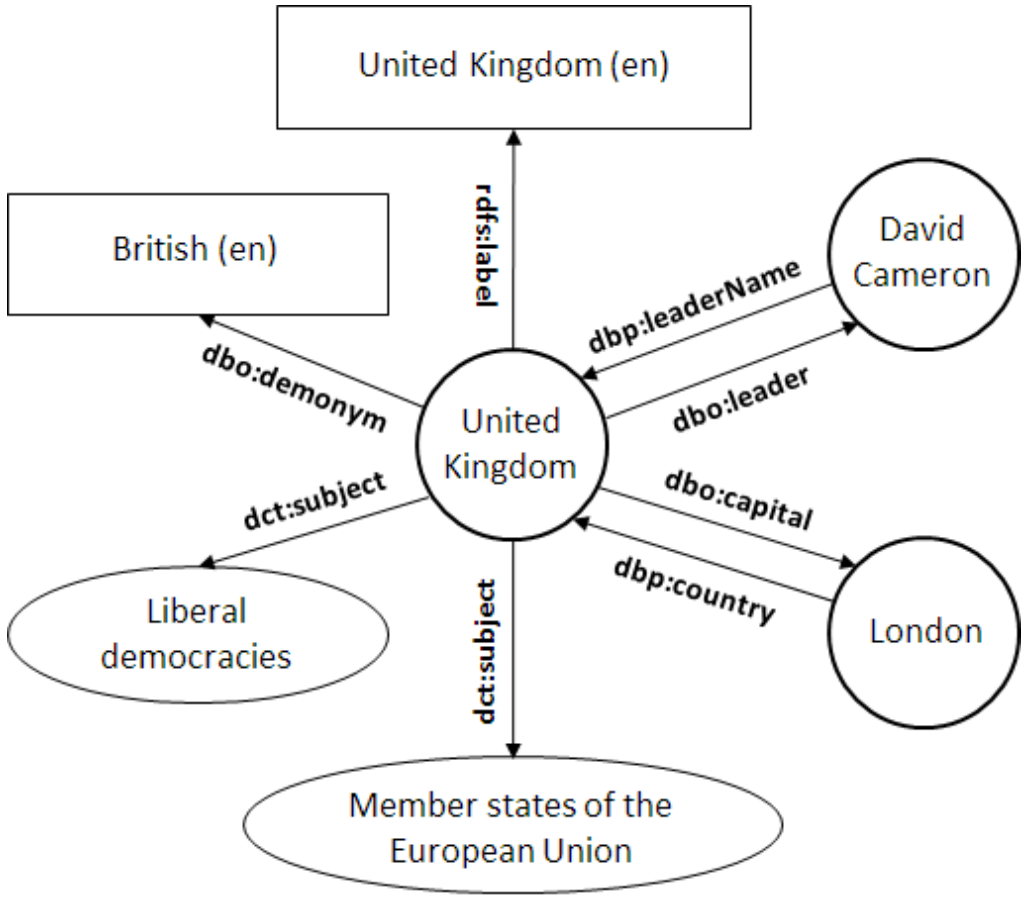
Entity Representation and Retrieval

Laura Dietz
University of New Hampshire

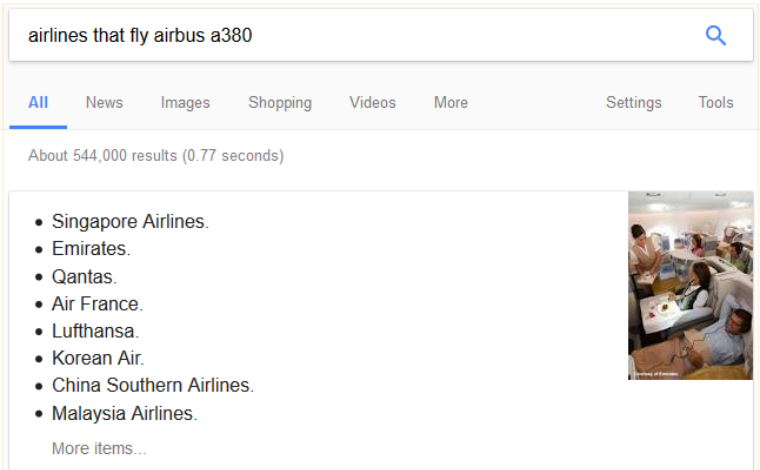
Alexander Kotov
Wayne State University

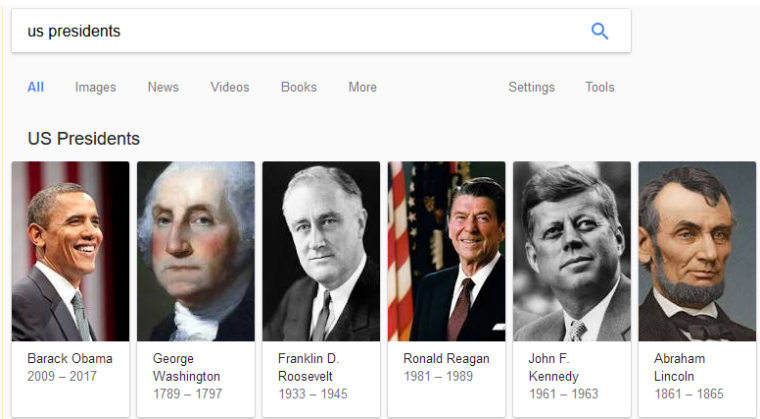
Edgar Meij
Bloomberg L.P.

Knowledge Graph Fragment



Entity Retrieval





- Users often search for concrete or abstract objects (i.e. people, products or locations), rather than documents
- Users are willing to express their information need more elaborately than with a few keywords [Balog et al. 2008]
- Search results are names of entities or entity representations (i.e. entity cards)
- Knowledge graphs are perfectly suited for entity retrieval

Entity Retrieval Tasks

- **Entity Search**: simple queries aimed at finding a particular entity or an entity which is an attribute of another entity
  - *"Ben Franklin"*
  - *"Einstein Relativity theory"*
  - *"England football player highest paid"*
- **List Search**: descriptive queries with several relevant entities
  - *"US presidents since 1960"*
  - *"animals lay eggs mammals"*
  - *"Formula 1 drivers that won the Monaco Grand Prix"*
- **Question Answering**: queries are questions in natural language
  - *"Who founded Intel?"*
  - *"For which label did Elvis record his first album?"*

Entity Retrieval from Knowledge Graph(s) (ERKG)

- Different from ad-hoc entity retrieval, which is focused on retrieving entities embedded in documents, e.g:
  - Entity track at TREC 2009–2011
  - Entity Ranking track at INEX 2007–2009
  - Expert Finding in Enterprise Search
- Different from entity linking, which aims at identifying entities mentioned in queries (part 1 of this tutorial)
- Can be combined with methods using KGs for ad-hoc or Web search (part 3 of this tutorial)

Why ERKG?

- **Unique IR problem:** there are *no documents*
- **Challenging IR problem:** knowledge graphs are designed for graph pattern-based SPARQL queries

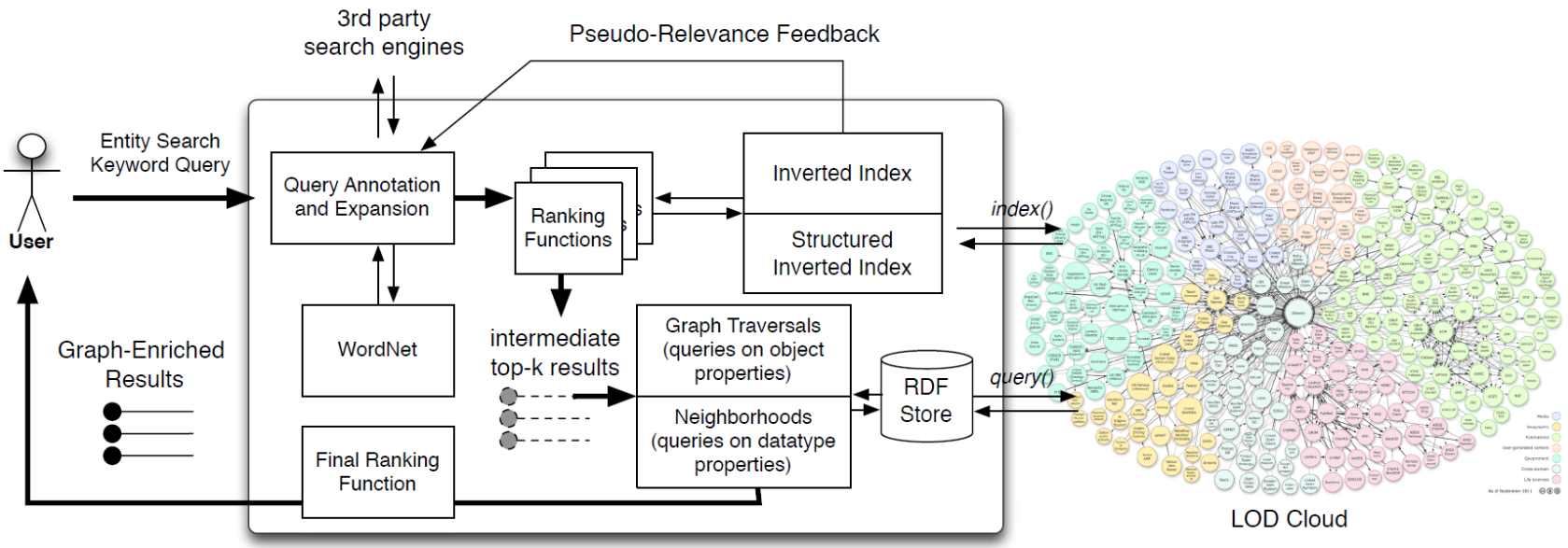
Research challenges in ERKG

ERKG requires accurate interpretation of unstructured textual queries and matching them with entity semantics:

1. How to design entity representations that capture the semantics of entity properties and relations to other entities?
2. How to develop accurate and efficient entity retrieval models?

Architecture of ERKG Methods

[Tonon, Demartini et al., SIGIR'12]

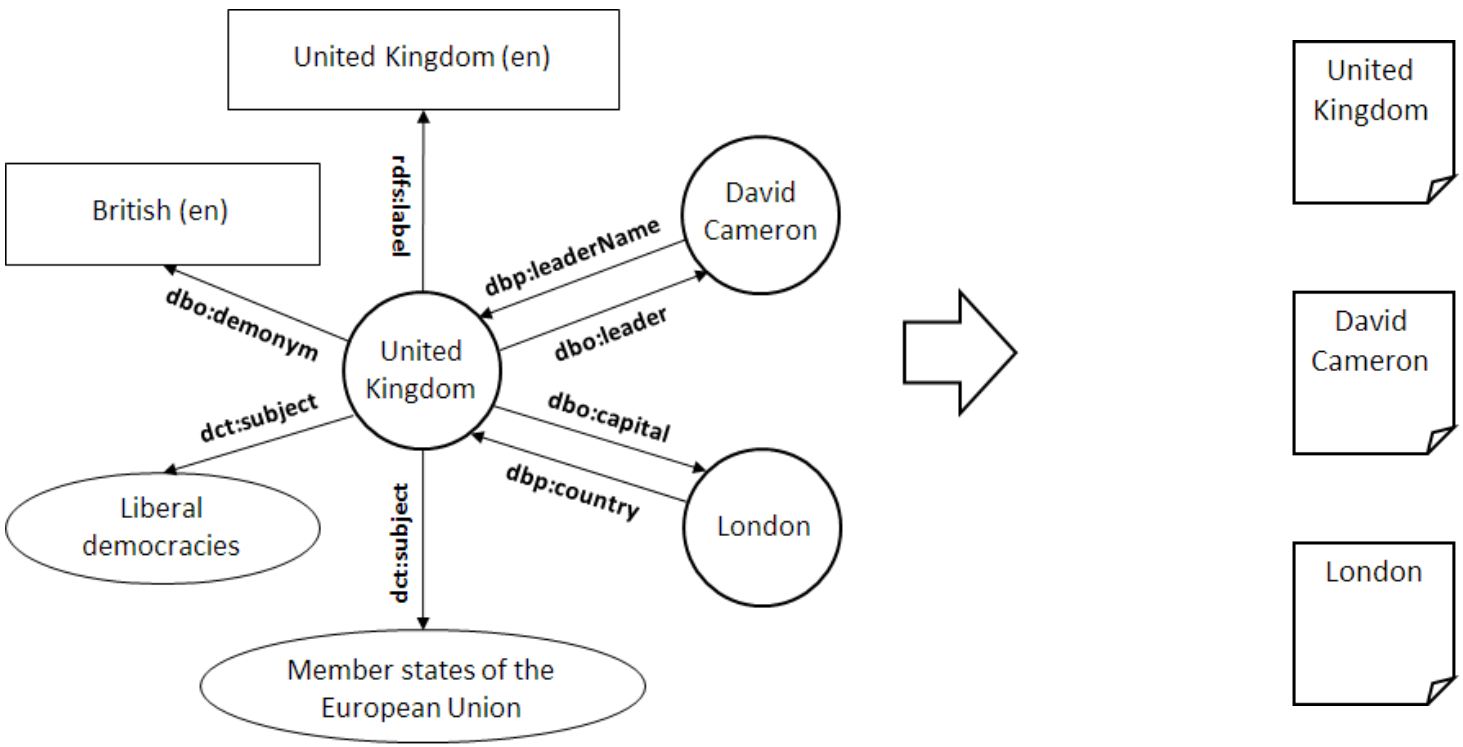


Outline

- Entity representation
- Entity retrieval
- Entity set expansion
- Entity ranking

Structured Entity Documents

Build a textual representation (i.e. "document") for each entity by considering all triples, where it stands as a subject (or object)



Predicate Folding

- **Simple approach:** each predicate corresponds to one entity document field
- **Problem:** there are infinitely many predicates $\rightarrow$ optimization of field importance weights is computationally intractable
- **Predicate folding:** group predicates into a small set of predefined categories $\rightarrow$ entity documents with smaller number of fields
  - by predicate type (attributes, incoming/outgoing links)[Pérez-Agüera et al. 2010]
  - by predicate importance (determined based on predicate popularity)[Blanco et al. 2010]
- The number and type of fields depends on a retrieval task

Predicate Folding Example

names	**rdfs:label**	United Kingdom (en)
	foaf:name	United Kingdom (en)
attributes	**dbo:demonym**	British (en)
	dbo:foundingDate	1707-05-01
categories	**dct:subject**	dbc:Member_states_of_the_European_Union
	dct:subject	dbc:Liberal_democracies
outgoing relations	**dbo:leader**	dbr:David_Cameron
	dbo:capital	dbr:London
incoming relations	is **dbo:country** of	dbr:London
	is **dbp:leaderName** of	dbr:David_Cameron

2-field Entity Document

[Neumayer, Balog et al., ECIR'12]

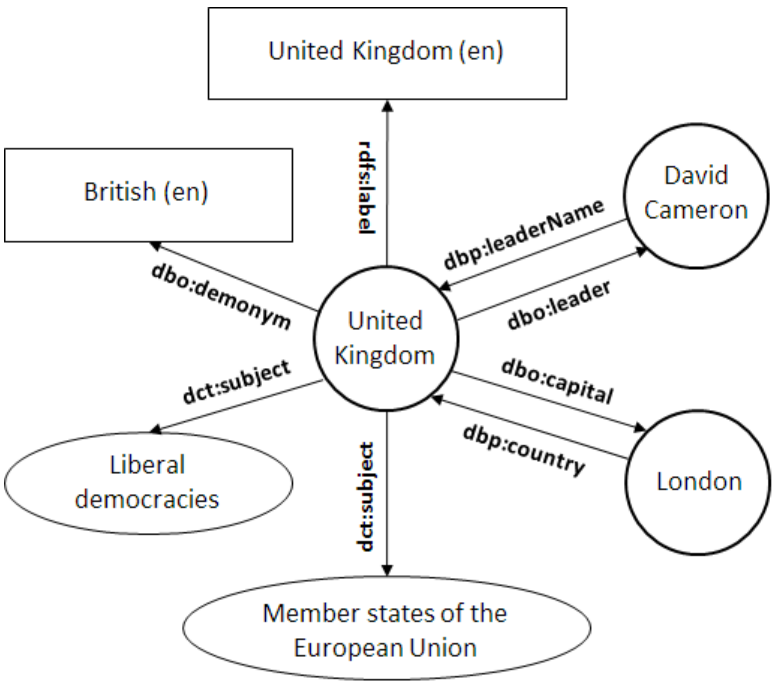
Each entity is represented as a two-field document:

title
: object values belonging to predicates ending with "name", "label" or "title"

content
: object values for 1000 most frequent predicates concatenated together into a flat text representation

This simple scheme is effective for entity retrieval

2-field Entity Document Example



title	united kingdom
content	british founding date 1707-05-01 united kingdom great britain northern ireland capital london leader david cameron

3-field Entity Document

[Zhiltsov and Agichtein, CIKM'13]

Each entity is represented as a three-field document:

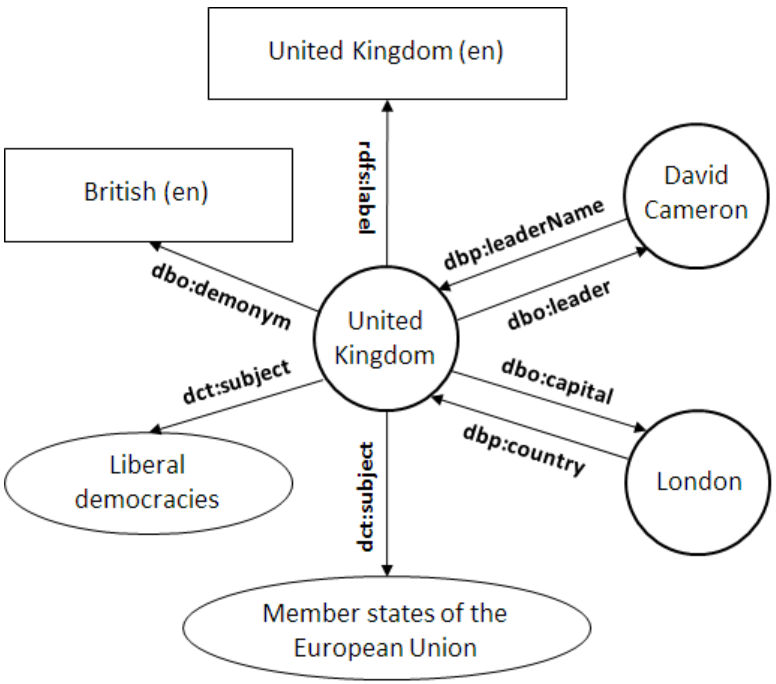
names
: literals of `foaf:name`, `rdfs:label` predicates along with tokens extracted from entity URIs

attributes
: literals of all other predicates

outgoing links
: names of entities in the object position

This scheme is effective for entity retrieval

3-field Entity Document Example



names	united kingdom
attributes	british founding date 1707-05-01
outgoing links	united kingdom great britain northern ireland capital london leader david cameron

5-field Entity Document

[Zhiltsov, Kotov et al., SIGIR'15]

Each entity is represented as a five-field document:

names
: labels or names of entities

attributes
: all entity properties, other than names

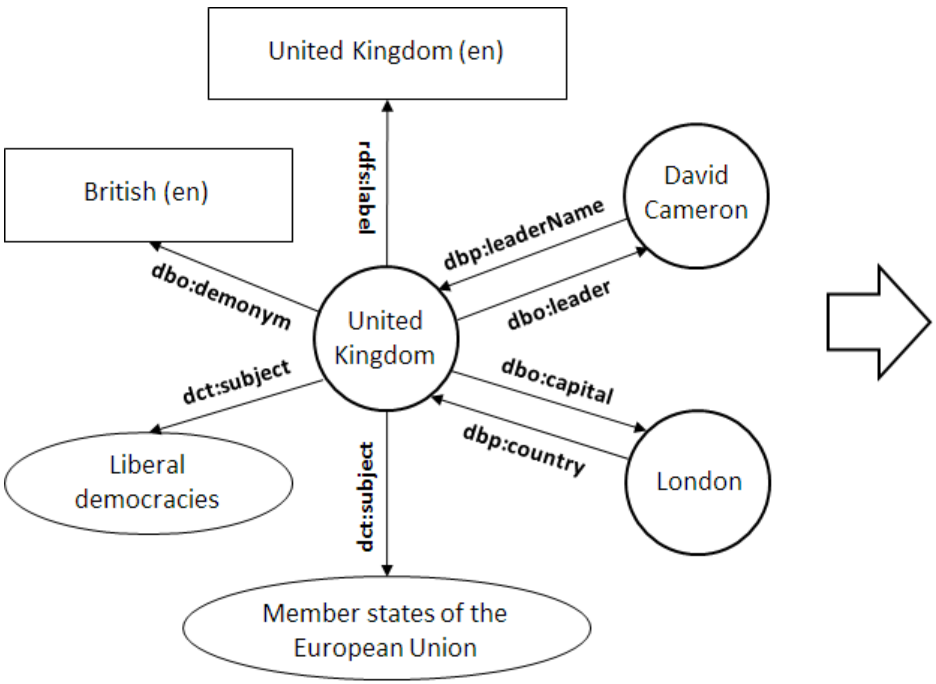
categories
: classes or groups, to which the entity has been assigned

similar entity names
: names of the entities that are very similar or identical to a given entity

related entity names
: names of entities in the object position

This flexible scheme is effective for a variety of tasks: entity search, list search, question answering

5-field Entity Document Example



names	united kingdom
attributes	british founding date 1707-05-01
categories	member state european union liberal democracy
similar entity names	united kingdom great britain northern ireland
related entity names	capital london leader david cameron

Problems with Entity Representations

- Vocabulary mismatch between relevant entity(ies) description(s) and the query terms that can be used to search for it(them)
- Associations between words and entities depend on the context:
  - `Germany` should be returned for queries related to World War II and 2014 Soccer World Cup
- Real-life events change perception of entities:
  - `Ferguson, Missouri` before and after August 2014

Dynamic Entity Representation

[Graus, Tsagkias et al., WSDM'16]

Idea: create static entity representations using knowledge bases and leverage different social media sources to dynamically update them

- Represent entities as fielded documents, in which each field corresponds to different source
- Tweak the weights of different fields over time

Static Sources

Dynamic Sources

biggest penguin
anthropornis
extinct penguin
prehistoric birds

megafauna

Dynamic sources

Queries
Queries from MSN query logs that yield Wikipedia clicks.
47,002 descriptions for 18,724 entities.

Tweets
Tweets w/ links to Wikipedia pages (2011-2014)
52,631 descriptions for 38,269 entities.

Social tags
Delicious tags for Wiki pages, from the SocialBM0311 corpus.
4.4M descriptions for 289,015 entities.

Method

- Single-field ranker supervised by clicks learns the optimal feature weights
- Features:
  - **Field similarity**: TF-IDF cosine similarity of query and field f at time t_i
  - **Field importance**: field's length, new terms, number of updates
  - **Entity importance**: time since last entity description update

Results

- Social tags are the best performing single entity description source
- KB+queries yield substantial relative improvement $\rightarrow$ queries provide a strong signal for entity ranking
- As new content comes in, it is beneficial to retrain the ranker
- Rankers that incorporate dynamic description sources (i.e KB+tags, KB+tweets and KB+queries) show the highest learning rate $\rightarrow$ entity content from these sources accounts for changes in entity representations over time

Outline

- Entity representation
- Entity retrieval
- Entity set expansion
- Entity ranking

Fielded Sequential Dependence Model

[Zhiltsov, Kotov et al., SIGIR'15]

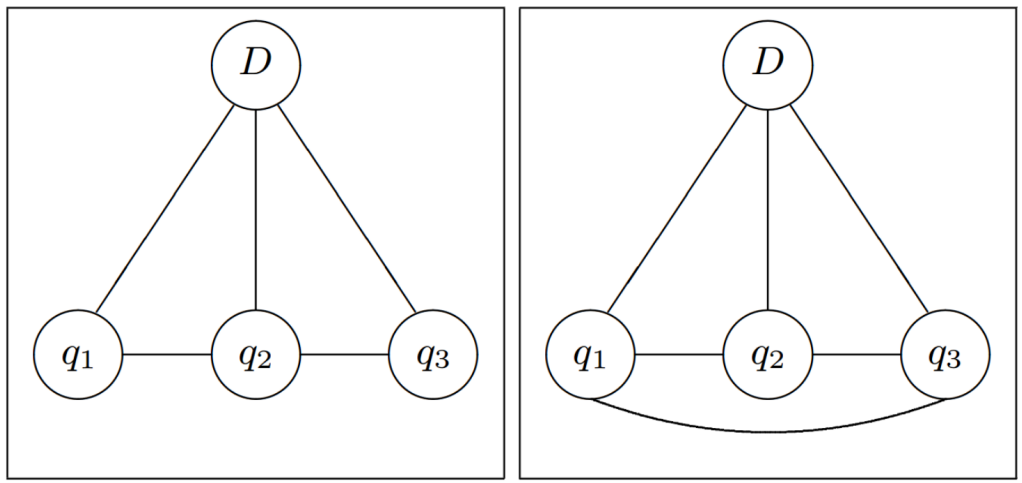
Two major directions in ad-hoc IR:

- unigram bag-of-words retrieval models for structured documents
  - Ogilvie and Callan. Combining Document Representations for Known-item Search, SIGIR'03 (MLM)
  - Robertson et al. Simple BM25 Extension to Multiple Weighted Fields, CIKM'04 (BM25F)
  - Kim and Croft. A Probabilistic Retrieval Model for Semistructured Data, ECIR'09 (PRMS)
- retrieval models incorporating term dependencies (bigrams)
  - Metzler and Croft. A Markov Random Field Model for Term Dependencies, SIGIR'05 (SDM)

Idea: model for structured document retrieval that captures term dependencies

Sequential and Full Dependence Models

[Metzler and Croft, SIGIR'05]



Document score is a linear combination of matching functions for unigrams and bigrams *in the entire document LM*:

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

SDM only considers only sequential bigrams, FDM all query term pairs

Fielded Sequential Dependence Model

[Zhiltsov, Kotov et al., SIGIR'15]

Document score is a linear combination of matching functions for unigrams and bigrams *in each document field*:

$$P_\Lambda(D|Q) \overset{rank}{=} \lambda_T \sum_{q \in Q} \tilde{f}_T(q_i, D) + \lambda_O \sum_{q \in Q} \tilde{f}_O(q_i, q_{i+1}, D) + \lambda_U \sum_{q \in Q} \tilde{f}_U(q_i, q_{i+1}, D)$$

MLM is a special case of FSDM, when $\lambda_T = 1, \lambda_O = 0, \lambda_U = 0$

Fielded Sequential Dependence Model

[Zhiltsov, Kotov et al., SIGIR'15]

Document score is a linear combination of matching functions for unigrams and bigrams *in each document field*:

$$P_\Lambda(D|Q) \overset{rank}{=} \lambda_T \sum_{q \in Q} \tilde{f}_T(q_i, D) + \lambda_O \sum_{q \in Q} \tilde{f}_O(q_i, q_{i+1}, D) + \lambda_U \sum_{q \in Q} \tilde{f}_U(q_i, q_{i+1}, D)$$

MLM is a special case of FSDM, when $\lambda_T = 1, \lambda_O = 0, \lambda_U = 0$

Fielded Sequential Dependence Model

[Zhiltsov, Kotov et al., SIGIR'15]

Document score is a linear combination of matching functions for unigrams and bigrams *in each document field*:

$$P_\Lambda(D|Q) \overset{rank}{=} \lambda_T \sum_{q \in Q} \tilde{f}_T(q_i, D) + \lambda_O \sum_{q \in Q} \tilde{f}_O(q_i, q_{i+1}, D) + \lambda_U \sum_{q \in Q} \tilde{f}_U(q_i, q_{i+1}, D)$$

MLM is a special case of FSDM, when $\lambda_T = 1, \lambda_O = 0, \lambda_U = 0$

Fielded Sequential Dependence Model

[Zhiltsov, Kotov et al., SIGIR'15]

Document score is a linear combination of matching functions for unigrams and bigrams *in each document field*:

$$P_\Lambda(D|Q) \overset{rank}{=} \lambda_T \sum_{q \in Q} \tilde{f}_T(q_i, D) + \lambda_O \sum_{q \in Q} \tilde{f}_O(q_i, q_{i+1}, D) + \lambda_U \sum_{q \in Q} \tilde{f}_U(q_i, q_{i+1}, D)$$

MLM is a special case of FSDM, when $\lambda_T = 1, \lambda_O = 0, \lambda_U = 0$

FSDM ranking function

FSDM matching function for unigrams:

$$\tilde{f}_T(q_i, D) = \log \sum_j w_j^T P(q_i|\theta_D^j) = \log \sum_j w_j^T \frac{tf_{q_i,D^j} + \mu_j \frac{cf_{q_i}^j}{|C_j|}}{|D^j| + \mu_j}$$

Example:

apollo astronauts who walked on the moon

Parameters:

1. Field importance weights for unigrams and bigrams
2. Relative importance weights of matching unigrams and bigrams

FSDM ranking function

FSDM matching function for unigrams:

$$\tilde{f}_T(q_i, D) = \log \sum_j w_j^T P(q_i|\theta_D^j) = \log \sum_j w_j^T \frac{tf_{q_i,D^j} + \mu_j \frac{cf_{q_i}^j}{|C_j|}}{|D^j| + \mu_j}$$

Example:

apollo astronauts who walked on the moon
category

Parameters:

1. Field importance weights for unigrams and bigrams
2. Relative importance weights of matching unigrams and bigrams

FSDM ranking function

FSDM matching function for unigrams:

$$\tilde{f}_T(q_i, D) = \log \sum_j w_j^T P(q_i|\theta_D^j) = \log \sum_j w_j^T \frac{tf_{q_i,D^j} + \mu_j \frac{cf_{q_i}^j}{|C_j|}}{|D^j| + \mu_j}$$

Example:

apollo astronauts who walked on the moon
category category

Parameters:

1. Field importance weights for unigrams and bigrams
2. Relative importance weights of matching unigrams and bigrams

Performance

- 20%/52% improvement in MAP over MLM/SDM for entity search (SemSearch ES)
- 7%/3% improvement in MAP over MLM/SDM for list search (INEX-XER, TREC Entity, SemSearch LS)
- 28%/6% improvement in MAP over MLM/SDM for question answering (QALD-2)
- 18%/20% improvement in MAP over MLM/SDM for all queries

FSDM limitation

Same field weights for all query unigrams and all query bigrams

Example:

capitals in Europe which were host cities of summer Olympic games

FSDM limitation

Same field weights for all query unigrams and all query bigrams

Example:

capitals in Europe which were host cities of summer Olympic games
category

FSDM limitation

Same field weights for all query unigrams and all query bigrams

Example:

capitals in Europe which were host cities of summer Olympic games
category attribute

FSDM limitation

Same field weights for all query unigrams and all query bigrams

Example:

capitals in Europe which were host cities of summer Olympic games
category attribute category

Parametric extension of FSDM

[Nikolaev, Kotov et al., SIGIR'16]

Idea: calculate field weight for each unigram and bigram based on features:

$$w^T_{q_i,j} = \sum_k \alpha^U_{j,k} \phi_k(q_i, j)$$

- $\phi_k(q_i, j)$ is the the k-th feature value for unigram q_i in field j
- $\alpha^U_{j,k}$ are feature weights that are learned by coordinate ascent to maximize target retrieval metric

Parametric extension of FSDM

[Nikolaev, Kotov et al., SIGIR'16]

Idea: calculate field weight for each unigram and bigram based on features:

$$w^T_{q_i,j} = \sum_k \alpha^U_{j,k} \phi_k(q_i, j)$$

- $\phi_k(q_i, j)$ is the the k-th feature value for unigram q_i in field j
- $\alpha^U_{j,k}$ are feature weights that are learned by coordinate ascent to maximize target retrieval metric

Parametric extension of FSDM

[Nikolaev, Kotov et al., SIGIR'16]

Idea: calculate field weight for each unigram and bigram based on features:

$$w^T_{q_i,j} = \sum_k \alpha^U_{j,k} \phi_k(q_i, j)$$

- $\phi_k(q_i, j)$ is the the k-th feature value for unigram q_i in field j
- $\alpha^U_{j,k}$ are feature weights that are learned by coordinate ascent to maximize target retrieval metric

Features

Source	Feature	Description	CT
Collection statistics	$FP(\kappa, j)$	Posterior probability $P(E_j\|w)$.	UG BG
	$TS(\kappa, j)$	Top SDM score on j-th field when κ is used as a query.	BG

Features

Source	Feature	Description	CT
Collection statistics	$FP(\kappa, j)$	Posterior probability $P(E_j\|w)$.	UG BG
	$TS(\kappa, j)$	Top SDM score on j-th field when κ is used as a query.	BG
Stanford POS Tagger	$NNP(\kappa)$	Is concept κ a proper noun?	UG
	$NNS(\kappa)$	Is κ a plural non-proper noun?	UG BG
	$JJS(\kappa)$	Is κ a superlative adjective?	UG
Stanford Parser	$NPP(\kappa)$	Is κ part of a noun phrase?	BG
	$NNO(\kappa)$	Is κ the only singular non-proper noun in a noun phrase?	UG
	INT	Intercept feature ($= 1$).	UG BG

Performance

- 7% improvement in MAP over FSDM for entity search queries (SemSearch ES)
- 12% improvement in MAP over FSDM for question answering (QALD-2)
- 6% improvement in MAP over FSDM for all queries

Outline

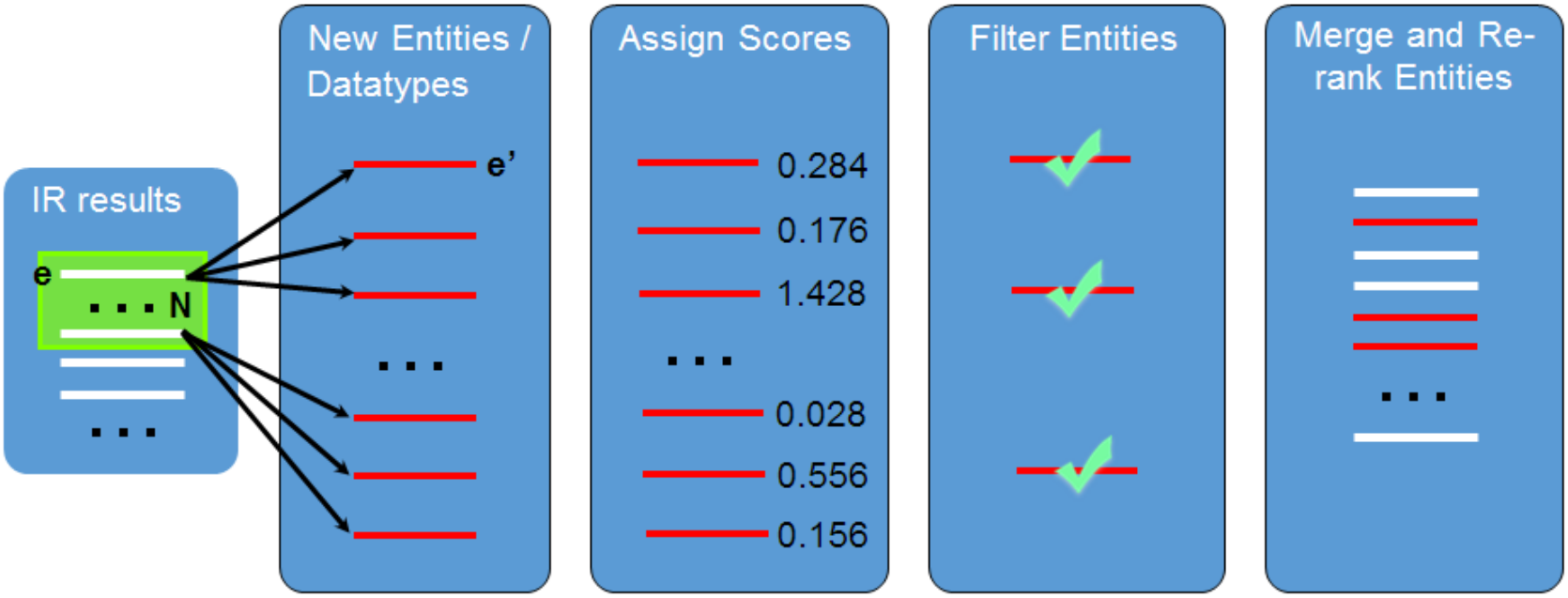
- Entity representation
- Entity retrieval
- Entity set expansion
- Entity ranking

Combining IR and Structured Search

[Tonon, Demartini et al., SIGIR'12]

- Maintain inverted index for entity representations and triple store for entity relations
- **Hybrid approach**: IR models for initial entity retrieval and SPARQL queries for expansion

Pipeline



score(q, e) (e.g. BM25(F), MLM, (P)FSDM)

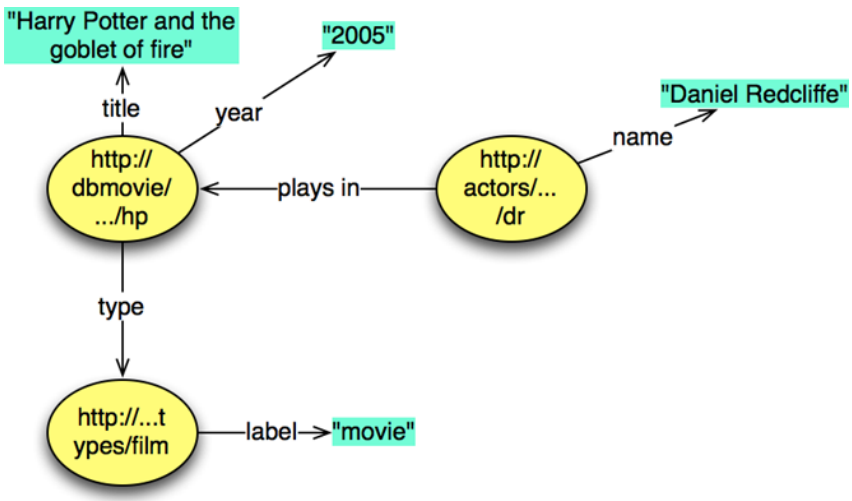
SPARQL queries to follow links/properties from original entities

sim(e', q) (e.g. Jaro-Winkler score)

sim(e', q) > δ ?

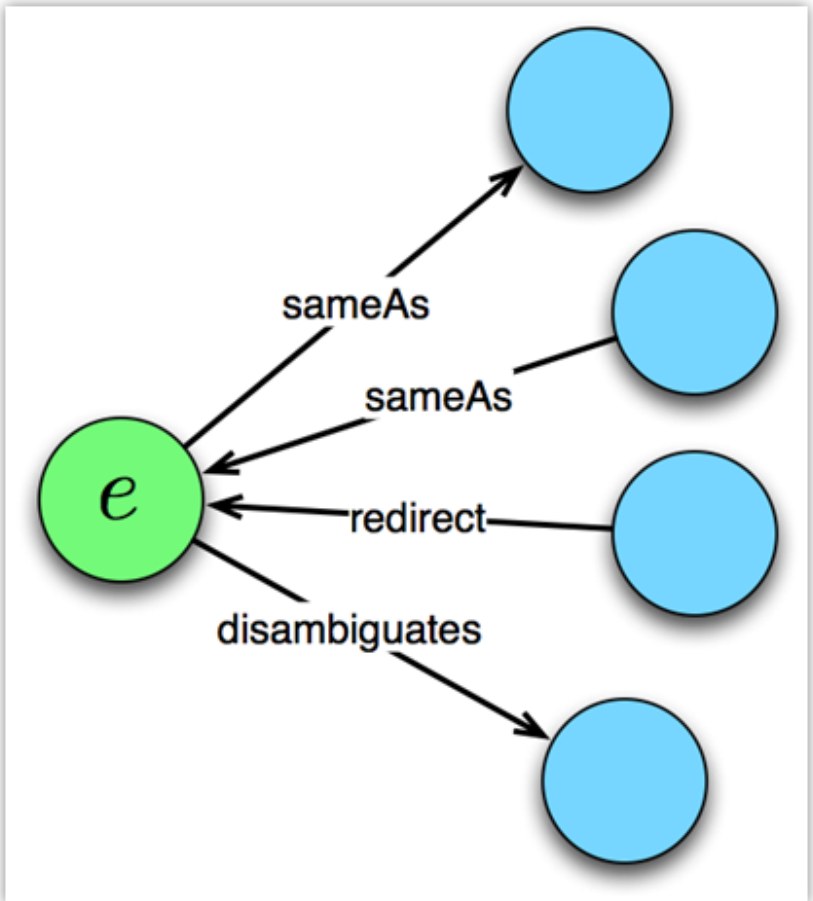
λ·score(q, e) + (1 - λ)·sim(e', q)

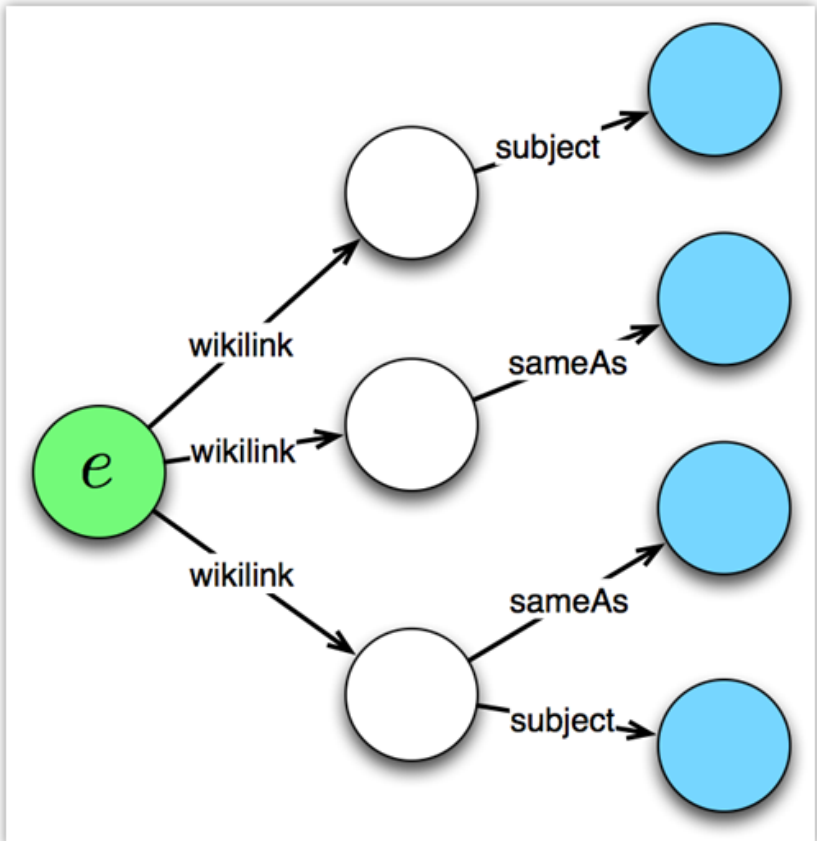
Result Expansion Strategies



- Follow predicates leading to other entities
- Follow predicates leading to entity attributes
- Explore entity neighbors and the neighbors of neighbors

Predicates to Follow





Results

	2010 Collection		2011 Collection	
	MAP	P@10	MAP	P@10
BM25	0.2070	0.3348	0.1484	0.2020
SAS	0.2293* (+11%)	0.363* (+8%)	0.1612 (+9%)	**0.2200** (+9%)
SAS+DIS+RED	**0.2586*** (+25%)	**0.3848*** (+15%)	**0.1657** (+12%)	0.2140 (+6%)

- Best performing method exploits entity neighbors by following <owl:sameAs> (SAS) as well as <dbpedia:redirect> (RED) and <dbpedia:disambiguates> predicates (DIS)
- Looking further into KG for related entities and following general predicates (<dbpedia:wikilink>, <skos:subject>, <foaf:homepage>, etc.) does not improve results (similar findings for terms graphs by [Kotov and Zhai, WSDM'12] and [Balaneshinkordan and Kotov, CIKM'16])

Outline

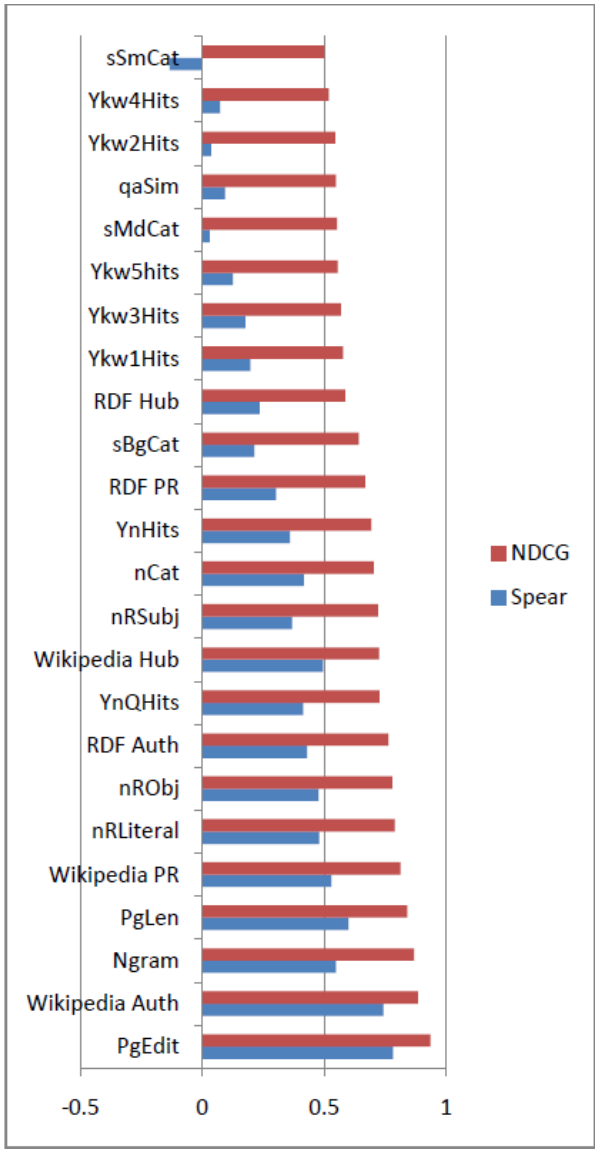
- Entity representation
- Entity retrieval
- Entity set expansion
- Entity ranking

Learning-to-Rank Entities

[Dali and Fortuna, WWW'11]

- Variety of features:
  - Popularity and importance of Wikipedia page: # of accesses from logs, # of edits, page length
  - RDF features: # of triples E is subject/object/subject and object is a literal, # of categories Wikipedia page for E belongs to, size of the biggest/smallest/median category
  - HITS scores and Pagerank of Wikipedia page and E in the RDF graph
  - # of hits from search engine API for the top 5 keywords from the abstract of Wikipedia page for E
  - Count of entity name in Google N-grams
- RankSVM learning-to-rank method

Feature Importance



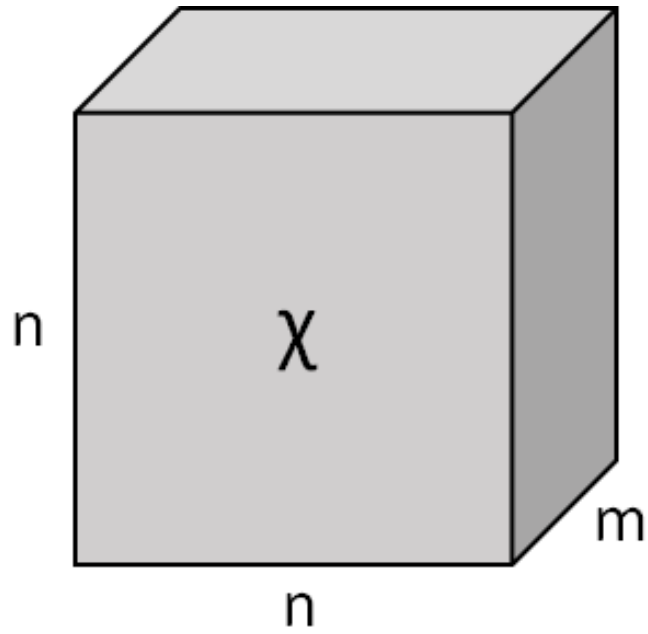
- Features approximating the entity importance (hub and authority scores, PageRank) of Wikipedia page are effective
- PageRank and HITS scores on RDF graph are not effective (outperformed by simpler RDF features)
- Google N-grams is effective proxy for entity popularity, cheaper than search engine API
- Feature combinations improve both robustness and accuracy of ranking

Latent Dimensional Representation

[Zhiltsov and Agichtein, CIKM'13]

- Compact representation of entities in low dimensional space by using a modified algorithm for tensor factorization
- Entities and entity-query pairs are represented with term-based and structural features

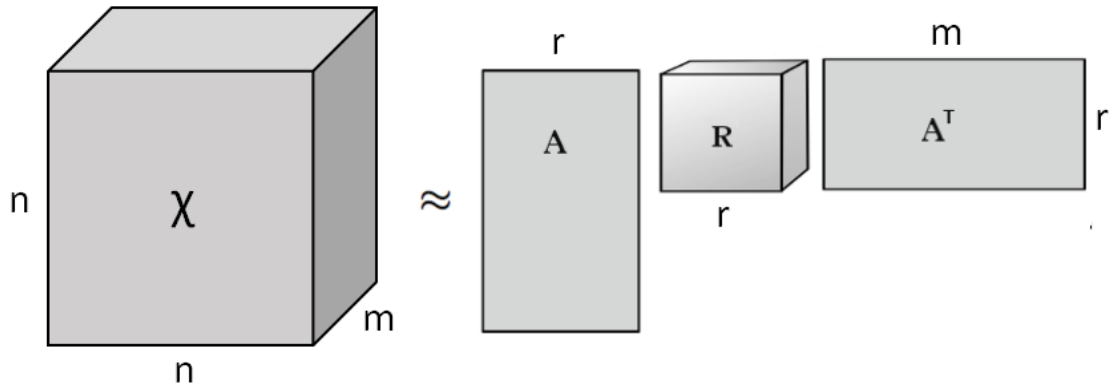
Knowledge Graph as Tensor



- For a knowledge graph with n distinct entities and m distinct predicates, we construct a tensor $\mathcal{X}$ of size $n \times n \times m$, where $\mathcal{X}_{ijk} = 1$, if there is k-th predicate between i-th entity and j-th entity, and $\mathcal{X}_{ijk} = 0$, otherwise
- Each k-th frontal tensor slice $\mathcal{X}_k$ is an adjacency matrix for the k-the predicate

RESCAL Tensor Factorization

[Nikel, Tresp, et al., WWW'12]



- Given r is the number of latent factors, factorize each X_k:

$$X_k = AR_kA^T, k = \overline{1, m},$$

where A is a dense $n \times r$ matrix, a matrix of latent embeddings for entities, and R_k is an $r \times r$ matrix of latent factors

Retrieval Method

1. Retrieve initial set of entities using MLM
2. Re-rank the entities using Gradient Boosted Regression Tree (GBRT)

Features

#	Feature
Term-based features	
1	Query length
2	Query clarity
3	Uniformly weighted MLM score
4	Bigram relevance score for the "name" field
5	Bigram relevance score for the "attributes" field
6	Bigram relevance score for the "outgoing links" field
Structural features	
7	Top-3 entity cosine similarity, $cos(\mathbf{e}, \mathbf{e}_{top})$
8	Top-3 entity Euclidean distance, $\|\mathbf{e} - \mathbf{e}_{top}\|$
9	Top-3 entity heat kernel, $e^{-\frac{\|\mathbf{e}-\mathbf{e}_{top}\|^2}{\sigma}}$

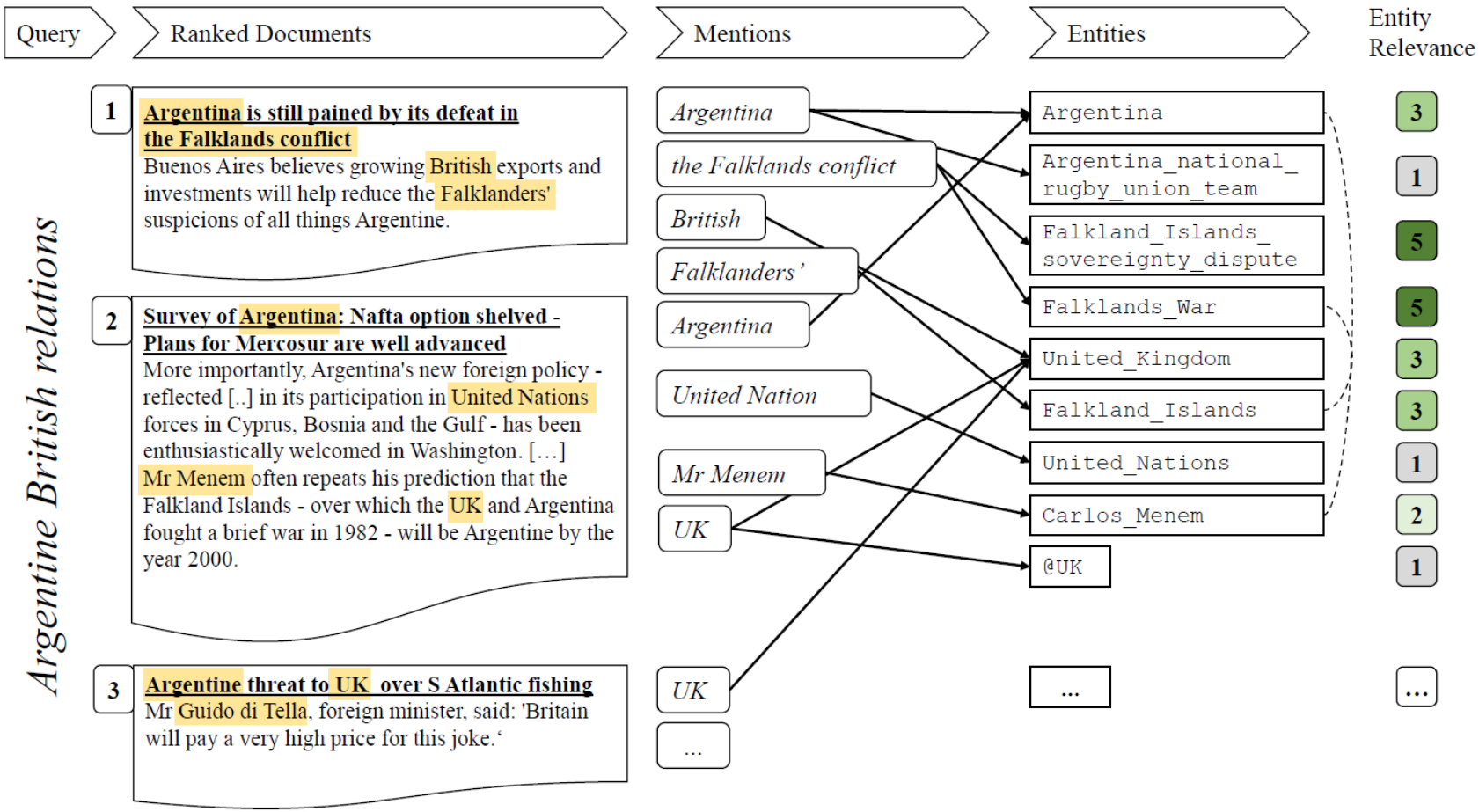
Results

Features	Performance		
	NDCG	**MAP**	**P@10**
Term-based baseline	0.382	0.265	0.539
All features	**0.401** (+ 5.0%)*	**0.276** (+ 4.2%)	**0.561** (+ 4.1%)*

Ranking KG Entities using Top Documents

[Schuhmacher, Dietz et al., CIKM'15]

Aim: complex entity-focused informational queries (e.g. "Argentine British relations")



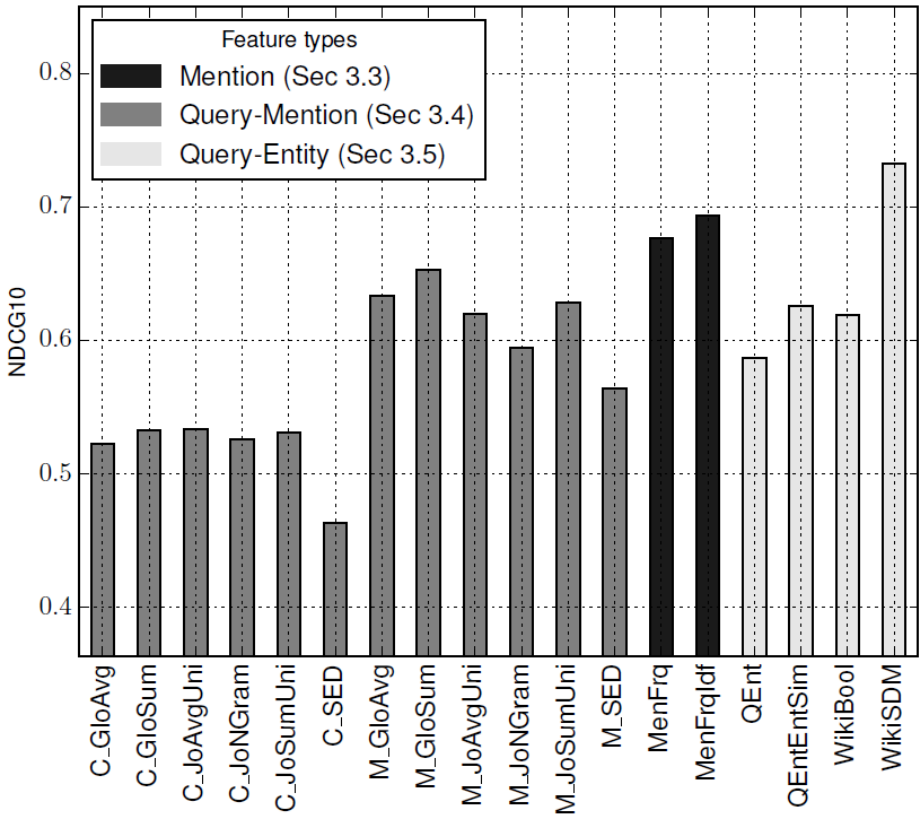
Features and rankers

Mention Features	
MenFrq	# of entity occurrences in top documents
MenFrqIdf	entity IDF
Query-Mention Features	
SED	normalized Levenshtein distance
Glo	similarity based on GloVe embeddings
Jo	similarity based on JoBimText embeddings
Query-Entity Features	
QEnt	is document entity linked in query
QEntEntSim	is there a path in KG between document and query entities
WikiBoolean	is entity Wikipedia article retrieved by query using Boolean model
WikiSDM	SDM retrieval score of entity Wikipedia article using query
Entity-Entity Features	
Wikipedia	is there a path between two entities in DBpedia KG

Rankers:

- rankSVM with linear kernel and linear+semantic smoothing kernels (pairwise)
- coordinate ascent with RankLib

Feature importance



- Strongest features are SDM retrieval score and entity IDF
- Authoritativeness marginally correlates with relevance (entities ranked high by PageRank are very general)
- DBpedia-based features have positive but insignificant influence on performance, while Wikipedia-based features show strong and significant influence

Takeaway messages

- Use dynamic entity representations built from different sources (not only KB)
- Use retrieval models that account for query unigram and bigrams (FSDM and PFSDM) rather than bag-of-words structured document retrieval models (BM25F and MLM) to obtain candidate entities
- Expand candidate entities by following KG links and using top-retrieved documents
- Re-rank candidate entities by using a variety of features including the ones based on latent dimensional entity representations

Thank you!

References (1)

Entity representation methods:

- Neumayer, Balog et al. When Simple is (more than) Good Enough: Effective Semantic Search with (almost) no Semantics, ECIR'12
- Zhiltsov, Kotov et al. Fielded Sequential Dependence Model for Ad-hoc Entity Retrieval in the Web of Data, SIGIR'15
- Graus, Tsagkias et al. Dynamic Collective Entity Representations for Entity Ranking, WSDM'16

References (2)

Entity retrieval and ranking:

- Zhiltsov, Kotov et al. Fielded Sequential Dependence Model for Ad-hoc Entity Retrieval in the Web of Data, SIGIR'15
- Nikolaev, Kotov et al. Parameterized Fielded Term Dependence Models for Ad-hoc Entity Retrieval from Knowledge Graph, SIGIR'16
- Tonon, Demartini et al. Combining Inverted Indices and Structured Search for Ad-hoc Object Retrieval, SIGIR'12
- Dali and Fortuna. Learning to Rank for Semantic Search, WWW'11
- Zhiltsov and Agichtein. Improving Entity Search over Linked Data by Modeling Latent Semantics, CIKM'13
- Schuhmacher, Dietz et al. Ranking Entities for Web Queries through Text and Knowledge, CIKM'15