Semisupervised Query Expansion with Minimal Feedback

Masayuki Okabe and Seiji Yamada

Abstract—Query expansion is an information retrieval technique in which new query terms are selected to improve search performance. Although useful terms can be extracted from documents whose relevance is already known, it is difficult to get enough of such feedback from a user in actual use. We propose a query expansion method that performs well even if a user makes practically minimum effort, that is, chooses only a single relevant document. To improve searches in these conditions, we made two refinements to a well-known query expansion method. One uses transductive learning to obtain pseudorelevant documents, thereby increasing the total number of source documents from which expansion terms can be extracted. The other is a modified parameter estimation method that aggregates the predictions of multiple learning trials to sort candidate terms for expansion by importance. Experimental results show that our method outperforms traditional methods and is comparable to a state-of-the-art method.

Index Terms—Information search and retrieval, query formulation, relevance feedback, machine learning.

1 Introduction

When searching for information, people usually do not obtain much relevant information in the initial search and need to modify queries and search again until they get satisfactory results. Query expansion is an automatic search technique that extracts useful terms from selected documents to improve the search. Although many techniques have been proposed, a standard method is to use relevant information provided by a user [1]. Although a large amount of relevant information would help in selecting effective search terms, users generally do not provide a lot of it. They usually provide little or no relevant feedback because identifying relevant documents is very time consuming [2].

Pseudofeedback, which assumes that top-ranked documents are relevant, is a standard way of enhancing initial search results that requires no human input regarding relevance. Though this technique has been proven to be effective, its performance depends on the quality of the initial search. If there are no relevant documents among the search results, pseudofeedback may improve little on the initial result [3]. There are several improved pseudofeedback approaches, including use of clustering [4], one-class support vector machine (SVM) [5], large external corpora [6], and implicit feedback [7].

In contrast to the above approaches, we consider a type of manual feedback where users stop the search as soon as the first relevant document is found. This approach is very important because even a single relevant document is much more effective than many nonrelevant documents [3]. Although this type of feedback can be gotten simply by inputting one relevant document into the traditional retrieval methods, some methods, such as OKAPI [8] and SMART [9], cannot get enough statistics and variety to expand query terms because they use frequency of relevant documents to select expansion terms.

• *M. Okabe is with the Information and Media Center, Toyohashi University of Technology, Tempaku 1-1, Toyohashi, Aichi, 441-8580, Japan. E-mail: okabe@imc.tut.ac.jp.*
• *S. Yamada is with the National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Toyoko 101-8430, Japan. E-mail: seiji@nii.ac.jp.*

Manuscript received 30 Aug. 2006; revised 16 Apr. 2007; accepted 19 July 2007; published online 2 Aug. 2007.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-0407-0806.
Digital Object Identifier no. 10.1109/TKDE.2007.190646.

We use transductive learning [10] to overcome the problem of lack of relevant information. This learning approach is more effective than traditional inductive learning, especially when there are few training examples. Using this technique, we attempt to increase the number of pseudorelevant documents based on relevant documents. Once pseudorelevant documents are obtained, they are used to select expansion terms. Simple application of transductive learning does not work well because parameter tuning is very difficult. Therefore, we modify the basic scoring function of OKAPI to apply learning to query expansion.

The remainder of this paper is structured as follows: Section 2 introduces our query expansion method step by step and explains two fundamental techniques: Robertson's *wpq* method and the Spectral Graph Transducer (SGT) algorithm. Section 3 explains how our method overcomes problems that occur when SGT is applied to it. Section 4 demonstrates the effectiveness of our method as compared with other query expansion techniques. Section 5 analyzes the experimental results. Finally, Section 6 summarizes our findings.

2 Basic Methods

2.1 Query Expansion with Transductive Learning

First of all, we explain our query expansion method step by step. In the following procedure, (U) represents a step executed by a user and (S) represents a step executed by the system.

1. **Initial search (U).** A user starts a retrieval by inputting a query to an IR system.
2. **Minimal user feedback for relevance judgment (U).** When an IR system returns, a hit list for the initial query, the user checks it and judges whether or not each document is relevant in descending order of the ranking. At a minimum, the user stops perusing documents after he finds the first relevant one. Based on this minimum, our method uses one relevant and several irrelevant documents as training examples for transductive learning in the next step.
3. **[Retrieval/Discovery] of other relevant documents by transductive learning (S).** After the user selects a relevant document, a transductive learning algorithm is used to find relevant documents the user did not find. As shown in Fig. 1, it carries out the retrieval process by assigning a label (relevant or irrelevant) to each unjudged (unlabeled) document based on a small set of judged (labeled) data from the previous step.
    - Selecting expansion terms (S). After a set of relevant documents has been retrieved, the system calculates a score for each term in those documents. This conventional scoring function will be described later. The terms with the top m scores are selected for expansion.
    - Search with an expanded query (S). The initial query is expanded by the additional terms as described above, and the expanded query is inputted to the IR system to get a new hit list. This is the end of the first cycle of query expansion.

In these procedures, we naturally introduced learning into query expansion as an effective way of automatically finding relevant documents. Thus, we do not need to modify the basic query expansion procedure and can fully utilize the potential power of the basic query expansion.

2.2 Score Function for Query Expansion

Many query expansion techniques have been proposed. Although a few techniques use precategorized training documents to

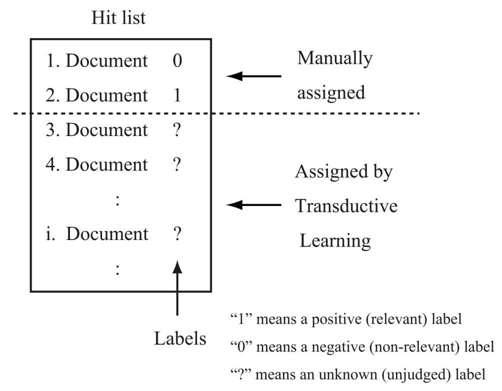
Hit list
1. Document 0
2. Document 1
3. Document ?
4. Document ?
i. Document ?
Manually assigned
Assigned by Transductive Learning
Labels
"1" means a positive (relevant) label
"0" means a negative (non-relevant) label
"?" means an unknown (unjudged) label

Fig. 1. Method of tentatively identifying relevant documents.

prepare expansion terms for domain specific search [11], [12], most of them are based on relevance feedback that is given automatically or manually at the moment of ad hoc retrieval. How well this type of method performs depends on the amount of feedback it receives and on its ability to rank candidate terms for expansion. The scoring function proposed in Robertson's *wpq* method is generally considered to be the effective one and has been used in many researches [1], [13], [14], [15]. Our method is based on this function. Since we actually use a modified version of the function, we introduce it here and discuss it in Section 3.

The function calculates the score of each term using the following equation:

$$score(t) = \left(\frac{r_t}{R} - \frac{n_t - r_t}{N - R}\right) * \log \frac{r_t/(R - r_t)}{(n_t - r_t)/(N - n_t - R + r_t)}, \quad (1)$$

where r_t is the number of relevant documents containing the term t, n_t is the number of documents containing t, R is the number of relevant documents for a query, and N is the number of all documents in the collection. This equation is based on the Robertson/Spark Jones weight [16], which is the core of the term weighting function in the Okapi system [8]. Note that this equation is derived from a probabilistic framework, which can be shown by transforming it as follows:

$$score(t) = (p_t - q_t) \log \frac{p_t(1 - q_t)}{q_t(1 - p_t)}, \quad (2)$$

$$p_t = \frac{r_t}{R}, \quad (3)$$

$$q_t = \frac{n_t - r_t}{N - R}, \quad (4)$$

where p_t and q_t represent the probability that a term t appears in relevant and nonrelevant documents, respectively. Equations (3) and (4) can be understood as estimates of these probabilities. This means that a term's score becomes higher when it appears more frequently in relevant documents and less frequently in nonrelevant documents.

However, if a user only identifies a single relevant document, as in this work, p_t is 0 or 1 based on (3) because both R and r_t are 1. Similarly, q_t is $\frac{n_t}{N-1}$ or $\frac{n_t-1}{N-1}$ based on (3). Since scores are mostly determined by the value of n_t, there would be little advantage in the feedback. The value of R and r_t must be increased for p_t and q_t to be estimated with higher reliability. That is why we use transductive learning, which is effective even if there are only a few training examples.

2.3 Transductive Learning

Transductive learning is a machine learning technique based on the transduction that creates classification labels for test data directly without making any approximate function from a training data [10]. The learning task is defined on a data set X of n points. X consists of a training data set $L = (\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_l)$ and a test data set $U = (\vec{x}_{l+1}, \vec{x}_{l+2}, \ldots, \vec{x}_{l+u})$, typically, $l \ll u$. The purpose of the learning is to assign a label to each data point in U under the condition that the label of each data point in L is given.

Normal inductive learning consists of two phases, a *learning phase* and an *inference phase*. In the learning phase, a subset of data points L and labels Y_L is given as training examples. The learner will produce a model to predict labels for the rest of the data points in U. Using the model in the inference phase, we can finally get those labels for Y_U. In contrast to inductive learning, transductive learning uses both L and U. Moreover, the inference phase is not separated from the learning phase. Learning and inference are conducted at the same time in a transductive setting. The transductive learner presumes labels of test examples using similarities between labeled and unlabeled data points to complement the lack of training examples. Transductive learning is well suited to settings in which $|L|$ is very small.

Transductive learning or semisupervised learning has recently become a popular subject in the machine learning field. Several algorithms have been proposed, and their superiority in various learning tasks has been demonstrated [17], [18], [19]. To apply transductive learning to our query expansion, we chose SGT [20], a state-of-the-art transductive learning algorithm. SGT formalizes the problem of assigning labels to U with an optimization problem of the constrained ratio cut in an undirected graph. By solving its relaxed problem, it produces a solution that approximates the original one.

SGT first makes an undirected graph G, where vertices are directed from each datum x_i to the k data most similar to x_i. Then, it divides the graph into two parts. One part includes all the positive data but no negative data, and the other part contains negative but no positive data. Finally, it assigns common labels to all data in each part. The learning performance depends strongly on how graph G is divided.

This problem can be formulated by the following optimization problem:

$$\begin{array}{lll} \min\limits_{\vec{y}} & \dfrac{\mathrm{cut}(G^+, G^-)}{|\{i : y_i = 1\}||\{i : y_i = -1\}|} & \\ s.t. & y_i = 1, \quad \textit{if } x_i \textit{ is positive} & \\ & y_i = -1, \quad \textit{if } x_i \textit{ is negative} & \\ & \vec{y} \in \{+1, -1\}^n. & \end{array} \quad (5)$$

If there is no constraint, this problem is a simple MinCut problem. Solving it without those constraints raises the possibility of producing a very biased partitioning in terms of the number of data. This partitioning often produces undesirable results. To avoid this risk, SGT adds the above constraints, which are used to create a balanced partitioning. However, adding the constraints makes the problem NP-hard. SGT transforms it to a relaxed one and creates an approximate solution.

When applying SGT to our query expansion, a whole data set X corresponds to a set of top n ranked documents in a hit list. X does not correspond to a whole document collection because the number of documents in a collection is too huge (normally, it is more than a hundred thousand) for any learning system to process. L corresponds to a set of manually judged documents, relevant documents, and nonrelevant documents. U corresponds to a set of documents whose relevance is unknown. SGT is used to determine the relevance of documents in U.

By solving the problem, SGT assigns a value z_i to each document in U. If the problem is not relaxed, z_i takes $\gamma_+ = +\sqrt{\frac{1-p}{p}}$ (for relevant documents) or $\gamma_- = -\sqrt{\frac{p}{1-p}}$ (for nonrelevant documents). Here, p is a fraction of relevant documents in X. However, the z_is assigned by solving the relaxed problem are real values. p is decided by a threshold to make a binary label assignment. However, the threshold cannot be decided unless p is fixed. Thus, p and threshold are dependent of each other. To produce the final decision, SGT parameterizes p (we represent it as f_p in the following description) and estimates γ_+ and γ_- as $\hat{\gamma}_+ = +\sqrt{\frac{1-f_p}{f_p}}$ and $\hat{\gamma}_- = -\sqrt{\frac{f_p}{1-f_p}}$, respectively. SGT sets the middle point between $\hat{\gamma}_+$ and $\hat{\gamma}_-$ as a threshold for the final decision. Here, we need to pay close attention to the behavior of z_i when changing f_p. Although the value of f_p does not need to be rigid according to Joachims' work [20], we observed that the number of positive data that SGT assigns differed greatly in our preliminary experiments. Assuming the minimum of one relevant document, the amount of training data is extremely small, so z_i does not change as f_p changes. This means that the value of the threshold, namely, the value of f_p, dominates the learning results in our method. Because these results directly affect our method's performance, we need an appropriate way of setting f_p, which will be described in Section 3.

3 Parameter Estimations Based on Multiple SGT Predictions

3.1 Sampling Values for the Fraction of Positive Examples

SGT uses two estimation methods to set f_p automatically [20]. The first sets the fraction of positive data directly to f_p. However, because the amount of labeled data is so small, we cannot get a reliable value for f_p. The second method takes a heuristic approach, iteratively approximating f_p to an actual fraction of positive data. In practice, starting from 0.5, SGT resets f_p to the fraction of positive data assigned in each learning trial until the difference between preset f_p and an actual fraction converges into a predefined range. The problem with this approach is that SGT does not guarantee the convergence. In fact, it did not work well in our preliminary experiments.

First, it is unrealistic to set f_p precisely in advance, especially when there is very little training data as in our condition. If f_p is set to a small value, we are confronted with the problem described in Section 2, in which many terms have the same p_t, and q_t is the only factor that differentiates term candidates for query expansion. Instead of setting a single f_p, we take another approach, which runs multiple SGT trials with a variable f_p based on a sampling procedure [21]. **Algorithm 1** illustrates the procedure that returns a set of values to which f_p can be set. When choosing a sampling interval for this procedure, we consulted Buckley's work [22], which tried to approximate a recall-precision value at rank n in a hit list using linear regression. On the basis of this work, we assume that the number of relevant documents increases in proportion to the logarithm of the number of n. Thus, we change f_p based on the integral multiplication of $\ln(n)$. We sample less than 10 different f_p in the procedure. Although the number of times we should change f_p cannot be decided theoretically, it is important that the sampling number should not be too small or too large in actual use. The objective of this approach is not to find a single appropriate value of f_p but to increase the variety of term candidates by increasing the number of relevant documents. After a prediction value assigned by SGT has been aggregated in each trial, each document has a real value between 0 and 1 (not a binary value). This value represents the possibility that a document is relevant. Based on this value, any document that has a positive possibility value can be considered a relevant document. However, the score of candidate terms is calculated based on the possibility value of the document where the candidates appear.

Algorithm 1 Sampling Procedure for f_p

```
 1: Input : $n$            // top $n$ documents in a hit list
 2: Output: $S$            // a set of values for $f_p$
 3: $piv \Leftarrow \ln(n)$;      // sampling interval
 4: for $i = 1$ to 10 do
 5:     $v = (piv * i)/n$;
 6:     if $v > 1.0$ then
 7:         exit;
 8:     else
 9:         add $v$ to $S$;
10:     end if
11: end for
```

3.2 Modified Estimations for p_t and q_t

Once a set of sampling points $S = \{f_p^i : i = 1 \sim 10\}$ is determined, we run SGT using f_p^i in turn. Then, we aggregate each prediction value when calculating p_t and q_t as follows:

$$p_t = \frac{\sum_i r_t^i}{\sum_i R_i}, \tag{6}$$

$$q_t = \frac{\sum_i (n_t - r_t^i)}{\sum_i (N - R_i)}, \tag{7}$$

where R_i is the number of documents that SGT predicts as relevant with the ith value of f_p^i, and r_t^i is the number of documents in R_i, where a term t appears. In each trial, SGT predicts the relevance of documents by a binary value of 1 (for relevant) and 0 (for nonrelevant). However, if multiple predictions are aggregated, the binary prediction value can be changed to a real value that can represent rankings of relevance of documents. The main merit of this approach in comparison with fixing f_p to a single value is that it can differentiate a value of p_t even if the number of training examples L is small. Since the number of relevant documents is generally small, f_p increases gradually. Furthermore, by using a small f_p first, we keep the same ranking of documents and avoid the risk of invalid f_p.

4 Experiments

In this section, we present evidence that demonstrates the superiority of our query expansion method. We also compare our method with traditional methods.

4.1 Data Set and Evaluation Measures

We used the data set from the ad hoc track of the TREC-8 workshop [23] in our experiments. This data set consists of about 520,000 news documents on 50 topics (No.401-450) and relevance judgments for the topics. Before the experiment, we removed stopwords from documents and applied stemming. We extracted query terms for each topic's initial search from the title field of a topic.

We used three evaluation measures: P10, P30, and mean average precision (MAP). P10 and P30 are the precision for the first 10 and 30 retrieved documents, respectively. MAP is the average precision for a single topic. It is the mean of the precision obtained when each relevant document is retrieved. Zero is used

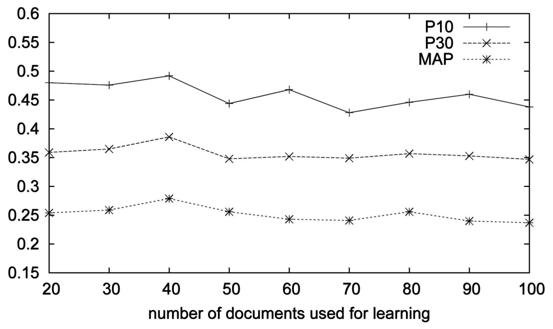

Fig. 2. Results for the parameter n.

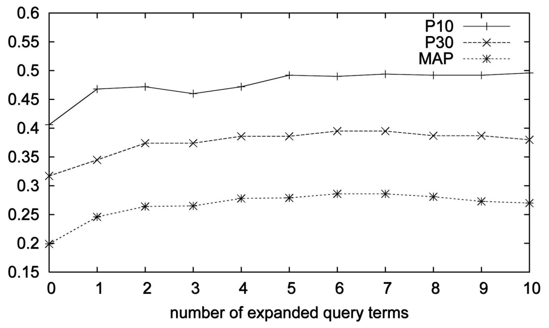

Fig. 3. Results for the parameter q.

as the precision for relevant documents that are not retrieved. P10 and P30 are user-oriented measures intended to evaluate the upper part of a ranking. This type of measure is suited to an ad hoc search, where a user is satisfied if just a few relevant documents are found. In contrast, MAP is a system-oriented measure that is based on recall and considers both top-ranked documents and relatively low-ranked documents. This type of measure is suitable when a user wants to collect as many relevant documents as possible. Since our method is not suited to types of retrieval, we use these three measures.

4.2 Parameter Tuning

We first tested our method's performance using various parameters. We used Joachims' SGT package[1] in our experiments. Parameters for SGT were set to default values except for k (the number of nearest neighbors) and d (the number of eigenvalues to use for learning). These were set as $k = 0.5 * n$ and $d = 0.8 * n$, respectively, where n is the number of data used for learning (top n documents in a hit list). Each datum is represented as a document vector with normal $tf \cdot idf$ weighting. SGT uses the cosine coefficient as a similarity measure.

Fig. 2 shows the results of our method when changing n from 20 to 100. The number of expansion (added) terms q is fixed to 5. Though the graph of P10 is not stable, all measures peaked at $n = 40$. Fig. 3 shows the results of our method when q was changed from 0 to 10. n was set to 40. In this graph, P10 is also unstable until $q = 5$, but it maintains constant values for $q > 6$. P30 and MAP gradually increased and reached peaks at $q = 6$ and gradually decreased after that. The best parameters for this data set are $n = 40$ and $q = 6$. We used these parameters in the following comparison.

4.3 Comparison with Other Methods

We compared our method with three others: OKAPI [24], SMART [25], and LANG [26]. Since OKAPI is a basic retrieval model of our method, we chose it to test the utility of our extension. SMART is an OKAPI competitor, as shown by the TREC-8 results. LANG is a language modeling approach that outscored OKAPI on the TREC-8 data set, as shown in [26]. Detailed descriptions of these models can be found in [24], [25], [26]. We implemented them using the same scoring formulas described in the above references. Parameters of OKAPI (BM25) are set as $k_1 = 1.2$, $k_3 = 1{,}000$, and $b = 0.75$. We tested various $0 \leq q \leq 30$ and selected the best one for OKAPI and SMART. In addition to q, we tested all combinations of parameters $10 \geq \alpha \geq \beta \geq \gamma \geq 1$ to update the SMART query vector and to identify the best one. We used the mixture approach for LANG [26]. We tested various $0.1 \leq \alpha \leq 0.9$ to determine the best balance of the probabilistic model in LANG (α is different from the one in SMART). All terms having more than 0.001 are added as query expansion. Table 1 shows the results with parameters optimized as described above.

1. http://sgt.joachims.org.

In addition to comparing each method with manual feedback, we also attached the results of initial retrieval and pseudofeedback for reference. A summary of these results is also shown in Table 1. The columns Man-rsd and Man are manual feedback results, and Pse and Ini are pseudofeedback and initial retrieval results, respectively. The difference between Man-rsd and Man is whether it is evaluated using the residual collection or the whole collection. Though manual feedback is usually evaluated using the residual collection where already seen (judged) relevant and nonrelevant documents are removed, we added Man as a comparison with Pse and Ini.

The best parameters for manual feedback (see the column of Man-rsd in Table 1) are given as follows: SGT: $n = 40$, $q = 6$, OKAPI: $q = 0$, SMART: $q = 0$, $(\alpha, \beta, \gamma) = (3, 0, 0)$, and LANG: $\alpha = 0.8$. The relevance judgment procedure is the same as described in Section 2 in each method. We set $q = 0$ for OKAPI and SMART because when we set q larger than 1, they got worse results even than initial retrieval.[2] However, SGT is used, OKAPI performs 12 percent $\sim$ 18 percent better on all measures. SGT performed 8 percent $\sim$ 12 percent better than SMART on all measures. In contrast, SGT performed 6 percent worse than LANG on P10 and about the same on P30 and MAP.

Although it is not a fair comparison, it can be seen that our method is better than any of the pseudofeedback methods. Parameters set for each method in pseudofeedback retrieval are listed as follows: OKAPI: $q = 6$, SMART: $q = 16$, $(\alpha, \beta, \gamma) = (7, 7, 4)$, and LANG: $\alpha = 0.8$.

5 Discussion

The amount of manual feedback assumed in our experiment was clearly not sufficient for OKAPI and SMART whose results were marginally better than the initial retrieval but worse than the pseudofeedback, which does not use any manual feedback. Applying our extension to OKAPI with manual feedback produces better results than pseudofeedback. Although SGT does not perform as well as LANG on the P10 evaluation measure, it is comparable on the other two measures.

There is a big difference between LANG and other methods in the number of expansion terms q. The average q in LANG is about 255. In contrast, qs of other methods are less than 20. The computational cost of working with large numbers of query terms might be critical in some computer environments. The number of documents that require manual feedback, which affects the cost of user feedback, averaged to 6.02 (minimum is 1, maximum is 121, and variance is 294.7) in our experiments. This means that a user

2. The poor performance is caused by the lack of relevance information. Query expansion works well if more relevant documents are given. For example, OKAPI-Man ($q = 20$) shows $\mathrm{P10} = 0.604$ and $\mathrm{MAP} = 0.281$, and SMART-Man ($q = 20, \alpha = 3, \beta = 2$, and $\gamma = 0$) shows $\mathrm{P10} = 0.666$ and $\mathrm{MAP} = 0.320$ when three relevant documents are given in the feedback.

TABLE 1
Results for All Feedback Methods

	P10				P30				MAP			
	Man-rsd	Man	Pse	Ini	Man-rsd	Man	Pse	Ini	Man-rsd	Man	Pse	Ini
SGT	0.490	0.520	-	-	0.395	0.409	-	-	0.286	0.301	-	-
OKAPI	0.436	0.472	0.468	0.466	0.339	0.352	0.382	0.345	0.242	0.245	0.264	0.239
SMART	0.426	0.464	0.480	0.460	0.325	0.336	0.363	0.336	0.227	0.233	0.267	0.229
LANG	0.524	0.558	0.462	0.442	0.396	0.419	0.390	0.355	0.290	0.308	0.289	0.258

needs to judge 6.02 documents until he or she finds the first relevant document when using our method. This seems high. However, if we eliminate topics with maximum values as exceptional cases, the average number drops to 3.6 (the maximum was 27, and the variance was 24.9). LANG's average is not very different from the above values. In addition to reducing the number of documents that must be judged, we can also reduce the cognitive load required to judge the relevance of documents by refining the user interface. For example, we can use clustering to collect up similar documents and have users judge only the representative documents in a clustered group. Using this approach, users can judge more documents with less effort.

Computational time of our query expansion method increases linearly with the number of unlabeled documents to be used for learning. For example, it took about 5.0 sec with 40 unlabeled documents in our environment (Pentium4-M, 1.8-GHz processor, 1 Gbye of memory). Since our experimental program is written as perl script, we can reduce the time with more sophisticated implementation.

6 Conclusion

In this paper, we proposed a query expansion method using practical manual feedback where relevance of documents is only given until a user finds one relevant document. Since the amount of relevant information is so small, we use the SGT algorithm, a transductive learning technique, to obtain pseudorelevant documents. We then use the SGT results in the extended *wpq* method. That was our original extension where the scores of query term candidates were calculated by aggregating several SGT results by slightly changing the parameters. In our experiments, we first showed how the number of documents used in SGT, and the number of expansion terms affects our method's performance. The results showed that the method performs well when the number is relatively small. However, our method is not very sensitive to the parameters. We also compared our method with other three techniques. As for the retrieval with manual feedback, we confirmed that our method improved the Okapi model and is comparable to a state-of-the-art method based on the language model. As a reference, we also compared our method with the results of a pseudofeedback. Although our method requires human effort, it enables users to work more efficiently.

Acknowledgments

The authors would like to sincerely thank the editor and the anonymous reviewers for their valuable comments and suggestions. This research was supported by MEXT, KAKENHI (18700084).

References

[1] I. Ruthven, "Re-Examining the Potential Effectiveness of Interactive Query Expansion," *Proc. 26th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 213-220, 2003.
[2] S. Dumais et al., "Sigir 2003 Workshop Report: Implicit Measures of User Interests and Preferences," *Proc. SIGIR Forum,* 2003.
[3] S.E. Robertson and S. Walker, "On Relevance Weights with Little Relevance Information," *Proc. 20th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 16-24, 1997.
[4] M. Iwayama, "Relevance Feedback with a Small Number of Relevance Judgements: Incremental Relevance Feedback vs. Document Clustering," *Proc. 23rd Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 10-16, 2000.
[5] T. Onoda et al., "Non-Relevance Feedback Document Retrieval," *Proc. 2004 IEEE Conf. Cybernetics and Intelligent Systems,* pp. 456-461, 2004.
[6] F. Diaz and D. Metzler, "Improving the Estimation of Relevance Models Using Large External Corpora," *Proc. 29th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 154-161, 2006.
[7] X. Shen, B. Tan, and C. Zhai, "Context-Sensitive Information Retrieval Using Implicit Feedback," *Proc. 28th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 43-50, 2005.
[8] S.E. Robertson, "Overview of the Okapi Projects," *J. Documentation,* vol. 53, no. 1, pp. 3-7, 1997.
[9] G. Salton, "The Smart Document Retrieval Project," *Proc. 14th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 356-358, 1991.
[10] V. Vapnik, *Statistical Learning Theory.* Wiley, 1998.
[11] G.W. Flake et al., "Extracting Query Modification from Nonlinear SVMS," *Proc. 11th Int'l World Wide Web Conf.,* 2002.
[12] S. Oyama et al., "Keyword Spices: A New Method for Building Domain-Specific Web Search Engines," *Proc. 17th Int'l Joint Conf. Artificial Intelligence,* 2001.
[13] R.W. White and D. Kelly, "A Study on the Effects of Personalization and Task Information on Implicit Feedback Performance," *Proc. 15th ACM Int'l Conf. Information and Knowledge Management (CIKM '06),* pp. 297-306, 2006.
[14] A. Amir, M. Berg, and H. Permuter, "Mutual Relevance Feedback for Multimodal Query Formulation in Video Retrieval," *Proc. Seventh ACM SIGMM Int'l Workshop Multimedia Information Retrieval (MIR '05),* pp. 17-24, 2005.
[15] S. Yu et al., "Improving Pseudo-Relevance Feedback in Web Information Retrieval Using Web Page Segmentation," *Proc. 12th Int'l World Wide Web Conf.,* 2003.
[16] S.E. Robertson, "On Term Selection for Query Expansion," *J. Documentation,* vol. 46, no. 4, pp. 359-364, 1990.
[17] T. Joachims, "Transductive Inference for Text Classification Using Support Vector Machines," *Proc. Int'l Conf. Machine Learning (ICML '99),* 1999.
[18] X. Zhu et al., "Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions," *Proc. 20th Int'l Conf. Machine Learning,* pp. 912-914, 2003.
[19] A. Blum et al., "Semi-Supervised Learning Using Randomized Mincuts," *Proc. Int'l Conf. Machine Learning (ICML '04),* 2004.
[20] T. Joachims, "Transductive Learning via Spectral Graph Partitioning," *Proc. Int'l Conf. Machine Learning (ICML '03),* pp. 143-151, 2003.
[21] M. Okabe, K. Umemura, and S. Yamada, "Query Expansion with the Minimum User Feedback by Transductive Learning," *Proc. Human Language Technology Conf./Conf. Empirical Methods in Natural Language Processing,* pp. 963-970, 2005.
[22] C. Buckley, G. Salton, and J. Allan, "The Effect of Adding Relevance Information in a Relevance Feedback Environment," *Proc. 14th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 292-300, 1994.
[23] E. Voorhees and D. Harman, "Overview of the Eighth Text Retrieval Conf.," *Proc. Eighth Text REtrieval Conf. (TREC '98),* 1999.
[24] S.E. Robertson and S. Walker, "Okapi/Keenbow at TREC-8," *Proc. Eighth Text REtrieval Conf. (TREC '99),* 1999.
[25] A. Singhal, J. Choi, D. Hindle, D.D. Lewis, and F. Pereira, "AT&T at TREC-7," *Proc. Seventh Text REtrieval Conf. (TREC '98),* 1998.
[26] C. Zhai and J. Lafferty, "Model-Based Feedback in the Language Modeling Approach to Information Retrieval," *Proc. 10th Int'l Conf. Information and Knowledge Management (CIKM '01),* pp. 403-410, 2001.