Information Retrieval Baselines for the ResPubliQA Task*

Joaquín Pérez-Iglesias, Guillermo Garrido, Álvaro Rodrigo, Lourdes Araujo, Anselmo Peñas
Dpto. Lenguajes y Sistemas Informáticos, UNED
{joaquin.perez, ggarrido, alvarory, lurdes, anselmo}@lsi.uned.es

Abstract

This paper describes the baselines proposed for the ResPubliQA 2009 task. These baselines are purely based on information retrieval techniques. The selection of an adequate retrieval model that fits the specific characteristic of the supplied data is considered as a core part of the task. Applying a not adequate retrieval function would return a subset of paragraphs where the answer could not appear, and thus the posterior techniques applied in order to detect the answer within the subset of candidates paragraphs will fail. In order to check the ability to retrieve the right paragraph by a pure information retrieval approach, two baselines are proposed. Both of them use the Okapi-BM25[3] ranking function, with and without a stemming pre-process. The main aim was to prove how well can a pure information retrieval system perform on this task.

Categories and Subject Descriptors

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

General Terms

IR4QA, Experimentation

Keywords

Question answering, Information retrieval for QA

1 Overview

This year the ResPubliQA proposed the challenge of returning the right passage, that contains the correct answer to a question from a collection with more than a million of paragraphs. The supplied collection was based on JRC-Acquis about EU documentation[1], questions and documents were translated to different EU languages.

The main aim to design and run baselines based on IR techniques is to check which results can be obtained with a system of these characteristics, and to establish a starting point for other participants in this task.

*This paper corrects a minor mistake found in the results of the not stemmed baseline. The previous results were slightly lower than those presented here.

[1] http://langtech.jrc.it/JRC-Acquis.html

Related works can be found on previously international competitions and workshops as NTCIR-7[4], TREC Genomics Track 2006[1] and 2007 and in the workshops IR4QA carried out within COLING 2008 and SIGIR 2004.

Different techniques can be found on the previous works specifically focused on applying classical retrieval models, to the selection of paragraphs or snippets within a document. In general these techniques are adapted to the data characteristics moreover of the straight use of the model.

The proposed baselines can be considered as a first phase within a typical pipeline architecture for question answering. That is, a first selection of paragraphs that are considered relevant for the proposed question are selected. Therefore the focus is to obtain a first set of paragraphs ordered according to their relevance with the question. The precision in terms of retrieving a correct answer for the question within the top k paragraphs, delimits in some sense the overall quality of the full system. In order to retrieve the most relevant paragraphs, the full collection has been indexed by paragraphs removing stopwords and applying a stemmer when there is some available. Both process are performed specifically by language.

2 Retrieval Model

The selection of an adequate retrieval model is a key part of this task, as only the returned paragraphs in this phase will be analysed to check if any of them contains the answer for the proposed question. In general, retrieval models are built around three basic statistics from the data: frequency of terms in a document; frequency of a term in the collection, where document frequency (DF) or collection frequency (CF) can be applied; and document length.

The ideal ranking function for this task should be adaptable enough to fit the specific characteristics of the data in use. For the ResPubliQA 2009 task the documents are actually paragraphs with an average length of ten terms, and where the frequency of a question term within a paragraph will hardly exceed one. Given the task characteristics, a paragraph candidate to contain the answer to a question will be one where the maximum number of question terms appear (excluding stopwords); with a length similar to the average, avoiding to give too much relevance to term frequency within the paragraph.

The use of the classic Vector Space Model[5] model is not an adequate option for this task, since this model typically normalises the weight assigned to a document with the document length[2]. This would cause that those paragraphs that contain at least one question term and has the lowest length will obtain the highest score. Moreover, the typical saturation of terms frequency, with the logarithm or root square, used in this model gives too much relevance to the term frequency. This can be seen in equation (1) where frequency is saturated with the root square and normalisation is carried out dividing by the square root of the length.

$$R(q,d) = \sum_{t\ en\ q} \frac{\sqrt{freq_{t,d}} \cdot idf_t}{\sqrt{length(d)}} \tag{1}$$

A more adequate ranking function for this task is BM25[3]. In this ranking function the effect of term frequency and document length to the final score of a document can be specified by setting up two parameters (b,k_1). Further explanation of the effect of these parameters over the ResPubliQA data appears below.

The parameter b defines the length normalisation applied and it is computed as in the next equation:

$$B = (1 - b) + b(\frac{dl}{avdl})$$

where parameter $b \in [0, 1]$, dl is the document length, and $avdl$ is the average document length within the full collection. To assign 0 to b is equivalent to avoid the process of normalisation and

[2]As it is done in the Lucene framework.

therefore the document length will not affect the final score ($B = 1$). If b takes 1, we will be carrying out a full normalisation $B = \frac{dl}{avdl}$.

$$tf = \frac{freq_{t,d}}{B}$$

Once the normalisation factor has been calculated it is applied to the term frequency. Final score is computed applying a term frequency saturation that uses the parameter k_1 allowing us to control the effect of frequency in final score, as it can be seen in next equation:

$$R(q,d) = \frac{tf}{tf + k_1} \cdot idf_t \tag{2}$$

where $\infty > k_1 > 0$. Finally, IDF (Inverse Document Frequency) is tipically computed as next:

$$idf_t = \frac{N - df_t + 0.5}{df_t + 0.5}$$

Where N is the total number of documents in the collection and df_t is the number of documents in the collection that contains t. An implementation of the BM25 ranking function over Lucene was developed for this work[3]. The details of this implementation can be seen in [2]. The final expression for BM25 ranking function can be expresed as next:

$$R(q,d) = \sum_{t\ en\ q} \frac{freq_{t,d}}{k_1((1-b) + b \cdot \frac{l_d}{avl_d}) + freq_t} \cdot idf_t$$

3 Experimentation Settings

In order to test the precision of our retrieval system we propose the execution of two baselines. In both baselines the paragraph selected in order to answer the question is the one that appears first in the ranking obtained after the retrieval phase. The only difference between both baselines is if a stemming pre-process is carried out or not. In order to proceed with the stemming process for each language[4] the Snowball implementation that can be found at http://snowball.tartarus.org/ has been applied. The resources used for each language can be downloaded from the following sites:

1. **Bulgarian:**
   - Stoplist: http://members.unine.ch/jacques.savoy/clef/bulgarianST.txt
   - Stemmer: Not Available
2. **German**
   - Stoplist: http://members.unine.ch/jacques.savoy/clef/germanST.txt
   - Stemmer: http://snowball.tartarus.org/algorithms/german/stemmer.html
3. **English**
   - Stoplist: http://members.unine.ch/jacques.savoy/clef/englishST.txt
   - Stemmer: http://snowball.tartarus.org/algorithms/english/stemmer.html
4. **French**
   - Stoplist: http://members.unine.ch/jacques.savoy/clef/frenchST.txt
   - Stemmer: http://snowball.tartarus.org/algorithms/french/stemmer.html

[3] `http://nlp.uned.es/~jperezi/Lucene-BM25/`

[4] Except Bulgarian for which not available stemmer could be found.

5. **Italian**
    - Stoplist: http://members.unine.ch/jacques.savoy/clef/italianST.txt
    - Stemmer: http://snowball.tartarus.org/algorithms/italian/stemmer.html
6. **Portuguese**
    - Stoplist: http://members.unine.ch/jacques.savoy/clef/portugueseST2.txt
    - Stemmer: http://snowball.tartarus.org/algorithms/portuguese/stemmer.html
7. **Romanian**
    - Stoplist: http://members.unine.ch/jacques.savoy/clef/roumanianST.txt
    - Stemmer: http://snowball.tartarus.org/algorithms/romanian/stemmer.html
8. **Spanish**
    - Stoplist: http://members.unine.ch/jacques.savoy/clef/spanishSmart.txt
    - Stemmer: http://snowball.tartarus.org/algorithms/spanish/stemmer.html

The parameters values are equivalent for both baselines and have been fixed as next:

1. b: 0.6. Those paragraphs with a length over the average will obtain a slightly higher score.
2. k_1: 0.1. The effect of term frequency over final score will be minimised.

Both parameters have been fixed to these values after a training phase with the English development set supplied by the organisation. The obtained results are shown and described in detail in the next section:

4 Results

After the execution of both baselines the obtained results can be observed in Table 1, where the obtained results, the paragraphs average length and the best result obtained for each language. Where no data appears is due to no run was submitted for the specific language. The data for both baselines is the number of right paragraphs returned for the 500 questions. Between brackets an average of the P@1 measure per question appears. That is the number of questions answered correctly divided by the total of questions.

Table 1: Results obtained for both baselines.

	Avg. Length	**No Stemmed**	**Stemmed**	**Best**
Bulgarian	6.10	189 (.38)	189 (.38)	–
English	10.62	264 (.53)	263 (.53)	303 (.61)
French	12.41	219(.44)	225 (.45)	173 (.35)
German	10.37	185 (.37)	189 (.38)	202 (.40)
Italian	11.80	209 (.42)	212 (.42)	256 (.51)
Portuguese	12.69	237 (.47)	246 (.49)	–
Romanian	7.63	196 (.39)	220 (.44)	260 (.51)
Spanish	11.88	181 (.36)	199 (.40)	218 (.44)

Some preliminary conclusions can be extracted from the obtained results. First, it is clear that the best results have been obtained for the English language (.53). This can be easily explained by the fact that the parameters were fixed using the English development set. It is expected that a window for improvement could be found fixing the parameters specifically for each language.

As it can be observed in Table 1 a general behaviour is the fact that best results are achieved with the use of stemming.

Moreover, it appears that for those languages with more lexical variability (Spanish, French) the improvement with the use of stemming is clearly higher in relation with lexically more simple languages as English, where less variation appears.

The lowest performance can be observed for languages like Bulgarian or German, we believe that it is due to the fact of the higher complexity of these languages. The performance for these languages could be increased with the use of a lemmatiser instead of a stemmer.

Finally, no correlation has been found between the performance and the paragraph average length for each language.

Acknowledgements

This work has been partially supported by the Spanish Ministry of Science and Innovation within the project QEAVis-Catiex (TIN2007-67581-C02-01), the TrebleCLEF Coordination Action, within FP7 of the European Commission, Theme ICT-1-4-1 Digital Libraries and Technology Enhanced Learning (Contract 215231), the Regional Government of Madrid under the Research Network MAVIR (S-0505/TIC-0267), the Education Council of the Regional Government of Madrid and the European Social Fund.

References

[1] William R. Hersh, Aaron M. Cohen, Phoebe M. Roberts, and Hari Krishna Rekapalli. TREC 2006 Genomics Track Overview. In Ellen M. Voorhees and Lori P. Buckland, editors, *TREC*, volume Special Publication 500-272. National Institute of Standards and Technology (NIST), 2006.

[2] Joaquín Pérez-Iglesias, José R. Pérez-Agüera, Víctor Fresno, and Yuval Z. Feinstein. Integrating the Probabilistic Models BM25/BM25F into Lucene. *CoRR*, abs/0911.5046, 2009.

[3] Stephen E. Robertson and Steve Walker. Some Simple Effective Approximations to the 2-Poisson Model for Probabilistic Weighted Retrieval. In W. Bruce Croft and C. J. van Rijsbergen, editors, *SIGIR*, pages 232–241. ACM/Springer, 1994.

[4] Tetsuya Sakai, Noriko Kando, Chuan-Jie Lin, Teruko Mitamura, Hideki Shima, Donghong Ji, Kuang-Hua Chen, and Eric Nyberg. Overview of the NTCIR-7 ACLIA IR4QA Task. 2008.

[5] G. Salton, A. Wong, and C. S. Yang. A Vector Space Model for Automatic Indexing. *Commun. ACM*, 18(11):613–620, 1975.