Query Expansion by Pseudo Relevance Feedback

Zheyun Feng
fengzhey@msu.edu
Department of Computer Science and Engineering
Michigan State University

August 27, 2012

1 Introduction

In this document, we describe our algorithms for automatic query expansion. The proposed algorithms are implemented in Java where the Lucene library [1] is modified and used by the proposed algorithms for document retrieval. In order to support accurate document retrieval, we have implemented the okapi(BM25) formulation [2] for measuring the document-query similarity measure. Three query expansion methods are studied and implemented in the attached software, including query expansion based on pseudo relevance feed back [1], query expansion using documents returned by Google search engine, and query expansion using the synonym sets defined by WordNet [3].

The rest document is organized as follows: Section 2 describes the Okapi (BM25) formulation for document-query similarity measure, Section 3 describes different approaches for query expansion, and Section 4 presents the evaluation results for the developed approaches for query expansion.

2 Okapi (BM25) Formulation for Document-Query Similarity Measure

The Okapi (BM25) formulation for document-query similarity measure is considered to be the state-of-the-art for document retrieval. Given a query

[1]http://lucene.apache.org/core/
[2]http://nlp.uned.es/~jperezi/Lucene-BM25/
[3]http://wordnet.princeton.edu/

$\mathbf{q}$ that contains keywords $t_1, t_2, \cdots, t_n$, the Okapi BM25 score for a document $\mathbf{d}$ is given by [2]:

$$Score(\mathbf{d}, \mathbf{q}) = \sum_{i=1}^{n} idf(t_i) \cdot \frac{f(t_i, \mathbf{d}) \cdot (k+1)}{f(t_i, \mathbf{d}) + k \cdot (1 - b + b \cdot \frac{|\mathbf{d}|}{avg\_dl})} \tag{1}$$

Where $f(t_i, \mathbf{d})$ measures the occurrence of term t_i in document $\mathbf{d}$, $|\mathbf{d}|$ measures the number of words in document $\mathbf{d}$, and avg_dl is the average document length for the entire collection of documents. $idf(t_i)$ is the inverse document frequency weight for term t_i, and is computed as:

$$idf(t_i) = \log \frac{N - n(t_i) + 0.5}{n(t_i) + 0.5} \tag{2}$$

where N is the total number of documents in the collection, and $n(t_i)$ is the number of documents containing t_i. The parameters k and b are determined empirically, and was set to be 2 and 0.75, respectively, in our implementation as suggested in [2].

3 Query Expansion

Query expansion reformulates the original query issued by the user with the goal of improving the retrieval performance. In particule, query expansion could improve the original query by either re-weighting the terms in the original query, or adding extra terms to the original query to address the vocabulary mismatch problem. Below, we describe four approaches for automatic query expansion, i.e. query expansion based on pseudo relevance feedback, query expansion based on documents returned by Google search, query expansion based on the synonyms defined in WordNet, and query expansion based on the random walk model.

3.1 Query Expansion based on Pseudo Relevance Feedback

The most common approach for relevance feedback is the Rocchio method, where the key idea is to expand the original query with popular words appearing in the relevant documents and remove words from the original query if they are frequently used by the irrelevant documents. Let $\mathbf{q}$ be the term frequency vector for the original query, let $\mathcal{S}_+ = \{\mathbf{d}_1^+, \ldots, \mathbf{d}_m^+\}$ be the collection of documents that are deemed to be relevant to the given query $\mathbf{q}$, and $\mathcal{S}_- = \{\mathbf{d}_1^-, \ldots, \mathbf{d}_\ell^-\}$ be the collection of irrelevant documents for query

$\mathbf{q}$. The expanded query, based on the set of relevant documents from $\mathcal{S}_+$ and the set of irrelevant documents from $\mathcal{S}_-$, is given by

$$\widetilde{\mathbf{q}} = \mathbf{q} + \frac{\alpha}{m}\sum_{\mathbf{d}\in\mathcal{S}_+}\mathbf{d} - \frac{\beta}{\ell}\sum_{\mathbf{d}\in\mathcal{S}_-}\mathbf{d}$$

where parameters α and β are determined empirically.

Since the Rocchio method requires the knowledge of relevant and irrelevant documents for a given query, it can not be applied directly to the standard setup of document retrieval. Pseudo relevance feedback addresses this limitation by (i) assuming that the top returned documents are likely to be relevant (ii) ignore the factor of irrelevant documents in the Rocchio equation. In other words, $\mathcal{S}_+$ will include the first m returned documents, where m is determined empirically.

3.2 Query Expansion Using Google Returned Documents

This method is similar to query expansion based on pseudo relevance feedback described in the previous section. The key difference is that instead of using the top ranked documents returned from a given collection, the top m documents returned by Google are treated as relevant documents and used as the basis for query expansion.

3.3 Query Expansion Using Synonyms Defined by WordNet

WetNet define a large number of synonym sets (referred as synset), each corresponding to a different concept. We expand the original query by first finding the matched synsets that include the query words, and then including in the original query the words that appear in the matched synsets. The resulting expanded query vector is given by

$$\widetilde{\mathbf{q}} = \mathbf{q} + \alpha\mathbf{q}_s \tag{3}$$

where $\mathbf{q}_s$ includes the synonyms found in the matched synsets, and parameter $\alpha \in (0, 1)$ is introduced to down weight the significance of expanded query words when computing the similarity between queries and documents.

3.4 Query Expansion based on Random Walk Model

The key idea of random walk model for document retrieval is to view the process of identifying relevant documents as a random walk over a weighted graph: both documents and the query are mapped to the vertices in the graph; each document and the query is connected by an edge that is weighted by the similarity between the document and the query; similarly, any two documents are connected by an edge weighted by the similarity between the two documents. Consider a random surfer, starting from the query, performs random walk over the connected graph, with the probability of jumping from one vertex from another set to be proportional to the similarity between the two vertices. We measure the final similarity between a document and the query as the chance for the random surfer to land on the vertex corresponding to the document. It can be shown that, if the random surfer is only allowed to walk $m + 1$ steps over the connected graph, the chance of landing on individual documents, denoted by $\mathbf{p} = (p_1, \ldots, p_N)$, where N is the total number of documents in the collection, is given by

$$\mathbf{p} \propto (AA^\top)^m A\mathbf{q}$$

where $A = (\mathbf{d}_1, \ldots, \mathbf{d}_N)^\top$ is the document-term matrix, with each row corresponding to the term frequency vector of a document. Given the probabilities $\mathbf{p}$, the expanded query vector is given by

$$\widehat{\mathbf{q}} = \sum_{i=1}^{N} p_i \mathbf{d}_i \propto A^\top \mathbf{p} = A^\top (AA^\top)^m A\mathbf{q}$$

We note that the random walk model is closely related to pseudo relevance feedback and Latent Sematic Index (LSI). In fact, if we set $m = 0$, the probabilities $p \propto A\mathbf{q}$ are proportional to the document-query similarity, and therefore the expanded query will be similar to the one obtained by pseudo relevance feedback. On the other hand, if we set m to be a large number, probabilities p will be determined mostly by the top eigenvectors of A, and the resulting expanded query will be similar to the one obtained by LSI. In our implementation, we found that $m = 1$ yields the best performance compared the other choices of m.

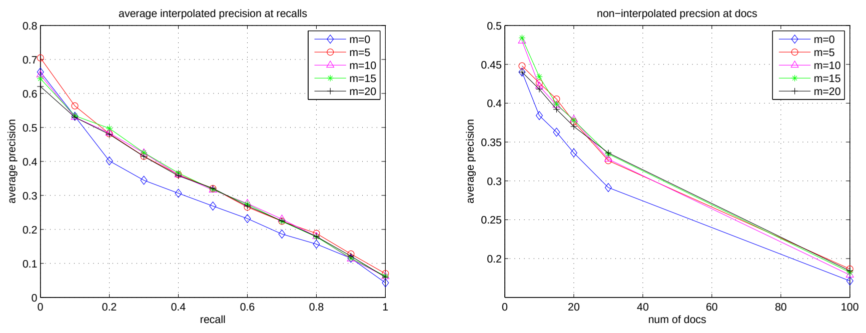
4 Evaluation

4.1 Evaluation by Document Retrieval

In the first experiment, we evaluate the performance of thee three methods for query expansion (i.e. query expansion based on pseudo relevance feedback, Google returned documents, and WordNet) by a retrieval experiment.

The data set used for evaluation is consisted of 79,923 documents and relevance judgments for 50 queries. More information about this collection can be found [4].

Pseudo relevance feedback Figure 1 show the precision/recall curve and the average precision for the first 100 returned documents using different numbers of top ranked documents for query expansion. Parameter α set to be 0.3 in this experiment. We observed that (i) pseudo relevance feedback significantly improves the retrieval performance compared to the retrieval method without using pseudo relevance feedback ($m = 0$). and (ii) pseudo relevance feedback achieves good retrieval performance when setting $m = 15$. Figure 2 shows the precision/recall curve and the average precision for the first 100 returned documents using different values for parameter α. m is set to 15 in this experiment. We observed that pseudo relevance feedback achieves good retrieval performance when $\alpha = 0.1$.



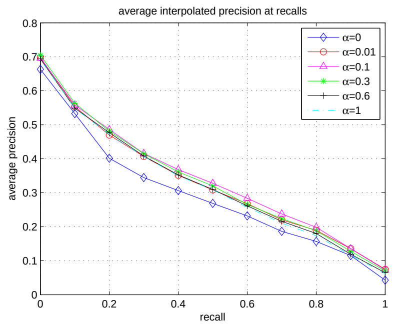
(a) Precision/recall Curve (b) Precision for top ranked documents

Figure 1: Precision-recall curve (left) and precision for top ranked documents (right) using different number (m) of top returned documents for query expansion. The simplified Rocchio algorithm is used for query expansion. α is set to 0.3. Lucene default scoring function is used to measure the document-query similarity.

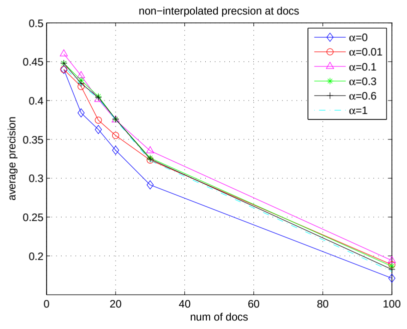
Query expansion using Google returned documents To extract documents returned by Google, we use the Google Web search API [5]. Since the result returned is in Javascript format, to make it compatible with our

[4] `http://www.cse.msu.edu/~cse484/hw/hw4.zip`

[5] `https://developers.google.com/web-search/docs/`#fonje



(a) Precision/recall Curve



(b) Precision for top ranked documents

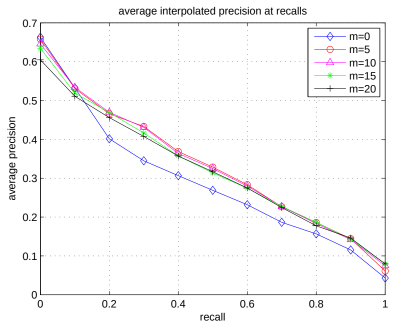
Figure 2: Precision-recall curve (left) and precision for top ranked documents (right) using different values for α in the simplified Rocchio algorithm. m is set to 15. Okapi BM25 scoring function is used to measure the document-query similarity.

java version code, we use the *Gson* library [6] to convert the javascript objects to JSON representation, which can be used directly in the java code. One shortcoming with using the Google returned documents is the limited number of queries can be submitted through the Google Web search API where the related discussion can be found `https://developers.google.com/web-search/terms` and `https://developers.google.com/errors/`.

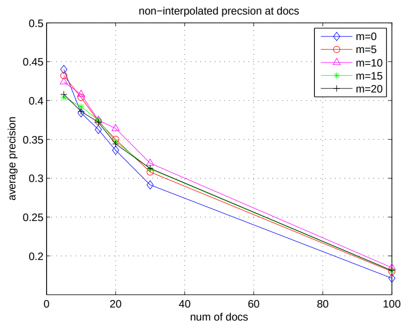
Figure 3 and 4 show the retrieval performance (measured in precision/recall curve and average precision for top returned documents) for varying m, the number of top ranked documents, and parameter α used by the simplified Rocchio formulation, respectively. We observed similar results as pseudo relevance feedback, namely (i) overall the query expansion based on Google returned documents improves the retrieval performance, and (ii) the query expansion achieves the best performance when $m = 10$ and $\alpha = 0.6$.

Query expansion using synonym sets defined in WordNet Figure 5 show the retrieval performance (measured in precision/recall curve and average precision for top returned documents) for varying the parameter α used by the simplified Rocchio formulation. We observed that no matter which value is used for α, the query expansion based on the synonyms defined in WordNet is unable to improve the retrieval performance compared to the method that directly uses the original query. This is consistent with

[6]`http://code.google.com/p/google-gson/`.

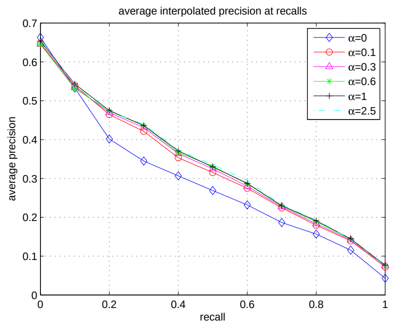


(a) Precision/recall Curve

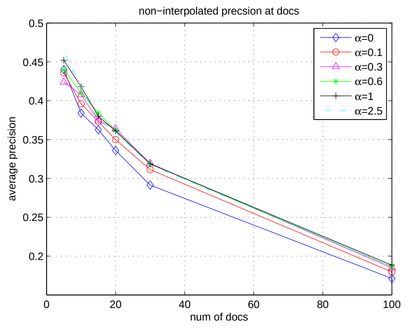


(b) Precision for top ranked documents

Figure 3: Precision-recall curve (left) and precision for top ranked documents (right) using different number (m) of top Google returned documents for query expansion. The simplified Rocchio algorithm is used for query expansion. α is set to 0.3. Lucene default scoring function is used to measure the document-query similarity.



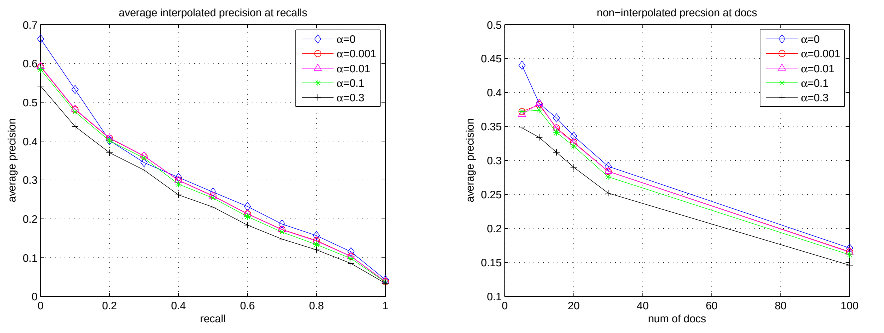
(a) Precision/recall Curve



(b) Precision for top ranked documents

Figure 4: Precision-recall curve (left) and precision for top ranked documents (right) using different values for α in the simplified Rocchio algorithm. The first 10 Google returned documents are used as the basis for query expansion. Lucene default scoring function is used to measure the document-query similarity.

previous studies of using WordNet for query expansion. The common belief for the failure of using WordNet is that the words in synonym sets tend to have multiple senses, and as a result, the expanded synonyms may lead to the retrieval of irrelevant documents.



(a) Precision/recall Curve (b) Precision for top ranked documents

Figure 5: Precision-recall curve (left) and precision for top ranked documents (right) for query expansion using the synonym sets defined in WordNet with varied weight α. Lucene default scoring function is used to measure the document-query similarity.

Comparison Fig.6 shows the comparison of different methods for query expansion:

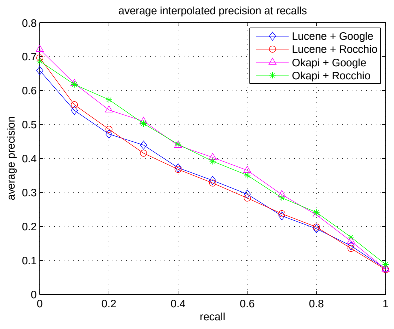
- Lucene+Google that uses Lucene default scoring function for document-query similarity measure and expands the original query based on the top 10 document returned by Google. Weight α is set to 0.6.

- Lucene+Rocchico that uses Lucene default scoring function for document-query similarity measure and expands the original query based on the top 15 document returned from the given document collection. Weight α is set to 0.1.

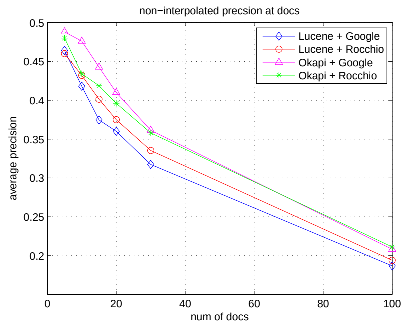
- Okapi+Google that uses the Okapi formulatoin for document-query similarity measure and expands the original query based on the top 15 document returned by Google. Weight α is set to 0.6.

- Okapi+Rocchico that uses the Okapi formulation for document-query similarity measure and expands the original query based on the top 15 document returned from the given document collection. Weight α is set to 0.1.

Overall, we observe that the Okapi method delivers significantly better performance for document retrieval than the Lucene default scoring function. The difference in the retrieval performance between using Google returned documents and documents returned from the given collection is relatively small.



(a) Precision/recall Curve



(b) Precision for top ranked documents

Figure 6: Precision-recall curve (left) and precision for top ranked documents (right) for different query expansion methods where m and α are chosen to optimize the performance.

4.2 Word Expansion

We used four methods to expand a single word or a small set of words: pseudo relevance feedback, Google returned documents based expansion, WordNet based expansion and latent semantic indexing expansion. The first three ones are explained in details in the previous sections.

5 Evaluation by Examining Expanded Query Words

We also manually check the expanded query words to see which makes more sense. In this experiment, we include the results from all four methods for query expansion, i.e. query expansion based on pseudo relevance feedback, Google returned documents, synonym sets defined in WordNet, and random walk model. Nine single word queries are used in our study. They are: aids, crop, education, Israel, legal, murder, politics, price, and stock. Table 1 to 9 include the first five words expanded by the four methods. We observed that the query expansion method based on random walk model appears to work best. Compared to the other methods, it is able to identify relevant keywords for almost all queries.

In these tables, PRFL represents Pseudo relevance feedback with Lucene default scoring, and PRFO represents Pseudo relevance feedback with Okapi BM25 scoring, and RW represents Random walk model.

Method	1	2	3	4	5
PRFL	aid	viru	infect	disea	drug
PRFO	aid	viru	infect	disea	drug
Google	hiv	aid	immunode	viru	immun
WordNet	aid	assist	attent	care	tend
RW	aid	viru	infect	disea	contra

Table 1: Query expansion for word ”aids”.

Method	1	2	3	4	5
PRFL	crop	farmer	insur	appl	gantz
PRFO	crop	farmer	drought	cotton	acr
Google	crop	noun	resiz	produc	programm
WordNet	crop	brows	clip	craw	cultiv
RW	crop	cent	bushel	soybean	corn

Table 2: Query expansion for word ”crop”

Method	1	2	3	4	5
PRFL	educ	univ	phd	school	cathol
PRFO	educ	univ	cathol	school	phd
Google	educ	student	teacher	inform	colleg
WordNet	educ				
LSI	educ	school	student	teacher	colleg

Table 3: Query expansion for word ”education”

Method	1	2	3	4	5
PRFL	israel	hungari	isra	ti	hungarian
PRFO	israel	isra	hungari	mubarak	taba
Google	israel	inform	geographi	economi	tourism
WordNet	israel	sion	yisrael	zion	
LSI	israel	isra	palestiniar	arab	plo

Table 4: Query expansion for word ”Israel”

Method	1	2	3	4	5
PRFL	legal	marcum	drug	countian	frazier
PRFO	legal	drug	durant	expen	marcum
Google	legal	onlin	dictionari	seafood	lesbian
WordNet	legal	effectu	sound		
LSI	legal	court	law	attornei	judg

Table 5: Query expansion for word "legal"

Method	1	2	3	4	5
PRFL	murder	caldwel	mathebula	degr	weed
PRFO	murder	westi	racket	thompson	convict
Google	murder	unlaw	malic	aforethought	26666
WordNet	murder	dispatch	execut	hit	mangl
LSI	murder	sentenc	convict	polic	death

Table 6: Query expansion for word "murder"

Method	1	2	3	4	5
PRFL	polit	relea	prison	redman	ministri
PRFO	polit	shim	cartoon	cartoonist	relea
Google	polit	opinion	cbsnews.com	reuters.com	obama
WordNet	polit				
LSI	polit	parti	dukaki	democrat	bush

Table 7: Query expansion for word "politics"

Method	1	2	3	4	5
PRFL	price	increa	cent	gallon	relea
PRFO	price	increa	whole	percent	inflat
Google	price	websit	electron	comput	priceutah.net
WordNet	price	cost	damag	term	toll
LSI	price	cent	stock	market	percent

Table 8: Query expansion for word "price"

Method	1	2	3	4	5
PRFL	stock	tokyo	nikkei	point	averag
PRFO	stock	index	futur	trade	otc
Google	stock	market	quot	data	nasdaq
WordNet	stock	ancestri	banal	blood	bloodlin
LSI	stock	nikkei	tokyo	point	exchang

Table 9: Query expansion for word ”stock”

References

[1] T. Joachims. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML ’97, pages 143–151, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

[2] S. E. Robertson, S. Walker, and M. Hancock-Beaulieu. Okapi at trec-7: Automatic ad hoc, filtering, vlc and interactive. In *TREC*, pages 199–210, 1998.