Melange: Components for cross-lingual retrieval

Max Pfingsthorn Koen van de Sande Vladimir Nedovic
University of Amsterdam
{mpfingst, ksande, vnedovic}@science.uva.nl

Abstract

We present the finalized version of our cross-lingual search engine Melange, and results obtained by running it on WebCLEF topics in an attempt to solve Mixed Monolingual and Multilingual tasks. We concentrate on certain features of the system which are relevant to the CLIR field and which can be developed further independently. These are our data extraction and indexing methods, our language detection module (with an accuracy of 88% on WebCLEF query strings), PageRank ranking scheme and query translation.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.1 Content Analysis and Indexing; H.3.3 Information Search and Retrieval; H.3.4 Systems and Software

General Terms

Languages, Measurement, Performance, Experimentation

Keywords

Cross-lingual Information Retrieval, Data Extraction, Language Detection

1 Introduction

We were introduced to the Cross-Language Evaluation Forum (CLEF) during the Internet Information course [1] at the University of Amsterdam, given in Spring 2005 by Dr. Maarten de Rijke and Gilad Mishne. The second half of the course consisted of an 8-week project. As the course lecturers are organizing the Web track of CLEF this year (WebCLEF), it was natural that there was a project aiming at participating at WebCLEF. The course project team consisted of 11 people, including the authors of this paper. Within the team it was decided to focus more on creating features needed for cross-lingual retrieval, than on the engineering task of participating in WebCLEF. In a follow-up project, however, we continued working on the system in order to fulfill the initial project goal of being part of WebCLEF 2005 (the system as delivered at the end of the initial project was not yet suitable for participation).

In this paper, we present the system that was the result of that follow-up project. Distinguishing features of the system are: data extraction methods, language detection module (applicable to both documents and queries), anchor text indexing, PageRank ranking scheme and query translation.

Dictionary creation by alignment of identical documents in multiple languages and our web interface, developed during the initial course project, are not of particular interest for our WebCLEF participation, since they are not used.

In section 2 we will give a technical overview of our system MELANGE[1]. In the following sections we will discuss the distinguishing elements of our system: data extraction methods, language detection method and query translation.

2 System overview

The retrieval part of our system is a modified version of the Terrier system [4]. For each document, we separately index its title, its text, the bold-faced terms within the text and the anchor texts other documents use to refer to it[2].

Every word indexed is given a two-letter prefix in one large index (unfortunately, it was not possible to have separate indexes for each language in case of Terrier). The first letter is used to distinguish between different kinds of data that we index. The second letter is used for the language of the word[3]. Effectively the word prefixes mean that we have one index per language and per data type. We reclassified the entire EuroGOV collection using our language detector (see section 4).

We extended the Terrier document index to also store the PageRank [5] values and the language of a document. The document domain is already encoded in the EuroGOV document identifier.

In order to use the extra information provided in topics, we wrote score modifiers which plug in to the Terrier system to take the PageRank values into account as well as to boost scores of documents of a specified language or domain.

The Terrier data readers were not extended to support the EuroGOV collection. Instead, we let our data extraction scripts (see section 3) convert the data into a TREC format which is already supported in Terrier.

3 Data Extraction

Our data extraction tools are written in Python [9] and accelerated using PsyCo [7]. We created a reusable module is used within all our tools. The module can be used to look up the language of a document and the kind of document (HTML, PDF or Word document)[4]. It also supports operations such as URL extraction (in normalized form), text extraction, etc. All HTML operations are carried out using the Beautiful Soup HTML parsing module [6], which is agnostic to bad HTML[5].

Of particular importance in the CLEF setting is that we correctly extract the codepage of a document from its HTML META header, if present, and otherwise from the HTTP header.

We think that, when redoing language detection on all documents (like we did), the accuracy of the language detected can be much higher if the input text does not contain garbage (such as English JavaScript comments in a Polish document) and the codepage of the document is respected. We think the same holds for the index of our search engine: if we can index data of higher quality, then this can only improve our results.

Our most notable data extraction tools are:

- EuroGOV text extractor which converts the data into a TREC format.

[1] MELANGE stands for Multiple European LANGuage Engine

[2] With a limit to the number of times the exact same anchor text can be used for referral.

[3] Actually, the language of the document the text comes from, since we perform language detection on whole documents. All data fields for a single document are not necessarily in one language, because anchor texts come from other documents.

[4] Only HTML documents are used in this years WebCLEF. Because of this most operations are only supported on HTML pages.

[5] However, it will fail on documents containing binary data, such as Word documents.

- Link structure extractor, which extracts all URLs from an HTML document. Using all these URLs we can build a graph of pages linking to each other in the collection. Based on this graph we can calculate the PageRank value for individual pages [5], which is a measure of how often other pages link to this page.
- Anchor text extractor. Anchor text is the hyperlinked words on a webpage. Texts that other pages use to refer to a page can be a good description of it. The output of this tool can be used to easily lookup all texts used in links to a document. These anchor texts can be in different languages, but this is correctly handled because of the language prefix to words (see section 2).

4 Language Detection

The language detection module primarily needs to be able to detect the (most probable) language of the query that the user has typed into a search engine. This is a highly challenging task since the average query length is between two and three words. Another way in which we can utilize the module is in the form of a language classifier, in order to improve the annotations in the dataset (since for many EuroGOV documents, a set of possible languages is given instead of a single language label).

In our language detector, we make use of character n-grams, in the way they are used in stochastic language modeling. Such language models (LM) define the syntax of a language in a probabilistic way, and are usually acquired through machine learning [2]. The basic idea is to find grammatical models for a language. However, since we are primarily dealing with very short online queries, learning language grammars at a sentence level is not good enough. By using character n-grams for this task, we view our LM as a grammar to generate words instead of sentences.

We choose n-grams of size three and for every word we store tuples of the form (*Context*, *Character*, *Probability*). *Probability* can be calculated as the total number of times we encountered a *Character* together with the *Context*, divided by the total number of times we encountered the *Context*.

For example, from the text 'Test text', we would learn the tri-grams (∧∧, t, 1.0), (∧t, e, 1.0), (te, s, 0.5), (es, t, 1.0), (st, , 1.0), (te, x, 0.5), (ex, t, 1.0) and (xt, , 1.0).

Note that the beginning and end of a word are denoted by special characters, namely '∧' and ' ' (space). This is important when generating words, since we would otherwise never start generating any words or produce infinitely long ones.

Having such tri-gram models, the probability of a word being generated by a language L is calculated assuming partial independence between the tri-grams (the same holds for collections of words, where independence is assumed between terms):

$$P(c_1, c_2, c_3, \ldots, c_n|L) = \prod_{i=1}^{n} P(c_i|c_{i-1}, c_{i-2})$$

Once we have calculated these conditional probabilities, we can assign a class label to an unknown text by choosing the language with the highest probability.

To solve machine precision problems (i.e. when character probabilities become infinitesimally small in large texts), we take the logarithm, and scale if necessary for large documents. To address the problem of zero frequency n-grams, we perform a smoothing on all n-gram probabilities according to the following formula:

$$P(w_n|L_c) = \frac{(n-1)P(w_1, w_2, \ldots, w_{n-1}|L)}{\lambda}$$

Another problem we encounter is due to the nature of the data set. Since the corpus is gathered from the EuroGOV domain, it contains documents with lots of foreign terms, which can make the quality of learned LMs very poor. To tackle this, we explicitly restrict a given language to a set of characters that can occur in its alphabet.

5 Query Translation

In case of WebCLEF's Multilingual task, the language of the query need not be identified by the n-gram module (English translation can be used as the source), but the query needs to be translated into all relevant languages. For this purpose, we need a translating tool, which ideally supports all EU languages. As mentioned in [3], when dealing with cross-language retrieval, one of the biggest problems is that '*resources in terms of parallel corpora or commercial machine translation are very difficult to obtain*'. Whereas we could actually consider some official EU documents as parallel corpora, translating the queries was still a big issue.

For the query translation, we would have preferred to have offline dictionaries or translators, but we could not obtain solutions with a good API. Instead, we use the online translating tool WorldLingo [8]. However, since WorldLingo only offers translations in 9 major European languages (i.e. English, French, German, Dutch, Italian, Spanish, Portuguese, Greek and Russian), the multilingual translation support of our engine is restricted to those languages only.

Encoding is generally a big issue in cross-language IR - in accordance with the dominance of English language, there is '*a lack of standards for character and font representation for many languages*' [3]. In our case, WorldLingo presented us with different encodings based on the language: it uses UTF-8 for Russian, CP1253 for Greek and ISO-8859-1 for all other languages. Unifying these encodings into UTF-16 was not a big problem, but getting the Terrier query parser to accept them was, as it was designed for ASCII characters only.

6 Runs

We submitted runs for the Mixed Monolingual and Multilingual tasks. In the former case, the language of the query is typically identical to that of the target page, whereas in the latter case, the English translation of the query is used, targeting pages in all languages. The test set consisted of 547 topics with the following format:

```
<topic>
  <num>WC0005</num>
  <title>Minister van buitenlandse zaken</title>
  <metadata>
    <topicprofile>
      <language language="NL"/>
      <translation language="EN">dutch minister of foreign affairs</translation>
    </topicprofile>
    <targetprofile>
      <language language="NL"/>
      <domain domain="nl"/>
    </targetprofile>
    <userprofile>
      <native language="NL"/>
      <active language="EN"/>
      <passive language="FR"/>
      <passive_other>Frisian</passive_other>
      <countryofbirth country="NL"/>
      <countryofresidence country="NL"/>
    </userprofile>
  </metadata>
</topic>
```

The topic title is the original query, used in the Monolingual task. Topic profile contains its language label, as well as its English translation, used in the Multilingual task. Target profile indicates the domain in which the target page resides, and the language of that page (which might differ from the official language of the country to which the domain corresponds). User profile tells about user's language preferences. In both tasks, our motivation was to use as little extra data as possible, to test the benefit of using specific pieces of data and/or specific features. In the next subsection we show the results of our language detector on the supplied topics. In the two following subsections, the results for both tasks are discussed.

6.1 Language Detector

Detecting the language of a topic is crucial, because we have one index per language and running a query with a different language than is contained in the index will yield bad results. We ran our language detector on the topic title for all 547 topics. 452 of those were correctly classified, which is 82.6%. Misclassifications of Spanish titles available accounts for 67 of the 95 misclassifications. Accuracy on Spanish is only 50%[6]. Leaving out Spanish would yield an accuracy of 93.2%.

Accuracy on the 547 English translations of the topics gives an accuracy of 93.8%. Combining results of all 1094 query strings gives an accuracy of 88.2%.

6.2 Mixed Monolingual

For a Mixed Monolingual task, we submitted five runs with the following properties:

BaselineMixed A run using only our word-level inverted index. We utilized only our language detector of all the special features. For this run, we expected the worst results of all monolingual runs. This baseline run serves as a reference for all other monolingual runs.

AnchorMixed A run like *BaselineMixed*, but using the additionally collected and indexed anchor text of document's incoming links. In this run (which still only uses the title of the topic), we test the relevancy of our anchor text index, by comparing the results with those obtained in *BaselineMixed*. We expected a slight improvement of results overall.

DomLabelMixed A run in which scores of documents which matched the target profile's domain were boosted. For this run, we used some metadata in addition to the topic's title, namely target profile's indication of the domain to which the target page belongs. This enables us to utilize another feature of our ranking mechanism, in which it is possible to assign different weights to specific domains. Contrary to the *BaselineMixed* run, the language detector was not used in this run: the target language is derived from the domain (with the exception of the *eu.int* domain, where we did use it). Since weighting should result in a certain removal of noise in the final ranking, we expected a significant improvement over the baseline run.

LangLabelMixed Very similar to *DomLabelMixed*, boosting the scores of pages belonging to domains whose native language is the one indicated in a topic profile. In this case, we utilized another piece of metadata (instead of target profile's domain), namely topic profile's language label. Since different domains roughly correspond to specific languages, we explicitly weighted certain pages during ranking again. However, domains can contain pages in various languages and the weights are quite different compared to a *DomLabelMixed* run. The language detector was not used. We expected an improvement in results over the baseline run (following the same reasoning as from *DomLabelMixed*), although not as significant as in the case of *DomLabelMixed*.

LangCueMixed A run in which the query itself gives a hint about the language (or domain) of the target page, giving us the opportunity to boost certain domains again (i.e. there are queries implying the language, or domain, of the target page, and such hints are made explicit within the query's translation into English). We use the English translation of the query (which is part of the metadata) to look for a hint such as 'Dutch'. These hints could be of potential use as a language label. For topics which contain such language hints, this run is equivalent to *LangLabelMixed* (assuming the hint is right, since it is made manually). Otherwise, it simplifies to a *BaselineMixed* run.

Figures 1 and 2 show some interesting properties of our runs. In general, our system performed quite well for Russian, but not for other languages. Since query translation is not an issue, we have to think that this is due to stopping and stemming issues. Our stopword lists were very

[6]Statistics per language: Danish 28/30, Dutch 54/59, English 113/121, French 1/1, German 47/57, Greek 16/16, Hungarian 34/35, Icelandic 5/5, Portuguese 57/59, Russian 30/30, Spanish 67/134.

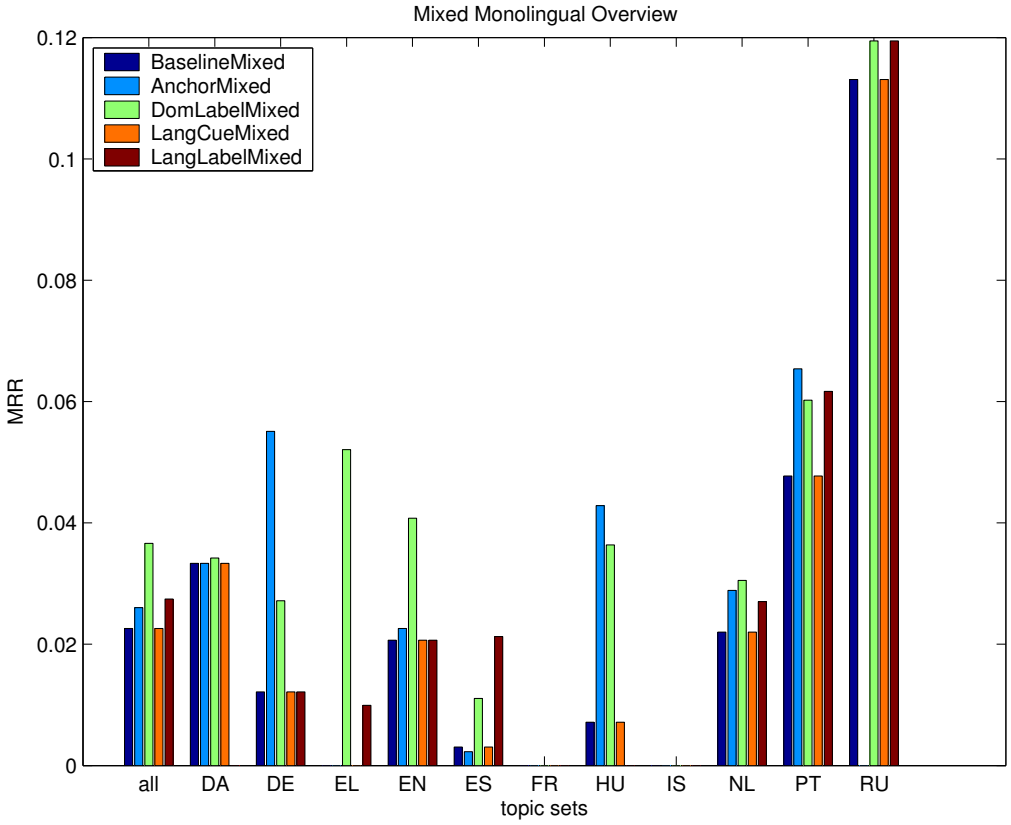

Figure 1: Overview of Mean Reciprocal Rank for Mixed Monolingual runs by topic language, grouped by topic language.

short, leading to more noise in the index. Also, we did not have stemmers for all languages and for some languages it is better not to stem. The Russian Snowball stemmer is probably quite good, considering relative performance. Note that we lacked a stopword list for Russian, so the good results cannot be explained that way.

French and Icelandic did not work well, but with 1 and 5 topics respectively we cannot be certain that the system does not work. For Icelandic there are very few documents in EuroGOV (the IS domain is not indexed). We suspect that not all Icelandic documents that are present are labelled correctly (and thus end up in the wrong index).

For Spanish we can see what disastrous effects a wrong language classification can have: using the supplied language label instead of our detector yields a 5-fold increase in MRR. Only the run which uses the target domain metadata can slightly compensate for the wrongly labelled topics.

Using anchor text generally improves results slightly. For German and Portuguese it improves results significantly, to the extent that it performs better than using the target domain information. For Greek and Russian no relevant results are returned when anchor text is activated.

Looking at Figure 2, we can see that if a relevant page was found within the 50 results allowed for submission, then that page was ranked well within the top 15 results. This encourages us to believe that the main drawbacks of our system currently are rather specific problems with character encodings, stoppers, stemmers and missing data for the language detection; especially a good language detection is vital.

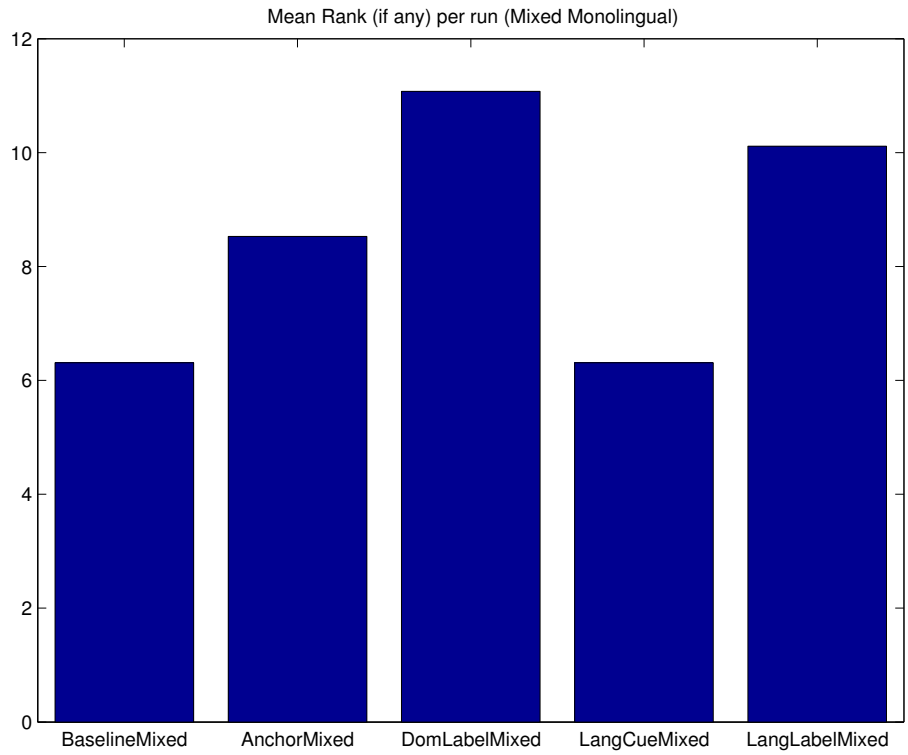

Figure 2: Overview of mean rank (if any) for Mixed Monolingual runs.

6.3 Multilingual

For the Multilingual task, we submitted five runs with the following properties:

BaselineMulti A multilingual run only using the word-level inverted index. This baseline run is a reference for all multilingual runs. Contrary to the monolingual case, the English translation of the query is used instead of the title and it is the source of the translation. The query is translated into all languages supported. For this run, we expected the worst results of all multilingual runs. These results should serve as a reference for other runs.

AnchorMulti A run like *BaselineMulti*, but using the additionally collected and indexed anchor text of document's incoming links. The only difference compared to the baseline run is that we have turned on the usage of anchor text data contained in the index. We expect that using anchor text gives improved results over the baseline run.

AccLangsMulti A run like *BaselineMulti*, but scores of documents whose domain language matched one of those acceptable (readable) by the user were boosted. For this run, the user profile part of the metadata was used as an indication of which languages the user can speak or understand. Those acceptable languages impose a restriction on the target of the translation. The scores of domains with those languages were given higher weights during ranking. As in the previous case, the language (actually domain) restriction is expected to reduce the noise in the results.

LangCueMulti Just like in the monolingual case, a run in which the query itself gives a hint about the language or the domain of the target page, giving us the opportunity to boost certain domains again. Here, we utilized the English translation of the query, which is part of the metadata. For topics in which the hint was found, the obtained language was used

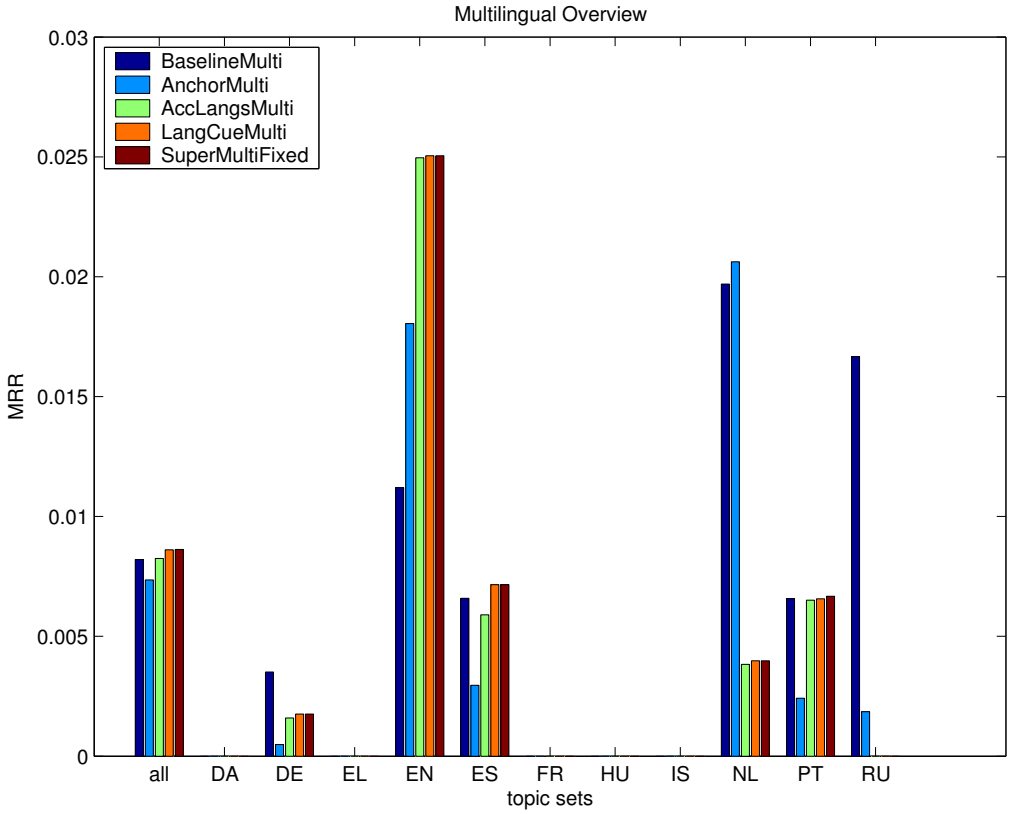

Figure 3: Overview of MRR for Multilingual runs by topic language.

as a source for translation, and otherwise it was the English translation (in which case this run simplifies to the *BaselineMulti* run described above). In case the hint is found, this run should produce 'cleaner' results because of the language restriction, and thus results should be slightly better than in the baseline case.

SuperMulti A run which uses all features described above, with the exception of the language detector. Contrary to the baseline run, which is supposed to be a lower-bound reference in terms of results, this run uses all the available features and is thus expected to produce the best results of all multilingual runs.

In Figures 3 and 4, we can see that our system did not perform very well on the Multilingual task. This was mainly due to the earlier mentioned translation limitation of the web service we used. Again, as in the Mixed Monolingual case, we have no results for French. The other languages which are missing results were just not supported by the translation service.

Just as in the Monolingual case, Russian combined with anchor texts does not perform well. Using either our language hints module or the user language profile can hurt results badly, as can be seen with Dutch and Russian. Only for English they provide a significant gain.

The reason for the relatively low MRR for German may be the possible different spellings of the same word. Some words may use 'ss' while others may use 'ß'. The same can occur with umlauts.

Also, from Figure 4, we can see that if we did find a relevant page within the first 50 results, we did usually within the first two fifths. However, this is pretty close to one half, so it might not be a significant difference.

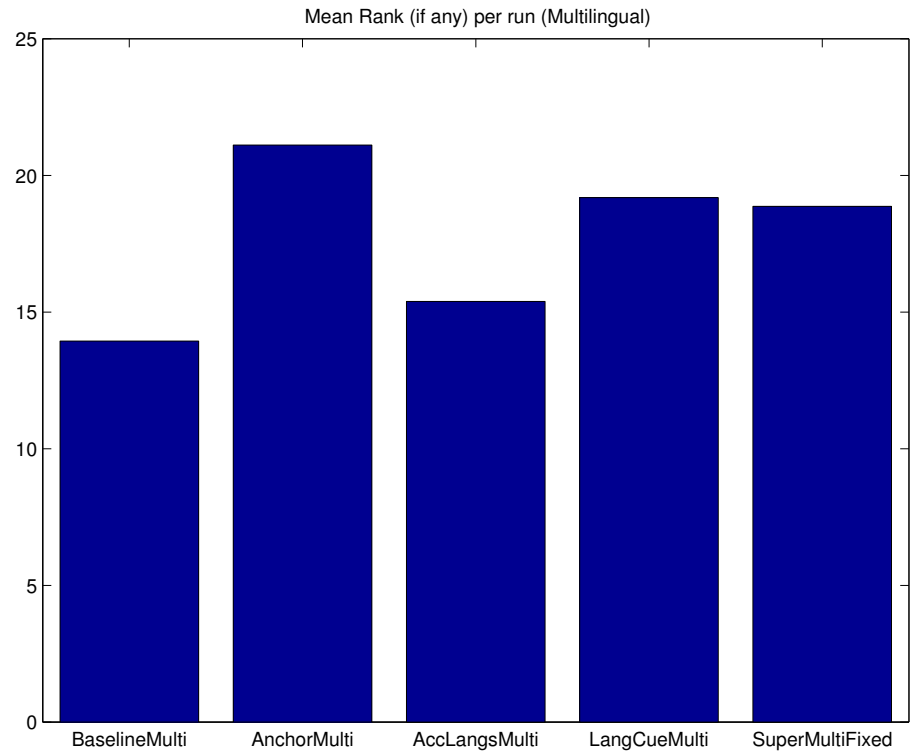

Figure 4: Overview of mean rank (if any) for Multilingual runs.

7 Conclusions and future work

In this work, we present the finalized version of our cross-lingual search engine Melange, as well as results obtained after running it on this year's WebCLEF topics in Mixed Monolingual and Multilingual categories. The emphasis of the paper is on certain components of the system, since we believe that some of them are our biggest contribution to the field of Cross-Language Information Retrieval. These components include our tools for data extraction and indexing (especially for anchor text), our language detection module, PageRank ranking scheme and query translation. These features of our system can be further utilized, as they proved to be quite robust and to work well, even though the obtained results for the entire system are not as good as we expected. The primary reasons are limitations of our translating tools, stoplists, stemmers and for some languages the language detector. Character encoding is a standard problem in the CLIR community which is by no means solved. Our special attention to this problem does mean that we follow most common encoding standards, but we cannot address it perfectly.

Therefore, in future work it would be worthwhile to look into encoding issues more closely, and try to find a more elegant way to index Unicode in a retrieval engine. This could mean switching from Terrier to another indexing engine, as most of our tools do not depend on it. We could also try to obtain more language resources, possibly exploiting the parallel corpora of EuroGOV to create multiple dictionaries (dictionaries for three languages only have already been developed during the class project). Our language detector could be trained with more data and made robust more for all required languages (such as Spanish).

8 Acknowledgements

We would like to thank our instructors, Maarten de Rijke and Gilad Mishne, for their information-rich course on search engines. Our workload had never been that high before. We would like to thank the original team for their efforts on Melange: Aron Abbo, Maarten Corzilius, Sebastian Eigenmann, Felix Hageloh, Peter van der Meer, Bayu Slamet, Roberto Valenti and Janneke van der Zwaan.

References

[1] Maarten de Rijke and Gilad Mishne. Lecture Internet Information. `http://ilps.science.uva.nl/Teaching/0405/II/`.

[2] Ted Dunning. Statistical identification of language. Technical Report MCCS-94-273, New Mexico State University, 1994.

[3] Fredric Gey, Noriko Kando, and Carol Peters. Cross language information retrieval: a research roadmap. *SIGIR Forum*, 36(2):72–80, 2002.

[4] University of Glasgow. Terrier. `http://ir.dcs.gla.ac.uk/terrier/`.

[5] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[6] Leonard Richardson. Beautiful Soup - Python HTML/XML parser. `http://www.crummy.com/software/BeautifulSoup/`.

[7] Armin Rigo. Psyco - Python Optimizer. `http://psyco.sourceforge.net/`.

[8] WorldLingo Translations. Worldlingo. `http://www.worldlingo.com/`.

[9] Guido van Rossum. Python. `http://www.python.org/`.