Neural Processing Letters (2005) 22:149–161 © Springer 2005
DOI 10.1007/s11063-005-2659-y

Merging Strategy for Cross-Lingual Information Retrieval Systems based on Learning Vector Quantization

M.T. MARTÍN-VALDIVIA*, F. MARTÍNEZ-SANTIAGO and L.A. UREÑA-LÓPEZ
Departamento de Informática, University of Jaén, Campus Las Lagunillas s/n, Edif A3, Jaén, E-23071 Spain. e-mail: maite@ujaen.es

Abstract. We present a new approach based on neural networks to solve the merging strategy problem for Cross-Lingual Information Retrieval (CLIR). In addition to language barrier issues in CLIR systems, how to merge a ranked list that contains documents in different languages from several text collections is also critical. We propose a merging strategy based on competitive learning to obtain a single ranking of documents merging the individual lists from the separate retrieved documents. The main contribution of the paper is to show the effectiveness of the Learning Vector Quantization (LVQ) algorithm in solving the merging problem. In order to investigate the effects of varying the number of codebook vectors, we have carried out several experiments with different values for this parameter. The results demonstrate that the LVQ algorithm is a good alternative merging strategy.

Key words. cross-lingual information retrieval, Kohonen neural network, learning vector quantization, merging strategy, retrieve status value

1. Introduction

Information Retrieval (IR) systems present to the user a set of items that will satisfy his or her information needs. We refer to the concrete expression of the information needs in words as a "query", and we call the items from which we select "documents". The goal of an IR system [4] is to find those documents that are relevant to a natural language query from within a collection of natural language documents. Many IR systems return not only the retrieved documents but also a numeric score (called Retrieved Status Value (RSV)) indicating the similarity of strength between the retrieved document and the query [15].

If the IR system must deal with several languages (document languages are not the same as the query user language) the IR system is called a Cross Language Information Retrieval (CLIR) system [6].

The CLIR system uses a query in one language to retrieve documents in different languages. In addition to language translation issues, how to conduct a ranked list that contains documents in different languages from several text collections is

*Corresponding author.

also critical. This issue is known as merging strategy problem or collection fusion problem, and it is not a trivial problem, since the weight assigned to each document (RSV) is calculated not only according to the relevance of the document and the IR model used, but also the rest of monolingual corpus to which the document belongs is a determining factor [3].

There are various attempts to solve the merging strategy problem, but even so a large decrease of precision is generated in the process (depending on the collection, between 20 and 40%) [19,17]. Recently, Towell et al. [18] trained a back-propagation network with one output unit per collection to solve the collection fusion problem. Although the obtained precision is comparable with other methods, the results could be improved. However, as Towell et al. point out, an alternative would be to train one network per collection and then merge the results.

In this study, we suggest using a competitive learning algorithm based on the Kohonen neural model [8]. We use the LVQ algorithm as a methodology for combining multiple lists of relevant documents obtained from several monolingual collections. First, we take into account the rank and the score of the first N relevant documents retrieved for each monolingual collection and we generate a neural net for each list by training and testing with the LVQ algorithm. Then we merge the M lists (where M is the number of different languages) in a single multilingual list reordered according to the new RSV. The results obtained are very promising.

In addition, we use the LVQ algorithm to train a neural net with a single multilingual list. This list is formed by $N \times M$ documents retrieved for each query. However, the results obtained are worse than when we use one network per monolingual collection.

Also, we present a comparative study varying the number of codebooks used for each neural network.

Finally, we compare our approaches with several fusion strategies. The results show that the LVQ algorithm performs noticeably better than traditional methods.

The rest of this paper as organized as follows. Section 2 briefly describes the CLIR systems. The most popular merging approaches are described in Section 3. Section 4 summarises the LVQ algorithm and describes our merging strategy. Results and experiment descriptions are shown in Section 5. Finally, we present the conclusion.

2. CLIR Systems

Since the second half of the 1990s, CLIR has become more prominent in the IR community and it is today a discipline which receives as great an interest as traditional IR [6]. A CLIR system is basically an IR system capable of operating over a cross-lingual document collection. That is, if a user consults a CLIR system, all relevant documents in the collection are retrieved, independently of the language used in the query and the documents. So the result of one of these systems will frequently be a heterogeneous list of documents written in English, Spanish, French,

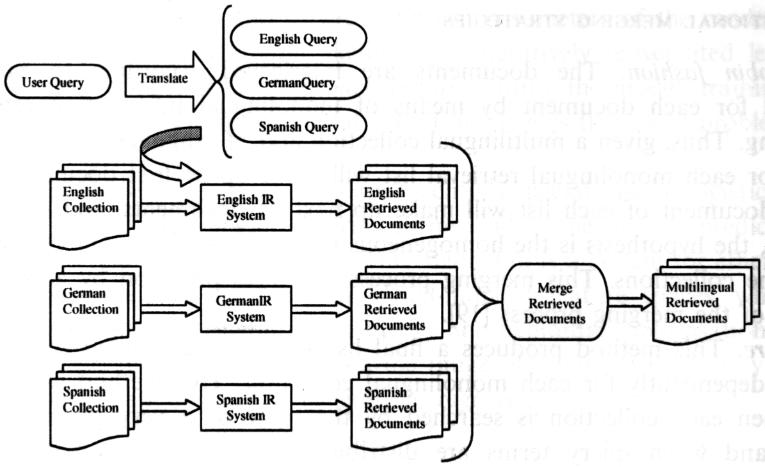

Figure 1. Query translation based CLIR.

German and ordered according to the score given to each document for a given query.

The effectiveness of a CLIR system is typically characterized by two statistics, precision and recall [11]. Precision is the fraction of the selected documents which are actually relevant to the user's information need (what proportion of documents found are relevant), while recall is the fraction of the actual set of relevant documents that are correctly classified as relevant by the CLIR system (how many potentially relevant target documents are found).

In CLIR systems, queries and documents are in different languages. We can translate queries, or translate documents, or translate both queries and documents into an intermediate language to unify the languages of queries and documents. Figure 1 shows the architecture when query translation is adopted. This architecture uses several monolingual document collections. Documents in different languages are indexed and retrieved separately. The ranked lists of all monolingual and cross-lingual runs are merged into one multilingual ranked list. How to merge result lists is a problem. Several works have proposed various approaches to deal with merging problem. The next section describes some traditional merging strategies used in CLIR systems.

3. Merging Strategies

Several merging strategies have been proposed in order to obtain a single list of relevant documents [16]. However, a large decrease of precision is generated in the process (depending on the collection, between 20 and 40%). The approaches used in this paper are briefly described below.

3.1. Traditional merging strategies

1. *Round-robin fashion.* The documents are interleaved according to ranking obtained for each document by means of monolingual information retrieval processing. Thus, given a multilingual collection and M languages, the first document for each monolingual retrieval list will make up M first documents, the second document of each list will make up next M documents, and so on. In this case, the hypothesis is the homogeneous distribution of relevant documents across the collections. This merging process decreases precision by about 40% because of the merging process [19].
2. *Raw score.* This method produces a final list sorted by document score computed independently for each monolingual collection. This method works well both when each collection is searched by the same or a very similar search engine, and when query terms are distributed homogeneously over all the monolingual collections. Heterogeneous term distribution may generate a wide variation of query weights among collections [3], and therefore this phenomenon may invalidate the Raw score merging hypothesis.
3. *Raw score normalised.* An attempt to make document scores comparable is by normalising in some way the document score reached by each document [13]
   - Given a monolingual collection, by dividing each RSV by the maximum RSV reached in such a collection

$$\mathrm{RSV}'_i = \frac{\mathrm{RSV}_i}{\max(\mathrm{RSV})}. \tag{1}$$

   - A variant of the previous method is to divide each RSV by the difference between the maximum and minimum document score values reached for each collection:

$$\mathrm{RSV}'_i = \frac{\mathrm{RSV}_i - \min(\mathrm{RSV})}{\max(\mathrm{RSV}) - \min(\mathrm{RSV})}. \tag{2}$$

3.2. Machine learning strategies

1. *Logistic regression.* This approach is a statistical methodology for predicting the probability of a binary outcome variable according to a set of independent explanatory variables. The probability of relevance to the corresponding document d_i will be estimate according to both the original score and logarithm of the ranking. Based on these estimated probabilities of relevance, the monolingual list of documents will be interleaved making up a single list [10, 17]

$$\mathrm{Prob}(d_i) = \frac{e^{\alpha+\beta_1\cdot\ln(rank_i)+\beta_2\cdot rsv_i}}{1+e^{\alpha+\beta_1\cdot\ln(rank_i)+\beta_2\cdot rsv_i}}. \tag{3}$$

The coefficients α, β_1 and β_2 are unknown parameters of the model that must be adjusted using maximum likelihood or iteratively re-weighted least squares methods. Because of this approach requires fitting the model, training set must be available for each monolingual collection. This is the major problem with the logistic regression method.

2. *Bayesian logistic regression*. This approach is a special case of logistic regression. It is a very efficient method during fitting and at the time of prediction [5].
3. *Support Vector Machine (SVM)*. This algorithm is a system for efficiently training linear learning machines in kernel-induced feature spaces, while respecting the insights of generalisation theory and exploiting optimisation theory [2]. SVMs have proved effective in a wide range of applications: binary and multi-class classification, pattern recognition, regression, etc.

4. A New Neural Approach to Merging Documents

We propose a new method based on the LVQ algorithm to solve the merging strategy problem. This approach allows us to include not only the rank and score of documents but also other information such as the publication date, term number of query, etc.

The LVQ algorithm is a competitive supervised learning based on the Kohonen model [8] which permits the definition of a group of categories in the space of input data by a reinforced learning.

The basic LVQ algorithm is quite simple. It starts with a set of input vectors x_i and weights vectors w_k (codebook vectors or prototype vectors) which represent the classes to learn. In each iteration, an input vector x_i is selected and the vectors w_k are updated, so that they fit x_i better. The LVQ algorithm works as follows.

For each class, k, a weight vector w_k is associated. In each repetition, the algorithm selects an input vector, x_i, and compares it with every weight vector, w_k, using the Euclidean distance $\|x_i - w_k\|$, so that the winner will be the codebook vectors w_c closest to x_i in the input space for this distance metric. The determination of c is achieved by following decision process:

$$\|x_i - w_c\| = \min_k \|x_i - w_k\|, \tag{4}$$

i.e.,

$$c = \arg\min_k \|x_i - w_k\|. \tag{5}$$

The classes compete between themselves in order to find the most similar to the input vector, so that the winner is the one with minor Euclidean distance with regard to the input vector. Only the winner class will modify its weights using a reinforced learning algorithm, either positive or negative, depending on the classification being correct or not. Thus, if the winner class and the input vector have the same class (the classification has been correct), it will increase the weights, coming

Table 1. Brief description of test-collection.

	English	German	French	Spanish	Italian
Number of documents	113,005	225,371	87,191	215,738	108,578
Size (in MB)	425 MB	527 MB	243 MB	509 MB	278 MB
Number of queries CLEF 2001	47	49	48	49	47
Number of relevant items CLEF 2001	856	2,238	1,193	2,694	1,246
Mean rel.request CLEF 2001	18.21	42.04	24.85	54.97	26.51
Number of queries CLEF 2002	42	50	50	50	49
Number of relevant items CLEF 2002	821	1,938	1,383	2,854	1,072
Mean rel.request CLEF 2002	19.55	38.76	27.66	57.08	21.88

slightly closer to the input vector. On the contrary, if the winner class is different from the input vector class (the classification has not been correct), it will decrease the weights, moving slightly further from the input vector.

Let $x_i(t)$ be an input vector at time t, and $w_k(t)$ represent the weight vector for the class k at time t. The following equations define the basic learning process for the LVQ algorithm:

$$\begin{aligned} w_c(t+1) &= w_c(t) + \alpha(t)[x_i(t) - w_c(t)] \quad && \text{if } d = c, \\ w_c(t+1) &= w_c(t) - \alpha(t)[x_i(t) - w_c(t)] \quad && \text{if } d \neq c, \\ w_k(t+1) &= w_k(t) \quad && \text{if } k \neq c, \end{aligned} \tag{6}$$

where c is the class of w_c, d is the class of x_i, and $\alpha(t)$ is the learning rate that decreases with the number of iterations of training ($0 < \alpha(t) < 1$). It is recommended that $\alpha(t)$ be rather small initially, say, smaller than 0.5, and that it decrease to a given threshold, ν, very close to 0 [8].

5. Experimental Results

5.1. Collection and pre-processing data

The experiments have been carried out for 5 languages: English, German, French, Spanish and Italian. We have used the CLEF[1] 2001 and 2002 collection data and relevance assessments. Table 1 presents a brief description of these collection data (extracted from Refs. 16 and 17).

[1]The Cross Language Evaluation Forum (CLEF) is an activity of annual character and of European ambit, since year 2000 and coordinated by DELOS Network of Excellence for Digital Libraries conferences in collaboration with the NIST and the TREC Conferences.

Table 2. Bilingual experiments.

Language	Average Precision CLEF 2001	Average Precision CLEF 2002
English	0.458	0.505
English → French	0.411	0.468
English → German	0.328	0.319
English → Spanish	0.453	0.387
English → Italian	0.315	0.282

Every collection has been pre-processed as usual in IR systems [4], using stop-word lists and stemming algorithms available via the Web.[2] Due to the German morphological wealth, compound words have been reduced to simple words with the MORPHIX package [12]. Once the collections have been pre-processed, they are indexed with the ZPrise IR system,[3] using the Okapi probabilistic model [14].

The next step consists of translating the query. The translation approach is very simple. We have used Babylon[4] to translate query terms word by word. Table 2 shows the bilingual precision obtained with the two collection data. We have taken into account only *Title* and *Description* query fields.

Once we had carried out the bilingual experiments, we obtained a single list of retrieved documents per language. These monolingual lists have to be merged in order to obtain a single multilingual list.

5.2. Experiments with the LVQ algorithm

The major problemwith the LVQ algorithmis that it requires data training. In this case, we have used CLEF-2001 queries and relevance assessments to train the neural networks. The CLEF-2002 collection is used to test the neural nets.

We use 2 classes: class 0 represents the non-relevant documents and the class 1 represents the relevant documents. We consider the same number of codebook vectors in each class.[5] Thus, when we use N codebook vectors, class 0 uses $N/2$ codebook vectors and the class 1 uses $N/2$ codebook vectors.

For each language we have one list with 2 data: the rank obtained for document and the RSV. Also, we know the relevance assessments. If the relevance assessment

[2] http://www.unine.ch/info/clef

[3] ZPrise is an information retrieval system developed by Darrin Dimmick (NIST). Available on demand at http://www.itl.nist.gov/iaui/894.02/works/papers/zp2/zp2.html

[4] Babylon is a Machine Dictionary Readable available at http://www.babylon.com

[5] Our first experiments were carried out both with the same number of codebook vectors per class and with different number of codebook vectors per class (specifically, we used a proportional number of codebook vectors for class because that is what Kohonen proposes). However,the obtained results showed that there was practically no difference between using or not using the same number of codebook vectors for class. So, we decided to use the same number of codebook vectors in our experiments because it is the default value in the LVQ_PAK package.

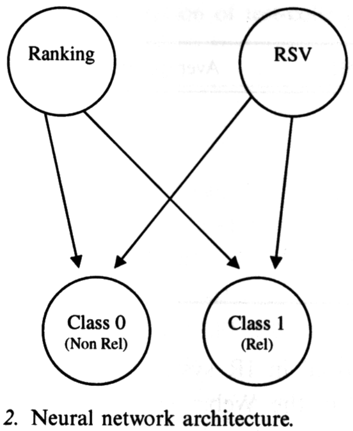

Figure 2. Neural network architecture.

```
1.   Randomly select an input vector $X$
2.   Calculate the Euclidean distance $D_1$ between $X$ and $W_1$
3.   Normalize the distance $D_1$ in range [0,1]
4.   The output network is $1 - D_1$
```

Figure 3. Evaluation LVQ algorithm with output in range [0,1].

is 0, the retrieved document is not relevant and if the relevance assessment is 1 the retrieved document is relevant.

In our approach, the input vectors x_i have 2 features (the document rank and the document RSV) and in addition the codebook vectors w_k are 2-dimensional vectors. Thus, during the training phase, we use the rank and the RSV as the x_i vectors and the relevance assessment as the desired output (the relevance assessment represents the output class for the input vector x_i). Figure 2 shows the network architecture.

Once the training phase has finished, the evaluation phase starts. Again, for each document, we use only the rank and the RSV as the neural net input, but the output network is the distance to the nearest codebook vectors belonging to the relevant class (class labelled with 1). This value represents the score assigned by the neural network to the document, i.e., the RSV of document obtained with the LVQ algorithm (RSV_{LVQ}). Figure 3 shows the algorithm to generate the new RSV.

In order to obtain this single list, we have used 2 different neural net architectures. First, we train 5 nets with the LVQ algorithm (one net per language) and then we merge the results obtained. Second, we have used a single neural network to carry out the complete process. In this case, the collection used is generated by the union of the 5 monolingual collections before training and testing the neural nets.

1. Monolingual-LVQ uses 5 neural nets: first, we obtain the 5 monolingual lists for the CLEF-2001 with ranking, RSV and relevance assessments. Then, we train 5

neural nets with the LVQ algorithm. The 5 monolingual lists for CLEF-2002 are used to test the nets trained. Thus, we obtain 5 new lists with different RSV (RSV_{LVQ-E}, RSV_{LVQ-F}, RSV_{LVQ-G}, RSV_{LVQ-I}, RSV_{LVQ-S}). Finally, we interleave the 5 lists taking into account the new RSV (i.e., the documents are sorted according to their new RSV obtained with the LVQ algorithm).

2. Multilingual-LVQ uses one neural net: first, we obtain the 5 monolingual lists for the CLEF-2001 with the ranking, RSV and relevance assessments. Then, we generate a single list by joining the 5 lists and use it to train the neural net with the LVQ algorithm. The 5 monolingual lists for CLEF-2002 are joined to create a single list and we use it to test the net trained. Thus, we obtained a new single list with different RSV (RSV_{LVQ-U}).

The experiments were carried out using the implementation described in LVQ_PAK documentation [9] with default parameters. Thus, every experiment started with the same number of codebooks per class and the learning rate was initialised to 0.3. The codebook vectors W_i were randomly initialised. In our experiments we have used a PC pentium 4, 3 GHz and 512 Mb RAM. The training phase takes about 3 s for each experiment.

In order to study the effects of varying the number of codebook vectors, we have compared the results obtained with different values. Table 3 shows the experiments with several numbers of codebooks for each language. The last step in order to obtain a single multilingual list from the 5 monolingual lists is to order the results, taking into account the monolingual RSV for each language (RSV_{LVQ-E}, RSV_{LVQ-F}, RSV_{LVQ-G}, RSV_{LVQ-I}, RSV_{LVQ-S}). The results obtained are also shown in Table 3.

Once the monolingual lists are merged, the best result is obtained with 10 codebook vectors (0.344). Therefore, this number of codebooks will be used to carry

Table 3. Average precision with several numbers of codebook vectors using one neural net per language.

CB	English	French	German	Italian	Spanish	Merging
2	0.390	0.380	0.320	0.334	0.417	0.318
4	0.441	0.371	0.335	0.329	0.445	0.334
6	0.275	0.387	0.341	0.331	0.481	0.313
8	0.298	0.391	0.345	0.336	0.485	0.321
10	0.410	0.399	0.342	0.334	0.480	0.344
12	0.433	0.427	0.335	0.356	0.416	0.343
14	0.382	0.423	0.337	0.359	0.442	0.339
16	0.467	0.423	0.325	0.341	0.391	0.339
18	0.425	0.423	0.251	0.349	0.474	0.334
20	0.446	0.429	0.260	0.362	0.388	0.327
30	0.441	0.425	0.234	0.362	0.402	0.323
50	0.379	0.423	0.216	0.362	0.476	0.321
100	0.393	0.409	0.282	0.363	0.449	0.329
200	0.375	0.416	0.287	0.345	0.427	0.320

Table 4. Average precision with several numbers of codebook vectors using a single neural net.

CB	Multilingual-LVQ	Monolingual-LVQ
2	0.322	0.318
4	0.300	0.334
6	0.295	0.314
8	0.289	0.322
10	0.295	0.344
12	0.292	0.343
14	0.296	0.339
16	0.294	0.340
18	0.292	0.335
20	0.274	0.328
30	0.297	0.323
50	0.319	0.322
100	0.320	0.330
200	0.320	0.321

Table 5. Cost factor argument values.

Language	Relevant Docs	Non-Relevant Docs	J
English	590	5110	8.67
Frech	820	4880	5.95
German	982	4718	4.80
Italian	727	4973	6.84
Spanish	1475	4225	2.86

out the experiments with the test data. The results obtained (0.309) is shown in next subsection to compare this value with the other merging methods.

In order to improve precision, we carried out another experiment using the best list for each language, i.e., RSV_{LVQ-E} with 16 codebook vectors, RSV_{LVQ-F} with 20 codebook vectors, RSV_{LVQ-G} with 8 codebook vectors, RSV_{LVQ-I} with 100 codebook vectors and RSV_{LVQ-S} with 8 codebook vectors. Then we merged these lists, sorting the RSV, and the average precision obtained was 0.318. This value is also shown in Table 5.

Table 4 presents the average precision obtained when we train a single neural net using the whole collection generated by joining the 5 individual collections per language (Multilingual-LVQ). In this case, the best result is obtained with 2 codebook vectors. Again, we will use this number of codebooks to carry out the experiments with the multilingual case in the evaluation phase. The results obtained with the test data (0.289) is shown in next Section 5.3.

As the results show, the LVQ algorithm performs much better when we use it to train 5 neural nets (one net per language) and then we merge the single lists. In all cases, the precision obtained with Monolingual-LVQ improves on the results obtained with Multilingual-LVQ. Moreover, the precision is below best value (0.322) only for 3 Monolingual-LVQ cases (2, 4 and 200 codebook vectors)

Table 6. Multilingual experiments.

Merging strategy	Avg Prec CLEF 2001	Avg Pre CLEF 2002
Round-Robin (baseline)	0.273 (0.0%)	0.251 (0.0%)
Raw score	0.291 (6.6%)	0.281 (12.0%)
Raw score normalized eq1	0.271 (−0.7%)	0.235 (−6.4%)
Raw score normalized eq2	0.297 (8.8%)	0.272 (8.4%)
Logistic regression	Training	0.286 (14.0%)
Bayesian logistic regression	Training	0.291 (15.9%)
SVM	Training	0.288 (14.7%)
Monolingual-LVQ	Training	0.309 (23.1%)
Monolingual-LVQ (best lists)	Training	0.318 (26.7%)
Multilingual-LVQ	Training	0.289 (15.1%)
Optimal performance	0.420 (53.8%)	0.367 (46.2%)

when we use a single network to carry out the whole merging process. Differences of precision between different numbers of codebook vectors are not very significant. When we compare with several languages, the differences are higher, but the same is true when we do not use the LVQ algorithm (Table 2).

5.3. Multilingual results

In order to prove the effectiveness our new approach based on LVQ, we have compared our methods (Monolingual-LVQ and Multilingual-LVQ) with several merging approaches: Round-robin, Raw score, Raw score normalized (equation 1 and equation 2), Logistic regression, Bayesian logistic regression and SVM. In our experiments, we have used the R package to implement the logistic regression.[6] The bayesian logistic regression has been implemented using the BBR package.[7] We have used default parameters except in the case of the Hyperparameter (parameter H=10). The reason for this is that empirically we have found the best results with this configuration. Finally, we have used the SVM_light implementation [7].[8] Since SVM is usually used in order to carry out categorization tasks, but we need a ranking value better than a category label, we have used a variant of SVM better suited for ranking tasks based on regression simulation. Thus, we have used SVM_light default parameters in regression mode except in the case of the cost factor argument J. For each language collections, we have applied a heuristic formula based on the quotient between negative and positive examples (Table 5)

$$J = \frac{\text{nonrelevantDocs}}{\text{relevantDocs}}. \tag{7}$$

The results are summarized in Table 6. In addition, theoretical optimal performance has been calculated by using the procedure proposed in [1] (labelled with

[6]This package is available at cran.r-project.org
[7]This package is available at http://www.stat.rutgers.edu/~madigan/BBR
[8]This package is available at http://svmlight.joachims.org/

"Optimal performance" in Table 6). This procedure computes the optimal performance that could possibly be achieved by a CLIR system.

As the results show, the results obtained with machine learning algorithms are better than traditional methods. The best performance is obtained with the Monolingual-LVQ. Our method with the best list per language improves on all the other approaches, reaching about 87% of theoretical optimal performance (traditional methods perform at about 70%). Even when we use the Multilingual-LVQ, the results are better than traditional approaches. However, it is necessary to comment that the advantage presented by traditional methods is that they do not require training, and so they are faster than the approaches based on learning. However, it is also necessary to emphasize that in all cases, this training is carried out off-line and once the learning phase has finished, the differences in speed between algorithms are not significant.

6. Conclusion

In this paper, we have presented a new approach based on the LVQ algorithm to solve a critical problem in CLIR systems: the merging strategy problem. A usual approach in CLIR is to translate the query to each language present in the corpus, and then run a monolingual query in each language. It is then necessary to obtain a single ranking of documents merging the individual lists from the separate retrieved documents. However, a problem is how to carry out such a merge. We have proposed a method based on LVQ that performs notably better than traditional merging strategies. In fact, even the worst case of Monolingual-LVQ approach surpasses the precision of Round-Robin method. We have presented a comparative study of the effects of varying the number of codebook vectors in LVQ algorithm, but as the results show, the differences are not very significant.

Acknowledgements

This work has been supported by Spanish Government (MCYT) with Grant TIC-2003-07158-C04-04.

References

1. Chen, A.: Cross-Language Retrieval Experiments at CLEF-2002, In: C. Peters (ed.), *Proceedings of the CLEF 2002 Cross-Language Text Retrieval System Evaluation Campaign. Lecture Notes in Computer Science*, pp. 5–20, 2003.
2. Cristianini, N. and Shawe-Taylor, J.: *An Introduction to Support Vector Machines.* CA: Cambridge University Press, 2000.
3. Dumais, S.: Latent Semantic Indexing (LSI) and TREC-2, In: NIST (ed.), *Proceedings of TREC'2*, Vol. 500. Gaithersburg, pp. 105–115, 1994.
4. Frakes, W. and R. Baeza-Yates (eds.): *Information Retrieval: Data, Structures and Algorithm.* NJ: Prentice Hall, 1992.

5. Genkin, A., Lewis, D. D. and Madigan, D.: Large-Scale Bayesian Logistic Regression for Text Categorization. Technical report, 2004.
6. Grefenstette, G.: *Cross-Language Information Retrieval.* Boston, USA: Kluwer academic publishers, 1998.
7. Joachims, T.: *Learning to Classify Text Using Support Vector Machines.* The Netherlands Kluwer, 2002.
8. Kohonen, T.: *Self-organization and Associative Memory.* Berlin: Springer Verlag, 2 edition, 1995.
9. Kohonen,T., Hynninen, J., Kangas, J., Laaksonen, J. and Torkkola, K.: LVQ-PAK: The Learning Vector Quantization Program Package, Technical Report FIN-02150, University of Technology, Laboratory of Computer and Information Science, Helsinki, Finland, 1996.
10. Le Calvé, A. and Savoy, J.: Database merging strategy based on logistic regression, *Information Processing and Management* **36** (2000), 341–359.
11. Manning, C. and Schtze, H. (eds.): *Foundations of Statistical Natural Language Processing.* MA: MIT Press, 2000.
12. Neumann, G.: Morphix Software Package, http://www.dfki.de/ñeumann/morphix/morphix.html, 2003.
13. Powell, A. L., French, J. C., Callan, J., Connell, M. and Viles, C. L.: The impact of database selection on distributed searching, In: T. A. Press (ed.), *Proceedings of the 23rd International Conference of the ACM-SIGIR'2000.* New York, pp. 232–239, 2000.
14. Robertson, S. E., Walker, S. and Beaulieu, M.: Experimentation as a Way of Life: Okapi at TREC, *Information Processing and Management* 1(36), 95–108, 2000.
15. Salton, G. and McGill, M. J. 1983, *Introduction to Modern Information Retrieval.* London,U.K.: McGraw-Hill, 1983.
16. Savoy, J.: Report on CLEF-2001 Experiments, In: C. Peters (ed.) *Proceedings of the CLEF 2001 Cross-Language Text Retrieval System Evaluation Campaign. Lecture Notes in Computer Science.* pp. 27–43, 2002.
17. Savoy, J.: Report on CLEF-2002 Experiments: Combining Multiple Sources of Evidence, In: C. Peters (ed.), *Proceedings of the CLEF 2002 Cross-Language Text Retrieval System Evaluation Campaign. Lecture Notes in Computer Science.* pp. 31–46, 2003.
18. Towell, G., Voorhees, E., Gupta, N. and Johnson-Laird, B. Learning Collection Fusion Strategies for Information Retrieval, In: *Proceedings Twelfth Anual Machine Learning Conference*, 1995.
19. Voorhees, E., Gupta, N. and Jhonson-Laird, B. The collection fusion problem, In: NIST (ed.), *Proceedings of the 3th Text Retrieval Conference TREC-3*, Vol. 500. Gaithersburg, pp. 95–104, 1995.