Query Expansion Using a Collection Dependent Probabilistic Latent Semantic Thesaurus

Laurence A.F. Park and Kotagiri Ramamohanarao

ARC Centre for Perceptive and Intelligent Machines in Complex Environments,
Department of Computer Science and Software Engineering,
The University of Melbourne, 3010, Australia
{lapark,rao}@csse.unimelb.edu.au

Abstract. Many queries on collections of text documents are too short to produce informative results. Automatic query expansion is a method of adding terms to the query without interaction from the user in order to obtain more refined results. In this investigation, we examine our novel automatic query expansion method using the probabilistic latent semantic thesaurus, which is based on probabilistic latent semantic analysis. We show how to construct the thesaurus by mining text documents for probabilistic term relationships, and we show that by using the latent semantic thesaurus, we can overcome many of the problems associated to latent semantic analysis on large document sets which were previously identified. Experiments using TREC document sets show that our term expansion method out performs the popular probabilistic pseudo-relevance feedback method by 7.3%.

1 Introduction

Short queries, consisting of only a few terms can be vague and hence cause an information retrieval system to return documents covering a broad number of topics which are not specific to the users information need. To assist the user, methods of query expansion have been developed, where the retrieval system adds terms to the short query in order to improve the precision of the results provided. This can be done with user interaction [6] or automatically without user interaction [8].

In this article, we will describe and examine our new method of automatic query expansion using a probabilistic latent semantic thesaurus. Our main contributions are: a generalisation model for ranking; and showing the probabilistic latent semantic thesaurus method is both efficient in storage and memory while yielding high precision. We show that our method of query expansion outperforms the popular BM25 pseudo-relevance feedback method by an average of 7.3% in terms of average reciprocal rank and also improves on our baseline BM25 by an average 8%.

This article will proceed as follows: In section 2, we briefly explain the information retrieval process and describes how the retrieval system can assist the user by adding to the query, this can be done with the local relevance feedback

Z.-H. Zhou, H. Li, and Q. Yang (Eds.): PAKDD 2007, LNAI 4426, pp. 224–235, 2007.

methods or the global thesaurus methods. In section 3, we further explain our new collection dependent thesaurus using probabilistic latent semantic analysis and show how to construct it. Experimental procedures and results are provided in section 4.

2 Query Term Expansion

Information retrieval systems are used to quickly assess whether a collection of unexamined text documents contains the information we desire. To the retrieval system, each text document is simply a sequence of terms. The query process requires a set of key terms to be supplied to the retrieval system, which are judged by the user as best describing their information need. Once the query is given, the retrieval system compares the query to the text documents and returns the top matching documents to the user. The user then evaluates whether the documents suit the information need. If the information need is met, the process is finished, but if it is not met, the user must ponder how to reword the query in order to obtain better results.

The problem with this process lies in the guess work involved in converting the information need into a set of query terms. The query can be expressed in many ways due to the versatility of our language. Unfortunately information retrieval systems use term matching to identify the documents relevant to the query, therefore to obtain the best results, the query must be expressed using the terms found in the document set. This implies that the user would require some knowledge of the content of the document collection to formulate a query. But as we mentioned, information retrieval systems are used to quickly assess whether a collection of unexamined text documents contains the information we desire, therefore we should not have to examine the document set in order to construct a query.

Rather than let the user manually examine the document set before querying, methods of term expansion have been derived that place the document analysis within the retrieval system. Term expansion is the process of adding terms to a query in order to create a query which is closer to the information need relative to the document set. The terms are chosen based on a similarity analysis of the query terms within the document set. To perform term expansion, we need a document-query scoring function ($S_q(d, Q)$) to rank documents based on the users query, a term scoring function ($S_t(t, Q)$) to select terms for the query expansion, and a document-expansion scoring function ($S_e(d, E)$) to rank documents based on the expansion terms. The final document score is a combination of the query and expansion term document scoring functions:

$$S(d, Q) = (1 - \alpha)S_q(d, Q) + \alpha S_e(d, E) \tag{1}$$

Note that using $\alpha = 0$ implies that there is no feedback used and $\alpha = 1$ implies that purely feedback is used. Typically, an α value of less than one is used to

put less emphasis on the expansion terms, so that they don't dominate the query. Each method we present will be based on the BM25 document scoring function, therefore they will all use the same document-query scoring function:

$$S_q(d,Q) = \sum_{t \in Q} w_{d,t} w_t \tag{2}$$

where

$$w_{d,t} = \frac{f_{d,t}(k_1+1)}{K + f_{d,t}} \qquad w_t = \log\left(\frac{N - f_t + 0.5}{f_t + 0.5}\right) \tag{3}$$

where $S_q(d,Q)$ is the score of document d based on the query Q, $f_{d,t}$ is the frequency of term t in document d, N is the number of documents in the collection, f_t is the number of documents containing term t, $K = k_1((1-b)+b\ dl/avdl)$, k_1 and b are constants, dl is the document length, and $avdl$ is the average document length.

In this section we will describe the two major methods of automatic term expansion called Pseudo-relevance feedback and Thesaurus expansion.

2.1 Pseudo-relevance Feedback

Interactive relevance feedback extends the involvement of the retrieval system in the information retrieval process to rely on user feedback. After the query has been supplied and the top matching documents have been calculated by the retrieval system, the system then proceeds by presenting the user with the matching documents and asks which are relevant to the query. Once the system receives the relevance information, it then continues by extracting terms from the set of relevant documents that will be included in the expanded query. After the query is formed, the retrieval system retrieves the documents that best match the new expanded query.

Pseudo-relevance feedback is a non-interactive version of the mentioned relevance feedback method. To remove the user interaction and hence speed up the query process, the retrieval system does not question the user about the relevance of the top matching documents to the query, but instead assumes that the documents that match the query are relevant. Terms are then extracted from this set of documents and used to build the expanded query.

A popular and effective pseudo-relevance feedback system comes from Robertson [4]. Given a query Q, the ten documents with the greatest $S_q(d,Q)$ are chosen for feedback. Using pseudo-relevance feedback, we assume that all ten documents are relevant to the query. Using terms from these ten documents, we must select a subset of terms to include in our expanded query. The selection process involves scoring each term and selecting the top terms to be included in our expanded query. The term scoring function used is:

$$S_t(t,Q) = f_{R,t} w_t \tag{4}$$

	dog	puppy	cat	kitten
dog	0.7	0.2	0.07	0.03
puppy	0.3	0.6	0.01	0.09
cat	0.06	0.04	0.7	0.2
kitten	0.02	0.08	0.6	0.3

Fig. 1. An example of a collection dependent thesaurus. All of the words found within the collection are listed with the probability of their relationship to each word. We can see that the words *dog* and *puppy* have a higher probability of being related to *dog* than the words *cat* and *kitten*.

where $f_{R,t}$ is the frequency of term t in the set of pseudo-relevant documents R. The terms with the greatest scores are then used to query the document collection using the document-expansion scoring function:

$$S_e(d, E) = S_q(d, E)$$

where E is the set of expansion terms. The final document score is calculated using equation 1 where $\alpha = 1/3$.

2.2 Collection Dependent Thesaurus Expansion

Relevance feedback is a local query expansion because it uses only a subset of the document set to calculate the set of expansion terms. Thesaurus expansion is a global query expansion because it makes use of the whole document set when calculating the set of expansion terms.

A thesaurus is a collection of words, where each word has an associated set of words that are related to it. Many thesauruses have been constructed manually, and can be used by those that have an understanding of the language. A typical entry in a manually built thesaurus contains the desired word and sets of related words grouped into different senses. To effectively find related words, we must know which sense to choose. Unfortunately, this is not an easy task for a machine and hence machine word-sense disambiguation is an active field in the area of computer science and linguistics.

A collection dependent thesaurus is one that is automatically built using the term frequencies found with a document collection. Since the thesaurus is document collection dependent, any word relationships found will be based on documents that can be retrieved. A collection dependent thesaurus is a square table that contains all of the words found within the collection and their relationship to each other. Each element of the table contains the probability of a word being related to another. A thesaurus built using a very small document set is shown in figure 1.

The information retrieval process using a collection dependent thesaurus is very similar to that of pseudo-relevance feedback. The difference being that the initial document retrieval step to obtain the candidates for the query expansion does not have to be performed since we already have a table of term relationships.

To obtain our set of expansion terms, we must choose terms that obtain the greatest score using the score function:

$$S_t(\tau, Q) = \sum_{t \in Q} P(\tau|t) w_t \tag{5}$$

where the thesaurus element $P(\tau|t)$ is the probability of term τ being related to term t and Q is the set of query terms. We can see that equation 5 gives higher scores to those terms having a higher probability of being related to the query. The document-expansion scores are calculated using:

$$S_e(d, E) = \sum_{t \in E} w_{d,t} S_t(t, Q) \tag{6}$$

where E is the set of terms with the greatest term score. The document-expansion score is the same as the document-query score except for the term weight (w_t) being replaced by the term score.

3 Probabilistic Latent Semantic Thesaurus

The previous section showed us how to use term expansion within the information retrieval process. We described how we can use a collection dependent thesaurus to obtain term relationships for our query expansion, but we did not explain how the thesaurus was built and the probabilistic values were obtained. In this section we will examine how to construct the thesaurus using probabilistic latent semantic analysis; but before we do, we will explain the concept of latent semantic analysis and show how it can be used to obtain term relationships.

3.1 Latent Semantic Analysis

Probabilistic [4] and vector space methods [1] of information retrieval base the document score on the occurrence of the query terms within the document. This implies that if the query terms do not appear within the document, it obtains a score of zero and is considered irrelevant to the query. Any terms that are not related to the query are ignored, even though their occurrence within a document could infer relevance.

Rather than observing the occurrence of terms, the method of latent semantic analysis [2] observes the occurrence of topics, where a topic is a set of related words. Its use becomes more intuitive once we observe the following document creation models. The document creation model with term based comparisons uses the following sequence:

1. the author starts by having an idea that needs to be conveyed
2. the idea is put to paper by choosing specific words
3. if other words were chosen during the writing process, the written document would not convey the same idea

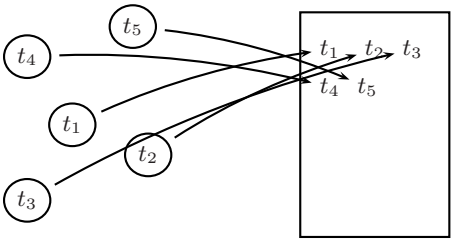

Fig. 2. A naïve document creation model. The author chooses specific terms for the document. If different terms are chosen the document will not convey the same message. This is the model used by retrieval systems that assume all terms are independent of each other.

We can see that this model (shown in figure 2) is not correct, since our language allows us to project the same idea using different words. This document creation model is projected in the probabilistic and vector space methods of information retrieval; if query terms do not appear in a document, the document is considered irrelevant even if it does contain related terms.

A more appropriate document creation model (shown in figure 3) uses the following sequence:

1. the author begins by having an idea that needs to be conveyed
2. the author chooses specific topics to convey the idea
3. the idea is put to paper by choosing words that are associated to each of the chosen topics
4. if different topics were chosen during the writing process, the written document would not convey the same idea

In this case, two documents containing different terms could project the same idea if the terms were associated to the same topics. This more realistic model takes into account the synonymy found in modern day languages by comparing topics rather than terms. Latent semantic analysis is the process of discovering these topics and their relationship to the set of terms.

3.2 Probabilistic Latent Semantic Analysis

Probabilistic latent semantic analysis (PLSA) [3] is the process of discovering the topics within a document set using probabilistic means. In this section we will describe how we can use it in our retrieval system.

The probability of choosing a specific term from a specific document within our document collection is given as:

$$P(d,t) = \frac{f_{d,t}}{\sum_{d \in D} \sum_{t \in T} f_{d,t}} \tag{7}$$

where D and T are the set of documents and terms respectively. Given the set of topics Z, we are able to form the following relationship between the set of documents D and set of terms T using Bayesian analysis:

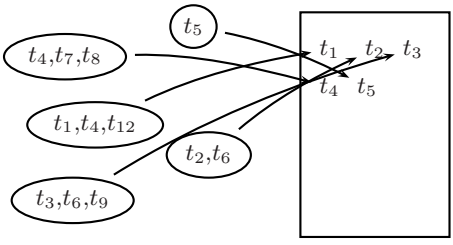

Fig. 3. The latent semantic analysis document model. The author chooses specific topics for the document and then chooses a term from the topic to place in the document. This model implies that documents containing different terms can convey the same message, as long as the replacement terms are associated to the same topic.

$$\begin{aligned} P(d,t) &= P(d|t)P(t) \\ &= \sum_{z \in Z} P(d|z)P(z|t)P(t) \\ &= \sum_{z \in Z} P(d|z)P(t|z)P(z) \end{aligned} \tag{8}$$

where d and t are conditionally independent given z, $d \in D$, and $t \in T$. Since the $P(d|z)$, $P(t|z)$ and $P(z)$ values are unknown, we must obtain the best fit that satisfies equation 8. Using expectation maximisation, we are able to obtain an approximation of the unknown probabilities $P(d|z)$, $P(t|z)$ and $P(z)$ for all d, t and z in D, T and Z respectively.

Before we obtain the probabilities, we must choose the size of Z. By choosing a small number of elements for Z (much less than the number of documents and terms in the document set), we make sure that our model is not over fitted. A small number of $z \in Z$ implies that there will be a small number of topics and hence, the documents and terms will cluster into topic sets.

3.3 Building the Thesaurus

Using probabilistic latent semantic analysis, we have obtained the probabilistic relationships between the topics and terms, and the topics and documents. To build a thesaurus, we need to calculate the probabilistic relationships between each of the terms. To do so, we have derived the following relationship:

$$\begin{aligned} P(t_x|t_y) &= \sum_{z \in Z} P(t_x|z)P(z|t_y) \\ &= \sum_{z \in Z} \frac{P(t_x|z)P(t_y|z)P(z)}{P(t_y)} \end{aligned} \tag{9}$$

where t_x and t_y are conditionally independent given z. To obtain the probabilistic term relationship values, we use the $P(t|z)$ and $P(z)$ probabilities obtained from PLSA, and:

$$P(t) = \frac{\sum_{d \in D} f_{d,t}}{\sum_{d \in D} \sum_{t \in T} f_{d,t}} \tag{10}$$

Once we have obtained the probabilistic term relationships, we store the values in a fast lookup index so that they can easily be accessed at the query time.

3.4 Thesaurus Performance

Every term will have a probabilistic relationship to every other term, therefore our thesaurus will be a $|T| \times |T|$ table of non-zero floating point values, where $|T|$ is the cardinality of T. This implies that the thesaurus will be very large for large document sets. This storage problem also exists in probabilistic latent semantic indexing.

Fortunately, our use of a thesaurus means that we do not need to store all of the terms. Each of the term relationships is based on the term samples found within the document set. We have shown that terms that are under-sampled (found in only a few documents) will not produce proper relationships [5], therefore it seems fitting to ignore the under-sampled terms. If we ignore all terms that are found in no more than N documents, we will remove a significant amount of terms due to the occurrence of terms following the Zipf distribution. By removing these terms, we are choosing to keep the terms that appear in at least N documents, which is directly related to the term weight (w_t). By choosing $N = 50$ [5], we reduce the storage required from a 4 gigabyte index to a 21 megabyte thesaurus.

We stated that there is a relationship between every term in the thesaurus. Therefore if we were to use any number of query terms, the expanded query would contain every term in the thesaurus. The query processing time is proportional to the number of query terms, therefore including every term in the query would lead to a very expensive query process. This query processing speed problem exists in every query expansion method and can be resolved by choosing only those terms that have the greatest term score ($S_t(t, Q)$) for the expansion. By doing so, we receive the top M related terms to the query. From this we can see that the query expansion size leads to a trade off between system precision and query speed.

As for the query speed, the difference between the pseudo-relevance feedback and thesaurus is the term expansion method. The former requires two lookups of a sparse index, while the latter requires one lookup of a dense index. This leads to similar query processing times.

4 Experiments

To examine the performance of our collection dependent thesaurus using probabilistic latent semantic analysis, we have run experiments on two well known document collections. The first document collection is the Associated Press articles from TREC disk-1 (AP1) containing 84,678 documents, the second is the

set of Associated Press articles from TREC disk-2 (AP2) containing 79,919 documents. For each of the document sets, we used the titles of queries 51 to 200 and the associated relevance judgements from TREC-1, 2 and 3.

Our experiments compared the increase in precision due to query expansion at various levels of expansion. We reported results using our probabilistic latent semantic thesaurus (PLST), pseudo-relevance feedback (PRFB) and a term co-occurrence thesaurus (COT). The increase in precision shown is compared to BM25 with no query expansion. The term co-occurrence thesaurus uses the thesaurus method found in section 2.2, where:

$$P(\tau|t) = \frac{\sum_{d \in D} f_{d,\tau} f_{d,t}}{\sum_{\tau \in T} \sum_{d \in D} f_{d,\tau} f_{d,t}} \tag{11}$$

where T is the set of terms and D is the set of documents.

We built a thesaurus for each of the document collections using the following suggested parameters [5]. The thesaurus included all terms that were found in at least 50 documents, the mixing parameter was set to $\alpha = 0.6$, and 100 topics were calculated. To compare our thesaurus method we also ran experiments using pseudo-relevance feedback on the same document sets using the suggested parameters of $\alpha = 0.25$ and using the top ten documents for feedback [4]. Within the BM25 model, we used the parameters $k_1 = 1.2$ and $b = 0.75$ [4].

The precision at 10 documents and average reciprocal rank increases are shown in figures 4 and 5 respectively. The increases are shown with respect to the BM25 (without expansion) ranking function. The two measures reported, measure the system for different uses. The reciprocal rank of a query is the inverse of the rank of the first relevant document (e.g. if the first relevant document is ranked third, the reciprocal rank is 1/3). The average reciprocal rank is the average of all reciprocal ranks from each query. If we are using the retrieval system to find one document, we would use this value to measure the system. Precision at 10 documents is the average number of relevant documents found in those that the system ranks in the top ten. We would use this measure if we wanted a few relevant documents.

The results show that the PLST outperforms the COT for all levels of expansion using both measures. We can see that our PLST method outperforms the pseudo-relevance feedback method in average reciprocal rank. In fact, we can see on both data sets that applying any expansion PRFB reduces the ARR. If we observe the precision at 10 documents, we find that PRFB provides better precision for low expansion sizes and PLST provides better precision for higher expansion sizes. If we use typical expansion sizes of 20 terms to PRFB and 100 terms for PLST, we find that PLST provides an average increase of 7.3% in ARR and 0.2% increase in prec10. This implies that for the typical Web surfer who only wants one relevant document, PLST is the better query expansion method to use, while for someone who wants a many relevant pages in the first ten ranked documents, either PLST or PRFB could be used.

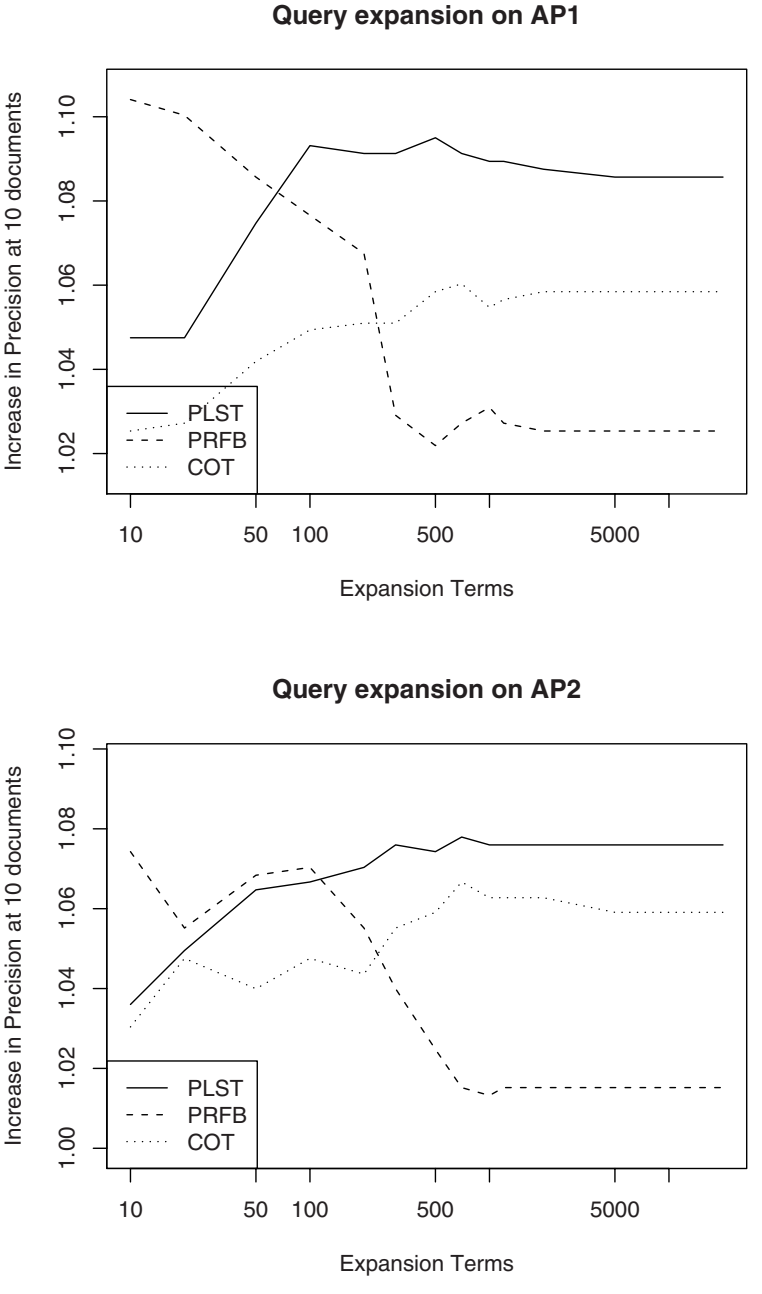

Fig. 4. A comparison of the increase in precision after 10 documents (prec10) due to query expansion of our collection dependent thesaurus using probabilistic latent semantic analysis (PLST) against pseudo-relevance feedback (PRFB) and a term co-occurrence thesaurus (COT) on the AP1 and AP2 document sets. The baseline BM25 (without expansion) precision after 10 documents is 0.3747 for AP1 and 0.3554 for AP2.

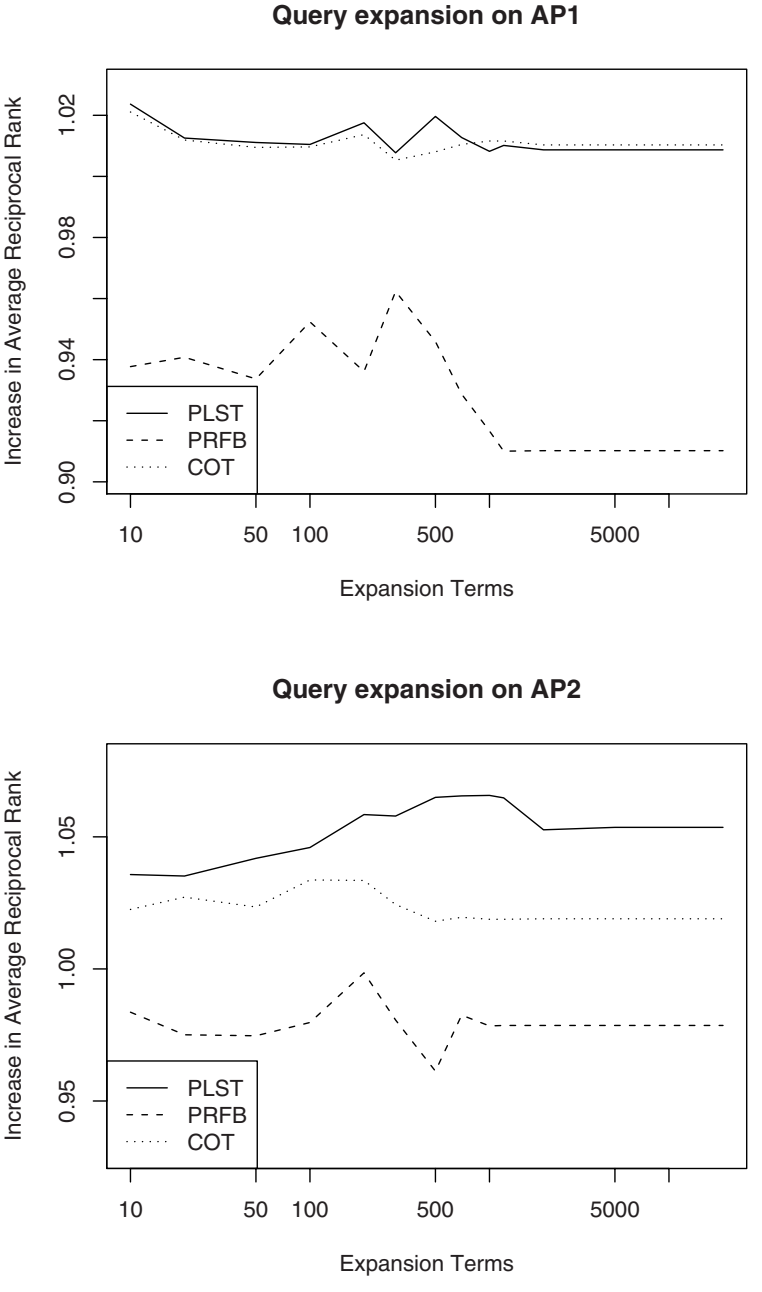

Fig. 5. A comparison of the increase in average reciprocal rank (ARR) due to query expansion of our collection dependent thesaurus using probabilistic latent semantic analysis (PLST) against pseudo-relevance feedback (PRFB) and a term co-occurrence thesaurus (COT) on the AP1 and AP2 document sets. The baseline BM25 (without expansion) average reciprocal rank is 0.6214 for AP1 and 0.5374 for AP2.

5 Conclusion

Automatic query expansion is a method of adding terms to the query without interaction from the user in order to obtain more refined results. We have presented our new method of automatic query expansion using a collection dependent thesaurus built with probabilistic latent semantic analysis. We have shown how to build the thesaurus using probabilistic latent semantic term relationships, and we have shown how to efficiently query the thesaurus in order to expand our query.

Experiments were performed and compared to the popular pseudo-relevance feedback using the BM25 weighting scheme and a term co-occurrence thesaurus. The results showed that our probabilistic latent semantic thesaurus outperformed the term co-occurrence thesaurus for all levels of recall and the pseudo-relevance feedback retrieval by an average 7.3% when one relevant document was desired, and an average 0.2% when observing the top ten ranked documents. This implies that a probabilistic latent semantic thesaurus would be the query expansion choice for the typical Web surfer.

References

1. Chris Buckley and Janet Walz. SMART in TREC 8. In Voorhees and Harman [7], pages 577–582.
2. Susan T. Dumais. Latent semantic indexing (lsi): Trec-3 report. In Donna Harman, editor, *The Third Text REtrieval Conference (TREC-3)*, pages 219–230, Gaithersburg, Md. 20899, March 1994. National Institute of Standards and Technology Special Publication 500-226.
3. Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM Press, 1999.
4. K. Sparck Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments, part 2. *Information Processing and Management*, 36(6):809—840, 2000.
5. Laurence A. F. Park and Kotagiri Ramamohanarao. Hybrid pre-query term expansion using latent semantic analysis. In Rajeev Rastogi, Katharina Morik, Max Bramer, and Xindong Wu, editors, *The Fourth IEEE International Conference on Data Mining*, pages 178–185, Los Alamitos, California, November 2004. IEEE Computer Society.
6. Xuehua Shen and ChengXiang Zhai. Active feedback in ad hoc information retrieval. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 59–66, New York, NY, USA, 2005. ACM Press.
7. Ellen M. Voorhees and Donna K. Harman, editors. *The Eighth Text REtrieval Conference (TREC-8)*, Gaithersburg, Md. 20899, November 1999. National Institute of Standards and Technology Special Publication 500-246, Department of Commerce, National Institute of Standards and Technology.
8. Ryen W. White, Ian Ruthven, and Joemon M. Jose. A study of factors affecting the utility of implicit relevance feedback. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 35–42, New York, NY, USA, 2005. ACM Press.