

Fast Exact Maximum Likelihood Estimation for Mixture of Language Models

Yi Zhang
School of Engineering
University of California Santa Cruz
Santa Cruz, CA, USA
yiz@soe.ucsc.edu

Wei Xu
NEC Lab America
10080 North Wolfe Road, Suite SW3-350
Cupertino, CA, United States
xw@sv.nec-labs.com

ABSTRACT

A common language modeling approach assumes the data D is generated from a mixture of several language models. EM algorithm is usually used to find the maximum likelihood estimation of one unknown mixture component, given the mixture weights and the other language models. In this paper, we provide an efficient algorithm of $O(k)$ complexity to find the exact solution, where k is the number of words occurred at least once in D . Another merit is that the probabilities of many words are exactly zeros, which means that the mixture language model also serves as a feature selection technique.

Categories and Subject Descriptors: B.3.3 [Information Search and Retrieval]: Information Search and Retrieval

General Terms: Algorithms, Languages

Keywords: language models

1. INTRODUCTION

Mixture of language models is commonly used in IR application. Assume a sequence of data D is generated from a language model r , which is a linear combination of two multinomial language models p and q :

$$r = \alpha p + \beta q \quad (1)$$

where α and β are interpolation weights that sum to 1.

The problem is to find q_i 's that maximize the likelihood of observed data D , for given f_i , p_i , α and β , subject to the constraints $\sum_i q_i = 1$ and $q_i \geq 0$. f_i is the observed frequency of word i in D .

The log-likelihood of the data is:

$$LL = \sum_i f_i \log(r_i) = \sum_i f_i \log(\alpha p_i + \beta q_i) \quad (2)$$

This modeling approach has been applied to various information retrieval tasks such as relevance feedback [4] [2] [1], context sensitive language model smoothing [6], and novelty detection [5]. For example, p is the corpus language model, r is the language model used to generate document related to a query, and q is a query language model to be learned from relevance feedback.

We can treat the maximum likelihood estimator (MLE) of p as already known, usually it is calculated directly as:

$$\hat{p}_i = \hat{P}(w_i|p) = \frac{df_i}{\sum_j df_j}$$

where df_i is the number of times word i occurs in the collection.

The major task is to find the MLE of q , which is usually obtained using the EM algorithm. Despite its popularity and simplicity, EM has two weaknesses. First, it is a computationally expensive greedy search algorithm. This expense discourages the use of the language modeling approach for the ad-hoc retrieval task in situations where computational efficiency is important, for example, in a large scale web search engine. Second, EM only provides an approximately optimal result with increased computational complexity, which we will discuss later.

2. EXACT SOLUTION

An exact solution for q and an $O(k \log(k))$ algorithm were presented in [3]. We briefly revisit the exact solution in this section, since the work is not commonly known for the IR community and it is the basis of our new algorithm.

For all the q_i such that $q_i > 0$, apply Lagrange multiplier method and calculate the derivatives with respect to q_i :

$$\begin{aligned} \frac{\partial}{\partial q_i} \left(LL - \lambda \left(\sum_i q_i - 1 \right) \right) &= \frac{f_i \beta}{\alpha p_i + \beta q_i} - \lambda = \alpha(3) \\ \Rightarrow q_i &= \frac{f_i}{\lambda} - \frac{\alpha}{\beta} p_i \end{aligned} \quad (4)$$

Applying the constraint that $\sum_i q_i = 1$ to all the $q_i > 0$:

$$\sum_i \left(\frac{f_i}{\lambda} - \frac{\alpha}{\beta} p_i \right) = 1 \Rightarrow \lambda = \frac{\sum_{i:q_i>0} f_i}{1 + \frac{\alpha}{\beta} \sum_{i:q_i>0} p_i} \quad (5)$$

We can prove by contradiction that if $q_i : i = 1 \dots k$ maximize the objective function (2), $q_2 > 0$ and $\frac{p_1}{f_1} \leq \frac{p_2}{f_2}$, then $q_1 > 0$. We refer the readers to [3] for more details. A direct consequence is that all the q_i 's greater than 0 correspond to the smallest $\frac{p_i}{f_i}$. So we can use the following algorithm to find the exact q that maximize the likelihood of observed data:

Algorithm 0: Sort $\frac{p_i}{f_i}$ so that $\frac{f_1}{p_1} > \frac{f_2}{p_2} \geq \dots \geq \frac{f_k}{p_k}$, find t such that

$$\frac{\beta}{\alpha} + \sum_{j=1}^t \frac{p_j}{f_j} - \frac{p_t}{f_t} > 0 \quad (6)$$

and

$$\frac{\beta}{\alpha} + \frac{\sum_{j=1}^{t+1} p_j}{\sum_{j=1}^{t+1} f_j} - \frac{p_{t+1}}{f_{t+1}} \leq 0 \quad (7)$$

Then q_i 's are given by:

$$q_i = \begin{cases} \frac{f_i}{\lambda} - \frac{\alpha}{\beta} p_i & \text{if } \frac{f_i}{p_i} \geq \frac{f_t}{p_t} \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

Where λ is given by:

$$\lambda = \frac{\sum_{i=1}^t f_i}{1 + \frac{\alpha}{\beta} \sum_{i=1}^t p_i} \quad (9)$$

3. NEW LINEAR ALGORITHM

The key part of the Algorithm 0 is to find the thresholding pair (f_t, p_t) . Borrowing ideas from the $O(k)$ algorithm for finding median, we have the following average $O(k)$ algorithm for finding (f_t, p_t) .

Algorithm 1 Find t

```

1:  $S_p \leftarrow 0, S_f \leftarrow 0$ 
2:  $A \leftarrow \{1, 2, \dots, k\}$ 
3: repeat
4:   select a pivot  $s$  from  $A$ 
5:    $L \leftarrow \emptyset, G \leftarrow \emptyset, S_f^* \leftarrow S_f, S_p^* \leftarrow S_p$ 
6:   for all  $i \in A$  do
7:     if  $\frac{f_i}{p_i} > \frac{f_s}{p_s}$  then
8:        $G \leftarrow G \cup \{i\}, S_f^* \leftarrow S_f^* + f_i, S_p^* \leftarrow S_p^* + p_i,$ 
9:     else if  $\frac{f_i}{p_i} < \frac{f_s}{p_s}$  then
10:       $L \leftarrow L \cup \{i\}$ 
11:     else
12:       $S_f^* \leftarrow S_f^* + f_i, S_p^* \leftarrow S_p^* + p_i$ 
13:     end if
14:   end for
15:   if  $\frac{\beta}{\alpha} \frac{S_f^*}{S_p^*} - \frac{p_s}{f_s} > 0$  then
16:      $A \leftarrow L, S_p \leftarrow S_p^*, S_f \leftarrow S_f^*, t \leftarrow s$ 
17:   else
18:      $A \leftarrow G$ 
19:   end if
20: until  $A = \emptyset$ 

```

The general idea of Algorithm 1 is to first select a random term/element/word s (Line 4) as a pivot to partition the words into two sets L and G (Line 7 and Line 9). Then we decide whether t is in set L or G (Line 15). Then we recurse on the appropriate subset (L or G) until we find t (Line 20). On average, it takes $O(\log(k))$ iterations, and the cost of each iteration being roughly half of the previous one. The whole process is a geometric series converges as an average linear time algorithm.

Now we prove that the average computation cost is $O(k)$ using recursion. Let C_k be the average computation cost for n , then we have the following recursion:

$$C_k = k + \frac{1}{k}(C_1 + \dots + C_{k-1}) \quad (10)$$

From (10), it can be derived that

$$C_k = 2k - \left(\frac{1}{1} + \frac{1}{2} + \dots + \frac{1}{k}\right) \quad (11)$$

Hence the average computation cost is $O(k)$. k is the number of words with none zero frequency in D . Because

words not in data D do not influence the estimation of λ and are ranked lower than the pivot, we can ignore them in Algorithm 1. This property is nice, because k could be much smaller than the vocabulary size. For example, if D is a document generated by the mixture of document language model q and background English language model p , we only need to process words in the document while estimating the document language model p using Algorithm 1. If we use EM, all words in the whole vocabulary need to be processed.

The algorithm can also be modified to a worst case $O(k)$ algorithm. This can be achieved by using s corresponding to the median of $\frac{f_i}{p_i}$ at Line 4 instead of randomly selecting a s .

If implemented appropriately, the division operations can be mostly avoided in this algorithm. For example, Line 7 can be implemented as *if* $f_i p_s > f_s p_i$. As we know, division is a much more time consuming floating point operation than multiplication. Our simulations show that EM algorithm converges to the results of our exact algorithm, while our algorithm takes about the same time as 1 to 3 EM iterations to finish. Needless to say that EM algorithm needs many iterations to converge [3]. The source code of our algorithm written in C is provide online at www.soe.ucsc.edu/~yiz/papers/sigir07poster.

Another merit of our algorithm is that we know the probabilities of some words in language model q are exactly zero (Equation (8)), which means that the mixture language model also serves as a feature selection technique.

4. REFERENCES

- [1] D. Hiemstra, S. Robertson, and H. Zaragoza. Parsimonious language models for information retrieval. In *SIGIR '04*, pages 178–185, New York, NY, USA, 2004. ACM Press.
- [2] W. Kraaij, R. Pohlmann, and D. Hiemstra. Twenty-one at trec-8: using language technology for information retrieval. In *TREC 8: Proceedings of the 1999 Text REtrieval Conference. National Institute of Standards and Technology, special publication*.
- [3] W. X. Y. Zhang and J. Callan. Exact maximum likelihood estimation for word mixtures. In *Text Learning Workshop in International Conference on Machine Learning, Sydney, Australia, 2002*.
- [4] C. Zhai and J. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the Tenth International Conference on Information and Knowledge Management, 2001*.
- [5] Y. Zhang, J. Callan, and T. Minka. Novelty and redundancy detection in adaptive filtering. In *SIGIR: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval, 2002*.
- [6] X. Zhou, X. Hu, X. Zhang, X. Lin, and I.-Y. Song. Context-sensitive semantic smoothing for the language modeling approach to genomic ir. In *SIGIR: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, 2006*.