

MRFs in IR & the theory of Latent Concept Expansion using MRFs

Ashutosh Agarwal

Paper by : Donald Metzler & W. Bruce Croft,
SIGIR'07

Query Expansion ? What ??

- Wikipedia :
 - “process of reformulating a seed query to improve retrieval performance in information retrieval operations”
- Involves
 - Finding synonyms of words & searching on them also
 - Finding all morphological forms of the word (searching on stem)
 - Fixing spelling errors
 - Weighting the terms in the query

Example

- Lets say we search for “aircraft”
 - It can match plane
 - Referring to airplane
 - But should not match to plane land or working plane (bench)

Query Expansion ? NLP ??

- Complex information needs are expressed as
 - Keywords list,
 - Sentence, question
 - Long narrative
- IR involves translating from a natural language sentence to a query
 - Loss of info in this process
- Used to augment the original query
- Representation representing the actual info need is required

Approaches to problem

- Global methods
 - Expand or reformulate the query independent of the given query/results. E.g.,
 - Query expansion using WordNet/Thesaurus
- Local Methods
 - Adjust the query by looking at the documents that appear initially to match the query
 - Relevance feedback techniques
 - Most widely used approach.

Relevance feedback mechanism

- Involve the user in the IR process to improve the final result
 - The user issues a (short, simple) query
 - System returns an initial set of results
 - Some results are marked relevant and some non-relevant
 - System computes better representation of the information need “query” based on the “feedback”
 - Revised set of results are returned
 - More iterations can be performed

Approaches to relevance feedback mech.

- The Rocchio algorithm
 - Classical algorithm
 - Define a set of features
 - “Expand” the user query using some knowledge of relevant and non-relevant documents
 - Approach not-practical in many cases
- Probabilistic relevance feedback (**main focus**)
 - Given relevant and non-relevant documents
 - Build a classifier

Probabilistic Relevance feedback – Language modeling

- Language model
- Assign a probability to a sequence of words
- $P(w_1, \dots, w_n)$
- A language model is associated with a document
- Used in ranking the documents
 - For example, Query Q, language models for each document
 - Retrieved documents ranked on the probability of generating the query
 - $P(Q | M_D)$
 - Markov assumptions commonly used
 - Thus n-grams models arise

Statistical Language Modeling approaches

- Term dependence models
 - Unigram, bigram, etc.
 - Most such systems failed to show consistent improvements over unigram
- Previous query expansion techniques
 - Based on bag of words models
 - Could not make use of arbitrary features
 - Hence less robust

Motivation for MRF

- Provides mechanism for
 - Combining term dependence with query expansion
 - Allows arbitrary features to be included in the model
 - More robust features can be used
 - Better discrimination b/w documents

MRF : Summary

- Undirected graphical models
- Graph G ,
 - Nodes represent the random variables
 - Edges represent the dependence relationship
- Potential function defined over the cliques in G
- Markov property
 - “A node is independent of all of its non-neighboring nodes given the observed values for its neighbors”
 - $P_{G,V}(Q,D) = (\text{Product of potential functions for each clique}) / Z_A$

MRFs (contd.)

- For ranking purposes we compute

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

- Potential functions

- Are non-negative
- Commonly parameterized as

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

- $f(c)$ is real valued called “feature function” over cliques of G
- λ_c is the weight given to each function

Therefore, IR using MRFs involves

1. Construct a graph G representing the query dependencies to model
2. Define a set of potential functions over the cliques C in graph G
3. Rank documents in descending order of $P_{\Lambda}(D|Q)$

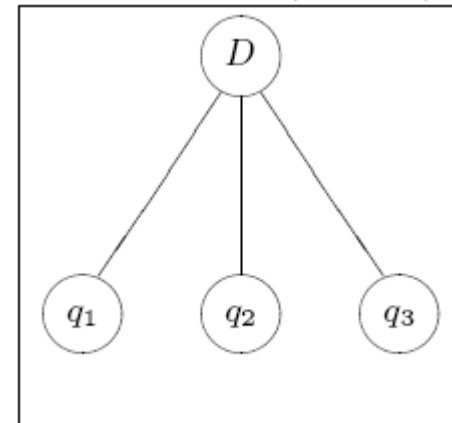
1. Constructing the graph G

- G depends on dependence assumptions
- Thus, three variants
 - Full Independence (FI)
 - Sequential Independence (SI)
 - Full dependence (FD)

1.a Full Independence

- Assumes all query terms are independent of each other given document D

$$P(q_i | D, q_{j \neq i}) = P(q_i | D)$$

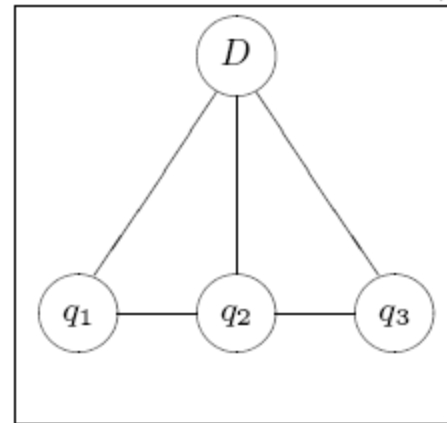


1.b Sequential Dependence

- Assumes dependence between neighboring terms

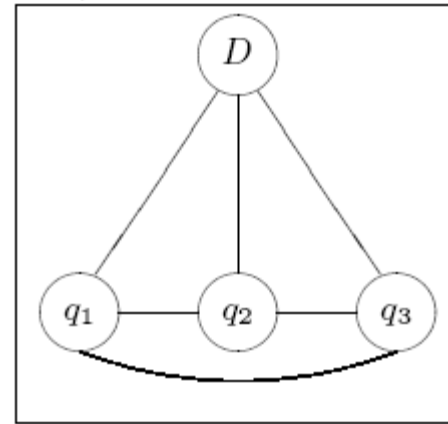
$$P(q_i|D, q_j) = P(q_i|D)$$

- If q_j is not adjacent to q_i
- Can emulate bi-gram language models



1.c Full dependence

- Assumes each query term is related to all other query terms
- Query of length n
 - Complete graph K_{n+1}
- Captures longer range dependencies



2. Potential Function

- Should assign high values to cliques where nodes are most “compatible” to each other
- Example, consider document on topic *information retrieval*
 - Assuming seq. dependence,
$$\psi(\text{information, retrieval, } D) > \psi(\text{information, assurance, } D), :$$
 - *Information* and *retrieval* more compatible than *information* and *assurance*
- Defined by feature function and weight
- Need to limit number of feature functions
 - For feasibility
 - Cliques within the same clique set share the same feature function

2.a Clique Sets in G

- Seven clique sets (feature functions)
 - T_D – cliques containing doc. node & exactly one query term
 - O_D – cliques containing doc. node & 2 or more query terms that appear in seq. order in the query
 - U_D – cliques containing doc. node & 2 or more query terms that appear in any order within the query
 - T_Q – cliques containing exactly one query term
 - O_Q – cliques containing 2 or more query terms that appear sequentially in the query
 - U_Q – cliques containing 2 or more query terms that appear in any order in the query
 - D – clique set containing only singleton node D

2.a Putting all together

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

2.b Choosing feature functions

- Need to choose feature functions for each clique set
- No universal optimum feature function
 - Depends on retrieval task, etc
 - Examples : tf-idf
 - Term proximity,
 - Document dependent features
 - Document length
 - Page rank
 - Readability
 - Genre

3. Ranking documents

$$P_{G,\Lambda}(D|Q) \stackrel{\text{rank}}{=} F_{DQ}(D, Q) + F_D(D)$$

(dropping the document and query independent features)

Query Expansion

Latent Concept

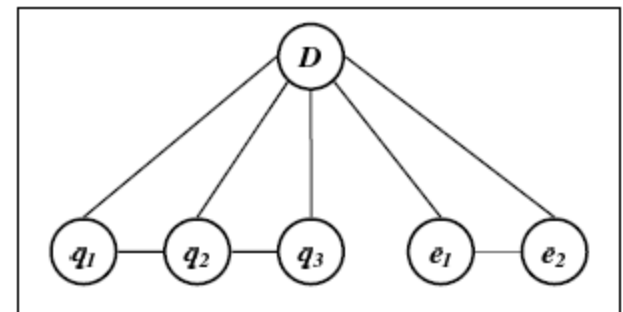
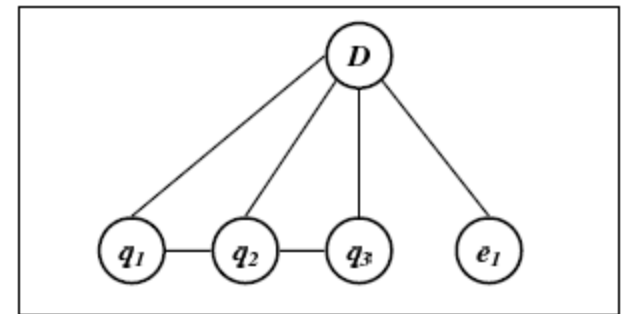
- Concepts that user has in mind, but did not explicitly expressed in the query
 - Consist of single terms
 - Multi-terms
 - Combination of two
- Recover these latent concepts from original query

Steps towards query expansion

1. Extend graph G to H to include the concept
2. Compute $P_{H,\Lambda}(E|Q)$
 - Probability distribution over latent concepts
 - E is latent concept (one or more terms)
 - Do this for all concepts
3. Select k best concepts
4. Construct new graph G' (adding the k concepts)
5. Re-rank the documents
 - Considering the new concepts as any other terms in query

1. Extend graph G to H

- Expand G to include the type of concept
 - New graph H
- Examples, (assuming seq. dependence)
 - Three term query
 - **Single term concepts**
 - **Two term concepts**



Most likely word concepts for query *hubble telescope* using top documents

1 word concepts	2 word concepts	3 word concepts
telescope	hubble telescope	hubble space telescope
hubble	space telescope	hubble telescope space
space	hubble space	space telescope hubble
mirror	telescope mirror	space telescope NASA
NASA	telescope hubble	hubble telescope astronomy
launch	mirror telescope	NASA hubble space
astronomy	telescope NASA	space telescope mirror
shuttle	telescope space	telescope space NASA
test	hubble mirror	hubble telescope mission
new	NASA hubble	mirror mirror mirror
discovery	telescope astronomy	space telescope launch
time	telescope optical	space telescope discovery
universe	hubble optical	shuttle space telescope
optical	telescope discovery	hubble telescope flaw
light	telescope shuttle	two hubble space

2. Calculate probability distribution

- Using constructed H,

$$P_{H,\Lambda}(E|Q) = \frac{\sum_{D \in \mathcal{R}} P_{H,\Lambda}(Q, E, D)}{\sum_{D \in \mathcal{R}} \sum_E P_{H,\Lambda}(Q, E, D)}$$

- \mathcal{R} set of all possible documents
- Too complex to compute as it is, hence approx.

$$\begin{aligned} P_{H,\Lambda}(E|Q) &\approx \frac{\sum_{D \in \mathcal{R}_Q} P_{H,\Lambda}(Q, E, D)}{\sum_{D \in \mathcal{R}_Q} \sum_E P_{H,\Lambda}(Q, E, D)} \\ &\propto \sum_{D \in \mathcal{R}_Q} \exp \left[F_{QD}(Q, D) + F_D(D) + F_{QD}(E, D) + F_Q(E) \right] \end{aligned}$$

- \mathcal{R}_Q is the set of relevant or pseudo-relevant documents for Q & cliques in H

2.2 Interpreting the distribution func.

$$\sum_{D \in \mathcal{R}_Q} \exp \left[\underbrace{F_{QD}(Q, D) + F_D(D)}_{\text{red double arrow}} + \underbrace{F_{QD}(E, D)}_{\text{blue double arrow}} + \underbrace{F_Q(E)}_{\text{black double arrow}} \right]$$

- Combination of
 - Original queries score for document
 - Concept E's score for the document
 - E's document independent score
- Measures how well
 - Q & E account for top ranked documents
 - “goodness” of E, independent of documents

Experimental Results

- Test set mean average precision for single-term expansion
 - language modeling(LM), MRF, Relevance models (RM3)
 - Latent concept expansion (LCE)
 - WSJ, AP, ROBUST, WT10g, GOV2 are different data sets
 - Clearly significant improvements

	LM	MRF	RM3	LCE
WSJ	.3258	.3425 ^α	.3493 ^α	.3943 ^{αβγ}
AP	.2077	.2147 ^α	.2518 ^{αβ}	.2692 ^{αβγ}
ROBUST	.2920	.3096 ^α	.3382 ^{αβ}	.3601 ^{αβγ}
WT10g	.1861	.2053 ^α	.1944 ^α	.2269 ^{αβγ}
GOV2	.3234	.3520 ^α	.3656 ^α	.3924 ^{αβγ}

Experimental Results : Multi-term expansion

- Negligible improvement in avg. precision
- Strong correlation exists b/w 2 word concepts and 1 word
 - Example: choosing “stock market” while “stock” & “market” were already chosen
 - Hence not much improvement
 - Novelty, diversity, term correlation etc should be taken into account

Term dependence

- Syntactic dependence
 - Covers phrases, term proximity, term co-occurrence
 - Implicit/Explicit positional dependence
- Semantic dependence
 - Relevance feedback, pseudo-relevance feedback, synonyms, and stemming
- Both are orthogonal dependencies in most cases
 - MRF captures syntactic dependence
 - LCE captures semantic dependence
- Thus, MRF improvements not lost in LCE

Conclusion

- Robustness
 - “The number of queries hurt/improved as a result of applying the models”
- LCE improves effectiveness for 60-80% queries
 - Hence robust
- MRFs allow getting rid of bag of words type approaches
- Future work
 - document-side dependencies

Bibliography

- Latent Concept Expansion using Markov Random Fields, Donald Metzler and W.Bruce Croft, SIGIR'07
- A markov random field model for term dependencies, Donald Metzler and W.Bruce Croft, SIGIR'05
- Conditional Random Fields : Probabilistic Models for Segmenting and Labeling Sequential Data, John Lafferty, Andrew McCallum, Fernando Pereira
- Wikipedia
- Introduction to Information Retrieval, Christopher Manning, Prabhakar Raghavan, H. Schutze, Cambridge University Press, 2008.