CONCEPTUAL RETRIEVAL BASED ON FEATURE CLUSTERING OF DOCUMENTS

Youjin Chang[+], Ikkyu Choi[+], Jongpill Choi[+], Minkoo Kim[+]
and Vijay V. Raghavan[#]

+(xaritas, ikchoi, cjp, minkoo)@ ajou.ac.kr
Dept. of Information & Computer Engineering, Ajou University
San5, Wonchun-dong, Paldal-gu, Suwon, 442-749, Korea

#raghavan@cacs.louisiana.edu
The Center for Advanced Computer Studies
University of Louisiana at Lafayette, Lafayette, LA 70504, USA

Abstract

In the Web search, since users' queries usually consist of only a few words, it is hard to identify their information needs. To solve this problem, many approaches have been tried to expand initial queries and to reweight the terms in the expanded queries using users' relevance judgments. Although relevance feedback is most effective when relevance information about retrieved documents is provided by users, it is not a fully automatic method. Another solution is to use correlated terms for query expansion. The main problem with this approach is how to construct the term-term correlations that can be used effectively to improve retrieval performance. In this study, we try to construct query concepts that denote users' information needs from a document space, rather than to reformulate initial queries using the term correlations and/or users' relevance feedback. To make query concepts, we extract features from each document, and cluster the features into primitive concepts that are used to form query concepts. Experiments are performed on a TREC collection.

Key words

Concept-based retrieval, query concepts, document features, feature clustering

1. INTRODUCTION

An information retrieval (IR) system returns a set of documents satisfying the information need expressed by user's question. The purpose of information retrieval is to retrieve all the relevant documents at the same time retrieving as few of the non-relevant as possible [14]. Nowadays, Web is staged in the center of the information technology. In search engines, the users might feel difficulty to formulate query. The problem is illustrated through the situation of information search on the Web, where the queries are usually of a few words long and a large number of hit documents are returned to the user [3]. Most people cannot design queries that are effective right away. Therefore, they spend large amount of time in reformulating their queries to accomplish effective retrieval.

Many researchers have tried to find appropriate solutions for representing users' interest correctly [3,12,19]. They have studied the query reformulation for improving the initial query through query expansion and term reweighting. The query reformulation involves two basic steps: expanding the initial query with new terms and reweighting the terms in the expanded query. These approaches are grouped in three categories [1]: (1) approaches based on feedback information from the user, (2) approaches based on information derived from the set of documents initially retrieved called pseudo relevance feedback (PRF), and (3) approaches based on global information from the document collection.

In our research, we propose a method that constructs query concepts that denote users' information needs from a document space. Our approach to generating a query concept is different in the sense that we do not employ either pseudo relevance feedback (PRF) or term-term correlations derived from the document collection. In this method, we try to extract features of documents and cluster them into primitive concepts that can be used as the basis concepts to form query concepts.

In Section 2, we describe related works. In Section 3, we outline how query concepts would be constructed. In Section 4, we show the experimental results that are obtained on a TREC collection.

2. RELATED WORKS

There are many works related with automatic query reformulation that improves initial queries through query expansion and term reweighting. The fully automatic methods for query reformulation do not rely on users to

make relevance judgments. They are often based on language analysis [2], term co-occurrences, PRF[1], or concept-based retrieval [12].

Bodner and Song [2] report that language analysis approaches require a deep understanding of queries and documents, usually at higher computational costs. Furthermore, deep understanding of queries is still an open problem in the field of artificial intelligence.

Without relevance feedback, there are other strategies to reformulate queries. The idea involves identifying terms that are related to the query terms. Those terms might be synonyms, stemming variations, or terms that are close to the query terms in the text. Two basic types of strategies are global analysis and local analysis. In automatic global analysis, the similarity thesaurus obtained is based on term-term relationships. Unfortunately, this approach does not work well in general because the relationships captured in a thesaurus frequently are not valid in the local context of a given user query [1]. Automatic local analysis adopts clustering techniques for query expansion. The local clustering techniques are based on the set of documents retrieved for the original query and use the top ranked documents for clustering neighbor terms. Such a clustering is based on term co-occurrence inside documents. However, local analysis is not suitable for Web documents because it is frequently necessary to access the text of documents. The idea of applying a global analysis technique to a local set of documents retrieved is called local context analysis. A recent work done by Xu and Croft in 1996 [19] illustrates the advantage of combining techniques form both local and global analysis.

Since user's feedback is difficult to obtain, there exist some variants such as pseudo relevance feedback (PRF). In PRF, a number of the documents at the top of the ranked list are assumed to be relevant and then relevance feedback methods are applied as if these documents were known to be relevant. This procedure has been found to be highly effective in some settings, most likely those in which the original query statement are long and precise [1]. This approach carried the risk that terms that are unrelated to relevance, but happen to meet the selection criteria, will be added to the query with subsequent adverse effects on retrieval behavior.

Another related area of work is concept-based retrieval. It treats those query words not as literal strings of letters, but as representing concepts, and then they can retrieve relevant documents even if they do not contain the specific words used in the query. Concept-based retrieval experiments often tested the effects of thesaurus-based query expansion on Boolean retrieval performance. In Klink's study [7], a method for improving the original query by an automatic reformulation method is proposed. Each phrase of the query corresponds to a concept where similar terms are stored. He used those terms to reformulate the original query and the user directly receives a hit list with more relevant documents without numerous searching cycles. Each term of

the query corresponds to a concept that is learned from the documents given by the feedback of the actual or of other users. However, this work can hardly be considered as a fully automatic reformulation method.

While query reformulation methods focus on reformulating initial queries by expanding and reweighting the terms in the queries, in our approach, we try to create query concepts that denote users' information needs, from a document space. To construct query concepts, we extract features from each document, and cluster them into primitive concepts that can be used to form query concepts. There are similar studies on extracting query concepts from a document space [10,12,18].

Nakata et al. [10] introduced a notion of Concept Index, which aims to index important concepts described in a collection of documents belonging to a group, and provide user-friendly cross-references among them to aid concept-oriented document space navigation. The Concept Index relied on users to identify important concepts by marking keywords and phrases that interest them. Nakata's work addressed a group of individuals who shared the same interest or a task and would profit from making use of the knowledge possessed by the group. However this approach is different from ours since they use collaboration between the members of group for extracting concept. In our approach, we try to automatically construct query concepts from a document space.

Wong and Fu [18] tried to construct concepts through the incremental document clustering technique for extracting features, which is more suitable to Web document classification. The main difference between their approach and ours is that their construction of concepts is for Web classification; not for constructing queries. However, we think that their approach can be applied to constructing query concepts.

The goal of our research is to make query concepts that are close to users' information needs from a document space. In the next section, we describe how to construct primitive concepts that can be used to form initial concepts from a document space.

3. EXTRACTING CONCEPTS IN A DOCUMENT SPACE

We suppose that there are primitive concepts (basis concepts) in a document space, and they can be used to form any concepts used in the field of information retrieval. In order to find out primitive concepts, we assume that documents contain features that characterize the primitive concepts. We have two steps to make primitive concepts: (1) to extract features from a document, and (2) to cluster the features into primitive concepts. The feature

extracting process has two sub-steps: (1) to select significant sentences, (2) to partition significant sentences into feature vectors. Figure 1 shows the main steps of constructing query concepts. In the next section, we describe each step in detail.

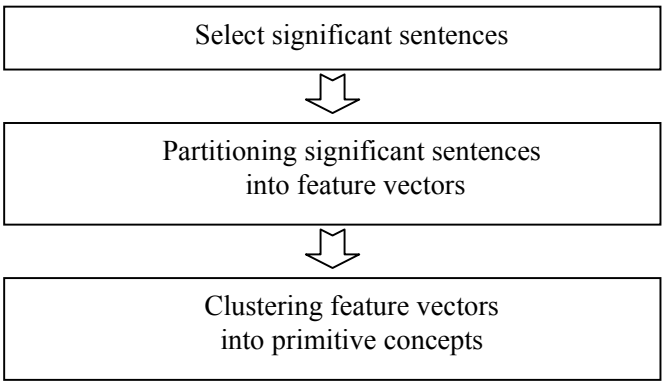

Figure -1. Procedure of constructing query concepts

3.1 Extracting Feature Vectors from a Document

In order to extract features (in the form of a vector) from a document, summarizing techniques are employed. The earlier research of Lam-Adesina and Jones [8] stated that summary generation methods seek to identify document contents that convey the most "important" information within the document, where importance may depend on the use to which the summary is to be put. Using the summary methods, we extract significant sentences of each document. For constructing feature vectors, we simply partition the significant sentences into feature vectors such that they do not contain the terms in other feature vectors for the same document. This means that the feature vectors are orthogonal.

For the extraction process of significant sentences of each document, we partially adopt Lam-Adesina and Jones's summary techniques [8]. In their study, the investigation explored the use of both context-independent standard summaries and query-biased summaries. They applied a very robust summarizer that can handle different text types likely to be encountered within a retrieval system. Sentence extracted summaries are formed by scoring the sentence in the document using some criteria, ranking the sentences, and then taking a number of the top ranking sentences as the summary. Various criteria of measuring sentence significance for effective summary generation are following: (1) sentence position within the document; (2) word frequency within the full-text; (3) the presence or absence of certain words or phrases in the sentence; (4) a sentence's relation

to other sentences, words or paragraphs within the source document. They computed each sentence score as the sum of its constituent words and the other scores.

In our research, however, we simply select significant sentences using the title term frequency method suggested by Lam-Adesina and Jones [8] and Luhn's keyword cluster method [9]. After selecting significant sentences for a document, we partition the sentence into "features", which characterize the document.

3.1.1 Selecting significant sentences

To select significant sentences from a document, we use Luhn's keyword cluster method [9] and title term frequency method.

3.1.1.1 Luhn's keyword cluster method

Luhn's keyword cluster technique, though simple, has been used alone or in combination with other methods in order to produce summaries. The technique of using frequency analysis of words is used to determine their significance in a document. Luhn used the most significant cluster in a sentence to measure the significance of the sentence. Luhn suggests in [9] that sentences in which the greatest number of frequently occurring distinct words are found in greatest physical proximity to each other, are likely to be important in describing the content of the document in which they occur.

In our approach, we slightly modify Luhn's keyword cluster method for the sake of simplicity. To determine significant words, we use TF/IDF weight values. If the weight values are higher than some threshold θ, we consider the words significant. In our experiment, the value of threshold θ = 0.3 is used. The significance score factor for a sentence is given by the following formula [8]

$$SS1 = \frac{SW^2}{TW} \qquad (1)$$

where **SS1** = the sentence score
SW = the number of significant words
TW = the total number of words

For example, suppose that the original sentence of full-text is; '**Orchid** is the flower of any of these plants, especially one **cultivated** for **ornament**.' After eliminating stopwords we get the following sentence; '**Orchid** flower plants especially **cultivated ornament**.'

Suppose that this sentence has three significant words (written in a bold face) according to the TF/IDF weight values that are higher than a fixed cutoff θ. The score of this sentence is $3^2/6 = 1.5$. After scoring, a minimum threshold for the number of significant words needs to be specified. This depends on the number of sentences in a document. Lam-Adesina and Jones [8] took 7 as a required minimum occurrence count for significant terms in a medium-sized TREC document; where a medium sized document is defined as on containing no more than 40 sentences and not less than 25 sentences. Since our experiment is also performed on TREC, we take the same measure of significance. For documents outside this range, the limit for significance is computed as

$$\mathbf{ms = 7 + [0.1(L - NS)]} \tag{2}$$

for documents with NS < 25 and

$$\mathbf{ms = 7 + [0.1(NS - L)]} \tag{3}$$

for documents with NS > 40
where **ms** = the measure of significance
L = Limit (25 for NS<25 and 40 for NS>40)
NS = number of sentences in the document

3.1.1.2 Title term frequency method

The title of an article often reveals the major subject of that document. This hypothesis was examined in TREC documents where the title of each article was found to convey the general idea of its contents. We exactly follow the Lam-Adesina and Jones's approach [8]. In order to utilize this attribute in scoring sentences, each constituent term in the title section is looked up in the body of the text. For each sentence a title score is computed as follows,

$$\mathbf{SS2} = \frac{\boldsymbol{TTS}}{\boldsymbol{TTT}} \tag{4}$$

where **SS2** = the title score for a sentence
TTS = the total number of title terms found in a sentence
TTT= the total number of terms in a title

3.1.1.3 Combining the score

The final score for each sentence is calculated by summing the individual score factors obtained for the above two methods [8]. Thus the final score for each sentence is

$$SSS = SS1 + SS2 \quad (5)$$

where SSS = Sentence Significance Score

Following with the Lam-Adesina and Jones's method [8], in order to generate an appropriate significant sentence list, it is essential to place a limit on the number of sentences to be used. It is important to take into consideration the length of the original document and the amount of information that is needed. They also mentioned that the optimal summary length is a compromise between maintaining terms that can be beneficial to the retrieval process. In our study, we select significant sentences by cutoff **ms** as described earlier in order to extract features for each document.

3.1.2 Partitioning significant sentences into feature vectors

As mentioned before, our goal is to construct query concepts that denote users' information needs. For this purpose, we try to find out primitive (basis) concepts that can be used to form the query concepts. We think of distinct features of documents as good candidates of primitive concept. In this research, we assume that a document can have several distinct features. This assumption is not strong, since it is possible that documents can contain several distinct topics. To find out the features from a document, we divide the significant sentences selected from the document into partitions such that they have distinct meanings.

Suppose that we represent a sentence as a vector of terms in the sentence with their TF/IDF weight values. We can consider a set of vectors $S = \{s_1, s_2, \ldots, s_{ms}\}$, where **ms** is the number of selected significant sentences for a document. To make feature vectors for the document, we partition S as follows. Let us consider each vector s_i as a subgraph such that the vertices of the subgraph are terms of the vector and they are connected. We then consider a feature vector as a connected component [4] of the graph consisting of subgraphs $s_1, s_2, \ldots, s_{ms}$. Since feature vectors are connected components of the graph, feature vector for the document is orthogonal to other feature vectors. For example, there are four significant sentences for document d and let us assume that a sentence is represented as a set of term and its TF/IDF weight value pairs.

s_1 = { (computer, 0.1), (information, 0.2), (software, 0.3) }
s_2 = { (hardware, 0.2), (computer, 0.1)}
s_3 = { (car, 0.2), (handle, 0.1), (signal, 0.3) }
s_4 = { (information, 0.2), (retrieval, 0.4) }

In the above example, for document d, we can construct two feature sets f_1 and f_2:

f_1 = { (computer, 0.1), (information, 0.2), (software, 0.3), (hardware, 0.2), (retrieval, 0.4) }
f_2 = { (car, 0.2), (handle, 0.1), (signal, 0.3) }

We can construct feature vectors vf_1 and vf_2 corresponding to f_1 and f_2, respectively, as follows

	car	computer	handle	h/w	information	retrieval	signal	s/w
vf_1=	0,	0.1,	0,	0.2,	0.2,	0.4,	0,	0.3
vf_2=	0.2,	0,	0.1,	0,	0,	0,	0.3,	0

We note that vf_1 and vf_2 are orthogonal.

3.2 Clustering Feature Vectors into Primitive Concepts

In the previous subsection, we construct feature vectors for each document. The feature vectors with a document are orthogonal to each other, but they might not be orthogonal to the feature vectors for other documents. Therefore, we cannot consider all the feature vectors as good candidates of primitive concepts. To alleviate the problem, we cluster the feature vectors such that the centroid vectors of the clusters are approximately orthogonal to each other. We call such centroid vectors “primitive” concepts.

Many effective clustering algorithms are available [6,13,16,18,20]. In order to cluster feature vectors into primitive concepts, we first selected two popular methods from already-developed methods; for instance, K-means and a X-means clustering methods [11]. However, they do not work well in our case. We conjecture that the reason why the previous clustering methods do not work well is that our feature vectors are extremely sparse. Therefore, we develop a new clustering method as described in Table 1. In our clustering method, if more than u% (e.g., 80%) terms in a feature vector f_i are contained in the centroid vector of a cluster C_j , we put f_i into C_j and recompute the centroid vector of C_j such that $C_j = (C_j + f_i) / 2$. If between

u% (e.g., 80%) and v% (e.g., 20%) of terms in f_i are contained in C_j, then we ignore f_i. If less than v% of terms in f_i are contained in C_j, we keep trying to put f_i into other clusters. If f_i, is not assigned to any clusters, we create a new cluster that contains f_i.

Table -1. Developed algorithm for Clustering feature vectors into primitive concepts

```
Let f1, ..., fn be feature vectors,
m the number of generated clusters,
and u and v are controlled parameters.

m  = 1;
Cm  = f1;
for i =2 to m
{
     done = FALSE;
     j = 1;
     while  (j <= N and not done) {
          if fi is more than u% overlapped with Cj then {
               Cj  = (Cj  + fi) / 2;
               done = TRUE;    /*assign fi to Cj*/
          }
          else if fi is more than v% overlapped with Cj then
               done = TRUE;  /* ignore fi */
          else
               j = j + 1;
     }
     if (not done) then {
          m = m + 1;
          Cm  = fi;     /* Create a new cluster */
     }
}
```

4. INFORMATION RETRIEVAL USING PRIMITIVE CONCEPTS

Finally, we construct query concepts by combining primitive concept vectors. The construction method is as follows.

(1) Select first top ten primitive concept vectors that are similar to the initial query q_0 .
(2) Generate all possible DNFs (Disjunctive Normal Forms) with at most three primitive concept vectors among the ten selected primitive concept vectors.
(3) Choose the DNF d that is most similar to the initial query.
(4) Construct the enhanced query $q = \alpha q_0 + \beta d$, where α and β are weighting constants.

We use MAX operation for OR operation in the DNFs that are generated in step (2) above. For example, let the selected primitive concepts be C_1, C_2, …, C_{10} and the initial query q_0. We can generate all possible DNFs as follows:

C_1, C_2, ..., C_{10}

$C_1 \vee C_2$, $C_1 \vee C_3$, ..., $C_9 \vee C_{10}$

$C_1 \vee C_2 \vee C_3$, $C_1 \vee C_2 \vee C_4$, …, $C_8 \vee C_9 \vee C_{10}$

From all possible DNFs, we select one that is most similar to q_0 as the query concept. Suppose that $C_1 \vee C_3$ is the DNF d that is the most similar to the initial query q_0. Then, we construct the enhanced query $q = \alpha q_0 + \beta d$.

5. EXPERIMENTS

For the experimental test, the DOE (Department of Energy) collection in TREC-1 is used. It contains about 220,000 documents. Ten topics (topic 65, 68, 82, 83, 96, 102, 111, 123, 134 and 135) are chosen to evaluate the performance. Since each document in DOE collection doesn't have title tag, we don't apply the title score for a sentence (SS2) to the sentence significant score (SSS) in this case.

Now, we describe our experimental results in a series of comparisons as follows. First, we report baseline retrieval results without feedback. Second, we conduct experiments using a PRF method. Finally, we give results for the query concept construction method proposed in this paper.

5.1 Baseline Runs & PRF Method

Table 2 shows retrieval precision at 10 documents cutoff, 30 documents cutoff, and the average precision for all retrieved documents. We test the baseline and PRF(pseudo relevance feedback).

For this test, we employ the Rocchio's relevance feedback method [15] for PRF. We use a Standard_Rocchio method to calculate the modified query as follows,

$$\vec{q}' = \alpha \vec{q} + \frac{\beta}{|D_r|} \sum_{\forall \vec{d}_j \in D_r} \vec{d}_j - \frac{\gamma}{|D_n|} \sum_{\forall \vec{d}_j \in D_n} \vec{d}_j \qquad (6)$$

Notice that D_r and D_n stand for the sets of relevant and non-relevant documents (among the retrieved ones) according to the user judgment, respectively. α, β, and γ are tuning constants.

We can see that the PRF method slightly improves the retrieval performance comparisons with the baseline.

Table -2. Baseline retrieval result and PRF retrieval result

Methods	10 docs	20 docs	average
Baseline	0.280	0.233	0.115
PRF	0.280	0.237	0.121

5.2 Query Concept Method

Table 3 shows the retrieval performance using our method denoted by QCM. In this test, we use the controlled parameter u = 0.8 and v =0.2 and two different sets of α and β, (α=0.5, β=0.5) and (α=0.4, β=0.6). Comparing with the baseline method, our method shows about 73% improvement on average precision.

Table -3. Result using the query concept method.

Methods	10 docs	30 docs	average
QCM(α=0.9, β=0.1)	0.430	0.373	0.177
QCM(α=0.8, β=0.2)	0.420	0.380	0.184
QCM(α=0.7, β=0.3)	0.410	0.367	0.192
QCM(α=0.6, β=0.4)	0.410	0.380	0.198
QCM(α=0.5, β=0.5)	0.410	0.373	0.198
QCM(α=0.4, β=0.6)	0.420	0.370	0.199
QCM(α=0.3, β=0.7)	0.400	0.357	0.194
QCM(α=0.0, β=1.0)	0.420	0.370	0.189

Figure 2 shows the performance curves of Baseline, PRF, QCM(α=0.5, β=0.5), QCM(α=0.4, β=0.6). It can be seen that the QCM methods perform consistently well and the average precision of QCM method is better than other methods. We attached three other tables that show more results of each topic in the appendix.

6. CONCLUSION

This paper has proposed a new paradigm for automatically enhancing initial queries. In the proposed approach, we have tried to construct query concepts that denote users' information needs. To construct the query concepts, we have extracted features from a document space, and clustered them into primitive concepts that are basis elements of the query concepts.

With the constructed concepts, we have showed promising experimental results. However, we can consider many other directions not only for extracting features for a document, but also for clustering features and constructing primitive concepts.

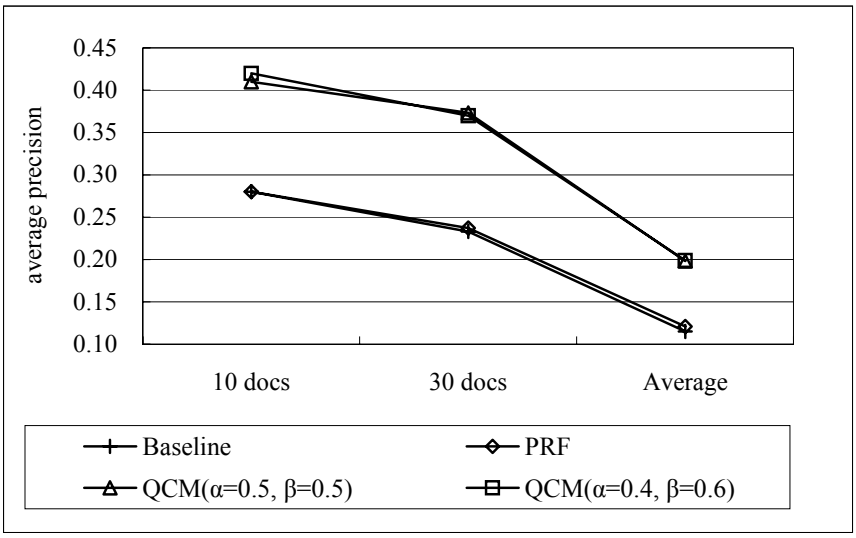

Figure -2. Performance of the baseline, PRF, and QCM methods

ACKNOWLEDGEMENTS

We thank to ByongHui Lee and Girim Park for programming and making experimental results of this paper.

APPENDIX

These tables show more results of experiments. Table 4 shows retrieval precision at 10 documents cutoff per topic. Table 5 shows retrieval precision at 30 documents cutoff per topic. Finally, table 6 shows the average precision for all retrieved documents per topic. The QCM column means that the retrieval using only concept vectors (weighting constant beta is 1.0).

Table -4. Average precision for all retrieved documents of Baseline, PRF, QCM methods.

TOPIC	Baseline	PRF	QCM	Rate of weighting constant beta						
				0.1	0.2	0.3	0.4	0.5	0.6	0.7
65	0.311	0.286	0.349	0.332	0.337	0.345	0.357	0.355	0.342	0.329
68	0.002	0.003	0.031	0.037	0.040	0.041	0.041	0.039	0.035	0.031
82	0.224	0.224	0.265	0.210	0.218	0.224	0.238	0.245	0.249	0.257
83	0.000	0.001	0.006	0.007	0.008	0.007	0.007	0.007	0.007	0.007
96	0.007	0.010	0.047	0.057	0.059	0.060	0.060	0.059	0.055	0.050
102	0.116	0.101	0.113	0.125	0.124	0.119	0.114	0.107	0.104	0.103
111	0.026	0.022	0.405	0.299	0.322	0.362	0.390	0.397	0.420	0.409
123	0.000	0.001	0.282	0.269	0.291	0.313	0.329	0.336	0.342	0.331
134	0.379	0.449	0.335	0.364	0.370	0.378	0.375	0.368	0.367	0.353
135	0.082	0.116	0.052	0.070	0.071	0.070	0.069	0.069	0.068	0.066
average	0.115	0.121	0.189	0.177	0.184	0.192	0.198	0.198	0.199	0.194

Table -5. Retrieval precision at 10 documents cutoff in Baseline, PRF, QCM methods.

TOPIC	Baseline	PRF	QCM	Rate of weighting constant beta						
				0.1	0.2	0.3	0.4	0.5	0.6	0.7
65	0.700	0.800	0.800	0.800	0.800	0.800	0.800	0.900	0.900	0.900
68	0.100	0.000	0.000	0.100	0.100	0.000	0.000	0.000	0.000	0.000
82	0.600	0.600	0.700	0.700	0.800	0.600	0.600	0.600	0.700	0.700
83	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
96	0.000	0.000	0.100	0.100	0.100	0.100	0.100	0.000	0.000	0.000
102	0.200	0.000	0.100	0.200	0.000	0.100	0.100	0.100	0.100	0.100
111	0.000	0.100	0.800	0.500	0.600	0.700	0.700	0.800	0.800	0.700
123	0.000	0.000	0.600	0.700	0.600	0.600	0.600	0.600	0.600	0.600
134	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.900	0.900	0.800
135	0.200	0.300	0.100	0.200	0.200	0.200	0.200	0.200	0.200	0.200
average	0.280	0.280	0.420	0.430	0.420	0.410	0.410	0.410	0.420	0.400

Table -6. Retrieval precision at 30 documents cutoff in Baseline, PRF, QCM methods.

TOPIC	Baseline	PRF	QCM	Rate of weighting constant beta						
				0.1	0.2	0.3	0.4	0.5	0.6	0.7
65	0.567	0.467	0.633	0.633	0.633	0.600	0.667	0.667	0.633	0.600
68	0.033	0.033	0.133	0.167	0.200	0.200	0.200	0.167	0.133	0.133
82	0.467	0.467	0.567	0.500	0.500	0.500	0.533	0.533	0.533	0.567
83	0.000	0.000	0.000	0.033	0.033	0.033	0.033	0.033	0.033	0.000
96	0.033	0.033	0.133	0.233	0.233	0.167	0.167	0.133	0.133	0.100
102	0.267	0.200	0.200	0.267	0.267	0.233	0.267	0.267	0.267	0.233
111	0.033	0.033	0.600	0.500	0.500	0.533	0.567	0.567	0.600	0.633
123	0.000	0.000	0.567	0.500	0.500	0.500	0.500	0.500	0.533	0.533

134	0.733	0.800	0.767	0.733	0.767	0.733	0.733	0.733	0.700	0.667
135	0.200	0.333	0.100	0.167	0.167	0.167	0.133	0.133	0.133	0.100
average	0.233	0.237	0.370	0.373	0.380	0.367	0.380	0.373	0.370	0.357

REFERENCES

1. Baeza-Yates, R. and Ribeiro-Neto, B. 1999. *Modern information retrieval*. Addison Wesley, 117-134.
2. Bodner, R. and Song, F. 1996. Knowledge-based approaches to query expansion in information retrieval. In McCalla, *Advances in Artificial Intelligence*, 146-158.
3. Chang, C. and Hsu, C. 1998. Integrating query expansion and conceptual relevance feedback for personalized web information retrieval. *7th International World Wide Web Conference*, http://www7.scu.edu.au/programme/posters/1887/com1887.htm.
4. Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein C. 2001. *Introduction to algorithm.* Second Edition. The MIT Press, 498-501.
5. Fellbaum, C. *WordNet*. 1998. An Electronic Lexical Database. The MIT Press
6. Gersho, A. and Gray, R. 1992. Vector quantization and signal compression. Kluwer Academic Publishers, Dordrecht, Netherlands.
7. Klink, S. 2001. Query reformulation with collaborative concept-based expansion. *In Proceedings of the First International Workshop on Web Document Analysis (WDA2001)*, Presentation I: Content Extraction and Web Mining, http://www.csc.liv.ac.uk/~wda2001/ .
8. Lam-Adesina, A. M. and Jones, F.J.G. 2001. Applying summarization techniques for term selection in relevance feedback. *In Proceedings of the 24th Annual International ACM SIGIR Conference*, ACM press, 1-9.
9. Luhn, H. P. 1958. The automatic creation of literature abstracts. *IBM journal of research & development*, 2 (2), April, 159-165.
10. Nakata, K., Voss, A., Juhnke, M. and Kreifelts, Th. 1998. Collaborative concept extraction from documents. *In Proceedings of the 2nd Int. Conf. on Practical Aspects of Knowledge management (PAKM 98)*, Basel, Switzerland, 29-30.
11. Pelleg, D. and Moore, A. 2000. X-means: extending K-means with efficient estimation of the number of clusters. *In Proceedings of the Seventieenth International Conference on Machine Learning (ICML2000)*, 727-734.
12. Qiu, Y. and Frei, H.P. 1993. Concept based query expansion. *In Proceedings of the 16th annual international ACM SIGIR conference on Research and Development in Information Retrieval*, ACM Press, 160-170.

13. Quaresma, P. and Rodrigues, I. P. 2000. Automatic classification and intelligent clustering for WWWeb information retrieval systems. *15th BILETA Conference*, England, http://elj.warwick.ac.uk/jilt/00-2/quaresma.html.
14. Rijsbergen, C. J. *Information retrieval*. Online text of a book. http://www.dcs.gla.ac.uk/~iain/keith/.
15. Rocchio, J. J. 1971. Relevance feedback in information retrieval in the SMART system. Prentice Hall, 313-323.
16. Willett, P. 1988. Recent trends in hierarchic document clustering: a critical review. *Information Processing & Management*, Vol. 24(5), 577-597.
17. Wishart, D. 1978. *Clustan 1C User Manual*. London, http://www.clustan.com/.
18. Wong, W. and Fu, A. 2000. Incremental document clustering for web page classification, *IEEE 2000 Int. Conf. on Info. Society in the 21st century: emerging technologies and new challenges (IS2000)*, Nov. , 5-8.
19. Xu, J. and Croft, W.B. 1996. Query expansion using local and global document analysis. *In Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, 4-11.
20. Zhang, T., Ramakrishnan, R., and Livny, M. 1996. BIRCH: An efficient data clustering method for very large databases, *In Proceedings of the ACM SIGMOD Conference on Management of Data*, 103-114.