

Faceted Search and Browsing of Indonesian Text Collection  
Using Shallow Parsing Techniques

by

Srinivasa Raviteja Sanaka

A Thesis Presented in Partial Fulfillment  
of the Requirements for the Degree  
Master of Science

Approved November 2010 by the  
Graduate Supervisory Committee:

Hasan Davulcu, Chair  
Arunabha Sen  
Thomas Taylor

ARIZONA STATE UNIVERSITY

December 2010

## ABSTRACT

Text search is a very useful way of retrieving document information from a particular website. The public generally use internet search engines over the local enterprise search engines, because the enterprise content is not cross linked and does not follow a page rank algorithm. On the other hand the enterprise search engine uses metadata information, which allows the user to specify the conditions that any retrieved document should meet. Therefore, using metadata information for searching will also be very useful. My thesis aims on developing an enterprise search engine using metadata information by providing advanced features like faceted navigation. The search engine data was extracted from various Indonesian web sources. Metadata information like person, organization, location, and sentiment analytic keyword entities should be tagged in each document to provide facet search capability. A shallow parsing technique like named entity recognizer is used for this purpose. There are more than 1500 entities that have been tagged in this process. These documents have been successfully converted into XML format and are indexed with “Apache Solr”. It is an open source enterprise search engine with full text search and faceted search capabilities. The entities will be helpful for users to specify conditions and search faster through the large collection of documents. The user is assured results by clicking on a metadata condition. Since the sentiment analytic keywords are tagged with positive and negative values, social scientists can use these results to check for overlapping or conflicting organizations and ideologies. In addition, this tool is the first of its kind for the Indonesian language. The results are fetched much faster and with better accuracy.

To My Parents, Brother  
And Family

## ACKNOWLEDGMENTS

First and foremost I offer my sincerest gratitude to my advisor, Dr. Hasan Davulcu, who has supported me throughout my thesis with his immense knowledge and excellence. His dedication and enthusiasm towards research is very inspirational. He has been an excellent mentor and guide throughout my thesis program.

I would like to extend my sincere thanks to Prof. Arunabha Sen and Prof. Thomas Taylor for agreeing to be on my master's thesis committee and for all the valuable suggestions they made.

I would not have done my master's without my brother. He has truly been a role model throughout my life. His continuous support and encouragement is unforgettable. Thank You for everything.

Special thanks to Sedat, Vishnu and Preetham for helping me through all the phases of my master's program. Thanks to the entire CIPS lab teammates. Sukru, Hamy, Sashi, Yaksh for their valuable suggestions and support.

This acknowledgement would be incomplete without mentioning the following names. Siddhu, Siddhant, Bunty, Vamsy and Vivek, thank you so much for making my stay memorable. I am forever grateful to you guys.

I owe a lot to my parents. I cannot put into words the affection, love and blessing they show me. They always support me for everything I do. I would like to thank all my family members who supported me throughout my stay here.

## TABLE OF CONTENTS

	Page
LIST OF FIGURES.....	vi
CHAPTER	
1 INTRODUCTION.....	1
1.1 Motivation.....	1
1.2 Contribution.....	2
1.3 Organization of Thesis.....	4
2 RELATED WORK.....	5
2.1 Background.....	5
2.2 Existing Search domains for Indonesia.....	5
2.3 Related Research.....	8
3 DATA EXTRACTION.....	13
3.1 Web Extraction.....	13
3.2 Data Sources.....	13
3.3 URL Extraction.....	15
3.4 Article Content Extraction.....	16
3.5 Entity Tagging using Shallow Parsing Techniques.....	17
3.6 Tagging Keywords.....	19
4 SEARCH PLATFORM.....	20
4.1 About the Framework.....	20
4.2 Framework Requirements.....	20
4.3 Starting SOLR.....	21
4.4 Indexing Documents.....	22

4.5 Updating and Deleting Documents.....	24
5 WEB INTERFACE DEVELOPMENT.....	26
5.1 Manager.....	27
5.2 Parameter.....	28
5.3 Widgets.....	29
5.4 Web Server Implementation.....	31
5.5 Database.....	32
6 RESULTS.....	33
6.1 Sample Query Analysis and Results.....	33
7 CONCLUSION.....	38
7.1 Conclusions.....	38
7.2 Future Work.....	39
REFERENCES.....	40

## LIST OF FIGURES

Figure	Page
1. Sample search website homepage of Indonesia .....	6
2. Hallo Indonesia website homepage .....	7
3. Relation browser tool .....	8
4. Graphical interface of “Contexter” for browsing/visualizing the name- entity network .....	9
5. Faceted Navigation by IBM .....	10
6. Sample HTML from NU online .....	14
7. Sample HTML document from HIZBUT TAHRIR .....	15
8. SOLR server home page running on Jetty Server.....	21
9. XML documents from Erasmus Source with entities tagged.....	22
10. Changes made to schema.xml of SOLR.....	23
11. Screen shot of SOLR query output in XML format .....	24
12. Web Interface home page.....	28
13. Result widget output.....	29
14. Pager Widget .....	30
15. Facet search tag cloud widget .....	30
16. Text search widget .....	31
17. Minerva search engine home page.....	33
18. Architecture of the minerva faceted search application.....	34
19. Screen shot of the sample query results.....	35
20. Intrepretation of results .....	36

21. Screen shots of positive and negative sentiment tag clouds ..... 37



## Chapter 1

### INTRODUCTION

This thesis explains the procedure of extracting organization, news, and blog data from websites and developing a web search interface with facet search capability. This tool can be extensively used by social scientists to retrieve information of a particular organization to analyze the activities performed by that organization for various research purposes.

#### **1.1 Motivation**

Search engines are one of the most widely used internet tools in day to day life. Most of them provide vast data which might not be specific to single category, being specific will reduce the time for searching and improves the efficiency of retrieved data. One way to overcome this problem is by building a search engine that contains data which is specific to a category or region. This will reduce the vastness but will improve the efficiency of search results and reduces the time of operation as mentioned above. People prefer to use an approach for searching that can be described as successive refinement (A Sprink 2002). This research will be useful for various social scientists whose work would greatly depend on the information retrieved from the search engine results and will also address the successive refinement issue which is mentioned above. It is very important to develop a tool that helps social scientists to retrieve information, where the information will be useful to effectively analyze trends using the data that is available. To efficiently and effectively run such a process, I need a tool with faceted search capability, which will be provided by Apache SOLR. As the web portal contains information from news websites, organization websites and blogs which have been crawled to the latest date, the data is real, accurate and up to date. My research

concentrates on extracting information from the web and developing a web interface search engine to access such kind of information all at a single location. The web extraction includes extracting information from particular radical and counter-radical organization websites, news websites and blogs. For example, information regarding trend change of a counter radical organization on the practice “polygamy” can be retrieved from this web portal. The faceted search and full text search over the collected documents will be available on the completion of this thesis.

## **1.2 Contribution**

This thesis is carried out in 4 main steps. Firstly, Web extraction. A list of 33 Indonesian organizations which were marked as radical and counter radical organizations was received from social scientists. Most of the websites were in Indonesia language. These websites were crawled to collect the data from the sources. The collected data contains junk information, which has to be cleaned to perform operations using shallow parsers. Now I perform the annotation of various entities in the text using various shallow parsing techniques. Thirdly, the tagged text data should be converted into SOLR accessible format and all the documents should be indexed with SOLR. Lastly, I build a user interface which provides the facet and full text search capabilities using Ajax-SOLR.

Various agencies and social scientists require a tool that provides access to all of the organizations data to understand the activities of the organization, which will assist in taking strategic decisions that can prevent violence activities in the country and throughout the world. Indonesia is the 5<sup>th</sup> largest populated country in the world with large number of radical and counter radical organizations. With the available sources, 33 various radical, counter radical organizations, and blog data has been fetched from

the World Wide Web. Various entities like name of organization, date, type of organization, location will be annotated in the articles that have been collected using shallow parsing techniques like Named-Entity tagging. Nearly 50,000 articles have been collected from various organization websites like HIZBUT TAHRIR, NU, MUHAMMADIYAH, etc. These websites contain information which span over a decade, which is significant. These articles are downloaded and processed to extract the entities like date, time, location, organization.

The data collected from the World Wide Web is in hyper text markup language format which has to be converted into text, so as to perform various parsing methodologies on the text. After the data cleaning process, the text content will now undergo named entity tagger algorithm. After this step has been successfully performed, the text contains various entities that have been tagged. The tags are Person, location, organization name, type of organization, etc. Now I convert this into Apache SOLR accessible XML format and will index all the documents. Once all the documents have been indexed, I develop a web interface using the Ajax-SOLR API to completely inherit the functionalities of Apache SOLR into our webpage.

### **1.3 Organization of the Thesis**

The remainder of this thesis is organized into 6 chapters.

**Chapter 2 – Related Work** This chapter introduces various websites that provide search functionality and existing shallow parsers for Indonesia.

**Chapter 3 – Data Extraction** This chapter introduces how to extract the information from websites.

**Chapter 4 – Search Platform** This chapter introduces the search platform SOLR, which will be used various search functions.

**Chapter 5 – Web interface development** This chapter discusses details about how the extracted data stored in XML format is outputted using Ajax SOLR. The web interface widgets are explained in detail.

**Chapter 6 – Results** This chapter discusses the results of the search platform using a sample query.

**Chapter 7 – Conclusion** This comprises of the final summary of the research done and possible future work.

## Chapter 2

### RELATED WORK

#### **2.1 Background**

Text search is a popular way to find information about a particular enterprise, organization or any other source. There are few reasons why enterprise search engines do not provide exact content that user wants, they have no proper page rank algorithm and lack of metadata information. There are few search platforms which provide very useful information. Which are managed by companies like Google, Microsoft, Yahoo, and Amazon which are been used mostly by users.

Web mining (R. Cooley 1997) is the process of identifying useful patterns in the web using state of the art data mining algorithms. Large amount of data is available on the web; my thesis concentrates on collecting important information from web regarding organizations in the country Indonesia and provides a user interface that provides text and faceted search capabilities through performing shallow parsing techniques on the text.

Collecting metadata from the documents and providing faceted navigation through the documents have been in research since a long time. Various commercial organizations have built such systems. In the next section, I will explain about few of them which are in use.

#### **2.2 Existing Search Domains for Indonesia**

“BAHASA” is the official language of Indonesia. Education and almost all kinds of formal communication i.e. media throughout the country are in the same language. Since the introduction of internet in 1994 very few search engine were developed in this country with local language as the preference. There are few which make their

impact now, they are indonesia.com, incari.com, halloindonesia.com. While Google provides its services in Indonesia language, most of the people use these available search engines also.



Figure 1 : Sample search website homepage of Indonesia

The above search engine provides access to about 23,000 documents. There is no faceted search capability for this website, but provides data of various types. As you can see the “ENDONESIA” website consists of many ads in its home page. Due to these advertisements the focus of the user general diverts for the norm and many users do not like such websites. This website is listed in many sources that provide information about search engines in Indonesia.

The websites incari and halo indonesia also provide text search capabilities. No faceted search is available in both the websites. Moreover, the websites do not concentrate on organizational data. So, building a search website with organizational data will be certainly useful for many social scientists across the world.

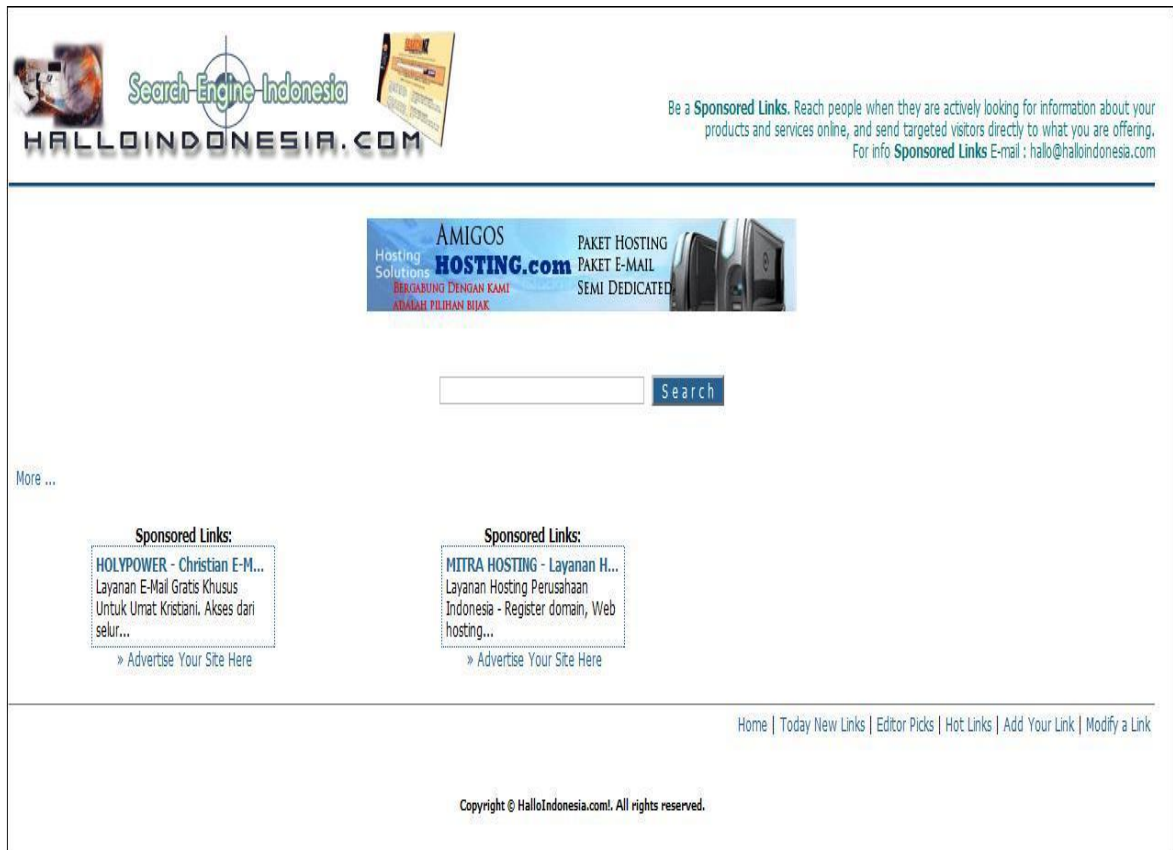


Figure 2 : Hallo Indonesia website homepage

These are few of the search engines available in Indonesia language. In the next sections, I will explain about existing faceted search enterprise engines and shallow parsing methodologies.

## 2.3 Related Research

Facet based search, text search and navigation web interfaces have become a commonly used feature in various websites. Part of my thesis work deals with web interfaces that provide such functionality. Robert G. Capra and Gary Marchionini have developed a relation browser tool to understand relationships between items in a collection and for exploring an information space (Robert 2009). It also provides a dynamic user interface that allows users to explore the data set through the use of faceted browsing and keyword search. This work is closely related to my thesis work on sentiment analytic data from Indonesia. A screen shot of the relation browser can be seen below.

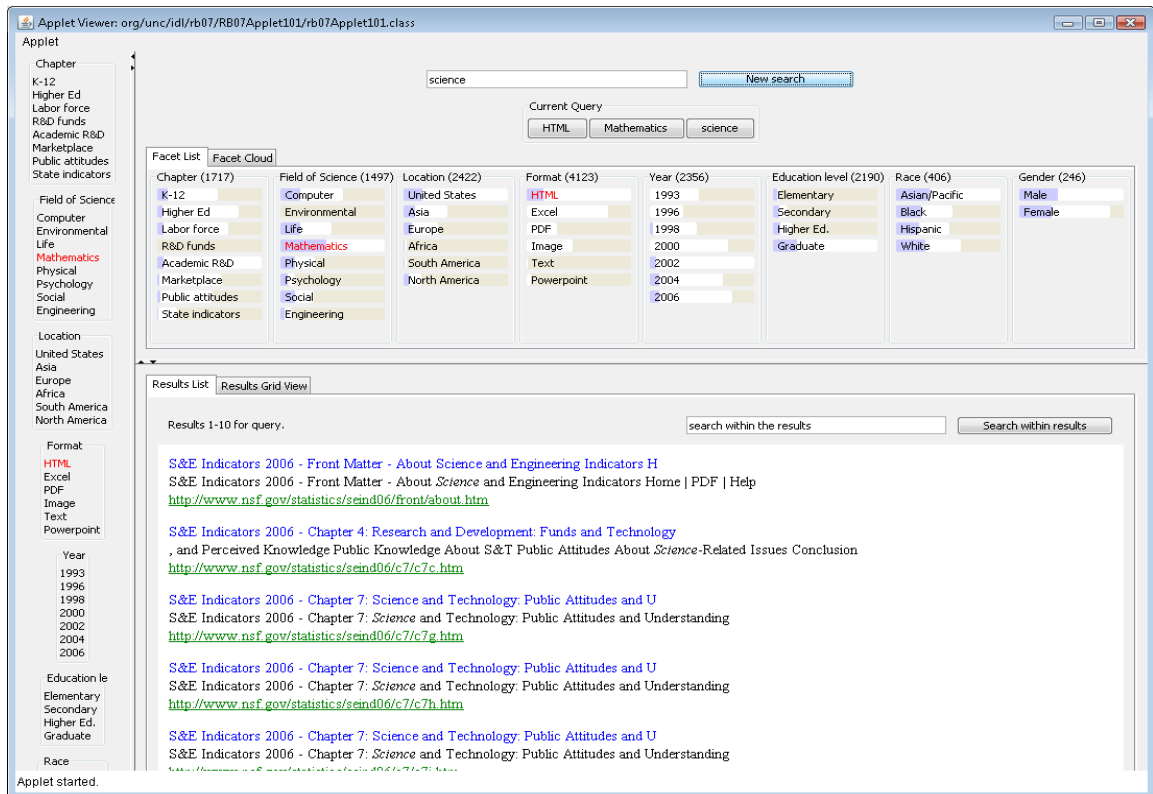


Figure 3: Relation browser tool



As you can see, relation browser is implemented as a Java Applet, which will communicate with the Apache web server and Apache SOLR search engine. The relation browser provides features like multiple facet views, static facet list, multiple result views, current query display and control, and the full text search functionality.

M Grobelnik and D Mladenic's have worked on Visualization of News Articles (M Grobelnik 2004). Their work was divided into several phases such as preprocessing the data, named entity extraction, creating graphs with relations between entities. Most of the steps have also been followed in my research to complete this thesis. "Contexter" was the name of the system they developed which is used by experts. A sample visualization system is shown in the figure below.

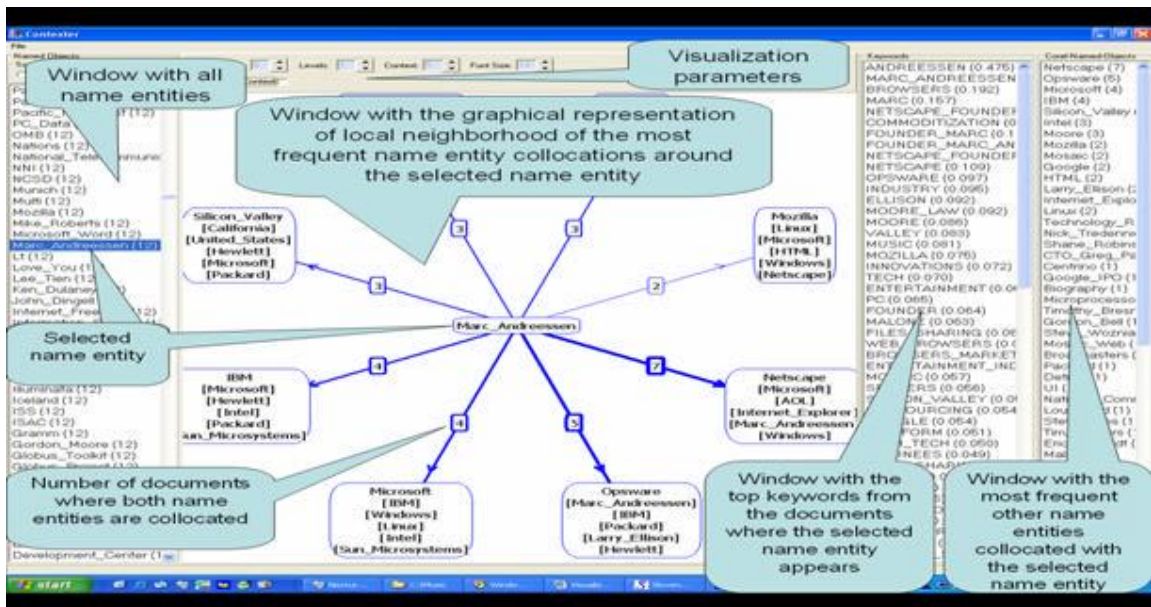


Figure 4: Graphical interface of "Contexter" for browsing/visualizing the name-entity network.

While several researchers have been working to develop search engines that provide facet selected search and text search capabilities, very few sources were found for Indonesia

language. Few of them have been mentioned above, but none of them provide a sentiment analytic way to retrieve organizational information.

IBM has developed has text search engine with faceted capability. The user is given an option to select the metadata conditions by clicking on the values which are displayed by the application.

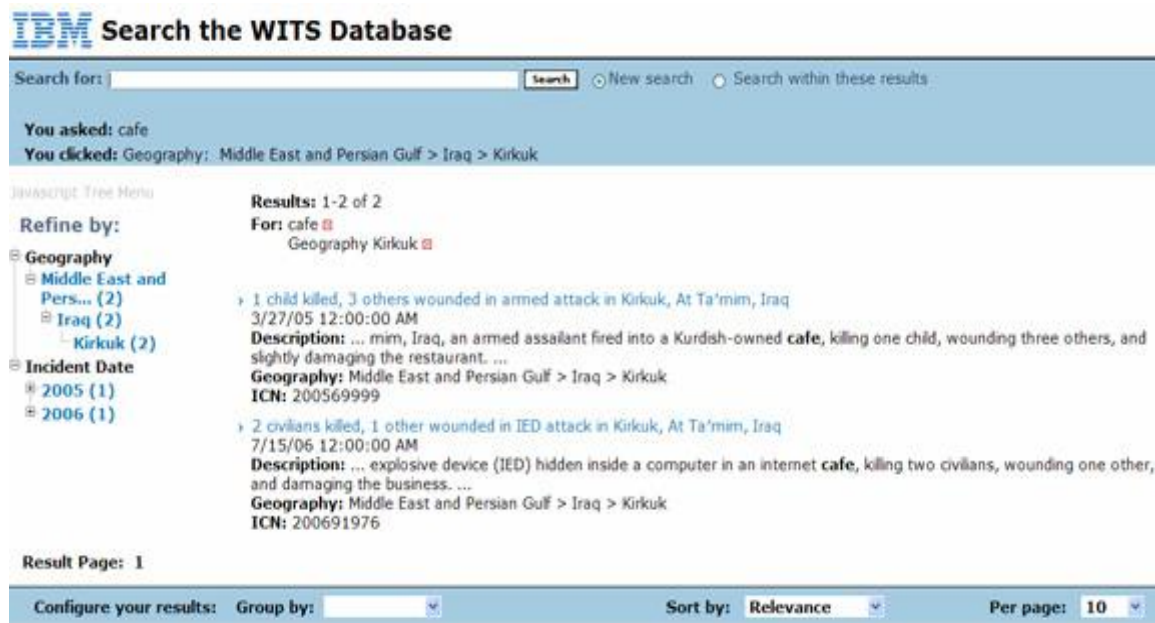


Figure 5: Faceted Navigation by IBM

By providing the metadata as the facets, there is no way the user will end up with no results. There can be multiple values of the same metadata in a single document. The query language used in this application is very similar to the internet search engine systems. They incorporate the metadata using query forms, whose fields will correspond to the elements of the metadata schema. This is widely used enterprise search engine which is similar to my research.

While part of my thesis goes to development of web search interface, the other side would be recognizing named entity tags in Indonesia data. Indra Budi and team have

developed “InNER”, named entity recognition for the Indonesian language. It combines the contextual, morphological and parts of speech features into a knowledge engineering approach (Indra Budi 2005). Research into Indonesian natural language processing is very little. The “InNER” obtains recall and precision values up to 63.43% and 71.84%. This method was used to extract the named entities such as person, organization and location from the data collection. This process is explained in more detail, in the next chapters.

Tagging parts of speech tags to words in a text is known as parts of speech tagging. It is not an easy process, because few words represent more than one part of a speech. While research is going on to develop various approaches to parts of speech tagging, Alfian Farizki Wicaksono developed a Parts of speech tagger using Hidden Markov Model for Indonesian language. I would like to explain few of its features in detail. First order and second order i.e. bigram and trigram Hidden Markov Model were used to develop the parts of speech tagger. Several methods have been used to improve the accuracy of the tagger. One of them is employing an affix tree which will be used to cover the prefix and suffix. Other is to use the succeeding parts of speech tag as a feature for Hidden Markov Model and final method is to use a lexicon to limit the candidate tags resulted by affix tree. All the methods together have significantly improved the accuracy of parts of speech tagger. An example of parts of speech tagger for Indonesia is given below.

Sentence before tagging:

ippnu memberikan beasiswa kepada

Sentence after using the HMM based parts of speech tagger:

ippnu/NNP memberikan/VBT beasiswa/NN kepada/IN

The sentence has been tagged successfully by the parts of speech tagger. The output format contains the word that has been tagged followed by a forward slash and the parts of speech tag that was specially created for this tagging purpose.

## Chapter 3

### DATA EXTRACTION

#### **3.1 Web Extraction**

Web extraction is a process of extracting data from the web. My research aims on extracting information from 33 various organization sources from the country Indonesia. The country with largest Muslim population is Indonesia. Knowing the opinions of the larger section would reflect the Ideologies and activities of a certain practice. That is the reason Indonesia has been chosen. Extracting the entities like person, location, and organization from the text and providing a facility to search over the entities is the major goal of the thesis. There are 33 different organization, news, and blog website sources available from social scientists which have been tagged as counter radical, radical and neutral organizations in the country. The web extraction of documents will be explained in more detail in this section.

#### **3.2 Data Sources**

List of 33 organizations were labeled as either radical, counter radical, or neutral organizations in the country. The list is as follows: abujibriel.com, adianhusaini.com, ansharuttauhid.com, arrahmah.com, web.bisnis.com, comops.com, eramuslim.com, fahmina.or.id, fpi.or.id, gatra.com, hidayatullah.com, hizb-ut-tahrir.org, v2.icrp-online.org, Ikhwanweb.com, inilah.com, interfidei.or.id, islamlib.com, kompas.com, lakpesdam.or.id, maarifinstitute.org, millahibrahim.net, mmjabodetabek.com, muhammadiyah.or.id, nu.or.id, paramadina.or.id, perspektif.net, pk-sejahtera.org, poskota.co.id, tempointeraktif.com.

The types of sources are news, organization, and blogs. This information was provided by social scientists. Each one labeled as mentioned above.

These data sources have different HTML formats, to get the required information few HTML's from each source should be manually analyzed to retrieve the exact information.



Figure 6: Sample HTML from NU online

The article mentioned above is from source NU, a counter radical organization website. As you can see, the HTML page contains of a lot of irrelevant information, which has to be removed for proper implementation of shallow parsing techniques.

The article content section contains one or many paragraphs of text and images describing the news.

The next article is from source HT, an organization which provides its services at various global locations. Similarly, we find the start and end tags for content, date, title, and any other specific data of interest from the document.



Figure 7: Sample HTML document from HIZBUT TAHRIR.

### 3.3 URL Extraction

A list of all the URL's of the website is required to fetch the respective HTML. The process of fetching the URL's is done by the GSITE crawler tool. The GSITE crawler takes the URL of the homepage as the input and gives a text file containing all the URL's of the website. A part of the sample output file for the website paramadina.or.id is shown below

<http://www.paramadina.or.id/>

<http://www.paramadina.or.id/2009/113/publikasi/artikel/menakar-peluang-tiga-pasangan.html>

<http://www.paramadina.or.id/2009/127/publikasi/resensi/demokrasi-sebuah-perdebatan-panjang.html>

<http://www.paramadina.or.id/2009/178/publikasi/artikel/saatnya-buaya-telan-reformasi.html>

<http://www.paramadina.or.id/2009/182/publikasi/gerakan-kebebasan-sipil.html>

<http://www.paramadina.or.id/2009/200/publikasi/resensi/cak-nur-di-mata-anak-anak-muda.html>

<http://www.paramadina.or.id/2009/229/agenda/diskusi-dan-nonton-bareng-islam-di-eropa.html>

The links are extracted with respect to the base URL. After fetching the list of URL's using the GSITE crawler, we put it into a database to fetch the HTML's for each URL.

### **3.4 Article Content Extraction**

All articles are downloaded from the websites as pure html. Firstly, given the source link of a particular website one needs to perform crawling to find its URL's. GSITE crawler is open source software that does the above functionality. After collecting the list of URL's for each source, I fetch the HTML of the URL. Fetching the HTML can be done using existing java libraries. Firstly, the links should be loaded into a database. In my thesis work, I use MySQL database. The list of URL's received from GSITE crawler is stored in a text file. I load this text file directly into the database. After loading the list, I run the program to fetch the HTML's of each URL. All the data sources are collected in the similar manner.

Now I extract the entities like title, date, content, name of sources from each HTML and store it as an XML file with entities labeled according to the specifications of



Apache SOLR. I need to find the starting tag and ending tag of each entity to get the exact content. Each source has its own format, so few HTMLs of each sources should be manually inspected to extract the start and end tag information. The content section in the html contains various tags and junk data, this data should be cleaned to run the shallow parsers on the content. Finally the HTML content can be converted to SOLR accessible XML format to perform search over the documents. The next section explains how the data has been used to find useful entities in the content.

### **3.5 Entity Tagging using Shallow Parsing Techniques**

Various tools have been developed for shallow parsing in the field of Natural language processing and Machine Learning in English, but a very few for Indonesia Language. InNER (Indra Budi 2005) is a Named Entity Recognizer for Indonesian language, which is based on a set of rules capturing the contextual, morphological, and parts of speech knowledge for extracting named entities from Indonesian text. I can extract the name, location, and organization entities from the Indonesian text using this parser. The text is now converted into one sentence per line. The input is in normal text form while the output will be in XML tagged format.

An example of named entity recognition using “InNER” is shown below:

Sentence before tagging:

“Presiden Habibie bertemu dengan Prof. Amien Rais di Jakarta kemarin”

Sentence after named entity recognition:

Presiden <ENAMEX TYPE=”PERSON”>Habibie</ENAMEX> bertemu dengan Prof.

<ENAMEX TYPE=”PERSON”>Amien Rais</ENAMEX> di <ENAMEX

TYPE="LOCATION">Jakarta</ENAMEX> kemarin.

The tag <ENAMEX TYPE="PERSON"> marks the beginning of the entity name and </ENAMEX> marks the end of the entity name. The "InNER" undergoes four main processing steps, namely Tokenization, Feature Assignment, Rule Assignment, and Name Tagging.

Tokenization identifies tokens from the input and is labeled with their kinds. Feature Assignment is the processing of assigning labels to various features like contextual, morphological and parts of speech features. The later steps are rule assignment and name tagging which are the vital operations.

If the token value is MPR, the method identifies that as an organization entity, similarly when amien is the token, it recognizes it as a person entity. But when the token is ketua, the method does not recognize any entity. Given below is a sample about the extraction process.

Example sentence:

Ketua <ENAMEX TYPE="ORGANIZATION">MPR</ENAMEX>,  
<ENAMEX

TYPE="PERSON">Amien Rais</ENAMEX> pergi ke <ENAMEX

TYPE="LOCATION">Bandung</ENAMEX> kemarin (24/4)

This example is taken from the paper published by Indra Budi.

Assigning tokens and tagging the entities appropriately is the main process.

<PERSON>, <LOCATION> and <ORGANIZATION> entities can be extracted from the Indonesian text collection. Since the date is already available from the html tag, we now have a list of various entities for each document.

### 3.6 Tagging Keywords

A list of 800 keywords has been identified by social scientists that are relevant to radical and counter radical organizations text documents based on various analysis. These 800 keywords were further divided into belief, demographics, economics, education, god, issue, location, person, organization, politics, practice, Prophet Muhammad, Quran, religious marker, social, and time. Since the person, organization and location entities have been identified from the named entity tagger. The rest of the keywords should be marked or tagged in the document collection. Since all the keywords have been manually identified by social scientists in Indonesia, these keywords are highly preferred. Simply tagging the documents with keywords will only facilitate facet search over these 800 keywords. Now methods have been followed to identify whether the keyword has been used with positive sentiment or a negative sentiment. This can be done using identifying several words before and after the keyword phrase in which it has been mentioned. The keyword length varies from 1 word to 5 words. It may be important to check the occurrence of certain keywords with length 5, which might provide some unexpected results. Sample keyword tagging is shown below.

```
<fieldname="positiveSentiment">asasimanusia</field>
```

```
<fieldname="negativeSentiment">budaya</field>
```

```
<fieldname="demographics">dunia islam</field>
```

The entities have been merged into either positive sentiment or negative sentiment. Demographics were tagged separately for providing better search for the majority keywords. This is the process of tagging various kinds of entities in the text.

## Chapter 4

### SEARCH PLATFORM

#### **4.1 About the framework**

In this chapter, I will explain about the Apache SOLR search platform and how the implementation would work on Indonesian text collection. Apache SOLR is an open source search platform from the famous Apache LUCENE project. Few of the exciting features of SOLR are full text search, faceted search, database integration, optimization for high web traffic, flexible with XML configuration, and various file type handling capacity. It uses Apache LUCENE java search library as its core and various other API's that makes it a powerful search platform tool. It is also scalable with other versions of SOLR search servers with extensible plug-in architecture. SOLR is written in java and has the capacity to run as a standalone server.

#### **4.2 Framework Requirements**

There are 2 requirements for SOLR, they are:

1. Java 1.5 or greater.
2. SOLR release

My system configuration

Java version "1.6.0\_16"

Java(TM) SE Runtime Environment (build 1.6.0\_16-b01)

Java Hotspot(TM) 64-Bit Server VM (build 14.2-b01, mixed mode)

SOLR version: SOLR 1.4.0

While SOLR can run on any java servlet container, I use jetty in my work.

### 4.3 Starting SOLR

SOLR allows users to index documents via XML over HTTP. After unzipping the latest version of SOLR, the SOLR can be started using a single command. Enter the example directory in SOLR package and use the command `java -jar start.jar`

I run the `start.jar` to instantiate a SOLR WAR with the Jetty server. The local host server will be up and running the location `http://localhost:8983/solr/admin/`

The below is a screen shot of the start page of Apache SOLR running on Jetty server.

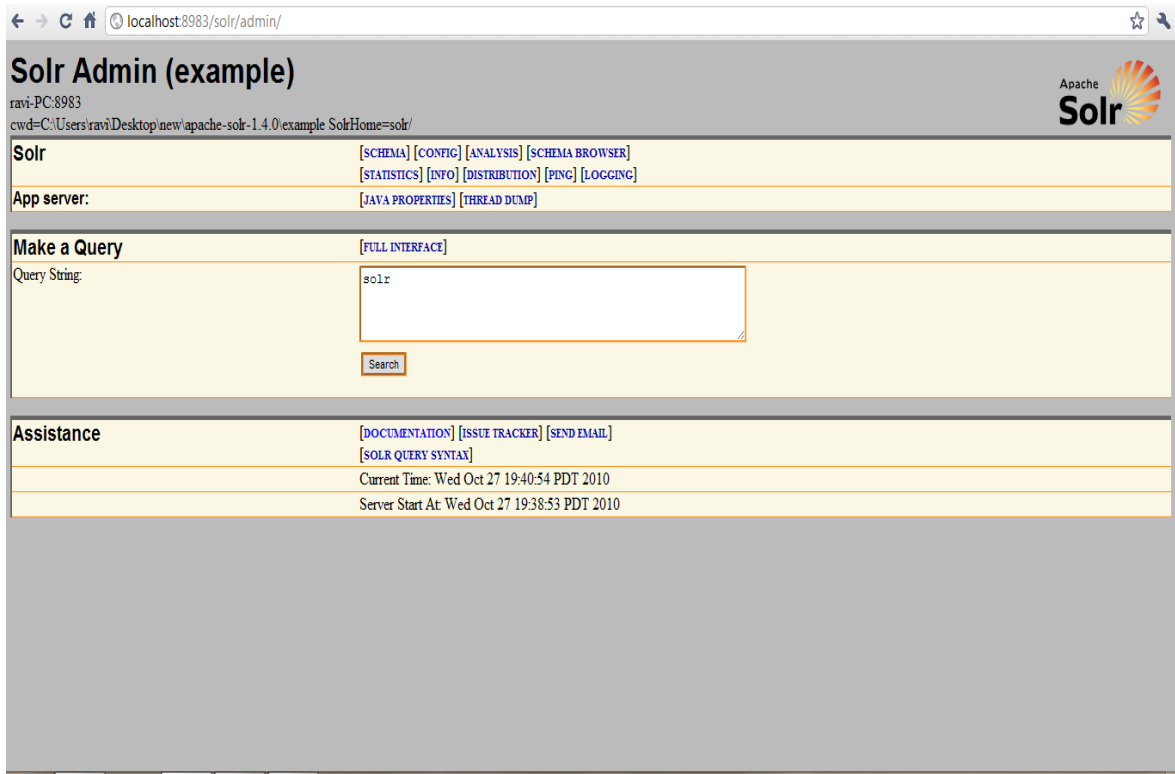


Figure 8: SOLR server home page running on Jetty server.

SOLR provides rich user interface with many functionalities, but the user should have knowledge about XML and HTTP, because the query is via over HTTP GET and the results will be retrieved in XML.

## 4.4 Indexing documents

SOLR allows user to index documents via XML over HTTP. This service is provided by the web-services like API. After the start command, the SOLR server is running but does not contain any data. The fields and types of the fields can be defined using the schema.xml file. SOLR provides the functionality to index data from databases using the “dataimporthandler” functionality. Now the data collection from Indonesia should be converted into SOLR accessible XML format to perform the full text and facet search.

A sample XML file after the conversion is given below:

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <add>
3 <doc>
4 <field name="source">Eramuslim</field>
5 <field name="URL">http://www.eramuslim.com/akhwat/wanita-bicara/arc</field>
6 <field name="content">* Mengapa Setan Wanita Lebih Mengerikan? Senin, 12/04/2010 10:19 WIB
7 * Menteri Peranan Wanita Atau Menteri Peranan Laki-Laki? Senin, 05/04/2010 15:20 WIB
8 * Perasaan Wanita Yang Dimadn Selasa, 30/03/2010 00:32 WIB
9
10 * Se jauh Apa Derita Wanita Bekerja? Senin, 22/03/2010 13:01 WIB
11
12 * Cemburu Yang Professional Senin, 15/03/2010 10:07 WIB
13
14 * Wanita Tercantik Senin, 08/03/2010 10:51 WIB
15
16 * Apakah Wanita Dapat Seajar Dengan Pria? Senin, 01/03/2010 10:17 WIB
17
18 * Rengakan Yang Membosankan Senin, 22/02/2010 17:22 WIB
19
20 * Valentine Day Senin, 15/02/2010 08:57 WIB
21
22 * Ketika Wanita Menjadi Pemimpin Senin, 08/02/2010 10:35 WIB
23
24 * Dibalik Keknatan Wanita Senin, 01/02/2010 07:05 WIB
25
26 * Pengantar Wanita Bicara Selasa, 26/01/2010 01:56 WIB
27
28 * Page 1 of 1</field>
29 <field name="id">000001</field>
30 <field name="title">Arsip</field>
31 <field name="demographics">menteri</field><field name="demographics">wanita</field>
32 </doc>
33 </add>
```

Figure 9: XML document from Eramuslim Source with entities tagged.

The document contains fields like title, source name, type of source, URL, content which was cleaned from the original HTML. To index documents of these field

names and types the schema.xml should be modified accordingly. The below fields should be mentioned in schema.xml

```

<field name="source" type="string" indexed="true" stored="true" multiValued="true"/>
<field name="type" type="string" indexed="true" stored="true" multiValued="true"/>
<field name="demographics" type="string" indexed="true" stored="true" multiValued="true"/>
<field name="title" type="string" indexed="true" stored="true" multiValued="true"/>
<field name="PERSON" type="string" indexed="true" stored="true" multiValued="true"/>
<field name="ORGANIZATION" type="string" indexed="true" stored="true" multiValued="true"/>
<field name="LOCATION" type="string" indexed="true" stored="true" multiValued="true"/>
<field name="date" type="string" indexed="true" stored="true" multiValued="true"/>
<field name="URL" type="string" indexed="true" stored="true" multiValued="true"/>
<field name="positiveSentiment" type="string" indexed="true" stored="true" multiValued="true"/>
<field name="negativeSentiment" type="string" indexed="true" stored="true" multiValued="true"/>
<field name="content" type="string" indexed="true" stored="true" multiValued="true"/>
<field name="text" type="text" indexed="true" stored="true" multiValued="true"/>
<field name="Text" type="text" indexed="true" stored="true" multiValued="true" omitNorms="true" termVectors="true" />

<copyField source="id" dest="text"/>
<copyField source="source" dest="text"/>
<copyField source="type" dest="text"/>
<copyField source="demographics" dest="text"/>
<copyField source="title" dest="text"/>
<copyField source="PERSON" dest="text"/>
<copyField source="ORGANIZATION" dest="text"/>
<copyField source="LOCATION" dest="text"/>
<copyField source="date" dest="text"/>
<copyField source="URL" dest="text"/>
<copyField source="positiveSentiment" dest="text"/>
<copyField source="negativeSentiment" dest="text"/>
<copyField source="content" dest="text"/>
<copyField source="text" dest="text"/>

<!-- Common metadata fields, named specifically to match up with
SolrCell metadata when parsing rich documents such as Word, PDF.

```

Figure 10: Changes made to schema.xml of SOLR.

In the same schema.xml user can specify various other features like token filters, tokenizers which can be used for indexing, querying and retrieval. The SOLR can be modified in many ways according to user preference. After all the required changes have been made, all the documents are moved into a directory and the post command will be called through command prompt. The command is given below

```
Java -jar post.jar *.xml
```

The \*.xml will index all the documents in the folder with .xml extension.

A screen shot of the XML output after indexing all the collection of Indonesian documents can be seen below.

```

<?xml version="1.0" encoding="UTF-8" ?>
- <response>
- <lst name="responseHeader">
  <int name="status">0</int>
  <int name="QTime">274</int>
  - <lst name="params">
    <str name="indent">on</str>
    <str name="start">0</str>
    <str name="q">poligami</str>
    <str name="version">2.2</str>
    <str name="rows">10</str>
  </lst>
</lst>
- <result name="response" numFound="198" start="0">
- <doc>
  - <arr name="URL">
    <str>http://www.erasmuslim.com/akhwat/muslimah/comment/poligami-berkah-atau-musibah</str>
  </arr>
  - <arr name="content">
    <str>Isu poligami selalu memicu reaksi keras dan menjadi isu yang meresahkan kalangan perempuan. Padahal diantara kita masih banyak yang bingung ketika dimintai tanggapan tentang gagasan poligami.</str>
  </arr>
  <str name="id">009871</str>
  - <arr name="source">
    <str>EraMuslim</str>
  </arr>
  - <arr name="title">
    <str>Poligami, Berkah atau Musibah? (Komentar)</str>
  </arr>
</doc>

```

Figure 11: Screen shot of SOLR query output in XML format.

After all the changes are made to schema.xml there are few changes to be made in solrconfig.xml. This file explains SOLR on how to perform the indexing and any abort/boundary conditions can be specified here. For example dataDir parameter can be specified in solrconfig to access any specific directory for data XML files. There are several sections involved in this file namely mainIndex section, update handler section, query section, Admin/GUI section, enable/disable components. All the sections have to be modified depending on needs. After indexing all the documents into SOLR, I can now develop a user friendly web interface using AJAX API that provides facet and text search capabilities over these indexed documents.

#### 4.5 Updating and deleting documents

Even though the SOLR is up and running, new documents can be updated at regular intervals. In the schema.xml file a unique key value for SOLR to identify each document separately should be specified. If a new document is indexed with the



same unique id, the document will be overridden and now the SOLR is updated with new document. The indexing can be done with calling the post.jar through command line. Example is `java -jar post.jar`. All the updates can be viewed in the next searches made.

Deletions can be made by posting a delete command. Specifying the unique id of the document can be more specific and different queries can be run to delete a set of documents. A complete delete of the entire indexed file is also possible through a command line input, the input `java -Ddata=args -jar post.jar "<delete>*. *</delete>"` can be given. A command can be given either to commit SOLR with changes or not. The above command actually does the commit automatically because no other parameter has been specified.

## Chapter 5

### WEB INTERFACE DEVELOPMENT

SOLR is a search application developed completely in java and supports a large number of exciting features. I will be using Ajax SOLR to provide a client side user interface which can access the data from SOLR and output search results with specific query specifications. Ajax SOLR is a JavaScript library which by default uses “jQuery”. JavaScript has several advantages when compared to JSP’s. It is more responsive, dynamic and easily testable language with the current tools available.

There are several classes involved in this application. I would explain few of the important classes. Ajax SOLR is based on the model-view-controller architecture. Where the ‘Parameter Store’ is the model, ‘Widgets’ are used as views, and ‘Manager’ performs the duties of the controller. (GITHUB social coding)

The file hierarchy of the lib folder of the Ajax SOLR is given below

Core/ contains all the framework agnostic managers, parameter stores, and abstract widgets.

Managers/ contain framework specific managers.

Widgets/ contains all framework specific widgets.

Helpers/ contains optional theme functions.

Proper coding standards are followed in this application. For example, every Ajax SOLR class, method, and property is properly implemented within the Ajax SOLR namespace. (GITHUB social coding)

The GITHUB social coding provides an easy to understand Ajax SOLR example built with sample data, which will be used in my application.

## 5.1 Manager

Manager class helps us to interact with the SOLR server. This can be done in 2 ways i.e. either directly using SOLR URL or using a proxy URL. The advantage of using the proxy URL is that the SOLR instance will not be exposed to the users of the website. Ajax SOLR can talk to SOLR as soon as the URL is specified. The Manager is specified in the below manner

```
Manager = new AjaxSolr.Manager({  
    solrUrl: 'http://localhost:8983'  
});
```

Presently the server is running locally, so the URL specifies the Jetty server 8983 location in its URL. The widgets and parameter store will be attached to the manager using the “addWidget” and “setStore” methods. Now the manager is initialized by calling the init method. Syntax shown below

```
Manager.init();
```

After the initialization, the “doRequest” method is called. This will be the first call to SOLR.

```
Manager.doRequest();
```

The manager catches the JSON response which is returned from SOLR and it typically calls all the widgets that are related to the request and shows the response. This is the basic functionality of Manager in SOLR. After the request the HTML should not display any data. The figure shows the basic layout of the index.html. The web server (Apache Tomcat) is locally installed and the index.html page is accessed through the ‘http: //localhost:8080’.

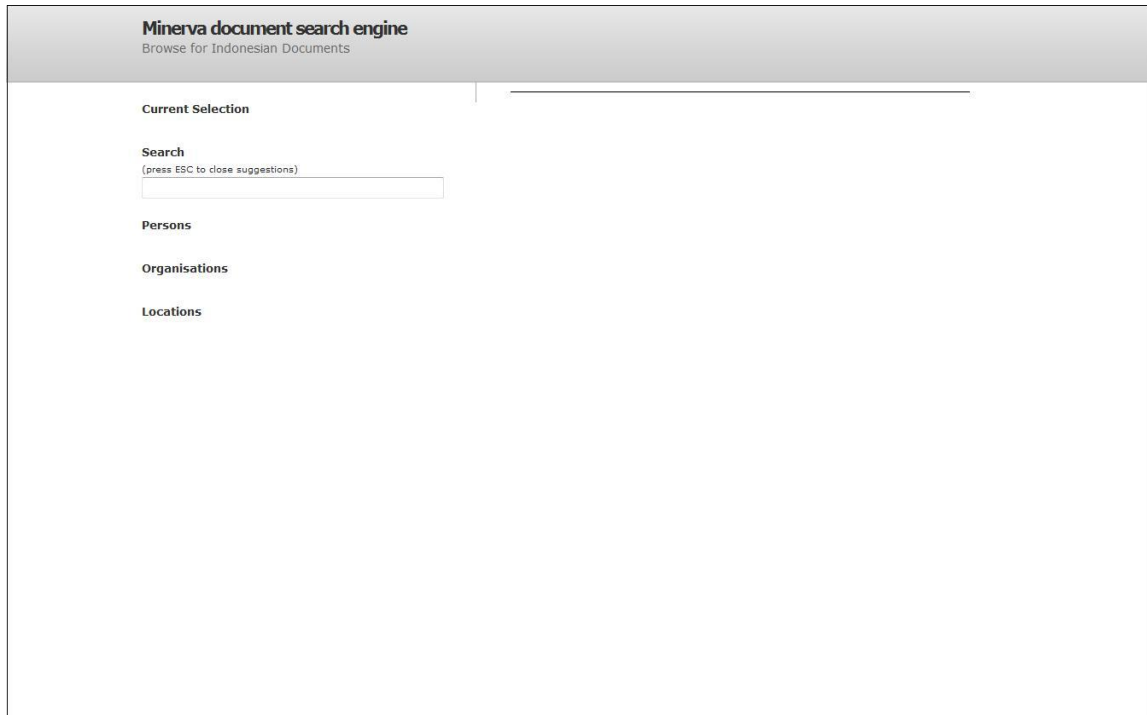


Figure 12: Web interface home page.

As specified, the html page does not output any data. Now, we will provide the functionality to see some results.

## 5.2 Parameter

The parameters of SOLR are represented as parameter objects. The parameter store contains the list of SOLR parameters and any special request to the SOLR can be made through modifications of parameter values in this store. For example ‘-‘ can be given to retrieve all the results which do not satisfy a particular query. User interacts with the widgets, all the states which the browser changes can be viewed by the user at this point and if required the user can save few of the states using the “ParameterHashStore” class for using them again. These are the basic functionalities of the parameter and parameter store methods.

### 5.3 Widgets

JSON response which is retrieved from SOLR is modified by several widgets to output the response on the user web interface. “AbstractWidget” is the base class for all the widgets. Remaining widgets inherit properties from this widget. Every widget has unique id through which it can be identified. Three methods are defined by this class, namely “init”, “doRequest”, “handleResponse”. The widgets used in my project are result widget, pager widget, tag cloud widget, and text widget.



Figure 13: Result Widget output

The result widget can be added by inheriting the result widget java script file from abstract widget. Since only 10 results are displayed per page, we also need a pagination widget to browse across all the documents.

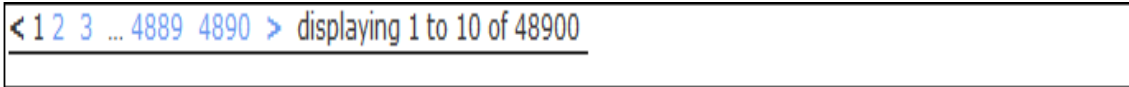


Figure 14: Pager widget

The pager widget displays the total number of documents that have been indexed. Currently there are 48900 documents from various sources. By clicking on the page number, the web page will show you 10 results of that page number. We will now see the tag cloud widget functionality.

The Reuters.js file should be modified accordingly to accommodate the functionality of tag cloud widget in to the system. Below is a screen shot that displays few facets of the tool.

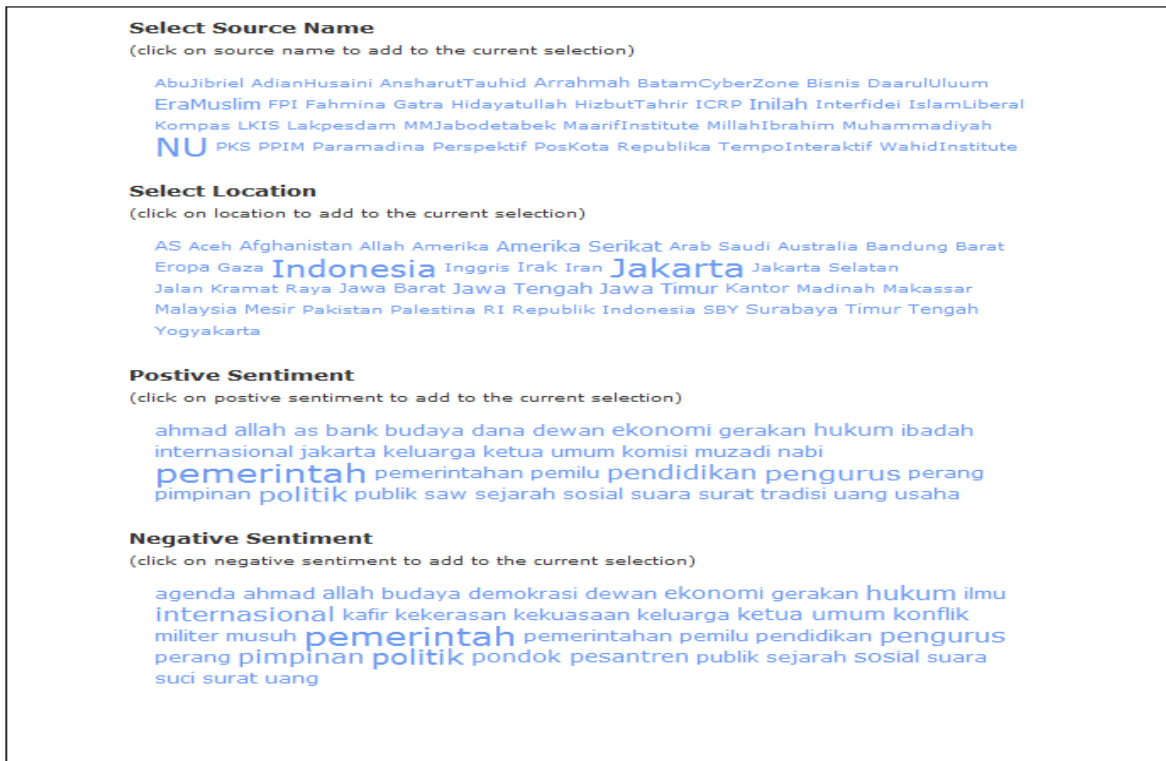


Figure 15: Facet search Tag cloud widget.

Click handler function provides various useful search techniques using 'fq' parameter. Negative queries can also be done using this parameter.

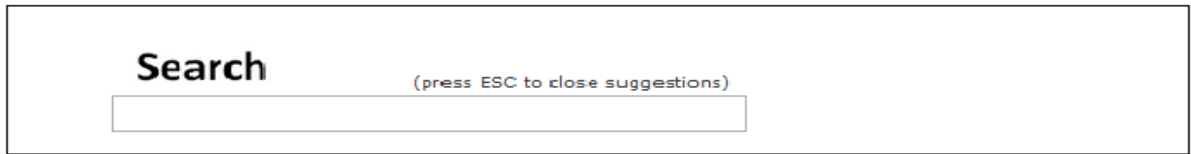


Figure 16: Text search widget.

Text widget can be integrated easily into this tool. Adding the text widget java script file and making few changes to reuters.js file should be sufficient for a fully functioning text search widget.

#### **5.4 Web Server Implementation**

The tool was developed and tested on the local server using Apache Tomcat and Jetty. This search interface will be of great use to various social scientists researching on Indonesian organizations behavior and methodology across the world. Providing this tool on the internet will be a great way for users to access this through web from anywhere at any time. The search runs on Jetty server, which is the default setup for Apache SOLR and the web interface runs on the local tomcat server. The functionality of SOLR is written in Java and the web interface code written in JavaScript, the tool can be easily hosted on the World Wide Web. SOLR can be invoked through command line and the web interface documents can be copied to the web folder of any domain to access the index.html page and other JavaScript files. So typically, the application requires a web server. The web server should provide the functionality to run the SOLR server on the web server. All the HTML and CSS files have been placed in respective folders and are accessed through various widgets. Since the data contains various tags that are extracted from the text documents, we

can either store this in a Database or XML files for later purpose. When an entity is searched or selected through the tag cloud, the request is sent to the SOLR server. Depending on the request the server runs the query and fetches the data from the indexed content. This framework allows us to add, delete or modify files dynamically. New documents have to be indexed in the specific format as mentioned in the Schema.xml file of SOLR.

### **5.5 Database**

The process of fetching the HTML content from the URL requires the use of database. MySQL database has been used in my thesis work. Once I receive a list of URL's for a website from GSITE crawler, I update the data from the text into database. Once the URL's are in the database, I fetch the HTML using java and connectivity through JDBC drivers. The data consists of various documents dated from around 2001. There are about 49,000 documents indexed in SOLR. All the documents have been created in XML formats and entities in the text have been modifies accordingly. This data can be uploaded into a database for future use.



## Chapter 6

### RESULTS

#### 6.1 Sample Query Analysis and Results

The below screen shot displays the home page of the fully functional tool. As you can see, it is a combination of several widgets.

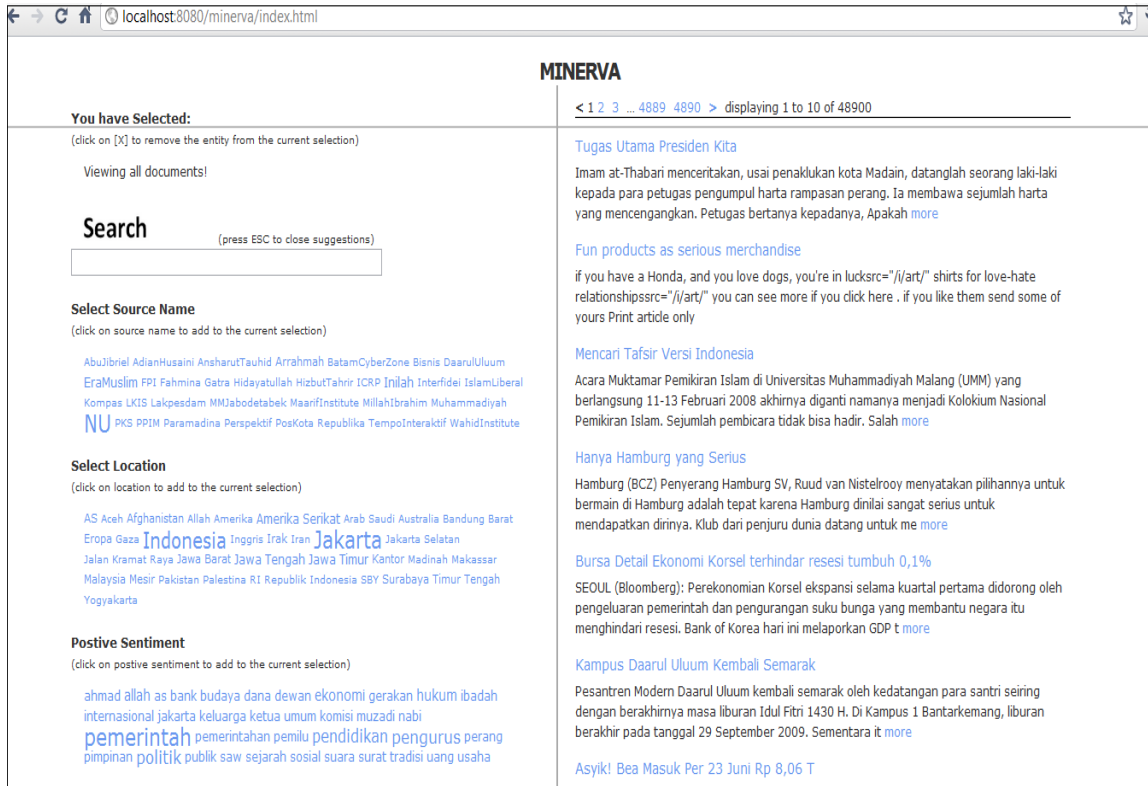


Figure 17: Minerva Search Engine Home Page

The home page as you can see, is divided into right and left sections. The right part provides all the documents of the search results, without any query the total number of documents will be around 49,000. The left part provides all the search functionalities i.e. the text search box, the faceted search. Since several entities have been tagged in the document collection, depending upon the user requirement, we can provide different facets.

A basic architecture of the faceted search navigation tool is provided below.

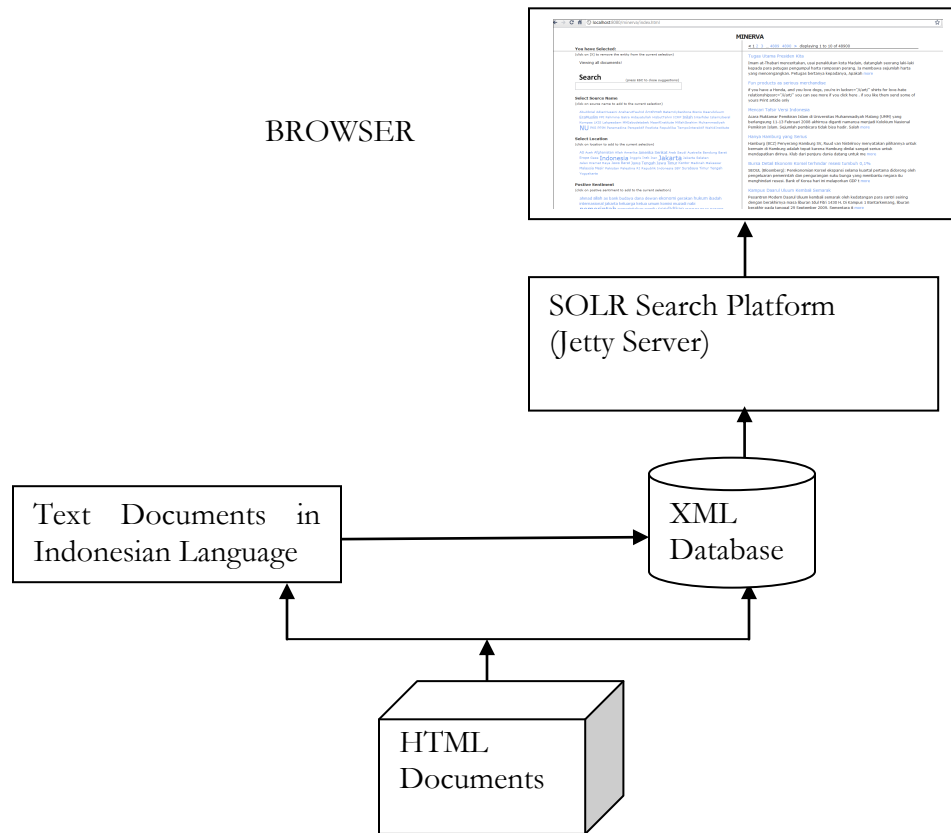


Figure 18: Architecture of the Minerva faceted Search Application

The above architecture represents the flow of operations of this thesis work. The process is done in 4 major steps i.e. Collection of HTML documents from various sources, extracting metadata information from the content of the HTML page, indexing the documents with SOLR search platform, implementing AJAX SOLR interface to interpret the results given by the search engine.

The results of the search engine can be viewed by implementing queries with keywords. I consider 5 keywords *musa*, *firaun*, *tubuh*, *laut*, *penyihir*. The English translations of these keywords are *musa*, *pharaoh*, *body*, *sea*, and *magician*. These keywords were provided by social scientists to test the efficiency of the search platform.

The query1 was run with musa + firauun + tubuh. A screen shot of the query results page is shown below.

**MINERVA**

< 1 > displaying 1 to 7 of 7

**You have Selected:**  
(click on [X] to remove the entry from the current selection)

remove all  
[X] text:musa  
[X] text:firauun  
[X] text:tubuh

**Search** (press ESC to close suggestions)

**Select Source Name**  
(click on source name to add to the current selection)

[Arrahmah](#) [EraMuslim](#) [Fahmina](#) [IslamLiberal](#)

**Select Location**  
(click on location to add to the current selection)

Abdullah Abu Musa Abu Umamah Afghanistan Aghanistan Alisan Allah Allah Swt Amerika Serikat Apabila Barat Arab Arab Saudi Babilonia Bandara Halim Perdana Kusuma Bani Israil Barat Basrah Benyamin Franklin Chechnya Ciganjur Cikeas Dinasti Rothschild Hadirat Yang Maha Esa Holywood IAIN Ibn Abbas Ibnu Abbas Ibnu Umar Institute Agama Islam Nur Jati Institute Ilmu Al Quran Irak [Mesir](#) Palestina Syam

**Positive Sentiment**  
(click on positive sentiment to add to the current selection)

ahmad [allah](#) allah swt bahasa dana ekonomi firman allah gerakan ilmu internasional jaringan jihad jihad yang kafir keadilan keluarga konflik krisis [mesir](#) militer muhammad saw [nabi](#) nabi muhammad nabi saw palestina pemerintahan perang politik rasul rasulullah saw sejarah suci terorisme

[Usamah bin Ladin, Sang Mujahid Pengguncang Tahta Firaun Abad Ini](#)

**Arrahmah**  
Bin LadenPegunungan Afghanistan. Seorang lelaki paruh baya nampak menuruni bebatuan terjal. Dengan menggunakan tongkat, pria bergamis coklat muda dengan rompi warna senada itu gesit dan cekatan memijakkan kaki di antara [more](#)

[MATAHARI TELAH PULANG; Merenungkan Sufisme Gus Dur](#)

**Fahmina**  
Langit Desember Yang Murung Jam 19.00, satu hari menjelang tahun 2009 berganti, HP berdering mengganggu makan malam gratis saya di rumah makan Jepun, milik N, sahabat saya. Jay, wartawan Koran Sindo mengkonfirmasi kabar [more](#)

[Meneladani Kesantunan Tuhan](#)

**IslamLiberal**  
Jika Anda mendefinisikan Tuhan sebagai zat yang boleh bersikap dan bertindak apa saja, berarti Anda merumuskan Tuhan yang otoriter. Tuhan dengan tangan besi. Yang kejam dan tak berkompromi. Sebaliknya, jika Anda merumusk [more](#)

[Karakteristik Isteri Shalihah](#)

**Arrahmah**  
Oleh Abu Muhammad Jibriel Abdul Rahman Wanita Shalihah (isteri shalihah) merupakan sebaik-baik dan semulia-mulia gelar yang diberikan kepada wanita kekasih Allah. Titel atau gelar itu bukan sekadar nama dan kebanggaan, [more](#)

[Khalid Al Islambuly - Eksekutor Anwar Sadat](#)

**Arrahmah**

Figure 19: Screen shot of the sample query results

As you can see the number of documents matching the query are 7. The point of interest is the source name and the key ideologies among the matching documents. The results would be interesting if there is an overlap or opposition in the positive and negative sentiment ideologies that are involved with the keyword. Similarly query2 and query3 are run with laut and peniyihir. The numbers of documents matching the scenario are 11 and 3 respectively.

The interpretation of the results in a sequential order has been given below. The interesting observation is that the number of radical organizations has been dominating than the counter radical organizations. Surprisingly, there are few results

in which both the type of organizations occur together. These results will be provided to social scientists through the web application that has been developed.

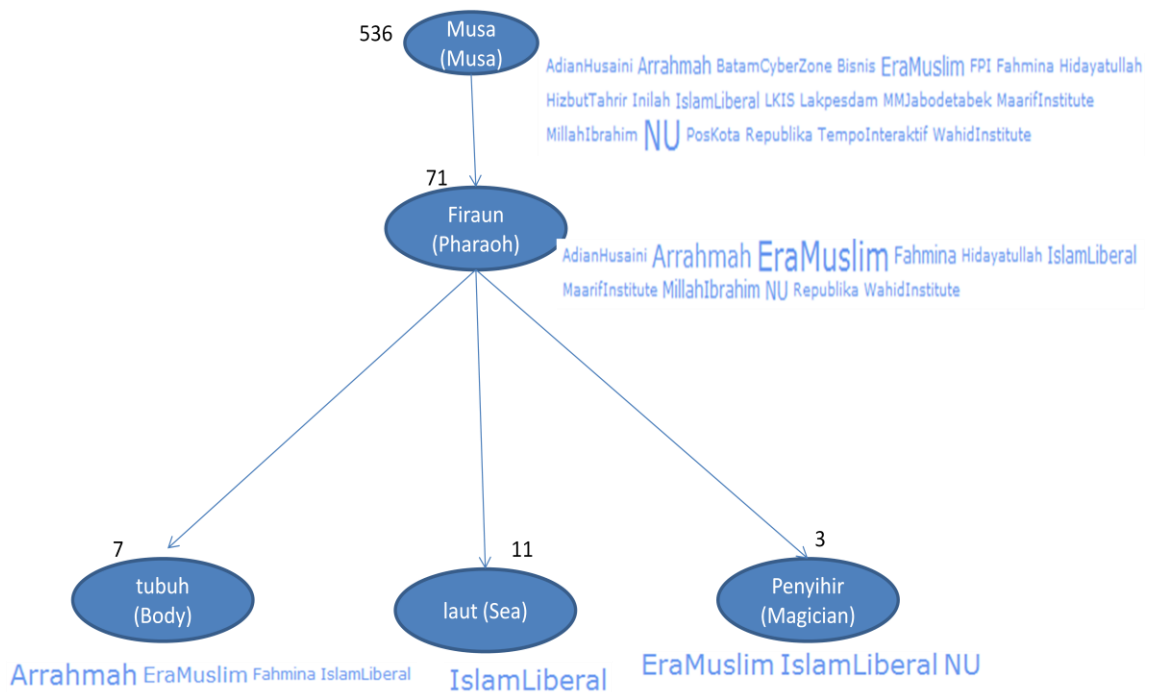
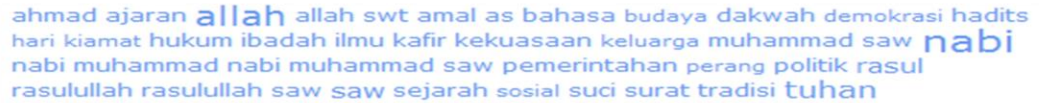


Figure 20: Interpretation of results

The positive and negative sentiments for each query will be displayed as well. So the radical and counter radical activities associated with the keyword can be interpreted. If it's an overlap, then it is interpreted as support to the keyword or else it is opposing the keyword. These results will be of great use for social scientists to analyze the trends of the organizations of a particular country or region.

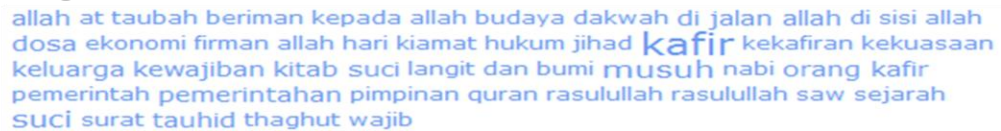
## Screenshots

- **Positive Sentiments:**



A screenshot of a tag cloud for positive sentiments. The text is displayed in various sizes and orientations, with the most prominent words being 'allah', 'nabi', and 'tuhan'. Other visible words include 'ahmad', 'ajaran', 'allah swt', 'amal', 'as', 'bahasa', 'budaya', 'dakwah', 'demokrasi', 'hadits', 'hari kiamat', 'hukum', 'ibadah', 'ilmu', 'kafir', 'kekuasaan', 'keluarga', 'muhammad', 'saw', 'nabi muhammad', 'nabi muhammad saw', 'pemerintahan', 'perang', 'politik', 'rasul', 'rasulullah', 'rasulullah saw', 'saw', 'sejarah', 'sosial', 'suci', 'surat', and 'tradisi'.

- **Negative Sentiments:**



A screenshot of a tag cloud for negative sentiments. The text is displayed in various sizes and orientations, with the most prominent words being 'allah', 'kafir', and 'musuh'. Other visible words include 'at', 'taubah', 'beriman', 'kepada', 'allah', 'budaya', 'dakwah', 'di', 'jalan', 'allah', 'di', 'sisi', 'allah', 'dosa', 'ekonomi', 'firman', 'allah', 'hari kiamat', 'hukum', 'jihad', 'kafir', 'kekefiran', 'kekuasaan', 'keluarga', 'kewajiban', 'kitab', 'suci', 'langit', 'dan', 'bumi', 'musuh', 'nabi', 'orang', 'kafir', 'pemerintah', 'pemerintahan', 'pimpinan', 'quran', 'rasulullah', 'rasulullah saw', 'sejarah', 'suci', 'surat', 'tauhid', 'thaghut', and 'wajib'.

Figure 21: Screen Shots of the positive and negative sentiment tag clouds

The tag cloud displays the top 35 keywords that have occurred maximum number of times. As you can see, the keywords like Allah, Kafir have appeared in both tag clouds. This data is will be useful for social scientists to determine the activities of respective organizations associated with the keywords.

## CONCLUSION

### **7.1 Conclusions**

This thesis presents the extraction of online information and provides a user interface to search through all the documents with facet capability to improve various factors such as time for retrieval, quality of information retrieved and many other factors. Data collection is the process of collecting information that has been posted by an organization on their respective websites. GSITE crawler is a fully functional tool that provides us with a list of URL's that a website contains. The data which is collected from websites has to be text mined to get the exact information required for the process. The process here means any kind of data mining algorithm process such as name entity tagger. The web portal will be built on the information collected from news websites, organization websites and blogs. After performing certain text cleaning processes, the shallow parsers will be run to extract entity information from the text and are tagged for faceted retrieval later. These documents will be converted to XML and are indexed. After indexing these documents with Apache SOLR, the documents can be searched and retrieved in a unique way. The entities such as person, organization, and location have been tagged using named entity recognizer of Indonesian language. There are several keywords that have also been tagged in the documents. These were provided by social scientists. These keywords will be very useful to identify the activities of certain organizations in the way of overlapping and opposing sentiments. Providing a web search interface with entities that are related to radical and counter radical organizations will be of great use for many social scientists

around the world. This tool will not only be used to track the previous data but will show a way to predict the future trends of the organizations in country Indonesia.

## **7.2 Future work**

The apache SOLR is a rich platform that provides various features. In my thesis work I use the full text search, faceted search, and filtering capabilities of SOLR. The probable next steps to improve the search server would be performance optimization and providing an administrative interface. Currently all the documents have been processed for several steps to get the SOLR search platform ready for searching. These documents can be indexed by an SQL database also. SOLR provides the facility to index document collections that have been stored in a relational database. Increasing the number of facets would be positive approach to enlarge the application onto a larger scale. The date entities of all the organizations have not been collected, because few websites did not provide that information on web page. Extracting the date entities of all the organizations will be helpful in future. Automating all these steps from HTML collection to indexing would be a great step.

## REFERENCES

- Apache LUCENE project. "apache lucene". <http://lucene.apache.org/index.html>.
- Apache SOLR project. "apache solr". <http://lucene.apache.org/solr/index.html>.
- Rajat Mukherje, Jianchang Mao. "Enterprise Search: tough stuff." *ACM Queue vol. 2, no. 2*, 2004.
- J Tang, T Arni, M Sanderson, P Clough. "Building Diversity featured seach system by fusing existing tools." 2009.
- GSITE crawler. "gsite crawler." <http://www.gsitecrawler.com>
- Github Social Coding. "Ajax SOLR." <http://github.com/evolvingweb/ajax-solr/>.
- Robert G.Capra, Gary Marchionini. "faceted exploratory search using the relation browser" *IEEE*, 2009.
- S. T. Ahmed. "Bioeve". <http://www.bioeve.org>.
- Alan Marwick. "faceted navigation for document discovery using metadata for better search" *IBM*, 2008.
- Alfan Farizki Wicaksono, Ayu Purwarianti. "HMM based part-of-speech tagger for Bahasa Indonesia" *in proceedings of International MALINDO workshop*, August 2010.
- Indra Budi, Stephane Bressan, Gatot Wahyudi, Zainal A.Hasibuan, Bobby A.A. Nazief. "Named Entity Recognition for the Indonesian Language: Combining Contextual, Morphological, and Part-of-Speech features into a Knowledge Engineering Approach" *Springer-Verlag Berlin Heidelberg*, 2005.
- M Grobelnik, D Mladenic. "Visualization of news articles." *Informatica journal*, 2004.
- Indonesia. *Endonesia*. <http://www.endonesia.com>.
- Incari. *Incari*. <http://www.incari.com>.
- Hallo Indonesia. *Search Engine indonesia*. <http://www.halloindonesia.com>.
- A Spink, T.D. Wilson, N. Ford, A.Foster and D.Ellis. "Information seeking and mediated searching study: part 3. Successive Searching." *J. American Society for information Science and technology*, 2002.
- Giorgio Sironi. "Client Applications with Ajax SOLR." <http://css.dzone.com/articles/client-applications-ajax-solr>.



H. Taney, J. Piskorski, and M. Atkinson, "Real-Time News Event Extraction for Global Crisis Monitoring," in *Proceedings of the 13th International Conference on Applications of Natural Language to Information Systems (NLDB 2008)*, London, UK. Springer, 2008, pp.

J. R. Finkel, T. Grenager, and C. D. Manning, "Incorporating non-local information into information extraction systems by gibbs sampling." *The Association for Computer Linguistics*, 2005

J. Allan, J. Carbonell, G. Doddington, J. Yamron, and Y. Yang, "Topic detection and tracking pilot study: Final report," in *Proceedings of the DARPA broadcast news transcription and understanding workshop*, vol.1998. Citeseer, 1998.

Microsoft. *Bing Maps*. <http://www.bing.com/maps/>.

S. T. Ahmed, R. Bhindwale, and H. Davulcu, "Tracking Terrorism News Threads by Extracting Event Signatures ," in *Proceedings of the IEEE International Conference on Intelligence and Security Informatics (ISI-2009)*. Dallas, Texas, USA, 2009

Shen, W., Li, X., and Doan, A. Constraint-Based Entity Matching. *Proceedings of the National Conference on Artificial Intelligence (AAAI)*. 2005.

Abujibriel. *Abujibriel*. <http://www.abujibriel.com>.

Adianhusaini. *Adianhusaini*. <http://www.adianhusaini.com>.

AnsharutTauhid. *Ansharuttauhid*. <http://www.ansharuttauhid.com/jamaah.html>.

Arrahmah. *arrahmah*. <http://www.arahmah.com>.

Daarululum. *Daarululum*. <http://www.darululum.com>.

Eramuslim. *Eramuslim*. <http://www.eramuslim.com>.

Fahmina. *fahmina*. <http://www.fahmina.or.id>.

FPI. *FPI*. <http://www.fpi.or.id>.

Hizbut Tahrir. *Hizbut Tahrir*. <http://www.hizb-ut-tahrir.org>.

ICRP. *ICRP*. <http://www.v2.icrp-online.org>.

Interfedei. *Abujibriel*. <http://www.interfedei.or.id>.

Islamliberal. *islamliberal*. <http://www.islamlib.com/id/>.

Lakpesdam. *lakpesdam*. <http://www.lakpesdam.or.id>.

LKIS. *LKIS*. <http://www.lkis.or.id>.

MaarifInstitute. *MaarifInstitute*. <http://www.maarifinstitute.org>.

MillahIbrahim. *MillahIbrahim*. <http://www.millahibrahim.net>.

Muhammadiyah. *Muhammadiyah*. <http://www.muhammadiyah.or.id>.

NU. NU. <http://www.nu.or.id>.

Paramadina. *paramadina*. <http://www.paramadina.or.id>.

PKS. *PKS*. <http://www.munaspks.info>.

PPIM. *PPIM*. <http://www.ppim.or.id>.

WahidInstitute. *wahidInstitute*. <http://www.wahidinstitute.org>.

Batam Cyber Zone. *Batam Cyber Zone*. <http://www.batamcyberzone.com>.

Bisnis. *Bisnis*. <http://web.bisnis.com>.

Gatra. *Gatra*. <http://www.gatra.com>.

Hidayatullah. *hidayatullah*. <http://www.hidayatullah.com>.

Inilah. *Inilah*. <http://www.inilah.com>.

Kompas. *Kompas*. <http://www.kompas.com>.

Perspektif. *perspektif*. <http://www.perspektif.net>.

Poskota. *Poskota*. <http://www.poskota.co.in>.

Republika. *Republika*. <http://www.republika.or.id>.

TempoInteraktif. *TempoInteraktif*. <http://www.tempointeraktif.com>

