Learning Concept Importance Using a Weighted Dependence Model

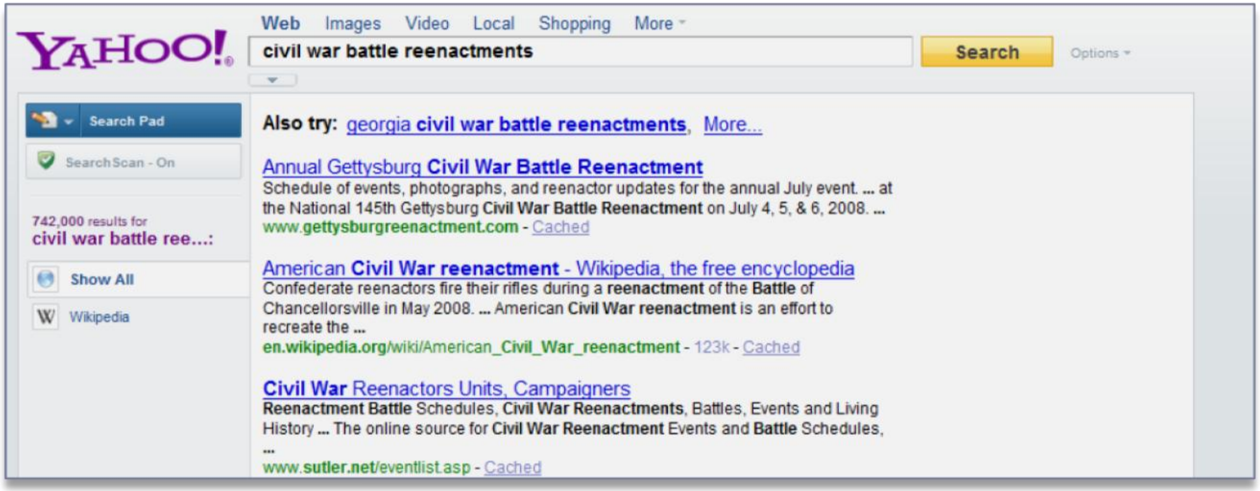
Michael Bendersky UMass Amherst
Donald Metzler Yahoo! Research
W. Bruce Croft UMass Amherst

Machine Learning & Friends Lunch, UMass
February, 2010

<u>Input</u> *free-text user query*

<u>Output</u> *Ranked list of documents*

Ranked Retrieval



- Search engine must ***accurately*** interpret query intent
  - Detect phrases
    - ***new york times*** ≠ ***time new york***
  - Detect relative term/phrase importance
    - ***CONTINENTAL airline BOOKING***

Textual Scoring

- Generally concerns computing the similarity between two pieces of text

- This talk will focus on matching (short, textual) **queries** to (long, textual) **documents**

- Not as popular as some other search problems (e.g., web link analysis), but very important for many search applications

Retrieval Models

1. Query/Document Representation
   - Bag-of-words
   - Bigrams
   - Phrases

2. Query/Document Similarity (Relevance Score)
   - Quantifies how 'relevant' a document is to an information need (expressed by a query)
   - Documents are ***ranked*** by their ***relevance score***

Term-Based Retrieval Models

- Term-based retrieval models treat the user's query as a "***bag-of-words***"
  - BM25 *(Robertson et al., 2000)*
  - Query Likelihood *(Ponte & Croft, 1998)*
  - DFR *(Amati, 2003)*
- A simple query model
  - Term order is interchangeable
  - Simple collection-based heuristics to weight query terms
    - e.g., ***IDF***
  - Term weights do not vary based on their context

Concept-Based Retrieval Models

- Recently, researchers focused on incorporating term dependence into the term-based retrieval models
  - Markov Random Fields for IR *(Metzler & Croft, 2005)*
  - BM25 with term proximities *(Song et al., 2008)*
  - DFR-SD, DFR-FD *(Peng et al., 2007)*

- A more realistic query model
  - Term order is important
  - Captures concepts - dependencies between query terms

- However, concept weighting is still ***rigid and ad-hoc***
  - e.g., ***IDF***

Term and Concept Weighting

- Term and concept weighting are usually handled outside the retrieval model
  - Automatic removal of redundant query terms
    - (Kumaran & Carvalho, 2009)
  - Classifying concepts (noun phrases) into key/non-key classes
    - (Bendersky & Croft, 2008)
  - Weighting query terms using regression on their expected performance
    - (Lease et al., 2009)

Model Desiderata

1. Concept weighting should be integrated into the retrieval model
   - Avoid pre/post-retrieval processing of queries
2. Retrieval model should handle different types of concepts
   - Moving beyond the bag-of-words
3. Retrieval model should be optimized to improve ranking
   - Avoiding metric divergence
4. Retrieval model should be general
   - Applicable to various document/query types

What is a Concept?

A concept is any syntactic expression that can be matched within a document

- Not an exhaustive definition
- Does not capture "semantics"
- *Practical definition for Information Retrieval*

Examples of Concepts

- **Unigrams**
  - (match each of the terms)

... four white churches. Our own house looked down over the town ...

- **Exact phrases**
  - (match exact phrase)

… The White House is the official residence and principal workplace of the President of the United States…

- **Proximities**
  - (match unordered phrase in a window of ***K*** terms)

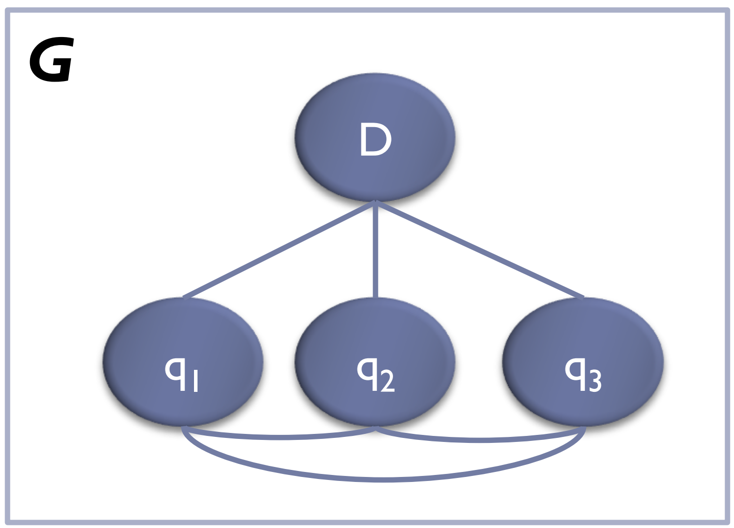
…it was during this construction that the house was painted white …

Markov Random Fields for IR

- In this work we extend the Markov Random Field approach to IR *(Metzler & Croft, 2005)*
  - provides a general way of modeling a joint distribution
  - allows incorporating arbitrary scoring functions
  - easy to train model weights
  - state-of-the-art performance
    - TREC Terabyte/Million Query/Web Tracks (2004-2009)

MRFs for IR

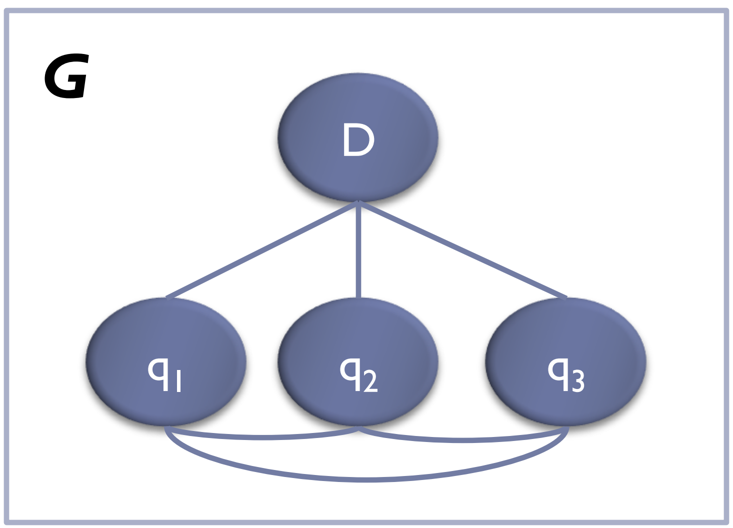
- Encode document and query terms in a graph ***G***
  - **vertices** represent document/query nodes
  - **edges** encode dependence semantics



MRFs for IR (Continued)

- Potentials over the cliques of ***G***
  - Non-negative functions over clique configurations
  - Measure query-document similarity

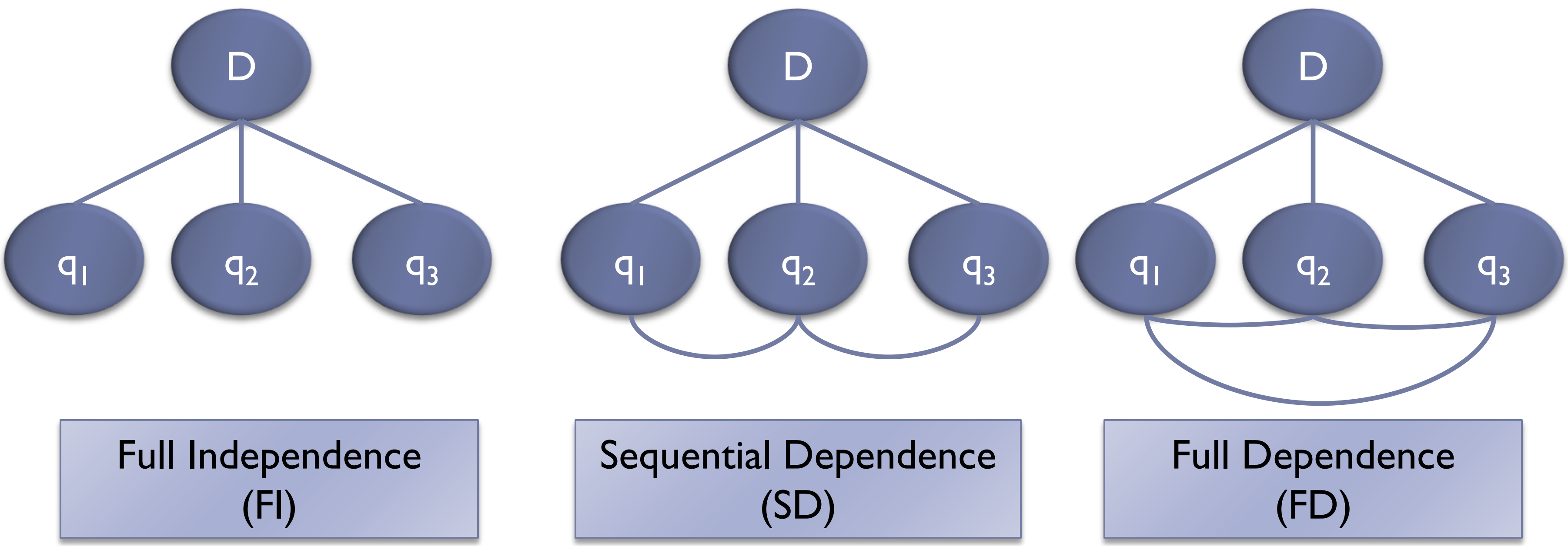
- Score the document using the joint probability over the cliques of ***G***



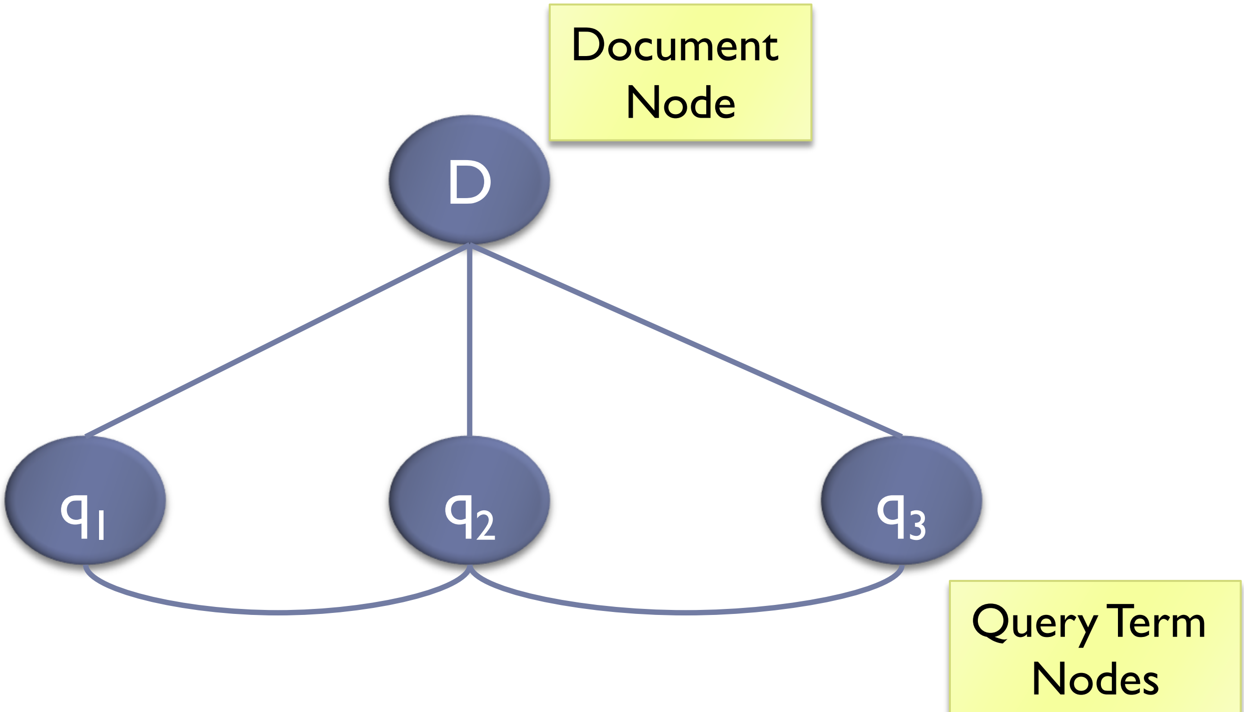
$$P_{G,\Lambda}(X_1, \ldots X_N) = \frac{1}{Z} \prod_{c \in C(G)} \psi(c; \Lambda)$$

MRFs for IR: Dependence Assumptions

- MRFs can encode different dependence assumptions



MRF - Sequential Dependence Model (SD)



- Assume dependence between adjacent terms
- Effectiveness/Efficiency tradeoff
- Empirically proven retrieval performance

SD Ranking Function

- Associate each clique in the graph with one or more potential function ***f***

$$
P(D|Q) \overset{rank}{=} \lambda_T \sum_{q \in Q} f_T(q, D) + \lambda_O \sum_{q_i, q_{i+1} \in Q} f_O(q_i, q_{i+1}, D) + \lambda_U \sum_{q_i, q_{i+1} \in Q} f_U(q_i, q_{i+1}, D)
$$

how well does q match D?
*[**bag of words** score]*

*how well **does** "q_i q_{i+1}" match D?*
*[**exact phrase** score]*

how well does prox(q_i q_{i+1}) match D?
*[**proximity** score]*

Limitations of SD

$$
\begin{aligned}
P(D|Q) \stackrel{rank}{=} \quad & \lambda_T \sum_{q \in Q} f_T(q, D) + \\
& \lambda_O \sum_{q_i, q_{i+1} \in Q} f_O(q_i, q_{i+1}, D) + \\
& \lambda_U \sum_{q_i, q_{i+1} \in Q} f_U(q_i, q_{i+1}, D)
\end{aligned}
$$

- Parameter tying
  - All matches of the same type are equally weighted
  - Especially detrimental for verbose queries
- Instead, we'd like query concept weights to vary

Weighted Sequential Dependence Model (WSD)

- Allow the parameters to depend on the concept
- Assume the parameters take a simple parametric form
  - maintains reasonable model complexity

$$\begin{aligned}\lambda(q_i) &= \sum_{j=1}^{k_u} w_j^u g_j^u(q_i)\\ \lambda(q_i, q_{i+1}) &= \sum_{j=1}^{k_b} w_j^b g_j^b(q_i, q_{i+1})\end{aligned}$$

w **- free parameters**

g - concept importance features

Defining Concept Importance

- Features **g** define the concept importance

$$\lambda(q_i) = \sum_{j=1}^{k_u} w_j^u g_j^u(q_i)$$

$$\lambda(q_i, q_{i+1}) = \sum_{j=1}^{k_b} w_j^b g_j^b(q_i, q_{i+1})$$

- Depend on the concept (term/bigram)
- Independent of a specific document/document corpus
- Combine several sources for more accurate weighting
  - ***Endogenous Features*** – collection dependent features
  - ***Exogenous Features*** – collection independent features

Concept Importance Features

	Data Source	Feature	Description
Endogenous	**Collection**	***cf(e)***	*Collection frequency for concept e*
		df(e)	*Document frequency for concept e*
Exogenous	**Google n-Grams**	***gf(e)***	*n-gram count of concept e*
	Query Log Sample	***qe_cnt(e)***	*# exact query matches for concept e*
		qp_cnt(e)	*# partial query matches for concept e*
	Wikipedia Titles	***we_cnt(e)***	*# exact title matches for concept e*
		wp_cnt(e)	*# partial title matches for concept e*

- **Unigram concepts**: *all features (7)*
- **Bigram concepts**: *all features (7) + PMI for each data source (4)*
- **Total features**: ***18***
- *All features are log-scaled and normalized*

Incorporating Concept Importance Features

$$\lambda(q_i) = \sum_{j=1}^{k_u} w_j^u g_j^u(q_i)$$

$$\lambda(q_i, q_{i+1}) = \sum_{j=1}^{k_b} w_j^b g_j^b(q_i, q_{i+1})$$

$$P(D|Q) \stackrel{rank}{=} \lambda_T \sum_{q \in Q} f_T(q, D) + \lambda_O \sum_{q_i, q_{i+1} \in Q} f_O(q_i, q_{i+1}, D) + \lambda_U \sum_{q_i, q_{i+1} \in Q} f_U(q_i, q_{i+1}, D)$$

WSD Ranking Function

- **Score document D by:**

$$P(D|Q) \overset{rank}{=} \sum_{i=1}^{k_u} w_i^u \sum_{q \in Q} g_i^u(q) f_T(q, D) + \sum_{i=1}^{k_b} w_i^b \sum_{q_j, q_{j+1} \in Q} g_i^b(q_j, q_{j+1}) f_O(q_j, q_{j+1}, D) + \sum_{i=1}^{k_b} w_i^b \sum_{q_j, q_{j+1} \in Q} g_i^b(q_j, q_{j+1}) f_U(q_j, q_{j+1}, D)$$

- Note that ***WSD*** model is also linear (with respect to ***w***)

Parameter Estimation

- Maximum likelihood or maximum *a posteriori* estimation possible from labeled data
- However, when ranking, it's better to optimize according to some rank-based loss function
- Many ***'learning to rank'*** approaches for learning linear ranking models that optimize various retrieval metrics
  - RankNet, LambdaRank, Ranking SVMs, AdaRank,…

Direct Optimization

- Learn the weights ***w*** to <u>directly optimize a retrieval performance metric</u>
  - *MAP*
    - *Mean Avg. Precision*
  - *DCG*
    - *Discounted Cumulative Gain*

$$P(D|Q) \stackrel{rank}{=} \sum_{i=1}^{k_u} w_i^t \sum_{q \in Q} g_i^u(q) f_T(q, D) + \sum_{i=1}^{k_b} w_i^b \sum_{q_j, q_{j+1} \in Q} g_i^b(q_j, q_{j+1}) f_O(q_j, q_{j+1}, D) + \sum_{i=1}^{k_b} w_i^b \sum_{q_j, q_{j+1} \in Q} g_i^b(q_j, q_{j+1}) f_U(q_j, q_{j+1}, D)$$

- We use a simple **coordinate-level ascent** algorithm
  - Efficient for a small number of parameters
  - Empirically good performance
- Most other LR4IR methods can be easily adapted for optimization

Query "civil war battle reenactments"

Concept	Importance Features			Weight
	GF	**…**	**DF**	
civil	16.9		14.1	*0.0619*
war	**17.9**		**12.8**	***0.1947***
battle	**16.6**		**12.6**	***0.0913***
reenactments	10.8		9.7	*0.3487*
civil war	14.5		10.8	*0.1959*
war battle	9.5		7.4	*0.2458*
battle reenactments	7.6		4.7	*0.0540*

Concept weights may vary even if concept DF is similar

<u>Query</u> "civil war battle reenactments"

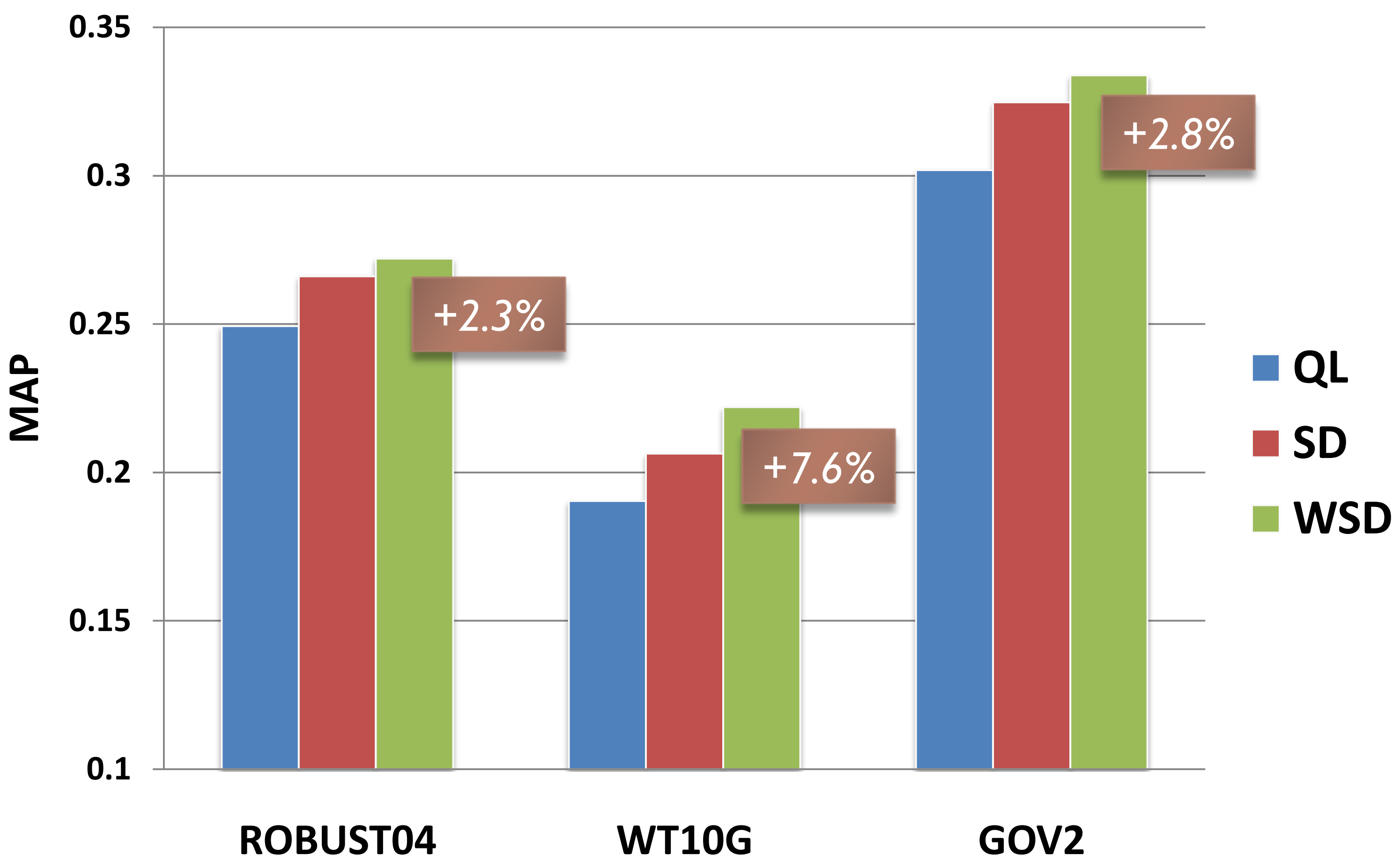
Concept	Importance Features			Weight
	GF	**…**	**DF**	
civil	16.9		14.1	<u>*0.0619*</u>
war	17.9		12.8	<u>*0.1947*</u>
battle	16.6		12.6	<u>*0.0913*</u>
reenactments	10.8		9.7	<u>*0.3487*</u>
civil war	**14.5**		**10.8**	<u>***0.1959***</u>
war battle	**9.5**		**7.4**	<u>***0.2458***</u>
battle reenactments	7.6		4.7	<u>*0.0540*</u>

Good segments do not necessarily predict important concepts

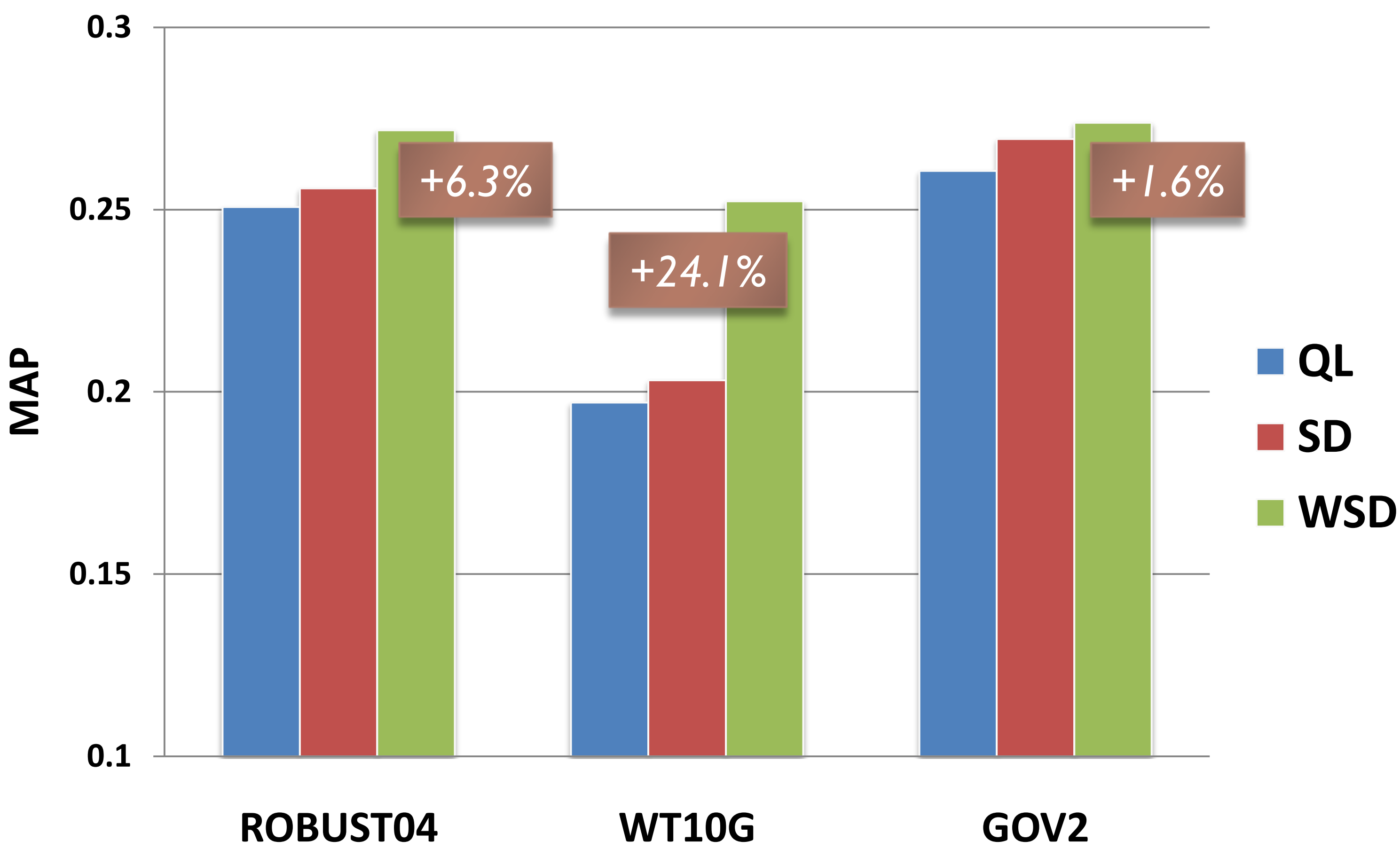
Experimental Results

- A detailed evaluation of our approach
  - ***TREC and web*** document collections
  - ***Short & Long*** queries
  - Contribution of *different* ***feature types***
  - Contribution of *different* ***concept types***

TREC Title (Short) Queries



TREC Description (Long) Queries



Endogenous & Exogenous Features

- Results with using ***either endogenous or exogenous*** features alone are comparable
- Using both types of features improves the performance over the unweighted sequential dependence model (SD)
- In most cases combining both types of features results in better performance

Term & Bigram Weights

- For short web queries (1-3 terms)
  - Bigram weights have more impact than term weights
- For TREC queries and longer web queries
  - Unigram weights have more impact than bigram weights
- In most cases combining both types of weights results in better performance, especially for longer queries

Web Queries

	DCG@1	DCG@5	DCG
QL	0.629	1.691	5.844
SD	0.864	2.383	6.681
WSD	0.884 ***(+2.3%)***	2.443 ***(+2.5%)***	6.741 ***(+0.9%)***

- Results using a large-scale commercial web search test collection
- A sample of long web search queries *(length 4+)*
- A total of 1,000 queries with 5-fold CV
- All improvements are statistically significant

Model Desiderata Revisited

1. Concept weighting should be integrated into the retrieval model
    - ***Concept importance weights are the model parameters***

2. Retrieval model should handle different types of concepts
    - ***(Potentially) handles any arbitrary term dependencies***

3. Retrieval model should be optimized to improve ranking
    - ***Concept importance weights are learned to optimize retrieval***

4. Retrieval model should be general
    - ***Improves retrieval on both newswire & web collections***
    - ***Improves retrieval with both short & long queries***

Conclusions

- Existing retrieval methods can be enhanced by
  - More accurate ***modeling*** of query concepts
  - More accurate ***weighting*** of query concepts
- Concept weight should be determined by a combination of both endogenous and exogenous features
- Dynamic concept weighting leads to significant improvements, especially for long queries

Future Work

- Incorporate concept weighting into the general "learning-to-rank" algorithms
  - Take into account both textual and non-textual (link-based, click-based) features
- Extend concept weighting beyond the original query
  - Query Expansion using (Pseudo-)Relevance Feedback
  - Query Reformulation
- Tighter integration of NLP, ML and IR
  - Using NLP/ML to find meaningful classes of concepts in both queries and documents

Thank you!