How Good is a Span of Terms? Exploiting Proximity to Improve Web Retrieval

Krysta M. Svore
Microsoft Research
1 Microsoft Way
Redmond, WA
ksvore@microsoft.com

Pallika H. Kanani
UMass., Amherst
140 Governors Drive
Amherst, MA
pallika@cs.umass.edu

Nazan Khan
Microsoft
1 Microsoft Way
Redmond, WA
nazanka@microsoft.com

ABSTRACT

Ranking search results is a fundamental problem in information retrieval. In this paper we explore whether the use of proximity and phrase information can improve web retrieval accuracy. We build on existing research by incorporating novel ranking features based on flexible proximity terms with recent state-of-the-art machine learning ranking models. We introduce a method of determining the goodness of a set of proximity terms that takes advantage of the structured nature of web documents, document metadata, and phrasal information from search engine user query logs. We perform experiments on a large real-world Web data collection and show that using the goodness score of flexible proximity terms can improve ranking accuracy over state-of-the-art ranking methods by as much as 13%. We also show that we can improve accuracy on the hardest queries by as much as 9% relative to state-of-the-art approaches.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*retrieval models, selection process*; D.0 [**Software**]: General

General Terms

Algorithms, Experimentation

Keywords

Web Search, Retrieval Models, Proximity, Learning to Rank, BM25

1. INTRODUCTION

Hundreds of millions of users issue queries to search engines daily. In order to improve the ranking of web documents, effective ranking features are necessary. One class of ranking features considered in both ad-hoc and web retrieval is proximity and phrasal information. In ad-hoc retrieval, relevance contributions from proximity and phrasal information have varied. For example, if a ranking function is already strong in retrieval effectiveness, then the addition of proximity and phrasal features can be neutral or even negative [10]. On the other hand, when considering only top retrieved documents, term proximity information can lead to improved retrieval effectiveness [12].

Despite relevance contributions from phrase and proximity terms varying in ad-hoc retrieval, the use of phrase and proximity terms in web retrieval may be more effective, in part due to the difference in query statistics; in ad-hoc retrieval queries are on average 4.9 terms compared to an average of 1.5–2.6 terms per query in web retrieval [9]. Intuitively, retrieval for longer queries should benefit more from proximity information, yet it has been shown previously that when using proximity features, query length is inversely correlated with retrieval effectiveness [9]. This may be because long queries are often a sequence of terms, whereas short queries form a linguistic phrase, thus proximity information is more informative. The structured nature of web documents may also contribute to the significant improvements that can be obtained through the use of phrases and term proximity information in web retrieval [9]. Recent work by Song et al. indicates that using flexible proximity terms within an information retrieval model such as BM25 results in improved retrieval effectiveness [16].

Our work is motivated by the previous experimental results in ad-hoc and web retrieval. In this paper, we exclusively focus on web retrieval. We address how much benefit phrases and proximity information have for web retrieval on the top retrieved documents. Rather than proposing changes on the query level through query alteration or query expansion, we propose changes to the features used in the ranking function. Our features are based on flexible proximity term information, called spans, as introduced in [16]. We introduce the notion of a "good" span by developing novel features of the document spans; we determine the "goodness" of a span by evaluating the presence of third-party phrasal information within spans, the formatting and structure of the spans, the density of query terms in the span, and so on. We then evaluate our span-based features within a larger ranking model on real-world Web data.

Through evaluation of our proximity features, we seek to answer if phrases and proximity terms have different impact on retrieval effectiveness for short 2–3 term queries versus long queries containing more than 4 terms. We also look at the retrieval effectiveness of phrases and proximity terms on popular, head queries versus rare, tail queries. We attempt

SIGIR'10, July 19–23, 2010, Geneva, Switzerland.
Copyright 2010 ACM 978-1-60558-896-4/10/07 ...$10.00.

to not only introduce a novel set of features for determining the importance of proximal terms that significantly improve web retrieval, but also attempt to unify previous work into a more cohesive story on proximity information in web retrieval. Throughout our work, we take a machine learning approach to ranking, allowing the model to determine where and how to use the phrase and proximity information while directly optimizing for the retrieval measure under consideration.

Our specific contributions in this paper include:

- Introduction and analysis of a novel approach for determining the "goodness" of a span of terms in a document (Section 5.2)
- Novel span features that exploit document metadata, structure, formatting, and third-party phrasal information from sources such as query logs and Wikipedia (Section 5.3)
- A large scale evaluation on a real-world Web dataset that shows significant improvements in retrieval effectiveness from using our span features (Section 7)
- An in-depth evaluation of the contributions of our span features on query segments based on length and popularity (Section 7)

2. RELATED WORK

The use of proximity information in retrieval has recently become a more interesting research avenue due to the large number of search engine users. Several retrieval models have been developed to capture proximity. Peng et al. proposed a statistical language model applied to both the query and relevant documents [11]. The model predicts query stemming operations; query term matches in the document are considered valid only if the match appears in the same context and order as in the stemmed query. A simple retrival model that focuses on efficiency rather than accuracy is proposed in [15]. Beigbeder and Mercier [1] use a model based on fuzzy proximity degree of term occurrences, but focus mainly on boolean queries. A formal, Markov Random Field model for term proximity is introduced in [8]. More recently, Lv and Zhai [7] proposed a positional language model that accounts for proximity and passage retrieval scores.

One of the most widely used models for information retrieval, BM25 [13, 6], does not include proximity information. One method of adding proximity information is to determine the frequencies of n-grams of the query in the document, such as bigrams or trigrams, and to incorporate such frequencies into BM25 [12, 3]. Rasolofo and Savoy took such an approach and reported mixed effectiveness for web retrieval, showing almost negligible results on MAP [12]. Tao et al. [18] used a span-based relevance score, where a span must contain all query terms, in conjunction with existing retrieval models. They also reported mixed results when using proximity in web retrieval. Recently, flexible, non-overlapping spans were introduced in [16]; Song et al. account for proximity by segmenting a document into flexible spans of terms and performing a weighted count of the matched query terms in the segments, where the weighting is based on the number of query terms in the span and the length of the span, and finally incorporating the counts into the BM25 function.

We build upon the work of Song et al. by proposing novel features of spans based on the formatting of the document and third-party phrasal information. In addition, we extend the notion of a span to more general ranking features that can be used within a larger machine learning model and not restricted for use within BM25. We also unify the previous approaches to proximity information in BM25 [16, 12, 18] with a recent machine learning method for improving upon information retrieval functions [17].

3. PROXIMITY IN BM25

In this section, we review several retrieval methods that incorporate proximity into BM25; these techniques will serve as baselines for our experiments in Section 7. We refer the reader to the original papers for details. We choose to compare against BM25-based baselines since BM25 is an important feature in a trained ranking model [14, 21]. We ultimately seek to improve ranking features for web retrieval, and significance over BM25 typically indicates the new feature will be effective for web retrieval.

BM25 [13, 6] is a probabilistic model of information retrieval. The BM25 relevance score S for document d and query q is computed as follows:

$$S = \sum_{t \in q} w_t \cdot \frac{(k+1) \cdot f_t}{K + f_t}, \tag{1}$$

$$K = k \cdot \left[(1-b) + b \cdot \frac{\ell}{av\ell} \right], \tag{2}$$

where t is a term in query q, f_t is the frequency of t in document d, ℓ is the length of document d, $av\ell$ is the average document length in the collection, and k and b are tuning parameters. w_t is the Robertson-Sparck-Jones inverse document frequency of term t:

$$w_t = \log \frac{N - df_t + 0.5}{df_t + 0.5}, \tag{3}$$

where N is the number of documents in the collection and df_t is the document frequency of term t.

One technique of introducing proximity into BM25 incorporates matches of adjacent and non-adjacent query bigram frequencies[1] in the document into the BM25 formula [12]. We denote this technique as BM25-P1. The relevance score for BM25-P1 is calculated as:

$$\text{BM25-P1} = S + \tag{4}$$
$$\sum_{t_i, t_j \in q | i<j} \left[\min(w_i, w_j) \cdot \frac{(k+1) \cdot \sum_{occ(t_i,t_j)} |p_j - p_i|^{-2}}{K + \sum_{occ(t_i,t_j)} |p_j - p_i|^{-2}} \right],$$

where S is the BM25 relevance score (Eq 1), K is defined in Eq 2, $\min(w_i, w_j)$ is the minimum of the Robertson-Sparck-Jones inverse document frequencies of term i and term j, $occ(t_i, t_j)$ are the occurrences of a query term pair t_i, t_j in the document, and p_i, p_j are the respective positions of query terms t_i, t_j in the document.

[1] An adjacent query n-gram is an n-gram comprised of only query terms, where the terms appear adjacent in the document. A non-adjacent query n-gram is any n-gram formed from only query terms where the terms appear within some distance in the document.

We consider a variation that employs matches of adjacent query bigrams in the document, which we call BM25-P2:

$$\text{BM25-P2} = S + \sum_{t_i, t_{i+1} \in q} \left[w_{i,i+1} \cdot \frac{(k+1) \cdot f_{i,i+1}}{K + f_{i,i+1}} \right], \quad (5)$$

where $w_{i,i+1}$ and $f_{i,i+1}$ are the document and term frequencies of query bigram t_i, t_{i+1}, respectively.

4. PROXIMITY THROUGH SPANS

Previous approaches to proximity mainly consider n-grams of query terms and their matches in the document. Song et al. [16] propose a different, more flexible approach to proximity that segments a document into *spans* based on query term matches and their positions in the document. Spans are constructed as follows. Each term position in the document, beginning at the first term position, is checked for a match against a query term. When the first query term is found, a new span begins. Terms, including non-query terms, are added to the span until it is closed or split into two spans. A span is closed or split when one of three conditions occurs: (1) the distance between the current query term match position and the next query term match position in the document is greater than a predetermined threshold, (2) the current and next query term matches in the document contain the same query term, (3) if the current and the next query term matches are different, and the previous query term match and the next query term match are identical, then the span is split into two spans based on the larger distance. Note that spans cannot overlap and need not contain every query term. Previous span methods have required that every query term be contained in the span. Spans may, and likely will, contain non-query terms, however the span beginning and end positions must always be query term match positions. We use this method of span construction throughout our paper and determine the "goodness" of each span in a document through span-based features (Section 5.3).

Song et al. incorporate spans into BM25 by replacing f_t, the frequency of term t, in Eq 1 with a *relevance contribution*, rc, based on spans in which term t occurs, $rc = \sum_{i|t \in s_i} n_i^{\lambda} d(s_i)^{-\gamma}$ [16], where

$$d(s_i) = \begin{cases} p_{i,e} - p_{i,b} + 1 & p_{i,b} \neq p_{i,e} \\ d_{max} & \text{otherwise} \end{cases} \quad (6)$$

is the length of span s_i, $p_{i,b}, p_{i,e}$ are the span's beginning and end positions in the document, n_i is the number of query terms that occur in span s_i, d_{max} is the distance threshold, and λ and γ are tuning parameters. We denote this technique as BM25-P3.

5. THE GOODNESS OF A SPAN

In Song et al. [16], the relevance contribution of a span is calculated based on only the number of query terms in the span and the total number of terms in the span. We believe that by additionally exploiting the structured nature of web documents, the availability of third-party data, linguistic features, and by taking advantage of machine learning techniques, we can improve the calculation of the relevance of a span. We propose to determine the "goodness" of a span through the development of span-based features. In this section, we first introduce how span-based features can be used within a ranking model framework. Following [17], we believe that a machine learning model will have improved retrieval effectiveness over a BM25-based model. We therefore combine the span-based features into a larger ranking model. We then develop the "goodness" of a span and develop a technique for including the span "goodness" within a ranking model. Finally, we describe a novel set of span-based features based on formatting and third-party data that together represent the relevance of a span.

5.1 Ranking with Span-based Features

Web search engines rank results based on a large number of features including query dependent features, such as matches against query n-grams in the content (i.e., anchor text, body text, title, URL) or the BM25 score of a document, and query independent features such as the PageRank of the document. Most modern search engines use automatic methods for developing the ranking model based on learning to rank techniques. In this work, we perform two types of evaluation of our span-based features: (1) an evaluation against BM25 and state-of-the-art proximity methods that employ BM25, and (2) an evaluation against a modern ranking model based on a large number of features. We perform (1) since any scoring function can be used as a feature in (2) and because BM25 is an extremely powerful feature in any ranking model. We particularly want to understand improvements that can be made to individual features. We perform (2) since any feature needs to be effective in a larger ranking model. In this work, we evaluate the effectiveness of features on the body content of the document. Thus all features discussed are extracted from only the body text, but could be applied to other content, such as anchor text and so on.

In both cases, we take a machine learning approach and train models using LambdaRank [2], a state-of-the-art neural-net based ranking algorithm that has been shown to be empirically optimal for several IR measures [4, 20]. When comparing against BM25 in evaluation (1), we specifically compare against the machine learning approach to a BM25-style function in [17], called LambdaBM25 (λBM25), that achieves significant improvements over BM25. LambdaBM25 learns a ranking model using LambdaRank [2] on the input attributes of BM25, namely term frequency, document frequency, and body content length. When evaluating our features, we add them to the LambdaBM25 feature input set and train LambdaRank over the entire set of features. Details of the models will be given in Section 6.

5.2 Deriving Span Goodness

In this section, we propose a machine learning method to determine the "goodness" of a span. We then describe how the span "goodness" scores can be combined and input as a feature into LambdaRank [2] or any ranking model.

We derive a goodness score for a document by learning on labeled training data (labeled query-URL pairs), and span features. Each span is described by a vector of feature values; the features are described in detail in Section 5.3. We assign a "goodness" score g_s to each span s in the document based on a weighted linear combination of each span's feature vector. We calculate the span goodness score g_s as

$$g_s = \sum_f \alpha_f v_{f,s}, \quad (7)$$

where f is a feature of span s, α_f is the weight of feature

f, and $v_{f,s}$ is the value of feature f for span s. To calculate the score, we must assign a weight to each α_f. If we had a labeled training set indicating the goodness of a span for a query-document pair, then we could apply machine learning to learn the weights. Unfortunately, acquiring such labels is challenging and costly. We can, however, choose to model the overall span goodness of a document. We model the goodness score for document d, g_d, based on the spans contained in the document:

$$g_d = \sum_s \sum_f \alpha_f v_{f,s} \tag{8}$$

By reversing the summations,

$$g_d = \sum_f \alpha_f (\sum_s v_{f,s}), \tag{9}$$

we can input for each feature f, $\sum_s v_{f,s}$, the sum of the document's spans' feature vectors, as a document feature in LambdaRank and learn the feature weights α_f over the labeled training data. Our method provides the flexibility to easily add additional span-based features to any ranking model.

5.3 Span-based Features

Our feature vector for a span consists of several types of features. Table 1 lists the features used in our "goodness" approach. The first set of span features are basic query match features that determine how many query terms are matched in the span and how many total terms are in the span. The density of the span is calculated as the number of query terms in the span divided by the number of terms in the span. The second set of features are formatting and linguistic features. These features include information about the definite and indefinite articles in the span, the html markup contained in the span, and so on. The third set of features determines if the span contains an "important" phrasing of the query. The list of important phrases was extracted from Wikipedia titles and by mining a search engine's query logs for common n-gram occurrences. The features express if query terms found in the span match an important phrase.

Query Match Features
Span contains ≥ 2 query terms (binary)
Span contains ≥ 4 query terms (binary)
Span length (number of terms in span)
Count of query terms in span
Density of span
Formatting Features (F)
Count of indefinite articles in spans
Count of definite articles in spans
Count of stopwords in span
Span contains only stopwords (binary)
Span contains a sentence boundary (binary)
Span contains a paragraph boundary (binary)
Span contains html markup (bold, italic, tags) (binary)
Third-party Phrase Features (P)
Span contains important phrase (binary)
Count of important phrases in span
Density of important phrases in span

Table 1: Span goodness features.

I. λBM25 Features
Term frequency of query unigrams
Document frequency of query unigrams
Length of body content (number of terms)
II. λBM25-2 Features
Term frequency of query bigrams
Document frequency of query bigrams
III. Proximity Match Features
Relevance contribution per query term (Sec 4)
Number of spans in the document
Max, avg span length
Max, avg count of query matches in spans
Max, avg span density
Length of span with highest term frequency
Term frequency of span with longest length
Term frequency of span with largest density

Table 2: Model feature sets.

We also consider adding additional features to our models that express the attributes of specific span features. The features are listed in (III) of Table 2. In particular, we add features such as the total number of spans in the document, the max and average span length, and the max and average span density. In addition, we add features such as the length of the span with the highest term frequency, the term frequency of the longest span, and the term frequency of the most dense span since they are representative features of the "best" spans in the document.

In our evaluations, we perform feature ablation studies to determine which features are most impactful and effective for improving web retrieval.

6. EXPERIMENTAL SETUP

6.1 Data

We evaluate our methods on a real-world Web data collection. The data contains queries sampled from query logs of a commercial search engine and corresponding URLs. All queries are English queries and contain up to 10 query terms. We perform some stemming on queries. Each query is associated with on average 150-200 documents (URLs), each with a vector of feature attributes extracted for the query-URL pair and a human-generated relevance label from 0 to 4, with 4 meaning document d is the most relevant to query q and 0 meaning d is not relevant to q.

The training set consists of 27,959 queries. During model training, we use 20% of the training data for validation. The test set contains 11857 queries. We examine two splits of our test set to understand the performance of our methods on different query types. One split separates short queries ($<$ 4 terms in query) from long queries ($\geq$ 4 terms in query). The other split separates head (more popular) queries from tail (less popular) queries. We use the amount of click and anchor information as an indicator of query popularity[2]. Table 3 lists the respective split sizes of our test dataset.

6.2 Evaluation Measure

[2] We could also split based on the frequency (number of times issued by users) of the query.

Dataset	Query Split Description	# Queries
Full	Full Dataset	11857
Head	Queries with anchor and clicks	9166
Tail	Queries without anchor and clicks	2691
Short	Queries < 4 terms	8766
Long	Queries ≥ 4 terms	3091

Table 3: Description of test dataset.

We evaluate using NDCG, Normalized Discounted Cumulative Gain (NDCG) [5], a widely used measure for search metrics. NDCG for a given query q is defined as follows:

$$\mathrm{NDCG}@L_q = \frac{100}{Z} \sum_{r=1}^{L} \frac{2^{l(r)} - 1}{\log(1 + r)} \qquad (10)$$

where $l(r) \in \{0, \dots, 4\}$ is the relevance label of the document at rank position r and L is the truncation level to which NDCG is computed. Z is chosen such that the perfect ranking would result in NDCG@$L_q = 100$. Mean NDCG@L is the normalized sum over all queries: $\frac{1}{N} \sum_{q=1}^{N} \mathrm{NDCG}@L_q$. NDCG is particularly well-suited for Web search applications since it accounts for multilevel relevance labels and the truncation level can be set to model user behavior. In our work, relevance is measured on a 5-level scale. We evaluate our results using mean NDCG@$1, 3, 10$. For brevity, we write NDCG@$1, 3, 10$. We also perform a signficance test, i.e., a t-test with a significance level of 0.05 (95% level). To improve readability of our tables, we only report significance numbers when the gap between models is small, or when we are performing a feature ablation study. Significance is also stated in the text. Note that a gain of $0.3 - 0.5$ NDCG is considered substantial.

6.3 Ranking Model Comparison

For each scoring function, we tuned the parameters using grid search on our validation set as described in [19]. For each Span model variant, the model was trained using LambdaRank on the training set. The learning rate, found to be 10^{-5} in each case, and epoch were chosen based on our validation set. For evaluation (1), the models contain only the listed scoring function as a feature, unless additional features are explicitly listed in the descriptions below. For evaluation (2), the models contain additional features such as traditional query-dependent and query-independent ranking features, such as BM25, the PageRank of the document, and so on, as well as the features and scoring functions (input as features) listed below.

- **BM25:** The BM25 scoring function given in Eq 1 [13, 6]. BM25 has been used in the best performing TREC Web track systems [14, 21].
- **λBM25:** The method of training LambdaRank over the input features of BM25 [17]. The features used in our model are given in (I) of Table 2. We trained LambdaBM25 on our training set and found the learning rate of 10^{-5} and epoch according to accuracy on our validation set.
- **BM25-P1:** The scoring function given in Eq 4 [12].
- **BM25-P2:** The scoring function given in Eq 5. It is a slight modification to the function of BM25-P1.

Model	N@1	N@3	N@10
BM25	24.60	27.74	34.34
BM25-P1	26.06	29.54	36.00
BM25-P2	25.27	28.72	35.35
BM25-P3	25.97	29.36	35.84
λBM25	26.22	29.41	35.92
λBM25-2	26.34	29.54	36.42
λBM25-2RC	26.96	30.51	37.17
Span	**29.56**	**32.23**	**38.47**
Span-P	28.90	31.81	38.20
Span-F	26.03	29.45	36.81

Table 4: NDCG results on the full test set. Bold indicates statistical significance over all other listed models.

- **BM25-P3:** The scoring function described in Section 4 that incorporates spans into BM25 [16].
- **λBM25-2:** λBM25 with the additional features listed in (II) of Table 2 to incorporate bigrams.
- **λBM25-2RC:** λBM25-2 with an additional feature, the relevance contribution score per query term based on spans (see Sec 4 and (III) of Table 2) [16]. This model is used to evaluate the effectiveness of using span information as a basic feature in a machine learned ranking model versus its incorporation into an information retrieval function.
- **Span:** A model containing all of our "goodness" features listed in Table 1. Note that all of the features input into the model are a sum of all of the span feature vectors. The model also contains all features listed in Table 2; we want to compare if adding span-based features improves retrieval accuracy over BM25-based proximity functions and LambdaBM25.
- **Span-F:** The Span model above, but without the features listed in the formatting section (F) of Table 1.
- **Span-P:** The Span model above, but without the features listed in the third-party phrase section (P) of Table 1.

7. EXPERIMENTAL RESULTS

In this section, we evaluate our proposed features against the baseline models and perform feature ablation studies. Improvements cannot be attributed to a low baseline; each baseline is a state-of-the-art technique and has shown sufficiently high retrieval effectiveness. We first evaluate our features against BM25 and baselines using proximity in BM25. We then evaluate our features within larger ranking models.

7.1 Evaluation of Features versus BM25

We begin by evaluating our span-based features against BM25 and proximity versions of BM25 to understand how effective our span-based features are as simple ranking models. The models are described in the previous section. Table 4 lists the results of the various models on the full test set.

We first observe that λBM25 is statistically better than BM25 and BM25-P2 at all truncation levels. It is particularly notable that λBM25-2 exhibits significantly superior accuracy at truncation levels 3 and 10 over all models listed above it in Table 4. The strong retrieval effectiveness of λBM25-2 demonstrates the power of machine learning over set retrieval functions and further supports the results found in [17].

We next evaluate the addition of the relevance contribution score (see (III) in Table 2) as a feature to λBM25-2. We find that λBM25-2RC outperforms λBM25-2 with statistical significance at all truncation levels. The experimental result supports the claims in [16] that spans are an improved method of segmenting the document into flexible spans of terms and offer improvements when incorporating proximity into a ranking model over simple n-gram matches and term frequencies.

We now evaluate the effectiveness of our new span-based features. Our Span model outperforms all baselines, including λBM25-2RC, with statistical significance at all truncation levels with a gain of almost 3 points NDCG@1. The result implies that by utilizing the “goodness” feature vectors, we can enhance the effectiveness of spans for web retrieval. It is also a very flexible method that allows for easy insertion of new span features.

How effective are the span-based phrase and formatting features for web retrieval? We find that when removing phrase features (Span-P), retrieval accuracy on the full test set drops significantly at all truncation levels. In particular, the drop at level 1 is over 0.5 points NDCG@1. Most remarkable is the effectiveness of the span-based formatting features. We observe that when removing formatting features (Span-F), retrieval accuracy on the full set drops significantly at all truncation levels by as much as 12% and by over 3.5 points NDCG@1.

Table 5 lists the results of our evaluations on various splits of the test set. Note that on the head queries, our Span model significantly outperforms all other models, indicating that both phrase and formatting span-based features contribute significantly to our model’s superior accuracy. On short queries, which tend to highly correlate with head queries, we find very similar results, except that the removal of phrase span features has no effect on accuracy. Thus, we find that formatting features are effective for short queries, but phrase features contribute negligible gains.

On tail queries, we remark that Span is significantly better than λBM25-2RC at truncation levels 3 and 10. In addition, the removal of phrase span-based feature causes no significant difference compared to the Span model. Formatting span-based features, however, cause a significant drop in accuracy, indicating that tail queries benefit from these features. Tail queries show a significant benefit from span-based features within a machine learning framework. Results on long queries indicate that phrase features are not overly effective for long query retrieval, but that formatting features remain significantly effective. It is also noteworthy that on long and tail queries, the differences between λBM25-2RC and Span are negligible, indicating that a few span features are important for long and tail queries, but more specialized span features may not be needed. This result corresponds with previous results indicating that proximity is more beneficial for retrieval of short queries [9].

7.2 Evaluation of Features in a Full Ranking Model

In this section, we evaluate the retrieval effectiveness of span-based features within a full ranking model on the body

Split	Model	N@1	N@3	N@10
Head	BM25	25.59	28.05	35.01
	BM25-P1	26.89	29.77	35.99
	BM25-P2	25.95	28.98	35.48
	BM25-P3	26.58	29.65	36.13
	λBM25	27.37	30.06	36.3
	λBM25-2	26.94	29.76	36.45
	λBM25-2RC	29.73	32.04	38.18
	Span	**30.27**	**32.63**	**38.61**
	Span-P	29.65	32.10	38.27
	Span-F	26.46	29.40	36.77
Tail	BM25	21.23	25.13	32.05
	BM25-P1	23.21	28.73	36.04
	BM25-P2	22.93	27.82	34.91
	BM25-P3	23.91	28.38	34.85
	λBM25	22.31	27.17	34.62
	λBM25-2	24.31	28.77	36.31
	λBM25-2RC	26.04	30.71	37.86
	Span	26.23*	30.87*	37.99
	Span-P	26.34	30.80	37.96
	Span-F	24.56+	29.62+	36.94+
Short	BM25	24.77	28.08	34.86
	BM25-P1	25.49	29.08	35.76
	BM25-P2	22.93	27.82	34.91
	BM25-P3	25.75	29.24	35.87
	λBM25	26.05	29.29	35.93
	λBM25-2	25.62	29.02	36.07
	λBM25-2RC	28.15	31.16	37.76
	Span	28.73*	31.82*	38.23*
	Span-P	28.16	31.43	37.91
	Span-F	24.74+	28.27+	36.09+
Long	BM25	24.13	26.75	32.86
	BM25-P1	27.68	30.83	36.68
	BM25-P2	25.08	28.61	35.43
	BM25-P3	26.60	29.73	35.75
	λBM25	26.72	29.73	35.88
	λBM25-2	28.38	31.02	37.41
	λBM25-2RC	30.99	33.37	39.09
	Span	31.15	33.41	39.13
	Span-P	31.00	32.88	39.02
	Span-F	29.67+	32.81+	38.08+

Table 5: NDCG results on test set splits. Bold indicates statistical significance over all other models. * indicates statistical signficance of Span over λBM25-2RC. + indicates statistical signficance of Span over Span-F. Other significance markers have been removed for readability of the table and are stated when necessary in the text.

Model	N@1	N@3	N@10
R+BM25	36.86	39.17	44.62
R+BM25-P3	37.09	39.14	44.49
R+λBM25	37.51	39.58	44.93
R+λBM25-2	37.24	39.12	44.66
R+λBM25-2RC	37.94	39.93	45.34
R+Span	38.18	40.29*	45.65*
R+Span-P	38.43	40.49	45.75
R+Span-F	37.57+	39.69+	45.01+

Table 6: NDCG results on the full test set using features within a full ranking model. * indicates statistical signficance of R+Span over R+λBM25-2RC. + indicates statistical signficance of R+Span over R+Span-F.

content of web documents. Our goal is to determine how much proximity information, in particular in the form of span-based features, can improve web retrieval. We also seek to determine if previous results in web retrieval indicating that proximity is not that effective when paired with a larger ranking model remain true on a large real-world Web data collection. Additionally, we ask how effective formatting, linguistic, and phrase span-based features are in the presence of a larger ranking model.

For each model listed in Section 6, we combine traditional query-dependent and query-independent ranking features, such as BM25, the PageRank of the document, and so on, with the features listed for each model. The models are denoted with "R+" to indicate full ranking model. For a scoring function model, the scoring function is input as one of the features into the larger ranking model. We train LambdaRank on the various feature sets and determine the learning rate and epoch according to the highest accuracy on the validation set.

Table 6 lists the results of training a full ranking model on the various sets of features. We do not list R+BM25-P1 or R+BM25-P2 since R+BM25-P3 includes span information and performs similarly (see Table 4). R+λBM25 performs significantly better than the two R+BM25 models at all truncation levels. Interestingly, the bigram features contained in R+λBM25-2 do not cause an increase in accuracy over R+λBM25. However, proximity information expressed through spans in R+λBM25-2RC causes significant gains over R+λBM25 and R+λBM25-2 at all truncation levels. The results show the importance of representing span information within a machine learning framework and including the span information through individual features separate from BM25, as in [17].

An even better approach is to include additional span-based features directly in the model, as shown by using R+Span. R+Span is a significantly better model than R+λBM25-2RC at truncation levels 3 and 10. By removing phrase features, the model improves insignificantly at all truncation levels (R+Span-P), which may indicate that the method of determining important phrases or the list of important phrases could be further improved. When we remove formatting features, the model's accuracy decreases significantly at all truncation levels, indicating that our formatting features are an important class of span features to include in a full ranking model. Note that R+Span-P is significantly better than all other models except Span.

Split	Model	N@1	N@3	N@10
Head	R+BM25	39.11	40.73	45.79
	R+BM25-P3	39.20	40.62	45.59
	R+λBM25	39.68	41.19	46.13
	R+λBM25-2	39.17	40.63	45.84
	R+λBM25-2RC	40.29	41.70	46.70
	R+Span	40.29	41.97	46.96
	R+Span-P	40.55	42.09	47.01
	R+Span-F	39.66+	41.20+	46.29+
Tail	R+BM25	29.22	33.86	40.64
	R+BM25-P3	29.91	34.09	40.73
	R+λBM25	30.15	34.10	40.81
	R+λBM25-2	30.67	33.96	40.66
	R+λBM25-2RC	29.91	33.88	40.86
	R+Span	30.98*	34.55*	41.19*
	R+Span-P	31.19	35.04	41.44
	R+Span-F	30.43	34.58	41.04
Short	R+BM25	37.83	40.22	45.78
	R+BM25-P3	38.05	40.13	45.62
	R+λBM25	38.49	40.55	46.09
	R+λBM25-2	38.25	40.09	45.84
	R+λBM25-2RC	39.17	41.16	46.67
	R+Span	39.25	41.48*	46.93
	R+Span-P	39.45	41.53	47.02
	R+Span-F	38.49+	40.75+	46.27+
Long	R+BM25	34.12	36.20	41.32
	R+BM25-P3	34.39	36.32	41.3
	R+λBM25	34.76	36.84	41.61
	R+λBM25-2	34.39	36.36	41.31
	R+λBM25-2RC	34.46	36.44	41.72
	R+Span	35.15*	36.89	42.01
	R+Span-P	35.55	37.54*	42.13
	R+Span-F	34.96	36.69	41.76

Table 7: NDCG results on test set splits using features within a full ranking model. * indicates statistical signficance of R+Span over R+λBM25-2RC. + indicates statistical signficance of R+Span over R+Span-F.

Table 7 lists the results of various full ranking models on splits of the test data. Most interestingly, the gains of λBM25-2RC over λBM25-2 are significant on head and short queries, with over 1 point gain at truncation levels 3 and 10, but on tail and long queries there is no significant difference. The result may indicate that span features are more beneficial for short queries, which matches previous results showing proximity helps short queries more than long queries [9].

R+Span shows significant gains over λBM25-2RC on tail queries at all truncation levels, and on long queries at position 1, but negligible differences on short and head queries. The lack of formatting span-based features has a significant impact on short and head queries, but little impact on long and tail queries. The phrase span-based features have a negligible effect on all query splits, although Span-P shows significant gains over λBM25-2RC on all query splits and truncation levels except short at 1. Thus, our span-based features, without the important phrase features, significantly improve web retrieval accuracy.

8. CONCLUSIONS AND FUTURE WORK

We have proposed a new approach for combining term proximity into a machine learning framework. Specifically, we have introduced the goodness of a span and a corresponding framework for incorporating it into a machine learning ranking model. We have also introduced novel span-based ranking features based on document formatting, linguistics, and important phrases from Wikipedia and a search engine query log. Our in-depth analysis indicates that proximity information is best extracted using spans, originally introduced in [16]. Moreover, we find that span-based features outperform an information retrieval function such as BM25 that includes proximity information. Our feature ablation studies indicate that formatting span-based features are significantly effective, while important phrase features may not be effective in a larger ranking model. They also indicate that improvements of features in small ranking models may not necessarily correlate with gains when used in a larger ranking framework. We have also shown that head and short queries benefit from different span-based features than tail and long queries. Proximity information appears more effective for short and head queries than for long and tail queries, but span-based proximity features lead to significant gains across all query sets compared to ranking models without span-based features.

Future work includes extending our approach to additional document fields, such as anchor text and title fields. We also plan to explore novel sources of phrasal information that may further improve the important phrase span-based features. Finally, our training set contains a significant number of head and short queries. We would like to train our models on a training set consisting of only long and tail queries to determine if the models can better take advantage of the span-based features.

9. ACKNOWLEDGEMENTS

P.K. is supported in part by the Center for Intelligent Information Retrieval and in part by The Central Intelligence Agency, the National Security Agency and National Science Foundation under NSF grant #IIS-0326249. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor. P.K. conducted the work while interning at Microsoft Research.

10. REFERENCES

[1] M. Beigbeder and A. Mercier. An information retrieval model using the fuzzy proximity degree of term occurences. In *SAC*, page 1018, 2005.

[2] C. J. C. Burges, R. Ragno, and Q. V. Le. Learning to rank with nonsmooth cost functions. In *NIPS*, page 193, 2006.

[3] S. Buttcher, C. L. A. Clarke, and B. Lushman. Term proximity scoring for ad-hoc retrieval on very large text collections. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, page 621, 2006.

[4] P. Donmez, K. M. Svore, and C. J. C. Burges. On the local optimality of LambdaRank. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, 2009.

[5] K. Jarvelin and J. Kekalainen. IR evaluation methods for retrieving highly relevant documents. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 41–48, 2000.

[6] S. Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments - part 1. *Information Processing & Management*, 36:779, 2000.

[7] Y. Lv and C. Zhai. Positional language models for information retrieval. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, 2009.

[8] D. Metzler and W. B. Croft. A markov random field model for term dependencies. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 472–479, 2005.

[9] G. Mishne and M. de Rijke. Boosting web retrieval through query operations. In *ECIR*, pages 502–516, 2005.

[10] M. Mitra, C. Buckley, A. Singhal, and C. Cardie. An analysis of statistical and syntactic phrases. In *Proceedings of RIAO-97*, 1997.

[11] F. Peng, N. Ahmed, X. Li, and Y. Lu. Context sensitive stemming for web search. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, 2007.

[12] Y. Rasolofo and J. Savoy. Term proximity scoring for keyword-based retrieval systems. In *ECIR*, page 207, 2003.

[13] S. Robertson and S. Walker. Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 345–354, 1994.

[14] S. E. Robertson, H. Zaragoza, and M. Taylor. Simple BM25 extension to multiple weighted fields. In *CIKM*, page 42, 2004.

[15] R. Schenkel, A. Broschart, S. won Hwang, M. Theobald, and G. Weikum. Efficient text proximity search. In *SPIRE*, page 287, 2007.

[16] R. Song, M. J. Taylor, J. R. Wen, H. W. Hon, and Y. Yu. Viewing term proximity from a different perspective. In *ECIR*, page 346, 2008.

[17] K. M. Svore and C. J. C. Burges. A machine learning approach for improved BM25 retrieval. In *CIKM*, 2009.

[18] T. Tao and C. Zhai. An exploration of proximity measures in information retrieval. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, page 295, 2007.

[19] M. Taylor, H. Zaragoza, N. Craswell, S. Robertson, and C. Burges. Optimisation methods for ranking functions with multiple parameters. In *ACM Conference on Information Knowledge Management (CIKM)*, 2006.

[20] Y. Yue and C. J. C. Burges. On using simultaneous perturbation stochastic approximation for IR measures, and the empirical optimality of LambdaRank. In *NIPS Machine Learning for Web Search Workshop*, 2007.

[21] H. Zaragoza, N. Craswell, M. Taylor, S. Saria, and S. Robertson. Microsoft cambridge at TREC 13: Web and hard tracks. In *Proceedings of TREC 2004*, 2004.