Relevance Scores for Triples from Type-Like Relations

Hannah Bast, Björn Buchhold, Elmar Haussmann
Department of Computer Science
University of Freiburg
79110 Freiburg, Germany
{bast, buchhold, haussmann}@cs.uni-freiburg.de

ABSTRACT

We compute and evaluate relevance scores for knowledge-base triples from type-like relations. Such a score measures the degree to which an entity "belongs" to a type. For example, Quentin Tarantino has various professions, including Film Director, Screenwriter, and Actor. The first two would get a high score in our setting, because those are his main professions. The third would get a low score, because he mostly had cameo appearances in his own movies. Such scores are essential in the ranking for entity queries, e.g. "american actors" or "quentin tarantino professions". These scores are different from scores for "correctness" or "accuracy" (all three professions above are correct and accurate).

We propose a variety of algorithms to compute these scores. For our evaluation we designed a new benchmark, which includes a ground truth based on about 14K human judgments obtained via crowdsourcing. Inter-judge agreement is slightly over 90%. Existing approaches from the literature give results far from the optimum. Our best algorithms achieve an agreement of about 80% with the ground truth.

1. INTRODUCTION

Knowledge bases allow queries with a well-defined result set. For example, we can easily formulate a precise query that gives us a list of all *american actors* in a knowledge base. Note the fundamental difference to full-text search, where keyword queries are only approximations of the actual search intent, and thus result lists are typically a mix of relevant and irrelevant hits.

But even for well-defined result sets, a ranking of the results is often desirable. One reason is similar as in full-text search: when the set of relevant hits is very large, we want the most "interesting" results first. But even if the result set is small, an ordering often makes sense. We give two examples. The numbers refer to Freebase [7], the largest general-purpose knowledge base to date.

Example 1 (american actors): Consider the query that returns all entities that have *Actor* as their profession and *American* as their nationality. On our Freebase dataset, this query has 64,757 matches. A straightforward ranking would be by popularity, as measured, e.g., by counting the number of occurrences of each entity in a reference text corpus. Doing that, the top-5 results for our query look as follows (the first item means the younger Bush):

George Bush,*Hillary Clinton*,*Tim Burton*,*Lady Gaga*,*Johnny Depp*

All five of these are indeed listed as actors in Freebase. This is correct in the sense that each of them appeared in a number of movies, and be it only in documentary movies as themselves or in short cameo roles. However, Bush and Clinton are known as politicians, Burton is known as a film director, and Lady Gaga as a musician. Only Johnny Depp, number five in the list above, is primarily an actor. He should definitely be ranked before the other four.

Example 2 (professions of a single person): Consider all professions by Arnold Schwarzenegger. Our version of Freebase lists 10 entries:

Actor, *Athlete*, *Bodybuilder*, *Businessperson*, *Entrepreneur*, *Film Producer*, *Investor*, *Politician*, *Television Director*, *Writer*

Again, all of them are correct in a sense. For this query, ranking by "popularity" (of the professions) makes even less sense than for the query from Example 1. Rather, we would like to have the "main" professions of that particular person at the top. For Arnold Schwarzenegger that would be: *Actor*, *Politician*, *Bodybuilder*. Note how we have a very-ill defined task here. It is debatable whether Arnold Schwarzenegger is more of an actor or more of a politician. But he is certainly more of an actor than a writer.

1.1 Problem definition

In this paper, we address the following problem, which addresses the issues behind both of the examples above.

Definition: Given a type-like relation from a knowledge base, for each triple from that relation compute a score from $[0, 1]$ that measures the degree to which the subject belongs to the respective type (expressed by the predicate and object). In the remainder of this paper we often refer to these scores simply as *triple scores*.

Here are four example scores, related to the queries above:

Tim Burton has-profession Actor	0.3
Tim Burton has-profession Director	1.0
Johnny Depp has-profession Actor	1.0
Arnold Schwarzenegger has-profession Actor	0.6

An alternative, more intuitive way of expressing this notion of "degree" is: how "surprised" would we be to see *Actor* in a list of professions of Johnny Depp. We use this formulation in our crowdsourcing task when acquiring human judgments.

Note that the "degree" in the definition above is inherently ill-defined, much like "relevance" in full-text search. In particular, different users might have different opinions on the correct "degree" (for example, on the four scores from above). However, it is one of the results of our crowdsourcing-based experiments that there is

SIGIR'15, August 09 - 13, 2015, Santiago, Chile.

DOI: http://dx.doi.org/10.1145/2766462.2767734.

broad general consensus. Note that we do not take user preferences into account in this paper; we consider this an interesting topic on its own.

1.2 Variants and related problems

The reader may wonder about variants of the problem definition above, and related problems from the literature. We briefly discuss these here.

Degree of Belonging vs. Correctness vs. Accuracy

Our scores are different from correctness or confidence scores. For example, in knowledge fusion, multiple sources might provide conflicting information on the birth date of a person. In those situations we need assessments on the confidence of a particular information and which information is probably correct [12].

Our scores are also different from scores that measure the accuracy. For example, an estimate for the population of a country might be off by a certain margin but not completely wrong, or only a range for the value is given.

We consider a single knowledge base with non-conflicting and precise information. Indeed, for a knowledge base like Freebase, over 99% of all triples are reported to be correct [7]. Even if there are incorrect triples, it is a pleasant side effect of our framework that they would get low scores.

Graded scores vs. Binary scores

We started our research on this topic with binary scores, which we called *primary* and *secondary*, in accordance with existing research on graded relevance in full-text search.[1] We later found that both our crowdsourcing experiments and our algorithms naturally provide finer grades that also make sense in practice. We hence use a continuous score in our definition above. The range $[0, 1]$ is merely a matter of convention.

In our crowdsourcing tasks in Section 3 we still ask each judge for only a binary judgment. However, aggregation of multiple judgments for each triple leads again to a graded score.

Which kind of relations

In principle, our algorithms work for all relations. However, we found that for relations other than type-like, the kind of scores we are interested in are either trivial or can be obtained using other, simpler methods. We distinguish three types of relations.

- Functional relations like *birth date* or *weight*, where each subject has exactly one object. Since we assume a single knowledge base where (almost) all triples are correct, we could simply assign a score of 1.0 to all such triples.

- Non-functional relations between two concrete entities, like *acquired* (between two companies) or *acted in* (between an actor and a movie). For such triples there are two simpler options for a good relevance score, which we do not explore further in this paper. The first option is to take the value from another relation of the knowledge base, like the date of the acquisition or the rank of the actor in the cast listing. This relation could be assigned manually (the number of relations in a knowledge base is relatively small). Or it could be determined heuristically (interesting, but technically completely different from the methods we present). The second option is to adapt our method for type-like relations algorithms also for these relations. The main challenge for finding witnesses in the full text is then to recognize variations of the (fixed) predicate name and not of the (varying) object name. This makes our problem much easier; see Section 4.

[1]We assume that all triples in our knowledge base are correct, so there are no triples expressing a relationship that is *not relevant*.

- Non-functional relations between an entity and an abstract group or type, like *profession* (between a person and a profession) or *nationality* (between a person and a nationality). We know of no knowledge base with explicit values for the "degree" to which an entity belongs to a certain type. In particular, this is true for *profession* and *nationality*. As we will see in Section 4, such relations are also the hardest for our approaches, because they require a separate classifier for each distinct type (e.g., each distinct profession and nationality), instead of just one for the whole relation. This is why we selected only type-like relations for our benchmark.

We further remark that type information is central to many entity queries. For example, all three tasks from the TREC 2011 Entity Track [2] ask for lists of entities of a particular type. Also, most of the entity queries currently supported by Google are for entities of a certain type, like in the two use cases from our introduction; presumably because of their popularity amongst users.

Triple Scores vs. Full Ranking

The reader may ask why we study scores for individual triples when the motivation is the ranking of results for entity queries. We give two main reasons.

First, the triple scores we consider are a crucial component of any such ranking, and the problem by itself is hard enough to warrant separate study. In fact, in Section 2 we discuss related works on ranking for entity queries that require such scores as input.

Second, for many popular entity queries (like our two use cases from the introduction), our triple scores are basically all that is needed to obtain a good ranking. For the first use case ("american actors"), ranking is a simple matter of combining our triple scores with a popularity measure for each entity. Popularity measures are easily obtained, e.g., by counting the number of mentions in a reference corpus. For the second use case ("professions of a single person"), triple scores are exactly what is needed. Note that for the second use case, a proper ranking is still missing on Google.

1.3 Basic Idea and Challenges

Our basic idea is to find "witnesses" for each triple in a given large text corpus. The more witnesses we find and the more significant they are, the larger the score for that triple.

For example, consider the *profession* relation. In a "learning" step, we compute words that are associated with each profession. For example, for the profession *Actor* these could be: actor, actress, film, cast. For each entity (e.g., *Johnny Depp*) we identify occurrences of these words semantically related to that entity in the text (e.g., *Paramount decided to cast Depp ...*). Then we compute weighted sums of the number of such occurrences for each possible profession. In a final step, we normalize these sums to obtain our scores. This sounds simple, but there are a number of challenges:

- Learning must be unsupervised, given that we have to learn different words for every different possible "type". For example, there are more than 3,552 different professions in Freebase.

- When is a word "semantically related" to an entity in a text? It turns out that considering co-occurrence in the same sentence works, but that a deeper natural language processing helps.

- How much weight to give to each occurrence of a word? It turns out that results can be boosted by graded scores for different words.

- How to get from weighted sums to normalized scores? It turns out that a simple linear mapping works but is often not optimal.

- Why use text and not the knowledge base? For a long time, we also experimented with other facts from the knowledge base as complementary or even as the sole information source. For example, the fact that someone is or once was president of the U.S.

implies a high score for the profession Politician. However, we found this knowledge to be consistently less available (especially for less popular entities) and nowhere more reliable than full text. More details about this from our experiments in Section 5.4.

1.4 Contributions and Overview

We consider the following as our main contributions:

- We identify the computation of relevance scores for triples from type-like relations as an interesting research problem, which so far has not achieved much attention. These scores are essential for properly ranking many popular kinds of entity queries.
- We designed and make available a first benchmark for this problem. The required large number of relevance judgments (over 14K) were obtained via crowdsourcing. Given the hard-to-define nature of our scores, designing the task such that untrained human judges provide proper and consistent judgments was a challenging task on its own. Inter-judge agreement is about 90%, which confirms the soundness of our problem definition. See Section 3.
- We introduce a variety of methods (partly new, partly adapted) to compute these scores. All our methods work with an arbitrary knowledge base and text corpus. See Section 4.
- We implemented and evaluated all methods against three baselines, using Freebase as knowledge base and Wikipedia as text corpus. Our best methods achieve an agreement of about 80% with the ground truth. We considered many variants and ideas for improvement, none of which gave better results. See Section 5.
- We make our evaluation benchmark as well as all our code publicly available (see Section 5). In particular, this allows full reproduction of our results.

2. RELATED WORK

There exists a large amount of work about ranking and other problems based on the kind of relevance scores we study here. However, in all these works such scores are assumed to be given. We know of no work that addresses how to compute these scores in the first place. We give a short overview of the various approaches.

Triple Scores for Ranking Structured Queries Several pieces of work deal with integrating scores in a framework for ranking structured queries. For example, in [8], the authors propose an extension to RDF they call Ranked RDF, [13] proposes a ranking model for SPARQL queries with possible text extensions based on Language Models, and [11] discusses how to combine several kinds of scores associated with triples into a meaningful ranking. In all these frameworks, scores that are similar to our triple scores are assumed to be given.

Fuzzy Databases / Knowledge bases This is an old and well-established research area, considering all kinds of "fuzziness" in an information system, including: uncertainty, imprecision, and fractional degrees of membership in sets; see [20] for a survey. However, this body of work is almost entirely about modeling this fuzziness, and how to solve standard tasks (like queries or reasoning) based on fuzzy information. The fuzzy information itself is, again, assumed to be given. We know of no work in this area, where fuzzy scores of the kind we consider are actually computed.

Ranking for Relational Databases SQL allows explicit ordering of the query results (using the *order by* keyword). Still, there are many meaningful queries to databases that return a huge number of results. When confronting users with those results, ranking plays an important role. This problem is tackled in [9]. Apart from the many-answers problem, ranking for databases becomes important when adding a keyword-search component. BANKS [5] has the goal of ranking possible interpretations of a keyword query. Similarly, ObjectRank [1] also takes keyword queries, but ranks "database objects" (e.g., a paper, an author, etc.) according to a query. These approaches share the fact that they work exclusively on the data (or knowledge) base. Going from there, they try to use the structure induced by foreign keys to provide a ranking. For our specialized use case, this is not well suited. Even in a perfect world, where such an approach is able to find the perfect connections to influence the ranking (a hard task in itself), it is restricted to data in the knowledge base. However, as discussed in Section 1.3, we have found structured knowledge to be less available than that from full text for computing our relevance scores.

Ranking Semantic Web Resources In the Semantic Web, one is confronted with data from many differences sources, and of highly varying quality. This gives rise to the problem of ranking these data sources with respect to a given query or topic. For example, TripleRank [16] achieves this by representing the Semantic Web as a 3D tensor and then performs a tensor decomposition. Despite the related-sounding name, this is very different from our scenario, which is about ranking of entities from a single knowledge base.

Topic Models On a technical level, some of our methods are related to *topic models*. These derive high-level topics of documents by inferring important words for each topic and the topic distribution for each document. In our setting, a document could be seen as a subject (e.g., person) and topics as different types (e.g., her professions). One state-of-the-art unsupervised topic model is Latent Dirichlet Allocation (LDA) [6]. LDA (and related methods) infer topics given only the text as input. In a supervised extension called Labeled LDA (L-LDA) [21], topic labels (e.g., our professions) can be provided for each document as part of the input. We compare our methods against L-LDA in Section 5.

3. ESTABLISHING A BENCHMARK

Ranking problems and the notion of relevance are inherently subjective and vague. Human relevance judgments are not only necessary to evaluate algorithms that attempt to solve the problem, but also help understanding it. This has happened for keyword search, where ranking became an established problem with a universally shared understanding.

We decided to use crowdsourcing to create a suitable benchmark for our triple scores for exactly these two reasons: evaluation and problem refinement. Since this helps to understand the problem at hand, we discuss our benchmark before we describe our algorithms to solve the problem.

Recall our use case example from Section 1. We want humans to judge to what extent a person has a certain profession. In this section, we discuss how we select a representative sample of entities, present two ways to set up the task, and explain why one is superior. We then analyze the result of the crowdsourcing experiment. The ground truth files are available for download together with all reproducibility material under the URL given in Section 5.

3.1 Selecting Persons for Evaluation

Our benchmark should feature a representative sample of persons that contains all levels of popularity. This is important for two reasons. First, it can be expected that more information is available for popular persons, so that some approaches might work better (or even only) for those. Second, as discussed in the motivation, while the ranking problem we address also exists for less popular persons, it is more pronounced for popular persons.

Selecting persons from Freebase with more than one profession leaves us with about 240K persons. We use the number of times

a person is mentioned in Wikipedia to approximate popularity and restrict our choice to persons having Wikipedia articles. This has practical reasons. In principle, any text collection could be used, but Wikipedia is easy to obtain and, due to hyperlinks, no hard Entity Linking problem has to be solved. Apart from that, we can point human judges directly to the Wikipedia article and hence enable them to make an informed decision without much effort. We observe that (as could be expected) there are lots of people with none or very few mentions and few people with lots of mentions (up to almost 100K mentions). Therefore, a random sampling would almost exclusively contain unpopular entities.

We define buckets of popularity and take a uniform random sample from each bucket. The buckets used were for a popularity (number of mentions) $< 2^4$, between 2^i and 2^{i+1} for $4 \leq i \leq 13$, and $\geq 2^{14}$. In total, we took about 25 samples from each of the 12 buckets. This left us with 298 persons or a total of 1225 person-profession tuples that nicely cover different levels of popularity.

3.2 Evaluate a Single Profession of a Person

An obvious approach is to present a person and his or her profession to a judge and ask whether the profession is primary or secondary for that person. For example:

Arnold Schwarzenegger and the profession *Bodybuilder*. Is the profession primary or secondary for this person?

The task was enriched with instructions, including definitions of the factors listed above. However, this definition is extremely hard to communicate precisely. Also, there is a lot of subjectivity involved: what "feels" primary to one judge does not to another.

Eventually, it turned out that judges mostly resorted to the following strategy: label the profession that is first mentioned in Wikipedia as primary, and all others as secondary. Indeed this is one of our simple baselines in Section 5. Obviously, this cannot work for persons with more than one primary profession.

3.3 Evaluate All Professions of a Person

An improved version that turned out to work well is the following. Instead of labeling a single profession of a person, we ask to label all professions of a person. Additionally, we provide simpler instructions but enrich them with a set of diverse examples of labeled persons. An example is depicted in Figure 1. All professions of *Barack Obama* must be dragged into the box for primary or secondary professions.

Figure 1: Condensed illustration of the crowdsourcing task. All professions must be dragged into the box for primary or secondary professions. For the complete version including instructions and examples see, go the URL from Section 5.

Below that, we show four examples illustrating a diverse set of possible labelings: Michael Jackson (many professions, some primary some secondary), Ronald Reagan (many professions, two of them primary), Mike Tyson (three professions, only one primary), James Cook (two professions, both primary).

This worked well, and we assume this is mainly due to two reasons:

(1) People could do research on the person, and thus judge all the professions of that person in relation to each other.

(2) The diverse examples helped to form an intuitive theory of what we expected.

3.4 Crowdsourcing Results

We had the professions of each person labeled by seven different judges. As final score for each triple we sum up the number of primary labels, which is then in the range of 0 (all judges label secondary) to 7 (all judges label primary).

On the 298 persons (1225 person-profession tuples) the inter-annotator agreement according to Fleiss' Kappa [15] is 0.47 for the binary judgments. This can be considered moderate agreement. However, what our experiment actually does is to determine a hidden probability p with which a human judge finds a person's profession to be primary. In fact, we use a binary experiment to get gradual scores. Note that, for certain triples, the desired result may actually be a probability around 0.5. Measuring inter-annotator agreement does such a scenario no justice. Instead, we want to make sure that (1) all judges base their decision on the task and the data, and (2) the obtained probabilities are significantly different from random decisions.

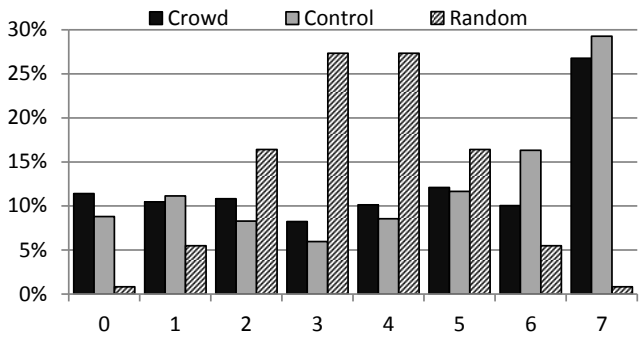

Figure 2: Histogram of score distribution of our crowdsourcing task, the control run, and expected results for randomly (with $p = 0.5$) guessing judges (rounded).

For (1), we have run the same task again (with different annotators) on a subset of every third person (386 person-profession tuples) from our initial sample. Figure 2 depicts how much the scores differ between this control run (*control*) and the full run (*crowd*).

For (2), looking at the distribution of scores shows that they are very far from a random decision. Figure 2 shows a distribution that prefers border cases (especially primary professions) and contains only reasonably few triples with average score (3 or 4). In contrast, a random or unclear task would lead to a distribution with mostly average scores and rare border cases.

In Section 5, we report more details on the results of the control run. The human judges based 95% of their decisions only on the Wikipedia page of the respective person. As we report in Section 5.4, our automatic methods perform much worse when we restrict them in this way. This shows how hard the judgment task is: the relative relevance of the various professions mentioned on a person's Wikipedia page is often highly implicit in the text.

4. COMPUTING TRIPLE SCORES

We propose a variety of algorithms to compute triple scores. For illustration purposes we describe the algorithms using the *profession* relation of a person, but the algorithms are not specific to that relation and should work with any type-like relation. In Section 5 we show that they are equally effective to derive scores for nationalities of persons. Likewise, we use Wikipedia as our source of text and Freebase as knowledge base, but the approaches are not specific to that data. None of our algorithms requires manually labeled data. Instead, we make use of existing facts in an unsupervised or distantly supervised fashion. The crowdsourcing judgments are only used for evaluation.

The different algorithms provide different output: binary classifications, weighted sums and probabilities. In Section 4.6, we describe how we map their output to the range $[0, 1]$.

4.1 Training Data

We want to avoid manually labeled training data. Instead, we use the following definition to obtain positive and negative training examples for each profession:

Positive: people with only the given profession or any specialization of it, i.e. those for which the profession has to be primary.
Negative: people that do not have that profession at all

Thus, Humphrey Bogart is a positive example for the profession *Actor* (because this is his only profession according to Freebase) and Barack Obama, who is listed as politician, lawyer and more (but not as actor), is a negative example for that profession.

We also experimented with other criteria (e.g., allowing the persons used as *positive* examples to also have other professions) but found this selection to work best across all approaches.

4.2 Selecting Text for each Person

All approaches analyze text about a person to derive triple scores. As text we use the English Wikipedia[2], utilizing the fact that persons in Freebase often have a link to their Wikipedia article. We have pre-processed Wikipedia and performed Entity Recognition and Linking, as well as anaphora resolution as described in [3]. With each person we associate all words that co-occur with a linked mention of the person in the same *semantic context*, as provided by the pre-processing. Each semantic context is a sub-sequence of the containing sentence that expresses one fact from the sentence. This yields slight improvements over using full sentences (where numerous facts can be mixed) as contexts. Further, we have found that stemming has almost no effect. These and other variations are discussed in more detail in Section 5.4.

4.3 Binary Classifier per Profession

We train a binary classifier to decide whether a profession is primary or secondary for a given person. For each person we extract features from her associated text. We use word counts as features, which we normalize by the total number of word occurrences in the person's associated text. Thus, feature values are between 0 and 1 (much closer to 0). Weighting feature values by their tf-idf score (instead of the normalization) had no positive effect.

Before training the classifier, we balance positive and negative training examples by randomly selecting an appropriate subset of the negative training examples (there are always more negative than positive training examples).

A logistic regression classifier with L2-regularization is then trained on the balanced training data. As a linear model, logistic regression has the benefit of an intuitive interpretation of features weights. Table 1 lists the features with top weights learned for profession *Actor*.

[2]downloaded in August 2013

1. cast	28.21	5. directed	-6.64	9. university	-5.16
2. actor	12.46	6. starring	6.22	10. written	-4.97
3. actress	11.50	7. role	5.62	11. voiced	4.50
4. played	7.19	8. stars	5.26	12. actors	4.48

Table 1: Top features (positive and negative) and their weight learned by the logistic regression classifier for the profession *Actor*.

The learned feature weights form a model for each profession. These models can then be used for persons with multiple professions to make a binary decision if each of his or her professions is primary or not (secondary).

4.4 Count Profession Words

We want to mimic the behavior of humans that want to solve our problem. Therefore, we look at the text associated with an entity and find out how big the portion of text is that discusses profession matters. In the simplest way, we could count how often a profession is mentioned in text associated with the person.

We cannot count exact occurrences, however, because of professions that consist of more than one word, e.g., Film Score Composer. Looking for mentions of such professions already puts them at a disadvantage (or advantage depending on full vs. partial match counting) compared to one-word professions like Composer. Additionally, often slight variations of a profession are mentioned, e.g., actor and actress.

To overcome this issue, we define a suitable prefix for each profession. We can do this automatically (using an off-the-shelf stemmer and the longest common prefix of the stem and the original word) or manually once for each profession. In our experiments (Section 5), we compare manually chosen prefixes as a strong baseline.

Besides the profession names themselves (or their prefixes), we have found that there are many other words that indicate, that some part of the text is about a given profession. For example, consider the positive features learned by the binary classifier as presented in Table 1. We extend our counting based approach by automatically computing indicator words in the following way: For each profession, take all words in the associated text for persons in the *positive* training data (see Section 4.1), compute their tf-idf value and rank the words by that value. During prediction, we set the weight of an indicator mention to 1/rank and build the sum over the weights of all indicator mentions.

1. cast	1.00	5. role	0.20	9. television	0.11
2. actor	0.50	6. starring	0.16	10. appeared	0.10
3. actress	0.30	7. played	0.14	11. born	0.09
4. film	0.25	8. best	0.13	12. series	0.08

Table 2: Top words (by tf-idf) and their weight $(1/\text{rank})$ for the profession *Actor*.

Table 2 shows the top words and their weights for the profession *Actor*. There is high overlap with the top features learned by the binary classifier shown in Table 1. Note, that the word weights were computed only from the text associated with *positive* training examples and idf values based on the whole corpus. The *negative* examples were not needed.

4.5 Generative model

We formulate a generative model where each person is associated with text that can be generated as follows. For a person with k professions and n word occurrences in the associated text: Pick one of the k professions with probability $P(p_i)$; generate word w_j with $P(w_j|p_i)$; repeat until all n words are generated. The joint probability for word w and profession p is then $P(w,p) = P(w|p)P(p)$.

Note that for a given text, the professions selected in the first step above are unobserved. We would like to infer those, because, intuitively, they represent the amount of text that can be associated with a certain profession. We derive these using a maximum likelihood estimate.

Let tf_j be the term frequency of word j, $P(p_i)$ be the probability of profession i and let $P(w_j|p_i)$ be the probability for word j under profession i. The log-likelihood of producing text consisting of n words for k professions is:

$$\log \mathcal{L} = \sum_{j=1}^{N} [tf_j \cdot \log \sum_{i=1}^{k} (P(w_j|p_i)P(p_i))]$$

Training We use the *positive* training examples to derive word probabilities $P(w|p)$. We distinguish two ways. (1) use the term frequency of terms in the text associated with all training examples and assign probabilities accordingly. (2) use the tf-idf values as term frequencies to avoid common, unspecific terms with high probabilities. When using tf-idf values, we use them for both: $P(w|p)$ and as new tf_j when calculating LP. For efficiency reasons, we only keep the probabilities for the top 30K words. We have found that tf-idf values work better and restrict our examples to this setup for the remainder of the paper.

Text that is associated with a person often contains information that is not related to professions, e.g., family and personal life. Therefore, we add a pseudo profession to each person in order to account for these words. We use all text in the English Wikipedia to calculate $P(w|p)$ for the pseudo profession.

1. cast	0.023	5. role	0.007	9. appeared	0.004
2. actor	0.009	6. starring	0.005	10. television	0.004
3. actress	0.008	7. played	0.005	11. born	0.004
4. film	0.008	8. best	0.004	12. roles	0.004

Table 3: Top word probabilities for the profession *Actor*.

Naturally, the top word probabilities depicted in Table 3 are in line with the top word weights presented in Table 2. This is not surprising, since both values are based on the tf-idf scores of words in text associated with the *positive* training samples. The probabilities, however, do not differ as much as the weights we have assigned for the counting approach.

Prediction To derive profession probabilities $\vec{p}$, which have to sum to 1, we maximize the log-likelihood:

$$\vec{p} = \arg\max_{\vec{p}} \sum_{j=1}^{N} [tf_j \cdot \log \sum_{i=1}^{k} (P(w_j|p_i)P(p_i))], \text{s.t.} \sum_{i=1}^{k} p_i = 1$$

To find the maximum likelihood estimate we use Expectation Maximization [10]. Similar to the generative model for pLSI [17] we treat the topic (profession) assignments to word-occurrences as hidden variables. EM iteratively computes posterior probabilities given the hidden variables in the expectation step (E) and then maximizes parameters for the previously computed probabilities in the maximization step (M). The E and M steps are identical to the steps in [17], with the difference that we treat $P(w|p)$ as fixed, because we already computed those from the *positive* examples as described above.

4.6 Mapping to Triple Scores

The above approaches yield a variety of results. Binary classifications, weighted sums and probabilities. However, we actually want to compute scores for triples. While this is trivial for binary classifications (assign minimum and maximum), we distinguish two approaches for sums and probabilities. Keep in mind that we assume there is at least one primary profession for each person. For comparison with the crowdsourcing judgments, we want scores from 0 to 7 but the methods apply for any desired magnitude and, without rounding, naturally allow continuous scores as well.

Maplin Linearly scale computed values to the range 0 to 7. In practice, just divide all sums or probabilities by the highest one. Then multiply by 7 and round to the nearest integer.

Maplog Scale computed values to the range 0 to 7 such that the next highest score corresponds to twice the sum or probability. In practice, divide all sums or probabilities by the highest one. Then multiply by 2^7, take the logarithm to base 2, and finally take the integer portion of the result.

We have found that the intuitive *Maplin* works much better with the sums of weights obtained by counting triples. *Maplog*, however, is stronger when mapping probabilities to triples scores. This is true for both, our generative model, and the topic model, Labeled LDA, we compare against. The probabilistic models tend to assign high probabilities to the top professions, leaving small probabilities to all others. Differences between debatable professions and definite secondary are small in total probability difference value but still observable when comparing magnitudes.

5. EVALUATION

We first discuss the experimental setup: data + ground truth, algorithms compared and quality measures. In Section 5.2, we present and discuss our main results. In Section 5.4, we discuss many variants (algorithms and quality measures) that we also experimented with but which did not turn out to work well.

All of the data and code needed to reproduce our experiments are available under `http://ad.informatik.uni-freiburg.de/publications`.

5.1 Experimental Setup

Data We extracted all triples from the Freebase relations *Profession* and *Country of nationality*. In all experiments we only consider persons with at least two different profession or nationalities, i.e., we only consider the non-trivial cases. Files with all these triples are available under the above URL.

Ground truth For the *profession* relation, we randomly selected a total of 1225 triples pertaining to 298 different people entities, as described in Section 3.1. Each of these triples was then labeled independently by 7 human judges, using crowdsourcing, as described in Section 3.1. This gives a total of 8.575 labels. We repeated the task for all 386 triples of a random subset of 98 from the 298 people. This gives us another 2.702 labels. We used these as control labels to assess the quality of our main set above; see Section 5.2 below. We have presented the distribution of the scores in the ground truth in Section 3.4 in Figure 2. For the *nationaliy* relation, we randomly selected a total of 162 triples pertaining to 77 different people entities and ran the same experiment. This gives us another 1134 judgments.

Algorithms Compared We compare the following: three baselines, six algorithms, and the output of two control experiments.

We normalized all approaches to yield an integer score from the range 0..7 for each triple.

First For each person, take the entity's description from Freebase[3] Look for the first literal mention of one of that person's profession. That profession gets a score of 7, all other professions get a score of 0. This may look overly simplistic, but actually this is what most of the judges did in the previous version of our task as presented in Section 3.2.

Random Make 7 independent judgments for each triple, where for each judgment primary and secondary are equally likely. That is, pick the score at random from a binomial distribution with $n = 7$ and $p = 0.5$. This simulates human judges that guess randomly.

Prefixes For each triple, count the number of occurrences of a hand-picked prefix of the profession (same for all professions) in the Wikipedia article of the person. Map these counts to scores (per person) using *Maplin*, as described in Section 4.6.

Labeled LDA (LLDA) The closest approach from previous work, as described in Section 2. We use a topic label for each profession and label each person with all of her professions. The Dirichlet prior, α, is set to 2.0 and we run Gibbs sampling for 100 iterations. Other parameter settings for α gave much worse results and more iterations showed no improvement. We map the probabilities to scores using *Maplog*.

Words Classification For each triple, make a binary decision (score 0 or 7) using the logistic-regression classifier (see Section 4.3).

Words Counting For each profession, learn a weighted list of words, as described in Section 4.4. For each triple, add the weights of all profession words in all contexts containing the entity. Map these weight sums to scores using *Maplin*.

Words MLE For each triple, we derive a score using the generative model described in Section 4.5 and the *Maplog* mapping.

Counting Combined For each triple, use the *Words Counting* method. Additionally look at the decision of the *Words Classification* method. If the binary classification is positive (score 7), increase the score to the average of the two. The intuition behind this is, that for strongly related professions (e.g., Poet and Playwright), it is hard to attribute a text passage to one of the two. The binary decisions made by the *Words Classification* method do not have this problem.

MLE Combined Similar to *Counting Combined* but starting with the *Words MLE* method instead of the counting.

Control Judges Take the labels from the human control judges, as described above.

Control Expected The expected values when comparing two scorings, where in each scoring each of the seven judgments for a triple is primary with probability p, where p is chosen from the posterior distribution given score s, where s is the score given by the crowdsourcing judges.[4]

Evaluation measures We use two kinds of measures: score-based and rank-based.

Score-based The score-based measures directly compare the scores of two sequences s and s' of triple scores for the same sequence of n triples. We consider two measures: *accuracy* and *average score deviation*. The accuracy has an integer parameter δ and measures the percentage of triple scores that deviate by at most δ. That is, Acc-$\delta = |\{i : |s_i - s'_i| \leq \delta\}|$. The average score deviation is simply the average over all absolute score differences. That is, $\text{ASD} = \sum_i |s_i - s'_i|/n$. We will see that the relative performance of the various methods shows in both measures, but that the Acc figures are more intuitive and insightful.

Rank-based The rank-based measures compare two rankings of the same sequence of triples. We will use them to compare two rankings of all professions / nationalities of a single person, and the average over all persons. We consider three standard ranking measures: Kendall's Tau, the footrule distance, and nDCG.

Because scores in the gold standard are discrete, items often share a rank. Such a ranking with ties constitutes a *partial ranking* [14]. To compare partial rankings we use adapted versions of Kendall's τ from [14]: $\tau_p = \frac{1}{Z}(n_d + p \cdot n_t)$, where n_d is the number of *discordant* (inverted) pairs, n_t is the number of pairs that are tied in the gold standard but not in the predicted ranking or vice versa, p is a penalization factor for these pairs which we set to 0.5, and the normalization factor Z (the number of ordered pairs plus p times the number of tied pairs in the gold standard).

The Spearman footrule distance counts the displacements of all elements. It is $f = \frac{1}{Z}\sum |\sigma_p(i) - \sigma_g(i)|$, where i ranges over the items of the list, and $\sigma_p(i)$ and $\sigma_g(i)$ is the rank of item i in the predicted partial ranking and gold standard partial ranking, respectively. The normalization factor Z corresponds to the maximum possible distance and causes f to be in the interval $[0, 1]$.

For nDCG, we remark that it also takes the exact scores into account: ranking a triple lower than it should be is punished more, the higher the score of that triple is.

5.2 Main results

Score-based evaluation of the *profession* triples We first discuss the results for our score-based measures for the *profession* relation, shown in Table 4.

Method	Accuracy (Acc)			Average Score Diff
	$\delta = 1$	$\delta = 2$	$\delta = 4$	
First	41%	**53%**	71%	2.71
Random	31%	**55%**	91%	2.39
Words Classification	47%	**61%**	78%	2.09
Prefixes	50%	**64%**	83%	2.07
LLDA	50%	**68%**	89%	1.86
Words Counting	57%	**75%**	94%	1.61
Words MLE	57%	**77%**	95%	1.61
Count Combined	58%	**77%**	95%	1.52
MLE Combined	63%	**80%**	96%	1.57
Control Expected	75%	**91%**	99%	0.93
Control Judges	76%	**94%**	99%	0.92

Table 4: Accuracies and ASD for the *profession* relation, best methods last.

The last line (Control Judges) shows that the human judges rarely disagree by more than 2 on the 0-7 scale. The Acc-4 measure shows that there are almost no stark disagreements, that is, by more than 4.[5] A disagreement up to 1 is not unusual though. The average score deviation is 0.92. As the next to last line (Control Expected) shows, this is essentially what would be expected from random

[3]For most entities, this is just the abstract of the corresponding Wikipedia page.

[4]That is, $Pr(p = k/7) \propto \Pr(X = k)$, for $k = 0..7$, where X is from the binomial distribution $B(7, s/7)$.

[5]Note that such disagreements can only happen for scores 0-2 and 5-7.

fluctuation. Acc-2 hence seems to be the single most intuitive measure (for integer scores on a scale 0-7). We therefore emphasized the results for that measure in Table 4.

Our binary classification algorithm achieves an Acc-2 of 61%, which gradually improves to 80% for our most sophisticated algorithm. There is hardly a difference between the MLE approach and word counting with proper normalization. Our best approach makes glaring mistakes (Acc-4) for only 4% of all triples. For the Acc-2 measure, our best approach is still more than 10% away from the ideal. In Section 5.4, we discuss various options for improvement, none of which actually gives a significant further improvement though. We conclude that our problem is what could be called "NLP-hard". Under that condition, we consider an Acc-2 of 80% a very good result.

The simple "First" baseline performs similarly bad as the "Random" baseline, which simulates a random guess for each judge. Note that the "Random" baseline makes glaring mistakes (Acc-4) only for 9% of the triples. This is because the scores of this baseline follow a binomial distribution, with a probability of 0.55 that the score is 3 or 4. For these two scores, the deviation from the "true" value cannot be larger than 4. For the more extreme scores, the probability of being off by more than 4 is also low. Acc-2, however, is only about 55% for "Random". Note that a more extreme version of "Random", which picks only one of the scores 3 or 4 at random for each triple, would achieve an Acc-4 of 100%, but an Acc-2 of only 52%.

The relative performance of the various approaches is reflected by the average score difference (ASD), shown in the last column of Table 4. The various Acc measures provide a more refined picture though.

Rank-based results for the *profession* triples We next discuss the results for our rank-based measures for the *profession* relation, shown in Table 5.

Method	**Kendall**	Footrule	nDCG
Random	**0.51**	0.58	0.80
First	**0.40**	0.47	0.92
LLDA	**0.32**	0.38	0.88
Words Classification	**0.32**	0.35	0.88
Prefixes	**0.31**	0.35	0.88
Words MLE	**0.23**	0.28	0.94
Words Counting	**0.24**	0.29	0.94
Counting Combined	**0.24**	0.28	0.94
MLE Combined	**0.22**	0.27	0.94
Control Judges	**0.18**	0.21	0.97

Table 5: Average rank differences for the *profession* relation, best methods last.

We observe that the relative quality of the various baselines and algorithms is about the same in all three measures. Since Kendall is the simplest and perhaps most intuitive measure, we have highlighted the results for that measure in Table 5.

Footrule is always slightly larger than Kendall. The nDCG values are relative large already for the simple Random baseline. This is an artifact of the short lists we are comparing here (for each person, his or her professions, which are often only two or three). Indeed, for a list with only two items, there are only two possible values for nDCG: 1 and $1/\log_2 3 \approx 0.63$. The average of the two is ≈ 0.82.

The relative order of the methods is similar as for the score-based evaluation of Table 4. The only notable exception is that in the rank-based evaluation, the simple "First" baseline is significantly better than "Random", while in the score-based evaluation, it was the other way round. The reason is simple: "First" almost always gets the first item in the list right (the first profession listed in Wikipedia, is almost always the most obvious profession of the person in question). For persons with only two professions, this already gives the perfect ranking. For persons with three professions, this already gets 2 of the 3 possibly transpositions right. In the score-based evaluation, the score for the triple with this "first" profession is just one out of k, when the person has k professions. Apart from that, "First" (and "Words Classification") make binary decisions for one of the most extreme scores. This is punished by the Accuracy measures. Random, on the other hand, tends to select the middle ground scores 3 and 4 which are not punished as strongly.

Note that the compared rankings are invariant under score mapping (*Maplin* or *Maplog*, see Section 4.6). Hence, score mapping affects neither Kendall nor Footrule. It does, in principle, affect nDCG, since that measure also takes the actual scores into account. We found the effect to be insignificant though (at most 0.02 difference for affected approaches).

Score- and ranked-based evaluation of the *nationality* triples We repeated the experiments above for the triples from the *nationality* relation. We focused on the baselines and the best-performing methods.

Method	Accuracy (Acc)			Average Score Diff
	$\delta = 1$	$\delta = 2$	$\delta = 4$	
First	28%	**36%**	51%	3.89
Random	31%	**55%**	91%	2.83
Prefixes	41%	**52%**	71%	2.21
Words MLE	58%	**78%**	95%	1.71
Words Counting	63%	**78%**	96%	1.34
Control Expected	76%	**92%**	99%	0.82

Method	**Kendall**	Footrule	nDCG
Random	**0.51**	0.58	0.80
First	**0.41**	0.42	0.90
Prefixes	**0.51**	0.53	0.88
Words MLE	**0.44**	0.45	0.92
Words Counting	**0.36**	0.37	0.94

Table 6: Score-based and ranked-based results for the *nationality* relation, best methods last.

Table 6 shows the results for our experiments with triples from the *nationality* relation. Sophisticated approaches have performance similar to the run on triples of the *profession* relation (see Tables 4 and 5). This is good news, because it was not guaranteed that Wikipedia texts provided a sufficiently strong signal for our algorithms to perform well. Interestingly, baselines "First" and "Prefixes" are weaker on this task. Apparently, direct mentions of a nationality in the Wikipedia article are not as useful as they are for professions. A typical example could be the term "German-American" to describe (primary) Americans with German roots. Not only is the secondary nationality mentioned first, the article may also discuss the person's ancestry further, frequently mentioning that country. Statistical signals for the primary nationality, like

mentions of the place of residence or typical institutions are only considered by our more advanced approaches.

5.3 Refined analysis

We also conducted an analysis of the "clear cases", that is, the triples, where the score from the crowdsourcing judges were either 6 or 7 or 1 or 0 (all judges, except at most one, agree). We analyzed the percentage of triples, which got the "reverse" scores for these triples, that is 0-1 for 6-7 and 6-7 for 0-1. The figures were very similar to those already captured by the Acc-4 figure in Table 4. Recall that a score deviation of more than 4 can only happen for the extreme scores. Thus, no new insights were provided by this analysis.

We also analyzed the dependency between accuracy and popularity of a person.

Method	Popularity Bucket					
	1	2	3	4	5	6
Words Counting	**58%**	76%	71%	71%	76%	81%
Count Combined	**63%**	81%	72%	68%	76%	81%
Words MLE	**68%**	84%	76%	79%	84%	76%
MLE Combined	**74%**	84%	75%	79%	84%	79%

Method	Popularity Bucket					
	7	8	9	10	11	12
Words Counting	77%	75%	70%	72%	82%	83%
Count Combined	86%	78%	76%	76%	78%	78%
Words MLE	76%	74%	69%	77%	79%	82%
MLE Combined	82%	75%	76%	80%	80%	82%

Table 7: Accuracy-2 for the *profession* relation, breakdown by person popularity. Popularity Bucket i contains persons with a number of occurrences between 2^{i+2} and 2^{i+3}. Buckets 1 and 12 are special cases and contain persons with less than 2^4, resp. more than 2^{14}, occurrences.

The buckets of popularity used in Table 7 pertain to the buckets from our selection for the crowdsourcing task (see Section 3.1). The breakdown shows that approaches function well on all persons with more than 16 occurrences in the text. This corresponds to buckets 2 and above. While more text is always better for our statistical models, we observe that the minimum required to work reasonably well, is already very low. The persons in bucket 1 with very little associated text are harder to get correct. We can observe two things: *Words Counting* has more problems than *Words MLE*. Further, the combination with the classifier specifically helps to even out the problems with entities in this bucket.

5.4 Variants

Our main results show a gap of 11-14% between our best method, and what human judges can achieve. We have experimented with numerous variations of the methods described in Section 4 and 5.1. None of these variations turned out to give an improvement.

Stemming. All approaches from Section 4 can be used with and without stemming. We tried both the Porter [22] and the Lancaster stemmer [19]. For all methods, stemming (in either variant) did more harm than good.

Word pairs instead of just words. One significant source of errors in the methods from Section 4 is that *single* words are ambiguous indicators for certain triples. For example, the word *film* is a strong indicator for both *Actor* and *Film Director*. Some persons have both professions. To distinguish between the two, the context of the word *film* is important. One obvious approach to address this is to consider pairs of words as features, instead of just single words. For the *profession* relation, the influence on both the score-based and the rank-based measure was insignificant though. For the *nationality* relation, results became significantly worse when using word pairs. This is understandable, since for the majority of nationalities single words are perfectly suited to uniquely identify them (e.g., *Canadian*, *Italian*, *German*).

TF versus TF-IDF In our evaluation, we used tf-idf for all our features. We also experimented with tf features, as well as with variations of tf-idf that give more emphasis to the idf part or that dampened the tf factor like in BM25. Results were either unaffected or became slightly worse.

Non-linear score mappings. In Section 3, we discussed two score mappings: Maplin and Maplog. The latter uses the mapping $x \mapsto \log_2 x$. The rationale was that the MLE approach produces raw scores (the probabilities) that grow super-linearly with the actual score. We experimented with a variety of other non-linear score mappings to capture this behavior, in particular, $x \mapsto x^\alpha$, for various value of α. The results were similar but never significantly better than for our Maplog.

Sentences vs. contexts In our methods in Section 4, we consider only those words that occur "in the same context" as the entity in question. We experimented with two realizations of this context: co-occurrence in the same *sentence* and co-occurrence in the same *context*, as described in Section 4. Contexts consistently outperformed sentences by a few percent. The improvement was more pronounced for popular entities, where we have many occurrences of the indicator words. This is in accordance with the theory from [4]. Co-occurrence in the same sentences does not necessarily mean that the two entities / words that co-occur have something to do with each other. When they co-occur in the same context, they have, by the definition of context from [4].

Whole corpus vs. only the Wikipedia article Almost all human judges indicated that they only used the Wikipedia article of a person to make their decision (95% of all decisions). For our methods from Section 4, we considered co-occurrences anywhere in the text corpus. When restricting to only the Wikipedia page of the respective person, like the human judges did, results consistently became much worse. This is easy to understand, since all our methods are statistics-based, and the whole corpus simply provides more information, and hence also more certainty to make a good decision.

Treating hierarchy / specialization In Freebase, a person's professions may include generalizations of another profession that is listed. For example, for John Lennon there are (among other professions) singer-songwriter, keyboard player, guitarist and also musician. Like for Hierarchical Classification [18] problems, two obvious approaches are to (1) ignore the hierarchy and classify each triple and to (2) only classify specializations and infer a score for parent professions. We have experimented with both approaches and found that there is only little difference. If the same inference rules from (2) are used on both, our scores and the human judgments (human judgments are corrected towards a higher score if the human judges have given that higher score to a specialization of the profession), this increases the final scores by a few percent (e.g., 82% for *MLE Combined*). Otherwise, there is not much difference.

Knowledge-base facts An obvious alternative to using text for computing triple scores is to use information from the knowledge base. However, we have made the experience that 1) only few properties

are reflected by other facts in the knowledge base and 2) even in cases where they are, they are typically only available for popular instances. For example, an indication for the relevance of an actor may be in how many movies he acted, or how many awards he won for them. But there are no such facts in the knowledge base for a geologist, a surgeon, or a firefighter. Furthermore, in Freebase, out of 612K persons with more than one profession, 400K have less than 10 and 243K less than 6 other facts (besides type and profession information).

5.5 Manual Error Analysis

For all our methods from Section 4, as well as for all the variants discussed in Section 5.4, we conducted a manual error analysis. We discovered two main sources of errors.

Wrong reference An indicator word sometimes co-occurs with the person in question, but is actually relating to another person. For example, Michael Jackson is listed as a Film Director, which is certainly not one of his main professions. The Wikipedia article about him contains many mentions of the word *directed* or *director* though. But most of them relate not to him but to a person directing one of his shows.

Wrong meaning An indicator word is sometimes used with another meaning. For example. John F. Kennedy is listed as a Military Officer, which is certainly not one of his main professions. The Wikipedia article contains many mentions of variants of the word *military* though. But most of them relate to political actions during his presidency with respect to the military.

Some of the variants discussed in Section 5.4 mitigate the effect of these problems somewhat. For example, restricting co-occurrence to semantic contexts instead of full sentences helps the wrong-reference problem in several instances. Or, considering word pairs instead of single words helps with the wrong-meaning problem in several instances. However, a significant number of instances remain, where deep natural language understanding appears to be required. Consider the sentences "<Screenwriter X> was very happy with how his script was visualized" vs "<Film Director Y> was praised a lot for how he visualized the script". It is hard to understand the roles of the persons in these sentences. But it is necessary to correctly decide if the sentence is evidence for a profession *Screenwriter* or *Film Producer*.

6. CONCLUSION

We have studied the problem of computing relevance scores for knowledge-base triples from type-like relations. We have shown that we can compute good such scores in an unsupervised manner from a text corpus with an accuracy of about 80%.

We have described various attempts to further improve the accuracy of our triple scores, towards the over 90% accuracy achieved by human judges. These attempts, together with a manual error analysis, suggest that deep natural language understanding is required to close this gap. In particular, we identified two main error sources: *wrong reference* and *wrong meaning* of indicator words. We consider these a hard but worthwhile task for future work.

The focus of this work is on scores for individual triples. This problem is practically relevant on its own and also hard enough on its own. Still, it would be interesting to follow up on previous work about integrating such triple scores into a meaningful ranking for complex knowledge-base queries.

7. REFERENCES

[1] A. Balmin, V. Hristidis, and Y. Papakonstantinou. ObjectRank: Authority-based keyword search in databases. In *VLDB*, pages 564–575, 2004.

[2] K. Balog, P. Serdyukov, and A. P. de Vries. Overview of the TREC 2011 Entity Track. In *TREC*, 2011.

[3] H. Bast, F. Bäurle, B. Buchhold, and E. Haussmann. Broccoli: Semantic full-text search at your fingertips. *CoRR*, abs/1207.2615, 2012.

[4] H. Bast and E. Haussmann. Open information extraction via contextual sentence decomposition. In *ICSC*, pages 154–159, 2013.

[5] G. Bhalotia, A. Hulgeri, C. Nakhe, S. Chakrabarti, and S. Sudarshan. Keyword searching and browsing in databases using BANKS. In *ICDE*, pages 431–440, 2002.

[6] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. In *NIPS*, pages 601–608, 2001.

[7] K. D. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, pages 1247–1250, 2008.

[8] J. P. Cedeño and K. S. Candan. R^2DF framework for ranked path queries over weighted RDF graphs. In *WIMS*, page 40, 2011.

[9] S. Chaudhuri, G. Das, V. Hristidis, and G. Weikum. Probabilistic ranking of database query results. In *VLDB*, pages 888–899, 2004.

[10] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. R. Stat. Soc.*, pages 1–38, 1977.

[11] R. Q. Dividino, G. Gröner, S. Scheglmann, and M. Thimm. Ranking RDF with provenance via preference aggregation. In *EKAW*, pages 154–163, 2012.

[12] X. L. Dong, E. Gabrilovich, G. Heitz, W. Horn, K. Murphy, S. Sun, and W. Zhang. From data fusion to knowledge fusion. *PVLDB*, 7(10):881–892, 2014.

[13] S. Elbassuoni, M. Ramanath, R. Schenkel, M. Sydow, and G. Weikum. Language-model-based ranking for queries on RDF-graphs. In *CIKM*, pages 977–986, 2009.

[14] R. Fagin, R. Kumar, M. Mahdian, D. Sivakumar, and E. Vee. Comparing and aggregating rankings with ties. In *PODS*, pages 47–58, 2004.

[15] J. L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378, 1971.

[16] T. Franz, A. Schultz, S. Sizov, and S. Staab. TripleRank: Ranking semantic web data by tensor decomposition. In *ISWC*, pages 213–228, 2009.

[17] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR*, pages 50–57, 1999.

[18] C. N. S. Jr. and A. A. Freitas. A survey of hierarchical classification across different application domains. *Data Min. Knowl. Discov.*, 22(1-2):31–72, 2011.

[19] C. D. Paice. Another stemmer. *SIGIR Forum*, 24(3):56–61, 1990.

[20] S. Parsons. Current approaches to handling imperfect information in data and knowledge bases. *IEEE Trans. Knowl. Data Eng.*, 8(3):353–372, 1996.

[21] D. Ramage, D. L. W. Hall, R. Nallapati, and C. D. Manning. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In *EMNLP*, pages 248–256, 2009.

[22] C. J. Van Rijsbergen, S. E. Robertson, and M. F. Porter. *New models in probabilistic information retrieval*. Computer Laboratory, University of Cambridge, 1980.