An Empirical Study of Learning to Rank for Entity Search

Jing Chen
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
jingc1@cs.cmu.edu

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 work investigates the effectiveness of learning to rank methods for entity search. Entities are represented by multi-field documents constructed from their RDF triples, and field-based text similarity features are extracted for query-entity pairs. State-of-the-art learning to rank methods learn models for ad-hoc entity search. Our experiments on an entity search test collection based on DBpedia confirm that learning to rank methods are as powerful for ranking entities as for ranking documents, and establish a new state-of-the-art for accuracy on this benchmark dataset.

Keywords

Entity Search, Learning to rank, Knowledge Base, DBpedia

1. INTRODUCTION

Publicly available knowledge bases such as DBpedia, Freebase, and Wikipedia are beginning to be used for tasks such as document ranking, card retrieval, and question answering [3, 4, 5]. A key component in many such systems is ad-hoc entity retrieval - using a query to retrieve one or more entities that satisfy some underlying information need. Several methods of representing and ranking entities have been developed recently, however the problem is far from solved.

Recent knowledge bases store information in RDF triples. A triple is a piece of information about the entity, for example its name, alias, category, description or relationship to another entity. Some prior research represents each entity as a structured document by grouping RDF triples into fields [1, 6] or a tree [2]. This allows entities to be retrieved by typical document retrieval algorithms such as BM25, query likelihood, or sequential dependency models.

This work investigates the effectiveness of learning to rank (LeToR) models, the state-of-the-art in ranking documents, for entity search. It represents entities following the previous state-of-the-art's multi-field representations [6], extracts text similarity features for query-entity pairs, and studies the effectiveness of state-of-the-art learning to rank models for ranking entities. Experimental results on an entity search test collection based on DBpedia [1] confirm that learning to rank is as powerful for entity ranking as for document ranking, and significantly improves the previous state-of-the-art. Results also indicate that learning to rank models with text similarity features are especially effective on keyword queries.

Our analysis further shows the influence of query types on learning to rank models. Different types of queries use different parts of an entity's representation. It is more effective to learn different models for different types of queries than to use a single model for all types of queries.

SIGIR '16, July 17 - 21, 2016, Pisa, Italy

ISBN 978-1-4503-4069-4/16/07...$15.00

DOI: http://dx.doi.org/10.1145/2911451.2914725

2. LEARNING TO RANK ENTITIES

The first question in entity search is how to represent entities. We follow Zhiltsov et al. [6] and group RDF triples into five fields: `Name`, which contains the entity's names; `Cat`, which contains its categories; `Attr`, which contains all attributes except name; `RelEn`, which includes the names of its neighbor entities; and `SimEn`, which contains its aliases. We include all the RDF triples in DBpedia in the fields.

In state-of-the-art learning to rank systems for document ranking, most features are the scores of common unsupervised ranking algorithms applied to different document representations (fields). The different ranking algorithms and representations provide different views of the relevance of the document to the query. The multiple perspectives represented by these features are the backbone of any learning to rank system.

This approach can be applied to entity search by extracting features for query-entity pairs. We use the following ranking algorithms on each of an entity's five fields: `Language model` with Dirichlet smoothing ($\mu = 2500$), `BM25` (default parameters), `coordinate match`, `cosine similarity`, and `sequential dependency model (SDM)`. We also include Zhiltsov et al's. [6] `fielded sequential dependency model (FSDM)` score for the full document as a feature. As a result, there are in total 26 features as listed in Table 2.

With features extracted for all query-entity pairs, all learning to rank models developed for ranking documents can be used to rank entities. We use two widely-used LeToR models: `RankSVM`, which is a SVM-based pairwise method, and `Coordinate Accent`, which is a gradient-based listwise method that directly optimizes mean average precision (MAP). Both of these LeToR algorithms are robust and effective on a variety of datasets.

Table 1: Query sets used in experiments.

Query Set	Queries	Search Task
SemSearch ES	130	Retrieve one entity
ListSearch	115	Retrieve a list of entities
INEX-LD	100	Mixed keyword queries
QALD-2	140	Natural language questions

Table 2: Query-entity features used in learning to rank.

Features	Dimension
FSDM	1
SDM on all fields	5
BM25 on all fields	5
Language model on all fields	5
Coordinate match on all fields	5
Cosine similarity on all fields	5

3. EXPERIMENTAL METHODOLOGY

This section describes our experiment methodology in studying the effectiveness of learning to rank in entity search.

Dataset: Our experiments are conducted on the entity search test collection provided by Balog and Neumayer [1], which others also have used for research on entity retrieval [2, 6]. The dataset has 485 queries with relevance judgments on entities from DBpedia version 3.7. These queries come from seven previous competitions and are merged into four groups based on their search tasks [6]. Table 1 lists the four query groups used in our experiments.

Base Retrieval Model: We use the `fielded sequential dependency model` (`FSDM`) as the base retrieval model to enable direct comparison with prior work [6]. All learning to rank methods are used to rerank the top 100 entities per query retrieved by `FSDM`.

Ranking Models: `RankSVM` implementation is provided by SVMLight toolkit[1]. `Coordinate Ascent` implementation is provided by RankLib[2]. Both methods are trained and tested using five fold cross validation. We use linear kernel in `RankSVM`. For each fold, hyper-parameters are selected by another five fold cross validation on the *training* partitions only. The 'c' of `RankSVM` is selected from the range $1 - 100$ using a step size of 1. The number of random restarts and iterations of `Coordinate Ascent` are selected from the ranges $1 - 10$ and $10 - 50$ respectively using a step size of 1.

Baselines: The main baseline is `FSDM`, the previous state-of-the-art for the benchmark [6]. We also include `SDM-CA` and `MLM-CA` [6] results as they perform well in the test collection.

Evaluation Metrics: All methods are evaluated by MAP@100, P@10 and P@20 following previous work [6]. We also report NDCG@20. Statistical significance tests are performed by Fisher Randomization (permutation) tests with $p < 0.05$.

4. EVALUATION RESULTS

We first present experimental results for learning to rank on entity search. Then we provide analysis of the importance of features and fields, and the influence of different query types on LeToR models.

[1]https://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html

[2]http://sourceforge.net/p/lemur/wiki/RankLib/

4.1 Overall Performance

The ranking performances of learning to rank models are listed in Table 3a. We present results separately for each query group and also combine the query groups together, shown in the `All` section of Table 3a. Relative performances over `FSDM` are shown in parenthesis. $\dagger, \ddagger, \S$ indicate statistical significance over `SDM-CA`, `MLM-CA`, and `FSDM` respectively. The best performing method for each metric is marked **bold**. Win/Tie/Loss are the number of queries improved, unchanged and hurt, also compared with `FSDM`.

The results demonstrate the power of learning to rank for entity search. On all query sets and all evaluation metrics, both learning methods outperform `FSDM`, defining a new state-of-the-art in entity search. The overall improvements on all queries can be as large as 8%. On `SemSearch ES`, `ListSearch` and `INEX-LD`, where the queries are keyword queries like 'Charles Darwin', LeToR methods show significant improvements over `FSDM`. However, on `QALD-2`, whose queries are questions such as 'Who created Wikipedia', simple text similarity features are not as strong.

Similar trends are also found in individual query performances. Figure 1 compares the best learning method, `RankSVM`, with `FSDM` at each query. The x-axis lists all queries, ordered by relative performance. The y-axis is the relative performance of `RankSVM` over `FSDM` on NDCG@20. On keyword queries more than half of the queries are improved while only about a quarter of the queries are hurt. On questions (`QALD-2`), about the same number of queries are improved and hurt. A more effective method of handling natural question queries was developed recently by Lu et al. in which queries are parsed using question-answering techniques [2]. That method achieves 0.25 in P@10, but performs worse than `FSDM` on keyword queries. Section 4.3 further studies the influence of query types on entity-ranking accuracy.

4.2 Field and Feature Study

The second experiment studies the contribution of fields and feature groups to learning to rank models. For each field or feature group, we compare the accuracy of models when used without field or features from that group to those with all features. The change in accuracy indicates the contribution of the corresponding field or feature group. The field and feature studies for `RankSVM` are shown in Figures 2a and 2b respectively. The x-axis is the field or feature group studied. The y-axis is the performance difference between the two conditions (All versus held out). Larger values indicate greater contributions. Figure 2a organizes features by the fields they are extracted from, including `Name`, `Cat`, `Attr`, `RelEn`, and `SimEn`. Figure 2b organizes features into five groups, with one retrieval model per group. `SDM related` contains `FSDM` and `SDM` scores as they are very correlated.

Figure 2a shows that `RankSVM` favors different fields for different query sets. The `Name` field is useful for `ListSearch` and `QALD-2`, but does not contribute much to the other two query sets. `RelEn` provides the most gain to keyword queries, but is not useful at all for the natural language question queries in `QALD-2`. For feature groups we find that `SDM related` features are extremely important and provide the most gains across all query sets. This result is expected because all of the queries are relatively long queries and often contain phrases, which is where SDM is the most useful.

Table 3: Accuracy of learning to rank methods on entity search. Relative improvements over FSDM are shown in parentheses. Win/Tie/Loss show the number of queries improved, unchanged and hurt, comparing with FSDM. †, ‡, § indicate statistical significance over SDM-CA, MLM-CA, and FSDM respectively. The best method for each metric is marked **bold**.

(a) Overall accuracy on each query group. All section combines the evaluation results of the other four sections.

	SemSearch ES								
	MAP@100		**P@10**		**P@20**		**NDCG@20**		**Win/Tie/Loss**
SDM-CA	0.254	(-34.2%)	0.202	(-29.3%)	0.148	(-27.0%)	0.355	(-29.5%)	26/15/89
MLM-CA	0.320†	(-17.1%)	0.250†	(-12.6%)	0.178†	(-12.3%)	0.443†	(-12.0%)	30/32/68
FSDM	0.386†‡	-	0.286†‡	-	0.203†‡	-	0.503†‡	-	-
RankSVM	**0.410**†‡§	(+6.3%)	**0.304**†‡§	(+6.2%)	**0.213**†‡§	(+4.5%)	**0.527**†‡§	(+4.7%)	65/27/38
Coordinate Ascent	0.396†‡	(+2.6%)	0.295†‡	(+3.2%)	0.206†‡	(+1.1%)	0.511†‡	(+1.5%)	48/32/50
	ListSearch								
	MAP@100		**P@10**		**P@20**		**NDCG@20**		**Win/Tie/Loss**
SDM-CA	0.197	(-3.0%)	0.252	(-1.4%)	0.202	(-0.6%)	0.296	(+1.7%)	55/23/37
MLM-CA	0.190	(-6.6%)	0.252	(-1.4%)	0.192	(-5.3%)	0.275	(-5.3%)	39/28/48
FSDM	0.203	-	0.256	-	0.203	-	0.291	-	-
RankSVM	0.224†‡§	(+10.3%)	**0.303**†‡§	(+18.7%)	**0.235**†‡§	(+15.9%)	**0.332**†‡§	(+14.3%)	61/23/31
Coordinate Ascent	**0.225**†‡§	(+10.5%)	0.300†‡§	(+17.3%)	0.229†‡§	(+12.9%)	0.328†‡§	(+12.9%)	62/21/32
	INEX-LD								
	MAP@100		**P@10**		**P@20**		**NDCG@20**		**Win/Tie/Loss**
SDM-CA	0.117‡	(+5.2%)	0.258	(-1.9%)	0.199	(-7.2%)	0.284	(-0.9%)	43/7/50
MLM-CA	0.102	(-8.2%)	0.238	(-9.5%)	0.190	(-11.4%)	0.261	(-8.8%)	34/13/53
FSDM	0.111‡	-	0.263‡	-	0.214†‡	-	0.287‡	-	-
RankSVM	**0.126**‡§	(+12.9%)	**0.282**‡	(+7.2%)	**0.231**†‡§	(+7.7%)	**0.317**†‡§	(+10.6%)	55/9/36
Coordinate Ascent	0.121‡§	(+8.9%)	0.275‡	(+4.6%)	0.224†‡	(+4.4%)	0.306†‡§	(+6.7%)	53/7/40
	QALD-2								
	MAP@100		**P@10**		**P@20**		**NDCG@20**		**Win/Tie/Loss**
SDM-CA	0.184	(-6.0%)	0.106	(-22.0%)	0.090	(-19.2%)	0.244‡	(-6.8%)	36/66/38
MLM-CA	0.152	(-22.4%)	0.103	(-24.6%)	0.084	(-24.3%)	0.206	(-21.3%)	17/78/45
FSDM	0.195‡	-	0.136†‡	-	0.111‡	-	0.262‡	-	-
RankSVM	0.197‡	(+0.8%)	0.136†‡	(0.0%)	0.113†‡	(+1.6%)	0.266‡	(+1.6%)	31/74/35
Coordinate Ascent	**0.208**‡	(+6.6%)	**0.141**†‡	(+3.2%)	**0.115**†‡	(+2.9%)	**0.278**‡	(+5.9%)	40/71/29
	All								
	MAP@100		**P@10**		**P@20**		**NDCG@20**		**Win/Tie/Loss**
SDM-CA	0.192	(-16.9%)	0.198	(-14.3%)	0.155	(-13.7%)	0.294	(-13.1%)	160/111/214
MLM-CA	0.196	(-15.3%)	0.206	(-11.0%)	0.157	(-12.4%)	0.297	(-12.1%)	120/151/214
FSDM	0.231†‡	-	0.231†‡	-	0.179†‡	-	0.339†‡	-	-
RankSVM	**0.246**†‡§	(+6.5%)	**0.251**†‡§	(+8.7%)	**0.193**†‡§	(+7.8%)	**0.362**†‡§	(+7.0%)	212/133/140
Coordinate Ascent	0.245†‡§	(+5.8%)	0.248†‡§	(+7.2%)	0.189†‡§	(+5.4%)	0.358†‡§	(+5.7%)	203/131/151

(b) Accuracy of learning to rank models when queries from different groups are all trained and tested together. Relative accuracy and Win/Tie/Loss are compared with the same model but trained and tested separately on each query group.

	MAP@100		**P@10**		**P@20**		**NDCG@20**		**Win/Tie/Loss**
RankSVM	0.234	(-4.9%)	0.238	(-5.3%)	0.185	(-4.0%)	0.347	(-4.3%)	153/136/196
Coordinate Ascent	0.233	(-4.7%)	0.232	(-6.2%)	0.179	(-5.1%)	0.344	(-3.8%)	148/129/208

4.3 Query Type Differences

Previous experiments found that when different models are trained for different types of queries, each model favors different types of evidence. However, in live query traffic different types of queries are mixed together. The third experiment investigates the accuracy of a learning to rank entities system in a more realistic setting. The four query sets are combined into one query set, and new models are trained and tested using five fold cross validation as before.

Table 3a All shows the average accuracy when different models are trained for each of the four types of query. Table 3b shows the accuracy when a single model is trained for all types of queries. Despite being trained with more data, both learning to rank algorithms produce less effective models for the diverse query set than for the four smaller, focused query sets. Nonetheless, a single learned model is as accurate as the average accuracy of four carefully-tuned, query-set-specific FSDM models.

This results suggests that diverse query streams may benefit from query classification and type-specific entity ranking models. They may also benefit from new types of features or more sophisticated ranking models.

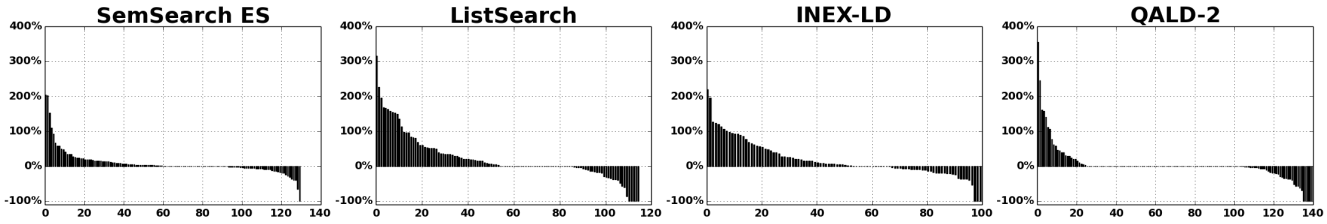
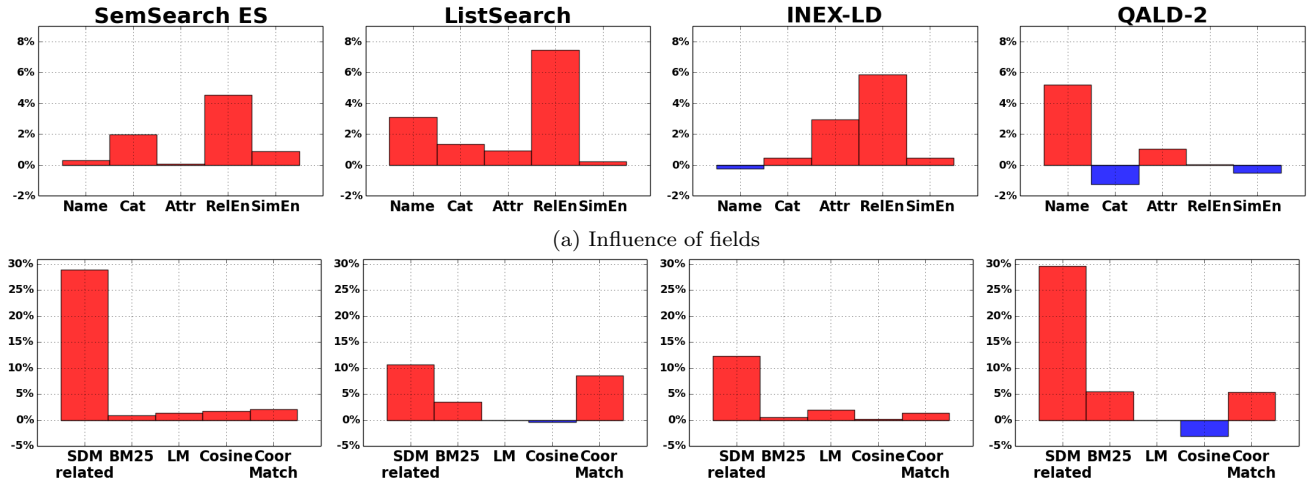


Figure 1: Query-level relative accuracy of four query groups. The X-axis lists all queries, ordered by relative accuracy. The Y-axis is the relative accuracy of `RankSVM` compared with `FSDM` using NDCG@20. A positive value indicates an improvement and a negative value indicates a loss.



(a) Influence of fields

(b) Influence of feature groups

Figure 2: The contribution of fields and feature groups to `RankSVM`'s performance. The X-axis lists the fields or feature groups. The Y-axis is the relative NDCG@20 difference between `RankSVM` used with all fields or feature groups and without the corresponding field or feature group. Larger values indicate greater contribution.

5. CONCLUSIONS AND FUTURE WORK

This paper uses learning to rank models, the state-of-the-art in document ranking, for more accurate ad-hoc entity ranking. Entities are represented by multi-field documents constructed from RDF triples. How well a query matches an entity document is estimated by text similarity features and a learned model. Experiments on an entity-oriented test collection reveal the power of learning to rank for entity retrieval. Moreover, statistically significant improvements over the previous state-of-the-art are observed on all evaluation metrics.

Further analysis reveals that query types play an important role in the effectiveness of learned models. Text similarity features are very helpful for keyword queries, but less effective with longer natural language question queries. Learned models for different query types favor different entity fields because each query type targets different RDF predicates. This difference between query types is a new challenge to the use of entities in diverse search environments, because currently a single learned model does not provide much gain on average. An interesting future research direction is to automatically detect the type of entity search required for a query or task, and then use a model adapted for that type or task.

6. ACKNOWLEDGMENTS

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

7. REFERENCES

[1] K. Balog and R. Neumayer. A test collection for entity search in dbpedia. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval (SIGIR 2013)*, pages 737–740. ACM, 2013.

[2] C. Lu, W. Lam, and Y. Liao. Entity retrieval via entity factoid hierarchy. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL 2015)*, pages 514–523. ACL, 2015.

[3] J. Pound, P. Mika, and H. Zaragoza. Ad-hoc object retrieval in the web of data. In *Proceedings of the 19th international conference on World wide web (WWW 2010)*, pages 771–780. ACM, 2010.

[4] 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.

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

[6] N. Zhiltsov, A. Kotov, and F. Nikolaev. Fielded sequential dependence model for ad-hoc entity retrieval in the web of data. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2015)*, pages 253–262. ACM, 2015.