs = conn.st.executeQuery(sql);

        while(conn.rs.next()){
            id  = conn.rs.getInt("iddisease");
            title = conn.rs.getString("title");
            description = conn.rs.getString("description");
            imageURL=conn.rs.getString("image");
        }
        conn.rs.close();
    }
}
```

Figure 4.12 Code of Class Ranking

4.3 Client-Side

Client-side means that the action takes place on the client's computer. Client-side scripting enables interaction within a web page. The Smart Image Search System uses client-side to pass the request to server-side and renders the results sent back from the server-side on a web browser.

The client-side scripting uses JSP and Cascading Style Sheets (CSS) [Wikipedia 2017_7]. CSS is a style sheet language used to describe the presentation of a document written in HTML. The Smart Image Search System layout can be accomplished visually through CSS-based design. Figure 4.13 shows the layout of the image in CSS.

```
<style>
div.img {
    margin: 5px;
    border: 1px solid black;
    float: left;
    width: auto;
}

div.img img {
    width: 100%;
    height: auto;
}
```

Figure 4.13 CSS example of a image

The image is located at the left side of the web page. Its width implies the element has the 100% of its parent container, and its height is flexible depends on upon the height of children elements of it. The border size is 1 pixel, and its color is black. The border style is solid.

The utility of JSP is useful for passing information, such as a user' search input and his search option, to the Smart Image Search System server. The server processes the information with servlets to perform Information Extraction, Semantic Processing, and Ranking. Figure 4.14 shows an example of JSP, which enables the Smart Image Search System to semantically process search queries.

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>semantic search </title>
</head>
<body>
<div class="title"> </div>

<%@ page import="java.sql.*" %>
<%@ page import="java.io.*" %>
<%@ page import="engine.search.process.associatedAttributesMap" %>

<% Class.forName("com.mysql.jdbc.Driver"); %>

<HTML>
    <HEAD>
        <TITLE>Disease map SWS </TITLE>
    </HEAD>

    <BODY>
        <H1>Disease Map Smart Search System </H1>

        <%
            Connection connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/engine","root","hao123");
            Statement statement = connection.createStatement() ;
            String query2="select * from disease where title like "+"'%"+ie.attributes[0].attributeName+"%'";
            while(num!=1){
               j2++;
               query2+=" or title like "+"'%"+ie.attributes[j].attributeName+"%'";
               num2--;
             }
             query2+=" order by relativity DESC";
             resultset=statement.executeQuery(query2);
       %>

       <table>
           <TR>
               <TH>No.</TH>
               <TH>title</TH>
               <TH>description</TH>

           </TR>
           <% while(resultset.next()){ %>

           <% String idString=resultset.getString(1);
              int id=Integer.parseInt(idString);
              String description=resultset.getString(3);
           %>
           <TR>
               <TD> <%= resultset.getString(1) %></td>
               <TD> <a href="MyServlet.do?id=<%=id%>&description=<%=description%>"
                        onclick="associatedAttributesMap.secondWeight(description)"><%=resultset.getString(2)%></a> </td>
               <TD> <%= resultset.getString(3) %></TD>
           </TR>
           <% } %>
       </table>

   </BODY>
</HTML>
```

Figure 4.14 JSP example of search page

In Figure 4.14, the Smart Image Search System semantically processes the search queries and displays the results to the user. Once the user clicks on the target result on the

client-side, the Smart Image Search System will perform the second weighting method on the server-side. The click action affects the results for the next search.

Chapter 5. Evaluation

The goal of this research is to use the new semantic method to design and build a search engine that achieves the objectives represented in chapter 1.

In order to evaluate the personalized semantic search method, there are two tasks in this evaluation chapter: one to compare the search results between non-option search and general search in the Smart Image Search System; another to compare the search results between general option search and private option search in the Smart Image Search System. Because the non-option search actually adopts the current traditional search engine, which called Bing Search Engine [Wikipedia 2017_8], to achieve the search results, the two comparisons explain why uses personalized semantic search method instead of traditional semantic search method.

5.1 Evaluation of the Smart Image Search System for Objective 1

To avoid the one-size-fits-all problem, the Smart Image Search System provides three search options for users: the non-option search, the general search, and the private search. Figure 5.1 shows that a user is able to choose from the options.

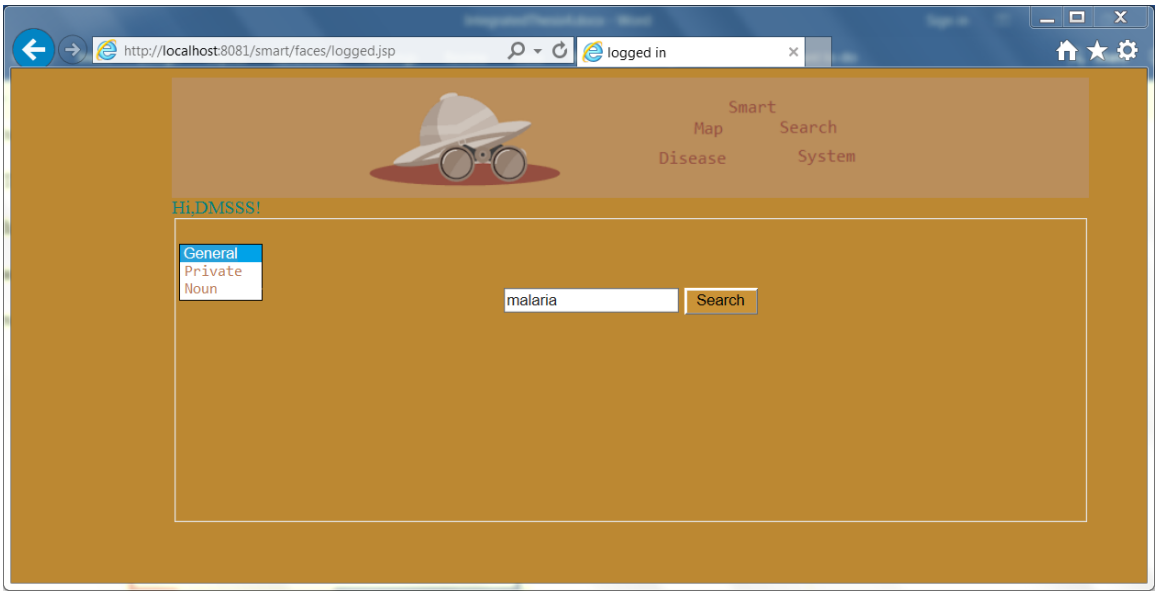
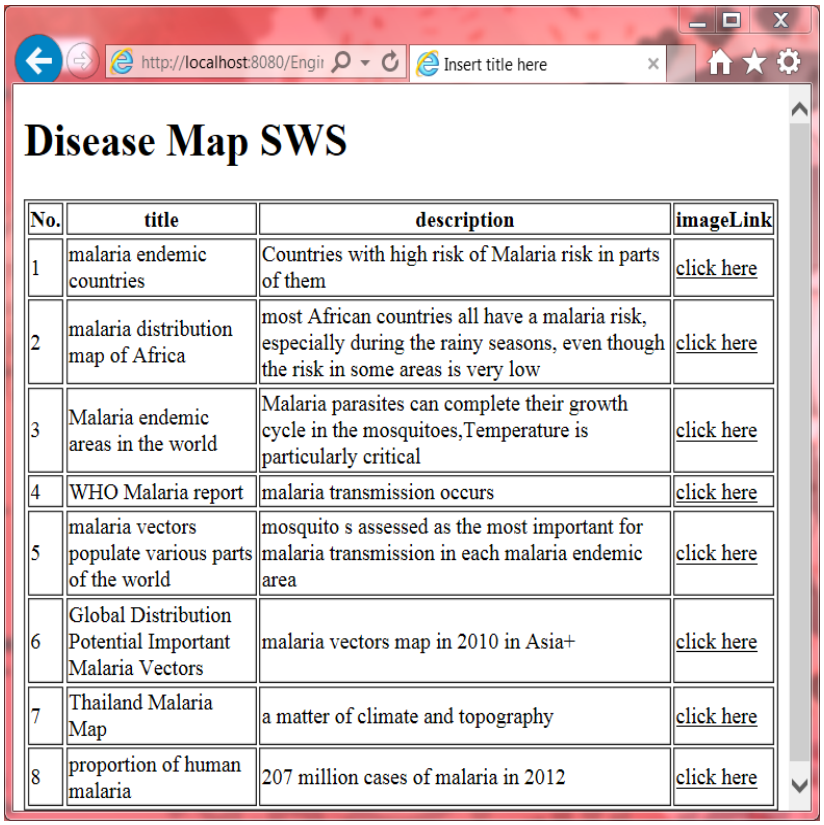


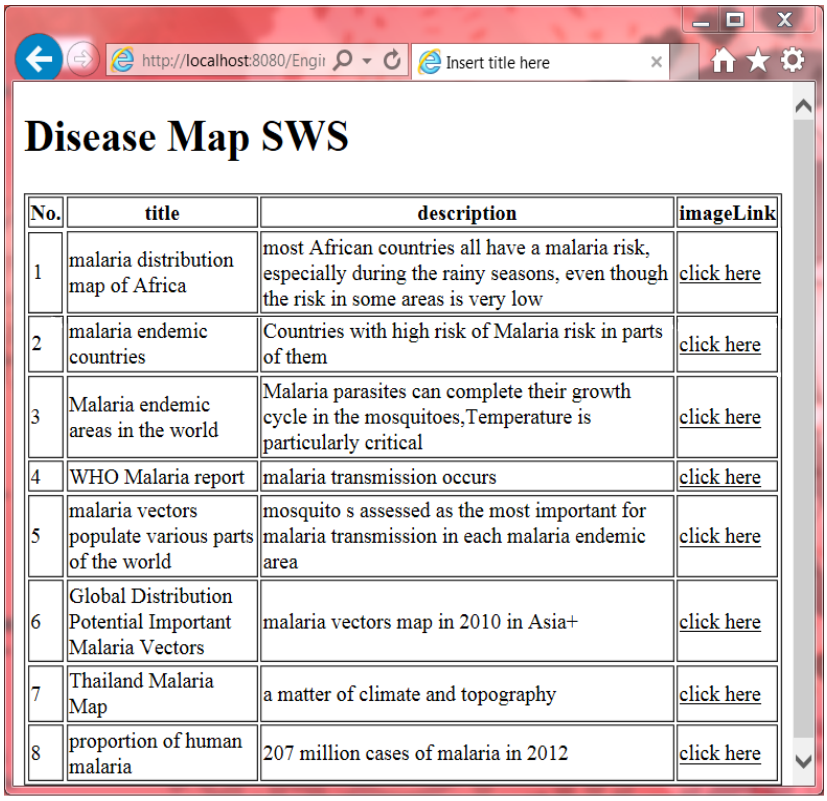
Figure 5.1 Search options

To compare the difference between private-option semantic search and general-option semantic search, the Smart Image Search System uses the same search queries to test and evaluate the search results. Figure 5.2 and Figure 5.3 show the results from the general search and private search, respecitvely.



No.	title	description	imageLink
1	malaria endemic countries	Countries with high risk of Malaria risk in parts of them	click here
2	malaria distribution map of Africa	most African countries all have a malaria risk, especially during the rainy seasons, even though the risk in some areas is very low	click here
3	Malaria endemic areas in the world	Malaria parasites can complete their growth cycle in the mosquitoes,Temperature is particularly critical	click here
4	WHO Malaria report	malaria transmission occurs	click here
5	malaria vectors populate various parts of the world	mosquito s assessed as the most important for malaria transmission in each malaria endemic area	click here
6	Global Distribution Potential Important Malaria Vectors	malaria vectors map in 2010 in Asia+	click here
7	Thailand Malaria Map	a matter of climate and topography	click here
8	proportion of human malaria	207 million cases of malaria in 2012	click here

Figure 5.2 General search for “malaria”



No.	title	description	imageLink
1	malaria distribution map of Africa	most African countries all have a malaria risk, especially during the rainy seasons, even though the risk in some areas is very low	click here
2	malaria endemic countries	Countries with high risk of Malaria risk in parts of them	click here
3	Malaria endemic areas in the world	Malaria parasites can complete their growth cycle in the mosquitoes,Temperature is particularly critical	click here
4	WHO Malaria report	malaria transmission occurs	click here
5	malaria vectors populate various parts of the world	mosquito s assessed as the most important for malaria transmission in each malaria endemic area	click here
6	Global Distribution Potential Important Malaria Vectors	malaria vectors map in 2010 in Asia+	click here
7	Thailand Malaria Map	a matter of climate and topography	click here
8	proportion of human malaria	207 million cases of malaria in 2012	click here

Figure 5.3 Private search for “malaria”

From Figure 5.2 and 5.3, we can observe that Smart Image Search System has two different search snippets with the same search queries. From the general search snippets shown in 5.2, which are highly related to the most popular historical searches of all users, we can say that “malaria endemic countries” is the most relevant result. The snippets in Figure 5.3 are obtained based on the user’s private historical search and have a higher relativity than others, which makes the “malaria distribution map of Africa” the most relevant result for this user. A different relativity for each user under the same search queries indicates that the Smart Image Search System avoids the one-size-fits-all problem,

which means objective 1 is achieved.

5.2 Evaluation of the Smart Image Search System for Objective 2

To achieve Objective 2, the Smart Image Search System should be able to avoid stop words. It uses the Information Extraction component to filter some stop words based on the POS tag and stores them as attributes. For example, when the search input is “Malaria is a common and serious malady,” the expected extracted attributes should be “malaria” and “malady” As shown in Figure 5.4, the number of actual attributes is less than the number of search queries. Apparently, in this example, the class Information Extraction has filtered the stop words and the expected results are achieved.

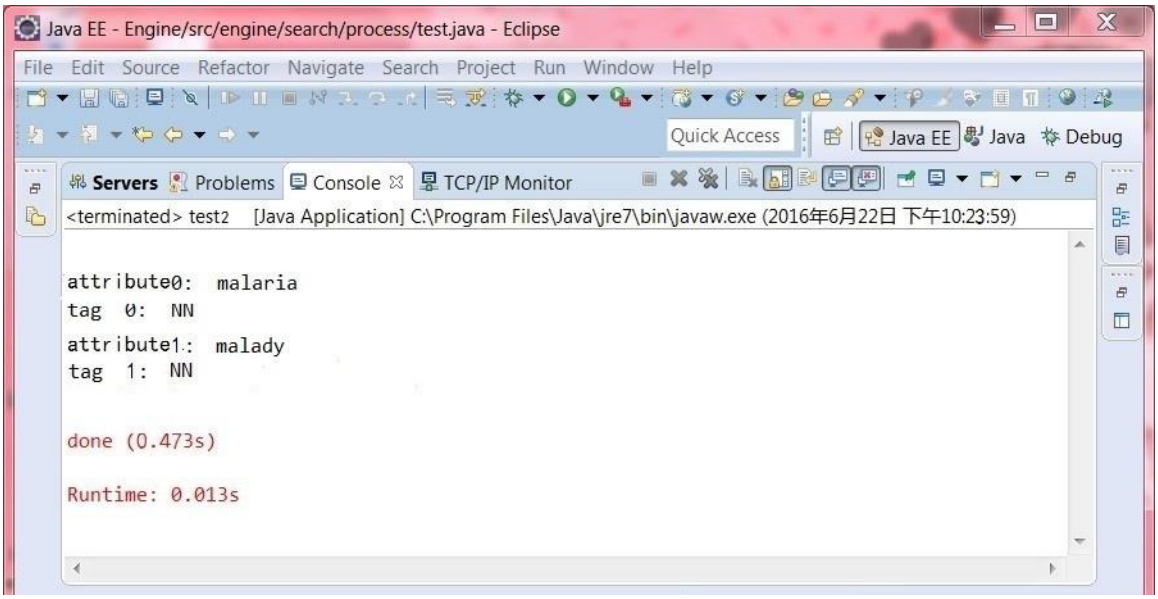


Figure 5.4 Example of Information Extraction test output

To indicate that the Smart Image Search System truly achieves Objective 2, Smart Image Search System tests four types of search queries: a single word, phrases, a single

sentence, and multiple sentences. The Smart Image Search System tests each type of search query 50 times. The test results are shown in Figure 5.5.

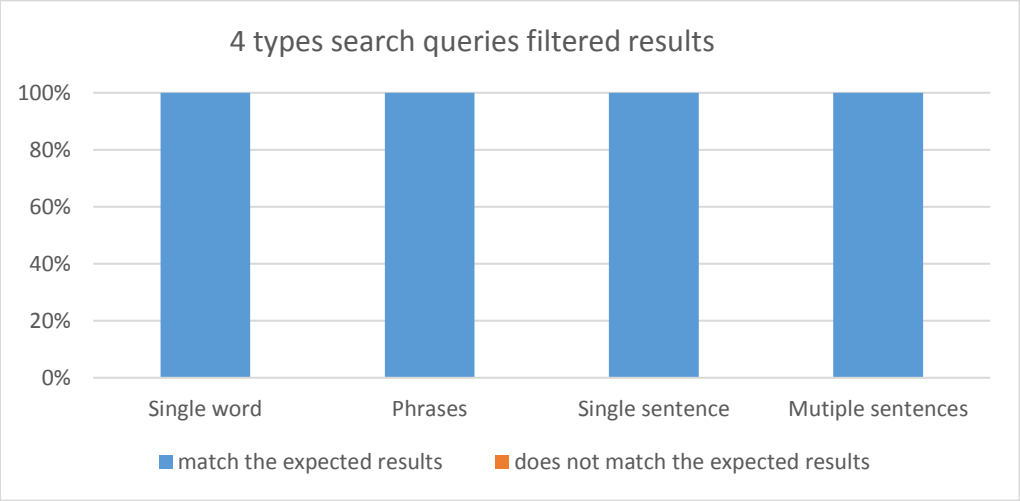


Figure 5.5 Objective 2 test results

The results show that the Smart Image Search System always removes the stop words among the four types of search queries. Thus, the Smart Image Search System achieves Objective 2 successfully.

5.3 Evaluation of the Smart Image Search System for Objective 3

The Smart Image Search System adopts the personalized semantic technology to self-improve the search results, which means the Smart Image Search System can keep updating the relativity for every search query based on users' actions and feedback. This self-improvement exists not only in the general-option search, but also in the private-option search.

To indicate the self-improvement of the Smart Image Search System, the common steps of a search task session can directly show whether or not the Smart Image Search System can self-improve. Smart Image Search System tests the same search queries 50 times with non-option search, general option search, and private search. The test results are shown in Figure 5.6.

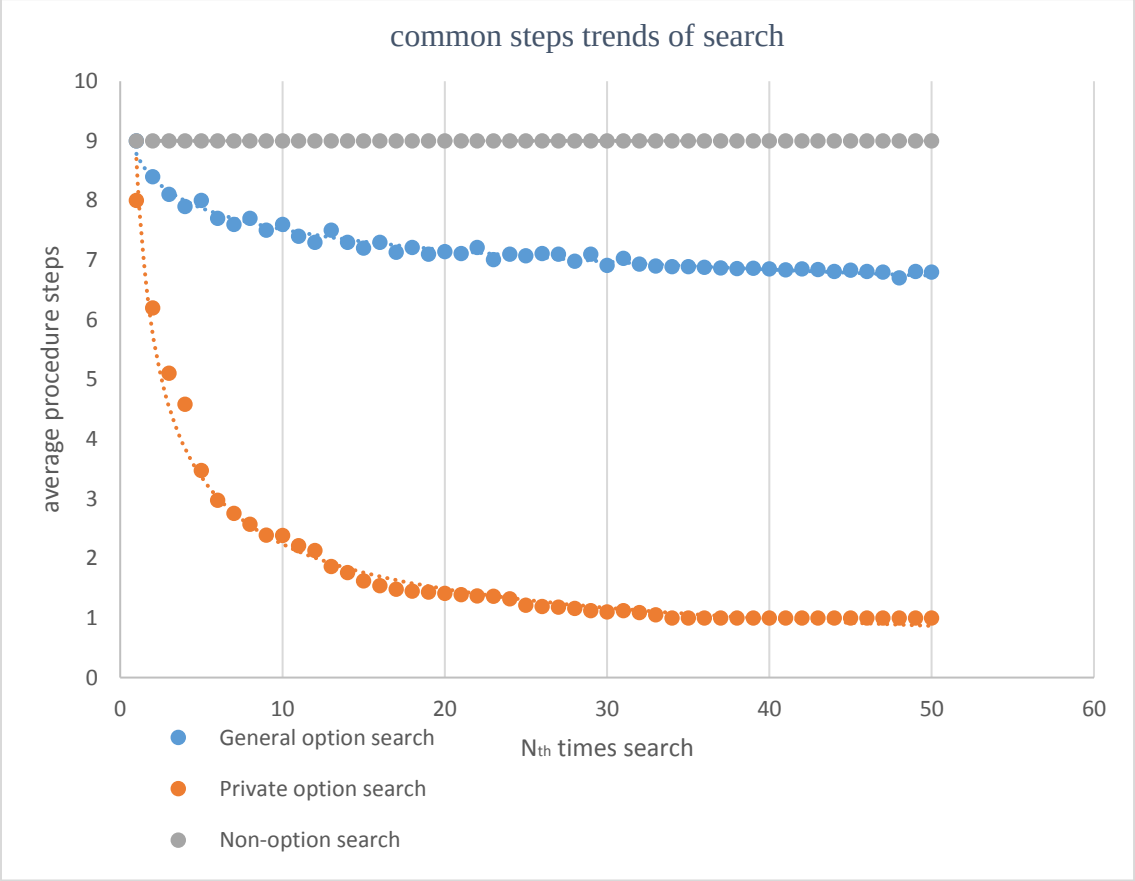


Figure 5.6 Procedure step trends

In Figure 5.6, the three procedure step trends indicate the advantages of the Smart Image Search System. The non-option semantic search adopts the traditional semantic search method. Compared to the general semantic search and private semantic search

results, the traditional semantic search results are always the same, which makes the number of procedure steps remain the same as well. However, the trends of both the general semantic search and the private semantic search show the procedure steps keep decreasing, which means the Smart Image Search System can self-improve the search results to find the most relevant results.

Additionally, compared to the private semantic search, the general semantic search still faces the one-size-fits-all issue. But the private semantic search successfully solves this problem.

Based on the results, we believe that the Smart Image Search System fully achieves Objectives 1, 2, and 3.

Chapter 6. Conclusion

This chapter reviews what the research accomplishes and discusses directions for future work.

6.1 Review

The Smart Image Search System adopts the personalized semantic search method to remove stop words, solve the one-size-fits-all problem, and self-improve the search results.

To address the objectives in Chapter 1, the Smart Image Search System includes three components – Information Extraction, Semantic Process, and Ranking – to achieve the objectives. First, the Smart Image Search System provides three options: the non-option, general option, and private option. The private option can easily solve the one-size-fits-all issue. Second, regarding avoiding the stop words, the Information Extraction component adopts natural language processing to filter stop words. The Smart Image Search System saves the storage space of attributes and processing time without many stop words. Moreover, it becomes more efficient. Additionally, the Smart Image Search System considers users' click actions and feedback satisfaction ratings to self-improve the search results.

By evaluating the personalized semantic search method, the Smart Image Search System is implemented as a study case to demonstrate that the personalized semantic search method successfully achieves the proposed objectives.

It can be concluded that the Smart Image Search System is a self-improving system. It can avoid stop words and fix the one-size-fits-all problem.

6.2 Future work

There are two directions to enhance the current work.

First, the Smart Image Search System is an image semantic search engine that employs images in its database. Future research will concentrate on retrieving more images by crawling different kinds of websites with proposed semantic search methodologies in the Smart Image Search System.

Second, to better understand the natural language to remove stop words, the Smart Image Search System will focus on machine learning [Kulesza 2012], which is a type is artificial intelligence technology, works with natural language processing, probabilities, data mining, to better understand the users' intent, and to improve the Smart Image Search System.

References

[Bhogal 2007] Bhogal, J., A. Macfarlane, and P. Smith. 2007. "A Review Of Ontology Based Query Expansion". Information Processing & Management 43 (4): 866-886. doi:10.1016/j.ipm.2006.09.003.

[Bollegala 2011] Bollegala, Danushka, Yutaka Matsuo, and Mitsuru Ishizuka. 2011. "A Web Search Engine-Based Approach To Measure Semantic Similarity Between Words". IEEE Transactions On Knowledge And Data Engineering 23 (7): 977-990. doi:10.1109/tkde.2010.172.

[Dong 2008] Dong, Hai, Farookh Khadeer Hussain, and Elizabeth Chang. 2008. "A Survey In Semantic Search Technologies". 2008 2Nd IEEE International Conference On Digital Ecosystems And Technologies, 403-408. doi:10.1109/dest.2008.4635202.

[HLWIKI 2016] "HLWIKI Canada". 2016. Hlwiki.Slais.Ubc.Ca. http://hlwiki.slais.ubc.ca/index.php/Semantic_search.

[Kulesza 2012] Kulesza, Alex. 2012. "Determinantal Point Processes For Machine Learning". Foundations And Trends® In Machine Learning 5 (2-3): 123-286. doi:10.1561/2200000044.

[Maven 2008] Maven. 2008. 1st ed. Sebastopol, Calif: Oreilly.

[Opennlp 2017] "Apache Opennlp - Welcome To Apache Opennlp". 2017. Opennlp.Apache.Org. https://opennlp.apache.org/.

[Pursnani 2001] Pursnani, Vandana. 2001. "An Introduction To Java Servlet Programming". Crossroads 8 (2): 3-7. doi:10.1145/567155.567157.

[Rahman 2013] Rahman, Mahmudur. 2013. "Search Engines Going Beyond Keyword Search: A Survey". International Journal Of Computer Applications 75 (17): 1-8. doi:10.5120/13200-0357.

[Seppänen 1970] Seppänen, Jouko J. 1970. "Algorithm 399: Spanning Tree". Communications Of The ACM 13 (10): 621-622. doi:10.1145/355598.362780.

[Stanford 2017] "The Stanford Natural Language Processing Group". 2017. Nlp.Stanford.Edu. http://nlp.stanford.edu/software/tagger.shtml.

[Techopedia 2017] "What Is Semantic Search? - Definition From Techopedia". 2017. Techopedia.Com. https://www.techopedia.com/definition/23731/semantic-search.

[Teodorovici 2013] Teodorovici, Vasile G. 2013. "Learning Javascript". ACM SIGSOFT Software Engineering Notes 38 (3): 35. doi:10.1145/2464526.2464554.

[Tf-Idf 2017] "Tf-Idf :: A Single-Page Tutorial - Information Retrieval And Text Mining". 2017. Tfidf.Com. http://www.tfidf.com/.

[Upenn 2017] "Penn Treebank P.O.S. Tags". 2017. https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html.

[W3schools 2017_1] "XML Tutorial". 2017. W3schools.Com. http://www.w3schools.com/xml/.

[W3schools 2017_2] "HTML Tutorial". 2017. W3schools.Com. http://www.w3schools.com/html/.

[WEN 2005] WEN, Ji-Jun. 2005. "SEEKER: Keyword-Based Information Retrieval Over Relational Databases". Journal Of Software 16 (7): 1270. doi:10.1360/jos161270.

[White 2007] White, Ryen W. and Steven M. Drucker. 2007. "Investigating Behavioral Variability In Web Search". Proceedings Of The 16Th International Conference On World Wide Web - WWW '07, 21-30. doi:10.1145/1242572.1242576.

[Wikipedia 2016_1] "Concept Search". 2016. En.Wikipedia.Org. https://en.wikipedia.org/wiki/Concept_search.

[Wikipedia 2016_2] "Google Trends". 2016. En.Wikipedia.Org. https://en.wikipedia.org/wiki/Google_Trends.

[Wikipedia 2017_1] "Stop Words". 2017. En.Wikipedia.Org.

https://en.wikipedia.org/wiki/Stop_words.

[Wikipedia 2017_2] "Semantic Search". 2017. En.Wikipedia.Org. https://en.wikipedia.org/wiki/Semantic_search.

[Wikipedia 2017_3] "Natural Language Processing". 2017. En.Wikipedia.Org. https://en.wikipedia.org/wiki/Natural_language_processing.

[Wikipedia 2017_4] "Context Analysis". 2017. En.Wikipedia.Org. https://en.wikipedia.org/wiki/Context_analysis.

[Wikipedia 2017_5] "Semantic Reasoner". 2017. En.Wikipedia.Org. https://en.wikipedia.org/wiki/Semantic_reasoner.

[Wikipedia 2017_6] "Part-Of-Speech Tagging". 2017. En.Wikipedia.Org. https://en.wikipedia.org/wiki/Part-of-speech_tagging.

[Wikipedia 2017_7] "Cascading Style Sheets". 2017. En.Wikipedia.Org. https://en.wikipedia.org/wiki/Cascading_Style_Sheets.

[Wikipedia 2017_8] "Bing". 2017. En.Wikipedia.Org. https://en.wikipedia.org/wiki/Bing.

[Winograd 1972] Winograd, Terry. 1972. "Understanding Natural Language". Cognitive Psychology 3 (1): 1-191. doi:10.1016/0010-0285(72)90002-3.

[Xu 2014] Xu, Zheng, Yunhuai Liu, Lin Mei, Chuanping Hu, and Lan Chen. 2014. "Generating Temporal Semantic Context Of Concepts Using Web Search Engines". Journal Of Network And Computer Applications 43: 42-55. doi:10.1016/j.jnca.2014.04.002.

[Zanasi 2007] Zanasi, A, C. A Brebbia, and N. F. F Ebecken. 2007. Data Mining VIII. 1st ed. Southampton: WIT Press.
comment: P273-274.