EsdRank: Connecting Query and Documents through External Semi-Structured Data

Chenyan Xiong
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
cx@cs.cmu.edu

Jamie Callan
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
callan@cs.cmu.edu

ABSTRACT

This paper presents EsdRank, a new technique for improving ranking using external semi-structured data such as controlled vocabularies and knowledge bases. EsdRank treats vocabularies, terms and entities from external data, as objects connecting query and documents. Evidence used to link query to objects, and to rank documents are incorporated as features between query-object and object-document correspondingly. A *latent listwise* learning to rank algorithm, Latent-ListMLE, models the objects as latent space between query and documents, and learns how to handle all evidence in a unified procedure from document relevance judgments. EsdRank is tested in two scenarios: Using a knowledge base for web search, and using a controlled vocabulary for medical search. Experiments on TREC Web Track and OHSUMED data show significant improvements over state-of-the-art baselines.

Keywords

Ranking, Learning to Rank, Semi-Structured Data, Controlled Vocabulary, MeSH, Knowledge Base, Freebase

1. INTRODUCTION

Prior research has produced many effective unsupervised retrieval models (e.g., BM25, language models, DPH) [9] and supervised learning to rank (LeToR) models (e.g., RankSVM, ListMLE) [19]. Unsupervised retrieval models successfully model relevance using information such as the terms in the query, the terms in the documents, and term level statistics such as term frequency (*tf*), inverse document frequency (*idf*), and document length. Learning to rank research derives new models and training methods to incorporate the rich information provided by unsupervised retrieved models, together with other evidence such as PageRank and spam score, as document ranking features, and provide state-of-the-art ranking performance.

Sometimes the information in the query and corpus alone is not sufficient to rank documents effectively. For example, query terms may be an incomplete description of a well-known concept, or there may be a *vocabulary mismatch* between queries and relevant documents. If the search engine relies solely on evidence from the query and corpus, its accuracy is limited.

One solution is to incorporate evidence that is *external* to the corpus, for example, from a controlled vocabulary, a thesaurus, WordNet, Wikipedia, or more recently, a knowledge base such as Freebase or DBpedia. Usually such resources are constructed manually or by carefully-designed information extraction systems and thus have different characteristics and greater consistency. The rich *semi-structured* information that they provide, for example, synonyms, ontologies, entities, and relationships, gives the search engine a different perspective for understanding the relationship between a query and a document, and provides opportunities for improving retrieval accuracy.

There is a long history of using *external semi-structured data* to improve full-text search [2, 10, 18, 20, 22, 29]. Prior research demonstrates that such resources can improve search accuracy. Many options have been proposed for various types of external resources, however there is little guidance about how to integrate new resources in a general manner. Prior work mainly focuses on how to better represent the query using external data, for example, how to select good expansion terms [29] and how to link to good entities [10]. The query representation formulated from external data provides useful enrichment of the original query. Then the ranking is produced by either an unsupervised retrieval model or an existing learning to rank model that combines multiple query representations. We strive to further boost ranking performance by developing a new learning to rank model that reasons jointly over query, external data, and documents.

This paper describes EsdRank, a new method for improving ranking using external, semi-structured data. EsdRank treats the vocabularies, terms and entities from external data as objects. The information from external data used for better query representation are used as features between the query and external objects. The document ranking evidence used in LeToR research is used as features between objects and documents. Instead of treating the features about query-objects and features about object-document individually, EsdRank uses a *latent-listwise* LeToR model, Latent-ListMLE, which treats the external objects as latent layer between query and documents, and learns how to handle the features between query, objects, and documents in one unified procedure directly from document relevance judgments.

CIKM'15, October 19–23, 2015, Melbourne, Australia.

DOI: http://dx.doi.org/10.1145/2806416.2806456.

One major challenge in using external data is to find related objects for a query and documents. Several methods have been used by prior research [10, 29], but it is not clear how each of them contributes to final performance. This research explores three popular methods to select related objects from external data, including query annotation, object search, and document annotation. To investigate their effectiveness, we apply EsdRank with related objects generated by these methods on one newer type of external data, the Freebase knowledge base [3], and one classic type of external data, the MeSH controlled vocabulary[1], in web search and medical search respectively. Experiments on TREC Web Track and OHSUMED datasets show EsdRank's significant effectiveness over state-of-the-art ranking baselines, especially when using related objects from query annotations. Experiments also show that the effectiveness not only comes from the additional information from external data, but also our Latent-ListMLE model that uses it properly.

The next section describes related research. Section 3 discusses EsdRank, its Latent-ListMLE ranking model, related objects selection, and features. The experimental methodology and evaluation results are described in Sections 4 and 5 respectively. The last section summarizes the paper's contributions and discusses future work.

2. RELATED WORK

There is a long history of research on using information that is external to the document corpus to improve retrieval. The earliest information retrieval systems represented queries and documents using terms that were selected manually from controlled vocabularies. Controlled vocabulary representations required more effort to create during indexing and search, but were less susceptible to vocabulary mismatch between queries and documents, and required less sophisticated retrieval models.

As full-text search became popular, controlled vocabularies continued to be used in some systems. The use of controlled vocabularies is almost a necessity in medical search engines because queries are often about diseases, treatments or genes, whose special meanings may not be covered by their names. Typical procedures include using controlled vocabularies as alternative representations for matching query and document [22, 25], and expanding queries using synonyms [21]. However, the mixed results of systems participating in the TREC Genomics Track illustrate the difficulty of using controlled vocabularies in medical search; most systems require human efforts to be successful [26].

There is a rich literature on using dictionaries, thesauruses, and lexical resources such as WordNet to overcome vocabulary mismatch between queries and documents by adding synonyms and related concepts to queries and documents [18]. These approaches can be effective in enterprise search and domain-specific search environments [14], or improving recall of relevant documents in general search environments [18].

Recently a new type of semi-structured data, knowledge bases, has emerged. Examples include Wikipedia, its structured version DBpedia, Freebase [3] and the NELL KB [6]. Knowledge bases encode human knowledge about real world entities, such as names, aliases, descriptions, attributes, categories and relationships to other entities, in a semi-structured representation that is easily processed. Knowledge bases are a new path for search engines to better 'understand' queries and documents, but new techniques are required to use their novel and heterogeneous data types.

One intuitive method is to use the names and descriptions of knowledge base entities to expand the query. Xu et al. use pseudo relevance feedback in the Wikipedia corpus to obtain expansion terms that can be used to search other corpora [29]. Supervised methods have been used to pick better expansion terms from Wikipedia [4]. Wikipedia and other resources can be combined using both query concept weighting and query expansion for a better query representation [1]. Pan et al. use names of Freebase entities to expand queries [24]. Xiong et al. select expansion terms from Freebase entities retrieved by the query or discovered in its top retrieved documents [28]. Liu et al. had the best results in the TREC 2014 Web Track ad-hoc task using Freebase with manual labeling [20]. They pick entities textually similar with manually identified query entities, and rank documents by KL-divergence with these entities' descriptions.

A recent success in using Freebase is EQFE by Dalton et al. [10]. They study several ways to link Freebase entities to a query using query annotation, textual similarity, entity context model, and annotations from top retrieved documents. The name, alias and type fields of these entities are considered as possible expansion candidates. The combination of different linking methods, expansion fields, and hyper parameters in expansion are enumerated to get various expanded queries. These enumerated expansion queries are then used to calculate ranking scores for each document using a query likelihood or sequential dependency model. A learning to rank model uses these ranking scores as features for each document and produces final document rankings. EQFE provides a thorough exploration of information available to introduce entities between a query and documents, and is a major inspiration for EsdRank.

3. ESDRANK

EsdRank is intended to be a general technique for using external semi-structured data to improve ranking. External data elements are modeled as *objects*. An object could be a controlled vocabulary term, a term from another corpus, or a knowledge base entity. The evidence from the query, external data, and corpus is incorporated as features to express the relationship between query, object and document. A ranking model is then learned to rank documents with these objects and evidence.

In the first part of this section, we propose a novel latent listwise learning to rank model, Latent-ListMLE, as our ranking model. Latent-ListMLE handles the objects as a latent space between query and documents. The evidence between query-object, and object-document is naturally expressed as features connecting the query to the latent space, and then connecting latent space to documents.

Prior research found that a major challenge in using external data is to find related objects [7, 10, 29]. In the second part of this section, we explore three popular related object selection methods used in prior research. In the final part of this section, we describe the features used to connect query, objects and documents.

[1]http://www.nlm.nih.gov/mesh/meshhome.html

3.1 Latent-ListMLE

Given a query q and an initial set of retrieved documents D, a set of objects $O = \{o_1, ..., o_j, ..., o_m\}$ related to q and D is produced by one of the related object selection methods as described in Section 3.2. Features that describe relationships between query q and object o_j are denoted as vector v_j. Features that describe relationships between object o_j and document d_i are denoted as u_{ij}. The goal of Latent-ListMLE, like other learning to rank methods, is to re-rank D, but with the help of the related objects O and feature vectors $U = \{u_{11}, ..., u_{ij}, .., u_{nm}\}$ and $V = \{v_1, ..., v_j, ..., v_m\}$.

Latent-ListMLE treats O as the latent space between q and D and uses V, U as features to describe the relationships between query-object, and object-document. We will first revisit ListMLE [27], the listwise learning to rank model which Latent-ListMLE is built upon. Then we discuss the construction, learning, and ranking of Latent-ListMLE.

3.1.1 ListMLE Revisited

ListMLE defines the probability of a ranking (a list) being generated by a query in a parametric model. Maximum likelihood estimation (MLE) is used to find parameters that maximize the likelihood of the best ranking(s). However, the sample space of all possible rankings is the permutation of all candidate documents D, which is too large. One contribution of ListMLE is that it reduces the sample space by assuming the probability of a document being ranked at position i is independent of those ranked at previous positions.

Specifically, with a likelihood loss and a linear ranking model, ListMLE defines the probability of a document d_i being ranked at position i as:

$$p(d_i|q, S_i) = \frac{\exp(w^T x_i)}{\sum_{k=i}^{n} \exp(w^T x_k)}, \tag{1}$$

where $S_i = \{d_i \dots d_n\}$ are the documents that were not ranked in positions $1 \dots i-1$, x_i is the query-document feature vector for d_i, and w is the parameter vector to learn.

Equation 2 defines the likelihood of a given ranking $\vec{D}$ of candidate documents.

$$p(\vec{D}|q; w) = \prod_{i=1}^{n} \frac{\exp(w^T x_i)}{\sum_{k=i}^{n} \exp(w^T x_k)}. \tag{2}$$

The parameter vector w is learned by maximizing the likelihood for all given queries q_k and their best rankings $\vec{D}_k^*$ given document relevance judgments:

$$\hat{w} = \arg\max_{w} \prod_{k} p(\vec{D}_k^*|q_k; w). \tag{3}$$

This is an unconstrained convex optimization problem that can be solved efficiently by gradient methods.

3.1.2 Latent-ListMLE Construction

Latent-ListMLE extends ListMLE by adding a latent layer in the ranking generation process. The latent layer contains related objects O as possible representations of the original query q.

With the latent layer O, the ideal generation process of a ranking $\vec{D}$ is to first sample a ranking of objects $\vec{O}$, and then sample document ranking $\vec{D}$ based on $\vec{O}$. However, this process is also impractical due to the huge sampling space. Similarly to ListMLE, we assume that the probabilities of picking objects and documents at each position are independent with those at previous positions. Thus, the generative process is redefined to be:

For each position from 1 to N:

1. Sample o_j from multinomial distribution $Multi(O|q)$, with probability $p(o_j|q)$; and
2. Sample d_i from multinomial distribution $Multi(S_i|o_j)$, with probability $p(d_i|o_j, S_i)$.

We further define:

$$p(o_j|q) = \frac{\exp(\theta^T v_j)}{\sum_{k=1}^{m} \exp(\theta^T v_k)} \tag{4}$$

$$p(d_i|o_j, S_i) = \frac{\exp(w^T u_{ij})}{\sum_{k=i}^{n} \exp(w^T u_{kj})}, \tag{5}$$

where v_j is the query-object feature vector for o_j; u_{ij} is the object-document feature vector between d_i and o_j; m is the number of objects; and θ and w are the model parameters.

In this generative process, the latent layer is the sampled objects produced by query q, and the document ranking probability is conditioned on the sampled objects instead of the query. With this extension, the probability of picking d_i at position i is:

$$p(d_i|q, S_i) = \sum_{j=1}^{m} p(d_i|o_j, S_i)p(o_j|q) \tag{6}$$

$$= \sum_{j=1}^{m} \frac{\exp(w^T u_{ij})}{\sum_{k=i}^{n} \exp(w^T u_{kj})} \frac{\exp(\theta^T v_j)}{\sum_{k=1}^{m} \exp(\theta^T v_k)} \tag{7}$$

and the probability of ranking $\vec{D}$ given q is:

$$p(\vec{D}|q; w, \theta) = \prod_{i=1}^{n} \sum_{j=1}^{m} p(d_i|o_j, S_i)p(o_j|q). \tag{8}$$

Latent-ListMLE also uses query-document features by adding a 'query node' o_0 to O that represents query q. The features relating query q to o_0 are set to 0, thus o_0 is the 'origin point' for related objects. The features relating o_0 to documents are typical LeToR query-document features. The combination of query-document features and object-document features is done by treating them as individual dimensions in U, and setting missing feature dimensions' values to zero. So the dimensions referring to object-document similarities for the query node are set to zero, and vice versa.

Our idea of representing the query via related objects ($p(o|q)$) is similar to query expansion. In fact, if we use expansion terms as our related objects, and discard the document ranking part, Latent-ListMLE becomes a supervised query expansion method. On the other hand, if we only use the query node as related object, Latent-ListMLE is exactly the same as ListMLE. The difference is that, in Latent-ListMLE, the connections from query to latent layer, and from latent layer to documents are learned together in one unified procedure, which finds the best combination of query representation ($p(o|q)$) and document ranking ($p(d|o)$) together, instead of only focusing on one of them.

3.1.3 Learning

The parameters w and θ are learned using MLE with given queries and their best rankings. To keep the notation clear, we present the derivation with one training query q, without loss of generality.

For a training query q and its best ranking $\vec{D}^*$ derived from relevance labels, their log likelihood is:

$$l(\vec{D}^*|q; w, \theta) = \log p(\vec{D}^*|q; w, \theta) \tag{9}$$

$$= \sum_{i=1}^{n} \log \sum_{j=1}^{m} \Bigg(\frac{\exp(w^T u_{ij})}{\sum_{k=i}^{n} \exp(w^T u_{kj})} \times \frac{\exp(\theta^T v_j)}{\sum_{k=1}^{m} \exp(\theta^T v_k)} \Bigg). \tag{10}$$

The goal of MLE is to find parameters w^*, θ^* such that:

$$w^*, \theta^* = \arg\max_{w,\theta} l(\vec{D}^*|q; w, \theta). \tag{11}$$

Directly maximizing Equation 10 is difficult due to the summation of the latent variables inside the log, thus we use the EM algorithm to solve this optimization problem.

The **E step** finds the posterior distribution of hidden variable o_j for each ranking position given the current parameters θ_{old} and w_{old}.

$$\pi(o_j|d_i, q) = p(o_j|d_i, q; \theta_{old}, w_{old}) = \frac{p(d_i|o_j, S_i; w_{old}) p(o_j|q; \theta_{old})}{\sum_{k=1}^{m} p(d_i|o_k, S_i; w_{old}) p(o_k|q; \theta_{old})}.$$

The **M step** maximizes the expected log complete likelihood.

$$E(\tilde{l}) = E_{\pi(\vec{O}|\vec{D}^*, q)} \log p(\vec{D}^*, \vec{O}|q; w, \theta)$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{m} \pi(o_j|d_i, q) \log p(d_i|o_j, S_i) p(o_j|q)$$

$$= \sum_{i,j} \pi(o_j|d_i, q) \log \frac{\exp(w^T u_{ij})}{\sum_{k=i}^{n} \exp(w^T u_{kj})} \frac{\exp(\theta^T v_j)}{\sum_{k=1}^{m} \exp(\theta^T v_k)}.$$

We use gradient ascent to maximize the expectation. The gradients are:

$$\frac{\partial E(\tilde{l})}{\partial w} = \sum_{i,j} \pi(o_j|d_i, q) \Big\{ u_{ij} - \frac{\sum_{k=i}^{n} u_{kj} \exp w^T u_{kj}}{\sum_{k=i}^{n} \exp w^T u_{kj}} \Big\} \tag{12}$$

$$\frac{\partial E(\tilde{l})}{\partial \theta} = \sum_{i,j} \pi(o_j|d_i, q) \Big\{ v_j - \frac{\sum_{k=1}^{m} v_k \exp(\theta^T v_k)}{\sum_{k=1}^{m} \exp(\theta^T v_k)} \Big\}. \tag{13}$$

The E step is very efficient with the closed form solution. The M step is an easy convex optimization problem. Intuitively, the E step finds the best assignment of object probabilities under current parameters and best document rankings, thus transferring document relevance information to latent objects. The M step learns the best parameters that fit the object probabilities provided by the E step. EM iterations are guaranteed to improve the likelihood until convergence. However, the overall optimization is not convex and has local optima. In practice the local optima problem can be suppressed by repeating the training several times with random initial w and θ, and using the result that converges to the largest likelihood.

3.1.4 Ranking

Given a learned model, a query, and an initial ranking, a new ranking is constructed by picking the document that has the highest probability $p(d_i|q, S_i)$ at each position from 1 to n. The complexity of picking one document is $O(nm)$, thus the total cost of ranking n documents with m related objects is $O(n^2m)$, which is slower than ListMLE's $O(n \log n)$ ranking complexity. We can restrict the number of documents n (e.g. 100) and related objects m (e.g. < 5) to maintain reasonable efficiency.

3.2 Related Objects

How to find related objects O given query q and documents D is important for using external data. Many options have been proposed by prior research [10, 20, 24, 29], but it is not clear which is the most reliable. This research studies the following three popular automatic methods to find related objects.

Query Annotation selects the objects that directly appear in the query. Based on specific object types in the external data, one can choose corresponding techniques to 'annotate' them to the query, for example, entity linking techniques [7] to find entities that appear in the query, or all query terms when the objects are terms from an external corpus.

Object Search selects the objects that are textually similar to the query. Search engines can be used to find such objects. We can build an index for all the objects in external data, in which each document is the textual data about the object, for example, name, alias, description and context words of an entity. Then textually similar objects can be retrieved by running the query to the index.

Document Annotation selects the objects that appear in retrieved documents D. The terms in retrieved documents have been widely used in query expansion. The entities that are annotated to D were also useful in prior work [10]. This method introduces objects that are more indirectly related to the query, such as 'President of United States' for the query 'Obama family tree'. A typical method to score and select objects from retrieved documents is the RM3 pseudo relevance feedback model [17].

3.3 Features

Representing the relationship between query and related objects is the major focus of prior research in using external data. We explore the following query-object features in EsdRank; most of them are inspired by Dalton et al. [10].

3.3.1 Features between Query and Objects

Object Selection Score features are the scores produced by the object selection step: Annotator confidence, object ranking score, and RM3 model score $s(q, o)$.

Textual Similarity features cover the similarities between the query and the object's textual fields. For example, one can use coordinate matches, BM25 scores, language model scores, and sequential dependency model (SDM) scores between the query and the object's textual fields, such as name, alias (if any) and descriptions if using entities.

Ontology Overlap features use the ontology, a common type of information for some external semi-structured datasets. When an ontology is available, one can build a multiclass classifier that classifies the query into the ontology, and use the overlaps with the object's ontology as features.

Object Frequency is the number of documents in the corpus the object appears in (e.g. term) or is annotated to (e.g. entity). This feature is query independent and distinguishes frequent objects from infrequent ones.

Similarity with Other Objects are the max and mean similarity of this object's name with the other related objects'. These features distinguish objects that have similar names with other related objects from those that do not.

3.3.2 Features between Objects and Documents

In learning to rank, rich query-document ranking features, such as multiple retrieval algorithms on different document fields, have been shown very important for ranking performance [19]. Our object-document features are similar to query-document features widely used in LeToR research. But objects may have richer contents than queries, enabling a larger set of features.

Textual Similarity features measure the similarity of a document and an object's textual fields. BM25, language model, SDM, coordinate match, and cosine similarity in the vector space model are used to calculate similarities between all combinations of an object's textual fields and a document's fields. These retrieval models are applied by using the object to 'retrieve' the document. The fusion of these similarity features provides multiple ways to express the object-document similarities for EsdRank.

Ontology Overlap features are the same as the ontology overlap features between query and object. The same multiclass classifier can be used to classify documents, and overlaps in the object's and document's top categories can be used as features.

Graph Connection features introduce information about the relationships in the external data and document annotations. If such graph-like information is available, one can start from the object, traverse its relations, and record the number of annotated objects reached at each step (usually within 2 steps) as features. These features model the 'closeness' of the object and the document's annotations in the external data's relationship graph.

Document Quality features are commonly used in learning to rank. We use classic document quality features such as document length, URL length, spam score, number of inlinks, stop word fraction, and whether the document is from Wikipedia (web corpus only).

3.4 Discussion

EsdRank provides general guidance about how to use external semi-structured data in ranking. When an external dataset is available, to use it in ranking, one can first use object selection methods to find related objects for each query, and then extract query-object and object-document features. Although the detailed object selection methods and features may vary for different datasets, we list some common related object selection methods and features that have been used widely in prior research to start with. One can also derive new related object selection methods and features based on other available information.

Related objects and features are handled by our latent listwise LeToR model, Latent-ListMLE. It models the objects as the latent space. As a result, evidence is decoupled into a query-object part and an object-document part, which is easier to learn than mixed together (as shown in Section 5.2). Instead of solely focusing on the query-object part, or the document ranking part, Latent-ListMLE learns them together using EM. Our experiments show that the additional evidence from external data and Latent-ListMLE are both necessary for EsdRank's effectiveness.

4. EXPERIMENTAL METHODOLOGY

EsdRank is intended to be a general method of using external semi-structured data, thus experiments test it with two types of semi-structured data and search tasks: Web search using the Freebase knowledge base, and medical search using the MeSH controlled vocabulary. The former uses a newer type of semi-structured data, a noisy corpus, and queries from web users; the latter uses a classic form of semi-structured data, a clean corpus, and queries from domain experts. Datasets and ranking features are determined by these two choices, as described below.

Data: Experiments on web search using Freebase were conducted with ClueWeb09-B and ClueWeb12-B13, two web corpora used often in IR research. Each corpus contains about 50 million English web documents. The 2009-2013 TREC Web Tracks provided 200 queries with relevance assessments for ClueWeb09 and 50 queries with relevance assessments for ClueWeb12. We used a snapshot of Freebase provided by Google on Oct 26th, 2014.[2]

Experiments on medical search using the MeSH controlled vocabulary were conducted with the OHSUMED corpus, a medical corpus used often in IR research. It contains abstracts of medical papers that are manually annotated with MeSH terms. The OHSUMED dataset includes 106 queries with relevance assessments. We used the 2014 MeSH dump provided by the U.S. National Libarary of Medicine (NIH).[3]

Indexing and Base Retrieval Model: The ClueWeb and OHSUMED corpora were indexed by the Indri search engine, using default stemming and stopwords. ClueWeb documents contain title, body and inlink fields. OHSUMED documents contain title and body fields.

Freebase and MeSH also were indexed by Indri, with each object (entity, controlled vocabulary term) treated as a document containing its associated text fields (name, alias, description). Objects without descriptions were discarded. Retrieval was done by Indri using its default settings.

Spam filtering is important for ClueWeb09, thus we removed the 70% spammiest documents using Waterloo spam scores [8]. Prior research questions the value of spam filtering for ClueWeb12 [10], thus we did not filter that dataset.

Related Objects: We implement three related generation methods as described in Section 3.2.

Query annotation (`AnnQ`) was done by TagMe [12], one of the best systems in the Short (Query) Track at the ERD14 workshop competition [7]. TagMe annotates the query with Wikipedia page ids. Wikipedia page ids were mapped to Freebase objects using their Wikipedia links and to MeSH controlled vocabulary terms using exact match.

Object search (`Osrch`) was done with the Indri search engine for both external semi-structured datasets. We investigated using Google's Freebase Search API to search for Freebase objects and PubMed's MeSH query annotations to annotate MeSH objects. Neither was significantly better than Indri search or TagMe annotation, thus we only report results with public solutions.

Google's FACC1 dataset provided annotations of Freebase entities in ClueWeb documents[4] [13]. The OHSUMED dataset contains MeSH annotations for each document. Document annotation (`AnnD`) objects are generated by first

[2]https://developers.google.com/freebase/data
[3]http://www.nlm.nih.gov/mesh/filelist.html
[4]http://lemurproject.org/clueweb09/

Table 1: Query-object features. 'Web' and 'Medical' indicate the number of features in each corpus.

Feature	Web	Medical
Score from Query Annotation	1	1
Score from Object Search	1	1
Score from Document Annotation	1	1
Language model of object's description	1	1
SDM of object's description	1	1
BM25 of object's description	1	1
Coordinate match of object's text fields	3	3
Top 3 categories overlap	1	1
Object's idf in corpus's annotation	1	1
Similarity with other objects	2	2
Total Dimensions	13	13

Table 2: Object-document features. 'Web' and 'Medical' indicate the number of features in each corpus. 'Field combinations' refers to combinations of an object's and a document's fields. An object's fields include name, alias, description, query minus name, query minus alias, and query minus description. ClueWeb documents (for Fb) contain title, body and inlink fields. OSHUMED documents (for MeSH) contain title and body fields.

Feature	Web	Medical
Language model of field combinations	18	12
SDM of field combinations	18	12
Cosine correlation of field combinations	18	12
Coordinate match of field combinations	18	12
BM25 of field combinations	18	12
Top 3 cateogories overlap	1	1
Is annotated to document	1	1
Connected in graph at 1,2 hop	2	0
Total Dimensions	94	62

retrieving documents from the ClueWeb or OHSUMED corpus, and then using the RM3 relevance model [17] to select annotated objects from these documents.

Thus, the experiments consider query annotation (`EsdRank-AnnQ`), object search (`EsdRank-Osrch`) and document annotation (`EsdRank-AnnD`) methods to select related objects.

Feature Extraction: We extract features described in Section 3.3. Our feature lists are shown in Tables 1–3.

For object-document text similarity features, each object field is dynamically expanded with a virtual field that contains any query terms that the field does not contain. For example, given an object with name 'Barack Obama' and query 'Obama family tree', the text 'family tree' is added to the object as a virtual field called 'query minus name'. Similar expansion is performed for 'query minus alias' and 'query minus description' fields. As a result, Latent-ListMLE also knows which part of the query is not covered by an object. This group of features is very important for long queries, in order to make sure documents only related to a small fraction of the query are not promoted too much.

The ontology features use the linear multiclass SVM implementation of SVMLight [15] as the classifier. The Freebase classes are the 100 most frequent (in FACC1 annotation) bottom categories in Freebase's two-level ontology tree. The classes for MeSH are all of the top level categories in

Table 3: Query-document and document quality features used by EsdRank and learning to rank baselines. 'Web' and 'Medical' indicate the number of features in each corpus.

Feature	Web	Medical
Language model of doc's fields	3	2
SDM of doc's fields	3	2
Cosine correlation of doc's fields	3	2
Coordinate match of doc's fields	3	2
BM25 of doc's fields	3	2
Spam Score	1	0
Number of inlinks	1	0
Stop word fraction	1	1
URL length	1	0
Document length	1	1
Is Wikipedia	1	0
Total Dimensions	21	12

MeSH's ontology. The training data is the descriptions from objects of each class. Objects are classified by their descriptions. Documents are classified by their texts. Queries are classified using voting by the top 100 documents that Indri retrieves for them [5]. The accuracy of Multiclass SVM in cross-validating on training data is above 70%.

The graph connection features between Freebase entities and ClueWeb documents are calculated using Freebase's knowledge graph. We treat all edges the same, leaving further exploration of edge type to future work. We only use the reachable at zero hop for OHSUMED, that is whether the MeSH term is annotated to the document.

Evaluation Metrics: We use ERR@20 and NDCG@20, which are the main evaluation metrics in the TREC Web Track ad hoc task. Besides these two metrics that focus on the top part of ranking, we also use MAP@100, which considers the quality over the entire reranked section.

Statistical significance was tested by the permutation test (Fisher's randomization test) with $p\text{-value} < 0.05$, which is a common choice in IR tasks.

Hyper-Parameters and Training: We follow standards in prior work to set hyper-parameters. The number of documents retrieved and re-ranked is set to 100. The same set of documents is used to generate document annotation `AnnD`. All supervised ranking models are evaluated on the testing folds in 10-fold cross validation. Cross-validation partitioning is done randomly and kept the same for all experiments. To suppress the local optima problem, in each cross-validation fold Latent-ListMLE is trained ten times with random initialization. The training result with the largest likelihood on training data is used for testing. In order to maintain reasonable ranking efficiency and avoid too many noisy objects, the number of related objects from `AnnD` and `Osrch` are restricted to at most 3. This restriction does not change `AnnQ` as none of our queries has more than 3 annotations.

Baselines: The first baseline is the Sequential Dependency Model (`SDM`) [23] approach which is widely recognized as a strong baseline. For ClueWeb datasets, we find that Dalton et al's SDM results obtained with Galago[5] are stronger than our SDM results obtained with Indri, thus we use their SDM results as a baseline. We also use `EQFE` rankings provided by them as second baseline on ClueWeb data

[5]http://ciir.cs.umass.edu/downloads/eqfe/runs/

sets. On OHSUMED we use the Indri query language to obtain `SDM` results. `EQFE` is not included for OHSUMED as it is specially designed for knowledge bases. The third and fourth baselines are two state-of-the-art learning to rank baselines: `RankSVM` (pairwise) [16] and `ListMLE` (listwise) [27]. Their features are shown in Table 3 and are trained and tested exactly the same as `EsdRank`.

Three of our baseline algorithms were used widely in prior research; the fourth (EQFE) is included because it has important similarities to EsdRank. We could have included other methods that use external data, for example, Wikipedia for query expansion [4, 29], Wikipedia as one resource for query formulations [1], and MeSH as an alternate document representation [22]. These methods mainly focus on using external data to better represent the query; they rank documents with unsupervised retrieval models. In recent TREC evaluations, supervised LeToR systems that use rich ranking features have shown much stronger performance than unsupervised retrieval systems. Thus we mainly compare with LeToR models with document ranking features, e.g., those in Table 3, which we believe are the strongest baselines available now.

5. EVALUATION RESULTS

This section first presents experiment results about `EsdRank`'s overall accuracy. Then it investigates the effectiveness of our `Latent-ListMLE` model, together with the influence of related object quality on `EsdRank`'s performance.

5.1 Overall Performance

Tables 4a, 4b, and 4c show the retrieval accuracy of several baseline methods and three versions of `EsdRank` on ClueWeb09, ClueWeb12, and OHSUMED datasets. The three variants of `EsdRank` differ only in how related objects are selected. †, ‡, § and ¶ indicate statistical significance over `SDM`, `RankSVM`, `ListMLE` and `EQFE`. The change relative to `ListMLE` is shown in parentheses. Win/Tie/Loss is the number of queries improved, unchanged or damaged as compared to `ListMLE` by ERR@20. The best performing method according to each evaluation metric is marked **bold**.

On all three data sets, the best `EsdRank` methods outperform all baselines on all metrics. The gain over `ListMLE` varies from 3.33% (OHSUMED, MAP@100) to 21.85% (ClueWeb09, ERR@20), with 5–7% being typical. We note that `ListMLE` is a very strong baseline; there is little prior work that provides statistically significant gains of this magnitude over `ListMLE` on the ClueWeb datasets. These improvements show the effectiveness of external data in ranking, which is `EsdRank`'s only difference with `ListMLE`.

On ClueWeb data sets, all three versions of `EsdRank` outperform `EQFE`, and the best performing version improves over `EQFE` by 10%-30%. Both `EQFE` and `EsdRank` use Freebase as external data and learning to rank models to rank documents. The features between query and object in `EsdRank` are very similar with those used in `EQFE`. Why does `EsdRank` perform better than `EQFE`?

One difference is in the object-document features. `EQFE` uses the language model or SDM score between an entity's textual fields and the whole document to connect the entity to the document, while `EsdRank` uses a much larger feature set (Table 2). The object-document features in Table 2 provide multiple different views of object-document relevancy, which has been shown very effective in LeToR research [19]. More object-document features also better propagate document relevance judgments to latent objects in `Latent-ListMLE`'s EM training procedure, and help it learn better weights for query-object features. We have tried `EsdRank` with only language model scores between an object's textual fields and the document as object-document features, however, results with this smaller feature set are no better than `EQFE`. In Section 5.2, our second experiment also shows that our `Latent-ListMLE` model is another essential factor for `EsdRank`'s performance with its ability to handle features in the two parts properly.

On ClueWeb09 and OHSUMED, query annotation (`EsdRank-AnnQ`) performs the best with statistically significant improvements on almost all evaluations over all baselines, indicating query annotation is still the most effective in finding reliable objects. On ClueWeb12, `EsdRank-AnnQ` also outperforms all baselines, although less statistical significance is observed due to fewer training queries (only 50). It is interesting to see that `EsdRank-AnnD` performs better than `EsdRank-AnnQ` on ClueWeb12. This is consistent with results of `EQFE` [10], showing that FACC1 annotation might be of better quality for ClueWeb12 queries than ClueWeb09 queries. `EsdRank-Osrch` performs the worst on all three data sets. The reason is that object search provides related objects that might be textually similar to the query, but are actually noise. The ranking models designed to rank document use assumptions that may not be suitable for objects, leaving room for improvement in future research.

The different effectiveness of `EsdRank` with different related object selection methods shows the importance of related object quality. It is also well recognized as one of the most essential factors in determining the usefulness of external data in prior research [7, 11, 20, 28]. In Section 5.3, our third experiment studies the correlation between related objects' quality and `EsdRank`'s performance.

5.2 Effectiveness of Latent-ListMLE

To investigate the effectiveness of `Latent-ListMLE`, in this experiment, we compare it with `ListMLE`, using the same external evidence, but with features generated by feature engineering techniques similar to those in Dalton et al. [10].

Formally, for each query q and document d_i pair, we have objects $\{o_1, ..., o_j, ..., o_m\}$, query-object features $V = \{v_1, ..., v_j, ..., v_m\}$ and object-document features $U_i = \{u_{i1}, ..., u_{ij}, ..., u_{im}\}$. Let $|u|$ and $|v|$ be the feature dimensions for query-object features and object-document features. We generate $|u| \times |v|$ features $x_i = \{x_i(1,1), ..., x_i(k_1,k_2), ..., x_i(|u|,|v|)\}$ by enumerating all combination of features in U_i and V. The combination of k_1-th feature in V and k_2-th feature U_i is defined as: $x_i(k_1,k_2) = \sum_j v_j(k_1) \times u_{ij}(k_2)$, the summation of corresponding features over all possible related objects. One may notice that if we only use language model scores as object-document features, this set up is exactly the same as `EQFE`. We name these 'flat' features in comparison with `Latent-ListMLE`'s latent space model.

We put these 'flat' features into `ListMLE` and use the same training-testing procedure to evaluate them on our datasets. Evaluation results are shown in Table 5. `ListMLE-AnnQ`, `ListMLE-Osrch` and `ListMLE-AnnD` refer to the flat model with features from related objects selected by query annotation, object search and document annotation respectively. Relative performance (in brackets) and Win/Tie/loss

Table 4: Performance of `EsdRank`. `AnnQ`, `Osrch` and `AnnD` refer to object selection methods: Query annotation, object search and document annotation. Relative changes compared to `ListMLE` are shown in parentheses. Win/Tie/Loss is the number of queries improved, unchanged or hurt compared to `ListMLE`. †, ‡, § and ¶ show statistic significance ($p < 0.05$ in permutation test) over `SDM`, `RankSVM`, `ListMLE` and `EQFE`. The best method for each evaluation metric is marked **bold**.

(a) ClueWeb09

Method	MAP@100		NDCG@20		ERR@20		Win/Tie/Loss
`SDM`	0.375	(−8.54%)	0.214	(−2.92%)	0.135	(−13.52%)	75/30/95
`EQFE`	0.364	(−11.14%)	0.213	(−3.73%)	0.139	(−11.21%)	77/27/96
`RankSVM`	$0.352^{\dagger}$	(−14.14%)	$0.193^{\dagger}$	(−12.54%)	0.147	(−6.10%)	59/37/104
`ListMLE`	$0.410^{\dagger,\ddagger,\P}$	–	$0.221^{\dagger,\ddagger}$	–	$0.156^{\dagger}$	–	–
`EsdRank-AnnQ`	$\mathbf{0.437}^{\dagger,\ddagger,\S,\P}$	(**6.59**%)	$\mathbf{0.237}^{\dagger,\ddagger,\S,\P}$	(**7.17**%)	$\mathbf{0.190}^{\dagger,\ddagger,\S,\P}$	(**21.85**%)	88/42/70
`EsdRank-Osrch`	$0.397^{\dagger,\ddagger,\P}$	(−3.16%)	$0.219^{\dagger,\ddagger}$	(−0.68%)	$0.155^{\dagger}$	(−0.75%)	71/47/82
`EsdRank-AnnD`	$0.403^{\dagger,\ddagger,\P}$	(−1.69%)	$0.221^{\dagger,\ddagger}$	(0.19%)	$0.167^{\dagger,\P}$	(6.70%)	75/36/89

(b) ClueWeb12

Method	MAP@100		NDCG@20		ERR@20		Win/Tie/Loss
`SDM`	0.293	(−26.19%)	0.126	(−27.53%)	0.091	(−25.66%)	18/7/25
`EQFE`	0.319	(−19.74%)	$0.146^{\dagger}$	(−16.02%)	0.106	(−13.61%)	17/7/26
`RankSVM`	$0.383^{\dagger}$	(−3.66%)	$0.161^{\dagger}$	(−7.29%)	0.114	(−7.48%)	16/13/21
`ListMLE`	$0.397^{\dagger,\P}$	–	$0.174^{\dagger,\P}$	–	$0.123^{\dagger}$	–	–
`EsdRank-AnnQ`	$0.410^{\dagger,\P}$	(3.12%)	$0.181^{\dagger,\P}$	(3.78%)	$\mathbf{0.133}^{\dagger,\ddagger,\S,\P}$	(**8.39**%)	20/12/18
`EsdRank-Osrch`	$0.391^{\dagger,\P}$	(−1.51%)	$0.167^{\dagger}$	(−3.88%)	$0.121^{\dagger}$	(−1.26%)	22/10/18
`EsdRank-AnnD`	$\mathbf{0.440}^{\dagger,\ddagger,\S,\P}$	(**10.69**%)	$\mathbf{0.186}^{\dagger,\ddagger,\S,\P}$	(**7.15**%)	$0.132^{\dagger,\P}$	(7.67%)	24/14/12

(c) OHSUMED

Method	MAP@100		NDCG@20		ERR@20		Win/Tie/Loss
`SDM`	0.413	(−0.27%)	0.332	(−2.96%)	0.453	(−6.14%)	38/11/57
`RankSVM`	0.396	(−4.31%)	0.336	(−1.89%)	0.453	(−6.12%)	43/14/49
`ListMLE`	0.414	–	0.343	–	0.483	–	–
`EsdRank-AnnQ`	$\mathbf{0.428}^{\ddagger,\S}$	(**3.33**%)	$\mathbf{0.355}^{\dagger,\ddagger,\S}$	(**3.54**%)	$\mathbf{0.511}^{\dagger,\ddagger,\S}$	(**5.77**%)	56/16/34
`EsdRank-Osrch`	$0.427^{\ddagger,\S}$	(3.22%)	0.347	(1.40%)	$0.489^{\ddagger}$	(1.29%)	44/17/45
`EsdRank-AnnD`	0.416	(0.62%)	0.342	(−0.16%)	$0.500^{\dagger,\ddagger}$	(3.59%)	44/20/42

values are compared with `ListMLE`, which uses the same model but query-document features. ▽ indicates significantly worse performance than `ListMLE` in the permutation test ($p < 0.05$). ▼ indicates significantly worse performance compared to the corresponding version of `EsdRank`. For example, `ListMLE-AnnQ` is significantly worse than `EsdRank-AnnQ` if marked by ▼.

Most times these flat features hurt performance, with significant drops compared to `ListMLE` and corresponding `EsdRank` versions. Even on ClueWeb09, where the most training data is available, it is typical for flat features to perform more than 10% worse than `EsdRank`. These results demonstrate the advantage of using a latent space learning to rank model instead of a flat model to handle external evidence: By modeling related objects as hidden variables in latent space, `Latent-ListMLE` naturally separates query-object and object-document evidence, and uses the EM algorithm to propagate document level training data to the object level. As a result, `Latent-ListMLE` avoids feature explosion or confusing machine learning algorithms with highly correlated features, while still only requiring document relevance judgments.

5.3 Influence of Related Object Quality

Prior research [7, 11, 20, 28] and our evaluation results on `EsdRank` with different related objects all indicate the great influence of selected objects' quality on ranking accuracy. The third experiment further studies its influence on `EsdRank`'s performance. Although directly measuring the quality is hard due to the lack of object level labels, we can indirectly characterize the quality by comparing the related objects selected by different methods, with the hypothesis that objects selected by multiple object selectors are more likely to have higher quality.

For each variant of `EsdRank`, queries were divided into two groups (`Agree`, `Differ`) based on whether the query had at least one related object in common with another variant. The two groups were compared using the same methodology reported earlier. The results are summarized in Table 6. Each row is the performance of one variant (first column), divided by overlaps with another variant (second column). Relative gains, Win/Tie/Loss, and statistical significance (§) are calculated based on comparison with `ListMLE`. Results for ClueWeb09, ClueWeb12 and OSHUMED queries are combined due to space limitations and because their trends are similar.

The two groups clearly perform differently. On queries with common objects (`Agree`), `EsdRank` is much more effective. For example, when the less effective `AnnD` and `Osrch` object selectors agree with the more effective `AnnQ` selector, they both outperform `ListMLE` with statistical significance. When they differ, they may be worse than `ListMLE`.

Table 5: Performance of `ListMLE` with flat features derived from external data. `AnnQ`, `Osrch` and `AnnD` refer to object selection methods: Query annotation, object search and document annotation. Relative changes and Win/Tie/Loss are compared to `ListMLE`. $\triangledown$ and $\blacktriangledown$ indicate statistic significantly weaker performance compared to `ListMLE` and `EsdRank` (which uses the same information but `Latent-ListMLE`).

Data Set	Method	MAP@100	NDCG@20	ERR@20	Win/Tie/Loss
ClueWeb09	`ListMLE-AnnQ`	$0.389^{\blacktriangledown}$ (−4.96%)	$0.208^{\blacktriangledown}$ (−5.88%)	$0.170^{\blacktriangledown}$ (8.99%)	73/35/92
	`ListMLE-Osrch`	$0.331^{\triangledown,\blacktriangledown}$ (−19.33%)	$0.168^{\triangledown,\blacktriangledown}$ (−23.79%)	$0.127^{\triangledown,\blacktriangledown}$ (−18.34%)	47/42/111
	`ListMLE-AnnD`	$0.357^{\triangledown,\blacktriangledown}$ (−12.79%)	$0.197^{\triangledown,\blacktriangledown}$ (−10.63%)	0.155 (−0.61%)	64/38/98
ClueWeb12	`ListMLE-AnnQ`	$0.355^{\blacktriangledown}$ (−10.71%)	$0.137^{\triangledown,\blacktriangledown}$ (−21.04%)	$0.098^{\triangledown,\blacktriangledown}$ (−20.37%)	14/11/25
	`ListMLE-Osrch`	0.368 (−7.43%)	0.154 (−11.32%)	0.105 (−14.39%)	16/11/23
	`ListMLE-AnnD`	$0.337^{\triangledown,\blacktriangledown}$ (−15.06%)	$0.149^{\triangledown,\blacktriangledown}$ (−14.38%)	$0.098^{\triangledown,\blacktriangledown}$ (−20.65%)	14/10/26
OHSUMED	`ListMLE-AnnQ`	$0.388^{\triangledown,\blacktriangledown}$ (−6.13%)	$0.323^{\triangledown,\blacktriangledown}$ (−5.64%)	$0.444^{\triangledown,\blacktriangledown}$ (−7.94%)	39/15/52
	`ListMLE-Osrch`	$0.400^{\blacktriangledown}$ (−3.37%)	$0.323^{\triangledown,\blacktriangledown}$ (−5.58%)	$0.445^{\triangledown,\blacktriangledown}$ (−7.84%)	38/15/53
	`ListMLE-AnnD`	$0.397^{\triangledown,\blacktriangledown}$ (−4.12%)	0.339 (−1.14%)	0.494 (2.43%)	40/16/50

Although `AnnQ` is the most effective individual method, its performance is further improved when it agrees with `AnnD`, outperforming `ListMLE` by more than 10% on all metrics. This level of improvement is close to the gains reported by Liu et al. [20] with *manual* query annotations.

The results also show that `EsdRank` behaves predictably. When simple object selectors agree, `EsdRank` is more likely to improve retrieval quality; when they disagree, the search engine can simply retreat to `ListMLE`.

Finding related objects for query and document is still an open problem in the literature. In the SIGIR ERD'14 workshop [7] many teams built their query annotators upon TagMe and obtained about 60% accuracy. It is still not clear how to adapt full text retrieval models to perform better object search. Document annotation can be done with very high precision but there is still room to improve recall, even in the widely used Google FACC1 annotations. Some prior research questions whether current Freebase query annotation is too noisy to be useful [11, 20]. With the help of features between query, object and document, `Latent-ListMLE` is able to improve retrieval accuracy even when object selection is noisy. As researchers improve the quality of object search and annotation, producing less noisy objects, we would expect `Latent-ListMLE`'s ranking accuracy to improve further.

6. CONCLUSION AND FUTURE WORK

EsdRank is a general method of using external semi-structured data in modern LeToR systems. It uses objects from external data, such as vocabularies and entities, as an interlingua between query and documents. Query-object and object-document relationships are expressed as features. Latent-ListMLE treats objects as latent layers between query and document, and learns how to use evidence between query, objects and documents in one unified procedure. This general method can be applied to a variety of semi-structured resources, and is easily trained using ordinary query-document relevance judgments.

Experiments with two rather different types of external semi-structured data – a classic controlled vocabulary and a newer knowledge base – and three well-known datasets show that EsdRank provides statistically significant improvements over several state-of-the-art ranking baselines.

Experiments with the single-layer LeToR model, which uses exactly the same information but expressed by 'flat' features, show much weaker performance than Latent-ListMLE. This result confirms the advantage of separating evidence about query-object and object-document relationships with the latent space, instead of mixing them together. The single-layer approach may result in a large and highly correlated feature space, which is hard for machine learning models to deal with.

Finding related objects for query and documents is a very important step for using external data in ranking. EsdRank is tested with three popular related object selection methods: query annotation, object search and document annotation. The evaluation results demonstrate the essential effect of related object quality on final ranking accuracy, and suggest that query annotation is the most reliable source for related objects under current techniques.

EsdRank is a general approach in which to explore the use of different types of semi-structured data in search. As better methods are developed for retrieving objects from semi-structured data and annotating objects in queries and documents, they can be used in EsdRank to improve retrieval and to cover a broader set of queries. Its use of objects as a latent space between queries and documents provides richer opportunities for feature and model development than seen so far in most learning to rank research. Thus, the implementation of EsdRank presented here can be seen as just a beginning.

7. ACKNOWLEDGMENTS

We thank Laura Dietz and the anonymous reviewers of a previous draft of this paper for comments that improved the paper. This research was supported by National Science Foundation (NSF) grant IIS-1422676 and a Google Research Award. Any opinions, findings, conclusions, and recommendations expressed in this paper are the authors' and do not necessarily reflect those of the sponsors.

8. REFERENCES

[1] M. Bendersky, D. Metzler, and W. B. Croft. Effective query formulation with multiple information sources. In *Proceedings of the fifth ACM International Conference on Web Search and Data Mining*, pages 443–452. ACM, 2012.

[2] A. Berger and J. Lafferty. Information retrieval as statistical translation. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 222–229. ACM, 1999.

[3] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings*

Table 6: The performances of `EsdRank`'s variants when they agree or differ with other variants, combined from three data sets. 'Agree' is defined by at least one overlap in related objects. Relative gain, Win/Tie/Loss and statistical significance § are compared with `ListMLE` on corresponding queries. The best relative gains for each variant are marked **bold**.

Method	Compare With	Group	MAP@100	NDCG@20	ERR@20	Win/Tie/Loss	Queries
`AnnQ`	`AnnD`	`Agree`	$0.541^{\S}$ (**14.80%**)	$0.299^{\S}$ (**12.89%**)	$0.254^{\S}$ (**24.18%**)	54/14/26	94
		`Differ`	0.390 (0.90%)	0.252 (2.45%)	$0.286^{\S}$ (8.14%)	110/56/96	262
	`Osrch`	`Agree`	$0.495^{\S}$ (7.10%)	$0.328^{\S}$ (6.69%)	$0.347^{\S}$ (7.53%)	74/17/39	130
		`Differ`	0.393 (3.75%)	$0.227^{\S}$ (4.28%)	$0.238^{\S}$ (15.31%)	90/53/83	226
`Osrch`	`AnnD`	`Agree`	0.490 (1.81%)	0.287 (1.66%)	0.210 (0.08%)	29/9/21	59
		`Differ`	0.388 (−1.70%)	0.243 (−0.56%)	0.258 (0.45%)	108/65/124	297
	`AnnQ`	`Agree`	$0.482^{\S}$ (**4.41%**)	$0.320^{\S}$ (**4.02%**)	$0.340^{\S}$ (**5.37%**)	72/16/42	130
		`Differ`	0.361 (−4.81%)	0.210 (−3.53%)	0.198 (−4.08%)	65/58/103	226
`AnnD`	`Osrch`	`Agree`	0.504 (4.71%)	0.292 (3.49%)	$0.233^{\S}$ (11.41%)	30/7/22	59
		`Differ`	0.394 (−0.28%)	0.244 (0.09%)	0.267 (3.92%)	113/63/121	297
	`AnnQ`	`Agree`	$0.527^{\S}$ (**11.69%**)	$0.285^{\S}$ (**7.74%**)	$0.248^{\S}$ (**21.19%**)	50/12/32	94
		`Differ`	0.371 (−4.12%)	0.241 (−1.99%)	0.266 (0.47%)	93/58/111	262

of the 2008 ACM SIGMOD International Conference on Management of Data, pages 1247–1250. ACM, 2008.

[4] W. C. Brandão, R. L. Santos, N. Ziviani, E. S. Moura, and A. S. Silva. Learning to expand queries using entities. *Journal of the Association for Information Science and Technology (JASIST)*, 65(9):1870–1883, 2014.

[5] A. Z. Broder, M. Fontoura, E. Gabrilovich, A. Joshi, V. Josifovski, and T. Zhang. Robust classification of rare queries using web knowledge. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 231–238. ACM, 2007.

[6] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka Jr, and T. M. Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, volume 5, page 3, 2010.

[7] D. Carmel, M.-W. Chang, E. Gabrilovich, B.-J. P. Hsu, and K. Wang. ERD'14: Entity recognition and disambiguation challenge. In *SIGIR '14: Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2014.

[8] G. V. Cormack, M. D. Smucker, and C. L. Clarke. Efficient and effective spam filtering and re-ranking for large web datasets. *Information Retrieval*, 14(5):441–465, 2011.

[9] W. B. Croft, D. Metzler, and T. Strohman. *Search engines: Information Retrieval in practice.* Addison-Wesley Reading, 2010.

[10] J. Dalton, L. Dietz, and J. Allan. Entity query feature expansion using knowledge base links. In *Proceedings of the 37th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 365–374. ACM, 2014.

[11] L. Dietz and P. Verga. Umass at TREC 2014: Entity query feature expansion using knowledge base links. In *Proceedings of The 23st Text Retrieval Conference*, page To Appear. NIST, 2014.

[12] P. Ferragina and U. Scaiella. Fast and accurate annotation of short texts with wikipedia pages. *arXiv preprint arXiv:1006.3498*, 2010.

[13] E. Gabrilovich, M. Ringgaard, and A. Subramanya. FACC1: Freebase annotation of ClueWeb corpora, Version 1 (Release date 2013-06-26, Format version 1, Correction level 0), June 2013.

[14] G. Grefenstette and L. Wilber. *Search-Based Applications: At the Confluence of Search and Database Technologies.* 2010.

[15] T. Joachims. Making large-scale svm learning practical. LS8-Report 24, Universität Dortmund, LS VIII-Report, 1998.

[16] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 133–142. ACM, 2002.

[17] V. Lavrenko and W. B. Croft. Relevance based language models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 120–127. ACM, 2001.

[18] H. Li and J. Xu. Semantic matching in search. *Foundations and Trends in Information Retrieval*, 8:89, 2014.

[19] T.-Y. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009.

[20] X. Liu, P. Yang, and H. Fang. Entity came to rescue - leveraging entities to minimize risks in web search. In *Proceedings of The 23st Text Retrieval Conference, (TREC 2014)*, page To Appear. NIST, 2014.

[21] Y. Lu, H. Fang, and C. Zhai. An empirical study of gene synonym query expansion in biomedical information retrieval. *Information Retrieval Journal*, 12(1):51–68, 2009.

[22] Z. Lu, W. Kim, and W. J. Wilbur. Evaluation of query expansion using mesh in pubmed. *Information Retrieval Journal*, 12(1):69–80, 2009.

[23] D. Metzler and W. B. Croft. Latent concept expansion using markov random fields. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 311–318. ACM, 2007.

[24] D. Pan, P. Zhang, J. Li, D. Song, J.-R. Wen, Y. Hou, B. Hu, Y. Jia, and A. De Roeck. Using dempster-shafer's evidence theory for query expansion based on freebase knowledge. In *Information Retrieval Technology*, pages 121–132. Springer, 2013.

[25] T. Rajashekar and B. W. Croft. Combining automatic and manual index representations in probabilistic retrieval. *Journal of the American society for Information science*, 46(4):272–283, 1995.

[26] N. Stokes, Y. Li, L. Cavedon, and J. Zobel. Exploring criteria for successful query expansion in the genomic domain. *Information Retrieval*, 12(1):17–50, 2009.

[27] F. Xia, T.-Y. Liu, J. Wang, W. Zhang, and H. Li. Listwise approach to learning to rank: theory and algorithm. In *Proceedings of the 25th International Conference on Machine learning*, pages 1192–1199. ACM, 2008.

[28] C. Xiong and J. Callan. Query expansion with Freebase. In *Proceedings of the fifth ACM International Conference on the Theory of Information Retrieval*. ACM, 2015. To appear.

[29] Y. Xu, G. J. Jones, and B. Wang. Query dependent pseudo-relevance feedback based on wikipedia. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 59–66. ACM, 2009.