Doctoral Advisor or Medical Condition: Towards Entity-specific Rankings of Knowledge Base Properties

Simon Razniewski[12], Vevake Baralaman[3], Werner Nutt[1]

[1]Free University of Bozen-Bolzano
[2]Max-Planck-Institute for Informatics [2]University of Trento
{razniewski,nutt}@inf.unibz.it, vvek.9291@gmail.com

Abstract. In knowledge bases such as Wikidata, it is possible to assert a large set of properties for entities, ranging from generic ones such as name and place of birth to highly profession-specific or background-specific ones such as doctoral advisor or medical condition. Determining a preference or ranking in this large set is a challenge in tasks such as prioritisation of edits or natural-language generation. Most previous approaches to ranking knowledge base properties are purely data-driven, that is, as we show, mistake frequency for interestingness. In this work, we have developed a human-annotated dataset of 350 preference judgments among pairs of knowledge base properties for fixed entities. From this set, we isolate a subset of pairs for which humans show a high level of agreement (87.5% on average). We show, however, that baseline and state-of-the-art techniques achieve only 61.3% precision in predicting human preferences for this subset. We then develop a technique based on a combination of general frequency, applicability to similar entities and semantic similarity that achieves 74% precision. The preference dataset is available at https://www.kaggle.com/srazniewski/wikidatapropertyranking .

1 Introduction

General-purpose knowledge bases such as Wikidata [23], YAGO [22] or DBpedia [3] are becoming increasingly popular, and are used for a variety of tasks such as structured search, entity recognition or question answering. These knowledge bases can store a large number of entity types, and for each entity type a large number of properties. For instance, for the class of *human* alone, more than 100 properties are used in Wikidata at least 1000 times, among which are the following:

1: sex or gender	70: doctoral advisor
2: occupation	71: pseudonym
3: date of birth	72: medical condition
...	...
39: height	78: convicted of
40: instrument	79: singles record
...	...

An issue with these properties is that it is not known how interesting they are for specific entities. For instance, while 90% of the data in Wikidata is created by bots [23], it is not clear whether the entered data captures actually what is of interest to humans. As a consequence, the large number of properties and their unclear interestingness severely hinder the usability of knowledge bases, and make many data analytics tasks difficult.

A way to better structure these properties would be rankings by interestingness. Such rankings would be useful for at least three tasks:

1. *Recommendations to authors:* Rankings by interestingness could help human authors in focusing their work [1,18,16]. For Wikidata, there exists a tool called *Wikidata property suggester*[1] for that purpose. However, the current instance is association-rule-based, which, as we show below, does not well approximate the human perception of interestingness.
2. *Automatically generating descriptions:* One of the major motivations for the Wikidata project is to automatically generate article stubs, which is especially relevant for low-resource languages. As a 2015 report of the Wikimedia Foundation found, "[the] lack of any clear identifier for importance or primacy in Wikidata items"[2] is one of the primary obstacles to this goal.
3. The lack of ordering also makes comparing the relative completeness of entities [21], as for instance attempted by the Recoin tool [2], difficult.

Predicting the interestingness of properties is difficult for at least three reasons: (i) Looking at the knowledge base alone is not sufficient: just because a property is very frequent in a knowledge base, one cannot conclude that the property is also very important. For instance, there are about 27k people with a blood type, but only 2k people with a hair color in Wikidata, but nevertheless, the latter is generally more interesting than the former. (ii) The interestingness of properties is very dependent on the person. For a politician, for instance, the political party is generally much more important than music instruments played, while for musicians, it is usually the other way around. (iii) There is a lack of datasets for this task, as most previous work used ablation studies, i.e., assessed performance on randomly removed portions of the data.

Previous work on property recommendation has mostly focused on data-driven approaches [24,1,16,12], which does not approximate human judgment too well. For instance, the Wikidata Property Suggester recommends to add *date of death* and *place of death* as most important missing properties to nearly all persons still alive. Similarly, it appears that frequent properties such as *gender* and *nationality* are overrated. Closest to ours, work by Atzori and Dessi [10] has investigated how to predict what human annotators actually find important, ignoring however the characteristics of individual entities, and using listwise learning-to-rank approaches that are not scalable.

[1] https://github.com/Wikidata-lib/PropertySuggester

[2] `https://meta.wikimedia.org/wiki/Research:Wikidata_gap_analysis#Conclusion`

Contribution Our technical contributions are:(i) We introduce the problem of property ranking and discuss its significance in Section 3. (ii) We develop a human-annotated gold-standard dataset containing 350 sets of an entity and two properties, each annotated with 10 preference judgments in Section 4. (iii) We evaluate baseline approaches and the state-of-the-art against our dataset, showing that these only achieve 61.3% precision on records where humans have 87.5% agreement (Section 5). (iv) We develop techniques based on regression, LSI, LDA and ensembles that are able to achieve 74% precision in Section 6.

2 Background

Learning to rank (L2R) Learning to rank is a classic machine learning problem, where one aims to learn how to optimally rank given items. There are three main approaches to L2R, the so-called pointwise, pairwise, and listwise ranking [6,17]. The pointwise approach, which is usually the easiest to implement, is based on the idea that each item has a score, which can be learned. Items can then be ranked by their score. Issues with the pointwise approach are mainly that the individual scores can be hard to interpret, and are not stable wrt. framing.

The pointwise and listwise approach aim to overcome this limitation of the pointwise approach by learning from ranked pairs and ranked lists, respectively. They can lead to more stable and better rankings, however, potentially require more effort during training data creation.

Wikidata Wikidata is a crowd-sourced knowledge base that maintains information about entities of human knowledge, called *items* in Wikidata parlance, which can be topics of Wikipedia articles (people, cities, movies, etc.) or anything else deemed of interest. Information about an entity is structured as a collection of *statements*, which are pairs consisting of a key, called, *property*, and a value, which can be an atomic data value, an item, a property, or some possibly complex structure. Currently, Wikidata contains roughly 25 mio. entities, and 2719 properties, whose usage follows roughly an exponential distribution. This has two implications: First, it confirms the perception that many properties are quite specific, and apply only to few people. Second, it indicates that solely frequency-based rankings of properties tend to become imprecise in the long tail.

Ranking of Knowledge Base Properties There have been various works on knowledge base property ranking. Most prominently, editors of Wikidata items are supported by the Property Suggester facility, which, given an entity, produces a list of typically 3 to 10 properties for which no statement exists as yet. It implements an approach by Abedjan and Naumann [1] that leverages techniques from association rule mining, and ranks rules according to squared confidence. On the one hand, this allows a few strong rules to outweigh many weaker ones, while on the other hand inapplicable properties may be suggested if they are highly correlated with some existing properties.

Atzori and Dessi [10] addressed the problem to rank the existing properties of individual items in a knowledge base, like Wikidata or DBpedia. They identified three parameters for automating such rankings: (i) algorithms for machine learning to rank; (ii) possible features of properties; and (iii) methods to create training sets. In a study, they chose eight algorithms, nine features, and six training sets and combined them in all possible ways. To test these combinations, they let a group of students pointwise rank the properties of 50 random Wikipedia entities. It turned out that the best combinations beat other state-of-the art techniques by improvements of precision and recall of 5 to 10%. Like the authors of the present paper, Atzori and Dessi aim at creating automated methods that approximate human judgment about property ranks. However, while they want to rank the properties that are already present for an item, we want to find out which properties of an item humans would find important or interesting, regardless of whether or not they are mentioned. Also, we aim to include information specific to the entity, not just use information on the level of the whole class.

Fact Ranking Ranking knowledge base facts by importance, interestingness or unexpectedness is a very related topic [16,11,20]. Recent work by Bast et al. [4], for instance, investigates how to predict the relevance of attribute values on a scale from 0 to 1 for the multi-valued attributes profession and nationality. In their work, they show that methods that are based on the Wikipedia articles of persons and use a generative model can achieve reasonable accuracies on this task, i.e., less than 28% numerical error in 80% of cases. The problem was subsequently posed as challenge at the WSDM 2017 conference [13]. However, having a ranking for facts does not help in the three applications above, as for the recommendation as to what to add, and the completeness comparison task, the properties that do not yet have facts are the ones that should be ranked, while for the natural language generation task, it is generally desired that values of multivalued properties appear together, thus, a ranking of individual facts will not do.

3 Problem Definition and Challenges

We define our problem as follows:

Problem: Given an entity and a set of knowledge base properties, rank the properties according to their interestingness for the entity.

The notion of *interestingness* is hereby left somewhat vague, which however is intentional as our ranking is not intended to serve only one specific purpose. We identify the following technical challenges for this task:

1. *Lack of datasets*: Previous approaches [24,16,1,12] rely on ablation studies, i.e., they randomly remove a portion of knowledge base facts, then evaluate how well they can predict removed facts or properties. This however does not say anything about their ability to rank properties by interestingness. The

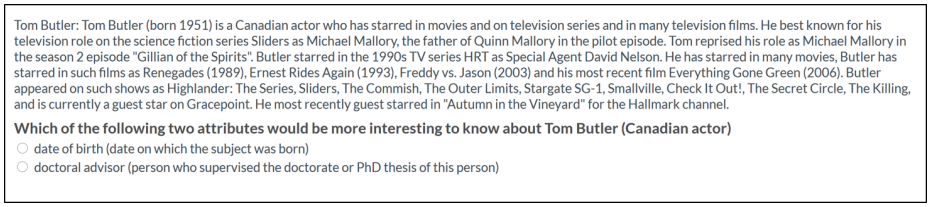

Fig. 1. Interface of the crowdsourcing task.

only dataset available is that of [10], which contains however only pointwise annotations for properties of 50 entities, and is of unknown quality.

2. *Inapplicability of pointwise and listwise annotations:* While pointwise annotations are easy to solicit, the resulting scores can only be interpreted within the context in which they were generated [14,15]. Using the annotation scheme from [4], for instance, we found that the 0-1 interestingness score of properties of medium interestingness was 0.43 when preceded by 7 questions about more interesting properties, but 0.62 if preceded by 7 less interesting ones. As the set of properties in Wikidata is constantly growing,[3] this would mean that pointwise scores collected now could not be interpreted well at later development stages of Wikidata. Likewise, listwise approaches are not applicable, as it is not possible to elicit meaningful rankings for large sets of items.
3. *No supervised learning possible:* Given that there are more than 3 million *humans* in Wikidata, and that we rely on pairwise annotations, it is clear that it is impossible to generate enough training data for supervised learning.

We next address Issue (1), the generation of a suitable dataset.

4 Dataset Preparation

Records We generated 350 random records consisting of a human and two properties, like (*Trump, doctoral advisor, medical condition*). As sampling humans at random from Wikidata would give mostly unknown persons, we decided to sample humans whose Wikidata pages had been edited within the month of November, 2016. As the pages of famous people tend to get edited more often than the pages of non-famous ones, this gave a better mix of humans of different fame. As Wikidata items contain a large fraction of properties that are identifiers, with many stemming from national libraries and directories that can only be understood by experts of the respective domain, we did not consider such ID properties. Of the properties that were not IDs, we considered all that were assigned to humans at least 1000 times, which resulted in 101 as of November 14, 2016.

[3] For instance, as of March 21st, 2016, there were 2202 properties, while as of February 7, 2017, there are 2719 according to `https://tools.wmflabs.org/hay/propbrowse/`

Name	Description	Property 1	Property 2	Preferred	Agrmt
Albert Johnson	Canadian soccer player	military conflict	drafted by	Prop. 2	1
Andrew Collins	British actor	field of work	sister	Prop. 1	0.9
Svetlana Navasardyan	Armenian musician	member of political party	place of detention	-	0.5
Filip Stanislaw Dubiski	Polish officer	residence	sports discipline	Prop. 1	1
Dipankar Bhattacharjee	Indian badminton player	Roman praenomen	bowling style	Prop. 2	0.6
David Ball	Electronic music producer	doubles record	military branch	Prop. 1	0.8
Kalim Kashani	17th century Persian poet	languages spoken/written	sexual orientation	Prop. 1	1

Table 1. Sample records from our gold dataset.

Annotation We used the CrowdFlower platform[4] for obtaining preference judgments. In the annotation task, a short biographical sketch and the two properties were presented to the annotators, and they were asked, knowing about which of the two properties would be more interesting. The core part of the interface is shown in Fig. 1. Quality was ensured via an entrance test and hidden test questions, based on questions unanimously answered in previous runs. For each record, 10 opinions were collected. At 2 ct. per annotation, the platform cost (including fees) to generate the whole dataset was $100. Six sample records are shown in Table 1.

Annotator Agreement The agreement of the annotators is shown in Fig. 2 (solid bars). The dashed bars also show the agreement distribution that would be expected if annotators would answer at random. As one can see, annotator agreement is significantly different from random answers, especially evident for the high agreement cases (e.g., if annotators were to answer at random, only 0.2% instead of 8% of records would have an agreement of 1). The average agreement is 73%, and Fleiss' Kappa is 0.40. We also had two authors of this paper annotate a subset of the records, finding that they had 78% agreement with 16 records where annotators had 80% agreement, and 100% agreement with 16 records where annotators had 100% agreement. In turn, a manual inspection of low-agreement records showed that many of them correspond to cases where both properties appear unrelated, like *goalsScored* and *militaryRank* for *Pope Francis*, on which agreement is difficult.

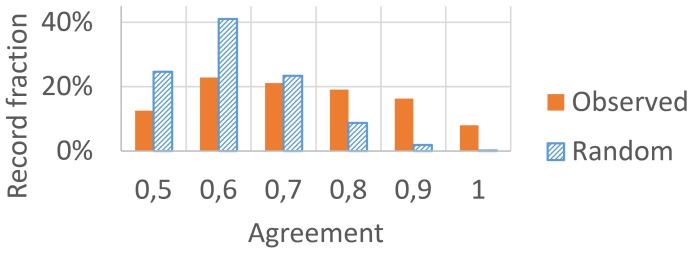

Fig. 2. Agreement distribution in our gold dataset.

[4] `https://www.crowdflower.com`

Method	ppref on records with agreement			
	≥70% (n=223)	≥80% (n=150)	≥90% (n=85)	=100% (n=28)
Random	50%	50%	50 %	50%
Annotators	81.8%	87.5%	93.3%	100%
Human frequency	57.4%	60.6%	62.3%	50%
Occupation frequency	57.4%	58.6%	61.2%	53.6%
Google count	58%	58.3%	61.2%	53.6%
Property suggester	58.7%	61.3%	62.3%	50%

Table 2. Performance of baseline approaches for property ranking.

5 Baselines

Techniques We evaluate four baselines, the first two being simple counts, while the latter two use the state of the art in textual information retrieval and property ranking, respectively:

1. *Human frequency:* This baseline always chooses the property that is more frequently used for *humans* as winner.
2. *Occupation frequency:* A modification of the previous that only looks at people having the same profession, aiming to capture the observation that professions are similar to classes.
3. *Google count:* This method chooses the property with more search results for Google queries concatenating entity name and property as winner. For instance, for *Pope Francis*, *goals scored* and *military rank*, the queries would be "Pope Francis goals scored" (470,000 results) and "Pope Francis military rank" (3,870,000 results), thus, *military rank* would be chosen as winner.
4. *Wikidata property suggester:* This baseline uses the suggestions of the Wikidata property suggester, and represents the state of the art in property ranking [24]. The version available online gives natively only few properties, so we modified the code by removing thresholds, in order to obtain further properties. Given an entity-property-property record from our dataset, the winning property according to this method was the one that was ranked higher by the property suggester.

Evaluation For evaluation, we use the *ppref* measure (precision of preference) [7], henceforth just called precision, which measures the percentage of records where a method proposes the same property as winner as the majority of the annotators does. We present results for records with at least 70%, 80%, 90% and 100% separately, of which there were 226, 152, 85 and 28, respectively. Where trends are similar, we focus in the discussion on the records with at least 80% agreement. If both properties were chosen with equal likelihood, a record would have a probability of 11.2% to fall into this group, i.e., it is unlikely that for many of the records in this group, the property with more votes has been voted that way only by chance.

Results and Analysis As Table 2 shows, all four baselines perform comparably bad, agreeing only in about 60% of the answers with the majority of the annotators. For Baselines (1), (2), and (4), we believe their main weakness is that they rely on describing the data that is present (remember *blood type* vs. *hair color*, where the former exists 14 times as often as the latter). While association rules (Baseline 4) are more sophisticated than simple counts (Baselines 1 and 2), apparently, better capturing correlations does not improve predictions on interestingness. It may be noteworthy that association rules were originally developed for discovering patterns in applications where data is complete, not for making statements about absent data. For Baseline (3), Google count, we trace the low performance to the fact that text search is not able to sufficiently capture the connection between entities and predicates. For instance, for *Pope Francis* and *military rank*, all top ranked results talk about his relation to the Argentinian dictatorship or the Swiss Guard, none about his *own* military rank.

6 Improving Property Ranking

6.1 Transfer Learning via Property Pivoting

We have seen that statistical approaches that learn patterns over the whole dataset do not yield a high precision in predicting pairwise interestingness preferences. We also discussed that it is virtually impossible to obtain enough training data for supervised learning. One approach often taken if supervised learning is not possible is *transfer learning*. Transfer learning refers to the training of models to solve a Problem A, for which enough training data is available, then applying them to a related Problem B [19]. An idea adapted from [4] is to predict, which of two properties is asserted for an entity, and which one is not.

Consider the case of *position played on team* (P413) and *religious order* (P611). There are 225,821 humans in Wikidata that have the former property but not the latter, and 7,976 humans that have the latter but not the former. Are there chances to accurately decide, for a person picked from either of these sets, to which of the two sets it belongs? We call this the *property pivoting problem*.

Deciding Property Pivoting via Regression In the following, we use a logistic regression classifier trained on bags of words taken from person descriptions from Wikipedia to decide the property pivoting problem.

Wikipedia articles provide a messier, but considerably larger source of information about a person than Wikidata. By considering Wikipedia articles as bags of words, we can use the number of occurrences of each word as feature on which to train a classifier. In particular, for each pair of properties, we trained a logistic regression classifier on up to 20,000 entities, if available: 10,000 entities that had the first property but not the second one, and 10,000 entities that had the second but not the first property.

As an example of what these classifiers learned, the box below shows the most distinctive weights learned by the TF-IDF-based regression classifier that was trained for *position played on team* versus *religious order*:

-3.09: footballer	-2.39: season	2.67: jesuit	1.88: work
-2.85: football	-2.21: career	2.43: catholic	1.85: life
-2.75: played	-2.16: league	2.41: died	1.77: order
-2.72: team	-2.08: cup	2.04: priest	1.64: church
-2.67: player	-2.02: club	1.99: works	1.50: death

Negative weights hereby indicate that the occurrences of the word appear frequently in articles of entities with *position played on team* being present instead of *religious order*, while positive weights indicate the opposite. For instance, *season* (weight -2.39) is a word in the former category, while *priest* (weight (2.04) belongs to the latter. Interestingly, also some rather general words like *works* and *died* appear in the latter category. We conjecture that soccer players are mostly recent figures still alive, while monks are more frequently from the past, thus, *died* is more relevant for them, and that monks are known for more diverse activities than sports players, thus the term *work*.

Property Pivoting Quality Evaluated each over 200 entities that had either the one or the other property, the regression classifiers achieved a respectable average precision of 94.8%. For *position played on team* versus *religious order*, for instance, the precision is even 100%, while difficult cases are for instance *field of work* versus *member of* (84% precision), or *child* versus *sister* (72% precision).

Transfer Learning Our hope was that characteristics that describe whether a person has one property, but not another also relate to how interesting the one property is over the other. That is, for people that have a *position played on team* but no *religious order*, maybe knowing about the former is indeed more interesting than the latter. The results of transferring our regression classifiers are shown in Table 5 (third and fourth row). As we can see, the precision on records with at least 80% annotator agreement is 69.3% and 72%, depending on whether TF-IDF is used or not. These precisions are remarkably better than those of the baselines, though still leaving a considerable gap to the 95% precision on the pivoting task, indicating that property pivoting and property interestingness are only moderately related problems. For instance, although all annotators agree that for the soccer player *Albert Johnson*, *drafted by* is more interesting than *military conflict* (Table 1 first row), property pivoting still chooses the latter property, presumably, because *drafted by* is used in Wikidata only in very specific contexts of baseball and ice hockey, not for soccer. Similarly, for Kalim Kashani, a 17th century Persian poet, property pivoting chooses *sexual orientation* over *languages spoken or written*, although crowd agreement is 100% on *languages spoken or written* (Table 1 last row). Presumably the reason is that many people that are not poets have information about languages as well, while sexual orientation is especially frequently asserted for artists.

6.2 Semantic Approaches

As exemplified above, while regression is very well able to decide the property pivoting problem, existence of a property is still only a moderate indicator for

interestingness. In particular, there are several cases where it is intuitive that one property does not apply to a person at all, while another is relevant, but regression is not able to discover that. In the following we thus propose to use semantic similarity as alternative proxy for interestingness. We conjecture that if a property bears some semantic similarity to an entity, like *goals scored* for *Ronaldo*, where humans can easily see an association of Ronaldo scoring goals, then it is more likely that that property is also interesting.

Concretely, we propose to look at the semantic similarity of the textual descriptions of entities and properties. For entities, we use their English Wikipedia articles as text sources, while for properties we use their textual label and description on Wikidata. To compute semantic similarity, we rely on standard latent topic models, in particular Latent Semantic Indexing (LSI) [9] and Latent Dirichlet Allocation (LDA) [5]. We proceed in three steps:

1. In the first step, we train topic models on Wikipedia.
2. In the second step, we represent each entity and each property as distribution over the learned topics.
3. In the third step, for each (entity, property, property) record in our gold dataset, we compute the similarity between the topics of the entity and the two properties. We then assert that the more similar property is the more interesting one.

We further detail each step below.

Learning Topic Models In the first step, we used LSI and LDA to learn 400 and 100 distinct topics over the English Wikipedia text corpus (12.5 GB). We used the *gensim* Python library, with which training could be done on a standard laptop within hours. LDA has a parameter α, which is a prior asserting whether texts are preferentially assigned to more or to fewer topics. This parameter is important for our application as assignment to fewer topics leads to sparser vectors. We report the results for the default value (1/#topics=0.01). We also tested a higher value, 0.05, and a setting called auto-optimization, where α is dynamically set for each topic, but both performed worse. Notably, we found in this setting that too high values of α lead to properties being assigned all to the same topics, resulting in entity-property distances that were indistinguishable. In Table 3 we show some of the most frequent words in four topics as learned by LDA.

Describing Entities and Properties using Topic Models In the next step we used the learned topic models to describe entities and properties. To that end, given a set of words, LSI and LDA are able to compute a distribution of topics that is the most likely one to generate the given set of words.

For the soccer player *Ronaldo*, for instance, LDA states that his Wikipedia article can be described using 52% of Topic 6, 12% of Topic 18, 7% of Topic 41, 6% of Topic 26 (all shown in Table 3), and low fractions of a few others. This distribution appears sensible, as he is most importantly known for playing soccer in leagues and tournaments (Topic 6), comes from Portugal (Topic 18, note

Topic 6: 0.011*league + 0.011*championships + 0.009*tournament + 0.008*cup + 0.008*club + 0.007*football + 0.007*women + 0.007*rank + 0.006*championship + 0.006*round + 0.006*games + 0.005*player + 0.005*goals + 0.005*men + 0.005*teams + 0.005*competed + 0.004*apps + 0.004*division + 0.004*event	**Topic 26:** 0.005*court + 0.004*law + 0.004*police + 0.003*rights + 0.003*women + 0.003*act + 0.003*sarpanch + 0.003*administrated + 0.002*prison + 0.002*case + 0.002*political + 0.002*party + 0.002*president + 0.002*legal + 0.002*justice + 0.002*arrested + 0.002*security + 0.002*said + 0.002*trump + 0.002*supreme
Topic 18: 0.017*brazil + 0.016*brazilian + 0.013*da + 0.012*paulo + 0.012*portuguese + 0.011*rio + 0.008*do + 0.007*portugal + 0.007*janeiro + 0.006*joão + 0.006*silva + 0.005*porto + 0.004*dos	**Topic 41:** 0.005*episode + 0.005*films + 0.004*cast + 0.004*directed + 0.004*television + 0.004*actor + 0.004*tv + 0.004*festival + 0.004*role + 0.004*awards + 0.003*actress + 0.003*drama + 0.003*award + 0.003*director + 0.003*filmography + 0.003*comedy

Table 3. Sample topics as learned by LDA over Wikipedia

that LDA has merged Brazil and Portugal into one topic), is featured frequently in the media (Topic 41) and has been under legal investigation (Topic 26). Similarly, the article of the former US president *Barack Obama* can be generated by combining 48% of Topic 26 (law and politics), 18% of a topic concerned with parties and elections, 12% of a topic concerned with business and industry, and various others.

In the same way, also properties can be described as combinations of topics. The property *height*, for instance, is composed of Topic 14 (geometry), 36 (abstraction), 84 (biology) and an unclear topic 88, whereas *member of sports team* is composed entirely of Topic 6.

Computing Similarities For computing similarities between entities and properties, we use cosine similarity between vectors, a standard approach in vector-space-modelling and word embedding. The idea is to interpret topic distributions as vectors in a high-dimensional space, then compute the distance using the cosine of the angle between these vectors. In the case of *Ronaldo* and *goals scored* versus *military rank*, for instance, we find a cosine similarity of 0.987 versus 0.611. Thus, *goals scored* is the semantically more similar property, and hence we propose this to be the more interesting one.

Analysis The performance of semantic similarity as a proxy for interestingness is shown in Table 5. As we can see, LDA does not outperform the baselines at 60% precision for records with at least 80% annotator agreement. In contrast, LSI performs considerably better at 65.3% precision, though still not achieving the performance of regression (72%).

We find that semantic similarity is better able to capture when one property does not make sense at all, as evidenced by the increase (LDA) or smaller drop (LSI) towards the records with 100% human agreement than regression. For instance, from the records with at least 80% agreement, the precision of LDA increases by 11.4%, the precision of LSI decreases by 1.1%, but the precision of regression drops by 6.1/7.5%.

But there are also various spectacular failures: For *Gabriel Kicsid*, a handball player, for instance, both LSI and LDA believe that *religious order* is more similar

	Human freq.	Occupation freq.	Google count	Property Suggester	Regression (plain)	Regression (TF-IDF)	LDA	LSI
Human frequency	-	0.37	0.11	0.99	0.29	0.33	-0.07	0.07
Occupation frequency	0.37	-	-0.02	0.36	0.33	0.25	0.08	0.00
Google count	0.13	-0.02	-	0.11	0.01	0.03	-0.10	-0.01
Property Suggester	0.99	0.36	0.11	-	0.28	0.33	-0.07	0.05
Regression (unweighted)	0.26	0.33	0.01	0.28	-	0.87	0.14	0.29
Regression (TF-IDF)	0.32	0.25	0.03	0.33	0.87	-	0.09	0.26
LDA	-0.09	0.08	-0.10	-0.07	0.14	0.09	-	0.16
LSI	0.04	0.00	-0.01	0.05	0.29	0.26	0.16	-

Table 4. Pearson correlation coefficients on records with at least 80% agreement

than *follows*, even though all annotators agree that the second is more interesting. A possible reason is that the description of *follows* is quite abstract and hard to match ("*immediately prior item in some series of which the subject is part*"[5]), even though human annotators correctly understand the usage, which is about succession for instance on a position in a team. Similarly, for the Polish officer *Filip Stanislaw Dubitzki* (Table 1, 4th row), unlike all annotators, LSI believes that *sports discipline* is more interesting than *residence*.

6.3 Ensembles

Considering the methods discussed so far along with their strong and weak points, the question arises whether it is possible to combine them and achieve a better performance. By Condorcet's jury theorem [8], it is beneficial to combine weak predictors whenever these show a sufficient level of statistical independence, and have each more than 50% accuracy. To see whether our methods exhibit sufficient statistical independence, we computed pairwise Pearson correlation coefficients, which are shown in Table 4.

There are two surprising insights. The first is the near-perfect correlation between human frequency and Property Suggester (0.99). In other words, even though the Property Suggester uses sophisticated ways for computing association rules and combining predictions, it does essentially nothing different from simply counting how often a property occurs. The second surprising insight is that apart from human frequency/property suggester and the two variants of regression, the pairwise correlation between the methods is very low. This even holds for similar techniques such as human frequency compared with occupation frequency (correlation 0.37) or LSI compared with LDA (correlation

[5] `https://www.wikidata.org/wiki/Property:P155`

0.16). Two methods, Google count and LDA, exhibit almost no correlation to any other method.

The results are in itself remarkable, and suggest that ensembles predictors[6] can give better performances than the individual methods. We tested to take the majority vote from various permutations of three and five of our presented methods. It turned out that the best performing ensemble was not only a combination of three advanced methods, but that even adding some of the baselines improved precision. In particular, the best performing combination used five methods that showed the biggest pairwise difference in Pearson correlation, namely the Google count, LSI, LDA, Occupation frequency, and regression (TF-IDF). This ensemble performed 2% and 4.6% better than the best single method, regression (TF-IDF) on records with at least 80% and 90% agreement, respectively.

6.4 Analysis

Results on the performance of the baselines are shown in Table 2, while the performance of the advanced methods is shown in Table 5. In both cases, we can see random agreement (50%) as a lower bound that any method should outperform, and annotator agreement as an upper bound. As one can see, regression trained on property pivoting as the best single advanced method beats the baselines by a margin of 10% on records with at least 80% agreement, with LSI performing 5% worse, and LDA being on par with the baselines. The best ensembles then add a further 2% precision on top of the regression. We draw the following conclusions:

1. *The state of the art methods alone are inadequate for property ranking.* The state of the art in property suggestion (Property Suggester) and document retrieval (Google count) achieved only 58.3 and 61.3% precision on records where annotators had 87.5% agreement, not significantly different from an approach that simply counts how often a property appears (60.6% precision).
2. *Regression is well-suited for property pivoting, but has still limitations, as it is data-driven.* Regression based on bags-of-words from Wikipedia articles achieves an accuracy of 94.8% for property pivoting on records with at least 80% agreement, i.e., deciding whether an entity has one property but not another. And while it is an expensive method, requiring to train $O(n^2)$ many classifiers for n properties, it is worth its price also for property ranking, as it outperforms the baselines by more than 10%. It has still limitations as it is data-driven, though, i.e., it predicts interestingness of properties based on their presence, which in some cases is not a good indicator.
3. *Semantic approaches are great at discovering applicability of properties, but struggle with short property descriptions.* Semantic similarity based on latent topic

[6] Not to be mixed with *ensemble learning*, a machine learning approach where consecutive instances of the same classifier are trained especially on records that previous instances predicted wrongly. Ensemble learning requires a sufficient amount of labeled training data, which is not available in our case.

Method	ppref on records with agreement ≥70% (n=223)	≥80% (n=150)	≥90% (n=85)	=100% (n=28)
Random	50%	50%	50%	50%
Annotators	81.8%	87.5%	93.3%	100%
Regression (unweighted)	67.7%	69.3%	70.4%	64.3%
Regression (TF-IDF)	70.4%	72%	71.8%	64.3%
LDA	57%	60%	61.2%	71.4%
LSI	59%	65.3%	67%	64.3%
Ensemble of Google count,LSI, LDA, Occupation frequency, regression (TF-IDF)	69.1%	74%	76.4%	67.9%

Table 5. Performance of advanced methods for property ranking.

modeling turned out to be better able to capture cases where certain properties did not at all make sense. Nevertheless, there are problems with description shortness.

4. *Ensembles work best.* Taking the majority vote among methods based on counting (Occupation frequency, Google count), correlation (regression) and semantic similarity (LSI, LDA) approximated human judgment best.

7 Conclusion

We have introduced the problem of property ranking, shown the limitations of the state-of-the-art, and developed approaches that combine classical frequency-based approaches, transfer learning and semantic similarity. Our methods outperform the state of the art by over 10% precision, though still being inferior to human agreement by 11.5%. We hope that the dataset developed in this paper can stimulate research that can further approach human agreement.

We see two interesting avenues to extend this work: One is to improve the methods presented in this paper, for instance, by using other learning algorithms for the property pivoting problem, or by extending the short descriptions from which the semantic methods currently learn. The other is to find completely new approaches to the problem, which, even if they do not individually outperform the existing methods, might add information to ensembles. The challenge would be here to find related problems that can be used for transfer learning.

References

1. Ziawasch Abedjan and Felix Naumann. Improving RDF data through association rule mining. *Datenbank-Spektrum*, 13(2):111–120, 2013.
2. Albin Ahmeti, Simon Razniewski, and Axel Polleres. Assessing the completeness of entities in knowledge bases. In *ESWC P&D*, 2017.
3. Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. *DBpedia: A nucleus for a web of open data*. 2007.

4. Hannah Bast, Björn Buchhold, and Elmar Haussmann. Relevance scores for triples from type-like relations. In *SIGIR*, pages 243–252, New York, NY, USA, 2015.
5. David M Blei, Andrew Y Ng, and Michael I Jordan. Latent Dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.
6. Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: From pairwise approach to listwise approach. In *ICML*, pages 129–136, 2007.
7. Ben Carterette, Paul N Bennett, David Maxwell Chickering, and Susan T Dumais. Here or there: Preference judgments for relevance. In *ECIR*, pages 16–27, 2008.
8. Marquis de Condorcet. Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix. *Paris: Imprimerie Royale*, 1785.
9. Scott Deerwester. Improving information retrieval with latent semantic indexing. 1988.
10. Andrea Dessi and Maurizio Atzori. A machine-learning approach to ranking RDF properties. *FGCS*, 54:366–377, 2016.
11. Nausheen Fatma, Manoj Chinnakotla, and Manish Shrivastava. The unusual suspects: Deep learning based mining of interesting entity trivia from knowledge graphs. In *AAAI 2017*, 2017.
12. Wolfgang Gassler, Eva Zangerle, and Günther Specht. Guided curation of semistructured data in collaboratively-built knowledge bases. *FGCS*, 31, 2014.
13. Stefan Heindorf, Martin Potthast, Hannah Bast, Björn Buchhold, and Elmar Haussmann. WSDM cup 2017: Vandalism detection and triple scoring. 2017.
14. Nicolas Jones, Armelle Brun, and Anne Boyer. Comparisons instead of ratings: Towards more stable preferences. In *WI-IAT*, pages 451–456, 2011.
15. Saikishore Kalloori, Francesco Ricci, and Marko Tkalcic. Pairwise preferences based matrix factorization and nearest neighbor recommendation techniques. 2016.
16. Philipp Langer, Patrick Schulze, Stefan George, Matthias Kohnen, Tobias Metzke, Ziawasch Abedjan, and Gjergji Kasneci. Assigning global relevance scores to dbpedia facts. In *ICDE workshops*, pages 248–253, 2014.
17. Hang Li. A short introduction to learning to rank. In *IEICE Transactions*.
18. Hamid Mousavi, Shi Gao, and Carlo Zaniolo. IBminer: A text mining tool for constructing and populating infobox databases and knowledge bases. *VLDB*, 2013.
19. Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *TKDE*, 2010.
20. Abhay Prakash, Manoj Kumar Chinnakotla, Dhaval Patel, and Puneet Garg. Did you know?-mining interesting trivia for entities from wikipedia. 2015.
21. Simon Razniewski, Fabian M Suchanek, and Werner Nutt. But what do we actually know. *AKBC*, pages 40–44, 2016.
22. Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. YAGO: a core of semantic knowledge. In *WWW*, pages 697–706, 2007.
23. Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledge base. *CACM*, 57(10):78–85, 2014.
24. Eva Zangerle, Wolfgang Gassler, Martin Pichl, Stefan Steinhauser, and Günther Specht. An empirical evaluation of property recommender systems for Wikidata and collaborative knowledge bases. In *Opensym*, 2016.