A Comparison of Experiments with the Bisecting-Spherical K-Means Clustering and SVD Algorithms

D. Jiménez (dajigon@doctor.upv.es), **V. Vidal** (vvidal@dsic.upv.es)
Departament of Computer Systems and Computation.
Polithecnic University of Valencia.
Spain

C. F. Enguix (carlos@must-es.com)
Mediterranean University of Science and Technology.
Spain

Abstract. In this paper we propose a modified version of the Spherical K-Means clustering algorithm, the Bisecting Spherical K-Means. The Bisecting clustering algorithm is used to determine the initial set in the Spherical K-Means clustering algorithm. We have prepare a set of experiments to compare the SVD with different number of singular values in order to find an optimal solution. Analogously, we have done with our modified version of the Spherical K-Means clustering algorithm, with different number of clusters. Finally we have compared both techniques with respect to precision-recall ratios.

1 Background and motivation

The information retrieval (IR) models used in our experiments are classified within the classic model, precisely on the vector space model. The concrete models used are the generalized vector space model and the latent semantic indexing (LSI). The basic vector space model is defined with a matrix of frequencies whereas in the generalized vector model the matrix of frequencies is substituted by a matrix of weights, both generically known as a matrix of terms by documents.

In figure 1 we briefly depict the preprocessing used, which is presented in section 2. In that we thoroughly explained the stemming algorithm and the reduction heuristics used for reducing the matrix space. Additionally it is explained the IR validation technique used along this paper for the whole set of experiments, known as average precision-recall ratio.

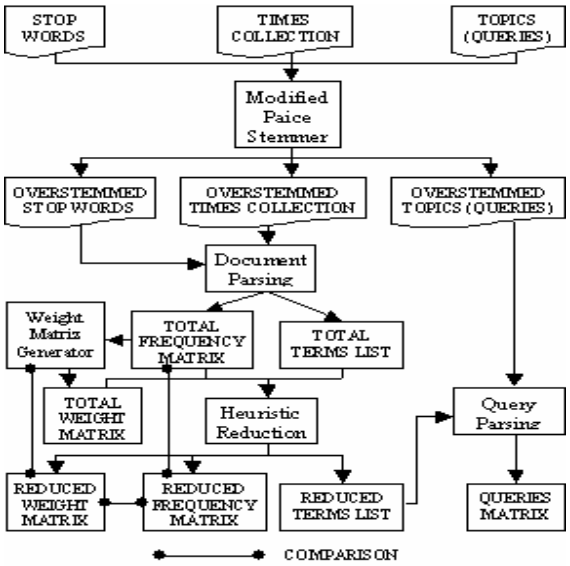

Figure 1. Steps followed in the preprocessing.

In section 3 we present an overview of the SVD technique applied to IR whereas in section 4 it is presented the modified version of the Spherical K-Means clustering algorithm. In section 5 we start by explaining the data set used and present a series of interesting statistics. Then, we compare the optimal reduced frequency and weight matrices (fig. 1).

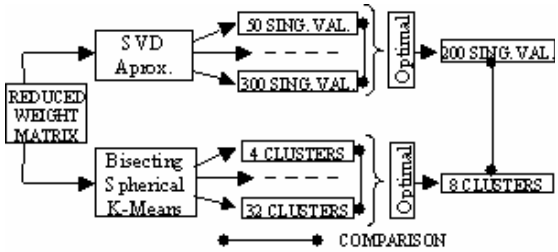

Figure 2. Comparison SVD and clustering algorithms.

Next, as we present in figure 2, we compare several singular values within the SVD. After choosing the optimal SVD approximation we compare it with the optimal weight matrix.

Analogously we chose the optimal number of clusters within the Bisecting Spherical K-Means algorithm. Finally we compare both optimal solutions.

2 Preprocessing

Several stemming algorithms were considered to pre-process our target collection. In order to evaluate which stemming algorithm to choose, we downloaded a set of stemming algorithms from the Efficient Stemmer Generation Project [7]. This site includes several stemming implementations and a corpus of words to be tested. We have slightly modified the Paice stemming algorithm to deal with hyphenated words. As stated in [10] the Porter algorithm is classified as a light stemming algorithm prone to under-stemming errors and the Paice and Lovins as heavy stemming algorithms prone to over-stemming errors. Light and heavy stemming algorithms cannot be compared in terms of accuracy as each type is suited for a different task. On the other hand in terms of performance accuracy we decided to include the Paice algorithm in our experiments as a better stemming algorithm as indicated in [10] and as table 1 demonstrate.

	Porter	Lovins	Paice
Nº Input terms	49528	49656	49656
Nº. output terms	41283	35029	33297
Reduction	16.65%	29.45%	32.94%

Table 1. Comparison of Stemming Algorithms

In the preprocessing we have also included a simple heuristics to eliminate terms that only appear in a small documents set. The following table presents the chosen options:

Nº. Docs.	Docs.	Number Terms	Size Reduction	Nº Non Zeros
0	0 %	12476	0 %	81082
1	0.24 %	6545	47.54 %	75151
2	0.47 %	4803	61.50 %	71667
3	0.71 %	3903	68.72 %	68967
4	0.94 %	3321	73.38 %	66639
5	1.18 %	2913	76.65 %	64599

Table 2. Heuristics used for pruning the term collection.

The criteria used to compare the reduction heuristics, has been the average precision-recall ratio as mentioned in [2]:

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

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 (relevance is equal to the cosine of the angle between the query and the document). 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 [2]: Let $r_j \in \{0,...,10\}$, be a reference to the j-th standard recall level. Then, $P(r_j) = \max_{r_j \le r \le r_j+1} P(r)$.

Now, we compare the reduced frequencies and weight matrices for the sizes indicated.

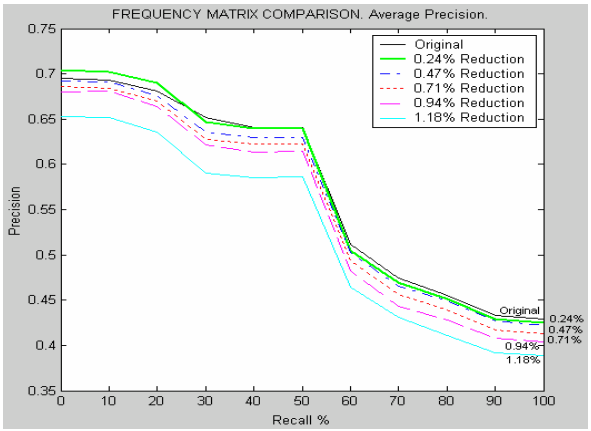

Figure 3. Frequency Matrices Comparison.

Generally speaking, when reducing the number of terms, precision drops but the dimension of the problem is reduced. The best case consists of when reducing the terms that appear in only one document as we reduce up-to a total of a 47.54% of terms meanwhile maintaining practically the same precision, even improving results in some cases.

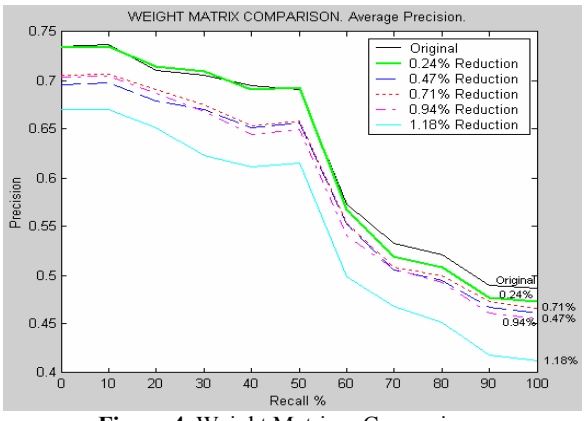

Figure 4. Weight Matrices Comparison.

As it occurred in the frequency matrix when reducing terms, in the weight matrix, the average precision drops too, although in this case it appears to be more evident. As it occurred in the previous case, it has been identified as the most optimal case, when reducing the terms that appeared only in one document or 0.24 % of the document collection. In spite of reducing almost 50% of the terms we still maintain almost the same precision.

3 IR with SVD

3.1 Singular Value Descomposition

From a matrix $M \in \Re^{m \times n}$ we can obtain its singular value decomposition (SVD) [8], [6] as:

$$M = U\Sigma V^T = \sum_{i=1}^{m} u_i \sigma_i v_i^T$$

where all the column vectors u_i of the matrix $U \in \Re^{m \times m}$ form a set of orthonormal vectors also known as left singular vectors of M. $\Sigma \in \Re^{m \times n}$ is a diagonal matrix containing the singular values of M fulfilling that $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_m \geq 0$. $V \in \Re^{n \times n}$ is an orthogonal matrix whose columns v_i are known as right singular vectors of M.

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 positive 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:

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

It can be found in the literature several methods to solve the SVD problem such as the complete SVD, partial or when the matrix is sparse [8]. When we deal with sparse matrices we use iterative methods such as Arnoldi [9], Lanczos, [11], [14] and subspace iteration [11], [14]. The LAPACK [1] and ARPACK [9] libraries efficiently implement SVD for both dense and sparse matrices.

3.2 Query evaluation with SVD

The SVD technique is classified within the LSI methodology [3], [5]. The evaluation of queries within the SVD technique is similar to the vector space model. It is based on the calculation of the angle between the query vector with all the document vectors of the collection. The main difference consists of the use of three (U, Σ, V) matrices instead of one, representing the document information system and being used to calculate the cosine of queries and documents. The general expression for calculating the cosine of the query and each document in the collection is defined by [3]:

$$\cos\theta_j = \frac{m_j^T q}{\|m_j\|_2 \|q\|_2} = \frac{\sum_{i=1}^{m} m_{ij} q_i}{\sqrt{\sum_{i=1}^{m} m_{ij}^2} \sqrt{\sum_{i=1}^{m} q_i^2}} \quad j = 1...n$$

where m_j is the document vector, q is the column vector that represents the query and n is the number of documents in the collection. Normally, both the terms by document matrix and the query vector are normalized such that $\|m_j\|_2 = \|q\|_2 = 1$, therefore we can simplify the calculation of the cosine to:

$$\cos\theta_j = m_j^T q = \sum_{i=1}^{m} m_{ij} q_i$$

When we use low rank approximation in the LSI such as the partial SVD, the calculation of the cosine or proximity of a query to a document varies. Taking into account the expression defined in (1) and that $\|q\|_2 = 1$ we obtain:

$$\cos\theta_j = \frac{m_j^T q}{\|m_j\|_2 \|q\|_2} = \frac{\left(M_p e_j\right)^T q}{\|M_p e_j\|_2} =$$

$$= \frac{\left(U_p \Sigma_p V_p^T e_j\right)^T q}{\|U_p \Sigma_p V_p^T e_j\|_2} = \frac{\left(e_j^T V_p \Sigma_p\right)^T \left(U_p^T q\right)}{\|\Sigma_p V_p^T e_j\|_2} = \frac{s_j^T \left(U_p^T q\right)}{\|s_j\|_2}$$

where e_j is the j-th canonical vector of dimension n (number of documents) and $s_j = \Sigma_p V_p^T e_j$ [3].

4 Clustering algorithms

The Spherical k-means clustering algorithm tries to find k disjoint clusters $\{\pi_j\}_{j=1}^{k}$, from the document collection expressed by matrix M such

that it maximizes the following objective function:

$$f\left(\{\pi_j\}_{j=1}^{k}\right)=\sum_{j=1}^{k}\sum_{m\in\pi_j} m^t c_j \qquad (2)$$

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

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

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

Finding an optimal solution in a clustering algorithm is a NP-complete problem. What follows next is the description of a modified version of the Spherical K-Means [4] algorithm, where we use the technique of iterative Bisection to determine the initial set of clusters.

Algorithm 1. Spherical K-Means algorithm.

Step 1. Calculate k initial clusters with the bisection technique described in algorithm 2, $\{\pi^{(0)}{}_j\}_{j=1}^{k}$ and its concept vectors $\{c^{(0)}{}_j\}_{j=1}^{k}$
Initialize *t=0*.

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

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

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, store $\{\pi^{(t+1)}{}_j\}_{j=1}^{k}$ and $\{c^{(t+1)}{}_j\}_{j=1}^{k}$. In other case increment *t=t+1* and go to step 2

The stopping criteria used in algorithm 1 is the following:

$$\frac{\left|f(\{\pi^{(t)}{}_j\}_{j=1}^{k})-f(\{\pi^{(t+1)}{}_j\}_{j=1}^{k})\right|}{\left|f(\{\pi^{(t+1)}{}_j\}_{j=1}^{k})\right|}\le\varepsilon=1\times10^{-2}$$

In step 1 of algorithm 1 we have used the iterative Bisecting clustering method to determine the k initial clusters. The following algorithm 2 describes the process:

Algorithm 2. Iterative Bisecting method for the determination of k-initial clusters

Step 1. Select a sparse vector c_l, where $c_l\in\Re^m$ with *nnz(M)/(n*m)* density:
- *nnz* is the number of non-zero elements.
- *m* is the number of terms.
- *n* is the number of documents.

Select a maximum number of iterations (*maxiter)* and the convergence criteria .
Initialize *iter*=0

Step 2. Divide M into two sub-clusters M_L and M_R according to:

$$m_i\in M_L \quad si \quad m_i^{\mathrm{T}}\mathrm{c}_1\ge\alpha$$
$$m_i\in M_R \quad si \quad m_i^{\mathrm{T}}\mathrm{c}_1<\alpha$$

Step 3. Calculate the concept vector of M_L, given c_l define as indicated in expression (3)

Step 4. Stop when the stopping criteria has been fulfilled or when *iter* is equal to *maxiter* and take M_L, $\mathrm{c_L}$,=c_l and M_R. In other case increment *iter=iter+1* and go to step 2.

The current version implemented in Matlab, executes a small number of iterations within algorithm 2, to determine the initial solution to step 1 in algorithm 1 having *maxiter*$\in$[2,3] and $\alpha\in$[0,1]. The parameter α determines the number and size of clusters generated.

5 Experiment Results

5.1 Case Study

The collection contains articles from the 1963 Time Magazine and were compiled from ftp://ftp.cs.cornell.edu/pub/smart/time/ site. It is stated on this site that the collection presents very high recall & precision when compared with more typical collections. 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.

5.2 Comparison between frequency & weight matrices

After having considered the preprocessing stages, we have selected the optimal cases for the reduction of the frequency and weight matrices. Now the question is to determine which of both techniques offer the best efficiency. If we consider storage requirements both of them have the same needs, due to both present the same structure and have the same number of non-zeros elements situated in the same position. With respect to computational costs the evaluation of the queries is the same, due to the cost of the cosine function is exactly equivalent. Therefore the differences appear when evaluating average query precision-recall ratios [2]. In the following figure it is presented a curve of average precision obtained from the optimal reduction frequency and weight matrices, *Mf* and *Mw* respectively.

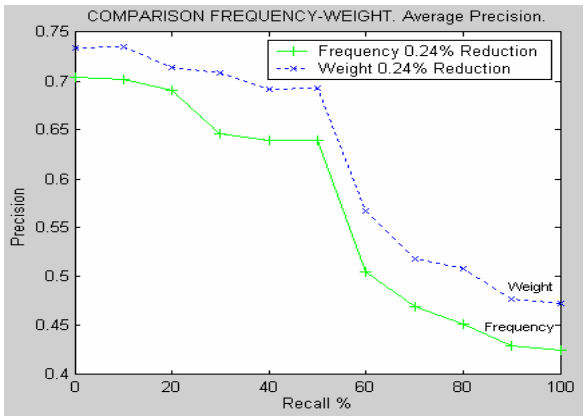

Figure 5. Frequency & Weight Matrix Comparison.

As it can be clearly seen the precision obtained in the weight matrix *Mw* is superior to the frequency one *Mf*.

5.3 SVD vs weight matrices

With the SVD we are using a low rank approximation model. We obtain a different approximation for the matrix depending on the number of singular values being calculated. Therefore, it is necessary to determine which is the optimal approximation according to precision-recall ratios. Next we present the curves of average precision obtained for different low rank approximations of *Mw*.

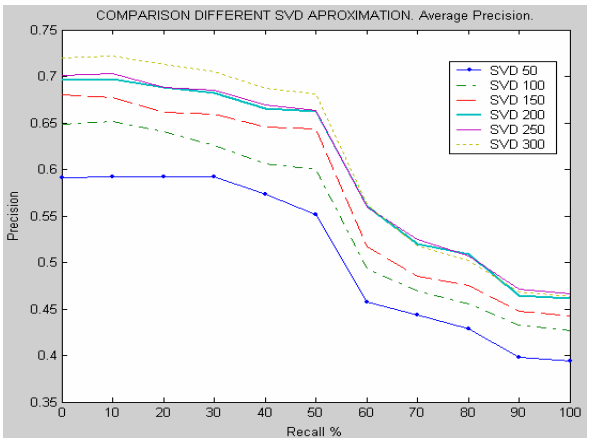

Figure 6. SVD Approximation Matrix Comparison.

It can be clearly seen as the approximation rank (the number of calculated singular values) grows, precision-recall ratios increase. It should be noted, it is more convenient to consider the set of low rank approximations compared to higher ones, due to both storage and computational requirements, which are lower. Taking into account these premises, we choose as the most optimal the approximation of 200 singular values, denoted here as *Msvd*.

In figure 7 we compare both optimal representatives *Msvd* and *Mw*.

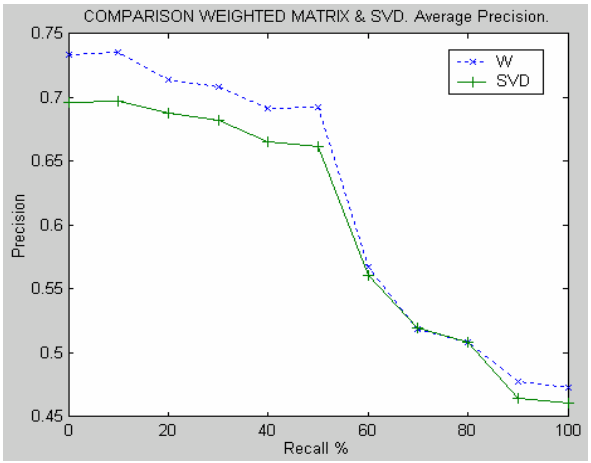

Figure 7. SVD & Weight Matrix Comparison.

Although it seams at first glance that *Mw* offers better performance with respect to precision-recall ratios, the fact that both curves are close to each other incites us to perform a more detailed study. Some average curves can be influenced by some abnormal queries. We have compared both systems on a query-by-query basis (fig. 8). In this case the comparison criteria used is the R-Precision [2] and the relationship among both has been represented with a histogram.

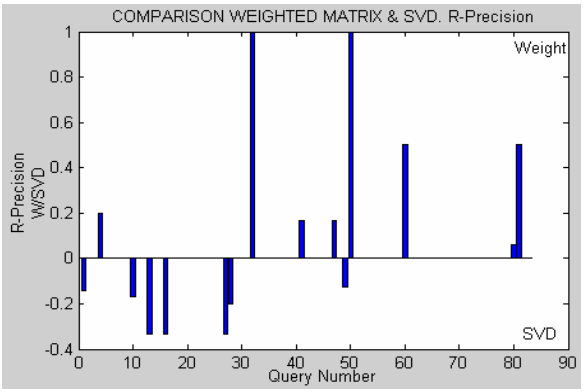

Figure 8. Precision Histogram SVD & Weight Matrix.

Both systems behave the same for a total of 81.5% of the queries and the differences among them are equally distributed (table 3):

IR System	Weight	SVD
Total Queries	8	7

Table 3. Total Queries Summary.

Query	**IR System**	**N° Rel. Docs**	**Query**	**IR System**	**N° Rel. Docs**
1	SVD	7	41	Weight	6
4	Weight	5	47	Weight	6
10	SVD	6	49	SVD	8
13	SVD	3	50	Weight	1
16	SVD	3	60	Weight	2
27	SVD	3	80	Weight	17
28	SVD	5	81	Weight	2
32	Weight	1			

Table 4. Conflictive Queries Set.

It can be observed that the differences are relatively small, except in a small subset of four queries (32, 50, 60 & 81), where we can identify very peculiar cases. Firstly, there are very few relevant documents (only one for queries 32 & 50 and two for queries 60 & 81). Secondly, R-Precision calculates the precision after having analyzed the list of documents in the answering set, therefore when analyzing so small document subsets precision could substantially vary.

Next we present different query comparisons of the precision curves obtained in both systems.

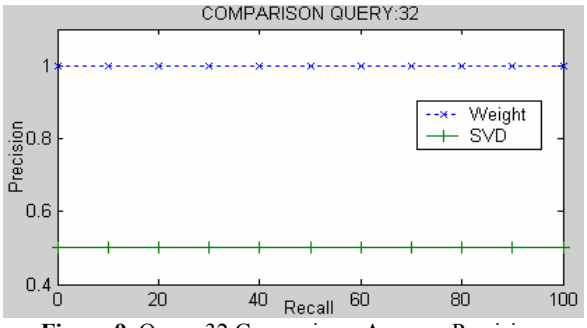

Figure 9. Query 32 Comparison. Average Precision.

In figure 9, it can be observed one of the extreme cases mentioned above. In the weight matrix the only relevant document is obtained in the first attempt where as in SVD is obtained in the second attempt.

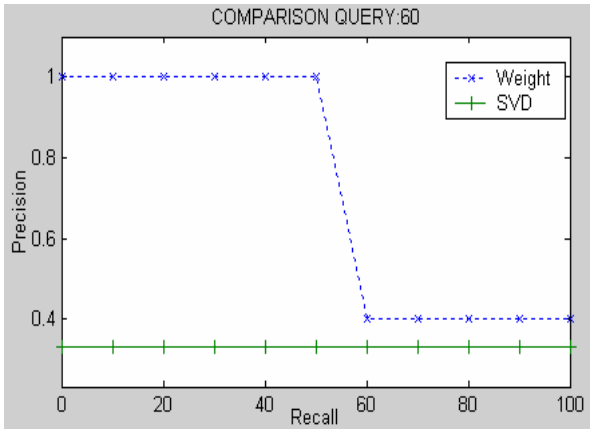

Figure 10. Query 60 Comparison. Average Precision.

An analogous situation is depicted in figure 10, where the set of relevant documents are only two.

From both previous cases we can deduce that when the number of relevant documents is low, *Mw* behaves better than *Msvd*, which in some way it is logical, considering that SVD it is based upon the semantic information among documents, returning more relevant documents but in these cases the solutions are so precise and delimited, the semantic information overflow is converted into noise.

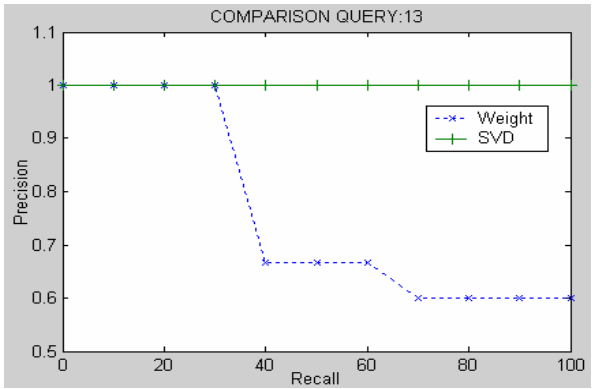

Figure 11. Query 13 Comparison.Average Precision.

In figure 11 we present the optimal case for the SVD. We can observe meanwhile SVD maintains constant total precision, the weight matrix looses precision when augmenting recall.

Finally we present an equilibrated example (fig. 12) where it is very difficult to deduce which is the best case or best curve.

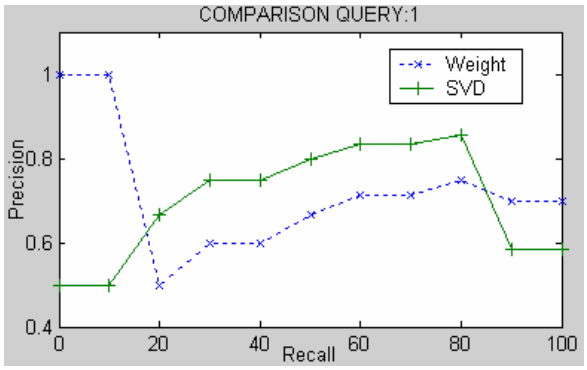

Figure 12. Query 1 Comparison. Average Precision.

An interesting fact is that the SVD usually plots smoother than the weight matrix. In both cases computation time is similar, as the optimization of the calculation of the cosine is of the same order, dependent on the number of non-zeros of the query. With respect to storage requirements, this is substantially affected by the type of collection. If we are dealing with a small and sparse collection, the sparse structure of the weight matrix represents the best method according to storage requirements. But, if we are dealing with a collection that generates a very large matrix, we could improve storage requirements with the SVD technique.

5.4 Bisecting Spherical K-Means clustering algorithm

The precision-recall ratio is used to obtain, for different number of clusters, the best partition of the collection. In this case, as we present in the next figure, the best partition is for 8 clusters, denoted here as *Mc*.

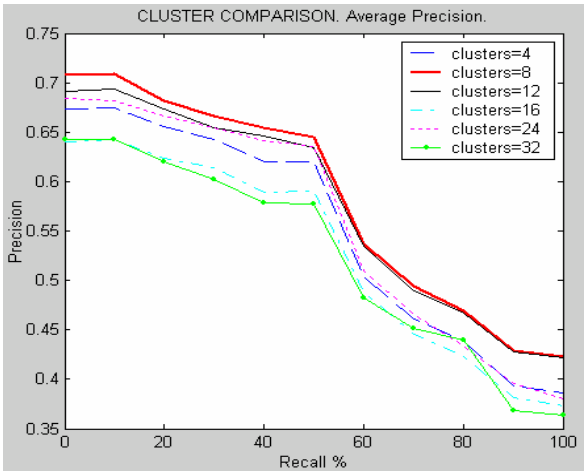

Figure 13. Cluster Comparison.

The experiments performed in the benchmarks used in this paper, indicates empirically, that having an $\alpha \in [0.2,0.4]$ gives the possibility to generate several clusters. When α is taken out of this range, very few clusters are generated. Within this range we have analyzed the behaviour of the optimization function (2) with respect to algorithm 1, for different number of clusters. We have obtained that for $\alpha \approx 0.3$, the optimization function increments with respect to other values. It should be mentioned that an optimal α has to be adapted for each particular collection.

Taking into account that the vector c_l in algorithm 2 is randomly chosen, each particular execution generates a new and probably different optimal distribution of clusters. Nevertheless, average precision-recall ratios are not generally influenced by differences in the optimal cluster distribution.

5.5 A comparison of the SVD and Bisecting-Spherical K-Means clustering algorithms

Again by using the same average precision-recall ratio, we will compare the results obtained in *Msvd* and *Mc*. To obtain the average precision-recall ratio curves, we have to ensure that we find all relevant documents. That is why we have extended the answering set of the clustering technique to several clusters until we guarantee a 100% of recall compared to a normal situation where only few clusters would have been evaluated. First we order the clusters according to the proximity of their centroids to the query and then we build an ordered list of the document answering set for the first cluster followed by the second one and so on until obtaining a 100% of the recall.

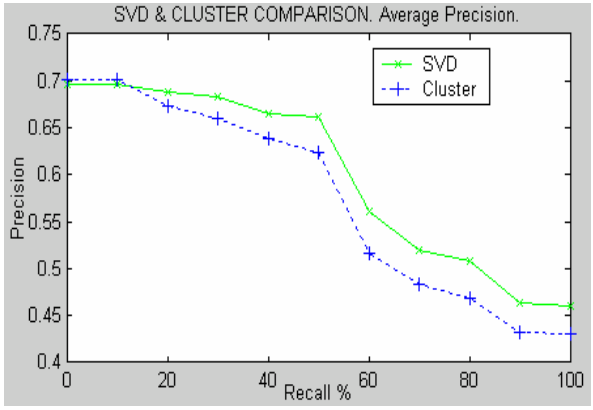

Figure 14. SVD & Cluster Comparison.

As it can be clearly seen in figure 14, the average precision of the clustering system is similar, although a bit inferior, to *Msvd*. What should be remarked is the lower computational cost and response time of the *Mc* based solution. As a matter of fact, the clustering system normally only evaluates the documents contained within a single cluster, not guaranteeing a 100% of recall. Nevertheless, we normally only revise the first set of documents returned by the answering set. Therefore we give more importance to high precision vs recall in the set of documents ranked at the top in the answering set.

6 Conclusions

In these experiments we have analyzed different IR techniques such as the generalized vector space model including document clustering and latent semantic indexing (SVD). Although in first instance it seams we obtain better results with reduced weight matrices, it is not feasible to use such technique for a real life case due to the fact it requires a huge storage space when we are dealing with large document collections. In conclusion, we recommend using methods based on SVD or clustering.

Clustering methods are very dependent on the initial selection of parameters to define the process such as defining the number of clusters to be obtained. On the other hand, the clustering partition obtained for different parameters does not sensibly affect the global behaviour of the system. The main advantage of clustering systems is the speed of the query response time, avoiding comparing entire document collections.

The SVD technique has the advantage of dealing with the semantic information of the collection, as also reducing the storage requirements for matrices of large dimensions.

Referencias

[1] Anderson, E., Bai, Z., Bischof, C., "*LAPACK Users' Guide*", SIAM, Second Edition, 1995

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

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

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

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

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

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

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

[9] Lehoucq, R., Maschhoff, K., Sorensen, D., Yang, C., "*ARPACK Homepage–Arnoldi Package*", The Dept. Comp. & Applied Maths., Rice University www.caam.rice.edu .

[10] Paice, C. D., "*An evaluation method for stemming algorithms",* Proceedings of the 17th ACM-SIGIR Conference on R&D in Information Retrieval, 1994, pp 42-50.

[11] Parlett, B., "*The Symmetric Eigenvalue Problem*",Prentice-Hall, 1980

[12] 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

[13] 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

[14] Vidal, V. Ginestar, D. Verdú, G, "*Análisis de imágenes mediante descomposición parcial en valores singulares*", Actas XVII CEDIA/XVII Congreso de Matemática Aplicada, 2001 ISBN: 84-699-6144-6