Learning to Explain Entity Relationships in Knowledge Graphs

Nikos Voskarides[*]
University of Amsterdam
n.voskarides@uva.nl

Edgar Meij
Yahoo Labs, London
emeij@yahoo-inc.com

Manos Tsagkias
904Labs, Amsterdam
manos@904labs.com

Maarten de Rijke
University of Amsterdam
derijke@uva.nl

Wouter Weerkamp
904Labs, Amsterdam
wouter@904labs.com

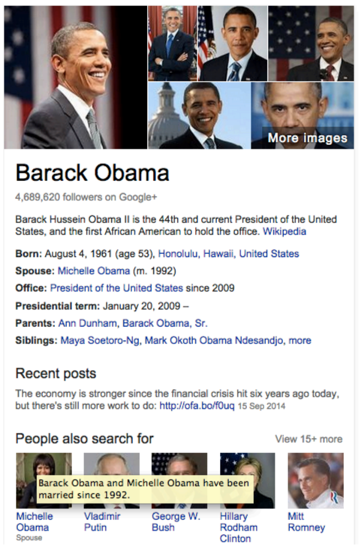
Abstract

We study the problem of explaining relationships between pairs of knowledge graph entities with human-readable descriptions. Our method extracts and enriches sentences that refer to an entity pair from a corpus and ranks the sentences according to how well they describe the relationship between the entities. We model this task as a learning to rank problem for sentences and employ a rich set of features. When evaluated on a large set of manually annotated sentences, we find that our method significantly improves over state-of-the-art baseline models.

1 Introduction

Knowledge graphs are a powerful tool for supporting a large spectrum of search applications including ranking, recommendation, exploratory search, and web search (Dong et al., 2014). A knowledge graph aggregates information around entities across multiple content sources and links these entities together, while at the same time providing entity-specific properties (such as age or employer) and types (such as actor or movie).

Although there is a growing interest in automatically constructing knowledge graphs, e.g., from unstructured web data (Weston et al., 2013; Craven et al., 2000; Fan et al., 2012), the problem of providing evidence on why two entities are related in a knowledge graph remains largely unaddressed. Extracting and presenting evidence for linking two entities, however, is an important aspect of knowledge graphs, as it can enforce trust between the user and a search engine, which in turn can improve long-term user engagement, e.g., in the context of related entity recommendation (Blanco et al., 2013). Although knowledge graphs exist that provide this functionality to a certain degree (e.g., when hovering over Google's suggested entities, see Figure 1), to the best of our knowledge there is no previously published research on methods for entity relationship explanation.



Barack Obama

4,689,620 followers on Google+

Barack Hussein Obama II is the 44th and current President of the United States, and the first African American to hold the office. Wikipedia

Born: August 4, 1961 (age 53), Honolulu, Hawaii, United States

Spouse: Michelle Obama (m. 1992)

Office: President of the United States since 2009

Presidential term: January 20, 2009 –

Parents: Ann Dunham, Barack Obama, Sr.

Siblings: Maya Soetoro-Ng, Mark Okoth Obama Ndesandjo, more

Recent posts

The economy is stronger since the financial crisis hit six years ago today, but there's still more work to do: http://ofa.bo/f0uq 15 Sep 2014

People also search for — View 15+ more

Barack Obama and Michelle Obama have been married since 1992.

Michelle Obama (Spouse) · Vladimir Putin · George W. Bush · Hillary Rodham Clinton · Mitt Romney

Figure 1: Part of Google's search result page for the query "barack obama". When hovering over the related entity "Michelle Obama", an explanation of the relationship between her and "Barack Obama" is shown.

In this paper we propose a method for explaining the relationship between two entities, which we evaluate on a newly constructed annotated dataset that we make publicly available. In particular, we consider the task of explaining relationships between pairs of Wikipedia entities. We aim to infer a human-readable description for an entity pair given a relationship between the two entities. Since Wikipedia does not explicitly define relationships between entities we use a knowledge graph to obtain these relations. We cast our task as a sentence ranking problem: we automatically extract sentences from a corpus and rank

[*]This work was carried out while this author was visiting Yahoo Labs.

them according to how well they describe a given relationship between a pair of entities. For ranking purposes, we extract a rich set of features and use learning to rank to effectively combine them. Our feature set includes both traditional information retrieval and natural language processing features that we augment with entity-dependent features. These features leverage information from the structure of the knowledge graph. On top of this, we use features that capture the presence in a sentence of the relationship of interest. For our evaluation we focus on "people" entities and we use a large, manually annotated dataset of sentences.

The research questions we address are the following. First, we ask what the effectiveness of state-of-the-art sentence retrieval models is for explaining a relationship between two entities (RQ1). Second, we consider whether we can improve over sentence retrieval models by casting the task in a learning to rank framework (RQ2). Third, we examine whether we can further improve performance by using relationship-dependent models instead of a relationship-independent one (RQ3). We complement these research questions with an error and feature analysis.

Our main contributions are a robust and effective method for explaining entity relationships, detailed insights into the performance of our method and features, and a manually annotated dataset.

2 Related Work

We combine ideas from sentence retrieval, learning to rank, and question answering to address the task of explaining relationships between entities.

Previous work that is closest to the task we address in this paper is that of Blanco and Zaragoza (2010) and Fang et al. (2011). First, Blanco and Zaragoza (2010) focus on finding and ranking sentences that explain the relationship between an entity and a query. Our work is different in that we want to explain the relationship between two entities, rather than a query. Fang et al. (2011) explore the generation of a ranked list of knowledge base relationships for an entity pair. Instead, we try to select sentences that describe a particular relationship, assuming that this is given.

Our approach builds on sentence retrieval, where one retrieves sentences rather than documents that answer an information need. Document retrieval models such as tf-idf, BM25, and language modeling (Baeza-Yates et al., 1999) have been extended to tackle sentence retrieval. Three of the most successful sentence retrieval methods are TFISF (Allan et al., 2003), which is a variant of the vector space model with tf-idf weighting, language modeling with local context (Murdock, 2006; Fernández et al., 2011), and a recursive version of TFISF that accounts for local context (Doko et al., 2013). TFISF is very competitive compared to document retrieval models tuned specifically for sentence retrieval (e.g., BM25 and language modeling (Losada, 2008)) and we therefore include it as a baseline.

Sentences that are suitable for explaining relationships can have attributes that are important for ranking but cannot be captured by term-based retrieval models. One way to combine a wide range of ranking features is learning to rank (LTR). Recent years have witnessed a rapid increase in the work on learning to rank, and it has proven to be a very successful method for combining large numbers of ranking features, for web search, but also other information retrieval applications (Burges et al., 2011; Surdeanu et al., 2011; Agarwal et al., 2012). We use learning to rank and represent each sentence with a set of features that aim to capture different dimensions of the sentence.

Question answering (QA) is the task of providing direct and concise answers to questions formed in natural language (Hirschman and Gaizauskas, 2001). QA can be regarded as a similar task to ours, assuming that the combination of entity pair and relationship form the "question" and that the "answer" is the sentence describing the relationship of interest. Even though we do not follow the QA paradigm in this paper, some of the features we use are inspired by QA systems. In addition, we employ learning to rank to combine the devised features, which has recently been successfully applied for QA (Surdeanu et al., 2011; Agarwal et al., 2012).

3 Problem Statement

We address the problem of explaining relationships between pairs of entities in a knowledge graph. We operationalize the problem as a problem of ranking sentences from documents in a corpus that is related to the knowledge graph. More specifically, given two entities e_i and e_j that form an entity pair $\langle e_i, e_j \rangle$, and a relation r between them, the task is to extract a set of candidate sentences $S_{ij} = \{s_{ij_1}, \ldots, s_{ij_k}\}$ that refer to $\langle e_i, e_j \rangle$ and to impose a ranking on the sentences in S_{ij}. The relation r has the general form $\langle type(e_i), terms(r), type(e_j) \rangle$, where $type(e)$ is the type of the entity e (e.g., `Person` or `Actor`) and $terms(r)$ are the terms of the relation (e.g., `CoCastsWith` or `IsSpouseOf`).

We are left with two specific tasks: (1) extracting candidate sentences S_{ij}, and (2) ranking S_{ij}, where the goal is to have sentences that provide a perfect explanation of the relationship at the top position of the ranking. The next section describes our methods for both tasks.

4 Explaining Entity Relationships

We follow a two-step approach for automatically explaining relationships between entity pairs. First, in Section 4.1, we extract and enrich sentences that refer to an entity pair $\langle e_i, e_j \rangle$ from a corpus in order to construct a set of candidate sentences. Second, in Section 4.2, we extract a rich set of features describing the entities' relationship r and use supervised machine learning in order to rank the sentences in S_{ij} according to how well they describe the relationship r.

4.1 Extracting candidate sentences

To create a set of candidate sentences for a given entity pair and relationship, we require a corpus of documents that is pertinent to the entities at hand. Although any kind of document collection can be used, we focus on Wikipedia in this paper, as it provides good coverage for the majority of entities in our knowledge graph.

First, we extract surface forms for the given entities: the title of the entity's Wikipedia article (e.g., "Barack Obama"), the titles of all redirect pages linking to that article (e.g., "Obama"), and all anchor text associated with hyperlinks to the article within Wikipedia (e.g., "president obama"). We then split all Wikipedia articles into sentences and consider a sentence as a candidate if (i) the sentence is part of either entities' Wikipedia article and contains a surface form of, or a link to, the other entity; or (ii) the sentence contains surface forms of, or links to, both entities in the entity pair.

Next, we apply two sentence enrichment steps for (i) making sentences self-contained and readable outside the context of the source document and (ii) linking the sentences to entities. For (i), we replace pronouns in candidate sentences with the title of the entity. We apply a simple heuristic for the people entities, inspired by (Wu and Weld, 2010):[1] we count the frequency of the terms "he" and "she" in the article for determining the gender of the entity, and we replace the first appearance of "he" or "she" in each sentence with the entity's title. We skip this step if any surface form of the entity occurs in the sentence.

For (ii), we apply entity linking to provide links from the sentence to additional entities (Milne and Witten, 2008). This need arises from the fact that not every sentence in an article contains explicit links to the entities it mentions, as Wikipedia guidelines only allow one link to another article in the article's text.[2] The algorithm takes a sentence as input and iterates over n-grams that are not yet linked to an entity. If an n-gram matches a surface form of an entity, we establish a link between the n-gram and the entity. We restrict our search space to entities that are linked from within the source article of the sentence and from within articles to which the source article links. This way, our entity linking method achieves high precision as almost no disambiguation is necessary.

As an example, consider the sentence "He gave critically acclaimed performances in the crime thriller Seven…" on the Wikipedia page for Brad Pitt. After applying our enrichment steps, we obtain "`Brad_Pitt` gave critically acclaimed performances in the crime thriller `Seven`…", making the sentence human readable and link to the entities `Brad_Pitt` and `Seven_(1995_film)`.

4.2 Ranking sentences

After extracting candidate sentences, we rank them by how well they describe the relationship of interest r between entities e_i and e_j. There are many signals beyond simple term statistics that can indicate relevance. Automatically constructing a ranking model using supervised machine learning techniques is therefore an obvious choice. For ranking we use learning to rank (LTR) and represent each sentence with a rich set of features. Table 1 lists the features we use. Below we provide

[1]We experimented with the Stanford co-reference resolution system (Lee et al., 2011) and Apache OpenNLP and found that they were not able to consistently achieve the level of effectiveness that we require.

[2]`http://en.Wikipedia.org/wiki/Wikipedia:Manual_of_Style/Linking`

#	Name	Gloss
Text features		
1	Sentence length	Length of s in words
2	Sum of *idf*	Sum of IDF of terms of s in Wikipedia
3	Average *idf*	Average IDF of terms of s in Wikipedia
4	Sentence density	Lexical density of s, see Equation 1 (Lee et al., 2001)
5–8	POS fractions	Fraction of verbs, nouns, adjectives, others in s (Mintz et al., 2009)
Entity features		
9	#entities	Total number of entities in s
10	Link to e_i	Whether s contains a link to the entity e_i
11	Link to e_j	Whether s contains a link to the entity e_j
12	Links to e_i and e_j	Whether s contains links to both entities e_i and e_j
13	Entity first	Is e_i or e_j the first entity in the sentence?
14	Spread of e_i, e_j	Distance between the last match of e_i and e_j in s (Blanco and Zaragoza, 2010)
15–22	POS fractions left/right	Fraction of verbs, nouns, adjectives, others to the left/right window of e_i and e_j in s (Mintz et al., 2009)
23–25	#entities left/right/between	Number of entities to the left/right or between entities e_i and e_j in s
26	common links e_i, e_j	Whether s contains any common link of e_i and e_j
27	#common links	The number of common links of e_i and e_j in s
28	Score common links e_i, e_j	Sum of the scores of the common links of e_i and e_j in s
29–30	#common links prev/next	The number of common links of e_i and e_j in previous/next sentence of s
Relationship features		
31	Match $\mathit{terms}(r)$?	Whether s contains any term in $\mathit{terms}(r)$
32	Match $\mathit{wordnet}(r)$?	Whether s contains any phrase in $\mathit{wordnet}(r)$
33	Match $\mathit{word2vec}(r)$?	Whether s contains any phrase in $\mathit{word2vec}(r)$
34–36	or's	Boolean OR of feature 31 and one or both of features 32 and 33
37–38	or(31, 32, 33) prev/next	Boolean OR of features 31, 32, 33 for the previous/next sentence of s
39	Average $\mathit{word2vec}(r)$	Average cosine similarity of phrases in $\mathit{word2vec}(r)$ that are matched in s
40	Maximum $\mathit{word2vec}(r)$	Maximum cosine similarity of phrases in $\mathit{word2vec}(r)$ that are matched in s
41	Sum $\mathit{word2vec}(r)$	Sum of cosine similarity of phrases in $\mathit{word2vec}(r)$ that are matched in s
42	Score LC	Lucene score of s with $\mathit{titles}(e_i, e_j)$, $\mathit{terms}(r)$, $\mathit{wordnet}(r)$, $\mathit{word2vec}(r)$ as query
43	Score R-TFISF	R-TFISF score of s with queries constructed as above
Source features		
44	Sentence position	Position of s in document from which it originates
45	From e_i or e_j?	Does s originate from the Wikipedia article of e_i or e_j?
46	#(e_i or e_j)	Number of occurrences of e_i or e_j in document from which s originates, inspired by document smoothing for sentence retrieval (Murdock and Croft, 2005)

Table 1: Features used for sentence ranking.

a brief description of the more complex ones.

Text features This feature type regards the importance of the sentence s at the term level. We compute the density of s (feature 4) as:

$$\mathit{density}(s) = \frac{1}{K \cdot (K+1)} \sum_{j=1}^{n} \frac{\mathit{idf}(t_j) \cdot \mathit{idf}(t_{j+1})}{\mathit{distance}(t_j, t_{j+1})^2}, \quad (1)$$

where K is the number of keyword terms in s and $\mathit{distance}(t_j, t_{j+1})$ is the number of non-keyword terms between keyword terms t_j and t_{j+1}. We treat stop words and numbers in s as non-keywords and the remaining terms as keywords. Features 5–8 capture the distribution of part-of-speech tags in the sentence.

Entity features These features partly build on (Tsagkias et al., 2011; Meij et al., 2012) and describe the entities and are dependent on the knowledge graph. Whether e_i or e_j is the first appearing entity in a sentence might be an indicator of importance (feature 13). The spread of e_i and e_j in the sentence (feature 14) might be an indicator of their centrality in the sentence (Blanco and Zaragoza, 2010). Features 15–22 capture the distribution of part-of-speech tags in the sentence in a window of four words around e_i or e_j in s (Mintz et al., 2009), complemented by the number of entities between, to the left of, and to the right of the entity pair (features 23–25).

We assume that two articles that have many common articles that point to them are strongly related (Witten and Milne, 2008). We hypothesize that, if a sentence contains common inlinks from e_i and e_j, the sentence might contain important information about their relationship. Hence, we add whether the sentence contains a common link (feature 26) and the number of common links (feature 27) as features. We score a common link l between e_i and e_j using:

$$score(l, e_i, e_j) = sim(l, e_i) \cdot sim(l, e_j), \quad (2)$$

where $sim(\cdot, \cdot)$ is defined as the similarity between two Wikipedia articles, computed using a variant of Normalized Google Distance (Witten and Milne, 2008). Feature 28 then measures the sum of the scores of the common links.

Previous research shows that using surrounding sentences is beneficial for sentence retrieval (Doko et al., 2013). We therefore consider the number of common links in the previous and next sentence (features 29–30).

Relationship features Feature 31 indicates whether any of the relationship-specific terms occurs in the sentence. Only matching the terms in the relationship may have low coverage since terms such as “spouse” may have many synonyms and/or highly related terms, e.g., “husband” or “married”. Therefore, we use WordNet to find synonym phrases of r (feature 32); we refer to this method as $wordnet(r)$.

Alternatively, we use word embeddings to find such similar phrases (Mikolov et al., 2013). Such embeddings take a text corpus as input and learn vector representations of words and phrases consisting of real numbers. Given the set V_r consisting of the vector representations of all the relationship terms and the set V which consists of the vector representations of all the candidate phrases in the data, we calculate the distance between a candidate phrase represented by a vector $\mathbf{v}_i \in V$ and the vectors in V_r as:

$$distance(\mathbf{v}_i, V) = \cos\left(\mathbf{v}_i, \sum_{\mathbf{v}_j \in V_r} \mathbf{v}_j\right), \quad (3)$$

where $\sum_{\mathbf{v}_j \in V_r} \mathbf{v}_j$ is the element-wise sum of the vectors in V_r and the distance between two vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ is measured using cosine similarity. The candidate phrases in V are then ranked using Equation 3 and the top-m phrases are selected, resulting in features 33, 39, 40, and 41; we refer to the ranked set of phrases that are selected using this procedure as $word2vec(r)$.

In addition, we employ state-of-the-art retrieval functions and include the scores for queries that are constructed using the entities e_i and e_j, the relation r, $wordnet(r)$, and $word2vec(r)$. We use the titles of the entity articles $titles(e)$ to represent the entities in the query and two ranking functions, Recursive TFISF (R-TFISF) and LC,[3] (features 42–43). TFISF is a sentence retrieval model that determines the level of relevance of a sentence s given a query q as:

$$R(s, q) = \sum_{t \in q} \log(tf_{t,q} + 1) \cdot \log(tf_{t,s} + 1) \cdot \log\left(\frac{n+1}{0.5 + sf_t}\right), \quad (4)$$

where $tf_{t,q}$ and $tf_{t,s}$ are the number of occurrences of term t in the query q and the sentence s respectively, sf_t is the number of sentences in which t appears, and n is the number of sentences in the collection. R-TFISF is an improved extension of the TFISF method (Doko et al., 2013), which incorporates context from neighboring sentences in the ranking function:

$$R_c(s, q) = (1 - \mu) R(s, q) + \mu [R_c(s_{prev}(s), q) + R_c(s_{next}(s), q)], \quad (5)$$

where μ is a free parameter and $s_{prev}(s)$ and $s_{next}(s)$ indicate functions to retrieve the previous and next sentence, respectively. We use a maximum of three recursive calls.

Source features Here, we refer to features that are dependent on the source document of the sentences. We have three such features.

5 Experimental setup

In this section we describe the dataset, manual annotations, learning to rank algorithm, and evaluation metrics that we use to answer our research questions.

5.1 Dataset

We draw entities and their relationships from a proprietary knowledge graph that is created from Wikipedia, Freebase, IMDB, and other sources, and that is used by the Yahoo web search engine. We focus on “people” entities and relationships between them.[4] For our experiments we need to select a manageable set of entities, which we obtain as follows. We consider a year of query logs

[3]In preliminary experiments R-TFISF and LC were the best performing among a pool of sentence retrieval methods.

[4]Note that, except for the co-reference resolution step described in Section 4.1, our method does not depend on this restriction.

from a large commercial search engine, count the number of times a user clicks on a Wikipedia article of an entity in the results page and perform stratified sampling of entities according to this distribution. As we are bounded by limited resources for our manual assessments, we sample 1 476 entity pairs that together with nine unique relationship types form our experimental dataset.

We use an English Wikipedia dump dated July 8, 2013, containing approximately 4M articles, of which 50 638 belong to "people" entities that are also in our knowledge graph. We extract sentences using the approach described in Section 4.1, resulting in 36 823 candidate sentences for our entities. On average we have 24.94 sentences per entity pair (maximum 423 and minimum 0). Because of the large variance, it is not feasible to obtain exhaustive annotations for all sentences. We rank the sentences using R-TFISF and keep the top-10 sentences per entity pair for annotation. This results in a total of 5 689 sentences.

Five human annotators provided relevance judgments, manually judging sentences based on how well they describe the relationship for an entity pair, for which we use a five-level graded relevance scale (perfect, excellent, good, fair, bad).[5] Of all relevance grades 8.1% is perfect, 15.69% excellent, 19.98% good, 8.05% fair, and 48.15% bad. Out of 1 476 entity pairs, 1 093 have at least one sentence annotated as fair. As is common in information retrieval evaluation, we discard entity pairs that have only "bad" sentences. We examine the difficulty of the task for human annotators by measuring inter-annotator agreement on a subset of 105 sentences that are judged by 3 annotators. Fleiss' kappa is k = 0.449, which is considered to be moderate agreement.

5.2 Machine learning

For ranking sentences we use a Random Forest (RF) classifier (Breiman, 2001).[6] We set the number of iterations to 300 and the sampling rate to 0.3. Experiments with varying these two parameters did not show any significant differences. We also tried several feature normalization methods, none of them being able to significantly outperform the runs without feature normalization.

We obtain POS tags using the Stanford part-of-speech tagger and filter out a standard list of 33 English stopwords. For the word embeddings we use *word2vec* and train our model on all text in Wikipedia using negative sampling and the continuous bag of words architecture. We set the size of the phrase vectors to 500 and m = 30.

Baseline	NDCG@1	NDCG@10	ERR@1	ERR@10
B1	0.7508	0.8961	0.3577	0.4531
B2	0.7511	0.8958	0.3584	0.4530
B3	0.7595	0.8997	0.3696	0.4600
B4	0.7767	0.9070	0.3774	0.4672
B5	**0.7801**	**0.9093**	**0.3787**	**0.4682**

Table 2: Results for five baseline variants. See text for their description and significant differences.

5.3 Evaluation metrics

We employ two main evaluation metrics in our experiments, NDCG (Järvelin and Kekäläinen, 2002) and ERR (Chapelle et al., 2009). The former measures the total accumulated gain from the top of the ranking that is discounted at lower ranks and is normalized by the ideal cumulative gain. The latter models user behavior and measures the expected reciprocal rank at which a user will stop her search. We consider these ranking-based graded evaluation metrics at two cut-off points: position 1, corresponding to showing a single sentence to a user, and 10, which accounts for users who might look at more results. We report on NDCG@1, NDCG@10, ERR@1, ERR@10, and Exc@1, which indicates whether we have an "excellent" or "perfect" sentence at the top of the ranking. Likewise, Per@1 indicates whether we have a "perfect" sentence at the top of the ranking (not all entity pairs have an excellent or a perfect sentence).

We perform 5-fold cross validation and test for statistical significance using a paired two-tailed t-test. We depict a significant difference in performance for p < 0.01 with $^\blacktriangle$ (gain) and $^\blacktriangledown$ (loss) and for p < 0.05 with $^\vartriangle$ (gain) and $^\triangledown$ (loss). Boldface indicates the best score for a metric.

6 Results and Analysis

We compare the performance of typical document retrieval models and state-of-the-art sentence retrieval models in order to answer RQ1. We consider five sentence retrieval models: Lucene ranking (LC), language modeling with Dirichlet

[5]`https://github.com/nickvosk/acl2015-dataset-learning-to-explain-entity-relationships`

[6]In preliminary experiments, we contrasted RF with gradient boosted regression trees and LambdaMART and found that RF consistently outperformed other methods.

Has one	# pairs	# sentences	Method	NDCG@1	NDCG@10	ERR@1	ERR@10	Exc@1	Per@1
fair	1 093	4 435	B5	0.7801	0.9093	0.3787	0.4682	–	–
			LTR	**0.8489**▲	**0.9375**▲	**0.4242**▲	**0.4980**▲	–	–
good	1 038	4 285	B5	0.7742	0.9078	0.3958	0.4894	–	–
			LTR	**0.8486**▲	**0.9374**▲	**0.4438**▲	**0.5208**▲	–	–
excellent	752	3 387	B5	0.7455	0.8999	0.4858	0.5981	0.7314	–
			LTR	**0.8372**▲	**0.9340**▲	**0.5500**▲	**0.6391**▲	**0.8298**▲	–
perfect	339	1 687	B5	0.7082	0.8805	0.6639	0.7878	0.7729	0.6136
			LTR	**0.8150**▲	**0.9245**▲	**0.7640**▲	**0.8518**▲	**0.8909**▲	**0.7227**▲

Table 3: Results for the best baseline (B5) and the learning to rank method (LTR).

smoothing (LM), BM25, TFISF, and Recursive TF-ISF (R-TFISF). We follow related work and set μ = 0.1 for R-TFISF, k = 1 and b = 0 for BM25 and μ = 250 for LM (Fernández et al., 2011).

In our experiments, a query q is constructed using various combinations of surface forms of the two entities e_i and e_j and the relationship r. Each entity in the entity pair can be represented by its title, the titles of any redirect pages pointing to the entity's article, the n-grams used as anchors in Wikipedia to link to the article of the entity, or the union of them all. The relationship r can be represented by the terms in the relationship, synonyms in $wordnet(r)$, or by phrases in $word2vec(r)$.

First, we fix the way we represent r. Baseline B1 does not include any representation of r in the query. B2 includes the relationship terms of r, while B3 includes the relationship terms of r and the synonyms in $wordnet(r)$. B4 includes the terms of r and the phrases in $word2vec(r)$, and B5 includes the relationship terms of r, the synonyms in $wordnet(r)$ and the phrases in $word2vec(r)$. Combining these variations with the entity representations, we find that all combinations that use the titles as representation and R-TFISF as the retrieval function outperform all other combinations.[7] This can be explained by the fact that titles are least ambiguous, thus reducing the possibility of accidentally referring to other entities. BM25 and LC perform almost as well as R-TFISF, with only insignificant differences in performance.

Table 2 shows the best performing combination of each baseline, i.e., varying the representation of r and using titles and R-TFISF. B4 and B5 are the best performing baselines, suggesting that $word2vec(r)$ and $wordnet(r)$ are beneficial. B5 significantly outperforms all baselines except B4.

We also experiment with a supervised combination of the baseline rankers using LTR. Here, we consider each baseline ranker as a separate feature and train a ranking model. The trained model is not able to outperform the best individual baseline, however.

6.1 Learning to rank sentences

Next, we provide the results of our method using the features described in Section 4.2, exploring whether our learning to rank (LTR) approach improves over sentence retrieval models (RQ2). We compare an LTR model using Table 1's features against the best baseline (B5). Table 3 shows the results. Each group in the table contains the results for the entity pairs that have at least one candidate sentence of that relevance grade for B5 and LTR.

We find that LTR significantly outperforms B5 by a large margin. The absolute performance difference between LTR and B5 becomes larger for all metrics as we move from "fair" to "perfect," which shows that LTR is more robust than the baseline for entity pairs that have at least one high quality candidate sentence. LTR ranks the best possible sentence at the top of the ranking for ~83% of the cases for entity pairs that contain an "excellent" sentence and for ~72% of the cases for entity pairs that contain a "perfect" sentence.

Note that, as indicated in Section 5.1, we discard entity pairs that have only "bad" sentences in our experiments. For the sake of completeness, we report on the results for all entity pairs in our dataset—including those without any relevant sentences—in Table 4.

6.2 Relationship-dependent models

Relevant sentences may have different properties for different relationship types. For example, a sentence describing two entities being partners would have a different form than one describing that two entities costar in a movie. A similar

[7]We omit a full table of results due to space constraints.

Has one	# pairs	# sentences	Method	NDCG@1	NDCG@10	ERR@1	ERR@10	Exc@1	Per@1
-	1 476	5 689	B5	0.5776	0.6733	0.2804	0.3467	–	–
			LTR	**0.6285**$^{\blacktriangle}$	**0.6940**$^{\blacktriangle}$	**0.3155**$^{\blacktriangle}$	**0.3694**$^{\blacktriangle}$	–	–

Table 4: Results for the best baseline (B5) and the learning to rank method (LTR), using all entity pairs in the dataset, including those without any relevant sentences.

Relationship	# pairs	# sentences	NDCG@1	NDCG@10	ERR@1	ERR@10
$\langle \mathit{MovieActor}, \mathit{CoCastsWith}, \mathit{MovieActor} \rangle$	410	1 403	0.8604	0.9436	0.3809	0.4546
$\langle \mathit{TvActor}, \mathit{CoCastsWith}, \mathit{TvActor} \rangle$	210	626	0.8729	0.9482	0.3271	0.3845
$\langle \mathit{MovieActor}, \mathit{IsDirectedBy}, \mathit{MovieDirector} \rangle$ $\langle \mathit{MovieDirector}, \mathit{Directs}, \mathit{MovieActor} \rangle$	112	492	0.8795	0.9396	0.4709	0.5261
$\langle \mathit{Person}, \mathit{isChildOf}, \mathit{Person} \rangle$ $\langle \mathit{Person}, \mathit{isParentOf}, \mathit{Person} \rangle$	108	716	0.8428	0.9081	0.6395	0.7136
$\langle \mathit{Person}, \mathit{isPartnerOf}, \mathit{Person} \rangle$ $\langle \mathit{Person}, \mathit{isSpouseOf}, \mathit{Person} \rangle$	155	877	0.8623	0.9441	0.6153	0.6939
$\langle \mathit{Athlete}, \mathit{PlaysSameSportTeamAs}, \mathit{Athlete} \rangle$	98	321	0.8787	0.9535	0.3350	0.3996
Average results over all data	1 093	4 435	**0.8661**	**0.9395**	**0.4615**	**0.5287**
LTR (Table 3; fair)			0.8489	0.9375	0.4242	0.4980

Table 5: Results for relationship-dependent models. Similar relationships are grouped together.

idea was investigated in the context of QA for associating question and answer types (Yao et al., 2013). To answer (RQ3) we examine whether learning a relationship-dependent model improves over learning a single model for all types. We split our dataset per relationship type and train a model per type using 5-fold cross-validation within each. Table 5 shows the results.[8] Our method is robust across different relationships in terms of NDCG. However, we observe some variation in ERR as this metric is more sensitive to the distribution of relevant items than NDCG—the distribution over relevance grades varies per relationship type. For example, it is much more likely to find candidate sentences that have a high relevance grade for $\langle \mathit{Person}, \mathit{isSpouseOf}, \mathit{Person} \rangle$ than for $\langle \mathit{Athlete}, \mathit{PlaysSameSportTeamAs}, \mathit{Athlete} \rangle$ in our dataset. We plan to address this issue by exploring other corpora in the future.

The second-to-last row in Table 5 shows the averaged results over the different relationship types, which is a significant improvement over LTR at $p < 0.01$ for all metrics. This method ranks the best possible sentence at the top of the ranking for ~85% of the cases for entity pairs that contain an "excellent" sentence (~2% absolute improvement over LTR) and for ~75% of the cases for entity pairs that contain a "perfect" sentence (~3% absolute improvement over LTR).

[8]We omit Exc@1 and Per@1 due to space constraints.

6.3 Feature type analysis

Next, we analyze the impact of the feature types. Table 6 shows how performance varies when removing one feature type at a time from the full feature set. Relationship type features are the most important, although entity type features are important as well. This indicates that introducing features based on entities identified in the sentences and the relationship is beneficial for this task. Furthermore, the limited dependency on the source feature type indicates that our method might be able to generalize in other domains. Finally, text type features do contribute to retrieval effectiveness, although not significantly. Note that calculating the sentence features is straightforward, as none of our features requires heavy linguistic analysis.

Features	NDCG@1	NDCG@10	ERR@1	ERR@10
All	**0.8661**	**0.9395**	**0.4615**	**0.5287**
All$\setminus$text	0.8620	0.9372	0.4606	0.5274
All$\setminus$source	0.8598	0.9372	0.4582	0.5261
All$\setminus$entity	0.8421$^{\triangledown}$	0.9282$^{\blacktriangledown}$	0.4497	0.5202$^{\triangledown}$
All$\setminus$relation	0.8183$^{\blacktriangledown}$	0.9201$^{\blacktriangledown}$	0.4352$^{\blacktriangledown}$	0.5112$^{\blacktriangledown}$

Table 6: Results using relationship-dependent models, removing individual feature types.

6.4 Error analysis

When looking at errors made by the system, we find that some are due to the fact that entity pairs might have more than one relationship (e.g., actors that costar in movies also being partners) but the selected sentence covers only one of the relationships.[9] For example, `Liza Minnelli` is the daughter of `Judy Garland`, but they have also costarred in a movie, which is the relationship of interest. The model ranks the sentence "Liza Minnelli is the daughter of singer and actress Judy Garland..." at the top, while the most relevant sentence is: "Judy Garland performed at the London Palladium with her then 18-year-old daughter Liza Minnelli in November 1964."

Sentences that contain the relationship in which we are interested, but for which this cannot be directly inferred, are another source of error. Consider, for example, the following sentence, which explains director `Christopher Nolan` directed actor `Christian Bale`: "Jackman starred in the 2006 film The Prestige, directed by Christopher Nolan and costarring Christian Bale, Michael Caine, and Scarlett Johansson". Even though the sentence contains the relationship of interest, it focuses on actor `Hugh Jackman`. The sentence "In 2004, after completing filming for The Machinist, Bale won the coveted role of Batman and his alter ego Bruce Wayne in Christopher Nolan's Batman Begins...", in contrast, refers to the two entities and the relationship of interest directly, resulting in a higher relevance grade. Our method, however, ranks the first sentence on top, as it contains more phrases that refer to the relationship.

7 Conclusions and Future Work

We have presented a method for explaining relationships between knowledge graph entities with human-readable descriptions. We first extract and enrich sentences that refer to an entity pair and then rank the sentences according to how well they describe the relationship. For ranking, we use learning to rank with a diverse set of features. Evaluation on a manually annotated dataset of "people" entities shows that our method significantly outperforms state-of-the-art sentence retrieval models for this task. Experimental results also show that using relationship-dependent models is beneficial.

In future work we aim to evaluate how our method performs on entities and relationships of any type and popularity, including tail entities and miscellaneous relationships. We also want to investigate moving beyond Wikipedia and extract candidate sentences from documents that are not related to the knowledge graph, such as web pages or news articles. Employing such documents also implies an investigation into more advanced coreference resolution methods.

Our analysis showed that sentences may cover different relationships between entities or different aspects of a single relationship—we aim to account for such cases in follow-up work. Furthermore, sentences may contain unnecessary information for explaining the relation of interest between two entities. Especially when we want to show the obtained results to end users, we may need to apply further processing of the sentences to improve their quality and readability. We would like to explore sentence compression techniques to address this. Finally, relationships between entities have an inherit temporal nature and we aim to explore ways to explain entity relationships and their changes over time.

[9]The annotators marked sentences that do not refer to the relationship of interest as "bad" but indicated whether they describe another relationship or not. We plan to account for such cases in future work.

Acknowledgments

This research was partially supported by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement nr 312827 (VOX-Pol), the Netherlands Organisation for Scientific Research (NWO) under project nrs 727.011.005, 612.001.116, HOR-11-10, 640.006.013, 612.066.930, CI-14-25, SH-322-15, Amsterdam Data Science, the Dutch national program COMMIT, the ESF Research Network Program ELIAS, the Elite Network Shifts project funded by the Royal Dutch Academy of Sciences (KNAW), the Netherlands eScience Center under project nr 027.012.105, the Yahoo! Faculty Research and Engagement Program, the Microsoft Research PhD program, and the HPC Fund.

References

Arvind Agarwal, Hema Raghavan, Karthik Subbian, Prem Melville, Richard D. Lawrence, David C. Gondek, and James Fan. 2012. Learning to rank for robust question answering. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 833–842. ACM.

James Allan, Courtney Wade, and Alvaro Bolivar. 2003. Retrieval and novelty detection at the sentence level. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and* *development in informaion retrieval*, pages 314–321. ACM.

Ricardo Baeza-Yates, Berthier Ribeiro-Neto, et al. 1999. *Modern information retrieval*, volume 463. ACM press New York.

Roi Blanco and Hugo Zaragoza. 2010. Finding support sentences for entities. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 339–346. ACM.

Roi Blanco, Berkant Barla Cambazoglu, Peter Mika, and Nicolas Torzec. 2013. Entity recommendations in web search. In *The Semantic Web–ISWC 2013*, pages 33–48. Springer.

Leo Breiman. 2001. Random forests. *Mach. Learn.*, 45(1):5–32.

Christopher J.C. Burges, Krysta Marie Svore, Paul N. Bennett, Andrzej Pastusiak, and Qiang Wu. 2011. Learning to rank using an ensemble of lambda-gradient models. In *Yahoo! Learning to Rank Challenge*, pages 25–35.

Olivier Chapelle, Donald Metzler, Ya Zhang, and Pierre Grinspan. 2009. Expected reciprocal rank for graded relevance. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 621–630. ACM.

Mark Craven, Dan DiPasquo, Dayne Freitag, Andrew McCallum, Tom Mitchell, Kamal Nigam, and Seán Slattery. 2000. Learning to construct knowledge bases from the world wide web. *Artificial Intelligence*, 118(1–2):69–113.

Alen Doko, Maja Štula, and Darko Stipaničev. 2013. A recursive TF-ISF based sentence retrieval method with local context. *International Journal of Machine Learning and Computing*, 3(2):195–200.

Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 601–610. ACM.

James Fan, Raphael Hoffman, Aditya Kalyanpur, Sebastian Riedel, Fabian Suchanek, and Pratim Partha Talukdar, 2012. *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*, chapter Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX). Association for Computational Linguistics.

Lujun Fang, Anish Das Sarma, Cong Yu, and Philip Bohannon. 2011. Rex: explaining relationships between entity pairs. *Proceedings of the VLDB Endowment*, 5(3):241–252.

Ronald T Fernández, David E. Losada, and Leif Azzopardi. 2011. Extending the language modeling framework for sentence retrieval to include local context. *Information Retrieval*, 14(4):355–389.

Lynette Hirschman and Robert Gaizauskas. 2001. Natural language question answering: the view from here. *Natural Language Engineering*, 7(04):275–300.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446.

Gary Geunbae Lee, Jungyun Seo, Seungwoo Lee, Hanmin Jung, Bong-Hyun Cho, Changki Lee, Byung-Kwan Kwak, Jeongwon Cha, Dongseok Kim, JooHui An, et al. 2001. SiteQ: Engineering high performance QA system using lexico-semantic pattern matching and shallow NLP. In *TREC*.

Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2011. Stanford's multi-pass sieve coreference resolution system at the CoNLL-2011 shared task. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 28–34. Association for Computational Linguistics.

David E. Losada. 2008. A study of statistical query expansion strategies for sentence retrieval. In *Proceedings of the SIGIR 2008 Workshop on Focused Retrieval*, pages 37–44.

Edgar Meij, Wouter Weerkamp, and Maarten de Rijke. 2012. Adding semantics to microblog posts. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 563–572. ACM.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

David Milne and Ian H. Witten. 2008. Learning to link with Wikipedia. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 509–518. ACM.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Vanessa Murdock and W. Bruce Croft. 2005. A translation model for sentence retrieval. In *Proceedings of the conference on Human Language Technology* *and Empirical Methods in Natural Language Processing*, pages 684–691. Association for Computational Linguistics.

Vanessa Graham Murdock. 2006. *Aspects of Sentence Retrieval*. Ph.D. thesis, University of Massachusetts Amherst.

Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. 2011. Learning to rank answers to non-factoid questions from web collections. *Computational Linguistics*, 37(2):351–383.

Manos Tsagkias, Maarten de Rijke, and Wouter Weerkamp. 2011. Linking online news and social media. In *WSDM 2011: Fourth ACM International Conference on Web Search and Data Mining*. ACM, February.

Jason Weston, Antoine Bordes, Oksana Yakhnenko, and Nicolas Usunier. 2013. Connecting language and knowledge bases with embedding models for relation extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013*.

Ian Witten and David Milne. 2008. An effective, low-cost measure of semantic relatedness obtained from wikipedia links. In *Proceeding of AAAI Workshop on Wikipedia and Artificial Intelligence: an Evolving Synergy, AAAI Press, Chicago, USA*, pages 25–30.

Fei Wu and Daniel S Weld. 2010. Open information extraction using wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 118–127. Association for Computational Linguistics.

Xuchen Yao, Benjamin Van Durme, and Peter Clark. 2013. Automatic coupling of answer extraction and information retrieval. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 159–165. Association for Computational Linguistics.