Relevance Ranking Using Kernels

Jun Xu[1], Hang Li[1], and Chaoliang Zhong[2,⋆]

[1] Microsoft Research Asia, 4F Sigma Center, No. 49 Zhichun Road, Beijing, China 100190
[2] Beijing University of Posts and Telecommunications, Beijing, China 100876
{junxu,hangli}@microsoft.com, clzhong@cn.fujitsu.com

Abstract. This paper is concerned with relevance ranking in search, particularly that using term dependency information. It proposes a novel and unified approach to relevance ranking using the kernel technique in statistical learning. In the approach, the general ranking model is defined as a kernel function of query and document representations. A number of kernel functions are proposed as specific ranking models in the paper, including BM25 Kernel, LMIR Kernel, and KL Divergence Kernel. The approach has the following advantages. (1) The (general) model can effectively represent different types of term dependency information and thus can achieve high performance. (2) The model has strong connections with existing models such as BM25 and LMIR. (3) It has solid theoretical background. (4) The model can be efficiently computed. Experimental results on web search dataset and TREC datasets show that the proposed kernel approach outperforms MRF and other baseline methods for relevance ranking.

Keywords: Information Retrieval, Relevance Ranking, Kernel.

1 Introduction

Relevance ranking is still a central issue of research in Information Retrieval. Traditionally, relevance ranking models are defined based on the bag-of-words assumption, i.e., query and document are viewed as two sets of words. These include Vector Space Model (VSM) [17], BM25 [15], and Language Models for IR (LMIR) [22].

There is a clear trend in IR recently on study of models beyond bag-of-words, particularly under the notion of term dependency or proximity, which assumes terms are dependent and takes into consideration in relevance ranking the cooccurrences of query terms in documents. For example, if the query is "machine learning book", then a document containing the phrase "machine learning" and the term "book" should be more relevant than a document in which the terms "machine", "learning", and "book" occur separately. It is obvious from the example that dependency information needs to be leveraged in relevance ranking. For instance, a model for using term dependency based on Markov Random Fields (MRF) has been proposed [11], in which the information on independent terms (unigrams), sequential terms (n-grams), and dependent terms is exploited. MRF is regarded as a state-of-the-art method for using term dependency.

In our view, an ideal ranking model using term dependency should have the following characteristics. (1) It can incorporate various types of term dependency information

⋆ The work was conducted when Chaoliang Zhong was visiting Microsoft Research Asia.

P.-J. Cheng et al. (Eds.): AIRS 2010, LNCS 6458, pp. 1–12, 2010.

to represent relevance, and thus can achieve high accuracy in ranking. (2) It has strong connections with existing models . (This is necessary, as existing models of BM25 and LMIR already perform quite well.) (3) It is based on solid theory. (4) It can be efficiently computed. Although previous work on term dependencies made progresses, further study on the problem still appears to be necessary. In this paper, we aim to conduct a comprehensive study on relevance ranking using term dependency, and to propose a general ranking model which has the advantages. Specifically this paper proposes a novel approach to relevance ranking, on the basis of String Kernel in statistical learning. It defines the relevance ranking model as a kernel function of query string and document string. The original query string and document string are mapped into vectors in a higher dimensional Hilbert Space and relevance (similarity) between the two representations is defined as dot product in the Hilbert Space.

As case studies, we introduce two kernel functions, BM25 Kernel and LMIR Kernel, and use one existing kernel function, Kullback-Leibler (KL) Divergence Kernel, in this paper. The former two are asymmetric kernel functions and the last is a symmetric kernel function. They correspond to different ways of mapping the original query and document strings to the Hilbert Space.

(1) The kernel based models can naturally represent various types of term dependency information, for example, n-grams and n-dependent-terms. Since the essence of relevance is to compare the similarity between query and document, a richer model on the similarity between query and document can then be realized with the approach and high performance in relevance ranking can be achieved. (2) BM25 Kernel and LMIR Kernel respectively include conventional BM25 model and LMIR model as special cases. (Note that many other relevance models can also be interpreted as kernel functions, for example, the traditional VSM.) Within the kernel framework, different ranking models can be easily studied and compared. (3) Kernel methods are becoming very popular in statistical learning. It is attractive because of its powerful representability. The proposed approach using kernel thus has a solid theoretical background. It bridges the gap between relevance ranking and machine learning, and provide a framework for automatically learning ranking model using kernel machines. (4) Kernel functions are essentially dot product. This enables an efficient computation in search.

The implication of the work is far beyond using term dependency. The proposed kernel-based ranking models are actually generalized versions of conventional ranking models. It provides a new view, namely kernel function, on relevance ranking.

We conducted experiments using data from a web search engine and TREC data, and found that BM25 Kernel, LMIR Kernel, and KL Kernel perform better than MRF and other bag-of-words models for relevance ranking. We conclude, therefore, that the kernel approach proposed in this paper really has advantages.

2 Related Work

Conventional Information Retrieval models such as Vector Space Model (VSM) [17], BM25 [15], and Language Models for IR (LMIR) [22] were originally proposed based on the bag-of-words assumption, which is obviously too strong, as in many cases query terms (words) are related, and their relations (dependencies) should also be considered

in relevance ranking. The ranking models beyond the bag-of-words may be categorized into two groups. One approach considers the use of term dependency, specifically, it assumes that the query is composed of several units of terms (e.g., n-grams) and utilizes the occurrences of the units in the document in ranking. For example, Metzler and Croft [11] proposed using Markov Random Fields to characterize different types of term dependency relations. For other related work see [1,2,4,13]. The other approach considers the use of degree of approximate match of query to document, for example, it looks at the minimum span of the query terms appearing in the document, also referred to as proximity. Tao and Zhai [19] conducted a comprehensive study on proximity measures for relevance ranking and found that the proximity measures are useful for further improving relevance. (Also refer to [8,6].)

Kernel function characterizes a kind of similarity between two representations $k(x, y)$, where x and y denote representations. For example, String Kernel is a special type of kernel function defined on text strings, i.e., x and y are strings [5,9]. KL Divergence Kernel is a type of kernel function defined based on KL Divergence [12]. Kernel functions have been applied into several tasks in IR. In [21], the KL Divergence Kernel was used for calculating the similarity between documents given a query. In [9], a method for text classification using String Kernels was studied. In [16], a kernel function for measuring similarity between short texts was proposed. In [10], a number of String Kernel functions were surveyed.

There are two ways to create (or identify) a kernel function. The Mercer's theorem provides a sufficient and necessary condition of kernel function, that is, any continuous, symmetric, positive semi-definite function $k(x, y)$ is a kernel function. One can also define a kernel function in a constructive way, i.e., through definition of mapping function. Usually a kernel function is symmetric in the sense $k(x, y) = k(y, x)$. Asymmetric kernel functions have also been introduced, which satisfy $k(x, y) \neq k(y, x)$ [20]. Obviously asymmetric kernel functions are more general than symmetric kernel functions.

3 Kernel Approach

The essence of relevance is to compare the similarity between query and document, or the matching degree between query and document. Different ways of defining the similarity or matching degree lead to different ranking models. In this paper, we use kernel techniques in statistical learning to construct ranking models. We first give a general model, and then some specific examples.

3.1 General Model

There are three issues we need to consider when defining a ranking model: query representation, document representation, and similarity calculation between query and document representations. Here, we assume that query and document are two strings of words (terms). Such a representation can retain all the major information contained in the query and document. We then define the similarity function between the query and document as kernel function between query string and document string.

Suppose that $\mathcal{Q}$ is the space of queries (word strings), and $\mathcal{D}$ is the space of documents (word strings). (q, d) is a pair of query and document from $\mathcal{Q}$ and $\mathcal{D}$. There is

a mapping function $\phi : \mathcal{Q} \mapsto \mathcal{H}$ from the query space to the Hilbert space where $\mathcal{H}$ denotes the Hilbert Space. $\phi(q)$ then stands for the mapped vector in $\mathcal{H}$ from q. Similarly, there is a mapping function $\phi' : \mathcal{D} \mapsto \mathcal{H}$ from the document space to the Hilbert Space $\mathcal{H}$. $\phi'(d)$ then stands for the mapped vector in $\mathcal{H}$ from d. The similarity function between q and d is defined as a kernel function between q and d,

$$k(q,d) = \langle \phi(q), \phi'(d) \rangle = \sum_i \phi(q)_i \cdot \phi'(d)_i. \tag{1}$$

The kernel function is actually the dot product between vectors $\phi(q)$ and $\phi'(d)$, respectively mapped from q and d. We call the kernel function $k(q,d)$ in Eq. (1) (general) kernel-based ranking model. Note that $k(q,d)$ is an *asymmetric* function, which means $k(q,d) \neq k(d,q)$, because ϕ and ϕ' are not necessarily identical. When ϕ and ϕ' are identical, the kernel function becomes a symmetric function, i.e., $k(q,d) = k(d,q)$.

The general kernel-based ranking model can include most existing relevance ranking models such as VSM, BM25, LMIR, and even LSI [3], as will be made clearer later. For example, in VSM, the query string and the document string are mapped into vectors of tf-idf values and dot product between the vectors is calculated and used, which is a very simple example of the general ranking function in Eq. (1).

3.2 Model Construction

Let us introduce the way of creating a kernel-based ranking model. We first identify a 'type of dependent query terms' for which we care about the occurrences in documents. (For example, if the type is bigram, we decompose the query "machine learning book" into two bigrams "machine learning" and "learning book", and we look at their occurrences in documents.) Next we assume to respectively map the query string and document string into vectors of the type in a Hilbert Space. (For example, we map query and document into vectors of bigrams.) We can then construct a kernel function for the type on the basis of the mapping functions. Finally, we can define a ranking model by combining the kernel functions in different types using the properties of kernel functions (cf., Section 4.3). That is to say we actually create kernel functions.

As types, we can consider the occurrences of independent query terms in the document (unigrams), the occurrences of sequential query terms in the document (n-grams) , or the occurrences of dependent query terms in the document (n-dependent-terms). Then we can create a vector of unigrams, a vector of n-grams, or a vector of n-dependent-terms in the Hilbert Space. The relation between unigrams, n-grams, and n-dependent-terms is illustrated in Fig. 1. For each type, the set of possible units (unigrams, n-grams, or n-dependent-terms) is fixed, provided that the vocabulary is fixed. We denote the set of units as $\mathcal{X}$ and each element of it x.

In this paper we give three examples of kernel-based models. They are BM25 Kernel, LMIR Kernel, and KL Kernel.

3.3 BM25 Kernel

BM25 Kernel is a generalization of conventional BM25 model using String Kernel. It has a similar form as BM25, but is more general in the sense that it can make use of term dependencies in different types. It is an asymmetric kernel function.

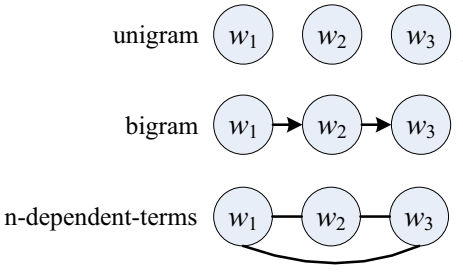

Fig. 1. Illustration of unigram, n-gram, and n-dependent-terms

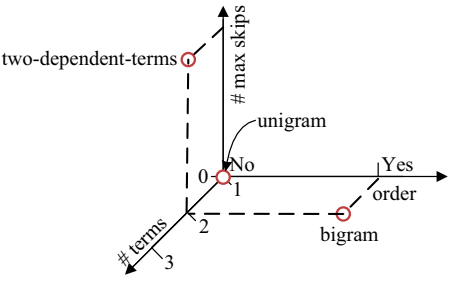

Fig. 2. Three factors for term dependency

The BM25 Kernel function between query q and document d in type t is defined as

$$\texttt{BM25-Kernel}_t(q,d) = \langle \phi_t^{\texttt{BM25}}(q), \phi_t'^{\texttt{BM25}}(d) \rangle,$$

where $\phi_t^{\texttt{BM25}}(q)$ and $\phi_t'^{\texttt{BM25}}(d)$ are two vectors of type t in the Hilbert Space mapped from q and d. Each dimension of the vector corresponds to a unit x:

$$\phi_t^{\texttt{BM25}}(q)_x = \frac{(k_3+1)\cdot f_t(x,q)}{k_3 + f_t(x,q)} \text{ and } \phi_t'^{\texttt{BM25}}(d)_x = \texttt{IDF}_t(x) \cdot \frac{(k_1+1)\cdot f_t(x,d)}{k_1\left(1-b+b\frac{f_t(d)}{avgf_t}\right) + f_t(x,d)},$$

where k_1, k_3, and b are parameters, $f_t(x,q)$ and $f_t(x,d)$ are frequencies of unit x with type t in q and d, respectively, $f_t(d)$ is total number of units with type t in document d , and $avgf_t$ is average number of units $f_t(d)$ with type t per document within the whole collection. $\texttt{IDF}_t(x) = \log \frac{df_t - df_t(x) + 0.5}{df_t(x) + 0.5}$ is inverse document frequency of unit x with type t, where $df_t(x)$ is number of documents in which unit x with type t occurs, and df_t is total number of documents that have units with type t. BM25 Kernel in type t becomes

$$\texttt{BM25-Kernel}_t(q,d) = \sum_x \texttt{IDF}_t(x) \times \frac{(k_3+1)\times f_t(x,q)}{k_3 + f_t(x,q)} \times \frac{(k_1+1)\times f_t(x,d)}{k_1\left(1-b+b\frac{f_t(d)}{avgf_t}\right) + f_t(x,d)}.$$

Finally, BM25 Kernel is defined as a linear combination of kernels over all the types:

$$\texttt{BM25-Kernel}(q,d) = \sum_t \alpha_t \texttt{BM25-Kernel}_t(q,d), \ \alpha_t \geq 0. \tag{2}$$

3.4 LMIR Kernel

We employ Dirichlet smoothing as an example. Other smoothing methods can also be used. The LMIR Kernel between query q and document d in type t is defined as

$$\texttt{LMIR-Kernel}_t(q,d) = \langle \phi_t^{\texttt{LMIR}}(q), \phi_t'^{\texttt{LMIR}}(d) \rangle + f_t(q) \cdot \log \frac{\mu}{f_t(d) + \mu},$$

where μ is free parameter, and $\phi_t^{\texttt{LMIR}}(q)$ and $\phi_t'^{\texttt{LMIR}}(d)$ are two vectors of type t in the Hilbert Space mapped from query q and document d, respectively. Each dimension of the vector corresponds to a unit x:

$$\phi_t^{\texttt{LMIR}}(q)_x = f_t(x,q) \text{ and } \phi_t'^{\texttt{LMIR}}(d)_x = \log\left(1 + \frac{f_t(x,d)}{\mu P(x|t)}\right),$$

where $P(x|t)$ is probability of unit x with type t in the whole collection. $P(x|t)$ plays a similar role as inverse document frequency $\texttt{IDF}_t(x)$ in BM25 Kernel. Combining the two, the LMIR Kernel function in type t becomes

$$\texttt{LMIR-Kernel}_t(q,d) = \sum_x f_t(x,q) \cdot \log\left(1 + \frac{f_t(x,d)}{\mu P(x|t)}\right) + f_t(q) \cdot \log\frac{\mu}{f_t(d)+\mu}.$$

Finally, we have

$$\texttt{LMIR-Kernel}(q,d) = \sum_t \alpha_t \texttt{LMIR-Kernel}_t(q,d),\ \alpha_t \geq 0. \tag{3}$$

3.5 KL Kernel

In this paper, we also use the KL Divergence kernel as a specific kernel-based ranking model, referred to as KL Kernel. Unlike BM25 Kernel and LMIR Kernel, KL Kernel is a *symmetric* kernel function. Another difference is that KL Kernel represents a mapping of the query and document into a Hilbert Space with an infinite number of dimensions. KL Kernel was previously used in document similarity comparison [12], and this is the first time it is used in relevance ranking.

In KL Kernel in a type t, the query and document are represented by distributions. We define a distribution $\mathbb{P}_t(q) = (P(x_1|t,q), P(x_2|t,q), \cdots, P(x_N|t,q))$ of units x with type t in query q, where $P(x_i|t,q)$ is probability of unit x_i given type t and query q, and N is size of $\mathcal{X}$. Similarly, we define a distribution $\mathbb{P}_t(d) = (P(x_1|t,d), P(x_2|t,d), \cdots, P(x_N|t,d))$ of units x with type t in document d. Then, the KL Kernel function between q and d in type t is defined as

$$\texttt{KL-Kernel}_t(q,d) = \exp\left\{-D(\mathbb{P}_t(q)||\mathbb{P}_t(d)) - D(\mathbb{P}_t(d)||\mathbb{P}_t(q))\right\}.$$

Note that the kernel function represents dot product between two vectors in a Hilbert Space with infinite number of dimensions. (We can use Mercer's theorem to prove that it is a kernel function. [12]) $D(\mathbb{P}_t(q)||\mathbb{P}_t(d))$ is the KL divergence of $\mathbb{P}_t(d)$ from $\mathbb{P}_t(q)$:

$$D(\mathbb{P}_t(q)||\mathbb{P}_t(d)) = \sum_{i=1}^{N} P(x_i|t,q) \log\frac{P(x_i|t,q)}{P(x_i|t,d)}.$$

The KL Kernel function is defined as

$$\texttt{KL-Kernel}(q,d) = \prod_t \texttt{KL-Kernel}_t(q,d)^{\alpha_t}. \tag{4}$$

According to the properties of kernel function, $\texttt{KL-Kernel}(q,d)$ is still a kernel. For efficiency consideration, we actually take logarithmic function of the kernel function. This will not change the ranking results.

$$\texttt{KL-Kernel}'(q,d) = \log\left(\texttt{KL-Kernel}(q,d)\right) = \sum_t \alpha_t \left(-D(\mathbb{P}_t(q)||\mathbb{P}_t(d)) - D(\mathbb{P}_t(d)||\mathbb{P}_t(q))\right).$$

Smoothing is also needed in estimation of the probability distributions in KL Kernel. In this paper we employ the Dirichlet smoothing method [22]: $P(x|t,d) = \frac{f_t(x,d)+\mu P(x|t)}{f_t(d)+\mu}$.

4 Advantages of the Kernel Approach

4.1 Representability

The kernel approach to ranking has rich representation ability. There are several orthogonal factors which one needs to consider when using term dependencies of different types. They are number of terms, order preservation, maximum number of skipping terms. Suppose that the query is "machine learning book". Number of terms indicates whether the term units we use consist of two terms or three terms, etc in documents ("machine learning" vs "machine learning book"). Order preservation means we care about the order of query terms ("machine learning" vs "learning machine") in documents. Maximum number of skipping terms includes the cases in which there are other terms occurring in between the terms in question ("machine book").

Combining the factors above can give us different types of dependencies, including some complex ones. The n-gram models belong to the cases in which different numbers of *ordered and non-skipping* terms are used. The n-dependent-terms models fall into the cases in which different numbers of *non-ordered and skipping* terms are used. The unigram model is the simplest. Fig. 2 shows the relations among the factors. Each factor is represented as one axis; the first axis corresponds to order preservation, the second maximum number of skips, and the third number of terms. It further shows the positions of different types of dependencies.

One can immediately notice that the kernel approach proposed in this paper can easily utilize the different types of term dependencies (types). In the experiments in this paper, we make use of three types: unigram, bigram, and two-dependent-terms.

4.2 Relation to Conventional Models

It is easy to verify that BM25 Kernel and LMIR Kernel respectively include conventional BM25 and LMIR models as special cases. Specifically, when the type is unigram, the kernel functions become the existing ranking models.

The kernel approach to relevance ranking proposed in this paper, thus, provides a new interpretation of many IR ranking models. These include not only BM25 and LMIR, but also VSM and even LSI. Relevance ranking models are nothing but dot product between vectors which are nonlinearly mapped from query and document.

The kernel functions (BM25 Kernel, LMIR Kernel, and KL Kernel) have similar shapes as functions of unit frequency in document (e.g., term frequency in document for the case of unigram). (Note that unit frequency in query is usually fixed to one, because queries are short.) Fig. 3 shows plots of the functions. From the figure we can see that although BM25 and LMIR were derived from different theories, they actually have similar shapes as functions of unit frequency. This can in part explain why BM25 and LMIR perform almost equally well in practice. This also strongly suggest kernel functions including KL Kernel would have similar performances in relevance ranking. As will be seen in Section 5, this is in fact true.

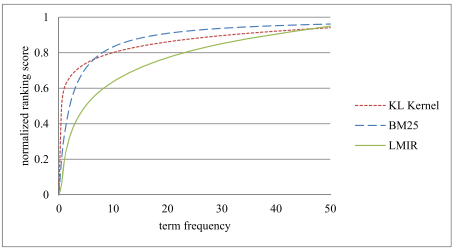

Fig. 3. KL Kernel, BM25 Kernel, and LMIR Kernel as a function of unit (term) frequency

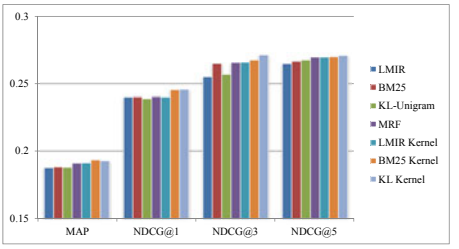

Fig. 4. Ranking accuracies on the web search data

4.3 Theoretical Background

Another advantage of taking the kernel approach is that it has solid theoretical background. As a result, deeper understanding of the problem and broader use of the technique become possible.

For example, it is easy to verify that kernel functions have the following closure properties, which means several ways of combining kernel functions will still result in kernel functions. Let k_1 and k_2 be any two kernels. Then the function k given by

$$1.\ k(x,y) = k_1(y,x); \qquad 2.\ k(x,y) = k_1(x,y) + k_2(x,y);$$
$$3.\ k(x,y) = \alpha k_1(x,y),\ \text{for all } \alpha > 0; \quad 4.\ k(x,y) = k_1(x,y) \cdot k_2(x,y)$$

are also kernels. Note that this is true even for *asymmetric* kernel functions. That means we can create different ranking models by taking sum or product of kernel functions.

In recent years, kernel learning[18,14], which aims to learn a kernel function from training data, has been proposed. The proposed kernel approach for relevance ranking bridges the gap between information retrieval and machine learning. It provides a framework for learning optimal ranking model using machine learning. We will further investigate the problem in our future work.

4.4 Efficient Implementation

The kernel functions proposed above can be calculated efficiently. This is because kernel functions are essentially dot products and their computations only need to be performed on units which occur in query or document. (The values of the other units are just zero). For example, for $D(\mathbb{P}_t(q)||\mathbb{P}_t(d))$ (with Dirichlet smoothing) in KL Kernel we actually only need to calculate

$$\sum_{x:f_t(x,q)>0 \vee f_t(x,d)>0} P(x|t,q) \log \frac{P(x|t,q)}{P(x|t,d)} + \sum_{x:f_t(x,q)=0 \wedge f_t(x,d)=0} \frac{\mu P(x|t)}{\mu + f_t(q)} \log \frac{f_t(d)+\mu}{f_t(q)+\mu}$$
$$= \sum_{x:f_t(x,q)>0 \vee f_t(x,d)>0} P(x|t,q) \log \frac{P(x|t,q)}{P(x|t,d)} + \frac{\mu}{\mu + f_t(q)} \log \frac{f_t(d)+\mu}{f_t(q)+\mu} \sum_{x:f_t(x,q)>0 \vee f_t(x,d)>0} (1 - P(x|t)),$$

where μ is the smoothing parameter.

5 Experiments

We conducted experiments to test the effectiveness of the proposed kernel approach, using a dataset from a web search engine and TREC data sets of OHSUMED [7] and AP. Specifically, we tested the performances of BM25 Kernel, LMIR Kernel, and KL Kernel. In the kernels, as types of term dependencies we used unigram, bigram, and two-dependent-terms. The final ranking score is defined as linear combination of the kernel function scores:

$$\text{Score} = (1 - \lambda_1 - \lambda_2) \cdot \text{kernel}_{unigram} + \lambda_1 \cdot \text{kernel}_{bigram} + \lambda_2 \cdot \text{kernel}_{two-dependent-terms},$$

where λ_1 and λ_2 are parameters. (It is still a kernel function, c.f., Section 4.3.)

BM25 and LMIR (with Dirichlet smoothing) were selected as baselines in the experiments, as representatives of bag-of-words models. We also viewed KL-Unigram as a baseline, in which only unigram is used in KL Kernel. MRF was also chosen as a baseline of using term dependencies. For MRF model the same term dependency information (unigram, bigram, and two-dependent-terms) was used for fair comparison.

We adopted mean average precision (MAP) and NDCG@N ($N = 1, 3$, and 5) as evaluation measures. The web search dataset has five levels of relevance: "Perfect", "Excellent", "Good", "Fair", and "Bad". We considered the first three as relevant when calculating MAP. The OHSUMED dataset have three levels of relevance: "definitely relevant", "partially relevant", and "not relevant". We considered "definitely relevant" and "partially relevant" as relevant when calculating MAP.

5.1 Experiments on the Web Search Dataset

In this experiment, we used the web search dataset. The dataset contains about 2.5 million web pages, and 235 long queries randomly selected from query log. Each query has at least 8 terms. The average query length is 10.25 terms. We use long queries because the paper focuses on the term dependency in search. For each query, the associated documents are labeled by several annotators. In total, there are 6,271 labeled query-document pairs. On average each query has 26.68 documents labeled.

We tested the accuracies of ranking models and obtained the results in Fig. 4. We can observe an overall trend that the kernel based ranking models perform better than the bag-of-words models and the MRF model. Specifically, the kernel-based models outperform their counterparts (for example, BM25 Kernel works better than BM25, etc). There are always kernel-based models working better than MRF in different settings.

In the experiments, we tuned two parameters λ_1 and λ_2 for each kernel function. We investigated the sensitivity of the performances with respect to the two parameters. Specifically, we changed the values of a parameter and fixed the other at its optimal value (the optimal values are $\lambda_1 = 0.4$ and $\lambda_2 = 0.1$). The performances in terms of MAP with respect to parameter values are reported in Fig. 5.

The KL kernel has another parameter μ. We also carried out experiments to investigate how this parameter impacts the performances. We changed the values of the parameter and observed the model's performances in terms of MAP. Fig. 6 shows the performance curve. We can see that there is a flat region near the optimal value ($\mu = 4.0$), which means the model's performances are not sensitive to μ.

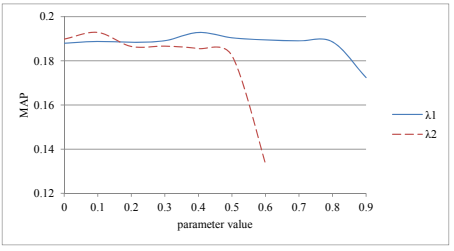

Fig. 5. Ranking accuracies of KL Kernel w.r.t. λ_1 and λ_2

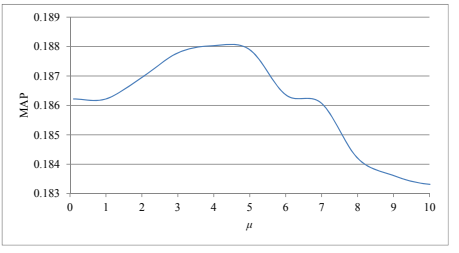

Fig. 6. Ranking accuracies of KL Kernel w.r.t. μ

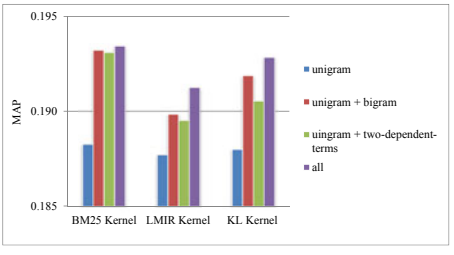

Fig. 7. BM25 Kernel, LMIR Kernel, and KL Kernel with different term dependencies

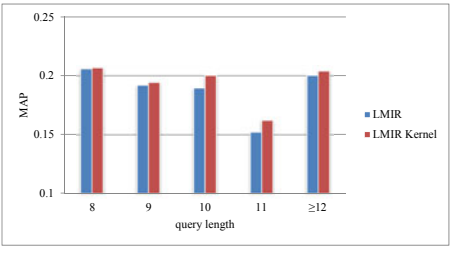

Fig. 8. Ranking accuracies of KL-Unigram and the KL Kernel by query lengths

We conducted experiments to verify how the use of term dependency information can improve ranking accuracy. Specifically, we tested the performances of BM25 Kernel (and also LMIR Kernel and KL Kernel) with unigram only, with unigram plus bigram, with unigram plus two-dependent-terms, and with unigram plus bigram and two-dependent-terms (denoted as "all"), in terms of MAP and NDCG. Fig. 7 shows the results in terms of MAP. We can see that bigram and two-dependent-terms really help to improve the ranking accuracies in terms of MAP (we observe the same tendency for the results in terms of NDCG). If they are combined together, the ranking accuracies can be further improved. This indicates that the kernel-based model really has the ability of incorporating different types of dependency information.

We also conducted experiments to investigate how KL Kernel can improve the ranking accuracies, especially on long queries. We calculated MAP (and NDCG) for KL-Unigram and KL Kernel on each query, and grouped the queries by length. Fig. 8 show the results. From the results, we can see that KL Kernel improve MAP scores when queries become longer. This indicates that the kernel-based models are more effective when queries become longer. The same tendency can be found for methods of BM25 Kernel, LMIR Kernel and for measures of NDCG.

5.2 Experiments on the OHSUMED and the AP Datasets

In this experiment, we used the OHSUMED [7] and TREC AP datasets to test the performances of the ranking models. The OHSUMED dataset consists of 348,566

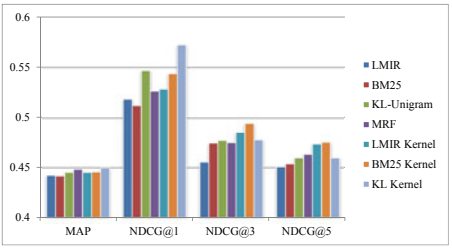

Fig. 9. Ranking accuracies on the OHSUMED

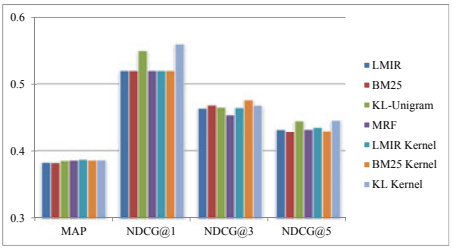

Fig. 10. Ranking accuracies on the AP data

documents and 106 queries. There are in total 16,140 query-document pairs upon which relevance judgments are made. The relevance judgments are either "d" (*definitely relevant*), "p" (*possibly relevant*), or "n"(*not relevant*). The AP collection contains 158,240 articles of Associated Press in 1988 and 1990. 50 queries are selected from the TREC topics (No.51 ~ No.100). Each query has a number of documents associated and they are labeled as "relevant" or "irrelevant". In total, there are 8,933 query-document pairs labeled. On average, each query has 178.66 labeled documents.

We used the Lemur toolkit[1] as our experiment platform. The datasets were indexed and queries were processed with Lemur. The experimental results are shown in Fig. 9 and Fig. 10. Again, we can see that the kernel-based models work better than the bag-of-words models and MRF. However, the improvements are smaller than those on the web search data. This is probably because the query length of data is short (on average 4.96 terms per query on OHSUMED and 3.22 terms per query on AP).

6 Conclusion

In this paper, we have studied relevance ranking models, particularly, those using term dependencies. Our work is new and unique in that we employ the kernel technique in statistical learning in the analysis and construction of ranking models. We have defined three kernel based ranking models: BM25 Kernel, LMIR Kernel, and KL Kernel. The former two are generalization of BM25 and LMIR, and the latter is a new model.

The general ranking model employed in the our approach has (1) rich representability for relevance ranking particularly for that using term dependencies (different types of term dependencies are summarized in the paper, and the kernel based ranking model can naturally represent them.), (2) strong connections with existing models, (3) solid theoretical background, and (4) efficient calculation.

Experimental results on web search data and TREC data show that (1) the kernel based ranking models outperform the conventional bag-of-words models and the MRF model using the same information, (2) the kernel functions can really effectively incorporate different types of term dependency information (from unigram to bigram and two-dependent-terms), (3) the kernel functions work particularly well for long queries.

There are several issues which need further investigations as future work. (1) The relationship between the proposed approach needs more studies. (2) Kernel functions

[1] http://www.lemurproject.org/

can be learned in machine learning. A future step would be to learn the ranking model by plugging it into some kernel machines.

References

1. Bai, J., Chang, Y., Cui, H., Zheng, Z., Sun, G., Li, X.: Investigation of partial query proximity in web search. In: Proc. of 17th WWW, pp. 1183–1184 (2008)
2. Büttcher, S., Clarke, C.L.A., Lushman, B.: Term proximity scoring for ad-hoc retrieval on very large text collections. In: Proc. of 29th SIGIR. pp. 621–622 (2006)
3. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. J. of American Society for Information Science 41, 391–407 (1990)
4. Gao, J., Nie, J.Y., Wu, G., Cao, G.: Dependence language model for information retrieval. In: Proceedings of the 27th Annual International ACM SIGIR Conference, pp. 170–177 (2004)
5. Haussler, D.: Convolution kernels on discrete structures. Tech. rep., Dept. of Computer Science, University of California at Santa Cruz (1999)
6. Hawking, D., Thistlewaite, P.B.: Proximity operators - so near and yet so far. In: Proceedings of the 4th Text Retrieval Conference (1995)
7. Hersh, W., Buckley, C., Leone, T.J., Hickam, D.: Ohsumed: an interactive retrieval evaluation and new large test collection for research. In: Proc. of 17th SIGIR, pp. 192–201 (1994)
8. Keen, E.M.: Some aspects of proximity searching in text retrieval systems. J. Inf. Sci. 18(2), 89–98 (1992)
9. Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., Watkins, C.: Text classification using string kernels. J. Mach. Learn. Res. 2, 419–444 (2002)
10. Martins, A.: String kernels and similarity measures for information retrieval. Tech. rep., Priberam, Lisbon, Portugal (2006)
11. Metzler, D., Croft, W.B.: A markov random field model for term dependencies. In: Proceedings of the 28th Annual International ACM SIGIR Conference, pp. 472–479 (2005)
12. Moreno, P.J., Ho, P.P., Vasconcelos, N.: A kullback-leibler divergence based kernel for svm classification in multimedia applications. In: NIPS 16, MIT Press, Cambridge (2003)
13. Na, S.H., Kim, J., Kang, I.S., Lee, J.H.: Exploiting proximity feature in bigram language model for information retrieval. In: Proc. of 31st SIGIR, pp. 821–822 (2008)
14. Ong, C.S., Smola, A.J., Williamson, R.C.: Learning the kernel with hyperkernels. J. Mach. Learn. Res. 6, 1043–1071 (2005)
15. Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M., Gatford, M.: Okapi at trec-3. In: Proceedings of the 3rd Text REtrieval Conference (1994)
16. Sahami, M., Heilman, T.D.: A web-based kernel function for measuring the similarity of short text snippets. In: Proc. of 15th WWW, pp. 377–386 (2006)
17. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Commun. ACM 18(11), 613–620 (1975)
18. Sonnenburg, S., Rätsch, G., Schäfer, C., Schölkopf, B.: Large scale multiple kernel learning. J. Mach. Learn. Res. 7, 1531–1565 (2006)
19. Tao, T., Zhai, C.: An exploration of proximity measures in information retrieval. In: Proc. of 30th SIGIR, pp. 295–302 (2007)
20. Tsuda, K.: Support vector classifier with asymmetric kernel functions. In: European Symposium on Artificial Neural Networks, pp. 183–188 (1999)
21. Xie, Y., Raghavan, V.V.: Language-modeling kernel based approach for information retrieval. J. Am. Soc. Inf. Sci. Technol. 58(14), 2353–2365 (2007)
22. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to information retrieval. ACM Trans. Inf. Syst. 22(2), 179–214 (2004)