INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Data-Based Knowledge Acquisition

Text Information Retrieval

This project aims at building a search engine. The implementation has to be evaluated on a provided dataset. More precisely, three versions of the project are proposed. Before their description, a bunch of natural language processing (NLP) methods and tools which can be used in the project is given.

Some NLP methods and tools (non exhaustive list)

- a lot of segmenters or tokenizers are available; so are stopword lists for several languages (e.g., at http://torvald.aksis.uib.no/corpora/1999-1/0042.html, or Jean Véronis's list but several others exist);
- stemmers or morphological analyzers: Lovins's, Porter's or Paice-Huster's stemmers; Flemm (morphological analyzer for French; F. Namer's website);
- part-of-speech taggers, with or without lemmatization: TreeTagger, Brill, etc.;
- synonyms or paradigmatically related lexical units: WordNet (univ. Princeton or Java version at source.net/projects/jwordnet); the Roget's thesaurus, GREYC's dictionary of synonyms (Caen), etc.;
- corpus-based paradigmatic relation acquisition, using non supervised machine learning techniques; corpus-based semantic relation extraction using ILP, etc.;
- complex term extraction: from French or English textual data, Acabit (B. Daille LINA Nantes), Ana (C. Enguehard LINA Nantes), Lexter (D. Bourigault ERSS Toulouse) and a more extended version Syntex.

1 Version 1 - Study of an existing search engine and comparative evaluation

Lucene (http://lucene.apache.org) or Terrier (http://terrier.org) are, together with the Lemur project (www.lemurproject.org), the most popular open source search engines for textual data. The objectives of this project are:

a- to run a first standard version of either Lucene or Terrier;

b- to add to the basic functionalities of the Java library of the chosen IR system some linguistic knowledge and to evaluate the gain or loss on a collection of documents;

b_alter.- to modify the standard functionalities of the Java library of the chosen IR system (weightings, etc.) and to evaluate the gain or loss on a collection of documents.

What is expected here is a precise methodology for the comparative evaluation of two versions of the system. One of the collections available on the website of the University of Glasgow (e.g., the CISI collection) can be used for evaluation http://ir.dcs.gla.ac.uk/resources/test_collections/.

2 Version 2 - Non guided implementation of an IR system and evaluation

In this version of the project, the aim is to develop a VSM-based IR system from scratch, integrating some linguistic information (e.g., stemming or whatever you want). The search engine has to be evaluated on one of the collections available on the website of the University of Glasgow: http://ir.dcs.gla.ac.uk/resources/test_collections/ (e.g., the CISI collection). A special focus on the development of an evaluation procedure is expected.

3 Version 3 - Guided implementation of an IR system and evaluation

Here, a guided implementation of a VSM-based IR system is proposed. The system also has to be evaluated on a provided collection.

3.1 Available data

A compressed version of the three files: CISI.ALLnettoye, CISI.QRY, and CISI.REL (around 1.3MB), is provided under Moodle (Data directory). The first file is a collection of about 1460 books and papers descriptions; the second one is a set of 112 queries related to that collection; the last one contains the relevance judgments for the documents wrt the queries (collection available at http://ir.dcs.gla.ac.uk/resources/test_collections/). A stopword list (motsvides.txt) is also present. Copy the files.

Documents are separated from their neighbors in the collection with a special marker; they all have a personal number, and are composed of a title and an abstract. A sample is given and explicited below.

```
.I 1
18 Editions of the Dewey Decimal Classifications
   The present study is a history of the DEWEY Decimal
Classification.  The first edition of the DDC was published
in 1876, the eighteenth edition in 1971, and future editions
will continue to appear as needed.  In spite of the DDC's
long and healthy life, however, its full story has never
been told.  There have been biographies of Dewey
that briefly describe his system, but this is the first
attempt to provide a detailed history of the work that
more than any other has spurred the growth of
```

```
librarianship in this country and abroad.
.I 2
Use Made of Technical Libraries
This report is an analysis of 6300 acts of use
in 104 technical libraries in the United Kingdom.
Library use is only one aspect of the wider pattern of
information use.  Information transfer in libraries is
restricted to the use of documents.  It takes no
account of documents used outside the library, still
less of information transferred orally from person
to person.  The library acts as a channel in only a
proportion of the situations in which information is
transferred.
Taking technical information transfer as a whole,
there is no doubt that this proportion is not the
major one.  There are users of technical information -
particularly in technology rather than science -
who visit libraries rarely if at all, relying on desk
collections of handbooks, current periodicals and personal
contact with their colleagues and with people in other
organizations.  Even regular library users also receive
information in other ways.
```

- **.I**: indicates a new document; the following number is the document identifier;
- the title and the abstract compose the rest of the text and will be considered together.

3.2 Work to do

The 3 following steps have to be done, using your favorite programming language. A preliminary reflection about the most relevant (global) data structuring is necessary.

3.2.1 Step 1: collection indexing

During the indexing step, a stemming algorithm can be used or not (e.g., Martin F. Porter's stemmer: http://www.tartarus.org/martin/PorterStemmer/, or any other one).

For each document, identified by its .I field, the indexing of both title and abstract is produced through:

1. a tokenization: this treatment isolates words within the sequences of letters and symbols in the document. All the spaces and punctuation marks will be considered as word delimiters (whatever the errors);
2. a comparison with the stopword list: each word identified through the tokenization step has to be filtered by a stopword list. Only words that do not appear in that list are kept (mind the character case when comparing);

3. a choice of indexing terms: elaborate a policy to choose the indexing terms among all the distinct words or stems (all the terms, only the most frequent ones, only the most discriminative ones, etc.). This choice mostly relies on a count of occurrences;

4. two different weightings for terms in the documents: the two weightings that will be compared are:

   - (a form of) the normalized frequency: $w(t_i) = tf(t_i)/max(tf(t))$, with $tf(t_i)$ the frequency of term t_i in document d; $max(tf(t))$ is the hightest term frequency in that document;
   - (a form of) $tf * idf$: $w(t_i) = tf(t_i) * log\frac{N}{df(t_i)}$, with N the size of the collection (1460) and $df(t_i)$ the number of documents in which t_i occurs;

5. the production of a representation vector for each document (a list of the weighted indexing terms). Note that this item has to be considered together with the following one; moreover, an alphabetically organized list may be relevant, depending on the data structure chosen;

6. the production of inverted files for your 2 weighting systems, for efficiency reasons. Thus an inverted file, corresponding to the structure *indexing term → list of couples (document containing this term, weight)*, has to be produced from the above representation.

3.2.2 Step 2: querying

In order to query each type of representation elaborated in the previous step within a VSM system, two phases have to be carried on: first queries of the CISI.QRY file have to be indexed the same way; then answers (i.e., document numbers) have to be produced for each query (and for each IR system; one per representation). The two following points have thus to be tackled:

1. query indexing: to index the queries, use the same methodology and the same weighting schemes as for documents.

2. implementation of a search engine to answer the queries: a similarity measure between the indexing vectors of the queries and the documents has to be chosen and implemented. The expected output is a ranked list of documents calculated by the system in response to a given query.

3.3 Step 3: evaluation

The performance of the two IR systems (i.e., the system with the 2 different weighting schemes) has to be evaluated and compared, not for a single query but for the provided set of queries (the performance on one single question randomly chosen cannot give any information on a representative behavior of the systems). Relying of the relevance judgments in the file CISI.REL, program an evaluation method for your systems; present and discuss the different results obtained in the final report.

Note: in the CISI.REL file, the relevance judgments are binary whereas your IR systems rank the documents according to a relevance (similarity) score.

3.4 If time is available...

If time is still available, you can compare the performance of your current systems to that of the same systems to which linguistic knowledge is added.