

---

# Modeling Query Term Dependencies in IR with Markov Random Fields

Donald Metzler and W. Bruce Croft

University of Massachusetts, Amherst  
Center for Intelligent Information Retrieval

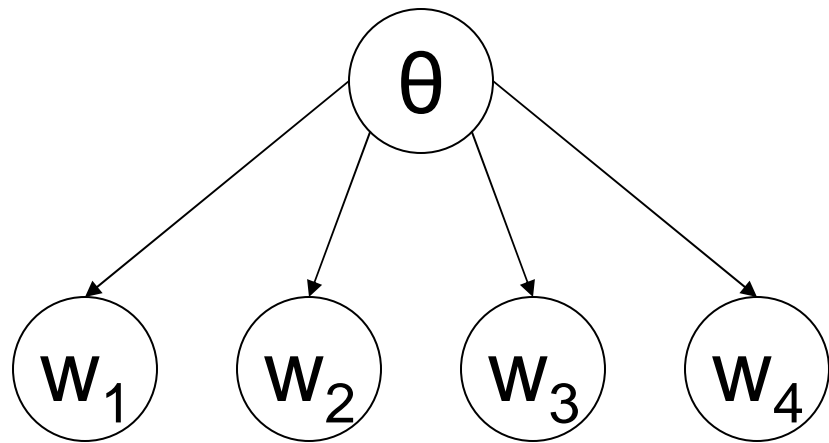


# Language Modeling for IR



- ◆ Assumes there exists some model  $\theta$  that generates both queries and documents
- ◆ Ranking documents
  - Query likelihood –  $P(Q | D)$
  - Document likelihood –  $P(D | Q)$
  - Model comparison –  $KL(\theta_Q || \theta_D)$
- ◆ Estimating complex joint distributions is difficult due to data sparsity
- ◆ Basic idea behind most of these models
  - Create a simple approximation for some complex underlying distribution

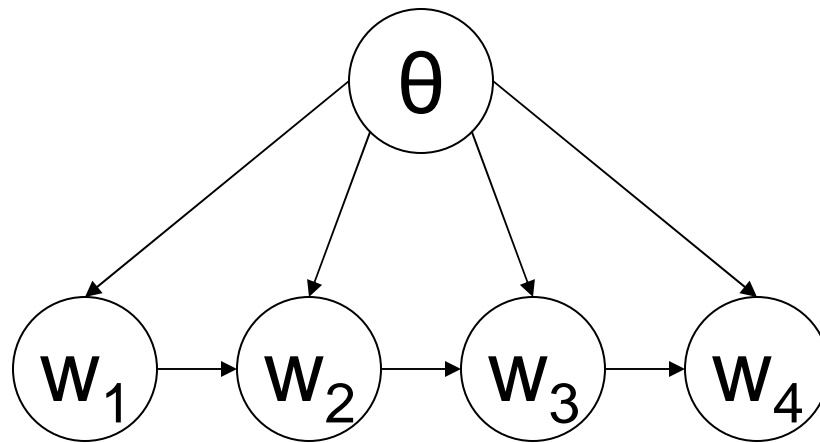
# Unigram Model



- ◆ Bag of words assumption
- ◆ Query terms independent
- ◆ Easy estimation

$$P_{\hat{\theta}}(Q, D) = \int_{\theta} P(Q | \theta) P(D | \theta) P(\theta)$$
$$\approx \prod_{q \in Q} P(q | \hat{\theta}) \prod_{d \in D} P(d | \hat{\theta})$$

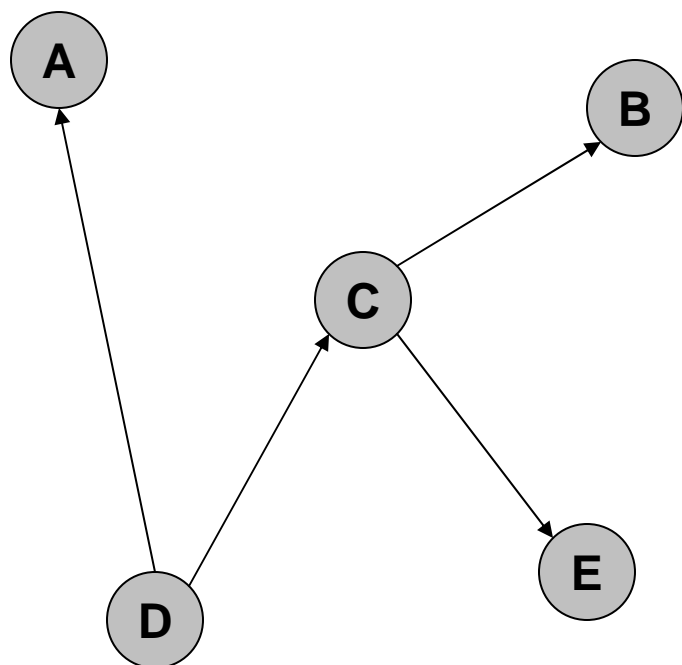
# Bigram Model



- Models sequential generation of text
- Term  $j$  dependent on term  $j-1$
- Estimation requires back-off due to sparsity

$$P_{\hat{\theta}}(Q, D) = \int_{\theta} P(Q | \theta) P(D | \theta) P(\theta)$$
$$\approx \prod_i P(q_i | q_{i-1}, \hat{\theta}) \prod_j P(d_j | d_{j-1}, \hat{\theta})$$

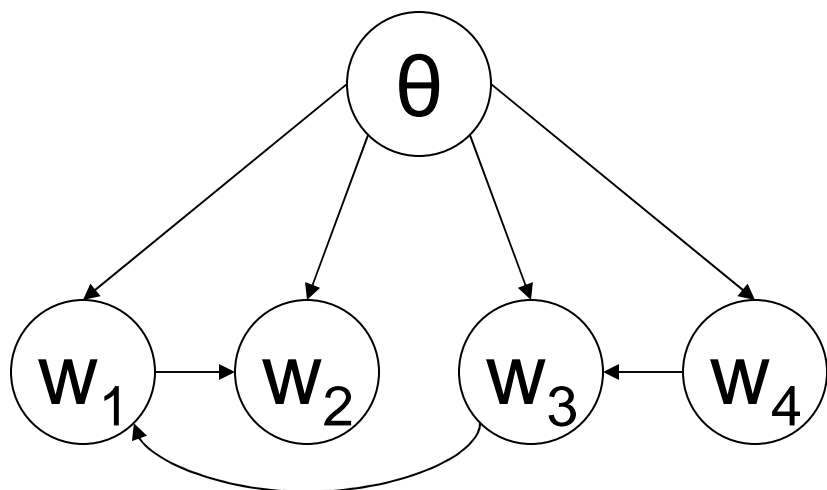
# Tree Dependence Model



- ◆ Method of approximating complex joint distribution
- ◆ Compute EMIM between every pair of terms
- ◆ Build maximum spanning tree
- ◆ Add directionality
- ◆ Tree encodes first order dependencies

$$P(A, B, C, D, E) \approx P(D)P(A | D)P(C | D)P(E | C)P(B | C)$$

# Tree Dependence Model



- Can captures non-sequential generation of text
- Estimation not as straightforward

$$\begin{aligned}
 P_{\hat{\theta}}(Q, D) &= \int_{\theta} P(Q | \theta) P(D | \theta) P(\theta) \\
 &\approx \prod_i P(q_i | q_{Parent(i)}, \hat{\theta}) \prod_j P(d_j | d_{Parent(j)}, \hat{\theta})
 \end{aligned}$$

# Pros and Cons

---

- ◆ Pros
  - Easy to implement and use
  - Perform relatively well
  - Well-studied
- ◆ Cons
  - Implicitly use textual features (unigrams, etc.)
  - Limited notion of term dependence
- ◆ Want to move away from modeling text generation to discriminating between relevant and non-relevant

# Desiderata



- ◆ Our desired model should be able to...
  - Support standard IR tasks (ranking, query expansion, etc.)
  - Easily model dependencies between terms
  - Handle wide range of features
  - Consistently and significantly improve effectiveness over bag of words models and existing dependence models
- ◆ Proposed solution: Markov random fields

# Markov Random Fields

- ◆ The anatomy of a MRF
  - Graph  $G$ 
    - vertices represent random variables
    - edges encode dependence semantics
  - Potentials over the cliques of  $G$ 
    - Non-negative functions over clique configurations
    - Measures ‘compatibility’

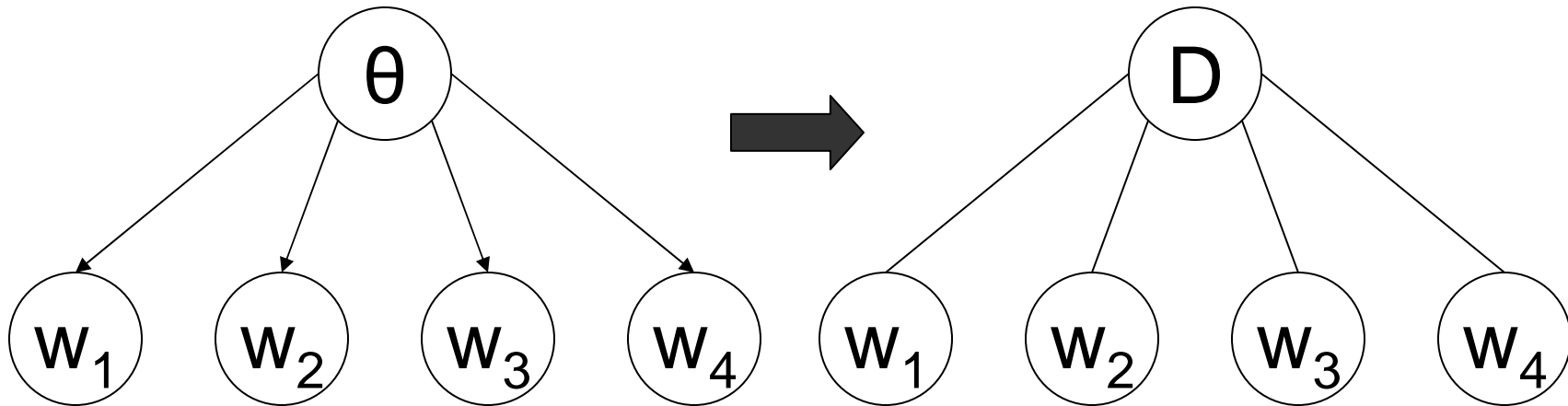
$$P_{G,\Lambda}(X) = \frac{1}{Z} \prod_{c \in \text{Cliques}(G)} \psi(c; \Lambda) \quad \psi_i(c; \Lambda) = \exp[\lambda_i f_i(c)]$$

# Using the model...



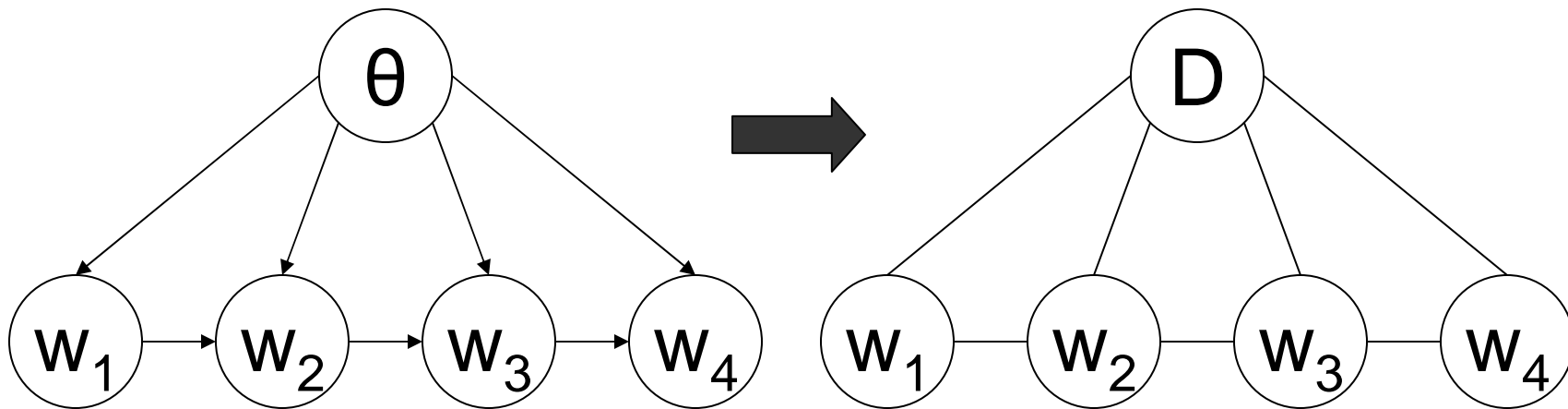
- ◆ Three steps:
  - Create graph  $G$  that encodes dependencies between query terms
  - Define set of features over cliques sets
  - Estimate parameters

# Full Independence



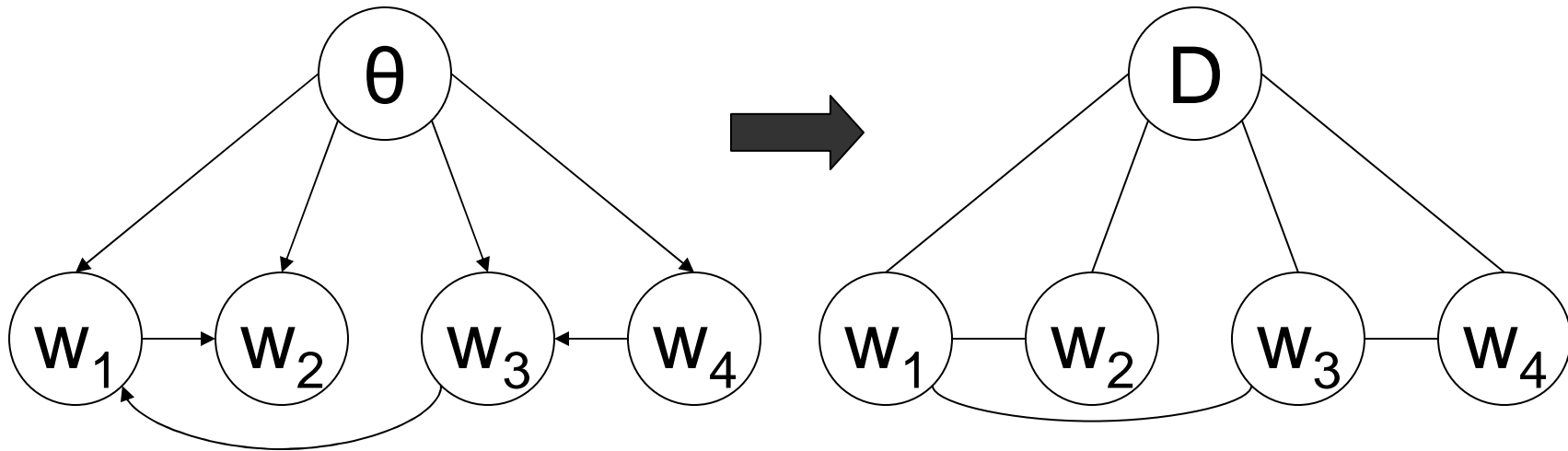
- ◆ Generalization of unigram model
- ◆ Shifts burden from making distributional assumptions on  $\theta$  to choosing good potentials
- ◆ Potentials measure compatibility of term and document
  - $\psi(w_i, D)$  – “how many times  $w_i$  occurs within document  $D$ ”

# Sequential Dependence



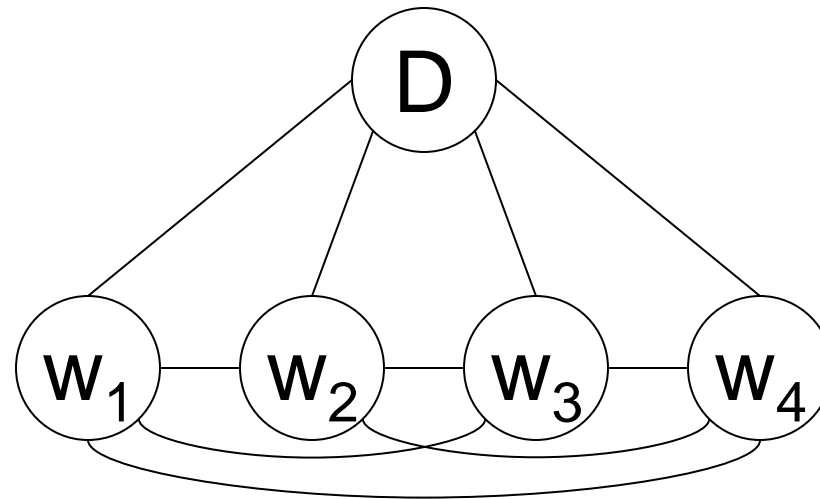
- ◆ Generalization of bigram model
- ◆ Potentials can go beyond bigram features and use generalized term proximity features
  - $\psi(w_i, w_{i+1}, D)$  – “how many times ‘ $w_i w_{i+1}$ ’ occurs as an exact phrase in document  $D$ ”

# Generalized Dependence



- ◆ Any dependence structure can be used
- ◆ Various possibilities for potentials
  - $\psi(w_i, w_j, D)$  – “how many times  $w_i$  and  $w_j$  occur within  $N$  words of each other in document  $D$ ”

# Full Dependence



- ◆ Most general model
- ◆ Makes no dependence assumptions

# Parameter Tying

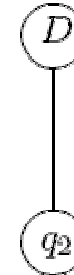
- ◆ In theory, a different potential function can be associated with every clique in a graph
- ◆ Typical solution is to define potentials over *maximal* cliques of  $G$
- ◆ Need more fine-grained control over our potentials
- ◆ Use clique sets
  - Set of cliques that share a parameter and potential function
  - We identified 7 clique sets that are relevant to IR tasks

# Clique Sets

## **Single term document/query cliques**

$T_D =$  cliques w/ one query term +  $D$

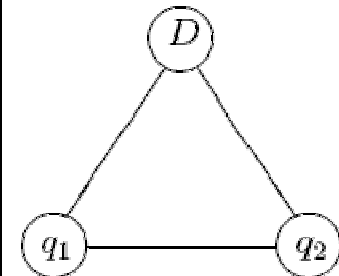
$\psi(\text{domestic}, D), \psi(\text{adoption}, D), \psi(\text{laws}, D)$



## **Ordered terms document/query cliques**

$O_D =$  cliques w/ two or more contiguous query terms +  $D$

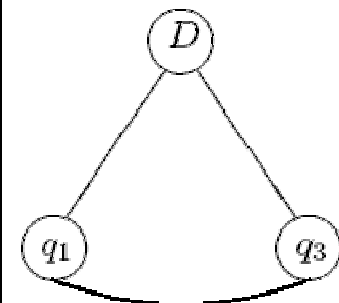
$\psi(\text{domestic}, \text{adoption}, D), \psi(\text{adoption}, \text{laws}, D),$   
 $\psi(\text{domestic}, \text{adoption}, \text{laws}, D)$



## **Unordered terms document/query cliques**

$U_D =$  cliques w/ two or more query terms (in any order) +  $D$

$\psi(\text{domestic}, \text{adoption}, D), \psi(\text{adoption}, \text{laws}, D),$   
 $\psi(\text{domestic}, \text{laws}, D), \psi(\text{domestic}, \text{adoption}, \text{laws}, D)$



# Clique Sets

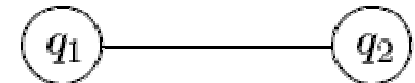
## **Single term query cliques**

$T_Q$  = cliques w/ one query term  
 $\psi(\text{domestic})$ ,  $\psi(\text{adoption})$ ,  $\psi(\text{laws})$



## **Ordered terms query cliques**

$O_Q$  = cliques w/ two or more contiguous query terms  
 $\psi(\text{domestic, adoption})$ ,  $\psi(\text{adoption, laws})$ ,  $\psi(\text{domestic, adoption, laws})$



## **Unordered terms query cliques**

$U_Q$  = cliques w/ two or more query terms (in any order)  
 $\psi(\text{domestic, adoption})$ ,  $\psi(\text{adoption, laws})$ ,  $\psi(\text{domestic, laws})$ ,  
 $\psi(\text{domestic, adoption, laws})$



## **Document clique**

$D$  = singleton clique w/  $D$   
 $\psi(D)$



# Simplified Joint Distribution



- Using these clique sets, the joint distribution simplifies to:

$$\begin{aligned}
 \log P_{G,\Lambda}(Q, D) = & \underbrace{\lambda_{T_D} \sum_{c \in T_D} f_{T_D}(c) + \lambda_{O_D} \sum_{c \in O_D} f_{O_D}(c) + \lambda_{U_D} \sum_{c \in U_D} f_{U_D}(c)}_{\text{document + query dependent}} + \\
 & \underbrace{\lambda_{T_Q} \sum_{c \in T_Q} f_{T_Q}(c) + \lambda_{O_Q} \sum_{c \in O_Q} f_{O_Q}(c) + \lambda_{U_Q} \sum_{c \in U_Q} f_{U_Q}(c)}_{\text{query dependent}} + \\
 & \underbrace{\lambda_D f_D(D)}_{\text{document dependent}} - \underbrace{\log Z_\Lambda}_{\text{document + query independent}}
 \end{aligned}$$

# Ranking with the Model

- Documents are ranked according to:

$$P_{G,\Lambda}(D|Q) \stackrel{\text{rank}}{=} \lambda_{T_D} \sum_{c \in T_D} f_{T_D}(c) + \lambda_{O_D} \sum_{c \in O_D} f_{O_D}(c) + \lambda_{U_D} \sum_{c \in U_D} f_{U_D}(c) + \lambda_D f_D(D)$$

- Can use  $P(Q | D)$  for query expansion (ongoing work)

# Features



- ◆ Features can be either textual or non-textual
- ◆ Should measure compatibility of clique configuration
- ◆ Examples:
  - TF – how common is this term in this document?
  - IDF – how rare is this term?
  - Named entities – is this a name? location? date?
  - Term proximity – where do the query terms occur?
  - Stylistic – is the matched term in a heading? table? title?
  - Document length – does this document contain 1 or 1M words?
  - PageRank – how ‘popular’ is this web document?
  - Readability – how readable is this document?

# Parameter Estimation

- Given a set of relevance judgments  $T$ , we want the maximum *a posteriori* estimate:

$$\hat{\Lambda} = \arg \max_{\Lambda} P(\Lambda | T)$$

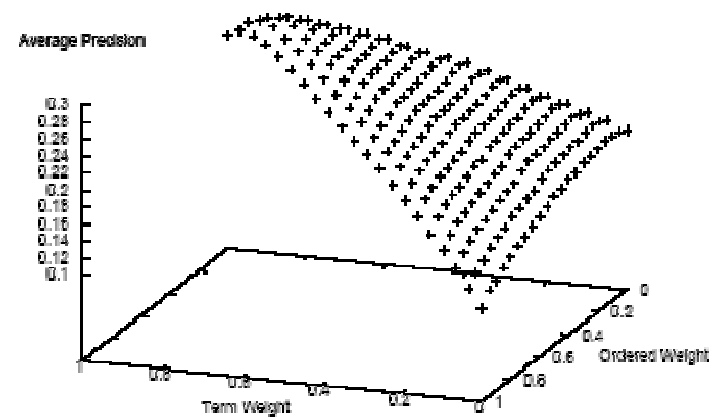
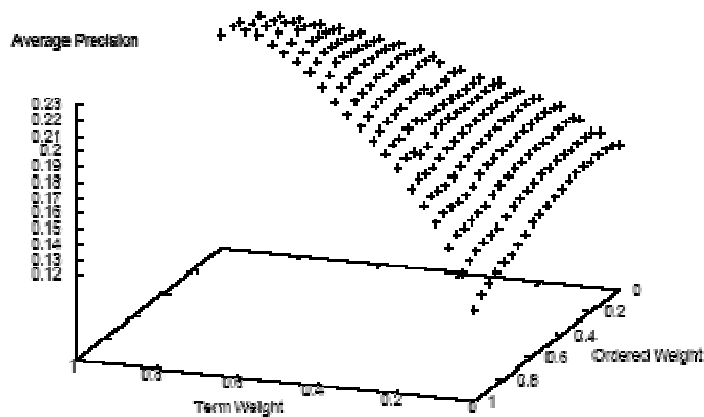
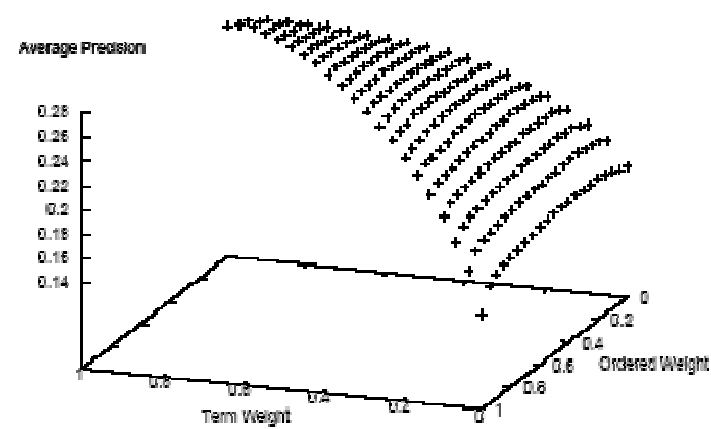
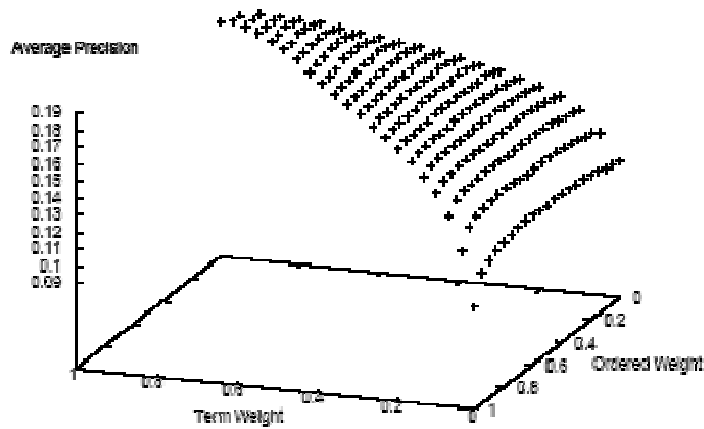
- What is  $P(\Lambda | T)$ ?  $P(T | \Lambda)$  and  $P(\Lambda)$ ?
  - Depends on how model is being evaluated!
  - Want  $P(\Lambda | R)$  to be peaked around the parameter setting that maximizes the metric we are interested in

# Direct Maximization



- ◆ Must search for maximum
  - Can be shown that intrinsic parameter space is multinomial manifold (a simplex)
- ◆ Most IR metrics are not differentiable with respect to the model parameters
- ◆ Perform coordinate ascent over the simplex
- ◆ Feasible since we have a small number of parameters

# Evaluation Metric Surfaces



# Application: Ad Hoc Retrieval



- ◆ Use Indri search engine (<http://www.lemurproject.org/indri>)
- ◆ Standard document preprocessing
  - Stopword removal
  - Stemming
- ◆ Use simple phrase and term proximity features

<b>Name</b>	<b>Description</b>	<b>Size</b>	<b># Docs</b>
WSJ	<i>Wall Street Journal</i> '87-'92	510MB	173,252
AP	<i>Associated Press</i> '88-'90	730MB	242,918
ROBUST04	Collection of news sources	860MB	528,155
WT10g	TREC Web Track collection	11GB	1,692,096
GOV2	Crawl of .GOV domain circa 2004	427GB	25,205,179

# Ad Hoc Retrieval Features



$$f_{T_D}(q_i, D) = \log \left[ (1 - \alpha_D) \frac{tf_{q_i, D}}{|D|} + \alpha_D \frac{cf_{q_i}}{|C|} \right]$$

$$f_{O_D}(q_i, \dots, q_{i+k}, D) = \log \left[ (1 - \alpha_D) \frac{tf_{\#1(q_i, \dots, q_{i+k}), D}}{|D|} + \alpha_D \frac{cf_{\#1(q_i, \dots, q_{i+k})}}{|C|} \right]$$

$$f_{U_D}(q_i, \dots, q_j, D) = \log \left[ (1 - \alpha_D) \frac{tf_{\#uwN(q_i, \dots, q_j), D}}{|D|} + \alpha_D \frac{cf_{\#uwN(q_i, \dots, q_j)}}{|C|} \right]$$

All other features set to 0 (potential function = 1)

# Ad Hoc Retrieval Results



## Mean Average Precision

<i>Collection</i>	<i>FI</i>	<i>SD</i>	<i>FD</i>
AP	0.1775	0.1867* (+5.2%)	0.1866* (+5.1%)
WSJ	0.2592	0.2776† (+7.1%)	0.2738* (+5.6%)
WT10g	0.2032	0.2167* (+6.6%)	0.2231** (+9.8%)
GOV2	0.2502	0.2832* (+13.2%)	0.2844* (+13.7%)

## Precision @ 10

<i>Collection</i>	<i>FI</i>	<i>SD</i>	<i>FD</i>
AP	0.2912	0.2980 (+2.3%)	0.3068* (+5.4%)
WSJ	0.4327	0.4427 (+2.3%)	0.4413 (+2.0%)
WT10g	0.2866	0.2948 (+2.9%)	0.3031 (+5.8%)
GOV2	0.4837	0.5714* (+18.1%)	0.5837* (+20.7%)

\* stat. sig. better than FI

\*\* stat. sig. over SD and FI

† stat. sig. over FI and FD

# Ad Hoc Retrieval Results



## *Mean Average Precision*

	<i>Unigram</i>	<i>Bigram</i>	<i>MRF</i>
AP	0.1824	0.1857 (+1.8%)	0.1896* (+3.9%)
WSJ	0.2741	0.2773 (+1.2%)	0.2857** (+4.2%)
ROBUST04	0.2512	0.2549 (+1.5%)	0.2616** (+4.1%)
WT10G	0.2008	0.2194* (+9.3%)	0.2238* (+11.5%)
GOV2	0.2957	-	0.3228* (+9.2%)

\* = significant over unigram

\*\* = significant over unigram and bigram  
(one tailed, paired t-test w/  $p < 0.05$  )

# Application: Web Search

- ◆ Three flavors of web search
  - Content-based search (ad hoc retrieval)
  - Homepage search
  - Named page finding } Known-item search
- ◆ Typically only one document on entire web is relevant for known-item search queries
- ◆ Use document structure (HTML tags) and other features such as PageRank

# Named-Page Finding Results



- ◆ Textual features
  - Multinomial features for each field (title, anchor, heading)
- ◆ Non-textual features
  - PageRank
  - Inlink count
- ◆ Results for 272 queries over GOV2 collection

Variant	MRR	S@10	Not Found
FI	0.414	0.563	0.175
FD	0.441	0.583	0.171
Best TREC	0.463	0.595	0.179

# Conclusions

---



- ◆ Markov random fields are robust models for information retrieval
- ◆ Benefits:
  - Easily model term dependencies
  - Ability to incorporate textual and non-textual features into the model
  - Query expansion framework
  - State of the art retrieval effectiveness

---

# Questions?

Contact:

<http://ciir.cs.umass.edu/~metzler>  
metzler@cs.umass.edu

