Dependency Structure Language Model for Information Retrieval

Changki Lee, Gary Geunbae Lee

Department of Computer Science and Engineering,
Pohang University of Science and Technology,
San 31 Hyoja dong, Nam Gu, POHANG, 790-784, KOREA
Phone: +82-54-279-5581, Fax: +82-54-279-2299,
{leeck, gblee}@postech.ac.kr

Abstract

– In this paper, we propose a new language model, namely, a dependency structure language model, for information retrieval to compensate for the weakness of bigram and trigram language models. The dependency structure language model is based on the Chow Expansion theory and the dependency parse tree generated by a dependency parser. So, long-distance dependencies can be handled by the dependency structure language model. We carried out some experiments to verify the proposed model. In our experiments, the dependency structure model gives a better performance than recently proposed language models, and the dependency structure is more effective than bigram in language modeling for information retrieval.

1. INTRODUCTION

Using language models for information retrieval has recently been studied extensively (Ponte and Croft, 1998; Miller, Leek and Schwartz, 1999; Song and Croft, 1999; Zhai and Lafferty, 2001). The basic idea is to compute the conditional probability $p(q|d)$, i.e. the probability of generating a query q given the observation of a document d. Several different methods have been applied to compute this conditional probability.

Ponte and Croft (1998) used several heuristics to smooth the Maximum Likelihood Estimate (MLE) of the document language model, and assumed that the query is generated under a multivariate Bernoulli model. The BBN method uses a two-state hidden Markov model as the basis for generating queries, which, in effect, is to smooth the MLE with linear interpolation (Miller, Leek and Schwartz, 1999). In Zhai and Lafferty (2001), it has been found that the retrieval performance is affected by both the estimation accuracy of document language models and the appropriate modeling of the query.

The language models used in most previous works are the unigram models (Similarly, Ponte and Croft (1998) assume that, given a particular language model, the query terms occur independently). The unigram language model makes a strong assumption that each word occurs independently, and consequently, the probability of a word sequence becomes the product of the probabilities of the individual words.

There are some explorations of bigram and trigram models to improve this unrealistic assumption by considering the local context (Miller, Leek and Schwartz, 1999; Song and Croft, 1999). For a bigram, the probability of a new word depends on the previous word, while for a trigram, the probability of a new word depends on the probabilities of the previous two words.

However, bigram and trigram models have a limitation in handling long-distance dependences (In the speech literature, the term ‘long-distance dependencies’ regularly refers to anything beyond the range of a trigram model). In a sentence like:

“Which book should Peter read?”

We can recognize a long distance dependency between *read* and *book*.

Recently, there are language models based on syntactic parsing to overcome the limitation of bigram/trigram in speech recognition field (Charniak, 2001; Chelba and Jelinek, 1998; Roark, 2001).

In this paper, which addresses similar concepts, we propose a dependency structure language model to overcome the limitation of bigram and trigram models in information retrieval. The dependency structure language model is based on a dependency parse tree generated by linguistic parser. So, long-distance dependencies can be naturally handled by the linguistic syntactic structure language model. Our dependency structure language model adopts Chow Expansion theory (Chow and Liu, 1968; Duda and Hart, 1973) and dependency parse tree to handle long-distance dependencies. Chow Expansion theory has been discussed in IR more than 20 years ago (van Rijsbergen, 1977).

The remainder of this paper is organized as follows. In section 2, we describe the Chow Expansion theory and the dependency parse tree, and in section 3, we describe the dependency structure language model. In section 4, we present some experiments and the results. Section 5 gives the conclusion and future work.

2. CHOW EXPANSION THEORY AND DEPENDENCY PARSE TREE

2.1 Chow Expansion

Consider a query $q=q_1q_2...q_n$ and a corresponding vector $x=\{x_1,x_2,...,x_n\}$, where $x_i=1$ if q_i appears in a document, $x_i=0$ otherwise. The problem of estimating a density becomes the problem of estimating the probability $p(x)$. Since there are 2^n possible vectors x, we must estimate 2^n probabilities, which is an enormous task.

If the components of x are statistically independent, the problem is greatly simplified. In this case we can write

$$p(x) = \prod_{i=1}^{n} p(x_i) = \prod_{i=1}^{n} p_i^{x_i} (1 - p_i)^{1-x_i}$$

where $p_i = p(x_i=1)$ and $1-p_i = p(x_i=0)$.

It is natural to ask whether or not there are any compromise positions between being completely accurate, which requires estimating 2^n probabilities, and being forced to assume statistical independence, which reduces the problem to estimating only n probabilities. One answer is provided by finding an expansion for $p(x)$ and approximating $p(x)$ by a partial sum, e.g. the Rademacher-Walsh Expansion and the Bahadur-Lazarsfeld Expansion (Duda and Hart, 1973).

Another interesting class of approximation to a joint probability distribution $p(x)$ is based on the identity

$$\begin{aligned} p(x) &= p(x_1, x_2, \cdots, x_n) \\ &= p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_2, x_1)\cdots p(x_n \mid x_{n-1}, \cdots, x_1). \end{aligned} \tag{1}$$

Suppose the variables are not independent, but we can number the variables so that $p(x_i|x_{i-1}, ...,x_1)$ is solely dependent on some preceding variable $x_{j(i)}$. For example, suppose that

$$p(x_5 \mid x_4, x_3, x_2, x_1) = p(x_5 \mid x_{j(5)}) = p(x_5 \mid x_2)$$
$$p(x_4 \mid x_3, x_2, x_1) = p(x_4 \mid x_{j(4)}) = p(x_4 \mid x_2)$$
$$p(x_3 \mid x_2, x_1) = p(x_3 \mid x_{j(3)}) = p(x_3 \mid x_1)$$
$$p(x_2 \mid x_1) = p(x_2 \mid x_{j(2)}) = p(x_2 \mid x_1)$$

and a corresponding dependence tree as Figure (1).

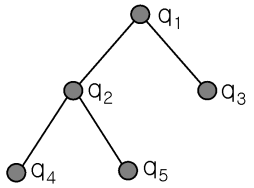

Figure -1. A dependence tree.

Then it follows from Equation (1) that $p(x_1,x_2,x_3,x_4,x_5)$ can be written as $p(x_1)p(x_2|x_1)p(x_3|x_1)p(x_4|x_2)p(x_5|x_2)$. In general, we obtain the product expansion

$$p(x) = p(x_1)p(x_2 \mid x_{j(2)})p(x_3 \mid x_{j(3)})\cdots p(x_n \mid x_{j(n)}) \tag{2}$$

where the function $j(i)$ exhibits the limited dependence of one variable on preceding variables. Thus we can write the probability of x_i given $x_{j(i)}$ as follows:

$$p(x_i \mid x_{j(i)}) = \left[p_{i|j(i)}^{x_i}(1-p_{i|j(i)})^{1-x_i}\right]^{x_{j(i)}}\left[p_i^{x_i}(1-p_i)^{1-x_i}\right]^{1-x_{j(i)}} \tag{3}$$

where $p_{i|j(i)} = p(x_i=1|x_{j(i)}=1)$ and $p_i = p(x_i=1 \mid x_{j(i)}=0)$.

By letting $p_i=p(x_i=1)$, substituting Equation (3) in Equation (2), taking the logarithm, and collecting terms, we obtain the Chow Expansion (Chow and Liu, 1968; Duda and Hart, 1973).

$$\log p(x) = \sum_{i=1}^{n}\log(1-p_i) + \sum_{i=1}^{n}x_i \log\frac{p_i}{1-p_i} + \sum_{i=2}^{n}x_{j(i)}\log\frac{1-p_{i|j(i)}}{1-p_i} + \sum_{i=2}^{n}x_i x_{j(i)}\log\frac{p_{i|j(i)}(1-p_i)}{(1-p_{i|j(i)})p_i} \tag{4}$$

A few observations about these results are in order. First, we note that if the variables are indeed independent, $p_{i|j(i)}=p_i$ and the last two sums in the expansion disappear, leaving the familiar expansion for the independent case. When dependence exists, we obtain additional linear and quadric terms.

2.2 Dependency Parse Tree

A dependency relationship is an asymmetric binary relationship between a word called head (or governor, parent), and another word called modifier (or dependent, daughter) (Hays, 1964). Dependency grammars represent sentence structures as a set of dependency relationships. Normally the dependency relationships from a tree connect all the words in a sentence. A word in the sentence may have several modifiers, but each word may modify at most one word. The root of the dependency tree does not modify any word. It is also called the head of the sentence.

For example, Figure (2) is a dependency structure of a sentence. The head of the sentence is "have". There are four pairs of dependency relationships, depicted by four arcs from heads to modifiers.

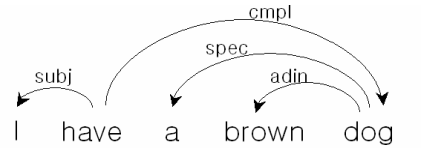

Figure -2. A dependency structure of a sentence.

We use Minipar as a dependency parser, which is a principle-based English parser (Lin, 1994). Minipar represents its grammar as a network where nodes represent grammatical categories and links represent types of dependency relationships.

Chow and Liu (1968) suggest the construction of a MST using mutual information for a dependence tree which was originally used in the Chow Expansion. However, we suggest using a dependency parse tree which is generated by linguistic dependency parser instead of the mutual information MST, because a dependency parse tree intuitively and linguistically represents the term dependence relations in the syntactic structure, which helps to capture the underlying semantics of a document.

3. DEPENDENCY STRUCTURE LANGUAGE MODEL

Given a query q and a document d, we are interested in estimating the conditional probability $p(d|q)$, i.e., the probability that d fits the observed q. After applying the Bayes' formula and dropping a document-independent constant, we have $p(q|d)p(d)$. Here $p(d)$ is a prior belief that d is relevant to any query and $p(q|d)$ is the query likelihood given the document, which captures how well the document generates the particular query q.

In the simplest case, $p(d)$ is assumed to be uniform, and so does not affect document ranking. This assumption has been taken in most previous works. In our study, we assume a uniform $p(d)$ in order to focus on the effect of dependency structure. With a uniform prior, the retrieval model reduces to the calculation of $p(q|d)$, where language modeling comes in.

The language models used in most previous works are the unigram models, which are the multinomial models that assign the probability,

$$p(q \mid d) = \prod_i p(q_i \mid d)$$

Clearly, the retrieval problem is now essentially reduced to unigram language model estimation. The unigram language model makes a strong assumption that each word occurs independently, and consequently, the probability of a word sequence becomes the product of the probabilities of the individual words. However the unigram model has some limitations to capture the term relations in a document.

The basic idea of our dependency structure language model is to capture the term relations in a linguistically practical way and can be described as follows. An interesting approximation to a joint probability distribution $p(q|d)$ is based on the identity

$$\begin{aligned} p(q \mid d) &= p(q_1, \cdots, q_n \mid d) \\ &= p(q_1 \mid d)p(q_2 \mid q_1, d)p(q_3 \mid q_2, q_1, d) \cdots p(q_n \mid q_{n-1}, \cdots, q_1, d) \end{aligned}$$

Suppose the words are not independent, but we can number the words so that $p(q_i|q_{i-1},...,q_1,d)$ is solely dependent on some preceding word $q_{j(i)}$ as in Chow Expansion theory. Then we obtain the product expansion

$$p(q \mid d) = p(q_1 \mid d)p(q_2 \mid q_{j(2)},d)p(q_3 \mid q_{j(3)},d)\cdots p(q_n \mid q_{j(n)},d) \tag{5}$$

where the function $j(i)$ exhibits the limited dependence of one variable on preceding variables.

By letting $x_i=1$ if q_i appears in document d, $x_i=0$ otherwise, we can write the probability of q_i given $q_{j(i)}$ as follows:

$$p(q_i \mid q_{j(i)},d) = \left[p_s(q_i \mid q_{j(i)},d)^{x_i}\, p_u(q_i \mid q_{j(i)},d)^{1-x_i}\right]^{x_{j(i)}} \times \left[p_s(q_i \mid d)^{x_i}\, p_u(q_i \mid d)^{1-x_i}\right]^{1-x_{j(i)}} \tag{6}$$

where $p_s(q_i|q_{j(i)},d)= p(x_i=1|x_{j(i)}=1,d)$, $p_u(q_i|q_{j(i)},d)= p(x_i=0|x_{j(i)}=1,d)$, $p_s(q_i|d)= p(x_i=1|d)$, and $p_u(q_i|d)= p(x_i=0|d)$. In the equation, $p_s(q_i|d)$ is used for "seen" word q_i that occurs in document d, and $p_u(q_i|d)$ for "unseen" word q_i that does not. $p_s(q_i|q_{j(i)},d)$ is used when q_i and $q_{j(i)}$ occur as dependency relation in document d, and $p_u(q_i|q_{j(i)},d)$ is used when q_i and $q_{j(i)}$ does not occur as dependency relation but $q_{j(i)}$ occurs in document d.

By substituting Equation (6) in Equation (5), taking the logarithm, and collecting terms, we obtain the following equation as in Chow Expansion theory.

$$\begin{aligned}
&\log p(q \mid d) \\
&= \log p(q_1 \mid d) + \sum_{i=2} \log p(q_i \mid q_{j(i)},d) \\
&= \log\left[p_s(q_1 \mid d)^{x_1}\, p_u(q_1 \mid d)^{1-x_1}\right] + \sum_{i=2} x_{j(i)} \log\left[p_s(q_i \mid q_{j(i)},d)^{x_i}\, p_u(q_i \mid q_{j(i)},d)^{1-x_i}\right] \\
&\quad + \sum_{i=2} (1-x_{j(i)}) \log\left[p_s(q_i \mid d)^{x_i}\, p_u(q_i \mid d)^{1-x_i}\right] \\
&= \sum_{i=1} x_i \log \frac{p_s(q_i \mid d)}{p_u(q_i \mid d)} + \sum_{i=2} x_{j(i)} \log \frac{p_u(q_i \mid q_{j(i)},d)}{p_u(q_i \mid d)} \\
&\quad + \sum_{i=2} x_i x_{j(i)} \left(\log \frac{p_s(q_i \mid q_{j(i)},d)}{p_u(q_i \mid q_{j(i)},d)} - \log \frac{p_s(q_i \mid d)}{p_u(q_i \mid d)} \right) + \sum_{i=1} \log p_u(q_i \mid d)
\end{aligned} \tag{7}$$

Let $c(q_i;d)$ denote the count of word q_i in document d, and $c(q_i,q_{j(i)};d)$ denote the count of occurrence of q_i and $q_{j(i)}$ as dependency relation in document d. Then $x_i=1$ means $c(q_i;d)>0$, $x_{j(i)}=1$ means $c(q_{j(i)};d)>0$, and $x_ix_{j(i)}=1$ means $c(q_i,q_{j(i)};d)>0$. So we can re-write Equation (7) in $c(q_i;d)$ and $c(q_i,q_{j(i)};d)$ terms as follows:

$$\begin{aligned}
\log p(q \mid d) = &\sum_{i:c(q_i;d)>0} \log \frac{p_s(q_i \mid d)}{p_u(q_i \mid d)} + \sum_{i:c(q_{j(i)};d)>0} \log \frac{p_u(q_i \mid q_{j(i)},d)}{p_u(q_i \mid d)} \\
&+ \sum_{i:c(q_i,q_{j(i)};d)>0} \left(\log \frac{p_s(q_i \mid q_{j(i)},d)}{p_u(q_i \mid q_{j(i)},d)} - \log \frac{p_s(q_i \mid d)}{p_u(q_i \mid d)} \right) + \sum_i \log p_u(q_i \mid d)
\end{aligned} \tag{8}$$

Now we can see that the retrieval function can actually be decomposed into four parts. The first part involves a weight for each term which is common between the query and the document (i.e., matched terms). The second part involves a weight for head (or governor, parent) terms of matched terms. The third part involves a weight for the occurrence of matched terms and their head terms as dependency relation. The last part only involves a document-dependent constant that is related to how much probability mass will be

allocated to unseen words, according to the particular smoothing method used. In our study, we ignore the last part (i.e., document-dependent constant) in order to focus on the effect of the second part and the third part.

In Zhai and Lafferty (2001), three smoothing methods (Jelinek-Mercer, Dirichlet, and absolute discounting) are compared. In the comparison, Jelinek-Mercer and Dirichlet clearly have a better average precision than absolute discounting. Considering these results, we use two smoothing methods (Jelinek-Mercer and Dirichlet) for our dependency structure language model.

3.1 Jelinek-Mercer Smoothing Method

This method involves a linear interpolation of the maximum likelihood model with the fallback model (i.e. collection model), using a coefficient λ to control the influence of each model.

$$p_\lambda(q_i \mid d) = (1-\lambda) \cdot p_{ml}(q_i \mid d) + \lambda \cdot p(q_i \mid C)$$

Using this smoothing method, we define $p_s(q_i|d)$ and $p_u(q_i|d)$ as follows:

$$\begin{aligned} &p_s(q_i \mid d) = (1-\lambda_1) \cdot p_{ml}(q_i \mid d) + \lambda_1 \cdot p(q_i \mid C) \\ &p_u(q_i \mid d) = \lambda_1 \cdot p(q_i \mid C) \\ &\text{where } p_{ml}(q_i \mid d) = \frac{c(q_i; d)}{\sum_k c(q_k; d)} \end{aligned} \tag{9}$$

We also define $p_s(q_i|q_{j(i)},d)$ and $p_u(q_i|q_{j(i)},d)$ using the same smoothing method as follows:

$$\begin{aligned} &p_s(q_i \mid q_{j(i)}, d) = (1-\lambda_2) \cdot p_{ml}(q_i \mid q_{j(i)}, d) + \lambda_2 \cdot p(q_i \mid q_{j(i)}, C) \\ &p_u(q_i \mid q_{j(i)}, d) = \lambda_2 \cdot p(q_i \mid q_{j(i)}, C) \\ &\text{where } p_{ml}(q_i \mid q_{j(i)}, d) = \frac{c(q_i, q_{j(i)}; d)}{\sum_k c(q_k, q_{j(i)}; d)} \end{aligned} \tag{10}$$

By substituting Equation (9) and Equation (10) into Equation (8), we obtain the following equation.

$$\begin{aligned} &\log(q \mid d) = \\ &\sum_{i:c(q_i;d)>0} \log\left(1 + \frac{(1-\lambda_1) \cdot p_{ml}(q_i \mid d)}{\lambda_1 \cdot p(q_i \mid C)}\right) + \sum_{i:c(q_{j(i)};d)>0} \log \frac{\lambda_2 \cdot p(q_i \mid q_{j(i)}, C)}{\lambda_1 \cdot p(q_i \mid C)} \\ &+ \sum_{i:c(q_i,q_{j(i)};d)>0} \left(\log\left(1 + \frac{(1-\lambda_2) \cdot p_{ml}(q_i \mid q_{j(i)}, d)}{\lambda_2 \cdot p(q_i \mid q_{j(i)}, C)}\right) - \log\left(1 + \frac{(1-\lambda_1) \cdot p_{ml}(q_i \mid d)}{\lambda_1 \cdot p(q_i \mid C)}\right) \right) + \sum_i \log\left(\lambda_1 \cdot p(q_i \mid C)\right) \end{aligned} \tag{11}$$

In the equation, we ignore the last part (i.e., document-dependent constant) in order to focus on the effect of the second part and the third part.

From the Equation (11), we define $MS_{DSLM\text{-}JM}(q,d)$[1], a query-document scoring function adapted from dependency structure language model using Jelinek-Mercer smoothing method as follows:

[1] MS: Matching Score, DSLM-JM: Dependency Structure Language Model – Jelinek-Mercer smoothing

$$MS_{DSLM-JM}(q,d) = \sum_{i:c(q_i;d)>0} \log\left(1+\frac{(1-\lambda_1)\cdot p_{ml}(q_i \mid d)}{\lambda_1 \cdot p(q_i \mid C)}\right) + \sum_{i:c(q_{j(i)};d)>0} k \cdot \log\frac{\lambda_2 \cdot p(q_i \mid q_{j(i)}, C)}{\lambda_1 \cdot p(q_i \mid C)} + \sum_{i:c(q_i,q_{j(i)};d)>0} k\left(\log\left(1+\frac{(1-\lambda_2)\cdot p_{ml}(q_i \mid q_{j(i)}, d)}{\lambda_2 \cdot p(q_i \mid q_{j(i)}, C)}\right) - \log\left(1+\frac{(1-\lambda_1)\cdot p_{ml}(q_i \mid d)}{\lambda_1 \cdot p(q_i \mid C)}\right)\right) \quad (12)$$

where k is a constant parameter to control the influence of second and third part. In the formula, $p(q_i|q_{j(i)},C)$ can be zero because of data sparseness problem. To solve this problem, we also apply smoothing to $p(q_i|q_{j(i)},C)$ as follows.

$$p(q_i \mid q_{j(i)}, C) = (1-\lambda_3)\cdot p_{ml}(q_i \mid q_{j(i)}, C) + \lambda_3 \cdot p(q_i \mid C)$$

3.2 Dirichlet Smoothing Method

A language model is a multinomial distribution, for which the conjugate prior for Bayesian analysis is the Dirichlet distribution with parameters:

$$(\mu p(q_1 \mid C), \mu p(q_2 \mid C), \cdots, \mu p(q_n \mid C))$$

Thus, the model is given by

$$p_\mu(q_i \mid d) = \frac{c(q_i;d) + \mu \cdot p(q_i \mid C)}{\sum_i c(q_i;d) + \mu}$$

Using this smoothing method, we define $p_s(q_i|d)$ and $p_u(q_i|d)$ as follows:

$$p_s(q_i \mid d) = \frac{c(q_i;d) + \mu_1 \cdot p(q_i \mid C)}{\sum_k c(q_k;d) + \mu_1}, \quad p_u(q_i \mid d) = \frac{\mu_1 \cdot p(q_i \mid C)}{\sum_k c(q_k;d) + \mu_1} \quad (13)$$

We also define $p_s(q_i|q_{j(i)},d)$ and $p_u(q_i|q_{j(i)},d)$ using the same smoothing method as follows:

$$p_s(q_i \mid q_{j(i)}, d) = \frac{c(q_i, q_{j(i)};d) + \mu_2 \cdot p(q_i \mid q_{j(i)}, C)}{\sum_k c(q_k, q_{j(i)};d) + \mu_2}$$
$$p_u(q_i \mid q_{j(i)}, d) = \frac{\mu_2 \cdot p(q_i \mid q_{j(i)}, C)}{\sum_k c(q_k, q_{j(i)};d) + \mu_2} \quad (14)$$

By substituting Equation (13) and Equation (14) into Equation (8), we obtain the following equation.

$$\log(q \mid d) = \sum_{i:c(q_i;d)>0} \log\left(1+\frac{c(q_i;d)}{\mu_1 \cdot p(q_i \mid C)}\right) + \sum_{i:c(q_{j(i)};d)>0} \log\left(\frac{\sum_k c(q_k;d)+\mu_1}{\sum_k c(q_k,q_{j(i)};d)+\mu_2} \cdot \frac{\mu_2 \cdot p(q_i \mid q_{j(i)},C)}{\mu_1 \cdot p(q_i \mid C)}\right) \quad (15)$$

$$+ \sum_{i:c(q_i,q_{j(i)};d)>0} \left(\log\left(1+\frac{c(q_i,q_{j(i)};d)}{\mu_2 \cdot p(q_i \mid q_{j(i)},C)}\right) - \log\left(1+\frac{c(q_i;d)}{\mu_1 \cdot p(q_i \mid C)}\right)\right) + (\text{document dependent constant})$$

In the equation, we again ignore the document-dependent constant.

From the Equation (15), we define $MS_{DSLM\text{-}D}(q,d)$, a query-document scoring function adapted from dependency structure language model using Dirichlet smoothing method as follows:

$$MS_{DSLM-D}(q,d) =$$

$$\sum_{i:c(q_i;d)>0} \log\left(1+\frac{c(q_i;d)}{\mu_1 \cdot p(q_i \mid C)}\right) + \sum_{i:c(q_{j(i)};d)>0} k \cdot \log\left(\frac{\sum_k c(q_k;d)+\mu_1}{\sum_k c(q_k,q_{j(i)};d)+\mu_2} \cdot \frac{\mu_2 \cdot p(q_i \mid q_{j(i)},C)}{\mu_1 \cdot p(q_i \mid C)}\right) \quad (16)$$

$$+ \sum_{i:c(q_i,q_{j(i)};d)>0} k\left(\log\left(1+\frac{c(q_i,q_{j(i)};d)}{\mu_2 \cdot p(q_i \mid q_{j(i)},C)}\right) - \log\left(1+\frac{c(q_i;d)}{\mu_1 \cdot p(q_i \mid C)}\right)\right)$$

where k is a constant parameter to control the influence of second and third part. In the formula, $p(q_i|q_{j(i)},C)$ can also be zero because of the data sparseness problem, so we again apply smoothing to $p(q_i|q_{j(i)},C)$ as follows.

$$p(q_i \mid q_{j(i)},C) = \frac{\sum_d c(q_i,q_{j(i)};d) + \mu_3 \cdot p(q_i \mid C)}{\sum_d \sum_k c(q_k,q_{j(i)};d) + \mu_3}$$

4. EXPERIMENT

4.1 Experiment Design

The goal of our experiment is to answer the following two questions:

1. *Will the dependency structure language model be effective for information retrieval?* To answer this question, we will compare the performance of dependency structure language model with that of the state-of-the-art information retrieval method, which is recently proposed language model for information retrieval.
2. *Will the dependency syntactic structure be more effective than a bigram model which only models the co-occurrence of terms in language modeling for information retrieval?* To answer this question, we will compare the result for the dependency structure language model using dependency parse tree with the result for the same model using only bigram dependency (i.e. assuming the dependency parse tree is linear).

We used two different TREC testing collections for evaluation: AP88 (Associated Press, 1988), WSJ90-92 (wall street journal from 1990 to 1992). We used TREC4 queries (202-250) and their relevance judgments for evaluation. We excluded the TREC topic 201 from the experiments, because the topic's relevant documents are not included in AP88 test collection.

4.2 Baseline Methods

The baseline method is Zhai's language model. In Zhai and Lafferty (2001), three smoothing methods are compared. We use two smoothing methods (Jelinek-Mercer and Dirichlet) as our baseline methods. The formula which uses Jelinek-Mercer smoothing method is given by

$$MS_{LM-JM}(q,d) = \sum_{i:c(q_i;d)>0} \log\left(1 + \frac{(1-\lambda)\cdot p_{ml}(q_i \mid d)}{\lambda \cdot p(q_i \mid C)}\right).$$

The formula applied Dirichlet smoothing method is given by

$$MS_{LM-D}(q,d) = \sum_{i:c(q_i;d)>0} \log\left(1 + \frac{c(q_i;d)}{\mu \cdot p(q_i \mid C)}\right).$$

4.3 Experiment Results

The results on AP88 test collection are shown in Table 1 and Table 2. Table 3 shows the result on WSJ90-92 test collection. In each table, we include the precision at different recall points and the average precision (non-interpolated).

	LM-D	DSLM-D-Bigram	DSLM-D-Minipar
Recall 0.0	0.5647	0.5859	**0.6279**
Recall 0.1	0.4646	0.4706	**0.4977**
Recall 0.2	0.4080	0.4209	**0.4422**
Recall 0.3	0.3613	0.3757	**0.3774**
Recall 0.4	0.3198	**0.3421**	0.3337
Recall 0.5	**0.2960**	0.2910	0.2872
Recall 0.6	0.2135	**0.2333**	0.2239
Recall 0.7	0.1631	0.1816	**0.1825**
Recall 0.8	0.0986	**0.1159**	0.1109
Recall 0.9	0.0591	0.0671	**0.0674**
Recall 1.0	0.0433	**0.0474**	0.0466
Avg. Prec.	0.2545	0.2659	**0.2716**

Table-1. Results for AP88 collection (for Dirichlet smoothing).

In Table 1, *LM-D* stands for Zhai's model applied to Dirichlet smoothing method, *DSLM-D-Bigram* stands for dependency structure language model which is restricted to only bigram dependency, and *DSLM-D-Minipar* stands for dependency structure language model using dependency parse tree generated by Minipar.

As seen from Table 1, the dependency structure language model applied to Dirichlet smoothing method using dependency parse tree generated by Minipar (*DSLM-D-Minipar*) has a significant gain in performance. *DSLM-D-Minipar* achieved about 6.72% improvement compared to Zhai's model (*LM-D*) in the average

precision (the difference is statistically significant at the 0.05 level)[2]. *DSLM-D-Minipar* also achieved about 2.1% improvement compared to *DSLM-D-Bigram* (statistically significant at the 0.1 level). So, the table shows that the dependency structure language model is effective for information retrieval. One notable fact is that the *DSLM-D-Minipar* performance includes the error of Minpar parsing in the given corpus. We hope we can archive much better performance if we can improve the dependency parsing accuracy, which is about 85% in the current Minipar system.

	LM-JM	DSLM-JM-Bigram	DSLM-JM-Minipar
Recall 0.0	0.5687	**0.5688**	0.5638
Recall 0.1	0.4470	0.4508	**0.4531**
Recall 0.2	0.3857	0.3910	0.4062
Recall 0.3	0.3233	0.3301	0.3402
Recall 0.4	0.2615	0.2785	0.2967
Recall 0.5	0.2364	0.2449	0.2621
Recall 0.6	0.1833	0.1907	**0.2107**
Recall 0.7	0.1521	0.1589	**0.1775**
Recall 0.8	0.1132	**0.1194**	0.1177
Recall 0.9	0.0693	0.0725	0.0789
Recall 1.0	0.0453	0.0515	**0.0566**
Avg. Prec.	0.2328	0.2400	**0.2527**

Table-2. Results for AP88 collection (for Jelinek-Mercer smoothing).

In Table 2, *LM-JM* stands for Zhai's model with Jelinek-Mercer smoothing method, *DSLM-JM-Bigram* stands for dependency structure language model which is restricted to only bigram dependency, *DSLM-JM-Minipar* stands for dependency structure language model using dependency parse tree, and *DSLM-JM-Interpolation* stands for the interpolation model of *DSLM-JM-Minipar* and *DSLM-JM-Bigram*.

Table 2 also shows that the dependency structure is more effective than the bigram in language modeling for information retrieval. The dependency structure language model (*DSLM-JM-Minipar*) has the best average precision. *DSLM-JM-Minipar* achieved about 8.55% improvement for Zhai's model with Jelinek-Mercer smoothing method (*LM-JM*) in the average precision (statistically significant at the 0.05 level). *DSLM-JM-Minipar* also achieved about 5.29% improvement for *DSLM-JM-Bigram* in the average precision (statistically significant at the 0.1 level).

	LM-D	DSLM-D-Minipar	LM-JM	DSLM-JM-Minipar
Recall 0.0	**0.5974**	0.5962	0.5015	**0.5133**
Recall 0.1	0.4226	**0.4339**	0.4033	**0.4162**
Recall 0.2	0.3281	**0.3457**	0.3414	**0.3533**
Recall 0.3	0.2365	**0.2574**	0.2595	**0.2711**
Recall 0.4	0.1620	**0.1972**	0.2074	**0.2152**
Recall 0.5	0.1361	**0.1699**	0.1637	**0.1713**
Recall 0.6	0.1097	**0.1408**	0.1249	**0.1315**
Recall 0.7	0.0619	**0.0918**	0.0676	**0.0725**
Recall 0.8	0.0513	**0.0699**	0.0517	**0.0550**
Recall 0.9	0.0331	**0.0499**	0.0284	**0.0314**
Recall 1.0	0.0207	**0.0346**	0.0148	**0.0166**
Avg. Prec.	0.1735	**0.1929**	0.1792	**0.1866**

Table-3. Results for WSJ90-92 collection.

Table 3 shows the results for WSJ90-92 test collection. As seen from Table 3, *DSLM-D-Minipar* and *DSLM-JM-Minipar* have significant gains in performance compared to *LM-D* and *LM-JM* respectively.

[2] We performed the Student t-test (one sided).

So, in general, we can verify that the dependency structure language model is more effective than the conventional language modeling for information retrieval.

5. CONCLUSION AND FUTURE WORK

In the language model for information retrieval, bigram and trigram language models have some limitations to capture the underlying semantics in a document due to their inability to handle the long-distance dependencies. In this paper, we propose a dependency structure language model to compensate for the weakness of the bigram and trigram language models in information retrieval. The dependency structure language model is based on the Chow Expansion theory and the dependency parse tree generated by a dependency parser. So, long-distance dependencies can be naturally handled by the dependency structure language model.

We carried out some experiments to verify the proposed model. The experiments were performed on both AP88 and WSJ90-92 test collection. Based on the experiment results, we can draw the following conclusions:

- Based on the comparison between the dependency structure language model and traditional language model, we can conclude that the dependency structure language model for information retrieval is an effective retrieval method. In our experiments, the dependency structure model gives a better performance than the traditional language model.
- Based on the comparison between the dependency structure and bigram dependency in language modeling for information retrieval, we can also conclude that the dependency structure is more effective than the bigram in language modeling for information retrieval.

The disadvantage in using the dependency parser is that the computational cost of dependency parse tree becomes high because the dependency parse tree of the user query is obtained by a dependency parser at the search time and the co-occurrence information between the two terms are obtained by a dependency parser at the indexing time. To reduce this computational cost, we need to find a way that we do not rely on a full dependency parser but use more simplistic phrase chunker or partial parser in the future.

REFERENCES

Charniak, E. (2001). Immediate-head parsing for language models. In *Proceedings of ACL' 2001*.

Chelba, C. and Jelinek, F. (1998). Exploiting syntactic structure for language modeling. *In Proceedings of COLING-ACL' 1998.*

Chow, C. and Liu, C. (1968). Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory, IT-14(3),* pp. 462-467.

Duda, R. O. and Hart, P. E. (1973). Pattern Classification and Scene Analysis. *A Wiley-Interscience publication*, pp. 111-113.

Hays, D. (1964). Dependency theory: a formalism and some observations. *Language*, 40:511-525.

Lin, D. (1994). Principar – an efficient, broad-coverage, principle-based parser. *In Proceedings of COLING-ACL' 1994.*

Miller, D., Leek, T. and Schwartz, R. M. (1999). A hidden Markov model information retrieval system. In *Proceedings of SIGIR' 1999,* pp. 214-222.

Ponte, J. and Croft, W. B. (1998). A language modeling approach to information retrieval. In *Proceedings of SIGIR' 1998,* pp. 275-281.

Roark, B. (2001). Probabilistic top-down parsing and language modeling. *Computational Linguistics*, 27(2).

Song, F. and Croft, B. (1999). A general language model for information retrieval. In *Proceedings of SIGIR' 1999,* pp. 279-280.

van Rijsbergen, C. J. (1977). A Theoretical Basis for the Use of Co-Occurrence Data in Information Retrieval. *Journal of Documentation, 33,* pp. 106-119.

Zhai, C. and Lafferty, J. (2001). A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of SIGIR' 2001,* pp. 334-342.