A Markov Random Field Model for Term Dependencies

Donald Metzler
metzler@cs.umass.edu

W. Bruce Croft
croft@cs.umass.edu

Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, MA 01003

ABSTRACT

This paper develops a general, formal framework for modeling term dependencies via Markov random fields. The model allows for arbitrary text features to be incorporated as evidence. In particular, we make use of features based on occurrences of single terms, ordered phrases, and unordered phrases. We explore full independence, sequential dependence, and full dependence variants of the model. A novel approach is developed to train the model that directly maximizes the mean average precision rather than maximizing the likelihood of the training data. Ad hoc retrieval experiments are presented on several newswire and web collections, including the GOV2 collection used at the TREC 2004 Terabyte Track. The results show significant improvements are possible by modeling dependencies, especially on the larger web collections.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Algorithms, Experimentation, Theory

Keywords

Information retrieval, term dependence, phrases, Markov random fields

1. INTRODUCTION

There is a rich history of statistical models for information retrieval, including the binary independence model (BIM), language modeling [16], inference network model [23], and the divergence from randomness model [1], amongst others [4]. It is well known that dependencies exist between terms in a collection of text. For example, within a *SIGIR* proceedings, occurrences of certain pairs of terms are correlated, such as *information* and *retrieval*. The fact that either one occurs provides strong evidence that the other is also likely to occur. Unfortunately, estimating statistical models for general term dependencies is infeasible, due to data sparsity. For this reason, most retrieval models assume some form of independence exists between terms. Some researchers even suggest modeling term dependencies is unnecessary as long as a good term weighting function is used [19].

Most work on modeling term dependencies in the past has focused on phrases/proximity [2, 3] or term co-occurrences [24]. Most of these models only consider dependencies between pairs of terms. In [3], Fagan examines how to identify and use non-syntactic (statistical) phrases. He identifies phrases using factors such as the number of times the phrase occurs in the collection and the proximity of the phrase terms. His results suggest no single method of phrase identification consistently yields improvements in retrieval effectiveness across a range of collections. For several collections, significant improvements in effectiveness are achieved when phrases are defined as any two terms within a query or document with unlimited proximity. That is, any two terms that co-occurred within a query or document were considered a phrase. However, for other collections, this definition proved to yield marginal or negative improvements. The results presented by Croft et. al. in [2] on the CACM collection suggest similar results, where phrases formed with a probabilistic AND operator slightly outperformed proximity phrases. Term co-occurrence information also plays an important role in the tree dependence model, which attempts to incorporate dependencies between terms in the BIM [24]. The model treats each term as a node in a graph and constructs a maximum spanning tree over the nodes, where the weight between a pair of terms (nodes) is the expected mutual information measure (EMIM) between them.

Other models have been proposed to capture dependencies between more than two terms, such as the Bahadur Lazarsfeld expansion (BLE) [17], which is an exact method of modeling dependencies between all terms, and the Generalized Dependence Model that generalizes both the tree dependence model and the BLE expansion [25]. Despite the more complex nature of these models, they have been shown to yield little or no improvements in effectiveness.

Several recent studies have examined term dependence models for the language modeling framework [5, 15]. These models are inspired by the tree dependence model and again only consider dependencies among pairs of terms. The model presented by Gao et. al in [5] showed consistent improve-

SIGIR'05, August 15–19, 2005, Salvador, Brazil.

ments over a baseline query likelihood system on a number of TREC collections. Unfortunately, the model requires computing a link structure for each query, which is not straightforward.

Recently, a paper by Mishne and de Rijke discussed the use of proximity information to improve web retrieval [12]. Our model, developed without knowledge of this work, shares many closely related insights. Despite the high level similarity, the details of the models differ greatly, with our model allowing more general query dependencies and features to be considered in a more formally well-grounded framework.

The consistent improvements presented by Gao et. al. are in stark contrast to the results reported historically for other term dependence models. It is our hypothesis that the poor and inconsistent results achieved in the past are caused by two factors. First, most dependence models are built upon the BIM. Therefore, term dependencies must be estimated in both the relevant and non-relevant classes, where there is often a very small or non-existent sample of relevant/non-relevant documents available to estimate the model parameters from. Second, the document collections used in past experiments, such as CACM and INSPEC consist of a very small number of short documents. There is very little hope of accurately modeling term dependencies when most pairs of terms only occur a handful of times, if at all. Essentially, the models themselves are not deficient. Instead, they simply lacked sufficient data for proper parameter estimation and evaluation.

Thus, we formulate the following two hypotheses: 1) dependence models will be more effective for larger collections than smaller collections, and 2) incorporating several types of evidence (features) into a dependence model will further improve effectiveness. Our first hypothesis is based on the fact that larger collections are noisier despite the fact they contain more information. As a result, independent query terms will match many irrelevant documents. If matching is instead done against more specific patterns of dependent query terms, then many of the noisy, irrelevant documents will be filtered out. In addition, we feel that considering various combinations of term features can yield further improvements over the previously researched methods, as it allows us to abstract the notions of dependence and co-occurrence.

The rest of this paper is presented as follows. In Section 2 we describe a general Markov random field retrieval model that captures various kinds of term dependencies. We also show how arbitrary features of the text can be used to generalize the idea of co-occurrence, how the model can be used to express three different dependence assumptions, and how to automatically set the parameters from training data. In Section 3 we describe the results of ad hoc retrieval experiments done using our model on two newswire and two web collections. We show that dependence models can significantly improve effectiveness, especially on the larger web collections. Finally, in Section 4 we summarize the results and propose future research directions.

2. MODEL

In this section we describe a Markov random field approach to modeling term dependencies. Markov random fields (MRF), also called undirected graphical models, are commonly used in the statistical machine learning domain to succinctly model joint distributions. In this paper we use MRFs to model the joint distribution $P_\Lambda(Q, D)$ over queries Q and documents D, parameterized by Λ. We model the joint using this formalism because we feel it is illustrative and provides an intuitive, general understanding of different forms of term dependence.

Much like the language modeling framework, our model does not explicitly include a relevance variable. Instead, we assume there is some underlying joint distribution over queries and documents ($P_\Lambda(Q, D)$). Given a set of query and document pairs, Λ must be estimated according to some criteria. In the case of modeling relevance or usefulness, which we wish to do here, imagine there exists a list of query and document pairs. Suppose elements of this list are gathered from many (infinite) users of some information retrieval system who theoretically examine every pair of queries and documents. If a user finds document D relevant to query Q, then the pair (Q, D) is added to the list. This list can be thought of as a sample from some relevance distribution from which Λ can be estimated. We feel that ranking documents by $P_\Lambda(D|Q)$ upholds the original spirit of the Probability Ranking Principle (PRP) [18] under the assumption that the issuer of query Q is likely to agree with the relevance assessments of a majority of users. One could also estimate such a model for non-relevance or user-specific relevance, amongst others. This view of relevance is similar to Lavrenko's Generative Relevance Hypothesis [8].

2.1 Overview

A Markov random field is constructed from a graph G. The nodes in the graph represent random variables, and the edges define the independence semantics between the random variables. In particular, a random variable in the graph is independent of its non-neighbors given observed values for its neighbors. Therefore, different edge configurations impose different independence assumptions. In this model, we assume G consists of query nodes q_i and a document node D, such as the graphs in Figure 1. Then, the joint distribution over the random variables in G is defined by:

$$P_\Lambda(Q, D) = \frac{1}{Z_\Lambda} \prod_{c \in C(G)} \psi(c; \Lambda)$$

where $Q = q_1 \dots q_n$, $C(G)$ is the set of cliques in G, each $\psi(\cdot; \Lambda)$ is a non-negative *potential function* over clique configurations parameterized by Λ and $Z_\Lambda = \sum_{Q,D} \prod_{c \in C(G)} \psi(c; \Lambda)$ normalizes the distribution. Note that it is generally infeasible to compute Z_Λ because of the exponential number of terms in the summation. The joint distribution is uniquely defined by the graph G, the potential functions ψ, and the parameter Λ.

For ranking purposes we compute the posterior:

$$\begin{aligned} P_\Lambda(D|Q) &= \frac{P_\Lambda(Q, D)}{P_\Lambda(Q)} \\ &\overset{rank}{=} \log P_\Lambda(Q, D) - \log P_\Lambda(Q) \\ &\overset{rank}{=} \sum_{c \in C(G)} \log \psi(c; \Lambda) \end{aligned}$$

which can be computed efficiently for reasonable graphs.

As noted above, all potential functions must be non-negative, and are most commonly parameterized as:

$$\psi(c; \Lambda) = \exp[\lambda_c f(c)]$$

where $f(c)$ is some real-valued *feature function* over clique values and λ_c is the weight given to that particular feature

function. Substituting this back into our ranking function, we end up with the following ranking function

$$P_\Lambda(D|Q) \stackrel{rank}{=} \sum_{c \in C(G)} \lambda_c f(c) \quad (1)$$

To utilize the model, the following steps must be taken for each query Q: 1) construct a graph representing the query term dependencies to model, 2) define a set of potential functions over the cliques of this graph, 3) rank documents in descending order of $P_\Lambda(D|Q)$.

Given its general form, it is of little surprise that the MRF model can easily emulate most of the popular retrieval and dependence models by following the steps above, such as unigram, bigram, and biterm language modeling [16, 20, 21], the BIM and the associated methods by which term dependence is modeled [17, 25, 24].

2.2 Variants

We now describe and analyze three variants of the MRF model, each with different underlying dependence assumptions. The three variants are *full independence* (FI), *sequential dependence* (SD), and *full dependence* (FD). Figure 1 shows graphical model representations of each.

The full independence variant makes the assumption that query terms q_i are independent given some document D. That is, the likelihood of query term q_i occurring is not affected by the occurrence of any other query term, or more succinctly, $P(q_i|D, q_{j \neq i}) = P(q_i|D)$. This assumption underlies many retrieval models.

As its name implies, the sequential dependence variant assumes a dependence between neighboring query terms. Formally, this assumption states that $P(q_i|D, q_j) = P(q_i|D)$ only for nodes q_j that are not adjacent to q_i. Models of this form are capable of emulating bigram and biterm language models [20, 21].

The last variant we consider is the full dependence variant. In this variant we assume all query terms are in some way dependent on each other. Graphically, a query of length n translates into the complete graph K_{n+1}, which includes edges from each query node to the document node D, as well. This model is an attempt to capture longer range dependencies than the sequential dependence variant. If such a model can accurately be estimated, it should be expected to perform at least as well as a model that ignores term dependence.

We must stress that we are not trying to model the *exact* joint $P_\Lambda(Q, D)$ here. Instead, we are attempting to approximate the distribution using a generalized exponential form. How close our approximation is to reality depends wholly on the choice of potential functions and how the parameter Λ is set. In the next two subsections, we explain some possible approaches to solving these problems.

2.3 Potential Functions

As we just explained, the potential functions ψ play a very important role in how accurate our approximation of the true joint distribution is. These functions can be thought of as compatibility functions. Therefore, a good potential function assigns high values to the clique settings that are the most "compatible" with each other under the given distribution. As an example, consider a document D on the topic of *information retrieval*. Using the sequential dependence variant, we would expect $\psi(information, retrieval, D) > \psi(information, assurance, D)$, as the terms *information* and *retrieval* are much more "compatible" with the topicality of document D than the terms *information* and *assurance*.

Since documents are ranked by Equation 1, it is also important that the potential functions can be computed efficiently. With this in mind, we opt to use potential functions that can be calculated using Indri[1], our new scalable search engine that combines language modeling and the inference network framework [10, 11, 23].

Based on these criteria and previous research on phrases and term dependence [2, 3] we focus on three types of potential functions. These potential functions attempt to abstract the idea of term co-occurrence. As we see from Equation 1, each potential is defined by a feature function and weight. In the remainder of this section we specify the features we use, and in the next section we show how to automatically set the weights.

Since potentials are defined over cliques in the graph, we now proceed to enumerate all of the possible ways graph cliques are formed in our model and how potential function(s) are defined for each. The simplest type of clique that can appear in our graph is a 2-clique consisting of an edge between a query term q_i and the document D. A potential function over such a clique should measure how well, or how likely query term q_i describes the document. In keeping with simple to compute measures, we define this potential as:

$$\begin{aligned} \psi_T(c) &= \lambda_T \log P(q_i|D) \\ &= \lambda_T \log \left[(1 - \alpha_D) \frac{tf_{q_i,D}}{|D|} + \alpha_D \frac{cf_{q_i}}{|C|} \right] \end{aligned}$$

where $P(q_i|D)$ is simply a smoothed language modeling estimate. Here, $tf_{w,D}$ is the number of times term w occurs in document D, $|D|$ is the total number of terms in document D, cf_w is the number of times term w occurs in the entire collection, and $|C|$ is the length of the collection. Finally, α_D acts as a smoothing parameter [26]. This potential makes the assumption that the more likely a term is given a document's language model, the more "compatible" the two random variables q_i and D are.

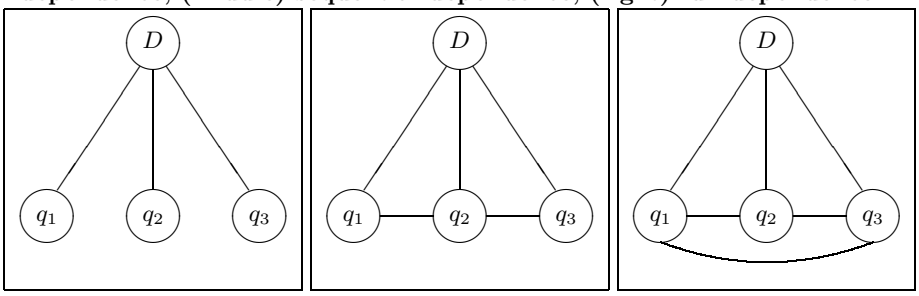
Next, we consider cliques that contain two or more query terms. For such cliques there are two possible cases, either all of the query terms within the clique appear contiguously in the query or they do not. The fact that query terms appear contiguously within a query provides different (stronger) evidence about the information need than a set of non-contiguous query terms. For example, in the query *train station security measures* (TREC topic 711), if any of the subphrases, *train station*, *train station security*, *station security measures*, or *security measures* appear in a document then there is strong evidence in favor of relevance. Therefore, for every clique that contains a contiguous set of two or more terms $q_i, \ldots, q_{i+k}$ and the document node D we apply the following "ordered" potential function:

$$\begin{aligned} \psi_O(c) &= \lambda_O \log P(\#1(q_i, \ldots, q_{i+k})|D) \\ &= \lambda_O \log \left[(1 - \alpha_D) \frac{tf_{\#1(q_i \ldots q_{i+k}),D}}{|D|} + \alpha_D \frac{cf_{\#1(q_i \ldots q_{i+k})}}{|C|} \right] \end{aligned}$$

where $tf_{\#1(q_i \ldots q_{i+k}),D}$ denotes the number of times the exact phrase $q_i \ldots q_{i+k}$ occurs in document D, with an analogous

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

Figure 1: Example Markov random field model for three query terms under various independence assumptions. (left) full independence, (middle) sequential dependence, (right) full dependence.



definition for $cf_{\#1(q_i \ldots q_{i+k})}$. For more information on estimating so-called *language feature models* see [11].

Although the occurrence of contiguous sets of query terms provide strong evidence of relevance, it is also the case that the occurrence of non-contiguous sets of query terms can provide valuable evidence. However, since the query terms are not contiguous we do not expect them to appear in order within relevant documents. Rather, we only expect the terms to appear ordered or unordered within a given proximity of each other. In the previous example, documents containing the terms *train* and *security* within some short proximity of one another also provide additional evidence towards relevance. This issue has been explored in the past by a number of researchers [2, 3]. For our purposes, we construct an "unordered" potential function over cliques that consist of sets of two or more query terms $q_i, \ldots, q_j$ and the document node D. Such potential functions have the following form:

$$\begin{aligned}\psi_U(c) &= \lambda_U \log P(\#\text{uwN}(q_i, \ldots, q_j)|D) \\ &= \lambda_U \log \left[(1-\alpha_D) \frac{tf_{\#\text{uwN}(q_i \ldots q_j),D}}{|D|} + \alpha_D \frac{cf_{\#\text{uwN}(q_i \ldots q_j)}}{|C|} \right]\end{aligned}$$

where $tf_{\#\text{uwN}(q_i \ldots q_j),D}$ is the number of times the terms $q_i, \ldots q_j$ appear ordered or unordered within a window N terms. In our experiments we will explore various settings of N to study the impact it has on retrieval effectiveness. It should also be noted that not only do we add a potential function of this form for non-contiguous sets of two or more query terms, but also for contiguous sets of two or more query terms. Therefore, for cliques consisting of contiguous sets of two or more query terms and the document node D we define the potential function to be the product $\psi_O(c)\psi_U(c)$, which itself is a valid potential function.

Using these potential functions we derive the following specific ranking function:

$$\begin{aligned}P_\Lambda(D|Q) &\overset{rank}{=} \sum_{c \in C(G)} \lambda_c f(c) \\ &= \sum_{c \in T} \lambda_T f_T(c) + \sum_{c \in O} \lambda_O f_O(c) + \sum_{c \in O \cup U} \lambda_U f_U(c)\end{aligned}$$

where T is defined to be the set of 2-cliques involving a query term and a document D, O is the set of cliques containing the document node and two or more query terms that appear contiguously within the query, and U is the set of cliques containing the document node and two or more query terms appearing non-contiguously within the query. For any clique c that does not contain the document node we assume that $\psi(c) = 1$ for all settings of the clique, which has no impact on ranking. One may wish to define a potential over the singleton document node, which could act as a form of document prior.

Finally, the feature functions are summarized in Table 1. The table also provides a summary of the Indri query language expressions that can be constructed to easily evaluate each.

2.4 Training

Given our parameterized joint distribution and a set of potential functions, the final step is to set the parameter values $(\lambda_T, \lambda_O, \lambda_U)$. Most methods for training generative models require training data that is assumed to be independent and identically distributed samples from the underlying distribution. From this data, the model parameters are estimated in various ways. Two standard approaches are maximum likelihood and maximum *a posteriori* estimation. Other methods exist for training MRFs, as well [22].

However, there are two hindrances that cause us to abandon traditional training methodologies. First, the event space $Q \times D$ is extremely large or even infinite depending on how it is defined. Generally, the only training data available is a set of TREC relevance judgments for a set of queries. The documents found to be relevant for a query can then be assumed to be samples from this underlying relevance distribution. However, this sample is extremely small compared to the event space. For this reason, it is highly unlikely that a maximum likelihood estimate from such a sample would yield an accurate estimate to the true distribution.

Even if a large sample from the underlying distribution existed, it would still be very difficult to compute a maximum likelihood estimate because of the normalization factor Z_Λ, which is infeasible to compute both in our model and in general. Methods exist for efficiently approximating Z_Λ, but none appear to be directly applicable to a problem of our size.

Many IR techniques that involve automatically setting parameters from training data make use of maximum likelihood or related estimation techniques. Even though these techniques are formally motivated, they often do not maximize the correct objective function. Simply because the likelihood of generating the training data is maximized does not mean the evaluation metric under consideration, such as mean average precision, is also maximized. This is true because models serve as rough approximations to complex distributions, and therefore the *likelihood surface* is unlikely to correlate with the *metric surface*. This has been shown to be true experimentally and is known as metric divergence [13].

Feature	Type	Indri Expression	Value
$f_T(q_i, D)$	Term	q_i	$\log\left[(1-\alpha_D)\frac{tf_{q_i,D}}{\|D\|} + \alpha_D \frac{cf_{q_i}}{\|C\|}\right]$
$f_O(q_i, q_{i+1} \ldots, q_{i+k}, D)$	Ordered Phrase	$\#1(q_i q_{i+1} \ldots q_{i+k})$	$\log\left[(1-\alpha_D)\frac{tf_{\#1(q_i \ldots q_{i+k}),D}}{\|D\|} + \alpha_D \frac{cf_{\#1(q_i \ldots q_{i+k})}}{\|C\|}\right]$
$f_U(q_i, ..., q_j, D)$	Unordered Phrase	$\#\text{uwN}(q_i \ldots q_j)$	$\log\left[(1-\alpha_D)\frac{tf_{\#\text{uwN}(q_i \ldots q_j),D}}{\|D\|} + \alpha_D \frac{cf_{\#\text{uwN}(q_i \ldots q_j)}}{\|C\|}\right]$

Table 1: Summary of feature functions and how each is represented in the Indri query language.

Therefore, directly maximizing the retrieval metric under consideration may prove to be more useful than focusing on "goodness of fit". Even though it may be infeasible in general, it is somewhat surprising there has been little research into directly maximizing the metric in question for probabilistic models. We note that there has been some work done into finding optimal rankings using ranking SVMs [6], which is more in the spirit of directly optimizing the metric of interest, rather than hoping to do so indirectly.

Therefore, we choose to train the model by directly maximizing mean average precision. Since our model only has three parameters, it is possible to do a parameter sweep to find the optimal parameters. However, such a parameter sweep can be computationally intensive. A few observations allow us to devise a relatively efficient training procedure. First, the ranking function is invariant to parameter scale. That is, for some fixed K, rank order will be preserved if we modify the parameters such that $\hat{\lambda}_T = K\lambda_T$, $\hat{\lambda}_O = K\lambda_O$, and $\hat{\lambda}_U = K\lambda_U$, since the constant can always be factored out. Therefore, we can enforce the constraint:

$$\lambda_T + \lambda_O + \lambda_U \quad = \quad 1$$

Plotting mean average precision surfaces for several collections subject to this constraint leads to the following observations: 1) the interesting part of the surface exists for non-negative parameter values, 2) over this interesting range of parameter values, the surface it is always concave or very close to concave, and 3) the surface always has the same general form with the maximum being achieved nearly always around the same point in parameter space. An example mean average precision surface for the GOV2 collection using the full dependence model plotted over the simplex $\lambda_T + \lambda_O + \lambda_U = 1$ is shown in Figure 2.

Therefore, a simple coordinate-level hill climbing search is used to optimize mean average precision by starting at the full independence parameter setting ($\lambda_T = 1, \lambda_O = \lambda_U = 0$). Since the surface is (nearly) concave we are likely to find the global maximum. More details can be found in [9].

Convergence speed is only bounded by the underlying search engine's efficiency at evaluating the training set queries. Since training requires the same queries to be evaluated repeatedly, most statistics including feature function values can be cached allowing for further speed improvements.

3. EXPERIMENTAL RESULTS

In this section we describe experiments using the three model variants. Our aim is to analyze and compare the retrieval effectiveness of each variant across collections of varying size and type. We make use of the Associated Press and Wall Street Journal subcollections of TREC, which are small homogeneous collections, and two web collections, WT10g and GOV2, which are considerably larger and less homogeneous.

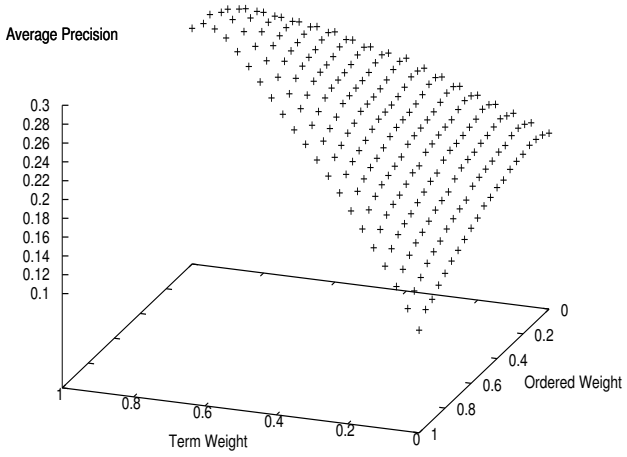


Figure 2: Mean average precision surface over simplex of parameter values for GOV2 using FD model.

Name	Description	Size	# Docs	Topics
WSJ	Wall St. Journal '87–'92	510 MB	173,252	50–200
AP	Associated Press '88–'90	730 MB	242,918	50–200
WT10g	TREC Web collection	11 GB	1,692,096	451–550
GOV2	2004 crawl of .gov domain	427 GB	25,205,179	701–750

Table 2: Overview of TREC collections and topics.

To put it all into perspective, the CACM collection used in previous experiments is 1.4MB and consists of 3204 documents, whereas the Wall Street Journal collection is 512MB and consists of 173,252 documents, and the GOV2 collection, a crawl of the entire .gov web domain that was used in the TREC 2004 Terabyte Track, is 427GB and consists of approximately 25 million documents. Table 2 provides a summary of the TREC collections and topics used.

All experiments make use of the Indri search engine [11]. Documents are stemmed using the Porter stemmer, but not stopped at index time. Instead, stopping is done at query time using a standard list of 421 stopwords. Only the title portion of the TREC topics are considered. The newswire queries are typically longer than the short, keyword-based web queries.

	AP	WSJ	WT10g	GOV2
AvgP	0.1775	0.2592	0.2032	0.2502
P @ 10	0.2912	0.4327	0.2866	0.4837
μ	3000	3500	4000	4000

Table 3: Full independence variant results.

3.1 Full Independence

For a fixed query, the full independence variant takes the form of the model given in Figure 1(left). Since the graph only consists of cliques containing a single query term and the document node, all of the cliques are only members of the set T, and therefore we set $\lambda_O = \lambda_U = 0$, which implies that $\lambda_T = 1$. Therefore, our ranking function is of the form:

$$\begin{aligned} P_\Lambda(D|Q) &\overset{rank}{=} \sum_{c \in T} f_T(c) \\ &= \sum_{c \in T} \log \left[(1-\alpha_D) \frac{tf_{q_i,D}}{|D|} + \alpha_D \frac{cf_{q_i}}{|C|} \right] \\ &\overset{rank}{=} \prod_{c \in T} \left[(1-\alpha_D) \frac{tf_{q_i,D}}{|D|} + \alpha_D \frac{cf_{q_i}}{|C|} \right] \end{aligned}$$

which shows the full independence model using the given features reduces to the language modeling approach to information retrieval under the assumption of a uniform document prior (i.e. $P(D|Q) \propto P(Q|D)P(D)$). We assume that $\alpha_D = \frac{\mu}{\mu+|D|}$, which is the form of the well-known Dirichlet smoothing [26].

Table 3 shows the results for the full independence model. In the table, AvgP refers to mean average precision, P@10 is precision at 10 ranked documents, and μ denotes the smoothing parameter used. The smoothing parameters were tuned to maximize mean average precision. The values found here are used throughout the remainder of the experiments. The full independence results provide a baseline from which to compare the sequential and full dependence variants of the model.

3.2 Sequential Dependence

The graphical form of the sequential dependence variant is shown in Figure 1(center). This variant models dependencies between adjacent query terms. Models of this form have cliques in T, O, and U, as defined previously, and therefore make use of the three feature functions. However, the unordered feature function, f_U, has a free parameter N that allows the size of the unordered window (scope of proximity) to vary. Previous research has suggested various choices for the window size. Fagan shows that the best choice of N varies across collections [3]. Optimal values found included setting N to 2, the length of a sentence, or "unlimited" (matches any co-occurrences of the terms within a document). Croft et. al. showed improvements could be achieved with passage-sized windows of 50 terms [2]. Therefore, we explore window sizes of 2, 50, sentence, and "unlimited" to see what impact they have on effectiveness. Instead of segmenting sentences at index time, we observe that the average length of an English sentence is 8-15 terms, and choose a window size of 8 terms to model sentence-level proximity.

The results are given in Table 4. For each window size, the model parameters are trained on the same collection as they are tested to provide an idea of the best performance possible. The results show very little difference across the various window sizes. However, for the AP, WT10g, and GOV2 collection the sentence-sized windows performed the best. For the WSJ collection, $N = 2$ performed the best. The only collection where mean average precision varies noticeably is the GOV2 collection. These results suggest that a limited scope of proximity (2-50 terms) performs reasonably, but can be approximated rather well by an "unlimited" scope, which reaffirms past research into dependence models based on co-occurrences. However, it appears as though smaller scopes of proximity may provide better performance for larger collections, as evidenced by the GOV2 results. Finally, for every collection and every scope of proximity, the sequential dependence variant outperforms the full independence variant.

Length	AP	WSJ	WT10g	GOV2
2	0.1860	0.2776	0.2148	0.2697
8	0.1867	0.2763	0.2167	0.2832
50	0.1858	0.2766	0.2154	0.2817
Unlimited	0.1857	0.2759	0.2138	0.2714

Table 4: Mean average precision for various unordered window lengths with the sequential dependence variant.

3.3 Full Dependence

The full dependence model, shown in Figure 1(right), attempts to incorporate dependencies between every subset of query terms, and thus also consists of cliques in T, O and U. The number of cliques is exponential in the number of query terms which limits the application of this variant to shorter queries. For this variant, we adaptively set the parameter N in the feature function f_U to be four times the number of query terms in the clique c.

In the previous experiments we tested on the training set, which allowed us to determine an upper bound on performance, but it is not a realistic setting. Therefore, for the full dependence model, we train on each collection and then use the parameter values found to test on the other collections. In addition, we analyze the impact ordered and unordered window feature functions have on effectiveness. Results for models trained using terms and ordered features, terms and unordered features, and terms, ordered, and unordered features are given in Table 5.

For the AP collection, there is very little difference between using ordered and unordered features. However, there is a marginal increase when both ordered and unordered features are used together. The results for the WSJ collection are different. For that collection, the ordered features produce a clear improvement over the unordered features, but there is very little difference between using ordered features and the combination of ordered and unordered. The results for the two web collections, WT10g and GOV2, are similar. In both, unordered features perform better better than ordered features, but the combination of both ordered and unordered features led to noticeable improvements in mean average precision.

From these results we can conclude that strict matching via ordered window features is more important for the smaller newswire collections. This may be due to the homogeneous, clean nature of the documents, where an ordered window match is likely to be a high quality match instead of noise. For the web collections, the opposite is true. Here, the fuzzy unordered window matches provide

	Term + Ordered				Term + Unordered				Term + Ordered + Unordered			
Train \ Test	AP	WSJ	WT10g	GOV2	AP	WSJ	WT10g	GOV2	AP	WSJ	WT10g	GOV2
AP	0.1847	0.2718	0.2180	0.2669	0.1840	0.2674	0.2179	0.2754	0.1866	0.2716	0.2226	0.2839
WSJ	0.1842	0.2733	0.2167	0.2613	0.1840	0.2674	0.2179	0.2754	0.1841	0.2738	0.2195	0.2694
WT10g	0.1847	0.2718	0.2180	0.2669	0.1838	0.2674	0.2189	0.2783	0.1865	0.2719	0.2231	0.2783
GOV2	0.1840	0.2705	0.2150	0.2675	0.1838	0.2674	0.2189	0.2783	0.1852	0.2709	0.2201	0.2844

Table 5: Mean average precision using the full dependence variant over different combinations of term, ordered, and unordered features.

better evidence. In these less homogeneous, noisy collections, an ordered window match is less likely to be a high quality match and more likely to be a noisy match. Instead, fuzzy matches are appropriate because they deal better with the noise inherent in web documents.

These results also suggest that parameters trained on any of the collections generalize well to other collections. This result is somewhat surprising; we expected parameters trained on newswire (web) data would generalize better to newswire (web) test data. However, this is not the case. It appears as though the parameters trained on any reasonable collection will generalize well, which allows one to use a single setting of the parameters across multiple collections. This may imply that the features used here only capture general aspects of the text. More domain-specific features may yield further improvements.

3.4 Summary of Results

We now briefly summarize and compare the results across the model variants. Table 6 gives mean average precision, precision at 10, and suggested model parameters for each variant. The results given use the optimal parameter values to allow a fair comparison. Both the sequential and full dependence variants significantly improve mean average precision over the full independence variant for all four collections. Therefore, modeling dependencies between terms can be done consistently and can result in significant improvements. We also note the considerable improvements on the WT10g and GOV2 collections. These improvements support our hypothesis that dependence models may yield larger improvements for large collections. As further evidence of the power of these models on large collections, we note that a slightly modified version of the full dependence variant of this model was the best automatic, title-only run at the 2004 TREC Terabyte Track [11]. Although not explored here, the P@10 results could likely be significantly improved by directly maximizing over the P@10 metric.

4. CONCLUSIONS

In this paper we develop a general term dependence model that can make use of arbitrary text features. Three variants of the model are described, where each captures different dependencies between query terms. The full independence variant assumes that query terms are independent. The sequential dependence variant assumes certain dependencies exist between adjacent query terms, which is akin to bigram and biterm language models [20, 21]. Finally, the full dependence model makes no independence assumptions and attempts to capture dependencies that exist between every subset of query terms.

Our results show that modeling dependencies can significantly improve retrieval effectiveness across a range of collections. In particular, the sequential dependence variant using term and ordered features is more effective on smaller, homogeneous collections with longer queries, whereas the full dependence variant is best for larger, less homogeneous collections with shorter queries. In all cases, however, the sequential dependence variant closely approximates the full dependence variant. This provides the ability to tradeoff effectiveness for efficiency.

We note that our model shares some similarities, such as the exponential form, with the maximum entropy model presented by Nallapati [14], but is fundamentally different. The maximum entropy model, like our model, can make use of arbitrary text features. However, since the model is discriminative, it can only be used to rank documents according to $P(R = 1|Q, D)$. Our model foregoes explicitly modeling relevance, but instead models the joint probability over queries and documents. Thus, with our model we can not only compute $P_\Lambda(D|Q)$ to rank documents by, but also $P_\Lambda(Q|D)$, which can be used for query expansion. For example, suppose D represents a document or a set of documents known to be relevant, then we can expand a query by $\hat{Q} = \text{argmax}\ P(Q|D)$. This is a possible direction of future work. In addition, we attempted to use our features within the maximum entropy model. We trained the model using maximum likelihood estimation. The trained models, when tested on other collections, performed significantly worse than the baseline full independence model, which again supports our case for training by directly maximizing the mean average precision. Finally, we note that our model can easily be reformulated as a conditional random field [7], although we feel such a formulation does not add anything to the model.

Other directions of possible future work include exploring a wider range of potential functions, applying the model to other retrieval tasks, exploring different training methods including the use of clickthrough data, and constructing the graph G in other ways. For example, one could compute the EMIM between all pairs of query terms and only choose to model dependencies between terms with a high EMIM value. Or, similarly, one could apply the link approach taken in [5] to determine the important dependencies to model for a given query.

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval and in part by Advanced Research and Development Activity and NSF grant #CCF-0205575. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor.

5. REFERENCES

[1] G. Amati and C. J. V. Rijsbergen. Probabilistic models of information retrieval based on measuring

	FI		SD		FD	
	AvgP	P@10	AvgP	P@10	AvgP	P@10
AP	0.1775	0.2912	0.1867* (+5.2%)	0.2980 (+2.3%)	0.1866* (+5.1%)	0.3068* (+5.4%)
WSJ	0.2592	0.4327	0.2776† (+7.1%)	0.4427 (+2.3%)	0.2738* (+5.6%)	0.4413 (+2.0%)
WT10g	0.2032	0.2866	0.2167* (+6.6%)	0.2948 (+2.9%)	0.2231** (+9.8%)	0.3031 (+5.8%)
GOV2	0.2502	0.4837	0.2832* (+13.2%)	0.5714* (+18.1%)	0.2844* (+13.7%)	0.5837* (+20.7%)
$(\lambda_T, \lambda_O, \lambda_U)$	(1.00, 0.00, 0.00)		(0.85, 0.10, 0.05)		(0.80, 0.10, 0.10)	

Table 6: Mean average precision and precision at 10 using optimal parameter settings for each model. Values in parenthesis denote percentage improvement over full independence (FI) model. The symbols indicate statistical significance ($p < 0.05$ with a one-tailed paired t-test), where * indicates a significant improvement over the FI variant, ** over both the FI and SD variants, and † over the FI and FD variants. Suggested parameter values for each variant are also given.

the divergence from randomness. *ACM Transactions on Information Systems*, 20(4):357–389, 2002.

[2] W. B. Croft, H. R. Turtle, and D. D. Lewis. The use of phrases and structured queries in information retrieval. In *Proc. 14th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 32–45, 1991.

[3] J. L. Fagan. Automatic phrase indexing for document retrieval: An examination of syntactic and non-syntactic methods. In *Proc. tenth Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 91–101, 1987.

[4] N. Fuhr. Probabilistic models in information retrieval. *The Computer Journal*, 35(3):243–255, 1992.

[5] J. Gao, J.-Y. Nie, G. Wu, and G. Cao. Dependence language model for information retrieval. In *Proc. 27th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 170–177, 2004.

[6] T. Joachims. Optimizing search engines using clickthrough data. In *Proc. eighth ACM SIGKDD international conference on Knowledge discovery and data mining (KDD 02)*, pages 133–142, 2002.

[7] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conference on Machine Learning*, pages 282–289, 2001.

[8] V. Lavrenko. *A Generative Theory of Relevance*. PhD thesis, University of Massachusetts, Amherst, MA, 2004.

[9] D. Metzler. Direct maximization of rank-based metrics. Technical report, University of Massachusetts, Amherst, 2005.

[10] D. Metzler and W. B. Croft. Combining the language model and inference network approaches to retrieval. *Information Processing and Management*, 40(5):735–750, 2004.

[11] D. Metzler, T. Strohman, H. Turtle, and W. B. Croft. Indri at terabyte track 2004. In *Text REtrieval Conference (TREC 2004)*, 2004.

[12] G. Mishne and M. de Rijke. Boosting web retrieval through query operations. In *Proc. 27th European Conference on Information Retrieval (ECIR '05)*, pages 502–516, 2005.

[13] W. Morgan, W. Greiff, and J. Henderson. Direct maximization of average precision by hill-climbing with a comparison to a maximum entropy approach. Technical report, MITRE, 2004.

[14] R. Nallapati. Discriminative models for information retrieval. In *Proc. 27th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 64–71, 2004.

[15] R. Nallapati and J. Allan. Capturing term dependencies using a language model based on sentence trees. In *Proc. of the 2002 ACM CIKM International Conference on Information and Knowledge Management*, pages 383–390, 2002.

[16] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proc. 21st Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 275–281, 1998.

[17] J. Robert M. Losee. Term dependence: Truncating the Bahadur Lazarsfeld expansion. *Information Processing and Management*, 30(2):293–303, 1994.

[18] S. Robertson. The probability ranking principle in IR. *Journal of Documentation*, 33(4):294–303, 1977.

[19] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

[20] F. Song and W. B. Croft. A general language model for information retrieval. In *Proc. eighth international conference on Information and knowledge management (CIKM 99)*, pages 316–321, 1999.

[21] M. Srikanth and R. Srihari. Biterm language models for document retrieval. In *Proc. 25th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 425–426, 2002.

[22] B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *Proc. of Advances in Neural Information Processing Systems (NIPS 2003)*, 2003.

[23] H. Turtle and W. B. Croft. Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems*, 9(3):187–222, 1991.

[24] C. J. van Rijsbergen. A theoretical basis for the use of cooccurrence data in information retrieval. *Journal of Documentation*, 33(2):106–119, 1977.

[25] C. T. Yu, C. Buckley, K. Lam, and G. Salton. A generalized term dependence model in information retrieval. Technical report, 1983.

[26] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proc. 24th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 334–342, 2001.