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

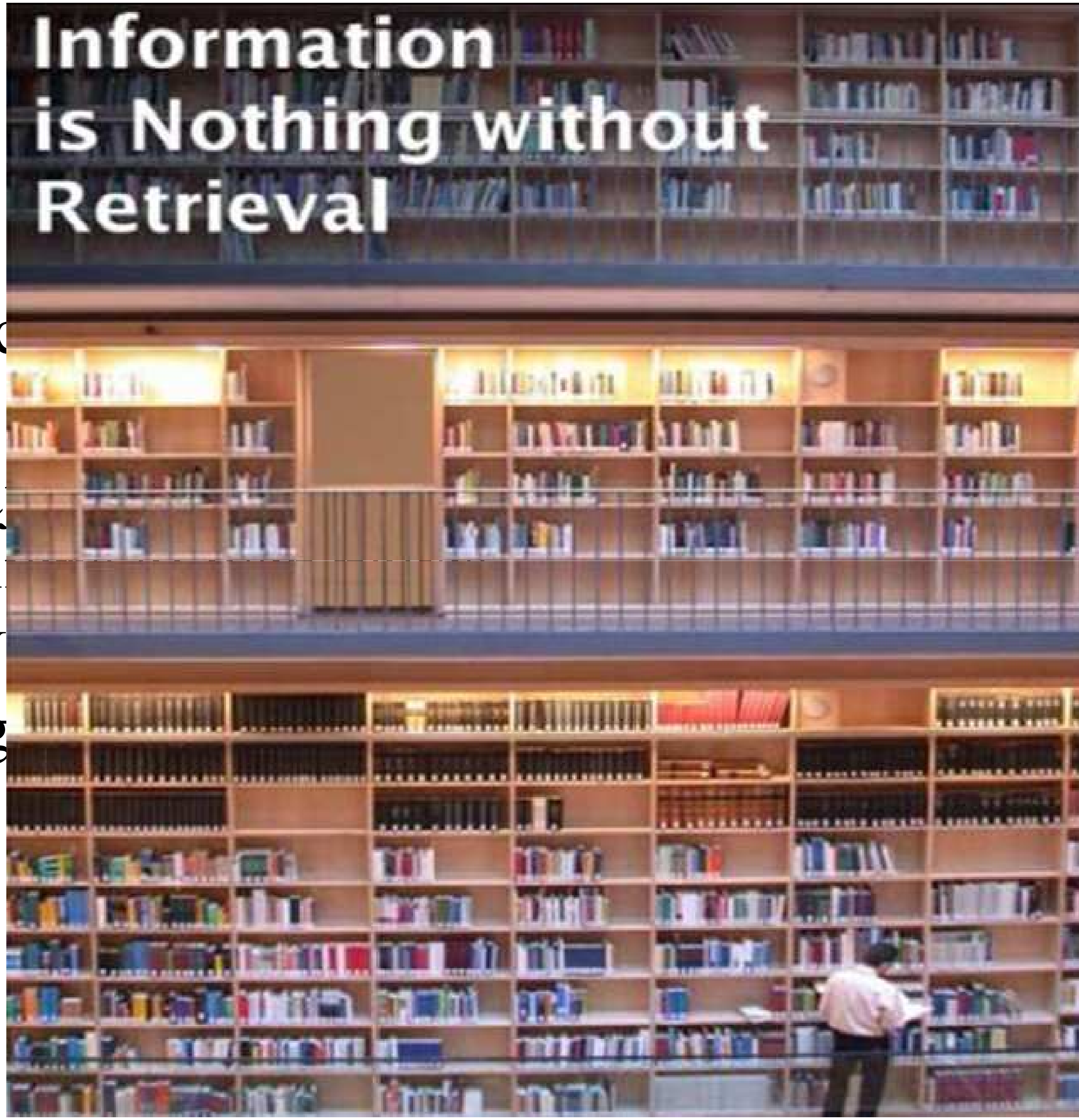
- Authors: Krysta M. Svore,
  Pallika H. Kanani,
  Nazan Khan.
- Presenter: Yury Bakanouski

INTRODUCTION

Overwhelmed by Flood of Information

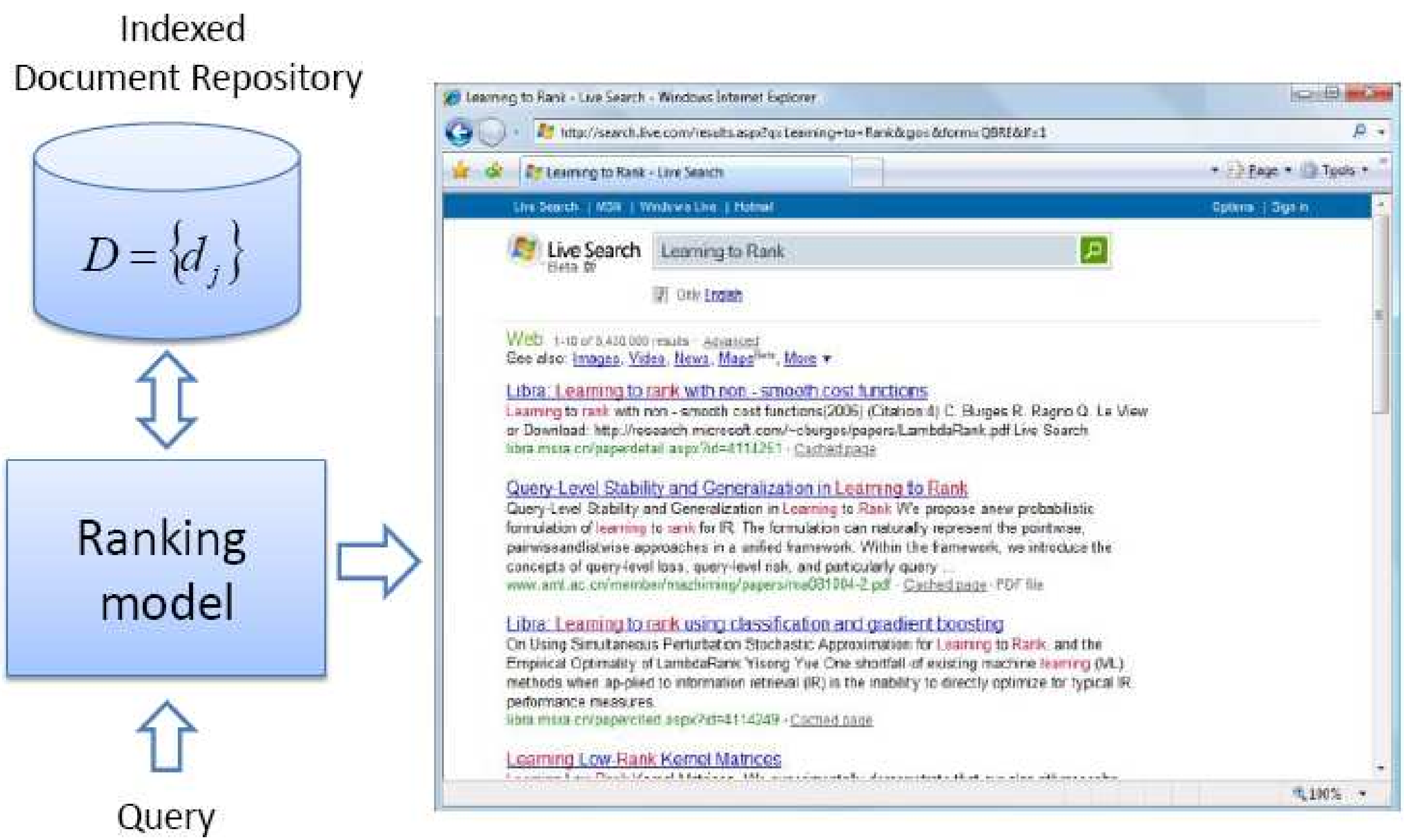
Facts about the Web

- According to www.worldwidewebsize.com, there are more than 25 billion pages on the Web.
- Major search engines indexed at least tens of billions of web pages.
- CUIL.com indexed more than 120 Billion web pages.



- Acc… , there are …
- Ma… of bill…
- CU… web pag…

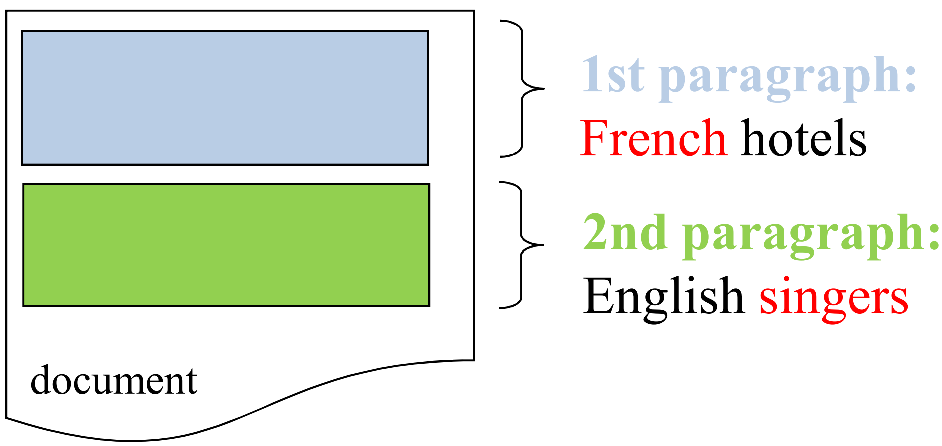
Ranking is Essential



Term Proximity

Content scores: „bag of words“, no term proximity
=> frequently unsatisfactory results

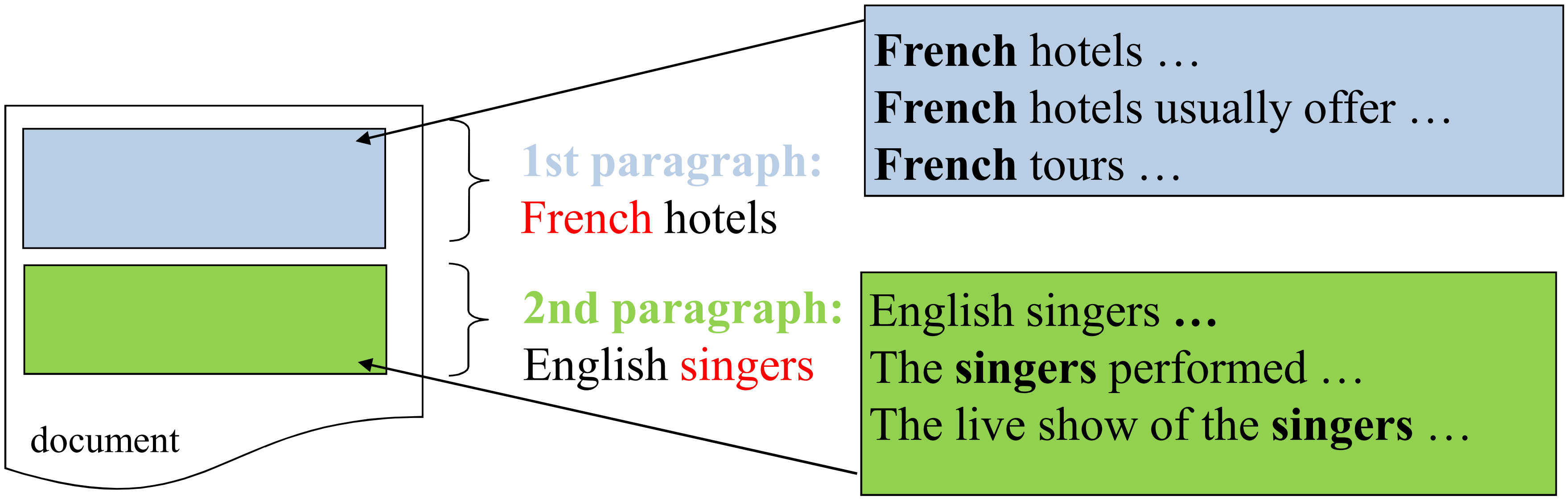
Example: query: *French singers*



All query terms individually important, but appear in different paragraphs.

Phrase queries can avoid such bad results.
But: prevent also many potentially good results.

Reason



Idea behind proximity scores:

Proximity scores will be low for high positional distances between query term

Ranking Models

- BM25; - BM25-P1; - BM25-P2; - BM25-P3;	- λBM25; - λBM25-2; - λBM25-2RC;	- SPAN; - SPAN-F; - SPAN-P.

BM25

– a probabilistic model of information retrieval

Relevance score S is computed as:

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

$$K = k\left[(1-b) + b \cdot \frac{l}{avl}\right]$$

t – a term in query *q;*
ℓ – the length of document *d*;
f_t – the frequency of *t* in document *d*;
$\alpha v\ell$ – the average document length in the collection;
w_t – Robertson-Sparck-Jones inverse document frequency of term *t*;
k, b – tuning parameters.

Robertson-Spark-Jones

– inverse document frequency of term t (IDF).

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

N – the number of documents in the collection;

dft – the document frequency of term t.

Integration Term Proximity into BM25
BM25-P1

– *incorporates matches of adjacent and non-adjacent query bigram frequencies.*

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

p_i, p_j – respective positions of query terms t_i, t_j in the document;

$occ(t_i, t_j)$ – occurrences of a query term pair t_i, t_j in the document;

$min(w_i, w_j)$ – minimum of the Robertson-Sparck- Jones inverse document frequencies of term i and term j.

Example of BM25-P1

Query $q = (t_i, t_j, t_k)$ is:

"**sea thousand years**" and $d_{max} = 10$.

Erosion[1]

It[2] took[3] the[4] *sea*[5] a[6] *thousand*[7] *years,*[8]

Obtain set of term query pairs: $q = \{(t_i, t_j), (t_i, t_k), (t_j, t_k)\}$

Term pair instance weight is:

$$\sum_{occ(t_i, t_j)} = \begin{cases} |p_j - p_i|^{-2}(\text{sea}^5, \text{thousand}^7) = \left(\dfrac{1}{7-5}\right)^2 = 0.25 \\ |p_k - p_i|^{-2}(\text{sea}^5, \text{years}^8) = \left(\dfrac{1}{8-5}\right)^2 = 0.111 \\ |p_k - p_j|^{-2}(\text{thousand}^7, \text{years}^8) = \left(\dfrac{1}{8-7}\right)^2 = 1 \end{cases}$$

BM25-P2

– employs matches of adjacent query bigrams in the document.

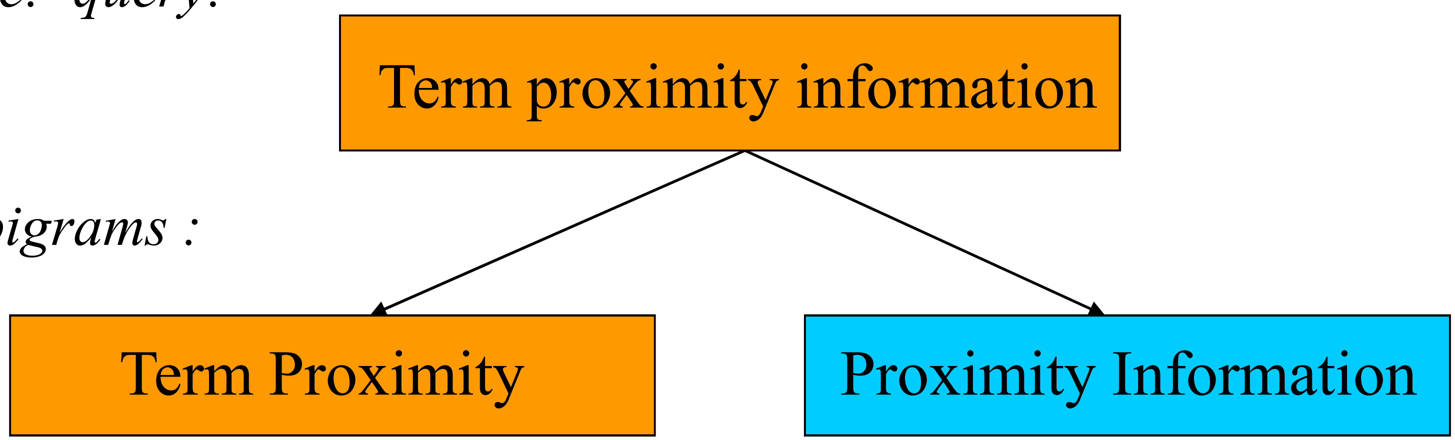
$$BM25-P2=\sum_{t_i,t_{i+1}\in q}\left[w_{i,i+1}\cdot\frac{(k+1)\cdot f_{i,i+1}}{K+f_{i,i+1}}\right],$$

$w_{i,i+1}$ – document frequency of query bigram $t_i,\ t_{i+1}$;

$f_{i,i+1}$ – term frequency of query bigram $t_i,\ t_{i+1}$.

Example: query:

Query bigrams :



Four Cases of Span

1. The distance between the current and the next is bigger than a threshold *dmax*, then the chain is separated between these two terms;

2. The current and the next terms are identical, then the chain is separated between these two terms;

3. The next term is identical to a term with former continuous sub-chain, then the distance between the current and the next and the distance between the identical term and its next is compared, the chain is separated at the bigger gap.

4. Otherwise, go on scanning the next term.

Example: How Does Span Proximity Work?

Query is: "**sea thousand years**" and $d_{max} = 10$.

Erosion[1]
It[2] took[3] the[4] *sea*[5] a[6] *thousand*[7] *years,*[8]
A[9] *thousand*[10] *years*[11] to[12] trace[13]
The[14] granite[15] features[16] of[17] this[18] cliff,[19]
In[20] crag[21] and[22] scarp[23] and[24] base[25].

It[26] took[27] the[28] *sea*[29] an[30] hour[31] one[32] night,[33]
An[34] hour[35] of[36] storm[37] to[38] place[39]
The[40] sculpture[41] of[42] this[43] granite[44] seams,[45]
Upon[46] a[47] woman[48]'s[49] face[50].
—E.[51] J.[52] Pratt[53] (1882[54]— 1964)[55]

First Span …

Erosion[1]
It[2] took[3] the[4] *sea*[5] a[6] *thousand*[7] *years,*[8]
A[9] *thousand*[10] *years*[11] to[12] trace[13]

– Scanning *sea*[5]. For *sea*[5] and *thousand*[7], the *4th case* is applied.

It[2] took[3] the[4] *sea*[5] a[6] *thousand*[7] *years,*[8]
A[9] *thousand*[10] *years*[11] to[12] trace[13]

– For *years*[8], next term is *thousand*[10], is identical to *thousand*[7], the *3rd case* is applied.

– As *thousand*[7] is nearer to *years*[8] than is *thousand*[10], so the chain is separated before *thousand*[10].

First span is: ***(sea*[5] **…** *years*[8] *)*** .

The Second, … Spans

A^{9} *thousand*^{10} *years*^{11} to^{12} trace^{13}
The^{14} granite^{15} features^{16} of^{17} this^{18} cliff,^{19}

Apply the *4th case* for *thousand*^{10}.
After scanning *years*^{11}, the distance between *sea*^{29} and *years*^{11} is further than d_{max}.
Applying the 1st case, expand span: ***(thousand^{10} years^{11})***

A^{9} *thousand*^{10} *years*^{11} to^{12} trace^{13}
The^{14} granite^{15} features^{16} of^{17} this^{18} cliff,^{19}
In^{20} crag^{21} and^{22} scarp^{23} and^{24} base^{25}.

It^{26} took^{27} the^{28} *sea*^{29} an^{30} hour^{31} one^{32} night,^{33}

An expanded span: ***(****sea*^{29} ***)*** is a single query term.

Spans of the document is:

{***(sea^{5} … years^{8}), (thousand^{10} years^{11}), (sea^{29})***}

Width? Relevance?

Width of an expanded spans is: { 4, 2, 10 }.

Relevance Contribution –
of one term occurrence, only for which contain term, is:

$$f(t, span_i) = \left(\frac{n_i}{d(span_i)}\right)^x \cdot (n_i)^y$$

$$f(sea, (sea^5 \ldots year^8)) = \frac{3}{4} \times 3 = 2.25$$

$$f(year, (thousand^{10}\, year^{11})) = \frac{2}{2} \times 2 = 2$$

$$f(sea, (sea^{29})) = \frac{1}{10} \times 1 = 0.1$$

Relevance Contribution:

$$rc = \sum_i f(t, span_i)$$

Relevance Contribution

$$rc(t) = \sum_{i/t \in s_i} n_i^{\lambda} d(s_i)^{-\gamma}$$

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

$d(s_i)$ – width of span s_i;

n_i – is the number of query terms that occur in span s_i;

λ, γ – tuning parameters;

$p_{i,b}$, $p_{i,e}$ – span's beginning and end positions in the document;

d_{max} – distance threshold.

BM25-P3, Song's Span Model

– *this approach segments a document into spans*

$$BM25-P3=\sum_{t\in q} w_t \frac{(k+1)\cdot rc(t)}{K+rc(t)}$$

rc(t) – relevance contribution;

w_t – Robertson-Sparck-Jones inverse document frequency of term t.

In *Span Model BM25* the relevance contribution of a span is the number of query terms in the span and the total number of terms in the span.

The idea of

Span Ranking Model is ... ?

The Goodness Of a Span

– through the span based features

Using:

– the structured nature of web documents;

– span features (formatting, third-party data, linguistic);

– machine learning techniques.

For improving the **relevance of a span**.

Reason:

– for improving retrieval effectiveness.

Deriving Span Goodness

– span is a vector of feature values

"Goodness" score g_s to each span s is:

$$g_s = \sum_f \alpha_f v_{f,s}$$

f – feature of span s;

$v_{f,s}$ – value of feature f for span s;

α_f – weight of feature f, apply machine learning to learn the weights.

Goodness Score – *for a document*

– based on the spans contained in the document:

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

By reversing the summations:

$$g_d = \sum_f \alpha_f \left(\sum_s v_{f,s} \right)$$

f – feature of span s;

α_f – weight of feature f;

$v_{f,s}$ – value of feature f for span s;

$\sum_s v_{f,s}$ – the sum of the document's spans' feature vectors.

Algorithm of “goodness” score

Span vector of features :

f_1	f_2	…	f_n

Document which contain *i* spans

Define the sum of values of each feature of the span vector:

$\sum_{s=1}^{i} v_{f_1}$	$\sum_{s=1}^{i} v_{f_1}$	…	$\sum_{s=1}^{i} v_{f_n}$

Learn the feature weights (α_f) over the labelled training data.
Using machine learning (LambdaRank).

“*Goodness” score of document* is the sum of multiplications feature weights with sums of value for each feature of document

Span-Based Features

Span vector consists of several types of *query dependent* features:

– *basic query match features*

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

– 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.

Formatting and Linguistic Features

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).

These features include information about:

- definite and indefinite articles in the span;
- the html markup contained in the span.

Third-party Phrase Features

Third-party Phrase Features (P)
Span contains important phrase (binary); Count of important phrases in span; Density of important phrases in span.

The third set of features determines:

– if the span contains an "important" phrasing of the query;

– if query terms found in the span match an important phrase.

The list of important phrases was extracted from Wikipedia.

Additional Features

– express the attributes of specific span 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.

Additional Features

– express the attributes of specific span features

III. Proximity Match Features
Relevance contribution per query term;
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.

The authors perform features which are most impactful and effective for improving web retrieval.

Experimental Setup

Datasets – Real-world Web data collection

– was used for evaluating proximity methods

Queries:

– are English;

– contain up to 10 query terms;

– sampled from query logs of a search engine;

– is associated with 150-200 URLs documents;

– human-generated relevance label from 0 to 4.

Splits separate:

– one separates short from long queries;

– the other separates head from tail queries.

Queries			
Head More popular queries	Tail Less popular queries	Short Less 4 terms in query	Long More 4 terms in query

Evaluation Measure

Normalized Discounted Cumulative Gain (NDCG) was used for evaluating results:

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

$l(r) \in \{0, \ldots, 4\}$ – relevance label of the document at rank position r;
L – truncation level to which NDCG is computed;
Z – chosen such that the perfect ranking would result in NDCG@Lq = 100.

Mean NDCG@L:

$$\frac{1}{N}\sum_{q=1}^{N} NDCG @ L_q$$

NDCG is well-suited for Web search applications for multilevel relevance labels.

Ranking Model Comparison

Model	*Differences*	*Used Features*
BM25	Scoring function has been used in the best performing TREC Web track systems.	Does not use features. (term frequency)
BM25-P1	Scoring function matches of adjacent and non-adjacent query bigram frequencies.	Does not use features.
BM25-P2	Scoring function. It is a slight modification to the function of BM25-P1	Does not use features.
BM25-P3	The scoring function that incorporates spans into BM25.	Does not use features.

Ranking Model Comparison

Model	*Differences*	*Used Features*
λBM25	The method of training *λ*Rank= 10^{-5} over the input features of BM25. λBM25 was trained on training set (learning rate = 10^{-5}).	"*λBM25 Features*"
λBM25-2	λBM25 with additional features to incorporate bigrams.	*λBM25 and λBM25-2*
λBM25-2RC	λBM25-2 with an additional feature, the relevance contribution score per query term based on spans.	All features.
Span model	Contains all features.	
Span-F	Contains all features except "*Formatting Features* ".	
Span-P	Contains all features except "*Third-party Phrase Features* "	

Results

Results of NDCG at all truncation levels

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

Evaluation of Features *vs* BM25

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

Scoring function is input as one of the features into ranking model.

Evaluation of Features *vs* BM25

Split	*Model*	*N@1*	*N@3*	*N@10*
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+

Evaluation of Features *vs* BM25

Split	*Model*	*N@1*	*N@3*	*N@10*
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+

Evaluation of Features *vs* BM25

Split	*Model*	*N@1*	*N@3*	*N@10*
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+

Evaluation of Features in a Full Ranking Model

Full ranking model "R+" :

– *C*ombine query-dependent and query-independent features;
– LambdaRank was trained on the various feature sets .

Previous scoring function is input as one of the features into ranking model.

Results of NDCG

– at all Truncation levels within a full ranking model

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+

NDCG results on test set splits

– full ranking model

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+

Split	*Model*	*N@1*	*N@3*	*N@10*
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+

Split	*Model*	*N@1*	*N@3*	*N@10*
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

Conclusion

- *Advantages:*

  Was proposed a new approach for combining term proximity into a machine learning framework;

  Introduced novel span-based ranking features ;

  Proximity information is best extracted using spans;

  Span-based features outperform BM25 function;

  Formatting features are more effective in retrieval.

- *Drawbacks:*

  There are not information about values of the following parameters:

  k, b – parameters in BM25;

  λ, γ – parameters (in Relevance Contribution);

  How was created ranking model BM25-P2 , it was not explained.

  For experiments was used Real-world Web data collection:

  – we do not know which documents are there;

  – documents have human-generated relevance label.

Thank you for your attention