Explicit Semantic Ranking for Academic Search via Knowledge Graph Embedding

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

Russell Power
Allen Institute for
Artificial Intelligence
Seattle, WA, USA
russellp@allenai.org

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

ABSTRACT

This paper introduces Explicit Semantic Ranking (ESR), a new ranking technique that leverages knowledge graph embedding. Analysis of the query log from our academic search engine, SemanticScholar.org, reveals that a major error source is its inability to understand the meaning of research concepts in queries. To addresses this challenge, ESR represents queries and documents in the entity space and ranks them based on their semantic connections from their knowledge graph embedding. Experiments demonstrate ESR's ability in improving Semantic Scholar's online production system, especially on hard queries where word-based ranking fails.

Keywords

Academic Search, Entity-based Ranking, Knowledge Graph

1. INTRODUCTION

The Semantic Scholar (S2) launched in late 2015, with the goal of helping researchers find papers without digging through irrelevant information. The project has been a success, with over 3 million visits in the past year. Its current production ranking system is based on the word-based model in ElasticSearch that matches query terms with various parts of a paper, combined with document features such as citation count and publication time in a learning to rank architecture [15]. In user studies conducted at Allen Institute, this ranking model provides at least comparable accuracy with other academic search engines.

Analysis of S2's query logs found that a large fraction of the online traffic is ad-hoc queries about computer science concepts or research topics. The information needs behind such queries sometimes are hard for term-frequency based ranking models to fulfill. For example, a user entering the query 'dynamic programming segmentation' has a complex semantic intent, roughly described as 'show me NLP papers that use dynamic programming algorithms to solve the word segmentation problem'. Our error analysis using user clicks found that word-based ranking models sometimes fail to capture the semantic meaning behind such queries. This constitutes a major error source in S2's ranking.

This paper introduces Explicit Semantic Ranking (ESR), a new ranking technique to connect query and documents using semantic information from a knowledge graph. We first build an academic knowledge graph using S2's corpus and Freebase. The knowledge graph includes concept entities, their descriptions, context correlations, relationships with authors and venues, and embeddings trained from the graph structure. We apply this knowledge graph and embeddings to our ranking task. Queries and documents are represented by entities in the knowledge graph, providing 'smart phrasing' for ranking. Semantic relatedness between query and document entities is computed in the embedding space, which provides a soft matching between related entities. ESR uses a two-stage pooling to generalize these entity-based matches to query-document ranking features and uses a learning to rank model to combine them.

The explicit semantics from the knowledge graph has the ability to improve ranking in multiple ways. The entities and surface names help the ranking model recognize which part of a query is an informative unit, and whether different phrases have the same meaning, providing a powerful *exact match* signal. Embeddings of entities from the knowledge graph can also be leveraged to provide a *soft match* signal, allowing ranking of documents that are semantically related but do not match the exact query terms. Our experiments on S2's ranking benchmark dataset demonstrate the effectiveness of this explicit semantics. ESR improves the already-reliable S2's online production system by more than 10%. The gains are robust, with bigger gains and smaller errors, and also favoring top ranking positions. Further analysis confirms that both exact match and soft match in the entity space provide effective and novel ranking signals. These signals are successfully utilized by ESR's ranking framework, and greatly improve the queries that S2's word-based ranking system finds hard to deal with.

In the rest of this paper, Section 2 discusses related work; Section 3 analyzes S2's search traffic; Section 4 discusses our Explicit Semantic Ranking system; Sections 5 and 6 describe experimental methodology and evaluation; conclusions and future work are in Section 7.

*Part of this work was done when the first author was interning at Allen Institute for Artificial Intelligence.

WWW 2017, April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4913-0/17/04.
http://dx.doi.org/10.1145/3038912.3052558

2. RELATED WORK

Prior research in academic search is more focused on the analysis of the academic graph than on ad-hoc ranking. Microsoft uses its Microsoft Academic Graph to build academic dialog and recommendation systems [20]. Other research on academic graphs includes the extraction and disambiguation of authors, integration of different publication resources [21], and expert finding [2, 7, 29]. The academic graph can also be used to model the importance of papers [23] and to extract new entities [1].

Soft match is a widely studied topic in information retrieval, mostly in word-based search systems. Translation models treat ranking as translations between query terms and document terms using a translation matrix [3]. Topic modeling techniques have been used to first map query and document into a latent space, and then matching them in it [24]. Word embedding and deep learning techniques have been studied recently. One possibility is to first build query and document representations using their words' embeddings heuristically, and then match them in the embedding space [22]. The DSSM model directly trains a representation model using deep neural networks, which learns distributed representations for the query and document, and matches them using the learned representations [13]. A more recent method, DRMM, models the query-document relevance with a neural network built upon the word-level translation matrix [11]. The translation matrix is calculated with pretrained word embeddings. The word-level translation scores are summarized by bin-pooling (histograms) and then used by the ranking neural network.

The recent development of knowledge graphs has motivated many new techniques in utilizing knowledge bases for text-centric information retrieval [9]. An intuitive way is to use the textual attributes of related entities to enrich the query representation. For example, Wikipedia articles have been used as a better corpus for pseudo relevance feedback to generate better expansion terms [28]. Freebase entities that are retrieved by the query [6] or frequently appear in top retrieved documents' annotations [10] are usually related to the query's information needs; better expansion terms can be found in these related entities' descriptions [26]. The related entities' textual attributes have also been used by entity query feature expansion (EQFE) to extract richer learning to rank features. Each textual attribute provides unique textual similarity features with the document [8].

Another way to utilize knowledge graphs in search is to use the entities as a source of additional connections from query to documents. Latent Entity Space (LES) uses query entities and closely related document entities as the latent space between query and documents, and the latent entities' descriptions are used to connect query and documents in an unsupervised model [16]. LES performs well with high-quality query annotations [17]. EsdRank uses entities from entity linking and entity search to build additional connections between query and documents [25]. The connections are expressed by textual features between the query, entities and candidate documents. A latent space learning to rank model jointly learns the connections from query to entities and the ranking of documents.

A more recent trend is to build entity-based text representations and improve word-based ranking with entity-based retrieval models. Entity-based language model uses the surface forms and entity names of document annotations to build an entity-oriented language model [12]. It successfully improves the retrieval accuracy when combined with typical word-based retrieval. A similar idea is used in the bag-of-entities model, in which the query and documents are represented by their entity annotations [27]. Boolean and frequency-based models in the entity space can improve the ranking accuracy of word-based retrieval. `ESR` further explores the entity-based representation's potential with knowledge graph embeddings and soft matches, and uses the knowledge graph in a new domain: academic search.

3. QUERY LOG ANALYSIS

`S2`'s current ranking system is built on top of ElasticSearch's vector space model. It computes a ranking score based on the tf.idf of the query terms and bi-grams on papers' title, abstract, body text and citation context. Static ranking features are also included, for example, the number of citations, recent citations, and the time of publication. To handle the request for an author or for a particular paper by title, a few additional features match explicitly against authors and boost exact title matches. The ranking system was trained using learning to rank on a training set created at Allen Institute. Before and after release, several rounds of user studies were conducted by researchers at Allen Institute (mostly Ph.D.'s in Computer Science) and by a third party company. The results agree that on computer science queries `S2` performs at least on par with other academic search engines on the market.

The increasing online traffic in `S2` makes it possible to study the information needs in academic search, which is important for guiding the development of ranking models. For example, if users are mostly searching for paper titles, the ranking would be straightforward exact-matches; if users are mostly searching for author names, the ranking would be mainly about name disambiguation, aggregation, and recognition.

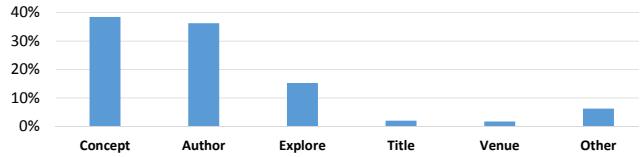
We manually labeled the intents of S2's 400 most frequent queries in the first six months of 2016. A query was labeled based on its search results and clicks. The result shows that the information needs on the head traffic can be categorized into the following categories:

- Concept: Searching for a research concept, e.g. 'deep reinforcement learning';
- Author: Searching for a specific author;
- Exploration: Exploring a research topic, e.g. 'forensics and machine learning';
- Title: Searching for a paper using its title;
- Venue: Searching for a specific conference; and
- Other: Unclear or not research related.

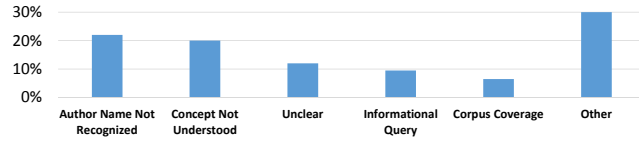
Figure 1a shows the distribution of these intents. More than half of `S2`'s head traffic is research concept related: searching for an academic concept (39%) and exploring research topics (15%). About one-third is searching an author (36%). Very little of the head queries are paper titles; most of such queries are in the long tail of the online traffic.

The large fraction of concept related queries shows that much of academic search is an ad-hoc search problem. The queries are usually short, on average about 2-3 terms, and their information needs are not as clear as the author, venue, or paper title queries. They are very typical ad-hoc search queries, in a specific domain – computer science.

To understand where `S2` is making mistakes, we studied the error sources of the failure queries in the query log.



(a) Query Intent Distribution



(b) Error Source Distribution

Figure 1: Query log analysis in S2's search traffic in the first six months of 2016. Query intents are manually labeled on the 400 most frequent queries. Error sources are manually labeled on the 200 worst performing queries.

These failure queries were picked based on the average click depth in the query log: The lower the click, the worse S2 might be performing. Among the queries with more than 10 clicks, we manually labeled the top 200 worst performing ones. The distribution of error types is shown in Figure 1b. The two major causes of failure are *author name not recognized* (22%) and *concept not understood* (20%).

S2's strategy for author queries is to show the author's page. When the author name is not recognized, S2 uses its normal ranking based on papers' textual similarity with the author name, which often results in unrelated papers.

A *concept not understood* error occurs when S2 returns papers that do not correctly match the semantic meanings of concept queries. Since this is the biggest error source in S2's ad-hoc ranking part, we further analyzed what makes these queries difficult.

The first part of the difficulty is noise in the exact-match signal. Due to language variety, the query term may not appear frequently enough in the relevant documents (vocabulary mismatch), for example, 'softmax categorization' versus 'softmax classification'. The segmentation of query concepts is also a problem. For example, the whole query 'natural language interface' should be considered as a whole because it is one informative unit, but the ranking model matches all words and n-grams of the query, and the result is dominated by the popular phrase 'natural language'.

The second part is more subtle as it is about the meaning of the query concept. There can be multiple aspects of the query concept, and a paper that mentions it the most frequently may not be about the most important aspect. For example, 'ontology construction' is about how to construct ontologies in general, but may not be about how a specific ontology is constructed; 'dynamic programming segmentation' is about word segmentation in which dynamic programming is essential, but is not about image segmentation.

To conclude, our analysis finds that there is a gap between query-documents' textual similarity and their semantic relatedness. Ranking systems that rely solely on term-level statistics have few principled ways to resolve this discrepancy. Our desire to handle this gap led to the development of ESR.

4. EXPLICIT SEMANTIC RANKING

This section first describes the knowledge graph we constructed, and then introduces our Explicit Semantic Ranking (ESR) method that models the relatedness of query and documents using the semantics from the knowledge graph.

4.1 Knowledge Graph

The prerequisite of semantic ranking systems is a knowledge graph that stores semantic information. Usually, an entity linking system that links natural language text with entities in the knowledge graph is also necessary [8, 16, 25]. In this work, we build a standard knowledge graph G with concept entities (E) and edges (predicates P and tails T) by harvesting S2's corpus and Freebase, and use a popularity based entity linking system to link query and documents.

Concept Entities in our knowledge graph can be collected from two different sources: corpus-extracted (Corpus) and Freebase. Corpus entities are keyphrases automatically extracted from S2's corpus. Keyphrases extraction is a widely studied task that aims to find representative keyphrases for a document, for example, to approximate the manually assigned keywords of a paper. This work uses the keyphrases extracted by S2's production system, which extracts noun phrases from a paper's title, abstract, introduction, conclusion and citation contexts, and selects the top ranked ones in a keyphrase ranking model with typical features such as frequency and location [4].

The second source of entities is Freebase. Despite being a general domain knowledge base, our manual examination found that Freebase has rather good coverage on the computer science concept entities in S2's head queries.

Entity Linking: We use CMNS which links surface forms (entity mentions) in a query or document to their most frequently linked entities in Google's FACC1 annotation [10, 12, 27]. Although CMNS does not provide entity disambiguation, it has been shown to be effective for query entity linking [12], and the language in computer science papers is less ambiguous than in a more general domain.

Edges in our knowledge graph include two parts: Predicates P and tails T. From Freebase and the CMNS annotated S2 corpus, the following four types of edges are harvested:

- Author edges link an entity to an author if the author has a paper which mentioned the entity in the title;
- Context edges link an entity to another entity if the two co-occur in a window of 20 words more than 5 times in the corpus;
- Desc edges link an entity to words in its Freebase description (if one exists); and
- Venue edges link an entity to a venue if the entity appears in the title of a paper published in the venue.

The knowledge graph G contains two types of entities: Corpus-extracted (Corpus) and Freebase, and four types of edges: Author, Context, Desc and Venue.

Embeddings of our entities are trained based on their neighbors in the knowledge graph. The graph structure around an entity conveys the semantics of this entity. Intuitively, entities with similar neighbors are usually related. We use entity embeddings to model such semantics.

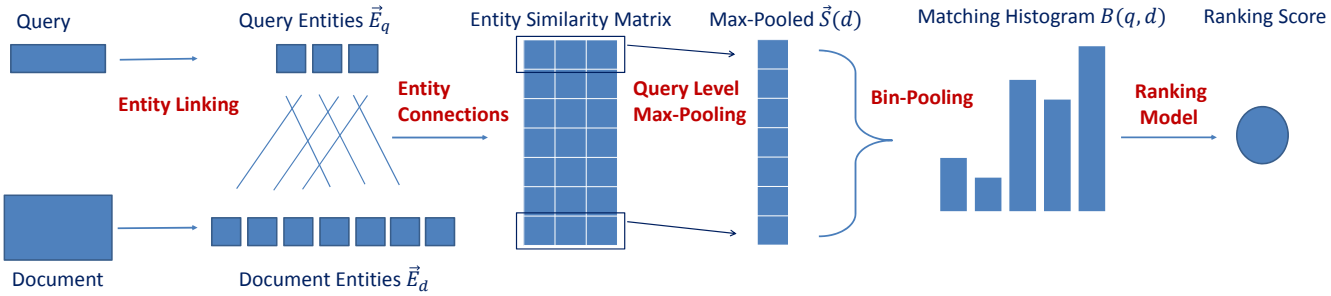


Figure 2: The Framework of Explicit Semantic Ranking (ESR).

We train a separate embedding model for each of {Author, Context, Desc, Venue} edges using the skip-gram model [18]:

$$l = \sum_{e \in E, t \in T} w(e,t)\big(\sigma(V(e)^T U(t)) - E_{\hat{t} \sim T}\sigma(-V(e)^T U(\hat{t}))\big).$$

The loss function l is optimized using a typical gradient method. V and U are the entity embedding matrices learned for entities and tails of this edge type. Each of their rows ($V(e)$ or $U(t)$) is the embedding of an entity e or a tail t, respectively. σ is the Sigmoid function. T is the collection of all tails for this edge type. $E_{\hat{t} \sim T}()$ samples negative instances based on the tails' frequency (negative sampling). $w(e,t)$ is the frequency of entity e and tail t being connected, for example, how many times an entity is used by an author.

4.2 Ranking Model

Given a query q, and a set of candidate documents $D = \{d_1, ..., d_n\}$, the goal of ESR is to find a ranking function $f(q, d|G)$, that better ranks D using the explicit semantics stored in the knowledge graph G. The explicit semantics include entities ($E = \{e_1, ..., e_{|E|}\}$) and edges (predicates P and tails T). The rest of this section describes the ESR framework, which is also shown in Figure 2.

Entity Based Representations.

ESR represents query and documents by their bag-of-entities constructed using their entity annotations linked by CMNS [19, 27]. Each query or document is represented by a vector ($\vec{E}_q$ or $\vec{E}_d$). Each dimension in the vector corresponds to an entity e in the query or document's annotation, and the weight is the frequency of the entity being annotated to it.

Match Query and Documents in the Entity Space.

Instead of being restricted to classic retrieval models [19, 27], ESR matches query and documents' entity representations using the knowledge graph embedding.

ESR first calculates a query-document entity translation matrix. Each element in the matrix is the connection strength between a query entity e_i and a document entity e_j, calculated by their embeddings' cosine similarity:

$$s(e_i, e_j) = \cos(V(e_i), V(e_j)). \quad (1)$$

A score of 1 in the entity matrix refers to an *exact match* in the entity space. It incorporates the semantics from entities and their surface forms: The entities in the text are recognized, different surface forms of an entity are aligned, and the exact match is done at the entity level. We call this effect 'smart phrasing'. Scores less than 1 identify related entities as a function of the knowledge graph structure and provide *soft match* signals.

Then ESR performs two pooling steps to generalize the exact matches and soft matches in the entity translation matrix to query-document ranking evidence.

The first step is a max-pooling along the query dimension:

$$\vec{S}(d) = \max_{e_i \in \vec{E}_q} s(e_i, \vec{E}_d). \quad (2)$$

$\vec{E}_q$ and $\vec{E}_d$ are the bag-of-entities of q and d. $\vec{S}(d)$ is a $|\vec{E}_d|$ dimensional vector. Its j^{th} dimension is the maximum similarity of the document entity e_j to any query entities.

The second step is a bin-pooling (histogram) to count the matches at different strengths [11]:

$$B_k(q,d) = \log \sum_j I(st_k \le \vec{S}_j(d) < ed_k). \quad (3)$$

$[st_k, ed_k)$ is the range for the k^{th} bin. B_k is the number of document entities whose scores fall into this bin.

The max-pooling matches each document entity to its closest query entity using embeddings, which is the exact-match if one exists. Its score describes how closely related a document entity is to the query. The bin-pooling counts the number of document entities with different connection strengths to the query. The bin with range $[1, 1]$ counts the exact matches, and the other bins generate soft match signals [11]. The two pooling steps together summarize the entity matches to query-document ranking evidence.

Ranking with Semantic Evidence.

The bin scores B are used as features for standard learning to rank models in ESR:

$$f(q, d|G) = w_0 f_{S2}(q,d) + W^T B(q,d) \quad (4)$$

where $f_{S2}(q,d)$ is the score from S2's production system, w_0 and W are the parameters to learn, and $f(q, d|G)$ is the final ranking score. Based on which edge type the entity embedding is trained, there are four variants of ESR: ESR-Author, ESR-Context, ESR-Desc, and ESR-Venue.

With entity-based representations, the exact matches (smart phrasing) allow ESR to consider multiple words as a single unit in a principled way, and the knowledge graph embeddings allow ESR to describe semantic relatedness via soft matches. The exact match and soft match signals are utilized by ESR's unified framework of embedding, pooling, and ranking.

5. EXPERIMENTAL METHODOLOGY

This section describes the ranking benchmark dataset of S2 and the experimental settings.

5.1 Ranking Benchmark

The queries of the benchmark are sampled from S2's query log in the first six months of 2016. There are in total 100 queries, 20 uniformly sampled from the head traffic (100 most frequent queries), 30 from the median (queries that appear more than 10 times), and 50 hard queries from the 200 worst performing queries based on the average click depth. Author and venue queries are manually ignored as they are more about name recognition instead of ranking.

The candidate documents were generated by pooling from several variations of S2's ranking system. First, we labeled several of the top ranked results from S2. Then several variations of S2's ranking systems, with same features, but different learning to rank models, are trained and tested on these labels using cross-validation. The top 10 results from these rankers obtained from cross-validation were added to the document pool. They also could be labeled for another iteration of training. The training-labeling process was repeated twice; after that, rankings had converged.

We also tried pooling with classic retrieval models such as BM25 and the query likelihood model, but their candidate documents were much worse than the production system. Our goal was to improve an already-good system, so we chose to use the higher quality pool produced by S2 variants.

We used the same five relevance categories used by the TREC Web Track. Judging the relevance of research papers to academic queries requires a good understanding of related research topics. It is hard to find crowd-sourcing workers with such research backgrounds. Thus we asked two researchers in the Allen Institute to label the query-document pairs. Label differences were resolved by discussion.

The distribution of relevance labels in our dataset is shown in Table 1. The same statistics from the TREC Web Track 2009-2012 are also listed for reference. Our relevance judgments share a similar distribution, although our data is cleaner, for example, due to a lack of spam.

The benchmark dataset is available at `http://boston.lti.cs.cmu.edu/appendices/WWW2016/`.

5.2 Experimental Settings

Data: Ranking performances are evaluated on the benchmark dataset discussed in Section 5.1. The entity linking performance is evaluated on the same queries with manual entity linking from the same annotators.

Baselines: The baseline is the Semantic Scholar (S2) production system on July 1st 2016, as described in Section 3. It is a strong baseline. An internal evaluation and a third-party evaluation indicate that its accuracy is at least as good as other academic search engines on the market. We also include BM25-F and tf.idf-F for reference. The BM25 and vector space model are applied to the paper's title, abstract, and body fields. Their parameters (field weights) are learned using the same learning to rank model as our method in the same cross-validation setting.

Evaluation Metrics: Query entity linking is evaluated by Precision and Recall at the query level (strict evaluation [5]) and the average of query level and entity level (lean evaluation [12]). Ranking is evaluated by NDCG@20, the main evaluation metric in the TREC Web Track. As an online production system, S2 especially cares about top positions, so NDCG@$\{1, 5, 10\}$ are also evaluated. Statistical significances are tested by permutation test with $p < 0.05$.

Table 1: Distribution of relevance labels in Semantic Scholar's benchmark dataset. **S2** shows the number and percentage of query-document pairs from the 100 testing queries that are labeled to the corresponding relevance level. **TREC** shows the statistics of the relevance labels from TREC Web Track 2009-2012's 200 queries.

Relevance Level	S2		TREC	
Off-Topic (0)	3080	65.24%	54660	77.45%
Related (1)	1060	22.45%	10778	15.27%
Relevant (2)	317	6.71%	3681	5.22%
Exactly-Right (3)	213	4.51%	598	0.85%
Navigational (4)	51	1.08%	858	1.22%

Table 2: Entity linking evaluation results. Entities are linked by CMNS. Corpus shows the results when using automatically extracted keyphrases as the targets. Freebase shows the results when using Freebase entities as the targets. **Prec**ison and **Rec**all from lean evaluation and strict evaluation are displayed.

	Lean Evaluation		Strict Evaluation	
	Prec	Rec	Prec	Rec
Corpus	0.5817	0.5625	0.5400	0.5400
Freebase	0.6960	0.6958	0.6800	0.6800

ESR Methods: Based on which edge type is used to obtain the entity embedding, there are four versions of ESR: ESR-Author, ESR-Context, ESR-Desc, and ESR-Venue. Embeddings for ESR-Author and ESR-Venue are trained with authors and venues with more than 1 publication. Description and context embeddings are trained with entities and terms with the minimum frequency of 5.

Entity linking is done by CMNS with all linked entities kept to ensure recall [12, 27]. Corpus entities do not have multiple surface forms so CMNS reduces to exact match. Freebase entities are linked using surface forms collected from Google's FACC1 annotation [10].

Entity connection scores are binned into five bins: $[1, 1], [0.75, 1), [0.5, 0.75), [0.25, 0.5), [0, 0.25)$ with the exact match bin as the first bin [11]. We discard the negative bins as negative cosine similarities between entities are not informative: most of them are not related at all.

All other settings follow previous standards. The embedding dimensionality is 300; The five bins and three paper fields (title, abstract, and body) generate 15 features, which are combined with S2's original score using linear RankSVM [14]. All models and baselines' parameters are trained and evaluated using a 10-fold cross validation with 80% train, 10% development and 10% test in each fold. The hyper-parameter 'c' of RankSVM is selected from $\{0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1\}$ using the development part of each fold.

6. EVALUATION RESULTS

Five experiments investigated entity linking and document ranking accuracy, as well as the effects of three system components (entity based match, embedding, pooling).

Table 3: Overall accuracy of ESR compared to Semantic Scholar (S2). ESR-Author, ESR-Context, ESR-Desc and ESR-Venue are ESR with entity embedding trained from corresponding edges. Relative performances compared with S2 are in percentages. **W**in/**T**ie/**L**oss are the number of queries a method improves, does not change, or hurts, compared with S2. Best results in each metric are marked **Bold**. Statistically significant improvements (P>0.05) over S2 are marked by †.

Method	NDCG@1		NDCG@5		NDCG@10		NDCG@20		W/T/L
tf.idf-F	0.2020	−59.65%	0.2254	−54.93%	0.2741	−47.57%	0.3299	−39.91%	28/01/71
BM25-F	0.2512	−49.81%	0.2890	−42.20%	0.3150	−39.74%	0.3693	−32.75%	32/01/67
Semantic Scholar	0.5006	–	0.5000	–	0.5228	–	0.5491	–	–/–/–
ESR-Author	0.5499†	+9.85%	0.5501†	+10.02%	0.5671†	+8.47%	0.5935†	+8.08%	60/10/30
ESR-Context	0.5519†	+10.25%	0.5417†	+8.35%	0.5636†	+7.80%	0.5918†	+7.77%	58/04/38
ESR-Desc	0.5304	+5.96%	0.5496†	+9.92%	0.5536†	+5.88%	0.5875†	+6.99%	55/11/34
ESR-Venue	**0.5638**†	+12.63%	**0.5700**†	+13.99%	**0.5795**†	+10.83%	**0.6090**†	+10.91%	59/11/30

Figure 3: ESR's relative NDCG@20 compared with Semantic Scholar (S2) on individual queries. Each point corresponds to a query. The value on the x-axis is S2's NDCG@20 on each query. The y-axis shows ESR's relative NDCG@20 (percentage) compared with S2.

6.1 Entity Linking Performance

The entity linking accuracy of CMNS on our queries is shown in Table 2. Corpus and Freebase refer to using entities from extracted keyphrases in S2's corpus or Freebase. **Prec**ison and **Rec**all are evaluated by lean evaluation (query and entity averaged) and strict evaluation (query only) metrics.

The results in Table 2 reveal a clear gap between the quality of automatically extracted entities and manually curated entities. Linking performance is 10-25% better with Freebase entities than Corpus entities on all evaluation metrics, demonstrating the significant differences in the quality of their entity sets. Further analysis finds that Freebase not only provides a larger set of surface forms, but also a cleaner and larger set of computer science concept entities. Indeed, our manual examination found that only the most frequent automatically extracted keyphrases (about ten thousand) are reliable. After that, there is much noise. Those most frequent keyphrases are almost all included in Freebase; little additional information is provided by the Corpus entities.

The absolute Precision and Recall are higher than entity linking with Freebase on the general domain (TREC Web Track) queries [12, 27]. Our manual examination finds that a possible reason is the lower ambiguity in academic queries than in TREC Web Track queries. Also, since our queries are from the head and middle of the online traffic, they are mostly about popular topics. Freebase's coverage on them is no worse than on general domain queries. Due to its dominating entity linking accuracy, the rest of our experiments used only Freebase.

6.2 Ranking Performance

The four versions of ESR only differ in the edges they used to obtain the entity embedding. Relative performances comparing with the production system Semantic Scholar (S2) are shown in percentages. **W**in/**T**ie/**L**oss are the number of queries outperformed, unchanged, and hurt by each method compared with S2. Statistically significant improvements over S2 are marked by †.

The production system S2 outperforms BM25-F by a large margin. NDCG@1 is almost doubled. This result confirms the quality of the current production system. Although half of the queries were deliberately picked to be difficult, at all depths S2 achieves absolute NDCG scores above 0.5.

ESR provides a further jump over the production system by at least 5% on all evaluation metrics and with all edge types. With Venue, the improvements are consistently higher than 10%. On the earlier positions which are more important for user satisfaction, ESR's performances are also stronger, with about $2-3\%$ more improvements on NDCG@1 and NDCG@5 than on NDCG@20. The improvements are statistically significant, with the only exception of ESR-Desc on

NDCG@1. We think this behavior is due to the edge types `Author`, `Context`, and `Venue` being domain specific, because they are gathered from `S2`'s corpus, whereas `Desc` is borrowed from Freebase and no specific attention is paid to the computer science domain.

`ESR` is designed to improve the queries that are hard for `Semantic Scholar`. To verify that `ESR` fulfills this requirement, we plot `ESR`'s performance on individual queries with respect to `S2`'s in Figure 3. Each point in the figure corresponds to a query. `S2`'s NDCG@20 is shown in the x-axis. The relative performance of `ESR` compared with `S2` is shown in the y-axis.

In all four sub-figures, `ESR`'s main impact is on the hard queries (the left side). On the queries where `S2` already performs well, `ESR` makes only small improvements: Most queries on the right side stay the same or are only changed a little bit. On the queries where `S2`'s word-based ranking fails, for example, on those whose NDCG < 0.3, the semantics from the knowledge graph improve many of them with large margins. The improvements are also robust. More than half of the queries are improved with big wins. Fewer queries are hurt and the loss is usually small.

This experiment demonstrates `ESR`'s ability to improve a very competitive online production system. `ESR`'s advantages also favor online search engines' user satisfaction: More improvements are at early ranking positions, the improvements are robust with fewer queries badly damaged, and most importantly, `ESR` fixes the hard queries that the production system finds difficult.

6.3 Effectiveness of Entity-Based Match

`ESR` matches query and documents on their entity representations using the semantics from the knowledge graph. The semantic contribution to `ESR`'s ranking can come from two aspects: exact match with smart phrasing and soft match with knowledge graph embedding. The third experiment investigated the effectiveness of `ESR`'s entity-based matching, both exact match and soft match.

Recall that `ESR` uses five bins to combine and weight matches of different strengths. This experiment used the same ranking model, but varies the bins used. We started with only using the first exact match bin $[1, 1]$; then we added in the second bin $[0.75, 1)$, the third $[0.5, 0.75)$, and the fourth $[0.25, 0.5)$, and evaluated effectiveness of exact match and soft matches at varying strength. The results with different numbers of bins are shown in Figure 4. For brevity, only NDCG@20 is shown (the y-axis). The behavior is similar for other depths. Results of `Semantic Scholar` and `ESR` with all bins are also displayed for comparison.

Entities' exact match information (the first bin) provides about 6% gains over `S2`, with `Context` and `Venue`. The exact match signals with `ESR-Author` and `ESR-Desc` are sparse, because some entities do not have `Author` or `Desc` edges. In that case, their rankings are reduced to `S2`, and their gains are smaller. Our manual examination found that this gain does come from the 'smart phrasing' effect: an entity's mention is treated as a whole unit and different phrases referring to the same entity are aligned. For example, the rankings of queries like 'natural language interface' and 'robust principle component analysis' are greatly improved, while in `S2` their textual similarity signals are mixed by their individual terms and sub-phrases.

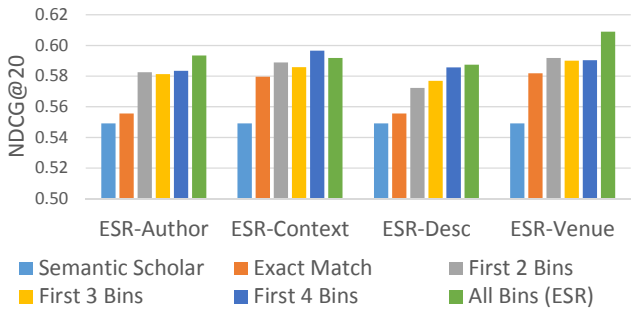


Figure 4: `ESR` accuracy using different numbers of matching bins. For each group, from left to right: `Semantic Scholar`, the baseline; `Exact match` which only uses the first bin; `First 2 Bins`, `First 3 Bins`, and `First 4 Bins` which refer to only using the first k bins with the highest matching scores; the last one `All Bins` is the original `ESR`.

The soft match information (later bins) contributes approximately another 5% of improvement. The soft-match bins record how many document entities are related to the query entities at certain strengths. Intuitively, the soft match should contribute for all queries as knowing more about the document entities should always help. But our examination finds this information is more effective on some queries, for example, 'dynamic segmentation programming', 'ontology construction' and 'noun phrases'. The soft match helps `ESR` find the papers whose meanings (e.g. research area, topic, and task.) are semantically related to these queries' information needs, while `S2`'s word-based match fails to do so.

This experiment helps us understand why the explicit semantics in knowledge graphs is useful for ranking. By knowing which entity a phrase refers to and whether different phrases are about the same thing, the *exact match* signal is polished. By knowing the relatedness between query entities and document entities through their embeddings, additional *soft match* connections that describe query-document relatedness with semantics are incorporated.

6.4 Effectiveness of Embedding

`ESR` captures the semantic relatedness between entities by using their distance in the embedding space. An advantage of using embeddings compared with the raw knowledge graph edges is their efficiency, which is tightly constrained in online search engines. By embedding the neighbors of an entity into a dense continuous vector, at run time, `ESR` avoids dealing with the graph structure of the knowledge graph, which is sparse, of varying length, and much more expensive to deal with than fixed length dense vectors.

However, does `ESR` sacrifice effectiveness for efficiency? The fourth experiment studied the influence of embeddings on `ESR`'s effectiveness, by comparing it with the ranking performance of using raw knowledge graph edges. In this experiment, a discrete vector representation is formed for each entity for each edge type. Each of the vector's dimensions is a neighbor of the entity. Its weight is the frequency of them being connected together. The `Raw` variants of ESR are then formed with the knowledge graph embedding replaced by this discrete entity representation and everything else kept the same.

Table 4: Performance of different strategies that make use of the knowledge graph in ranking. Raw directly calculates the entity similarities in the original discrete space. Mean uses mean-pooling when generalizing the entity translation matrix to query-document ranking evidence. Max uses max-pooling. Mean&Bin replaces the max-pooling in ESR's first stage with mean-pooling. Relative performances (percentages), statistically significant differences (†), and **W**in/**T**ie/**L**oss are compared with the ESR version that uses the same edge type and embedding; for example, Raw-Author versus ESR-Author.

Method	NDCG@20		W/T/L
ESR-Author	0.5935	–	–/–/–
ESR-Context	0.5918	–	–/–/–
ESR-Desc	0.5875	–	–/–/–
ESR-Venue	**0.6090**	–	–/–/–
Raw-Author	0.5821	−1.91%	45/06/49
Raw-Context	0.5642†	−4.66%	38/06/56
Raw-Desc	0.5788	−1.48%	46/05/49
Raw-Venue	0.5576†	−8.43%	28/07/65
Mean-Author	0.5685†	−4.22%	33/09/58
Mean-Context	0.5676†	−4.08%	39/06/55
Mean-Desc	0.5660	−3.66%	45/12/43
Mean-Venue	0.5599†	−8.07%	32/10/58
Max-Author	0.5842	−1.56%	38/10/52
Max-Context	0.5861	−0.95%	52/07/41
Max-Desc	0.5659†	−3.67%	41/12/47
Max-Venue	0.5763†	−5.38%	32/12/56
Mean&Bin-Author	0.5823	−1.89%	41/06/53
Mean&Bin-Context	0.5808	−1.85%	41/08/51
Mean&Bin-Desc	0.5694	−3.07%	38/14/48
Mean&Bin-Venue	0.5639†	−7.41%	31/10/59

Table 4 shows the results of such 'Raw' methods. Different versions of Raw use different edges (Author, Context, Desc and Venue) to represent entities. The relative performance and **W**in/**T**ie/**L**oss are compared with the corresponding ESR version that uses exactly the same information but with knowledge graph embedding. The result shows that ESR actually benefits from the knowledge graph embedding; Raw methods almost always perform worse than the corresponding ESR version. We believe that the advantage of knowledge graph embedding is similar with word embedding [18]: the embedding better captures the semantics by factoring the raw sparse data into a smooth low-dimensional space.

6.5 Effectiveness of Pooling

ESR applies a two stage pooling on the entity translation matrix to obtain query-document ranking evidence. The first, max-pooling, is used to match each document entity to its closest query entity. The second, bin-pooling, is used to summarize the matching scores of each document entity into match frequencies at different strengths. This experiment evaluates the effectiveness of ESR's pooling strategy.

We compare ESR's two-stage pooling with several common pooling strategies: mean-pooling that summarizes the translation matrix into one average score; max-pooling that only keeps the highest score in the matrix; and mean&bin-pooling which is the same as ESR's max-pooling and bin-pooling, but in the first step the document entities are assigned with their average similarities to query entities. Except for the pooling strategy, all other settings of ESR are kept.

The results of different pooling strategies are shown in the second half of Table 4. Relative performances and **W**in/**T**ie/**L**oss are compared with the corresponding ESR version. The results demonstrate the effectiveness of ESR's two-stage pooling strategy: All other pooling strategies perform worse. Abstracting the entire entity translation matrix into one mean or max score may lose too much information. Mean&bin pooling also performs worse. Intuitively, we would prefer document entities that match one of the query entities very well, rather than entities that have mediocre matches with multiple query entities. Also, the exact match information is not preserved in the mean&bin-pooling when there are multiple query entities.

This result shows that generalizing the entity level evidence to the query-document level is not an easy task. One must find the right level of abstraction to obtain a good result. There are other abstraction strategies that are more complex and could be more powerful. For example, one can imagine building an RNN or CNN upon the translation matrix, but that would require more training labels. We believe ESR's two-stage pooling provides the right balance of model complexity and abstraction granularity, given the size of our training data.

7. CONCLUSIONS AND FUTURE WORK

Analysis of Semantic Scholar's query logs revealed that a large percentage of head queries involve research concepts, and that a major source of error was the inability of even a well-tuned bag-of-words system to rank them accurately. To address this challenge, we developed Explicit Semantic Ranking (ESR), a new technique that utilizes the explicit semantics from a knowledge graph in academic search. In ESR, queries and documents are represented in the entity space using their annotations, and the ranking is defined by their semantic relatedness described by their entities' connections, in an embedding, pooling, and ranking framework.

Experiments on a Semantic Scholar testbed demonstrate that ESR improves the production system by 6% to 14%. Additional analysis revealed the effectiveness of the explicit semantics: the entities and their surface forms help recognize the concepts in a query, and polish the *exact match* signal; the knowledge graph structure helps build additional connections between query and documents, and provides effective and novel *soft match* evidence. With the embedding, pooling, and ranking framework that successfully utilizes this semantics, ESR provides robust improvements, especially on the queries that are hard for word-based ranking models.

Perhaps surprisingly, we found that using Freebase entities was more effective than keyphrases automatically extracted from S2's corpus. Although Freebase is considered general-purpose, it provides surprisingly good coverage of the entities in our domain - computer science. The value of this knowledge graph can be further improved by adding domain specific semantics such as venues and authors. This result is encouraging because good domain-specific knowledge bases can be difficult to build from scratch. It shows that entities from a general-domain knowledge base can be a good start.

Different edge types in the knowledge graph convey rather different semantics. In our experiments, there is less than 15% overlap between the close neighbors of an entity in dif-

ferent embedding spaces. The different variants of ESR also perform rather differently on different queries. This work mainly focuses on how to make use of each edge type individually. An important future research direction is to study how to better utilize knowledge graph semantics in a more unified way, perhaps with the help of ranking labels.

This paper presents a new method of using knowledge graphs to improve the ranking of academic search. Although the details of this work are specific to Semantic Scholar, the techniques and lessons learned are general and can be applied to other full-text search engines.

8. ACKNOWLEDGMENTS

This research was supported by National Science Foundation (NSF) grant IIS-1422676 and a gift from the Allen Institute for Artificial Intelligence. Any opinions, findings, and conclusions in this paper are the authors' and do not necessarily reflect those of the sponsors.

9. REFERENCES

[1] A. Arnold and W. W. Cohen. Information extraction as link prediction: Using curated citation networks to improve gene detection. In *International Conference on Wireless Algorithms, Systems, and Applications*, pages 541–550. Springer, 2009.

[2] K. Balog, L. Azzopardi, and M. De Rijke. Formal models for expert finding in enterprise corpora. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR 2006)*, pages 43–50. ACM, 2006.

[3] 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 (SIGIR 1999)*, pages 222–229. ACM, 1999.

[4] C. Caragea, F. A. Bulgarov, A. Godea, and S. Das Gollapalli. Citation-enhanced keyphrase extraction from research papers: A supervised approach. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1435–1446. Association for Computational Linguistics, 2014.

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

[6] J. Chen, C. Xiong, and J. Callan. An empirical study of learning to rank for entity search. In *Proceedings of the 39th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval,(SIGIR 2016)*. ACM, 2016. To Appear.

[7] N. Craswell, A. P. de Vries, and I. Soboroff. Overview of the trec 2005 enterprise track. In *Proceedings of The 14th Text Retrieval Conference (TREC 2005)*, volume 5, pages 199–205, 2005.

[8] 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 (SIGIR 2014)*, pages 365–374. ACM, 2014.

[9] L. Dietz, A. Kotov, and E. Meij. Utilizing knowledge bases in text-centric information retrieval. In *Proceedings of the 2016 ACM on International Conference on the Theory of Information Retrieval (ICTIR 2016)*, pages 5–5. ACM, 2016.

[10] 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.

[11] J. Guo, Y. Fan, A. Qingyao, and W. Croft. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM 2016)*, page To Appear. ACM, 2016.

[12] F. Hasibi, K. Balog, and S. E. Bratsberg. Entity linking in queries: Tasks and evaluation. In *Proceedings of the Fifth ACM International Conference on The Theory of Information Retrieval (ICTIR 2015)*, pages 171–180. ACM, 2015.

[13] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management (CIKM 2013)*, pages 2333–2338. ACM, 2013.

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

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

[16] X. Liu and H. Fang. Latent entity space: a novel retrieval approach for entity-bearing queries. *Information Retrieval Journal*, 18(6):473–503, 2015.

[17] 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)*. NIST, 2014.

[18] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 2tth Advances in Neural Information Processing Systems 2013 (NIPS 2013)*, pages 3111–3119, 2013.

[19] H. Raviv, O. Kurland, and D. Carmel. Document retrieval using entity-based language models. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval (SIGIR 2016)*, pages 65–74. ACM, 2016.

[20] A. Sinha, Z. Shen, Y. Song, H. Ma, D. Eide, B.-j. P. Hsu, and K. Wang. An overview of microsoft academic service (MAS) and applications. In *Proceedings of the 24th International Conference on World Wide Web (WWW 2015)*, pages 243–246. ACM, 2015.

[21] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD 2008)*, pages 990–998. ACM, 2008.

[22] I. Vulić and M.-F. Moens. Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2015)*, pages 363–372. ACM, 2015.

[23] A. D. Wade, K. Wang, Y. Sun, and A. Gulli. Wsdm cup 2016: Entity ranking challenge. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining (WSDM 2016)*, pages 593–594. ACM, 2016.

[24] X. Wei and W. B. Croft. Lda-based document models for ad-hoc retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR 2006)*, pages 178–185. ACM, 2006.

[25] C. Xiong and J. Callan. EsdRank: Connecting query and documents through external semi-structured data. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management (CIKM 2015)*, pages 951–960. ACM, 2015.

[26] C. Xiong and J. Callan. Query expansion with Freebase. In *Proceedings of the fifth ACM International Conference on the Theory of Information Retrieval (ICTIR 2015)*, pages 111–120. ACM, 2015.

[27] C. Xiong, J. Callan, and T.-Y. Liu. Bag-of-entity representation for ranking. In *Proceedings of the sixth ACM International Conference on the Theory of Information Retrieval (ICTIR 2016)*, pages 181–184. ACM, 2016.

[28] 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 (SIGIR 2009)*, pages 59–66. ACM, 2009.

[29] J. Zhang, J. Tang, and J. Li. Expert finding in a social network. In *International Conference on Database Systems for Advanced Applications*, pages 1066–1069. Springer, 2007.