CROSS LINGUAL INFORMATION RETRIEVAL

Anuradha Ashavatha Rao - 50207047
Dhruva Kumar Srinivasa - 50207093
Nagesh Rao - 50206741
Rajeev Vaswani - 50204855

ABSTRACT

This project deals with the development of a multilingual search engine, which enables the search for a specific query across different languages. The corpus consists of tweets crawled across multiple languages. The search engine should allow for querying in the languages contained in the corpus.

What is cross language information retrieval?

Cross-language information retrieval (CLIR) is a subfield of information retrieval dealing with retrieving information across languages, relevant to the user's query.

Introduction:

The purpose of this project was to build a cross-lingual information retrieval system. The basic function of a search engine is to accept a query from the user, and based on the keyword(s) it finds, return the appropriate search results. Our search engine searches across the corpus available to it, and returns search results across 4 different languages. We also implement the traditional method of faceting, in which we provide the user with the option of filtering the posts according to the language and the date.The search is performed on data collated from crawling tweets on Twitter using the Twitter Stream and Search API.

The tweets have been crawled across 4 different languages:

English - en
Spanish - es
Italian - it
French - fr

The major fields that we included in the indexed data were:

tweet_lang: The language in which the tweet is posted

username: The author of the original post

hashtags: The hashtags that have been used to tweet.

mentions: Any and all of the users that have been mentioned in the post.

tweet_text: This is a general field that is further divided into fields based on the languages.

text_en - For English
text_fr - For French
text_es - For Spanish
text_it - Italian

The text field consists of the textual content of the post.

Hashtags: This field includes all the hashtags that were used by a particular user in a post.

URLs: The expanded URLs mentioned in the post are displayed, if the user wishes to navigate to another page.

The major fields above have been indexed in Solr.

The displayed fields are:

User name
Text
URL

The result is returned in the language in which the user queries, and an advanced option to translate the post to other languages is also provided to the user, so that he can view the post in the language of his preference.

We have used Vector Space Model(VSM) as the information retrieval model of the search engine. In VSM, the term documents are represented by vectors in a n-dimensional space. The vectors for the same language would be located in proximity to each other. Querying in a particular language would return results with posts predominantly in the same language, as the vectors would be located in close proximity.

We have used Apache Velocity as the template engine for our UI.

Velocity:

Solr includes a sample search UI based on the VelocityResponseWriter (also known as Solritas) that demonstrates several useful features, such as searching, faceting, highlighting, autocomplete, and geospatial searching.

Velocity is a Java-based template engine. It permits anyone to use a simple yet powerful template language to reference objects defined in Java code.

Velocity can be used for the development of web applications according to the Model-View-Controller (MVC) model. It assists in keeping the Java code, and the web pages independent of each other.

IMPLEMENTATION

Overview:

The tweets are indexed in solr and collected using Twitter's API. We used a python script to clean the tweets and remove the undesired fields. The cleaned up data is translated at index-time into the desired languages and stored in the corpus. The keyword in the user's query is matched across the different languages in the corpus and the posts are returned in their original language.

UI:

Along with Velocity, we have used HTML5, CSS, and JavaScript for the front end.

Faceting:

Faceting is the arrangement of search results into categories based on query terms. We have included two facets in our search engine -

1) for language, where the user can choose filters to view posts from a particular language. 2) for a date filter, which can be used by the viewer to narrow down his search result to a specific time range.

Highlighting:

Solr provides certain highlighting utilities, which can be used by various RequestHandlers. The solr.Highlight Component is used to highlight the field values. We have used this in our search engine, to highlight the matching terms of the user's query.

Score boosting:

Filters:

Based on the languages, we had to add certain tokenizer filters. The following are the list of the same –

English – LowerCaseFilterFactory and PorterStemFilterFactory
French – ElisionFilterFactory, FrenchLightStemFilterFactory and LowerCaseFilterFactory
Spanish – SpanishLightStemFilterFactory and LowerCaseFilterFactory
Italian - ItalianLightStemFilterFactory and LowerCaseFilterFactory

Translation :

For our language translation, we have used Yandex.Translate, which is a web service that is intended for translation of web pages or text into another language. We have used Yandex to translate our queries to multiple text languages.

The reason for using Yandex API is that it is free with unlimited queries, and we determined the results to be accurate enough.

Query Boosting:

After trying out various Query Parsers, we decided to go with the Extended disMAx Query Parser. edisMax was used for query boosting, where the field of hashtags was boosted to improve relevance feedback.

The search engine is hosted on the URL: http://35.165.228.238:8983/solr/project4/browse
It has been hosted using AWS, provided by Amazon.

References:

http://velocity.apache.org/
http://velocity.apache.org/engine/1.7/developer-guide.html
http://www.cfilt.iitb.ac.in/resources/surveys/Swapnil-Cross-lingual-Information-Retrieval.pdf
https://en.wikipedia.org/wiki/Cross-language_information_retrieval
https://en.wikipedia.org/wiki/Yandex
https://cwiki.apache.org/confluence/display/solr/Apache+Solr+Reference+Guide