The Influence of Semantics in IR using LSI and K-Means Clustering Techniques

D. Jiménez[1] , E. Ferretti[2] , V. Vidal[1] , P. Rosso[1] & C. F. Enguix[3]

[1]{djimenez,vvidal,prosso}@dsic.upv.es [2]ferretti@unsl.edu.ar [3]carlos@must-es.com

[1] Department of Computer Systems and Computation.
Polythecnic University of Valencia, Spain.

[2] LIDIC-Department of Computer Science
National University of San Luis, Argentina.

[3] Mediterranean University of Science and Technology, Spain.

ABSTRACT

In this paper we study the influence of semantics in the information retrieval preprocessing. We concretely compare the reached performance with stemming and semantic lemmatization as preprocessing. Three techniques are used in the study: the direct use of a weighted matrix, the SVD technique in the LSI model and the bisecting spherical k-means clustering technique. Although the results seem not to be very promising, we believe that they should be improved in the future.

1. BACKGROUND AND MOTIVATION

The Information Retrieval (IR) models used in this work are classified within the vector space model, included in the classic model. The actual models used are the generalized vector space model and the Latent Semantic Indexing (LSI) [1]. These models are based in the well-known matrix of terms by documents, which generally is a weighted matrix and rarely a frequency matrix [2].

The terms by documents matrix is constructed from a collection of documents. The process to obtain this matrix requires a preprocessing of that collection. There are various techniques in the preprocessing part, each one handle one or more aspects, i.e. reduce the number of terms that represent the collection, treatment of related words, etc.

After the preprocessing of the collection, a frequency matrix is constructed, which usually is transformed to a weighted matrix. There are many schemes to weight a frequency matrix, but a reasonable election is to use "term frequency" as the term frequency component and to use "inverse document frequency" as the collection frequency component [14], this scheme is used in this work.

With the weighted matrix we model the information retrieval system induced by the document collection. But we assume, too, two others models, a LSI model (using the SVD technique) and a clustering model (using the bisecting spherical k-means algorithm [9]).

The criteria used to evaluate the experiments, has been the average precision-recall ratio [1]:

$$\overline{P}(r) = \sum_{i=1}^{N_q} \frac{P_i(r)}{N_q}$$

where $\overline{P}(r)$ is the average precision at the recall level r, N_q is the number of queries used, and $P_i(r)$ is the precision at recall level r for the i-th query. To get each $P_i(r)$, first we evaluate the i-th query obtaining a sorted document set ordered descendently by relevance. Then we calculate the precision each time a relevant document appears in the answering set. In this data set we have interpolated 11 standard recall levels as follows [1]: Let $r_j \in \{0,...,10\}$, be a reference to the j-th standard recall level. Then, $P(r_j) = \max_{r_j \le r \le r_j+1} P(r)$.

An important problem in IR is to relate different words but with the "same" information in order to perform a conceptual or semantic search (i.e., based on the meaning of the words). Therefore, it is necessary to take into account synonyms and words which refer to the same concept. Stemming handles the problem of related words, joining words with the same root, but ignoring synonyms. This work investigates the use of semantic lemmatization, indexing documents and queries with word senses [8].

2.- PREPROCESSING

The preprocessing basically consists of a process to optimize the list of terms that identify the collection, which is previous to the query process. This optimization can be focused to reduce the number of terms eliminating those with poor information. For example, the use of stop words or heuristic methods. In the more basic case of preprocessing, a list of stop words is used to reduce the number of terms that identify the collection. The included terms in the stop words list do not provide information about; typically, these words are prepositions, articles, etcetera [1]. The heuristic methods are used to eliminate the terms that appear in fewer documents or in lot of them. The optimization can be also focused to join related terms. This association can be by the root of the words (stemming) or by synonymous words (semantic lemmatization). In the semantic lemmatization, the problem of polysemy has to be taken into account.

2.1.- Stemming

The words usually have different morphological variants with similar semantic interpretations and would be considered as the same term in information retrieval systems. For this purpose various stemming algorithms (or stemmers) have been developed. They attempt to reduce a word to its stem or root form. Moreover, this joins words with the same information to a single term, it also reduces the number of terms that identify the document collection. This issue is very important because reduces the storage space and computational time [12].

There are many stemmers but we have selected the Paice stemming algorithm [6], a heavy stemmer because it has a better behaviour with the document collection used in experimental study [9].

2.2.- Semantic Lemmatization

The vector space model was used for coding the documents. Each document was represented by a term vector. The classical vector space model for IR is shown to give better results if senses are chosen as the indexing space instead of word forms (i.e., terms): up to 29% improvement in the experimental results was obtained for a manually disambiguated test collection derived from the SemCor [8]. The representation of a document through a vector of synomyms of its terms, would allow for performing a semantic search.

Due to the phenomenon of polysemy, it is important to identify the exact meaning of each term. The disambiguation of the meaning of the term is obtained through its context (the portion of the text in which it is embedded), an ontology and a collection of sense-tagged samples, in case of using a supervised method. As external lexical resource we have used the WordNet ontology [10], which is based on the concept of synset (set of synonyms). In the WordNet, a polysemic term belongs to more than one synset. The ontology is partitioned into three hierarchies, each one associated to a syntactical category (nouns, verbs, adjectives+adverbs). Therefore, in order to perform the Word Sense Disambiguation (WSD) task, each term of a document needs first to be sintactically tagged (as noun, verb, adjective or adverb) according to its syntactical category. This Part-Of-

Speech (POS) task is performed by the TnT POS-tagger [18]. The POS-tagged vector of a document is used as input data for the supervised sense-tagger [11]. The final output is a sense-tagged vector, that is, tagged with the synsets of the terms of the document.

The semantic preprocessing was carried out for the documents of the collection, as well as for the queries. In the final vectors of the documents and the queries, those terms which were not sense-tagged were removed.

3.- TECHNIQUES FOR IR

3.1.- LSI Technique

There are several techniques in the LSI model. We have selected the SVD (Singular Value Decomposition) technique for its characteristics [2][4]. It is normally sufficient and even better to calculate a part of the spectrum of the singular values of the matrix. In this context it is defined a partial SVD of an arbitrary matrix *M*, the problem of finding *p* singular values and its corresponding right and left singular vectors. In other words, we must find *p* numbers $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_p \geq 0$ and *p* vectors $u_i \in \Re^m$ and $v_i \in \Re^n$ such that [5][7]:

$$M \approx M_p = U_p \Sigma_p V_p^T = \sum_{i=1}^{p} u_i s_i v_i^T \qquad (1)$$

The evaluation of queries within the SVD technique is based on the calculation of the angle between the query vector with all the document vectors of the collection. Taking into account the expression defined in (1) and that $\|q\|_2 = 1$ we obtain:

$$\cos q_j = \frac{m_j^T q}{\|m_j\|_2 \|q\|_2} = \frac{(M_p e_j)^T q}{\|M_p e_j\|_2} = \frac{(U_p \Sigma_p V_p^T e_j)^T q}{\|U_p \Sigma_p V_p^T e_j\|_2} = \frac{(e_j^T V_p \Sigma_p)^T (U_p^T q)}{\|\Sigma_p V_p^T e_j\|_2} = \frac{s_j^T (U_p^T q)}{\|s_j\|_2}$$

where m_j is the document vector, *q* is the column vector that represents the query, e_j is the j-th canonical vector of dimension *n* (number of documents) and $s_j = \Sigma_p V_p^T e_j$ [2].

3.2.- Clustering Technique

The clustering technique used in this work to evaluate semantic lemmatization is the Bisecting-Spherical K-Means [9]. This algorithm tries to join the advantages of the Bisecting K-Means algorithm [15][16] with the advantages of a modified version of the Spherical K-Means [3].

The Bisecting-Spherical K-Means clustering algorithm tries to find *k* disjoint clusters $\{p_j\}_{j=1}^k$, from the document collection expressed by matrix *M* such that it maximizes the following objective function:

$$f(\{p_j\}_{j=1}^k) = \sum_{j=1}^{k} \sum_{m \in p_j} m^t c_j$$

where c_j is the normalised centroid or concept vector of the cluster p_j, which it is calculated given the following expression:

$$t_j = \frac{1}{n_j} \sum_{m \in p_j} m \; ; \; c_j = \frac{t_j}{\|t_j\|} \qquad (3)$$

where n_j is the number of documents in the cluster p_j.

Algorithm 1. *Bisecting-Spherical K-Means algorithm.*

Step 1. Calculate *k* initial clusters with the Bisection K-Means algorithm described in [15][16]: $\{p^{(0)}{}_j\}_{j=1}^k$ and its concept vectors $\{c^{(0)}{}_j\}_{j=1}^k$
Initialize *t=0*.

Step 2. Calculate the new partition $\{p^{(t+1)}{}_j\}_{j=1}^k$ induced by the concept vector $\{c^{(t)}{}_j\}_{j=1}^k$:

$$p_j^{(t+1)} = \{m \in \{m_j\}_{i=1}^n : m^T c_j^{(t)} > m^T c_l^{(t)}, \quad 1 \le l \le n, l \ne j\}, \quad 1 \le j \le k$$

Step 3. Calculate the concept vectors associated to the new clusters $\{c^{(t+1)}{}_j\}_{j=1}^k$, using expression (3)

Step 4. When the stopping criteria is fulfilled, return $\{p^{(t+1)}{}_j\}_{j=1}^k$ and $\{c^{(t+1)}{}_j\}_{j=1}^k$. In other case increment $t=t+1$ and go to step 2

The stopping criteria (a relative error) used in algorithm 1 is the following:

$$\frac{\left| f(\{p^{(t)}{}_j\}_{j=1}^k) - f(\{p^{(t+1)}{}_j\}_{j=1}^k) \right|}{\left| f(\{p^{(t+1)}{}_j\}_{j=1}^k) \right|} \le e = 1 \times 10^{-2}$$

4.- EXPERIMENTAL RESULTS

4.1.- Case of Study

The collection used for the experiments contains articles from the 1963 Time Magazine and were compiled from ftp://ftp.cs.cornell.edu/pub/smart/time/ site. A total number of 425 documents have been parsed, with an average of 546 words and 53 lines per document. The contents referred to world news, especially politics frequently mentioning the following words: nato, african, Nasser, political, communist, regime, said, China, Europe, nuclear, Germany, Khrushchev, Gaulle, president, soviet, Moscow. Which in fact, reminded us of the typical news contents available in the cold war era. We have used the same stop words list included on the web site to maintain consistency.

Query statistics were also obtained for the query collection, formed by a total of 83 queries with an average of 15 words and one line per query. Some of the most frequent words used in the queries were: arab, federation, british, chinese, nuclear, nato, britain, Indonesia, soviet, Syria, minister, political, Kennedy, Germany, treaty, communist, Khrushchev, president.

4.2.- Comparisons

Only the most representative results of the study are presented: concretely, the weight, SVD and clustering comparisons between semantic lemmatization and stemming. All the comparisons are done with the average precision-recall criterion.

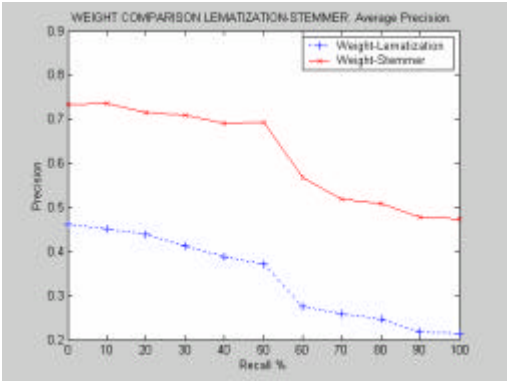

Figure 1. *Weight comparison. Semantic lemmatization vs. Stemming.*

When we compare the performance working with the weighted matrix (Figure 1), clearly the stemming preprocessing is better than semantic lemmatization preprocessing. The differences are not very great but significant.

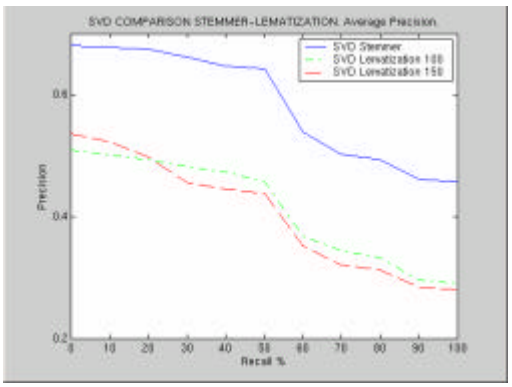

Figure 2. *SVD comparison. Semantic lemmatization vs. Stemming.*

Figure 2 compares the performance obtained with the SVD technique (within LSI model), concretely the optimal versions. In others words, in the SVD technique we can use a different number of singular values to do the approximation. Then, Figure 2 shows the reached performance with the number of singular values that improve the performance. In this figure it is also presented that stemming outperforms semantic lemmatization (the differences are approximately the same of Figure 1).

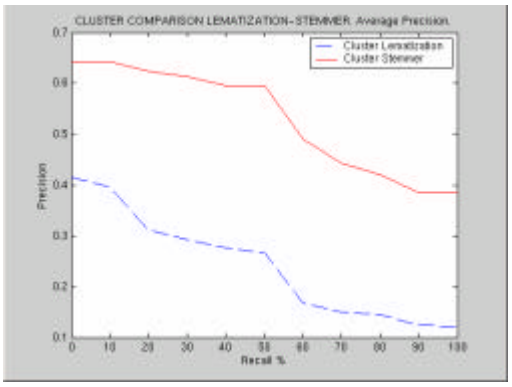

Figure 3. *Clustering comparison. Semantic lemmatization vs. Stemming.*

Finally, we present the comparison of the clustering case with the optimal number of clusters. Once again, the use of stemming preprocessing causes better performance than the semantic lemmatization, even if in this case the differences are more pronounced than in the previous cases.

In all the studied cases, the semantic lemmatization preprocessing has a worse performance than the stemming preprocessing, although their behaviours are similar. We can observe that the performance of the semantic lemmatization with the SVD is slightly better than the semantic lemmatization with the rest of the methods. It should be possible to improve the semantic lemmatization preprocessing and obtain better results. First, it is necessary to understand the reasons which cause such low results.

5.- CONCLUSIONS AND FURTHER WORK

The results were not so good as we hoped. We think that the poor performance is produced by mainly two reasons. First, indexing by synsets can be very helpful for text retrieval only if the error rate is below 30% [8]. Unfortunately, the state-of-the-art of WSD techniques performs with error rates from 30% to 60% which cannot guarantee better results than standard word indexing. Second, the semantic lemmatization relates synonyms when they are in the same morphologic group, but does not join related words as the stemming preprocessing does. This last aspect is very important for the text retrieval task.

Another possible reason could also due to the removal of the not sense-tagged terms (less terms are relevant for each document).

To improve the reached performance it should be interesting to study the use of morphologic lemmatization instead of stemming, and compare the results with those obtained with the semantic lemmatization. Moreover, it would be important to repeat the study with more document collections (e.g. [17]) in order to verify that the problem is not caused by the specific collection nature (politic text, many own names, little variety of subjects, etc.). As further work, we plan to simply applying the query expansion technique, expanding only the user's query with its synonyms, as well as to compare the results when an unsupervised WSD method is used [13].

ACKNOWLEDGMENTS

This work has been partially supported by the Ministerio de Ciencia y Tecnología, the FEDER DPI2001-2766-C02-02 project and also by the "TUSIR" CICYT project (TIC 2000-0664-C02). The work of E. Ferretti was made possible by AECI. We are grateful to A. Molina and F. Pla for making their sense-tagger available.

REFERENCES

[1] Baeza-Yates, R. and Ribeiro-Neto, B., "*Modern Information Retrieval*", Addison Wesley, 1999, ISBN: 0-201-39829-X

[2] Berry, M.W., Browne, M., "*Understanding Search Engines: Mathematical Modeling and Text Retrieval"*, 1999, SIAM, ISBN: 0898714370

[3] Dhillon, I. S., Fan, J. and Guan, Y., *"Efficient Clustering Of Very Large Document Collections*", 2001, Kluwer Academic Publishers ISBN 1-4020-0033-2

[4] Dumais, S., Furnas, G., and Landauer, T., "*Using latent semantic analysis to improve access to textual information. In Proceedings of Computer Human Interaction"*, 1988

[5] Forsythe, E., Malcolm, M. A., and Moler, C. B., *"Computer Methods for Mathematical Computations"*, Prentice-Hall, 1976.

[6] Fox, C. and Fox, B., "*Efficient Stemmer Generation Project*" www.cs.jmu.edu/common/projects/Stemming/

[7] Golub, G.H., and Van Loan, C.F., *"Matrix Computations*", The Johns Hopkins University Press, 1996, ISBN: 0-8018-5414-8

[8] Gonzalo J., Verdejo F., Chugur I. and Cigarrán J. *"Indexing with WordNet Synsets can improve Text Retrieval"* In Proceedings of the Workshop on Usage of WordNet for NLP, 1998.

[9] Jiménez, D. and Vidal, V. "*A Comparison of Experiments with the Bisecting-Spherical K-Means Clustering and SVD Algorithms*", Actas de las I Jornadas de Tratamiento y Recuperación de la Información, 2002, pp 45 – 52 ISBN: 8497051998

[10] Miller, A. "*WordNet: Lexical Database for English"* Communications of the ACM, 38 (11): 39-41, 1995.

[11] Molina, A., Pla, F., Segarra. E., *"A Hidden Markov Model Approach to Word Sense Disambiguation*" IBERAMIA2002. Sevilla, 12 al 15 de noviembre. 2002 © Springer-Verlag LNCS/LNAI.

[12] O'Neill, C. and Paice, C. D. "*What is Stemming?*" www.comp.lancs.ac.uk/computing/research/stemming/general/index.htm

[13] Rosso, P. et al. "*Automatic Noun Sense Disambiguation*". In Proc. of Int'l Conf. CICLing, Mexico City, Mexico, 2003 © Springer-Verlag LNCS(2588), pp. 273-275.

[14] Salton, G. and Buckley, C. "*Term weighting approaches in automatic text retrieval*" Information Processing and Management, vol. 24, no. 5, pp. 513--523, 1988.

[15] Savaresi,S.M. and Boley, D.L., *"On the performance of bisecting K-means and PDDP"*, First Siam International Conference on Data Mining, April 2001, Chicago, USA www.siam.org/meetings/sdm01/pdf/sdm01_05.pdf

[16] Steinbach, M., Karypis, G., Kumar, V., *"A Comparison of Document Clustering Techniques*", KDD-2000 Workshop on Text Mining, August 20-23, 2000, Boston, MA, USA

[17] "*Text Retrieval Conference (TREC) document collectiosn*" www.trec.nist.gov

[18] Thorsten Brants. "*TnT - A Statistical Part-Of-Speech Tagger*". www.coli.uni-sd.de/~thorsten/tnt 1998.