

# Semantic Search (Part V)

[DAT640] Information Retrieval and Text Mining

Krisztian Balog

University of Stavanger

November 5, 2019

# Retrieval models<sup>1</sup>

- Unstructured retrieval models
  - LM, BM25, **SDM**
- Fielded retrieval models
  - MLM, BM25F, PRMS, **FSDM**

---

<sup>1</sup>Note: while we introduce SDM and FSDM in the context of entity retrieval, nothing in the models listed on this slide is specific to entities so all of them can be applied to (fielded) documents as well.

# Markov random field (MRF) models

- Models so far employed a bag-of-words representation of both entities and queries
  - The order of terms is ignored
- The *Markov random field* (MRF) model provides a sound theoretical framework for modeling term dependence
  - Term dependencies are represented as a Markov random field (undirected graph  $G$ )
  - The MRF ranking function is computed as a linear combination of feature functions over the set of cliques<sup>2</sup> in  $G$ :

$$P_{\Lambda}(e|q) \stackrel{\text{rank}}{=} \sum_{c \in \mathcal{C}_G} \lambda_c f(c)$$

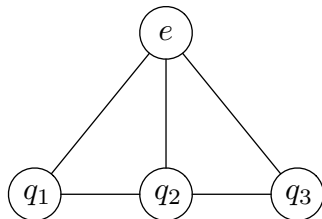
- MRF approaches belong to the more general class of linear feature-based models

---

<sup>2</sup>A clique is a subset of vertices of an undirected graph, such that every two distinct vertices are adjacent.

# Sequential Dependence Model (SDM)

- The *sequential dependence model* (SDM) is one particular instantiation of the MRF model
  - SDM assumes dependence between neighboring query terms
  - Strikes a good balance between effectiveness and efficiency
- Graph consists of an entity node  $e$  and query nodes  $q_i$
- Two types of cliques
  - Query term and the entity (unigram matches)
  - Two query terms and the entity; two variants:
    - The query terms occur contiguously (ordered bigram match)
    - They do not (unordered bigram match)



# Sequential Dependence Model (SDM)

- The SDM ranking function is given by a weighted combination of three feature functions
  - Query terms ( $f_T$ )
  - Exact match of query bigrams ( $f_O$ )
  - Unordered match of query bigrams ( $f_U$ )

$$\text{score}(e, q) = \lambda_T \sum_{i=1}^n f_T(q_i, e) + \lambda_O \sum_{i=1}^{n-1} f_O(q_i, q_{i+1}, e) + \lambda_U \sum_{i=1}^{n-1} f_U(q_i, q_{i+1}, e)$$

- The query is represented as a sequence of terms  $q = \langle q_1, \dots, q_n \rangle$
- Feature weights are subject to the constraint  $\lambda_T + \lambda_O + \lambda_U = 1$ 
  - Recommended default setting:  $\lambda_T = 0.85$ ,  $\lambda_O = 0.1$ , and  $\lambda_U = 0.05$

# Feature functions

- Feature functions are based on language modeling estimates using Dirichlet prior smoothing
- *Unigram matches* are based on smoothed entity language models:

$$f_T(q_i, e) = \log P(q_i | \theta_e)$$

## Feature functions (cont'd)

- *Ordered bigram matches:*

$$f_O(q_i, q_{i+1}, e) = \log \left( \frac{c_o(q_i, q_{i+1}, e) + \mu P_o(q_i, q_{i+1} | \mathcal{E})}{l_e + \mu} \right)$$

- $c_o(q_i, q_{i+1}, e)$  denotes the number of times the terms  $q_i, q_{i+1}$  occur in this exact order in the description of  $e$
- $l_e$  is the length of the entity's description (number of terms)
- $\mathcal{E}$  is the entity catalog (set of all entities)
- $\mu$  is the smoothing parameter
- The background language model is a maximum likelihood estimate:

$$P_o(q_i, q_{i+1} | \mathcal{E}) = \frac{\sum_{e \in \mathcal{E}} c_o(q_i, q_{i+1}, e)}{\sum_{e \in \mathcal{E}} l_e} .$$

## Feature functions (cont'd)

- *Unordered bigram matches:*

$$f_U(q_i, q_{i+1}, e) = \log \left( \frac{c_w(q_i, q_{i+1}, e) + \mu P_w(q_i, q_{i+1} | \mathcal{E})}{l_e + \mu} \right) ,$$

- $c_w(q_i, q_{i+1}; e)$  counts the co-occurrence of terms  $q_i$  and  $q_{i+1}$  in  $e$ , within an unordered window of  $w$  term positions
- Typically, a window size of 8 is used (corresponds roughly to sentence-level proximity)
- $l_e$  is the length of the entity's description (number of terms)
- $\mathcal{E}$  is the entity catalog (set of all entities)
- $\mu$  is the smoothing parameter
- The background language model is a maximum likelihood estimate:

$$P_w(q_i, q_{i+1} | \mathcal{E}) = \frac{\sum_{e \in \mathcal{E}} c_w(q_i, q_{i+1}, e)}{\sum_{e \in \mathcal{E}} l_e} .$$



# Discussion

## Question

How would you implement the SDM scoring function on top of Elasticsearch?

# Exercise #1

- Counting ordered and unordered bigram matches with Elasticsearch
- Code skeleton on GitHub: `exercises/lecture_19/exercise_1.ipynb`  
(make a local copy)

# Fielded Sequential Dependence Model (FSDM)

- Idea: base the feature function estimates on term/bigram frequencies combined across multiple fields (in the spirit of MLM and BM25F)
- The *fielded sequential dependence model* (FSDM) we present here combines SDM and MLM
- *Unigram matches* are MLM-estimated probabilities:

$$f_T(q_i, e) = \log \sum_{f \in \mathcal{F}} w_f^T P(t | \theta_{f_e})$$

- $w_f^T$  are the field mapping weights (for each field)

## Feature functions (cont'd)

- *Unordered bigram matches:*

$$f_O(q_i, q_{i+1}, e) = \log \sum_{f \in \mathcal{F}} w_f^O \frac{c_o(q_i, q_{i+1}, f_e) + \mu_f P_o(q_i, q_{i+1} | f_{\mathcal{E}})}{l_{f_e} + \mu_f}$$

- *Ordered bigram matches:*

$$f_U(q_i, q_{i+1}, e) = \log \sum_{f \in \mathcal{F}} w_f^U \frac{c_u^w(q_i, q_{i+1}, f_e) + \mu_f P_u^w(q_i, q_{i+1} | f_{\mathcal{E}})}{l_{f_e} + \mu_f}$$

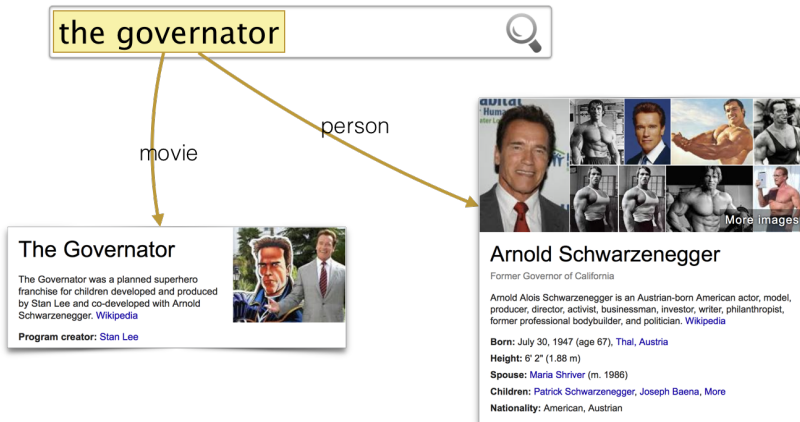
- All background models and smoothing parameters are made field-specific
  - But the same smoothing parameter ( $\mu_f$ ) may be used for all types of matches
- $w_f^O$  and  $w_f^U$  are the field mapping weights (for each field)
  - May be based on the field mapping probability estimates from PRMS

## Entity linking + retrieval


# Entity linking in queries

- Challenges
  - Search queries are short
  - Limited context
  - Lack of proper grammar, spelling
  - Multiple interpretations
  - Needs to be fast

# Example



# Example

the governor movie 



## The Governor

The Governor was a planned superhero franchise for children developed and produced by Stan Lee and co-developed with Arnold Schwarzenegger. [Wikipedia](#)

Program creator: [Stan Lee](#)





# Example

new york pizza manhattan



# Example

new york pizza manhattan



## New York-style pizza

New York-style pizza is characterized by large hand-tossed thin-crust pies, often sold in wide slices to-go. [Wikipedia](#)



# Discussion

## Question

Can we incorporate entity annotations of queries into entity retrieval models?

# Dual term-based and entity-based entity representations

Idea: preserve these entity references by employing a *dual entity representation*: on top of the traditional term-based representation, each entity is additionally represented by means of its associated entities.

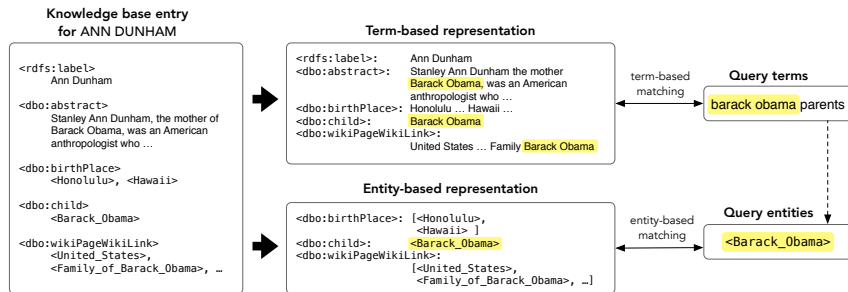
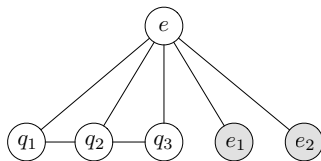


Figure: Illustration is based on (Hasibi et al., 2016).

# Entity linking incorporated retrieval (ELR)

- Entity-specific extension applied on top of the MRF framework
  - May be applied on top of any term-based retrieval model that can be instantiated in the MRF framework, but here we will focus on SDM
- The underlying graph representation of SDM is extended with query entity nodes
  - It is assumed that entities have already been identified (linked) in queries and in the descriptions of entities
- New type of clique: 2-cliques between the given entity (that is being scored) and the query entities



# Ranking function SDM+ELR

- The SDM+ELR ranking function is given by a weighted combination of four feature functions
  - Query terms ( $f_T$ )
  - Exact match of query bigrams ( $f_O$ )
  - Unordered match of query bigrams ( $f_U$ )
  - **Entity matches** ( $f_{\mathcal{E}}$ )

$$\begin{aligned} \text{score}(e, q) = & \lambda_T \sum_{i=1}^n f_T(q_i, e) + \lambda_O \sum_{i=1}^{n-1} f_O(q_i, q_{i+1}, e) \\ & + \lambda_U \sum_{i=1}^{n-1} f_U(q_i, q_{i+1}, e) + \lambda_{\mathcal{E}} \sum_{i=1}^m f_{\mathcal{E}}(e_i; e) \end{aligned}$$

- Issue: the number of entities ( $m$ ) varies per query! How can  $\lambda_{\mathcal{E}}$  be trained?

# Parameterizing weights

- Rewriting  $\lambda$  parameters as parameterized functions over each clique:

$$\begin{aligned}\lambda_T(q_i) &= \lambda_T \frac{1}{n} , \\ \lambda_O(q_i, q_{i+1}) &= \lambda_O \frac{1}{n-1} , \\ \lambda_U(q_i, q_{i+1}) &= \lambda_U \frac{1}{n-1} , \\ \lambda_{\mathcal{E}}(e_i) &= \lambda_{\mathcal{E}} \frac{w(e_i, q)}{\sum_{j=1}^m w(e_j, q)} .\end{aligned}$$

- The weight  $w(e_i, q)$  reflects the confidence in that entity annotation (entity linker's confidence score)

# Ranking function SDM+ELR

- Final ranking function using the parameterized weights:

$$\begin{aligned} \text{score}(e, q) = & \frac{\lambda_T}{n} \sum_{i=1}^n f_T(q_i; e) \\ & + \frac{\lambda_O}{n-1} \sum_{i=1}^{n-1} f_O(q_i, q_{i+1}; e) \\ & + \frac{\lambda_U}{n-1} \sum_{i=1}^{n-1} f_U(q_i, q_{i+1}; e) \\ & + \frac{\lambda_{\mathcal{E}}}{\sum_{j=1}^m w(e_j, q)} \sum_{i=1}^m w(e_i, q) f_{\mathcal{E}}(e_i; e) \end{aligned}$$

- Default setting:  $\lambda_T = 0.8$ ,  $\lambda_O = 0.05$ ,  $\lambda_U = 0.05$ , and  $\lambda_{\mathcal{E}} = 0.1$



# Feature function for entity matches

- Differences from term-based scoring
  - We assume that each entity appears at most once in each field
  - If a query entity appears in a field, then it shall be regarded as a “perfect match,” independent of what other entities are present in the same field
  - Unlike for terms-based representations, predicate folding is not performed ()

$$f_{\mathcal{E}}(e_i; e) = \log \sum_{f \in \tilde{\mathcal{F}}} w_f^{\mathcal{E}} \left( (1 - \lambda) \mathbb{1}(e_i, f_{\tilde{e}}) + \lambda \frac{\sum_{e' \in \mathcal{E}} \mathbb{1}(e_i, f_{\tilde{e}'})}{|\{e' \in \mathcal{E} : f_{\tilde{e}'} \neq \emptyset\}|} \right)$$

- $\tilde{e}$  denote the entity-based representation of entity  $e$
- $\tilde{\mathcal{F}}$  denotes the set of fields in the entity-based representation
- $\mathbb{1}(e, f_{\tilde{e}})$  is a binary indicator function, which is 1 if  $e_i$  is present in the entity field  $f_{\tilde{e}}$  and otherwise 0
- $\lambda$  is the smoothing parameter (default: 0.1)
- Field weights  $w_f^{\mathcal{E}}$  may be set manually or via dynamic mapping using PRMS

# Assignment 3

# Reading

- Entity-Oriented Search (Balog)<sup>3</sup>
  - Section 3.3
  - Section 4.2.2

---

<sup>3</sup>PDF: <https://rd.springer.com/content/pdf/10.1007%2F978-3-319-93935-3.pdf>