Language-Model-Based Ranking in Entity-Relation Graphs

Shady Elbassuoni
Max-Planck Institute for Informatics
elbass@mpii.de

Maya Ramanath
Max-Planck Institute for Informatics
ramanath@mpii.de

Gerhard Weikum
Max-Planck Institute for Informatics
weikum@mpii.de

ABSTRACT

We propose a language-model-based ranking approach for SPARQL-like queries on entity-relationship graphs. Our ranking model supports exact matching, approximate structure matching, and approximate matching with text predicates. We show the effectiveness of our model through examples.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Query formulation, Retrieval models*

General Terms

Algorithms, Design, Languages

1. INTRODUCTION

The availability of semi-structured information resources such as Wikipedia and advances in information-extraction techniques have made it possible to build large-scale "knowledge repositories" (e.g. [1, 8]). These repositories typically contain entities such as people, locations, movies, companies, etc. and the relationships between them such as bornIn, locatedIn, isCEOof, and so on. The repository itself is conceptually a large graph with nodes corresponding to entities and the edges between them corresponding to relationships. Typically the facts in the knowledge base can be expressed as RDF triples (subject-property-object) and can be queried using graph-based query languages like the W3C-endorsed SPARQL.

Solely using the expressive but Boolean-match SPARQL language is often too restrictive. For example, when asking for Woody Allen movies, users prefer a ranked result list rather than seeing all matches at once. Consider another query asking for movies that Woody Allen produced as well as acted in. An exact match would yield *no* results for this query. However, an *automatic* relaxation of the query could return movies which Woody Allen created (or directed) and acted in. Additionally, if the user is interested only in movies set in New York, the query could be augmented with keywords "New York" to push movies such as Vicky Cristina Barcelona lower in the rankings.

We propose a ranking method based on statistical language models (LMs) for exact, approximate, and keyword-augmented queries. Our ranking model constructs LMs for both the query and result graphs, and then ranks the results based on the KL-divergence between the query and result-graph LMs.

Related Work. LMs have been applied to passage retrieval, cross-lingual IR, and other advanced tasks that entail ranking. Recently, extended LMs have been developed for entity ranking in the context of expert finding and Wikipedia-based IR tasks [2, 5, 6, 7, 9]. These models are limited to entities – nodes in the graph. In contrast, our work addresses the role of relations – edges in the graph – for answering more expressive classes of entity-relationship queries.

Ranking has been studied for keyword search on databases (e.g., [3]), but the ranking is solely based on graph properties like edge weights, path lengths, and PageRank-style node-authority. In contrast, our ranking model provides a more effective ranking by considering the actual entities and relations in the result graph. The closest work to ours is the LM-based ranking in the NAGA system [4]. However, NAGA supports only exact query matches, while we support approximate and keyword-augmented queries.

2. FRAMEWORK

A knowledge base B of entities and relationships is a graph denoted by a set of *triples*, where a triple is of the form $< e1, r, e2 >$ with entities $e1$, $e2$ and relation r. A triple corresponds to an edge in the graph, e.g. Woody_Allen directed Manhattan. Each triple is associated with a set of textual terms (words or phrases) derived from the documents from which it was extracted. A triple t is associated with a witness count $c(t)$, which indicates the number of times the triple was extracted from the corpus (the corpus could be the Web). The witness count gives a measure of "importance" of the triple in the corpus. In addition, for each term w_i associated with t, a witness count $c(t; w_i)$ is also stored.

A query is a graph of *triple-templates*. A triple template is a triple with at most 2 variables. For example, the query Woody_Allen produced ?x;Woody_Allen actedIn ?x consists of 2 triple-templates. The result of a query is the set of all subgraphs of B that are isomorphic to the query when binding the query variables with matching nodes and edges of the knowledge graph.

A keyword-augmented query allows keywords to be associated with one or more triple-templates in the query. For example, the query Woody_Allen produced ?x*{new york}* could be issued to ask for movies which Woody Allen produced and which were somehow related to New York.

Generating Approximate Queries. From a given query $Q = \{q_1, ..., q_k\}$, where q_i is a triple-template, we generate an approximate query by replacing a constant in a q_i with a variable. We can systematically generate a number of such approximate queries by replacing each constant or set of constants in the query with variables. For the Woody Allen example, we could relax the relation produced to get the query: Woody_Allen ?y ?x*{new york}*.

KEYS'09, June 28, 2009, Providence, Rhode Island, USA.

3. RANKING MODEL

We estimate a query LM P_Q for the generation of query Q and a result-graph LM P_G for the generation of graph G (a subgraph of B). The result graphs are ranked in increasing order of the KL-divergence between the query LM and the result-graph LM.

Query Model. Let $Q = \{q_1, ..., q_n\}$ be a query with n triple-templates where q_i is a triple-template. In order for the query and result-graph LMs to be comparable, they both have to be a probability distribution over triples. To this end, we define the query LM as follows. Let $\hat{q}_i$ be the set of triples which match the triple-template q_i. Let $T = \{t_1, ..., t_n\}$ be an n-tuple where t_i is a triple and $c(t_i)$ is the number of witnesses of triple t_i. The query LM is a probability distribution P_Q over all possible n-tuples of triples. Assuming independence between triples, the probability of T is: $P_Q(T) = \prod_{i=1}^{n} P_Q(t_i)$, where $P_Q(t_i)$ is estimated as:

$$P_Q(t_i) = \begin{cases} \frac{c(t_i)}{\sum_{t \in \hat{q}_i} c(t)} & \text{if } t_i \in \hat{q}_i \\ 0 & \text{otherwise} \end{cases}$$

Approximate matches are obtained by replacing one or more constants in a query $Q_0 = \{q_{01}, ..., q_{0n}\}$ with a variable. Let $Q_1, ..., Q_r$ be the set of approximate queries generated where $Q_i = \{q_{i1}, ..., q_{in}\}$. As in the previous case, let $\hat{q}_{ij}$ be the set of triples which match the triple-template q_{ij}. Now, we estimate $P_Q(T)$ for an n-tuple T as: $P_Q(T) = \lambda_0 P_{Q_0}(T) + \lambda_1 P_{Q_1}(T) + ... + \lambda_r P_{Q_r}(T)$ where $\sum \lambda_i = 1$, and $P_{Q_i}(T)$ is computed as explained previously. λ_0 is set high so that the tuples which instantiate the original query Q_0 have the highest weight. The remaining λ_i's can be equal, or varying based on the degree of relaxation (e.g., number of constants replaced by variables).

Query Model with Keywords. Let $Q = \{\tilde{q}_1, ..., \tilde{q}_n\}$ be a keyword-augmented query where $\tilde{q}_i = q_i[w_1, ..., w_m]$, q_i is a triple-template and w_k is an associated keyword. Let $\hat{q}_i$ be the set of triples which match the triple-template q_i. Let $T = \{t_1, ..., t_n\}$ be an n-tuple where t_i is a triple and $c(t_i; w_k)$ is the number of witnesses of triple t_i containing keyword w_k. As before, the probability of T is: $P_Q(T) = \prod_{i=1}^{n} P_Q(t_i|w_1, ..., w_m)$ where

$$P_Q(t_i|w_1, ..., w_m) = \prod_{k=1}^{m} [\alpha P_Q(t_i|w_k) + (1-\alpha) P(t_i)]$$

and

$$P_Q(t_i|w_k) = \begin{cases} \frac{c(t_i; w_k)}{\sum_{t \in \hat{q}_i} c(t; w_k)} & \text{if } t_i \in \hat{q}_i \\ 0 & \text{otherwise} \end{cases}$$

The second component $P(t_i)$ is a smoothing prior which ensures non-zero probabilities for triples which match q_i, but not all of its associated keywords. In our implementation we used a uniform prior. Finally, the parameter α controls the influence of smoothing. The query model estimation for approximate matches is analogous.

Result-Graph Model. For query Q with n triple-templates, we are interested in potentially relevant results consisting of n triples. Given a result graph $T = \{t_1, t_2, ..., t_n\}$, we estimate its language model P_G. Analogous to the query LM, P_G is a probability distribution over all possible tuples where $P_G(T) = \beta P(T|G) + (1 - \beta) P(T|B)$ and the parameter β controls the influence of smoothing. $P(T|G)$ is 1 since the result graph G contains only the n-tuple T. For the smoothing component, we assume independence between triples: $P(T|B) = \prod_{i=1}^{n} P(t_i|B)$ where $P(t_i|B)$ is estimated given the entire knowledge base and its entirety of witnesses, and is equal to:

$$P(t_i|B) = \frac{c(t_i)}{\sum_{t \in B} c(t)}$$

4. EXAMPLES AND DISCUSSION

Q1	Brad_Pitt actedIn ?x	Brad_Pitt actedIn ?x*{brother}*
1	Se7en	Legends_of_the_Fall
2	Meet_Joe_Black	Across_the_Tracks
3	Troy	Se7en
4	Seven_Years_in_Tibet	Meet_Joe_Black
Q2	**Woody_Allen produced ?x Woody_Allen actedIn ?x**	**Woody_Allen produced ?x*{new york}***
1	Annie_Hall	Anything_Else
2	Vicky_Cristina_Barcelona	Annie_Hall
3	Mighty_Aphrodite	Bullets_Over_Broadway
4	Manhattan_Murder_Mystery	New_York_Stories
Q3	**?x actedIn ?y;?z actedIn ?y;?y hasGenre Comedy**	
1	Charlie_O'Connell, Ashton_Kutcher, Dude_Where's_My_Car?	
2	Sandra_Oh, Paul_Giamatti, Sideways	
3	George_Clooney, Julia_Roberts, Ocean's_Twelve	
4	Buddy_Ebsen, Audrey_Hepburn, Breakfast_at_Tiffany's	

Table 1: Examples of queries and top-ranked results

Our experimental knowledge base, extracted from the movie database IMDB, has over 600,000 triples about people and movies and relations between them (actedIn, directed, etc). Each triple is associated with a set of keywords obtained from the movie descriptions.

Since we extracted triples from only one source, we had to estimate the witness counts for each triple+keyword from the Web. We issued the two entities and the keyword in the triple as a query to a search engine, and used the number of hits returned as an approximation for the triple's witness count.

Table 1 shows examples for exact, approximate, and keyword-augmented queries and the corresponding top-4 ranked results. The first set of queries (Q1) ask for movies in which Brad Pitt acted. The top results correspond to well-known Brad Pitt movies. When we augment the query with the keyword "brother", the top-2 results – "Legends of the Fall" and "Across the Tracks" – have sibling rivalry as their major themes.

In the second set of queries (Q2), the first asks for movies which Woody Allen produced and acted in. There are *no* exact-match results for this query. The *approximate* match results returned by our ranking model correspond to movies which Woody Allen created and acted in. The top-4 results are very relevant. The second query asks for movies which Woody Allen produced, and is augmented with the keywords "New" and "York". The top-4 results now no longer include "Vicky Cristina Barcelona" which was set in Spain.

The third query (Q3) asks for pairs of actors acting in a movie with genre "Comedy". The top-4 results correspond to well-known comedies, and actor pairs.

5. REFERENCES

[1] S. Auer, et. al. Dbpedia: A nucleus for a web of open data. *ISWC/ASWC* 2007

[2] H. Fang, C. Zhai. Probabilistic models for expert finding. *ECIR* 2007

[3] V. Hristidis, H. Hwang, Y. Papakonstantinou. Authority-based keyword search in databases. *TODS*, 33(1), 2008

[4] G. Kasneci, et. al. Naga: Searching and ranking knowledge. *ICDE* 2008

[5] Z. Nie, et. al. Web object retrieval. *WWW* 2007

[6] D. Petkova, W. Croft. Hierarchical language models for expert finding in enterprise corpora. *Int. J. on AI Tools*, 17(1), 2008

[7] P. Serdyukov, D. Hiemstra. Modeling documents as mixtures of persons for expert finding. *ECIR* 2008

[8] F. M. Suchanek, G. Kasneci, G. Weikum. Yago: A core of semantic knowledge. *WWW* 2007

[9] D. Vallet, H. Zaragoza. Inferring the most important types of a query: a semantic approach. *SIGIR* 2008