

GIR experiments with Forostar at GeoCLEF 2007

Simon Overell¹, João Magalhães¹ and Stefan Rüger^{2,1}

¹Multimedia & Information Systems

Department of Computing, Imperial College London, SW7 2AZ, UK

²Knowledge Media Institute

The Open University, Milton Keynes, MK7 6AA, UK

{simon.overell101, j.magalhaes} @imperial.ac.uk and s.rueger@open.ac.uk

Abstract

In this paper we describe our Geographic Information Retrieval experiments with Forostar, our GIR application on the GeoCLEF 2007 corpus and query set. We compare the results from orthogonal *text with no geographic entities* and *only geographic entities* with standard text retrieval and combined text and geographic relevance methods. The text and named entity analysis and retrieval methods of Forostar are described in detail. We also detail our placename disambiguation and geographic relevance ranking methods.

The paper concludes with an analysis of our results including significance testing where we show our baseline method, in fact, to be best. Finally we identify weaknesses in our approach and ways in which the system could be optimised and improved.

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

Measurement, Performance, Experimentation

Keywords

Geographic Information Retrieval, Relevance Ranking, Disambiguation

1 Introduction

This paper describes the experiments performed by the Multimedia and Information Systems group at GeoCLEF 2007 with our GIR application: Forostar. We compare the results from orthogonal *text with no geographic entities* and *only geographic entities* with standard text retrieval and combined text and geographic relevance methods.

In Section 2 we outline how we index the GeoCLEF corpus and the three field types: Text, Named Entity and Geographic. We then describe how the manually constructed queries are expanded and submitted to the query engine. Section 3 describes and justifies the placename disambiguation methods and geographic relevance ranking methods in more detail. In Section 4 we describe our experiments followed by the results in Section 5. Finally Section 6 analyses the weaknesses of our system and identifies areas for improvement.

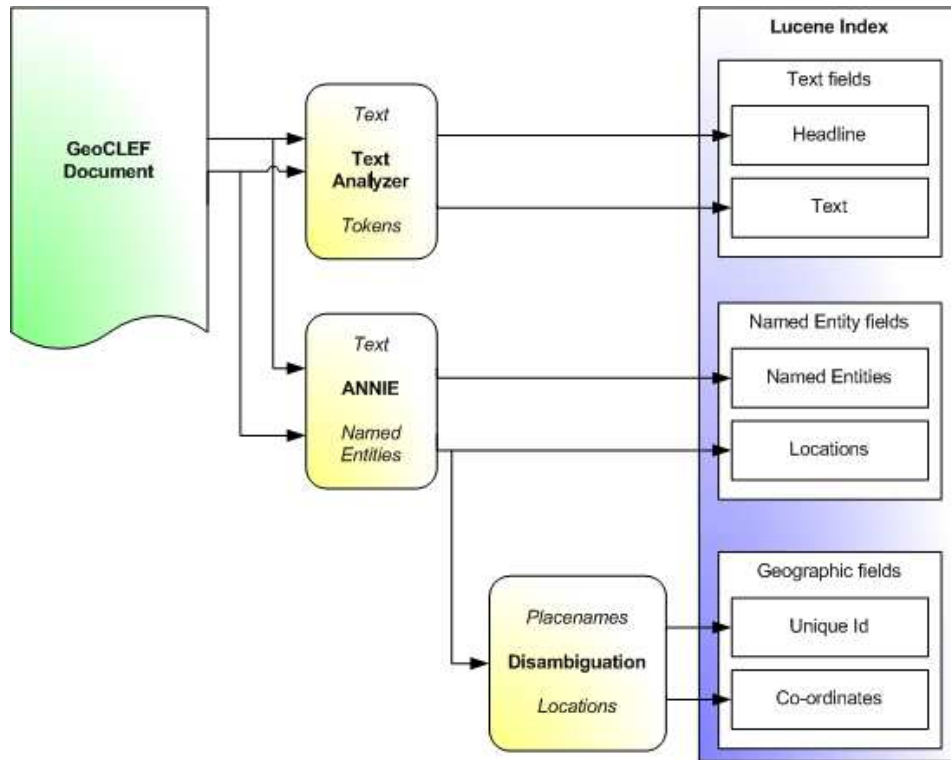


Figure 1: Building the Lucene Index

2 System

Forostar is our ad-hoc Geographic Information Retrieval system. At indexing time, documents are analysed and named entities extracted. Named entities tagged as locations are then disambiguated using our co-occurrence model. The free-text fields, named entities and disambiguated locations are then indexed by Lucene. In the querying stage we combine the relevance scores assigned to the Geographic fields and Textual fields using the vector space model. Fields designated as containing more information (i.e. The Headline) have a *boost* value assigned to them.

2.1 Indexing

The indexing stage of Forostar begins by extracting named entities from text using ANNIE, the Information Extraction engine bundled with GATE. GATE is Sheffield University’s General Architecture for Text Engineering [2]. Of the series of tasks ANNIE is able to perform, the only one we use is named entity recognition. We consider ANNIE a “black box” where text goes in, and categorised named entities are returned; because of this, we will not discuss the workings of ANNIE further here but rather refer you to the GATE manual [2].

2.1.1 Named Entity fields

We index all the named entities categorised by GATE in a “Named Entity” field in Lucene (e.g. “Police,” “City Council,” or “President Clinton”). The named entities tagged as Locations by ANNIE we index as “Named Entity – Location” (e.g. “Los Angeles,” “Scotland” or “California”) and as a Geographic Location (described in Section 2.1.3). The body of the GeoCLEF articles and the article titles are indexed as text fields. This process is described in the next section.

2.1.2 Text fields

Text fields are pre-processed by a customised analyser similar to Lucene’s default analyser [1]. Text is split at white space into tokens, the tokens are then converted to lower case, stop words discarded and stemmed with the “Snowball Stemmer”. The processed tokens are held in Lucene’s inverted index.

2.1.3 Geographic fields

The locations tagged by the named entity recogniser are passed to the disambiguation system. We have implemented a simple disambiguation method based on heuristic rules. For each placename being classified we build a list of candidate locations, if the placename being classified is followed by a referent location this can often cut down the candidate locations enough to make the placename unambiguous. If the placename is not followed by a referent location or is still ambiguous we disambiguate it as the most commonly occurring location with that name.

Topological relationships between locations are looked up in the Getty Thesaurus of Geographical Names (TGN) [4]. Statistics on how commonly different placenames refer to different locations and a set of synonyms for each location are harvested from our Geographic Co-occurrence model, which in turn is built by crawling Wikipedia [8].

Once placenames have been mapped to unique locations in the TGN, they need to be converted into *Geographic fields* to be stored in Lucene. We store locations in two fields:

- **Coordinates.** The coordinate field is simply the latitude and longitude as read from the TGN.
- **Unique strings.** The unique string is the unique id of this location, preceded with the unique id of all the parent locations, separated with slashes. Thus the unique string for the location “London, UK” is the unique id for London (7011781), preceded by its parent, Greater London (7008136), preceded by its parent, Britain (7002445)... until the root location, the World (1000000) is reached. Giving the unique string for London as 1000000\1000003\7008591\7002445\7008136\7011781.

Note the text, named entity and geographic fields are **not orthogonal**. This has the effect of multiplying the impact of terms occurring in multiple fields. For example if the term “London” appears in text, the token “london” will be indexed in the text field. “London” will be recognised by ANNIE as a Named Entity and tagged as a location (and indexed as Location Entity, “London”). The Location Entity will then be disambiguated as location “7011781” and corresponding geographic fields will be added.

Previous experiments conducted on the GeoCLEF data set in [7] showed improved results from having overlapping fields. We concluded from these experiments that the increased weighting given to locations caused these improvements.

2.2 Querying

The querying stage of Forostar is a two step process. First manually constructed queries are expanded and converted into Lucene’s bespoke querying language; then we query the Lucene index with these expanded queries and perform blind relevance feedback on the result.

2.2.1 Manually constructed query

The queries are manually constructed in a similar structure to the Lucene index. Queries have the following parts: a text field, a Named Entity field and a location field. The text field contains the query with no alteration. The named entity field contains a list of named entities referred to in the query (manually extracted). The location field contains a list of location – relationship pairs. These are the locations contained in the query and their relationship to the location being searched for.

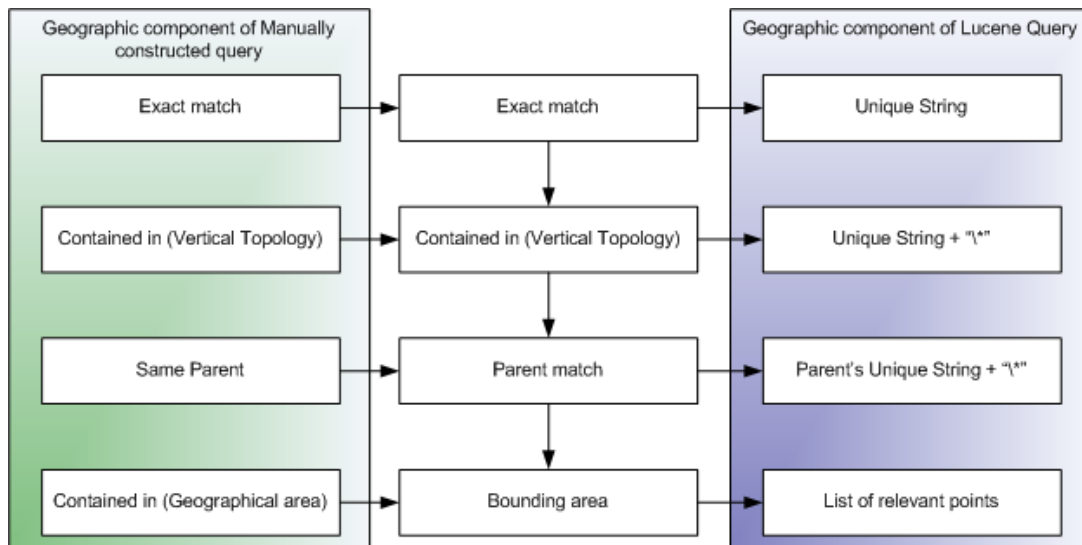


Figure 2: Expanding the geographic queries

A location can be specified either with a placename (optionally disambiguated with a referent placename), a bounding box, a bounding circle (centre and radius), or a geographic feature type (such as “lake” or “city”). A relationship can either be “exact match,” “contained in (vertical topology),” “contained in (geographic area),” or “same parent (vertical topology)”. The negation of relationships can also be expressed i.e. “excluding,” “outside,” etc.

We believe such a manually constructed query could be automated with relative ease in a similar fashion to the processing that documents go through when indexed. This was not implemented due to time constraints.

2.2.2 Expanding the geographic query

The geographic queries are expanded in a pipeline. The location – relation pairs are expanded in turn. The relation governs at which stage the location enters the pipeline. At each stage in the pipeline the geographic query is added to. At the first stage an exact match for this location’s unique string is added: for “London” this would be `1000000\1000003\7008591\7002445\7008136\7011781`. Then places within the location are added, this is done using Lucene’s wild-card character notation: for locations in “London” this becomes `1000000\1000003\7008591\7002445\7008136\7011781*`. Then places sharing the same parent location are added, again using Lucene’s wild-card character notation. For “London” this becomes all places within “Greater London,” `1000000\1000003\7008591\7002445\7008136*`. Finally the coordinates of all the locations falling *close* to this location are added. A closeness value can manually be set in the location field, however default values are based on feature type (default values were chosen by the authors). The feature listed in the Getty TGN for “London” is “Administrative Capital,” the default value of closeness for this feature is 100km.

2.2.3 Combining using the VSM

A Lucene query is built using the text fields, named entity fields and expanded geographic fields. The text field is processed by the same analyzer as at indexing time and compared to both the text and headline fields in the Lucene index. We define a separate boost factor for each field. These boost values were set by the authors during initial iterative tests (they are comparable to similar weighting in past GeoCLEF papers [6, 9]). The headline had a boost of 10, the text a boost of 7, named entities a boost of 5, geographic unique string a boost of 5 and geographic coordinates

a boost of 3. The geographic, text and named entity relevance are then combined using Lucene’s Vector Space Model.

We perform *blind relevance feedback* on the text fields only. To do this the whole expanded query is submitted to the Lucene query engine, and the top 10 documents considered relevant. The top occurring terms in these documents with more than 5 occurrences are added to the text parts of the query. A maximum of 10 terms are added. The final expanded query is re-submitted to the query engine and our final results are returned.

3 Geographic retrieval

Forostar allows us to perform experiments on placename disambiguation and geographic relevance ranking. Our geographic model represents locations as points. We choose a point representation over a more accurate polygon representation for several reasons: It makes minimal appreciable difference for queries at the small (city or county) scale; Eigenhofer and Mark’s *topology matters metrics refine* premise [3], suggests that for queries of a larger scale than city or county topology is of greater importance than distance; and far more point data is available. We represent each location referred to in a document with a single point rather than constructing an encompassing footprint because, we argue, if several locations are referred to in a document does, it does not imply locations occurring between the referenced locations are relevant.

3.1 Placename disambiguation

As discussed in Section 2.1.3 our placename disambiguation is performed using simple heuristic rules. A key part of the disambiguation is our default gazetteer, the generation of which is explained in this section. The default gazetteer is used to disambiguate placenames that are not immediately followed by a referent placename. The default gazetteer is a many-to-one mapping of Placenames to locations (i.e. for every placename there is a single location). We extract our default gazetteer from a co-occurrence model built from Wikipedia.

Our Geographic co-occurrence model contains a mapping of Wikipedia articles to locations in the TGN. It also contains the placenames used to refer to every article describing a location (extracted from anchor texts). In total we crawled 2.3 million links from Wikipedia to articles describing locations. This gave us a mapping of 75,322 placenames to 53,643 locations. The default gazetteer contains these 75,322 placenames mapped to a subset of the TGN. A full description and analysis of the co-occurrence model can be found in [8].

The motivation of this disambiguation method is to provide a baseline of placename disambiguation achievable with our co-occurrence model. Analysis of the co-occurrence model suggests its application should recognise $\sim 75\%$ of locations with an accuracy of between $\sim 80\%$ and $\sim 90\%$. The unrecognised $\sim 25\%$ of locations will only be indexed as “Named Entity – Location.”

3.2 Geographic relevance

In Section 2.1.3 our geographic relevance strategy is described. In this section we provide a justification for the methods used. We have 4 types of geographic relations each expanded differently:

- ‘Exact Match,’ the motivation behind this is the most relevant documents to a query will mention the location being searched for;
- ‘Contained in (Vertical Topology)’ assumes locations within a location being searched for are relevant, for example ‘London’ will be relevant to queries which search for ‘England’;
- Locations that share the same parent, these locations are topologically close. For example a query for ‘Wales’ would have ‘Scotland’, ‘England’ and ‘Northern Ireland’ added;
- The final method of geographic relevance is defining a viewing area, all locations within a certain radius are considered relevant.

Table 1: Mean Average Precision of our four methods

Text	0.185
TextNoGeo	0.099
Geo	0.011
Text+Geo	0.107

Each geographic relation is considered of greater importance than the following one. This follows Egenhofer and Mark’s ‘Topology Matters, Metrics Refine’ premise. The methods of greater importance are expanded in a pipeline as illustrated in Figure 2. The expanded query is finally combined in Lucene using the Vector space model.

4 Experiments

We compared four methods of query construction. All methods query the same index.

- **Standard Text (Text).** This method only used the standard text retrieval part of the system. The motivation for this method was to evaluate our text retrieval engine and provide a baseline.
- **Text with geographic entities removed (TextNoGeo).** For this method we manually removed the geographic entities from the text queries to quantify the importance of ambiguous geographic entities. The results produced by this method should be orthogonal to the results produced by the *Geo* method.
- **Geographic Entities (Geo).** The *Geo* method uses only the geographic entities contained in a query, these are matched ambiguously against the named entity index and unambiguously against the geographic index. Ranking is performed using the geographic relevance methods described in Section 3.2.
- **Text and geographic entities (Text + Geo).** Our combined method combines elements of textual relevance with geographic relevance using the vector space model. It is a combination of the *Text* and *Geo* methods. Our hypothesis is that it will show an improvement over the other tested methods.

Our hypothesis is that a combination of Text and Geographic relevance will give the best results as it uses the most information to discover documents relevant to the query. The Standard Text method should provide a good baseline to compare this hypothesis against and the orthogonal *Geo* and *TextNoGeo* entries should help us interpret where the majority of the information is held.

5 Results

The experimental results are displayed in Table 1. Surprisingly the *Text* result is the best with a confidence greater than 99.95% using the Wilcoxon signed rank test [5]. The *Text+Geo* method is better than the *TextNoGeo* method with a confidence greater than 95%. The *Geo* results are the worst with a confidence greater than 99.5%.

74.9% of named entities tagged by ANNIE as locations were mapped to locations in the default gazetteer. This is consistent with the prediction of $\sim 75\%$ made in Section 3.1.

Some brief observations of the per query results shows that the *Text+Geo* results are better than *Geo* in all except 1 case, while the *Text* results are better in all except 2 cases. The largest variation in results (and smallest significant difference) is the *Text+Geo* and the *TextNoGeo* results.

6 Conclusions

Surprisingly the *Text* method achieved significantly better results than the combination of textual and geographic relevance. We attribute the relatively poor results of the *Text+Geo* method to the way the textual and geographic relevance were combined.

The separate types of geographic relevance and the textual relevance were all combined within Lucene's vector space model with no normalisation. The motivation behind this was that using Lucene's term boosting we should be able to give greater weighting to text terms. The difference in information between the *Text+Geo* method and *Text* method are captured in the *Geo* method. Observations of the per query results shows that in cases where the *Geo* method performed poorly and the *Text* method performed well the *Text+Geo* method performed poorly. The intention of combining the two methods was to produce synergy, however, in reality the *Geo* method undermined the *Text* results.

The *geo* method alone performed poorly compared to the other methods. However, when considering the only information provided in these queries is geographic information (generally a list of placenames), the results are very promising. The highest per query result achieved by the *geo* method had an average precision of 0.097. Further work is needed to evaluate the accuracy of the placename disambiguation. Currently we have only quantified that 74.9% of locations recognised by ANNIE are disambiguated. We have not yet evaluated the disambiguation accuracy or the proportion of locations that are missed by ANNIE.

In future work we would like to repeat the combination experiment detailed in this paper however separating the geographic relevance and textual relevance into two separate indexes. Similarity values with respect to a query could be calculated for both indexes, normalised and combined in a weighted sum. A similar approach to this was taken in GeoCLEF 2006 by Martins et al. [6].

References

- [1] Apache Lucene Project. <http://lucene.apache.org/java/docs/>. Accessed 1 August 2007, 2007.
- [2] H. Cunningham, D. Maynard, V. Tablan, C. Ursu, and K. Bontcheva. Developing language processing components with GATE. Technical report, University of Sheffield, 2001.
- [3] M. Egenhofer and D. Mark. Naive geography. In *Proceedings of COSIT*, 1995.
- [4] Patricial Harping. *User's Guide to the TGN Data Releases*. The Getty Vocabulary Program, 2.0 edition, 2000.
- [5] David Hull. Using statistical testing in the evaluation of retrieval experiments. In *Annual international ACM SIGIR Conference*, pages 329–338, 1993.
- [6] Bruno Martins, Nuno Cardoso, Marcirio Chaves, Leonardo Andrade, and Mio Silva. The University of Lisbon at GeoCLEF 2006. In *Working Notes for the CLEF Workshop*, 2006.
- [7] Simon Overell, João Magalhães, and Stefan R uger. Forostar: A system for GIR. In *(to appear) Lecture Notes from the Cross Language Evaluation Forum 2006*, 2007.
- [8] Simon Overell and Stefan R uger. Geographic co-occurrence as a tool for GIR. In *(to appear) CIKM Workshop on Geographic Information Retrieval*, 2007.
- [9] Miguel E. Ruiz, Stuart Shapiro, June Abbas, Silvia B. Southwick, and David Mark. UB at GeoCLEF 2006. In *Working Notes for the CLEF Workshop*, 2006.