Optimizing Language Model Information Retrieval System with Expectation Maximization Algorithm

Justin Liang-Te Chiu
Department of Computer Science
and Information Engineering,
National Taiwan University
#1 Roosevelt Rd. Sec. 4, Taipei,
Taiwan 106, ROC
b94902009@ntu.edu.tw

Jyun-Wei Huang
Department of Computer Science
and Engineering,
Yuan Ze University
#135 Yuan-Tung Road, Chungli,
Taoyuan,Taiwan,ROC
s976017@mail.yzu.edu.tw

Abstract

Statistical language modeling (SLM) has been used in many different domains for decades and has also been applied to information retrieval (IR) recently. Documents retrieved using this approach are ranked according their probability of generating the given query. In this paper, we present a novel approach that employs the generalized Expectation Maximization (EM) algorithm to improve language models by representing their parameters as observation probabilities of Hidden Markov Models (HMM). In the experiments, we demonstrate that our method outperforms standard SLM-based and tf.idf-based methods on TREC 2005 HARD Track data.

1 Introduction

In 1945, soon after the computer was invented, Vannevar Bush wrote a famous article---“As we may think” (V. Bush, 1996), which formed the basis of research into Information Retrieval (IR). The pioneers in IR developed two models for ranking: the vector space model (G. Salton and M. J. McGill, 1986) and the probabilistic model (S. E. Robertson and S. Jones, 1976). Since then, the research of classical probabilistic models of relevance has been widely studied. For example, Robertson (S. E. Robertson and S. Walker, 1994; S. E. Robertson, 1977) modeled word occurrences into relevant or non-relevant classes, and ranked documents according to the probabilities they belong to the relevant one. In 1998, Ponte and Croft (1998) proposed a language modeling framework which opens a new point of view in IR. In this approach, they gave up the model of relevance; instead, they treated query generation as random sampling from every document model. The retrieval results were based on the probabilities that a document can generate the query string. Several improvements were proposed after their work. Song and Croft (1999), for example, was the first to bring up a model with bi-grams and Good Turing re-estimation to smooth the document models. Latter, Miller et al. (1999) used Hidden Markov Model (HMM) for ranking, which also included the use of bigrams.

HMM, firstly introduced by Rabiner and Juain (1986) in 1986, has been successfully applied into many domains, such as named entity recognition (D. M. Bikel et al., 1997), topic classification (R. Schwartz et al., 1997), or speech recognition (J. Makhoul and R. Schwartz, 1995). In practice, the model requires solving three basic problems. Given the parameters of the model, computing the probability of a particular output sequence is the first problem. This process is often referred to as decoding. Both Forward and Backward procedure are solutions for this problem. The second problem is finding the most possible state sequence with the parameters of the model and a particular output sequence. This is usually completed with Viterbi algorithm. The third problem is the learning problem of HMM models. It is often solved by Baum-Welch algorithm (L. E. Bmjm et al., 1970). Given training

Proceedings of the ACL-IJCNLP 2009 Student Research Workshop, pages 63–71,
Suntec, Singapore, 4 August 2009. ©2009 ACL and AFNLP

data, the algorithm computes the maximum likelihood estimates and posterior mode estimate. It is in essence a generalized Expectation Maximization (EM) algorithm which was first explained and given name by Dempster, Laird and Rubin (1977) in 1977. EM can estimate the maximum likelihood of parameters in probabilistic models which has unseen variables. Nonetheless, in our knowledge, the EM procedure in HMM has never been used in IR domain.

In this paper, we proposed a new language model approach which models the user query and documents as HMM models. We then used EM algorithm to maximize the probability of query words in our model. Our assumption is that if the word's probability in a document is maximized, we can estimate the probability of generating the query word from documents more confidently. Because they not only been calculated by language modeling view features, but also been maximized with statistical methods. Therefore the imprecise cases caused by special distribution in language modeling approach can be further prevented in this way.

The remainders of this paper are organized as follows. We review two related works in Section 2. In Section 3, we introduce our EM IR approach. Section 4 compares our results to two other approaches proposed by Song and Corft (1999) and Robertson (1995) based on the data from TREC HARD track (J. Allan, 2005). Section 5 discusses the effectiveness of our EM training and the EM-based document weighting we proposed. Finally, we conclude our paper in Section 6 and provide some future directions at Section 7.

2 Related Works

Even if we only focus on the probabilistic approach to IR, it is still impossible to discuss all up-to-date research. Instead we focus on two previous works which have inspired the work reported in this paper: the first is a general language model approach proposed by Song and Croft (1999) and the second is a HMM approach by Miller et al. (1999).

2.1 A General Language Model for IR

In 1999, Song and Croft (1999) introduced a language model based on a range of data smoothing technique. The following are some of the features they used:

Good-Turing estimate: Since the effect of Good-Turing estimate was verified as one of the best discount methods (C. D. Manning and H. Schutze, 1999), Song and Croft used Good-Turing estimate for allocating proper probability for the missing terms in the documents. The smoothed probability for term t in document d can be obtained with the following formula:

$$P_{GT}(t|d) = \frac{(tf+1)S(N_{tf+1})}{S(N_{tf})N_d}$$

where N_{tf} is the number of terms with frequency tf in a document. N_d is the total number of terms occurred in document d, and a powerful smoothing function $S(N_{tf})$, which is used for calculating the expected value of N_{tf} regardless of the N_{tf} appears in the corpus or not.

Expanding document model: The document model can be viewed as a smaller part of whole corpus. Due to its limited size, there is a large number of missing terms in documents, and can lead to incorrect distributions of known terms. For dealing with the problem, documents can be expanded with the following weighted sum/product approach:

$$P_{sum}(t|d) = \omega \times P_{doc}(t|d) + (1-\omega) \times P_{corpus}(t)$$
$$P_{product}(t|d) = P_{doc}(t|d)^{\omega} \times P_{corpus}(t)^{(1-\omega)}$$

where ω is a weighting parameter between 0 and 1.

Modeling Query as a Sequence of Terms: Treating a query as a set of terms is commonly seen in IR researches. Song and Croft treated queries as a sequence of terms, and obtained the probability of generating the query by multiplying the individual term probabilities.

$$P_{sequence}(Q|d) = \prod_{i=1}^{m} P(t_i|d)$$

where t_1, t_2 …, t_m is the sequence of terms in a query Q.

Combining the Unigram Model with the Bigram Model: This is commonly implemented with interpolation in statistical language modeling:

$$P(t_{i-1}, t_i|d) = \lambda_1 \times P_1(t_i|d) + \lambda_2 \times P_2(t_{i-1}, t_i|d)$$

where λ_1 and λ_2 are two parameters, and $\lambda_1 + \lambda_2 = 1$. Such interpolation can be modeled by HMM, and can learn the appropriate value from the corpus through EM procedure. A similar procedure is described in Hiemstra and Vries (2000).

2.2 A HMM Information Retrieval System

Miller et al. demonstrated an IR system based on HMM. With a query *Q*, Miller et al. tried to rank the documents according to the probability that *D* is relevant (R) with it, which can be written as *P*(*D* is *R*|*Q*). With Baye's rule, the core formula of their approach is:

$$P(D \text{ is } R|Q) = \frac{P(Q|D \text{ is } R) \cdot P(D \text{ is } R)}{P(Q)}$$

where *P*(*Q*|*D* is *R*) is the probability of query *Q* being posed by a relevant document *D*; *P*(*D* is *R*) is the prior probability that *D* is relevant; *P*(*Q*) is the prior probability of *Q*. Because *P*(*Q*) will be identical, and the *P*(*D* is *R*) is assumed to be constant across all documents, they place their focus on *P*(*Q*|*D* is *R*).

To figure out the value of *P*(*Q*|*D* is *R*), they established a HMM. The union of all words appearing in the corpus is taken as the observation, and each different mechanism of query word generation represent a state. So the observation probability from different states is according to the output distribution of the state.

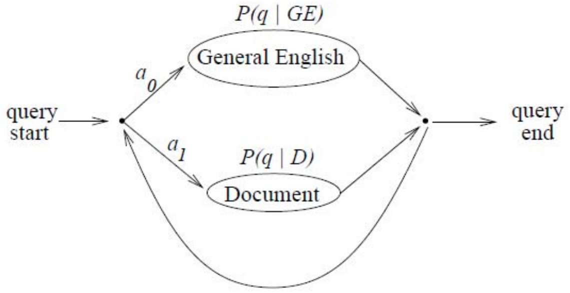

Figure 1. HMM proposed in "A Hidden Markov Model Information Retrieval System"

To estimate the transition and observation probabilities of HMM, EM algorithm is the standard method for parameter estimation. However, due to some difficulty, they make two practical simplifications. First, they assume the transition probabilities are same for all documents, since they establish an individual HMM for each document. Second, they completely abandon the EM algorithm for the estimation of observation probabilities. Instead, they use simple maximum likelihood estimates for each documents. So the probabilities which their HMM generate term q from their HMM states become:

$$\mathrm{P(q|D_k)} = \frac{\text{number of times q appears in } \mathrm{D_k}}{\text{length of } \mathrm{D_k}}$$

$$\mathrm{P(q|GE)} = \frac{\sum_k \text{number of times q appears in } \mathrm{D_k}}{\sum_k \text{length of } \mathrm{D_k}}$$

with these estimated parameters, they state the formula for P(Q|D is R) corresponding to Figure 1 as:

$$\mathrm{P(Q|D_k \text{ is } R)} = \prod_{q \in Q} (a_0 \mathrm{P(q|GE)} + a_1 \mathrm{P(q|D_k)})$$

the probabilities obtained through this formula is then used for calculating the P(D is R|Q). The document is then ranked according to the value of P(D is R|Q).

The HMM model we proposed is far different from Miller et al. (1999). They build HMM for every document, and treat all words in the document as one state's observation, and word that is unrelated to the document, but occurs commonly in natural language queries as another state's observation. Hence, their approach requires information about the words which appears commonly in natural language. The content of the provided information will also affect the IR result, hence it is unstable. We assume that every document is an individual state, and the probabilities of query words generated by this document as the observation probabilities. Our HMM model is built on the corpus we used and does not need further information. This will make our IR result fit on our corpus and not affected by outside information. It will be detailed introduced at Section 3.

3 Our EM IR approach

We formulate the IR problem as follows: given a query string and a set of documents, we rank the documents according to the probability of each document for generating the query terms. Since the EM procedure is very sensitive to the number of states, while a large number of states take much time for one run, we firstly apply a basic language modeling method to reduce our document set. This language modeling method will be detailed at Section 3.1. Based on the reduced document set, we then describe how to build our HMM model, and demonstrate how to obtain the special-designed observance sequence for our HMM training in Section 3.2 and 3.3, respectively. Finally, Section 3.4 introduces the evaluation mechanism to the probability of generating the query for each document.

3.1 The basic language modeling method for document reduction

Suppose we have a huge document set D, and a query Q, we firstly reduce the document set to obtain the document D_r. We require the reducing method can be efficiently computed, therefore two methods proposed by Song and Croft (1999) are consulted with some modifications: Good-Turing estimation and modeling query as a sequence of terms.

In our modified Good-Turing estimation, we gathered the number of terms to calculate the term frequency (*tf*) information in our document set. Table 1 shows the term distribution of the AQUAINT corpus which is used in the TREC 2005 HARD Track (J. Allan, 2005). The detail of the dataset is described in Section 4.1.

tf	N_{tf}	*tf*	N_{tf}
0	1,140,854,966,460	5	3,327,633
1	166,056,563	6	2,163,538
2	29,905,324	7	1,491,244
3	11,191,786	8	1,089,490
4	5,668,929	9	819,517

Table 1. Term distribution in AQUAINT corpus

In this table, N_{tf} is the number of terms with frequency *tf* in a document. The tf = 0 case in the table means the number of words not appear in a document. If the number of all word in our corpus is W, and the number of word in a document d is w_d, then for each document, the tf = 0 will add $W - w_d$. By listing all frequency in our document set, we adapt the formula defined in (Song and Croft, 1999) as follows:

$$P_{mGT}(t|d) = \frac{(tf+1)N_{tf+1}}{N_{tf}N_d}$$

In our formula, the N_d means the number of word tokens in the document d. Moreover, the smoothing function is replaced with accurate frequency information, N_{tf} and N_{tf+1}. Obviously, there could be two problems in our method: First, while in high frequency, there might be some missing N_{tf+1}, because not all frequency is continuously appear. Second, the N_{tf+1} for the highest *tf* is zero, this will lead to its P_{mGT} become zero. Therefore, we make an assumption to solve these problems: If the N_{tf+1} is missing, then its value is the same as N_{tf}. According to Table 1, we can find out that the difference between *tf* and *tf*+1 is decreasing when the *tf* becomes higher. So we assume the difference becomes zero when we faced the missing frequency at a high number. This assumption can help us ensure the completeness of our frequency distribution.

Aside from our Good-Turing estimation design, we also treat query as a sequence of terms. There are two reasons to make us made this decision. By doing so, we will be able to handle the duplicate terms in the query. Furthermore, it will enable us to model query phrase with local contexts. So our document score with this basic method can be calculated by multiplying $P_{mGT}(q|d)$ for every q in Q. We can obtain D_r with the top 50 scores in this scoring method.

3.2 HMM model for EM IR

Once we have the reduced document set D_r, we can start to establish our HMM model for EM IR. This HMM is designed to use the EM procedure to modify its parameters, and its original parameters are given by the basic language modeling approach calculation.

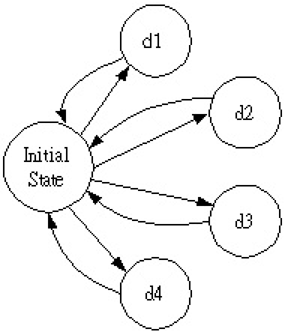

Figure 2. HMM model for EM IR

We define our HMM model as a four-tuple, $\{\mathbf{S},\mathbf{A},\mathbf{B},\boldsymbol{\pi}\}$, where $\mathbf{S}$ is a set of N states, $\mathbf{A}$ is a $N \times N$ matrix of state transition probabilities, $\mathbf{B}$ is a set of N probability functions, each describing the observation probability with respect to a state and $\boldsymbol{\pi}$ is the vector of the initial state probabilities.

In our HMM model, it composes of $|D_r|+1$ states. Every document in the document set is treated as an individual state in our HMM model. Aside from these document states, we add a special state called "Initial State". This state is the only one not associate with any document in our document sets. Figure 2 illustrates the proposed HMM IR model.

The transition probabilities in our HMM can be classified into two types. For the "Initial State", the transition to the other state can be regard as the probability of choosing that document. We assume that every document has the same probability to be chosen at the beginning, so the transition probabilities for "Initial State" are $1/|D_r|$ to every document state. For the document states, their transition probabilities are fixed: 100% to the “Initial State”. Since the transition between documents has no statistical meaning, we make the state transition after the document state back to the Initial State. This design helps us to keep the independency between the query words. We will detail this part at Section 3.3.

The observation probabilities for each state are similar with the concept of language modeling. There are three types of observations in our HMM model.

Firstly, for every document, we can obtain the observation probability for each query term according to our basic language modeling method. Even if the query term is not in the document, it will be assigned a small value according to the method described in Section 3.1.

Secondly, for the terms in a document, which is not part of our query terms, are treating as another observation. Since we mainly focus on the probability of generating the query terms from the documents, the rest terms are treated as the same type which means “not the query term”.

The last type of observation is a special imposed token “$” which has 100% observation probability at the Initial State.

Figure 3 shows a complete built HMM model for EM IR. The transition probability from Initial State is labeled with trans(d_n), and the observation probability in the document state and Initial State is showed with “ob”. The “N” symbol represents the “not the query term”. Summing all the token mentioned above, all possible observations for our HMM model are $|Q|$+2. The possible observation for each state is bolded, so we can see the difference between Initial State and Document State.

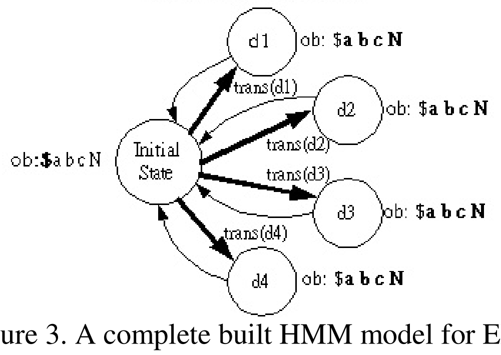

Figure 3. A complete built HMM model for EM IR with parameters

For Initial State, the observations are fixed with 100% for $ token. This special token help we ensure the independency between the query terms. The effect of this token will be discussed in Section 3.3. For the document states, the probabilities for the query terms are calculated with the simple language modeling approach. Even if the query term is not in the document, it will be assigned a small value according to the basic language modeling method. The rest of the terms in a document are treating as another kind of observation, which is the “N” symbol in the Figure 3. Since we mainly focus on the probability of generating the query terms from the documents, the rest of the words are treated as the same kind which means “not the query term”. Additionally, each document state represents a document, so the $ token will never been observed in them.

3.3 The observance sequence and HMM training procedure

After establishing the HMM model, the observation sequence is another necessary part for our HMM training procedure. The observation sequence used in HMM training means the trend for the observation while running HMM. In our approach, since we want to find out the document which is more related with our query, so we use the query terms as our observation sequence. During the state transition with query, we can maximize the probability for each document to generate our query. This will help us figure out which document is more related with our query.

Due to the state transitions in the proposed HMM model are required to go back to the Initial State after transiting to the document state, generating the pure query terms observation sequence is impossible, because the Initial State won’t produce any query term. Therefore, we add the $ token into our observation sequence before each query terms. For instance, if we are running a HMM training with query “a b c“, the exact observation sequence for our HMM training becomes “$ a $ b $ c”. Additionally, each document state represents a document, so the $ token will never been observed in them. By tuning our HMM model with the data from our query instead of other validation data, we can focus on the document we want more precisely.

The reason why we use this special setting for EM training procedure is because we are trying to maintain the independency assumption for query terms in HMM. The HMM observance sequence not only shows the trend of this model’s observation, but also indicate the dependency between these observations. However, the independency between all query terms is a common assumption for IR system (F. Song and W. B. Croft, 1999; V. Lavrenko and W. B. Croft, 2001; A. Berger and J. Lafferty, 1999). To ensure this assumption still works in our HMM system, we use the Initial State to separate each transition to the document state and observe the query terms. No matter the early or late the query term *t* occurs, the training procedure is fixed as “Starting from the Initial state and observed $, transit to a document state, and observe *t*”. We’ve made experiments to verify the independency assumption still work, and the result remains the same no matter how we change the order of our query terms.

After constructing the HMM model and the observance sequence, we can start our EM training procedure. EM algorithm is used for finding maximum likelihood estimates of parameters in probabilistic models, where the model depends on unobserved latent variables. In our experiment, we use EM algorithm to find the parameters of our HMM model. These parameters will be used for information retrieval. The detail implementation information can be found in (C. D. Manning and H. Schutze, 1999), which introduce HMM and the training procedure very well.

3.4 Scoring the documents with EM-trained HMM model

When the training procedure is completed, each document will have new parameters for the word’s observation probability. Moreover, the transition probabilities from Initial State to the document state are no longer uniform due to the EM training. So the probability for a document *d* to generate the query *Q* becomes:

$$P(Q|d) = \text{trans}(d) * \prod_{q \in Q} P(q|d)$$

In this formula, the trans(*d*) means the transition probability from the Initial State to the document state of *d*, which we called “EM-based document weighting”. The *P*(*q*|*d*) means the observation probability for query term *q* in document state of *d*, which is also tuned in our EM training procedure. With this formula, we can rank the IR result according to this probability. This performs better than the GLM when the document size is relatively small, since GLM gives those documents as with too high score.

4 Experiment Results

4.1 Data Set

We use the AQUAINT corpus as our training data set. It is used in the TREC 2005 HARD Track (J. Allan, 2005). The AQUAINT corpus is prepared by the LDC for the AQUAINT Project, and is used in official benchmark evaluations conducted by National Institute of Standards and Technology (NIST). It contains news from three sources: the Xinhua News Service (People's Republic of China), the New York Times News Service, and the Associated Press Worldstream News Service.

The topics we used are the same as the TREC Robust track (E. M. Voorhees, 2005), which are the topics from number 303 to number 689 of the TREC topics. Each topic is described in three formats including titles, descriptions and narratives. In our experiment, due to the fact that our observation sequence is very sensitive to the query terms, we only focus on the title part of the topic. In this way, we can avoid some commonly appeared words in narratives or descriptions, which may reduce the precision of our training procedure for finding the real document. Table 2 shows the detail about the corpus.

Datasize	2.96GB
#Documents	1,030,561
#Querys	50
Term Types	2,002,165
Term Tokens	431,823,255

Table 2. Statistics of the AQUAINT corpus

4.2 Experiment Design and Results

By using the AQUAINT corpus, two different traditional IR methods are implemented for comparing. The two IR methods which we use as baselines are the General Language Modeling (GLM) proposed by Song and Croft (1999) and the tf.idf measure proposed by Robertson (1995). The GLM has been introduced in Section 2. The following formulas show the core of tf.idf:

$$\text{tf.idf}(Q, D) = \sum_{q_i \in Q} \text{wtf}(q_i, D) \cdot \text{idf}(q_i)$$

$$\text{wtf}(q, D) = \frac{\text{tf}(q, D)}{\text{tf}(q, D) + 0.5 + 1.5\frac{l(D)}{al}}$$

$$\text{idf}(q) = \frac{\log\frac{N}{n_q}}{N + 1}$$

N is the number of documents in the corpus; n_q is the number of documents in the corpus containing *q*; tf(*q*, *D*) is the number of times *q* appears in *D*; *l*(*D*) is the length of *D* in words and the *al* is the average length in words of a *D* in the corpus.

For the proposed EM IR approach, two configurations are listed to compare. The first (Config.1) is the proposed HMM model without making use of the EM-based document weighting that is don't multiply the transition probability, trans(*d*), in equation (2). The second (Config.2) is the HMM model with EM-based document weighting. The comparison is based on precision. For each problem, we retrieved the documents with the highest 20 scores, and divided the number of correct answer with the number of retrieved document to obtain precision. If there are documents with same score at the rank of 20, all of them will be retrieved.

Methods	Precision	%Change	%Change
tf.idf	29.7%	-	
GLM	30.5%	2.69%	-
Config.1	28.8%	-5.58%	-3.14%
Config.2	32.2%	8.41%	5.57%

Table 3. Experiment Results of three IR methods on the AQUAINT corpus

As shown in Table 3, our EM IR system outperforms tf.idf method 8.41% and GLM method 5.57%.

5 Discussion

In this section, we will discuss the effectiveness of the EM-based document weighting and the EM procedure. Both of them rely on the HMM design we have proposed.

5.1 The effectiveness of EM-based document weighting

When we establish our HMM model, the transition probability from Initial State to the document state is assigned as uniform, since we don't have any information about the importance of every document. These transition probabilities represent the probability of choosing the document with the given observation sequence.

During EM training procedure, the transition probability, exclusive the transition probability from document states which is fixed to 100% to the Initial State, will be re-estimated according to the observation sequence (the query) and the observation probabilities of each state. As shown in Table 3, two configurations (Config.1 and Config.2) are conducted to verify the effectiveness of using the transition probability.

The transition probability works due to the EM training procedure. The training procedure works for maximizing the probability for generating the query words, so the weight for each document will be given according to mathematical formula. The advantage of this mechanism is it will use the same formula regardless of different content of document. Yet other statistical methods will have to fix the content or formula previously to avoid the noise or other disturbance. Some researches employee the number of terms in the document to calculate the document weighting. Since the observation probability already use the number of words in a document N_d as a parameter, using number of words as document weight will make it affect too much in our system.

The experiment results show an improvement of 11.80% by using the transition probability of Initial State. Accordingly, we can understand that the EM procedure helps our HMM model not only on the observation probability of generating query words, but also suggests a useful weight for each document.

5.2 The effectiveness of EM training

In HMM model training, the iteration numbers of EM procedure is always a tricky issue for experiment design. While training with too much iteration will lead to overfitting for the observation sequence, to less iteration will weaken the effect of EM training.

For our EM IR system, we've made a series of experiments with different iterations for examining the effect of EM training. Figure 3 shows the results.

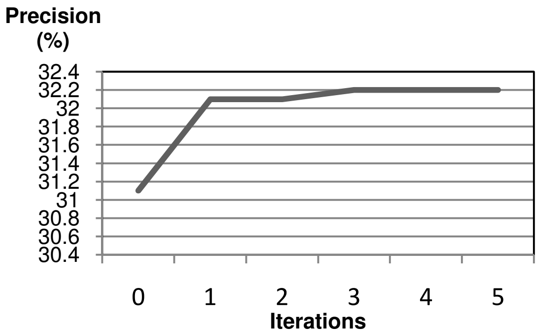

Figure 4. The precision change with the EM training iterations

As you can see in Figure 4, the precision increased with the iteration numbers. Still, the growing rate of precision becomes very slow after 2 iterations. We have analysis this result and find out two possible causes for this evidence. First, the training document sets are limited in a small size due to the computation time complexity for our approach. Therefore we can only retrieve correct document with high score in basic language modeling, which is used for document reduction. So the precision is also limited with the performance of our reducing methods. The number of correct answer is limited by the basic language modeling, so as the highest precision our system can achieve. Second, our observation only composed query terms, which gives a limited improving space.

6 Conclusion

We have proposed a method for using EM algorithm to improve the precision in information retrieval. This method employees the concept of language model approach, and merge it with the HMM. The transition probability in HMM is treated as the probability of choosing the document, and the observation probability in HMM is treated as the probability of generating the terms for the document. We also implement this method, and compare it with two existing IR methods with the dataset from TREC 2005 HARD Track. The experiment results show that the proposed approach outperforms two existing methods by 2.4% and 1.6% in precision, which are 8.08% and 5.24% increasing for the existing method. The effectiveness of using the tuned transition probability and EM training procedure is also discussed, and been proved can work effectively.

7 Future Work

Since we have achieved such improvement with EM algorithm, other kinds of algorithm with similar functions can also be tried in IR system. It might be work in the form of parameter reestimation, tuning or even generating parameters by statistical measure.

For the method we have proposed, we also have some part can be done in the future. Finding a better observance sequence will be an important issue. Since we use the exact query terms as our observance sequence, it's possible to use the method like statistical translation to generate more words which are also related with the documents we want and used as observance sequence.

Another possible issue is to integrate the bigram or trigram information into our training procedure. Corpus information might be used in more delicate way to improve the performance.

References

A. Berger and J. Lafferty, "Information retrieval as statistical translation," 1999, pp. 222-229.

A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society,* vol. 39, pp. 1-38, 1977.

C. D. Manning and H. Schutze, *Foundations of statistical natural language processing*: MIT Press, 1999.

D. Hiemstra and A. P. de Vries, *Relating the new language models of information retrieval to the traditional retrieval models*: University of Twente [Host]; University of Twente, Centre for Telematics and Information Technology, 2000.

D. M. Bikel, S. Miller, R. Schwartz, and R. Weischedel, "Nymble: a high-performance learning name-finder," 1997, pp. 194-201.

D. R. H. Miller, T. Leek, and R. M. Schwartz, "A hidden Markov model information retrieval system," 1999, pp. 214-221.

E. M. Voorhees, "The TREC robust retrieval track," 2005, pp. 11-20.

F. Song and W. B. Croft, "A general language model for information retrieval," 1999, pp. 316-321.

G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*: McGraw-Hill, Inc. New York, NY, USA, 1986.

J. Allan, "HARD track overview in TREC 2005: High accuracy retrieval from documents," 2005.

J. Makhoul and R. Schwartz, "State of the Art in Continuous Speech Recognition," *Proceedings of the National Academy of Sciences,* vol. 92, pp. 9956-9963, 1995.

J. M. Ponte and W. B. Croft, "A language modeling approach to information retrieval," 1998, pp. 275-281.

L. E. Bmjm, T. Petrie, G. Soules, and N. Weiss, "A MAXIMIZATION TECHNIQUE OCCURRING IN THE STATISTICAL ANALYSIS OF PROBABILISTIC FUNCTIONS OF MARKOV CHAINS," *The Annals of Mathematical Statistics,* vol. 41, pp. 164-171, 1970.

L. Rabiner and B. Juang, "An introduction to hidden Markov models," *ASSP Magazine, IEEE [see also IEEE Signal Processing Magazine],* vol. 3, pp. 4-16, 1986.

R. Schwartz, T. Imai, F. Kubala, L. Nguyen, and J. Makhoul, "A Maximum Likelihood Model for Topic Classification of Broadcast News," 1997.

S. E. Robertson, "The probability ranking principle in IR," *Journal of Documentation,* vol. 33, pp. 294-304, 1977.

S. E. Robertson and S. Jones, "Relevance Weighting of Search Terms," *Journal of the American Society for Information Science,* vol. 27, pp. 129-46, 1976.

S. E. Robertson and S. Walker, "Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval," 1994, pp. 232-241.

S. E. Robertson, S. Walker, and S. Jones, "M. Hancock-Beaulieu, M., and Gatford, M.(1995). Okapi at TREC-3," pp. 109–126.

V. Bush, "As we may think," *interactions,* vol. 3, pp. 35-46, 1996.

V. Lavrenko and W. B. Croft, "Relevance based language models," 2001, pp. 120-127.