English to French Query Translation
(Research on Cross Language Information Retrieval)

Final Project Report

Authors:
Shilpa Dhagat
Colin Etzel

Submitted to:
Prof. Laura Dietz
University of New Hampshire

Task Motivation and Description

Cross Language Information Retrieval (CLIR) is a kind of information retrieval system where we use queries written in a single language to retrieve documents from a pool of multilingual documents written in many different languages. The system involving several languages can be important for global information research, knowledge sharing etc. Usually, Cross language information retrieval works with a single language pair, retrieving documents in a language different from that of a query.
In our project, we are also dealing with a single language pair. The aim is to retrieve the relevant documents in French language given an English query. As part of the baseline approach, we are retrieving the English documents for the English query and finally matching two set of documents in both the languages. The motivation behind the task is to benefit different kind of users. Monolingual users who are keen to learn French language can be benefited specially by having the same document side by side in both the languages. It can also help the users who can read both the languages but inputs the query in most fluent language.
There are mainly three different ways in which cross-language information retrieval works – through query translation, document translation or both. We are using query translation technique in which we are converting an English query into a French query and then retrieving the documents in French language. There are four different techniques that uses translation resources such as Dictionary-Based, Parallel corpora, Comparable corpora and Machine translator. In our project, we focused on Machine translator and Parallel corpora based CLIR techniques.

Related Work

The several approaches that we have used in our project are described below:

1. TF-IDF or Term frequency-inverse document frequency
   We have used tf-idf weighting factor for ranking English and French documents per their relevance given an English query. Each term in the English query was translated into French. All document vectors used tf-idf weights.

2. Cosine Similarity
   It is a vector space model similarity metric that we have used to find closeness between documents and queries. It is the dot product of the vectors to be compared divided by the product of their magnitudes, or simply the product of the unit vectors of the two vectors to be compared.

3. Tokenization and Query Processing
   We have used tokenization in case of phrase queries. As soon as the user enters a phrase query, system should be able to convert it into several tokens excluding the stop words. Query Processing helps in translating a high-level query into a simplified query to get the results. Other parts of query processing include stemming, lowercase text processing.

4. Indexing
   Indexing is used to improve the speed and performance in finding the documents for our English query. We are using PyLucene for indexing in our project.

5. Evaluation metric – Normalized Discounted Cumulative Gain (NDCG)
   We have used NDCG as the ranking measure in our project. assigning a quality score to our implementation based on how much later in the query results the relevant document appears than in the English-only ceiling. The same or better ranking is the maximum possible score (1.0) with lower scores decreasing at a logarithmic pace, and a minimum score of 0 if a match is not found.

6. Paired-t test
   Allows statistical comparison of two related data sets. We compare the inverse of our NDCG metric (1 – NDCG) to get an idea of how much our implementation differs from the ideal baseline (the English-only query rankings). Dividing the mean difference from our baseline divided by the mean standard error of our baseline gives the t-score, allowing a comparison with a paired-t test chart to get the statistical significance of the difference. Needs an at least approximately normal (Gaussian) probability distribution to function well [8].

7. Query Expansion
   Query expansion helps to improve retrieval performance of the system. There are two different query expansion methods Local and Global methods. Local methods are relevance feedback and Pseudo relevance feedback and global methods include Thesauri and automatic thesaurus generation. We haven't fully implemented Query expansion in our system. But our system can get the synonyms of the query terms using PyDictionary. Query Expansion has three translation techniques, pre-translation, post-translation or both.

8. Translation
   Translation is a pivotal activity for CLIR engines. The three different types of translation techniques are Query translation, Document translation or both. We have used Query translation in our system which is both time and cost efficient as compared to document translation. We have used Microsoft translator (MS translator) for translating an English query into a French query.

9. PyLucene 6.2
   The project uses PyLucene version 6.2 for indexing, searching of documents, tokenization, Query processing and applying TF-IDF approach to the searched results.

Approach

We built on pyLucene's sample file indexer and file searcher for our pyLucene implementation of our cross-language retrieval model. Term and document frequency information was stored by the indexer using the Lucene classes EnglishAnalyzer and FrenchAnalyzer instead of the StandardAnalyzer.

Lucene's FrenchAnalyzer utilizes Dr. Jacques Savoy's French stopwords [1] and French Stemmer Plus [2]. Tokenization involves splitting words apart by punctuation or whitespace and converting all text to lowercase.

The Lucene EnglishAnalyzer adds to the StandardAnalyzer in Lucene, using the porter stemmer algorithm [3] and stemming English possessives. Both analyzers automatically lowercase text fed to them and tokenize text based on punctuation and whitespace as deliniators for the tokens.

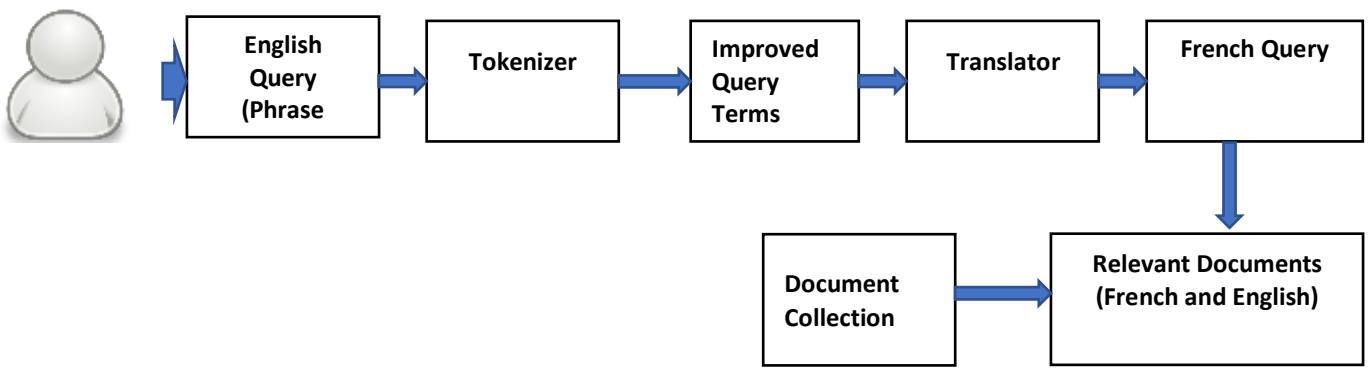
PyLucene's sample file searcher was modified to use the EnglishAnalyzer or FrenchAnalyzer while using the ClassicSimilarity (an optimized subclass of TFIDFSimilarity) object to rank the results. Ranking is determined by tf-idf (standard cosine similarity) after poor matches are filtered out of the results by a Boolean model in the similarity object [4].

The English file searcher has no changes from the default file searcher aside from the different analyzer and utilization. The French file searcher takes the raw text and filters out English stop words (as this happens under the hood in the English file searcher's query processing) and then translates each word individually to French using machine translation from Microsoft Translate. The French terms are then combined into a single query and analyzed using the before mentioned method with the FrenchAnalyzer. A query is inserted, very poor matches are filtered by the Boolean attachment, and the cosine similarity is calculated and used to rank the remaining documents based on how similar they are to the translated query.

Baseline Approach

In the baseline approach, we are just retrieving the English documents for an English query. By doing this we can compare the baseline approach with our standard approach which is to first translate an English query into a French query and then retrieving the relevant French documents. Having the English and French documents side by side for the same query will help in achieving the motivation behind the project. As well as to compare the scores and relevancy of both English and French documents.

System Architecture



The series of operations in our system are described below:

1. User input
   This is the first step where the user enters an English query which then goes to the tokenizer.
2. Tokenization
   If the user enters a sentence or a phrase query, then the tokenizer will convert the query into a list of important terms by removing the stop words.
3. Translation
   The translator will then convert the improved query into a French query by word to word translation.
4. Retrieving documents
   Finally, we get the list of relevant documents from the document collection with their scores. The baseline approach will retrieve only the English documents without translating the English query whereas the standard output will first do English-French query translation and then retrieve the relevant French documents.

Evaluation

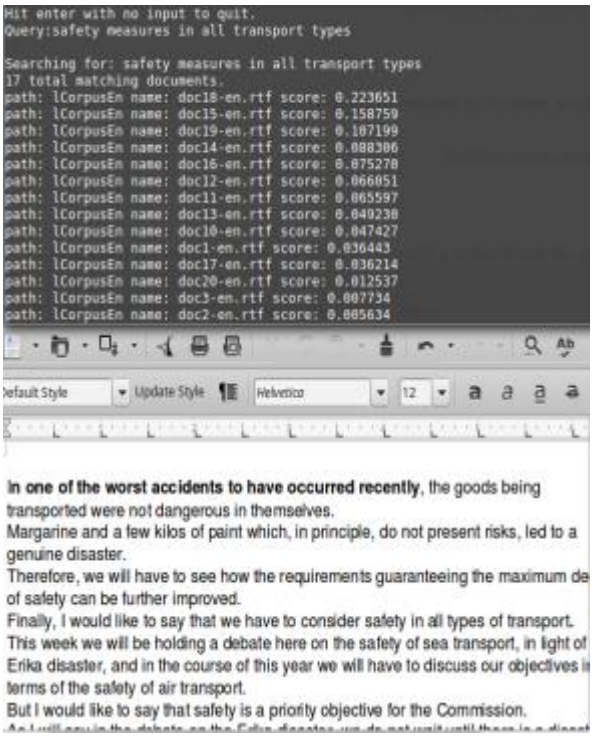
Data Used

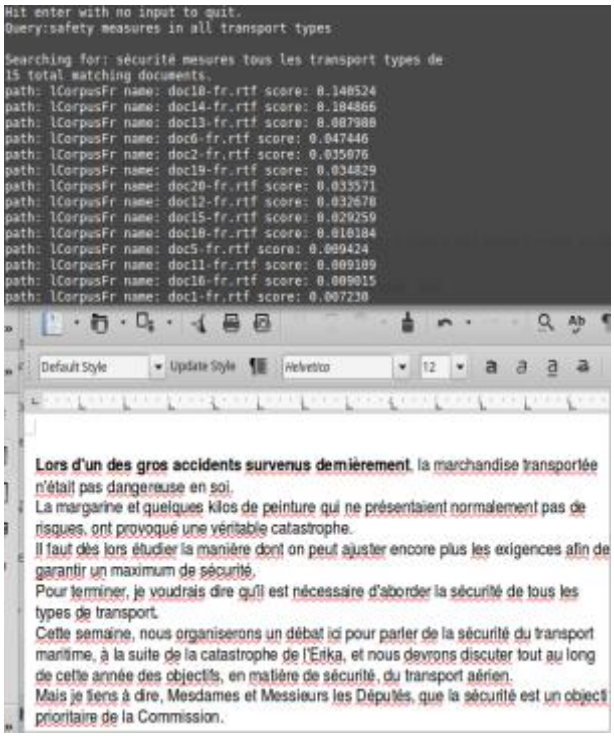
We have used EUROPARL corpus from: Jörg Tiedemann, 2012, Parallel Data, Tools and Interfaces in OPUS. In Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC 2012). The data is free and provides a parallel corpus in around 21 languages from the European parliament website. We have used the corpus and downloaded the French and English documents. And used one big document to generate several smaller documents in both English as well as in French. Testing has been performed on small as well as large documents.

Screenshots of Results:

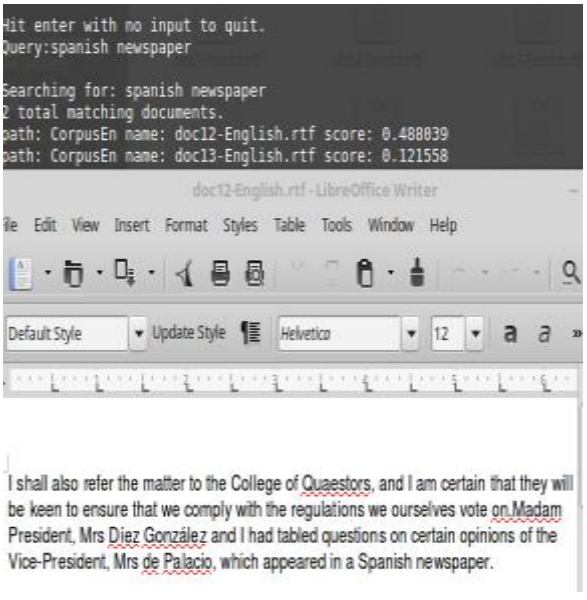
Execute the program through the command line and you will be asked to enter a query (English). You may enter a phrase query or just list of important words.
In the baseline approach, we are just retrieving the English documents with their scores.
Whereas, in the standard approach we are converting the English query into a French query and then retrieving the relevant documents in French.

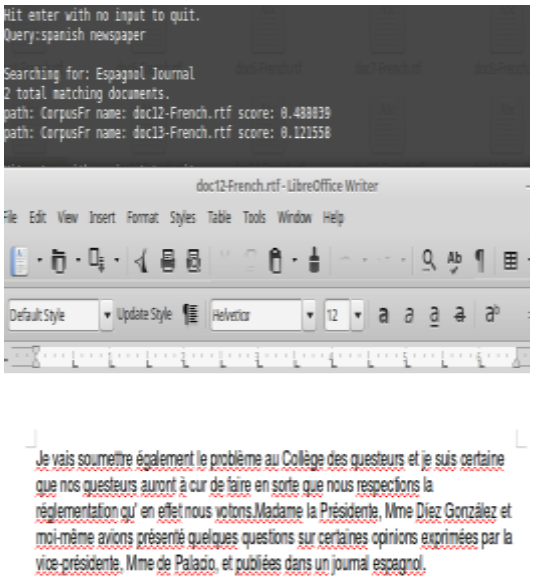
1. Large documents





2. Small documents





Evaluation Results - NDCG and Paired t-test

We used normalized distributive to quantify the accuracy of our cross-language retrieval implementation with the ideal discounted cumulative gain being the rank the baseline (English-only) search engine achieved. With only one correct document for each query the discounted cumulative gain was counted as $1/\log_2(\text{implementationRank} - \text{BaselineRank} + 2)$. The only exceptions were the cases in which our cross-language retrieval model achieved a higher rank than the baseline and when the cross-language retrieval model failed to produce the parallel document, where the NDCG was set to 1.00 and 0 respectively. The average NDCG difference from 1.00 (the perfect result) was used in a paired-t test to find the average difference from the baseline and the statistical likelihood of this difference not being true given as the p-value (less than 0.01 for both query sizes, meaning a greater than 99% confidence interval). The mean differences were 0.16 for the large queries and 0.28 for the small queries. For reference, our implementation getting a rank 1 below the ideal would result in a NDCG difference greater than both of these averages (0.37).

- **Large Documents**

Document	Query	Baseline Rank	Implementation Rank	NDCG	1-NDCG
1	Victims of millenium bug	1	1	1.00	0.00
2	Petition for young man's execution	1	1	1.00	0.00
3	Protection of Barents sea	1	1	1.00	0.00
4	Decision to Renew arms embargo	1	1	1.00	0.00
5	Support for President Prodi	1	2	0.63	0.37
6	Legislative programme	2	1	1.00	0.00
7	Clarity on new and annual debate	1	1	1.00	0.00
8	capital task	2	2	1.00	0.00
9	Results of vote	1	1	1.00	0.00
10	Explanation of votes	1	1	1.00	0.00
11	Penalties for violations	1	Not found	0.00	1.00
12	Imposing strict requirements on tankers	1	1	1.00	0.00
13	Proposal on frost ratings	1	2	0.63	0.37
14	Community regulations in transport	1	4	0.43	0.57
15	Reasons for road accidents	1	2	0.63	0.37
16	Increased volume goods to Europe	1	4	0.43	0.57
17	Flexible and country-specific rules	1	1	1.00	0.00
18	Erika disaster	1	1	1.00	0.00
19	Dangerous goods transportation	6	6	1.00	0.00
20	Difficulties faced due to delay in CEN work	1	1	1.00	0.00

Mean Difference	0.16
Standard Deviation of Difference	0.28
Standard Error of Difference	0.06
T-statistic	2.56
P-value	0.01

- **Small Documents**

Document	Query	Baseline Rank	Implementation Rank	NDCG	1-NDCG
1	Fine in fleet targets	1	1	1.00	0.00
2	Violent deaths	1	1	1.00	0.00
3	Casualties of terrible storms	1	1	1.00	0.00
4	displeasure on fishery policy	1	1	1.00	0.00
5	suit for young man's execution	1	1	1.00	0.00
6	influence in stay of execution	1	Not found	0.00	1.00
7	constitutional rights in Russia	1	1	1.00	0.00
8	Beneficiary involvement in criminal movement findings	1	1	1.00	0.00
9	Safety of Barents Sea	1	1	1.00	0.00
10	rise in channels	2	Not found	0.00	1.00
11	Status on Dutch channel	1	Not found	0.00	1.00
12	reaction on Spanish newspaper	1	1	1.00	0.00
13	ABC news Nov 18th	1	1	1.00	0.00
14	Judgement to refuse arms embargo	1	1	1.00	0.00
15	Final version on rule 110	1	Not found	0.00	1.00
16	Target for next 5 years	1	1	1.00	0.00
17	Prodi's repetition on his commitment	1	1	1.00	0.00
18	Movement for 5 years	3	Not found	0.00	1.00
19	Support for President Prodi	2	4	0.50	0.50
20	settlement on last week's presidents conference	1	1	1.00	0.00

Mean Difference	0.28
Standard Deviation of Difference	0.44
Standard Error of Difference	0.10
T-statistic	2.77
P-value	0.01

Conclusion and Discussion

The project was a success, but we faced difficulties dealing with some issues. The issues, messages, recommendations, limitations are discussed here:

1. Tolerant Retrieval
   Our model had several instances of failing to find the matching document, especially in the smaller documents. This is likely because of the built-in boolean search element of the Lucene similarity class we used. Removing this would allow all documents to appear in the search results and thus find the matching document. On the other side, experiments would need to be run where there is no matching document to find a good threshold cosine similarity value when a user might want to conclude that no matching document is present for their query, assuming they are patient enough to look past the first few results before giving up.

2. Query expansion
   Query Expansion is used to formulate the seed query to improve the retrieval performance. There are three stages to perform query expansion, pre-translation, post translation and both pre- and post-translation. This project can further be extended by applying Query-expansion. Combining pre- and post-translation feedback is most effective and helps in reducing the translation error. However, in our project we thought of applying Pre-translation expansion using Pseudo-relevance feedback which creates a stronger base for translation and improves precision. We can use PyDictionary to get the synonyms of the query-terms. Once we get several queries by expansion, we can then apply mean average precision to find the best query out of all and retrieve the relevant documents.

3. Data collection
   Finding corpus, dictionaries, changing the data to a form which can be used for retrieval and training purpose was very time consuming and challenging. With a little more data we are sure that the project would have been more than successful.

4. Other CLIR approaches
   We used Machine translator in our project, it makes good use of context and only returns one result while Dictionary-based translator return multiple meanings to a term. Machine translator can maintain quality control. We might be able to use our project with comparable corpora but that's not we have designed it for. Parallel Corpora is nothing but the subset of Comparable corpora.

5. The learning curve was very steep, we had to read a lot of literature to understand how a CLIR works, it's different techniques, approaches, expansion etc. PyLucene took about 12-16 hours to troubleshoot the installation of it on linux machine. Lucene seems to be a very powerful tool to master, but the documentation is hard to read and the source code is rather opaque. While the object-oriented design would be very good for making applications, it is frustrating from a research standpoint where one wants to know exactly what is going on to better understand the results.

Individual Contributions

Colin was the code monkey for this assignment due to having more familiarity with Python than Shilpa. He spent many an hour installing Lucene, figuring out how to use Lucene, figuring out how Lucene sort of works, figuring out how Lucene works, and running the experiments, gathering the data and putting together our evaluation metric in Excel.

Shilpa kept the big picture in mind while Colin dealt with the details of implementing the code. She did data preparation, gathering the data from the corpus and changed it to a form to use in our project and for testing purpose. She remembered the conversations with Professor Dietz and kept Colin from doing things the Professor told us not to do like implement computer-generated queries from the English document and other things he started trying to code that weren't useful. Put together the presentation and researched and learned all the big conceptual issues for cross-language information retrieval while reading tons of papers.

Both worked on the final paper, and both think the other did an excellent job and worked extremely hard.

References

[1] http://members.unine.ch/jacques.savoy/clef/frenchST.txt
[2] http://members.unine.ch/jacques.savoy/clef/frenchStemmerPlus.txt
[3] http://snowball.tartarus.org/algorithms/porter/stemmer.html
[4]https://lucene.apache.org/core/6_2_1/core/org/apache/lucene/search/similarities/TFIDFSimilarity.html
[5] https://en.wikipedia.org/wiki/Cross-language_information_retrieval
[6] https://en.wikipedia.org/wiki/Discounted_cumulative_gain
[7] http://dl.acm.org/citation.cfm?id=2379777
[8] http://www.statstutor.ac.uk/resources/uploaded/paired-t-test.pdf