Negative Knowledge for Open-world Wikidata

Hiba Arnaout
harnaout@mpi-inf.mpg.de
Max Planck Institute for Informatics
Saarbrücken, Germany

Simon Razniewski
srazniew@mpi-inf.mpg.de
Max Planck Institute for Informatics
Saarbrücken, Germany

Gerhard Weikum
weikum@mpi-inf.mpg.de
Max Planck Institute for Informatics
Saarbrücken, Germany

Jeff Z. Pan
j.z.pan@ed.ac.uk
The University of Edinburgh
Edinburgh, United Kingdom

ABSTRACT

The Wikidata knowledge base (KB) is one of the most popular structured data repositories on the web, containing more than 1 billion statements for over 90 million entities. Like most major KBs, it is nonetheless incomplete and therefore operates under the open-world assumption (OWA) - statements not contained in Wikidata should be assumed to have an *unknown* truth. The OWA ignores however, that a significant part of interesting knowledge is *negative*, which cannot be readily expressed in this data model.

In this paper, we review the challenges arising from the OWA, as well as some specific attempts Wikidata has made to overcome them. We review a statistical inference method for negative statements, called *peer-based inference*, and present **Wiki*neg*ata**, a platform that implements this inference over Wikidata. We discuss lessons learned from the development of this platform, as well as how the platform can be used both for learning about interesting negations, as well as about modelling challenges inside Wikidata. **Wiki*neg*ata** is available at https://d5demos.mpi-inf.mpg.de/negation.

1 INTRODUCTION

For years, the Wikimedia Foundation hosted many free, multilingual, and collaborative knowledge projects. These projects are important sources of information for people worldwide. Its most recent accomplishment is Wikidata [32], a central storage for the structured data of its Wikimedia sister projects, including Wikipedia, Wikivoyage, Wiktionary, and Wikisource. The content is created and maintained by Wikidata editors and automated bots. Wikidata, as well as other current web-scale KBs, are crucial in a range of applications like question answering, dialogue agents, and recommendation systems. Given the overwhelming size of these KBs, as well as the constant growth, tracking their quality is key for

future development. These KBs contain almost only *positive* statements, and this is engraved in the *open-world assumption* (OWA) employed on the *Semantic Web*, which states that asserted statements are *true*, while the remainder is *unknown*. For instance, it is asserted in Wikidata that "Stephen Hawking has won the Wolf Prize in Physics" and thus considered *true*, but the statement that "Hawking has won the Nobel Prize in Physics" is absent and thus considered *unknown*. In reality, the reason for the absence of the latter statement is its *falseness*. Being able to formally distinguish whether a statement is false, and asserting its falseness, would enhance Wikidata's ability to display more relevant knowledge in *entity summarization*. For instance, Wikidata lists more than 40 awards that Hawking has won, but does *not* mention one salient award that he did *not* win, the Nobel Prize in Physics. Another use case is *question answering*, where *explicit negative statements* can reduce the ambiguity, and improve the relevance of answers to queries that involve negation. For example, querying for physicists who did *not* win the Nobel Prize in Physics, or politicians with *no* political parties, where simple Wikidata queries [1] return 23k and 346k unranked names, respectively, by simply applying the *closed-world assumption* (CWA). The CWA is the opposite of the OWA, where a statement that does *not* exist in the KB is assumed to be *false*. With web-scale KBs, in particular, storing *positive only* data and applying the CWA at query time is not realistic, due to their incompleteness. Nevertheless, Wikidata's ability to express some kinds of negation is a notable exception among major KBs, which we discuss in more details in Section 2.

In this paper, we revisit the core of the *peer-based inference* method, published in [1], to derive negative statements that are both *relevant* and likely *correct*. In particular, it uses information present on related entities to identify statements of interest. For instance, in Wikidata, most people have *no* academic degree recorded, yet this is often just due to the degree not being relevant, e.g., for many sports people, artists, or politicians of medium to low fame, and hence, the OWA applies. We can only make the stronger deduction of negation in more specific cases. Consider the entity Bill Gates. We notice that many entities similar to him (e.g., Eric Schmidt or Tony Fadell) do have an academic degree. For this reason, assuming *completeness within a group of related entities* is reasonable to draw for this property of Gates, namely that he truly holds no academic degree. At the same time, Gates's peers could also share other information, such as that many of them have relatives. To avoid that negations

of such incidental information comes first, the *peer-based inference* incorporate, on top of collecting peers and inferring candidate negative statements, additional *ranking features*, such as frequency, unexpectedness, etc., tuned using a supervised regression model. Further details are in Section 3.

In this work, we consider three classes of negative statements: (i) grounded negative statements, such as "Tom Cruise did not win the Oscar"; (ii) universally negative statements, such as "Kate Winslet has no Twitter account"; and (iii) conditional negative statements, such as "Albert Einstein never studied at any U.S. university".

Moreover, we present **Wiki*neg*ata** (**NEG**ative statements about **Wiki**data entities), a platform where users can inspect *interesting* negations about Wikidata entities, discovered by the *peer-based inference* model. The interactive system allows users to adjust a combination of parameters that the approach grants. The demo is accessible at https://d5demos.mpi-inf.mpg.de/negation, including a demonstrative video on how to use it[2]. Further details are in Section 4.

Our contributions can be summarized as follows.

- We discuss the lessons learned about the challenges and opportunities arising from deploying the *peer-based inference* methodology in an actual real-world KB (i.e., Wikidata).
- We introduce the *conditional* negative statements, a new class of negative statements.
- We present an interactive platform for discovering interesting negations about Wikidata entities.

2 BACKGROUND

2.1 Logical Background

KB Model. A Knowledge Base (KB) is a collection of structured statements about real world entities and their relationships, with notable projects being Wikidata [32], DBpedia [3], Yago [30], and the Google Knowledge Graph [29]. A typical statement consists of a (subject; predicate; object) triple.

Logical Data Interpretation. Early database paradigms usually employed the CWA, i.e., assumed that all statements not stated to be true were *false* [15, 27]. In contrast, on the *Semantic Web* and for KBs that cover a wider and potentially open-ended range of topics, the OWA has become the standard. The OWA asserts that the truth of statements not stated explicitly is *unknown*. Both semantics represent somewhat extreme positions, as in practice it is neither conceivable that all statements not contained in a KB are false, nor is it useful to consider the truth of all of them as unknown, since in many cases statements not contained in KBs are indeed not there because they are known to be false. An intermediate ground is the *partially-closed-world assumption* (PCWA), where generic rules or metadata are used to specify *parts* of the data operate under the CWA [12, 26]. We show the difference between the 3 kinds of assumptions in Table 1. In this example, the parts of data where the CWA is considered for PCWA is all data related to memberships, i.e., all statements with predicate member of.

Approaching Completeness. An important part of tracking the KB quality is assessing its completeness. Some tools have been proposed to help editors understand the complete and incomplete areas

of Wikidata [8]. It allows them to record complete predicates for a certain entity. For instance, if *all* the children of a certain person are listed, it can be marked as complete, and if not, the editor can add the missing information and mark it as complete. However, relying on editors to maintain a web-scale KB is limited in scale. Therefore, another line of work focused on the automation of Wikidata's completenes assessment. For instance, in Recoin [5, 24], a relative recall model for Wikidata has been proposed. By comparing similar entities (e.g., people sharing a nationality) it lists a number of important missing properties. In [33], the focus is on how the completeness of Wikidata vary across different classes, given a predicate. For example, the degree of completeness of educated at to be measured between male, German, scientists, and female, Indonesian, scientists. [11] investigates different signals to determine the areas where the KB is complete, including popularity, how the KB changes over time, and class information. Through combining these signals in a rule mining approach, missing information can be predicted. [17] proposes a Wikidata completion tool, by extracting facts from Wikipedia, using latest machine learning algorithms. An editor can look at the extracted candidate statement, associated with its evidence, and assess its correctness. If it is correct, she will be redirected to Wikidata to add the newly extract statement.

This body of work offers an important contribution towards Wikidata completion, but it will always be nonetheless, with other web-scale KBs, incomplete. The real world will always contain much more information than what it covered in data stores. Thus, it is important for the KBs to be able to take a stance on information *not* contained in them.

2.2 Wikidata

Wikidata is a free crowd-sourced KB with more than 25k active users and 329 bots. It contains more than a billion statement about 92 million entities. On average, an entity has 14 statements. The taxonomy of Wikidata, i.e., the class hierarchy, is a collaborative work. Editors are encouraged to decide on rules of content creation and management.

2.3 Negation in Wikidata

Wikidata's generally is a positive-statement-only KB operating under the OWA. There are several exceptions however, which give glimpses towards negation.

Deleted Statements. Having access to the history of edits in Wikidata may offer promising candidates for negative information [31], namely by investigating the statements that were once part of a KB but got subsequently deleted (possible candidate for *grounded negative statements*). For instance, we compared the 2017 and 2018 versions of Wikidata, focusing on statements for people (approx. 0.5m deleted statements). On a random sample of 1k deleted statements, over 82% were just caused by ontology modifications, granularity changes, rewordings, or prefix modifications, such as: (Ghandi; lifestyle; Vegetarian) changed to Vegetarianism and (James Green; oxfordID; 101011386) changed to 11386. 15% were statements that were actually restored a year later, and the remaining 3% represented actual negation, yet we found them to be rarely *noteworthy*, i.e., presenting mostly things like corrections of birth dates or location updates reflecting geopolitical changes.

---

[2]Video: https://d5demos.mpi-inf.mpg.de/negation/documentation.html

Statement	$\in$ KB	OWA	CWA	PCWA
(Einstein; award; Nobel Prize in Physics)	✓	true	true	true
(Einstein; award; Oscar)	✗	unknown	false	unknown
(Einstein; member of; Royal Society)	✓	true	true	true
(Einstein; member of; Hungarian Academy of Sciences)	✗	unknown	false	false

Table 1: The difference between OWA, CWA, and PCWA, where membership data is assumed to be complete under the PCWA.

Count Predicates. One way of expressing negation is via counts matching with instances. For example, Joe Biden has 4 children listed for the property child, as well as the statement (number of children; 4). This allows the conclusion that anyone else is *not* his child. Yet while such count predicates exist in popular KBs, none of them has a formal way of dealing with these, especially concerning linking them to instance-based predicates [13].

Deprecation of Statements. Erroneous changes can also be directly recorded via the deprecated rank feature [18], as *grounded negative statements*. However, we found that this mostly relates to errors coming from various import sources, and its focus is not mainly materializing *interesting* negations like that "Stephen Hawking did not win the Nobel Prize in Physics.".

Negated Predicates. Wikidata contains predicates that carry a negative meaning, for instance, does not have part, does not have quality, and different from. Yet these present very specific pieces of knowledge, e.g., (public statement; does not have quality; confidentiality) and (death; different from; brain death), and do not generalize to other properties. Although there have been discussions to extend the Wikidata data model to allow generic property negations[3], these have not been worked out so far.

No-Values. Wikidata can capture statements about *universal absence* via the "no-value" symbol [9]. This allows KB editors to add a statement with an empty object. For example, what we express as $\neg\exists x$(Angela Merkel; child; x), Wikidata allows to be expressed as (Angela Merkel; child; no-value)[4]. There exist 122k of such "no-value" statements, yet only used in narrow domains. For instance, 53% of these statements come for just two properties country (used almost exclusively for geographic features in Antarctica), and follows (indicating that an artwork is not a sequel).

It is encouraging that the Wikidata community has pathways to express some negation. However, as shown above, they come with different purposes and each tackle only a specific subproblem. With our work, we are hoping to open a discussion on providing a generic way to express *noteworthy* negation in Wikidata.

2.4 Negation in Logics and Data Management

Logical rules and constraints, such as Description Logics [4], [6] or OWL, can be used to derive negative statements in *limited domains*. For instance, a person can only have one place of birth. Given that a person was born in Paris, it can be asserted that he was *not* born in any other city. OWL also allows the explicit assertion of negative statements [19], yet so far is predominantly used as ontology description language and for inferring intentional knowledge, not for extensional information (i.e., instances of classes and relations).

[3]https://www.wikidata.org/wiki/Wikidata:Property_proposal/fails_compliance_with
[4]https://www.wikidata.org/wiki/Q567

Different levels of negations and inconsistencies in Description Logic-based ontologies are proposed in a general framework [10]. In RuDiK [21], a rule mining system is proposed. It can learn rules with negative atoms in rule heads (e.g., people born in Japan cannot be U.S. president). This could be utilized towards predicting negative statements. Unfortunately, such rules predict way too many – correct, but uninformative – negative statements, essentially enumerating a huge set of people who are not U.S. presidents. [21] also proposes a precision-oriented variant of the CWA that assumes negation if at least one other relation is linking the subject and object.

2.5 Linguistics and Textual Information Extraction (IE).

Negation is an important feature of human language [20]. While there exists a variety of ways to express negation, state-of-the-art methods are able to detect quite reliably whether a segment of text is negated or not [7], [34]. Yet theories of conversational schemes indicate that negative statements can also be inferred from sentences that do not contain explicit negation: For instance, following Grice's maxims of cooperative communication [14, 25], a reasonable conclusion from the sentence "John has two children, Mary and Bob" is that nobody else is a child of John. In the last year, we have also seen a rising interest in discovering *useful* negation in text. In [16], the target is to build an anti-KB containing negations mined from Wikipedia change logs, with the focus being on finding factual mistakes. In [28], it is to obtain meaningful negative samples for augmenting commonsense KBs.

3 PEER-BASED NEGATION INFERENCE

Based on our previous work [1], we reproduce the most important core here.

Let K^i be an (imaginary) ideal KB that perfectly represents reality, i.e., contains exactly those statements that hold in reality. Under the OWA, (practically) available KBs, K^a contain correct statements, but may be incomplete, so the condition $K^a \subseteq K^i$ holds, but not the converse [26]. We distinguish three forms of negative statements.

DEFINITION 1 (NEGATIVE STATEMENTS).

(1) A grounded negative statement $\neg$(s, p, o) is satisfied if (s; p; o) is not $\in K^i$.

(2) A universally negative statement $\neg\exists o$(s; p; o) is satisfied if there exists no o such that (s; p; o) $\in K^i$.

(3) A conditional negative statement $\neg\exists o$(s; p; o).(o ; p′; o′) is satisfied if there exists *no* o such that (s; p; o) and (o; p′; o′) are $\in K^i$.

An example of a *grounded negative statement* is that "Margot Robbie has never won an Oscar", and is expressed as ¬(Margot
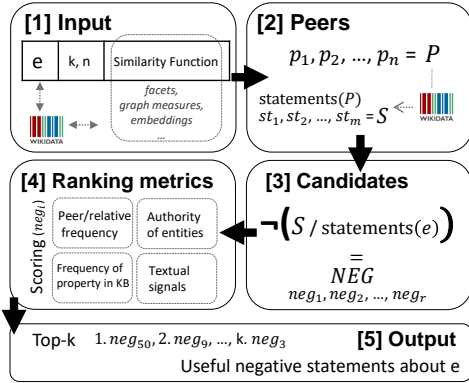
Figure 1: Overview of the peer-based negation inference from [1].

Robbie; award; Oscar), a *universally negative statement* is that "Tyra Banks has never been married", expressed as ¬∃x(Tyra Banks; spouse; x), and a *conditional negative statement* is that "Einstein did not study in any U.S university", expressed as ¬∃x(Albert Einstein; educated at; x).(x; location; U.S.).

For these classes of negative statements, without further constraints, checking that there is no conflict with a *positive* statement is trivial. In the presence of further constraints or entailment regimes, one could resort to (in)consistency checking services [4, 22]. However, deriving negative statements faces other challenges. Although necessary, it is not sufficient to check for conflict with *positive* statements for correctness of negation, due to the OWA. K^i is only a virtual construct, so methods to derive *correct negative* statements have to rely on the limited *positive* information contained in K^a, or utilize external evidence, e.g., from text. Moreover, the set of *correct negative* statements is near-infinite, especially for *grounded negative statements*. Thus, unlike for *positive* statements, negative statement construction or extraction needs a tight coupling with *ranking methods*.

Research Problem. Given an entity e, compile a ranked list of *interesting* grounded negative and universally negative statements.

Peer-based Inference. The method derives *noteworthy* negative statements by combining information from highly related entities, namely peers, with supervised calibration of ranking heuristics, illustrated in Figure 1. The intuition behind this method is that similar entities can suggest expectations for relevant statements about a given entity. For instance, many peers of Stephen Hawking, namely other physicists, have won the Nobel in Physics. We may conclude that the expectation that he did win this prize is feasible, which makes the fact that he did *not* win it an especially useful statement. However, many of his peers were born in Moscow, while he was born in Oxford. We thus need to devise ranking methods that take into account various cues such as frequency, importance, unexpectedness, etc.

Peer-based Candidate Retrieval. To scale the method to web-scale KBs, we first infer a candidate set of negative statements using the PCWA, to be then ranked using several ranking metrics.

Given an entity e, we proceed in three steps:

(1) *Obtain peers:* We collect the *peer groups* of e, and retrieve all their statements. These groups can be based on (i) structured facets of the subject [5], such as nationality for people, or

---

Algorithm 1: Peer-based candidate retrieval algorithm.

Input : knowledge base *KB*, entity e, peer group function *peer_groups*,
size of a group of peers s, number of results k
Output: k-most frequent negative statement candidates for e

2 $P[] = peer\_groups(e, s)$;   // collecting peer groups
3 $N[] = $;   // final list of scored negative statements
4 **for** $P_i \in P$ **do**
5    $candidates = []$;   // predicate and predicate-object pairs of group P_i
6    $ucandidates = []$;   // unique values of *candidates*
7   **for** $pe \in P_i$ **do**
8       $candidates += collect(pe, p, \_)$; // pe: peer, p: predicate
9       $candidates += collect(pe, p, o)$;   // o: object
10   **end**
11   $ucandidates = unique(candidates)$

12   **for** $st \in ucandidates$ **do**
13      $sc = \frac{count(st, candidates)}{s}$;   // scoring statements, st: statement
14   **if** $getscore(st, N) < sc$ **then**
15         $setscore(st, sc, N)$
16   **end**
17   **end**
18 **end**
19 $N = N \setminus inKB(e, N)$;   // remove statements e already has
20 **return** $max(N, k)$

---

classes for other entities, (ii) graph-based measures such as connectivity [23], or (iii) entity embeddings [35].

(2) *Count statements:* We compute relative frequency of all statements within the peer groups, and retain the maxima, if candidates occur in several groups.

(3) *Subtract positives:* We remove statements that hold for e.

The full procedure is shown in Algorithm 1. In a nutshell: peers are selected based on some blackbox function *peer_groups* (line 2). Next, for each peer group, all statements and properties that these peers have are retrieved, and ranked by their relative frequency. Across peer groups, we retain the maximum relative frequencies, if a property or statement occurs across several. Before returning the top results as output, we subtract those already possessed by e.

Consider e=Tom Cruise. Table 2 shows a few examples of his peers and candidate negative statements. We instantiate the peering function to be based on structured information. We collect people who share *professions* with e, as in Recoin [5]. In Wikidata, Cruise has 6 occupations *(actor, director, aircraft pilot, ...)*, thus we would obtain 6 peer groups of entities sharing one of these with him. For readability, we consider statements derived from only one group, *actor*, and only 3 members of the group. The list of negative candidates, *candidates*, are all the statements shown in the columns of the 3 actors. And in this particular example, N is just *ucandidates* with scores for only the *actor* group. Statements that *hold* for Cruise are then dropped from N, namely statements highlighted in green. The top-k in N are then returned. For k=3, for example, the top negative statements are highlighted in red.

Ranking Negative Statements. Often, the candidate negative statements set is large, thus, ranking metrics are needed. Our rationale in the design of the following four ranking metrics is to

Russel Crowe	Meryl Streep	Denzel Washington	Tom Cruise	Negative statements
(award; Oscar)	(award; Oscar)	(award; Oscar)	(citizen; U.S.)	¬(award; Oscar) , 1.0
(citizen; New Zealand)	(citizen; U.S.)	(citizen; U.S.)	(occ.; screenwriter)	¬∃v(convicted; v) , 0.33
(child; x)	(child; y)	(child; z)	(child; u)	¬(citizen; New Zealand) , 0.33
(occ.; screenwriter)		(occ.; screenwriter)		
(convicted; w)				

Table 2: Discovering candidate statements for Tom Cruise from one peer group with 3 peers.

combine frequency signals with popularity and probabilistic likelihoods in a *learning-to-rank model*.

Definition 2 (Ensemble Ranking Score).

$$Score = \begin{cases} \lambda_1 PEER + \lambda_2 POP(o) + \lambda_3 PIVO \\ \quad if \quad \neg(s; p; o) \; is \; satisfied \\ \\ \lambda_1 PEER + \lambda_4 FRQ(p) + \lambda_3 PIVO \\ \quad if \quad \neg\exists o(s; p; o) \; is \; satisfied \end{cases}$$

PEER measures the relative frequency, e.g., 0.9 of the peers have children, but only 0.1 are political activists. *POP* is the popularity of the object entity (Wikipedia page views), e.g., not winning a Nobel Prize would get a higher score than an Oskar Klein Medal. *FRQ* is the frequency of the property when the discovered statement is a *universally negative statement*, e.g., ¬∃x(citizen; x) will get a higher score (3.2m citizenships in Wikidata) than ¬∃x(Instagram; x) (112k Instagram accounts). And finally, *PIVO* is the pivoting likelihood measure where we consider textual background information (Wikipedia embeddings [35]) about e. More on these metrics in [1].

Table 3 shows the top-3 negative statements for Einstein. For instance, Einstein notably refused to work on the Manhattan project, and was suspected of communist sympathies. This makes the assertion that he was *not* a member of the Communist Party in the U.S. *noteworthy*.

3.1 Conditional Negative Statements

The *peer-based inference* method generates two classes of negative statements, namely *grounded negative statements* and *universally negative statements*. These two classes represent extreme cases. With the *grounded negative statements*, we are negating only a *single* assertion. And with the *universally negative statements*, we are negating *all* possible assertions for a property. A compromise between these extremes is to restrict the scope of universal negation. For instance, it is bulky to list all major universities that Einstein did not study at, and it is *not true* that he did not study at any university. However, salient statements are that he *did not study at any U.S. university*, or that he *did not study at any private university*. The challenge is, again, that there is a near-infinite set of *correct* conditional negative statements. And there is a need to identify *noteworthy* ones.

Traversing the space of possible *conditional negative statements*, and scoring them with another set of metrics, is one way to proceed. However, compared to *universally negative statements*, the search space is considerably larger, as for every property, there is a large set of possible conditions via novel properties and constants (e.g., *"that was located in Armenia/Brazil/China/Denmark/..."*, *"that was attended by Abraham/Beethoven/Cleopatra/..."*). So instead, for efficiency, we make use of previously generated *grounded negative statements*. In a nutshell, the *peer-based* inference first generates *grounded negative statements*. Next, subsets of these are *lifted* into more expressive *conditional negative statements*. More details on the lifting technique can be found in [2].

A few examples are shown in Table 4. The *grounded negative statements* for Airbus, a European multinational aerospace corporation, included 3 *cities* that are *not* the location of its headquarters. These cities share a specific aspect, namely they are *all* located in Germany, thus the conditional statement ¬∃x(Airbus; headquarters location; x).(x, country; Germany). In reality, Airbus's registered headquarters is in Leiden, Netherlands. Another example is Pakistan's form of government, which is federal republic, and not a monarchy.

4 THE WIKINEGATA PLATFORM

We have implemented the *peer-based inference* method in a demonstrator called **Wiki*neg*ata**, available at https://d5demos.mpi-inf.mpg.de/negation. Users can interact with the system by choosing the entity and adjusting a combination of parameters that the method grants, e.g., the peering function and the class of negative statements to display (see Figure 3).

System Description. Figure 2 illustrates the client-server architecture of **Wiki*neg*ata**. The web interface runs on Apache Tomcat. We used HTML, CSS, and Javascript, to build the server side of the system. JSP is used on the server side, and PostgreSQL is used to create and manage our database.

Precomputation. Due to the computational heaviness of the *peer-based inference*, and the easiness of the live validation of inferred negations, we rely on offline precomputation for a more efficient retrieval system. For this purpose, we have implemented three orthogonal functions for identifying peers, (i) structured facets, (ii) a graph-based similarity measure, and (iii) embedding-based similarity. For 600k popular entities belonging to 11 classes (including human, organization, country), we have then retrieved 100 most similar peer entities, and used these to identify negative statements. The total size of our database, indexed using B-tree indexes, is 64GB, including 681 million negative, and 100 million positive statements.

Live Validation. Due to real-world changes or KB completions, some of the precomputed negative statements may become incorrect. For example, Brad Pitt has won his *first* Oscar in 2020. So up until 2020, the statement ¬(Brad Pitt; award; Oscar) was valid. We therefore perform a real-time validation using the Wikidata SPARQL endpoint to check that a statement is not contained in Wikidata at interaction time.

Random rank	Property frequency	Ensemble
$\neg\exists x$(instagram; x)	$\neg\exists x$(doctoral student; x)	$\neg$(occupation; astrophysicist)
$\neg$(child; Tarek Sharif)	$\neg\exists x$(candidacy in election; x)	$\neg$(party; Communist Party USA)
$\neg$(award; BAFTA)	$\neg\exists x$(noble title; x)	$\neg\exists x$(doctoral student; x)

Table 3: Top-3 results for Einstein using 3 ranking metrics.

Conditional Statement	Grounded Negative Statements
$\neg\exists x$(Benjamin Franklin; position; x).(x, subclass of; head of state)	x = President of the U.S., Governor of Massachusetts, Gov. of Virginia,..
$\neg\exists x$(Airbus; headquarters location; x).(x, country; Germany)	x = Bonn, Ottobrunn, Hamburg,..
$\neg\exists x$(Pakistan; form of government; x).(x, subclass of; monarchy)	x = constitutional monarchy, absolute monarchy, federal monarchy,..
$\neg\exists x$(Taxi Driver; award; x).(x, subclass of; Academy Awards)	x = Academy Award for Best Picture, Best Writing, Best Actor,..

Table 4: Example of *grounded negative statements* lifted into *conditional negative statements*.
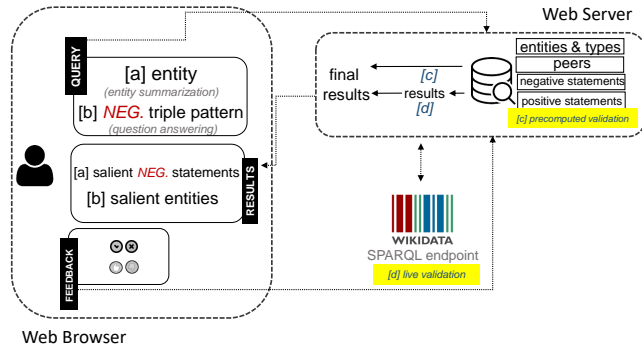

Figure 2: Architecture of Wiki*neg*ata.

Web Interface. Results for Einstein are shown in Figure 3. Although many of his peers were, he was not a member of the Russian Academy of Sciences. Even though he became a U.S. citizen later in his life, he was not granted the American Physical Society honor.

Entity Summarization. The main function allows users to search for *interesting* negations about entities of their choice, Figure 3. **(1)** is the input field with an auto-completion feature. **(2)** allows the choice between live or precomputed validation. **(3)** allows the display of positive statements on top of the negative results. **(4)** is a choice on *how to collect peers for the input entity*. **(5)** is a decision on which classes of negation to show (*regular* refers to the grounded and universally negative, and *conditional* refers to the conditional negative statements). **(6)** is the number of results to display. **(7)** serves as a glimpse into equivalent *positive* answers for every negated property, by either querying Wikidata [32] to show objects that hold for the same predicate, or by creating a Google query for a possible answer. For every result, **(8)** shows the peer entities that the statement holds for. One can give feedback on correctness and informativeness of every statement **(9)**. Finally, by clicking on any peer, **(8)** and **(10)**, a query for that peer is fired. The system retains all the feature values from the previous query (e.g., same similarity function and negation type).

Question Answering. Our platform offers a question answering function. One can search for entities using negative statements, where the entity is a variable. Unlike existing structured search engines, this function returns a ranked list of entities where the

negation is useful and, in many cases, unexpected, due to the scoring of these negations using the peer-based model. A few examples are shown in Table 5. For instance, the first example is looking for people with *no* academic degree. The top-3 results show interesting people, i.e., inventors, like Nikola Tesla, and political leaders, like Abraham Lincoln, who did *not* receive any formal education. In the second example, the statement is requesting *prominent* companies that are not part of the Dow. The Dow Jones Industrial Average is a stock market index that measures the stock performance of large companies in the United States. It includes companies like Intel, Apple Inc., and Microsoft. However, it does not include large American companies like Amazon and Google.

The average retrieval time ranges from 4 to 14 seconds. Expensive queries are mostly the ones that include many validation calls at query time, especially for the case of conditional statements.

5 DISCUSSION

5.1 Using Wiki*neg*ata to Find Interesting Negations in Wikidata

The goal of this work is to show the need of materializing *interesting* negative information in Wikidata, and other web-scale KBs. It offers an enhancement to common use cases, such as entity summarization and question answering. However, it not realistic to extend the KB with every possible negation, even when the KB allows you to, like in the case of Wikidata. We addressed the notion of ***interestingness*** of negative knowledge. What is noteworthy? And how can we discover it? There are more than 5.5k awards that an actor *can* win in Wikidata[5], and Tom Cruise did *not* win 99.99% of them. But one salient award that is relevant to add is the Oscar that he did *not* receive yet. Through peering and the PCWA, we obtain better *contextualizations* for inferring negative statements.

On top of the negations allowed in Wikidata, **Wiki*neg*ata** expresses a new kind of negation, the *conditional negative statements*, where *grounded negative statements* are aggregated based on meaningful aspects. For instance, results for the film Taxi Driver, are shown in Table 4. The movie did not win any category of the Academy Awards. The *conditional statement* here, obtained through the *lifting* technique, aggregated multiple negative statements, based on the one aspect they all share, namely (subclass of; Academy Awards). A second example is Benjamin Franklin. Although he was
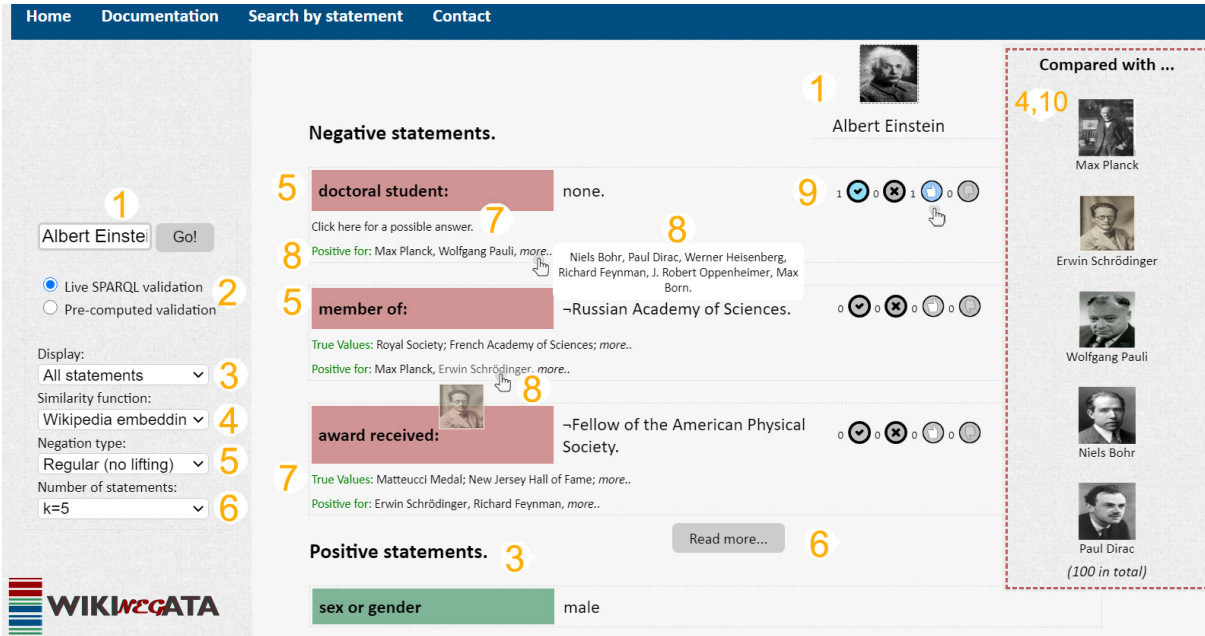
Figure 3: The interface for entity summarization, showing information for Einstein.

Negative Statement	Entity Type	Top-3 Results	Positive for
$\neg \exists y(x; \text{academic degree}; y)$	person	Nikola Tesla, Thomas Edison, Abraham Lincoln	Stephen Hawking, Barack Obama, Louis Pasteur
$\neg(x; \text{part of}; \text{Dow})$	company	Amazon, Google, Bloomberg	Intel, Apple Inc., Microsoft
$\neg(x; \text{member of}; \text{G20})$	country	Norway, Ireland, Sweden	France, U.K., Germany
$\neg(x; \text{award}; \text{Nobel Prize in Physics})$	person	Stephen Hawking, Alexander Bell, Nikola Tesla	Albert Einstein, Max Planck, Marie Curie

Table 5: Examples for question answering using negative statements.

one of the Founding Fathers of the U.S., and unlike his peers, he never held a head of state position.

5.2 Lessons from Real Deployment

Deployment of peer-based inference for Wikidata made several practical limitations surface, which we tackled as follows.

(1) Thresholds: We infer negations for entities that have at least an average number of statements within their class (e.g., on average a business-entity has 29 statements). With peering, we consider for every entity, the closest 100 peers. With peering functions such as structured facets (sharing a profession for instance), we further identify *closest* as in close popularity (Wikipedia page views).

(2) Hierarchical checks (using subclass of and instance of), such as dropping "Hawking is not a *physicist*" because he is a *theoretical physicist*. Or dropping the statement ¬(Douglas Adams; occupation; author) because he is a writer. The same goes for awards (Academy Award) and instances of awards (Academy Award for Best Actor, Best Actress, etc.).

(3) Wikidata contains 8k properties. We filter those that contain identifiers (67%), that describe quantities/dates/geographical coordinates (7%). For the rest, we manually annotate *functional* properties, namely properties that can take *only* one value, and where a *universally negative statement* does not make sense. For instance, there is no point in inferring that

a person has *no* biological father, or has *no* birth place. However, it might be interesting that an American Hollywood star was not born in the U.S. but Indonesia. So, we do not disregard these properties completely.

(4) With conditional negative statements, we manually predefine *interesting* lifting aspects. For instance, for the property educated at, one interesting aspect is the *location* of the universities, or their type *private/public*. We do this for the 30 most frequent properties per class. Automating relevant aspects for every property is a potential direction for future work.

(5) Wikidata's live endpoint validation: at query time, we fire a number of SPARQL queries to make sure the negation is still correct. We also offer an option for offline validation, for a faster run time, but a potential compromise of precision.

These adaptations provide insights about deploying a research concept in a real environment. They mirror experiences of Recoin [5], where in production, additional checks had to be implemented that prevented, for instance, to propose adding death dates to all people without.

5.3 Finding Wikidata Modelling Issues

Not all negative statements in **Wikinegata** are correct, and frequently, wrong negations arise from issues related to Wikidata's data model. An example is Dijkstra and the negative statement that

his field of work is *not* Computer Science, and *not* Information Technology, while he has the positive value Informatics, which is arguably near-synonymous. These entities are often used interchangeably by editors, which results in many inconsistencies. Some other incorrect negative statements could be due to a lack of constraints. For instance, for most businesses, the headquarters location property is completed using *cities*, but for Siemens, the *building* is listed instead (Palais Ludwig Ferdinand), making our inferred statement ¬(Siemens; headquarters location; Munich) incorrect.

5.4 Deployment as Wikidata plugin

We are currently investigating deployment of **Wiki*neg*ata** inside the hosting environment of Toolforge, provided by Wikimedia, to make it accessible from inside Wikidata as plugin. With the exception of the Wikipedia embeddings [35] option, the implementation only requires Wikidata itself to infer negative statements about its entities. However, we should give some considerations to the computational effort. The retrieval of peers and candidate negative statements can be expensive (especially for person-entity), as well as the cost of aggregating *conditional negative statements* by all possible aspects.

6 CONCLUSION

In this paper, we reviewed the OWA nature of KBs, the challenges caused by this assumption, as well as Wikidata's effort to overcome them. We revisited a statistical inference method for negative statements, the *peer-based inference*, and presented **Wiki*neg*ata**, a platform that implements this method over Wikidata. We discussed the observations we made during the development of this platform and how it can be helpful to both users and editors.

In future work, one of our goals is to improve the approach to handle *long tail* entities. It is difficult to find peers for these entities (lack of facets, no Wikipedia embeddings [35], etc.), and to infer *correct* negation candidates. Due to the lack of positive information about them, most of the inferred negations are merely missing information. We are also interested in deploying **Wiki*neg*ata** inside Wikidata to make it more accessible to its community.

ACKNOWLEDGMENTS

REFERENCES

[1] H. Arnaout, S. Razniewski, and G. Weikum. 2020. Enriching Knowledge Bases with Interesting Negative Statements. In *AKBC*.
[2] H. Arnaout, S. Razniewski, G. Weikum, and J. Pan. 2020. Negative Statements Considered Useful. *arXiv* (2020).
[3] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z. Ives, et al. 2007. DBpedia: A Nucleus for a Web of Open Data. In *ISWC*.
[4] F. Baader, D. Calvanese, D. Mcguinness, D. Nardi, and P. F. Patel-Schneider. 2007. *The Description Logic Handbook*. Cambridge University Press.
[5] V. Balaraman, S. Razniewski, and W. Nutt. 2018. Recoin: Relative Completeness in Wikidata. In *Wiki Workshop at WWW*.
[6] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. 2007. Tractable Reasoning and Efficient Query Answering in Description Logics: The DL-Lite family. *Journal Automated Reasoning* (2007).
[7] W. Chapman, D. Hillert, S. Velupillai, M. Kvist, M. Skeppstedt, B. Chapman, M. Conway, M. Tharp, D. Mowery, and L. Deleger. 2013. Extending the NegEx lexicon for multiple languages. *Studies in health technology and informatics* (2013).
[8] F. Darari, R. Eko Prasojo, S. Razniewski, and W. Nutt. 2017. COOL-WD: A Completeness Tool for Wikidata. In *ISWC*.
[9] F. Erxleben, M. Günther, M. Krötzsch, J. Mendez, , and D. Vrandečić. 2014. Introducing Wikidata to the linked data web. In *ISWC*.
[10] G. Flouris, Z. Huang, J. Z. Pan, D. Plexousakis, and H. Wache. 2006. Inconsistencies, Negations and Changes in Ontologies.. In *AAAI*.
[11] L. Galárraga, S. Razniewski, A. Amarilli, and F. M. Suchanek. 2017. Predicting Completeness in Knowledge Bases. In *WSDM*.
[12] Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian Suchanek. 2013. AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In *WWW*.
[13] S. Ghosh, S. Razniewski, and G. Weikum. 2020. Uncovering Hidden Semantics of Set Information in Knowledge Bases. *JWS* (2020).
[14] Herbert P Grice. 1975. Logic and conversation.
[15] Minker J. 1982. On Indefinite Databases and the Closed World Assumption. In *CADE*.
[16] G. Karagiannis, I. Trummer, S. Jo, S. Khandelwal, X. Wang, and C. Yu. 2019. Mining an "anti-knowledge base" from Wikipedia updates with applications to fact checking and beyond. In *VLDB*.
[17] B. Kratzwald, G. Kunpeng, S. Feuerriegel, and D. Diefenbach. 2020. IntKB: A Verifiable Interactive Framework for Knowledge Base Completion. In *COLING*.
[18] S. Malyshev, M. Krötzsch, L. González, L. Gonsior, and A. Bielefeldt. 2018. Getting the Most out of Wikidata: Semantic Technology Usage in Wikipedia's Knowledge Graph. In *ISWC*.
[19] D. McGuinness, F. Van Harmelen, et al. 2004. OWL web ontology language overview. *W3C recommendation* (2004).
[20] R. Morante and C. Sporleder. 2012. Modality and Negation: An Introduction to the Special Issue. *Comput. Linguist.* (2012).
[21] S. Ortona, V. Meduri, and P. Papotti. 2018. RuDiK: rule discovery in knowledge bases. In *VLDB*.
[22] Jeff Z. Pan, Diego Calvanese, Thomas Eiter, Ian Horrocks, Michael Kifer, Fangzhen Lin, and Yuting Zhao (Eds). 2017. *Reasoning Web: Logical Foundation of Knowledge Graph Construction and Query Answering*. Springer.
[23] M. Ponza, P. Ferragina, and S. Chakrabarti. 2017. A Two-Stage Framework for Computing Entity Relatedness in Wikipedia. In *CIKM*.
[24] S. Razniewski, V. Balaraman, and W. Nutt. 2017. Doctoral Advisor or Medical Condition: Towards Entity-Specific Rankings of Knowledge Base Properties. In *ADMA*.
[25] Simon Razniewski, Nitisha Jain, Paramita Mirza, and Gerhard Weikum. 2019. Coverage of information extraction from sentences and paragraphs. In *EMNLP*.
[26] S. Razniewski and W. Nutt. 2011. Completeness of queries over incomplete databases. In *VLDB*.
[27] R. Reiter. 1978. *On Closed World Data Bases*. Logic and Data Bases.
[28] T. Safavi and D. Koutra. 2020. Generating Negative Commonsense Knowledge. *arXiv* (2020).
[29] A. Singhal. 2012. Introducing the Knowledge Graph: things, not strings. https://www.blog.google/products/search/introducing-knowledge-graph-things-not.
[30] F. Suchanek, G Kasneci, , and G. Weikum. 2007. Yago: A Core of Semantic Knowledge. In *WWW*.
[31] T. Tanon and F. Suchanek. 2019. Querying the Edit History of Wikidata. In *ESWC*.
[32] D. Vrandečić and M. Krötzsch. 2014. Wikidata: A Free Collaborative Knowledge base. *CACM* (2014).
[33] A. Wisesa, F. Darari, A. Krisnadhi, W. Nutt, and S. Razniewski. 2019. Wikidata Completeness Profiling Using ProWD. In *K-CAP*.
[34] S. Wu, T. Miller, J. Masanz, M. Coarr, S. Halgrim, D. Carrell, and C. Clark. 2014. Negation's not solved: generalizability versus optimizability in clinical natural language processing. *PloS one* (2014).
[35] I. Yamada, A. Asai, H. Shindo, H. Takeda, and Y. Takefuji. 2018. Wikipedia2Vec: An Optimized Tool for Learning Embeddings of Words and Entities from Wikipedia. *arXiv* (2018).