Wiki*neg*ata: a Knowledge Base with *Interesting* Negative Statements

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
https://knowledge-representation.org/j.z.pan/
The University of Edinburgh
Edinburgh, United Kingdom

ABSTRACT

Databases about general-world knowledge, so-called knowledge bases (KBs), are important in applications such as search and question answering. Traditionally, although KBs use open world assumption, popular KBs only store positive information, but withhold from taking any stance towards statements *not* contained in them. In this demo, we show that storing and presenting *noteworthy* negative statements would be important to overcome current limitations in various use cases. In particular, we introduce the **Wiki*neg*ata** portal, a platform to explore negative statements for Wikidata entities, by implementing a peer-based ranking method for inferring interesting negations in KBs. The demo is available at http://d5demos.mpi-inf.mpg.de/negation.

PVLDB Reference Format:
Hiba Arnaout, Simon Razniewski, Gerhard Weikum, and Jeff Z. Pan. **Wiki*neg*ata**: a Knowledge Base with *Interesting* Negative Statements. PVLDB, 14(1): XXX-XXX, 2020.
doi:XX.XX/XXX.XX

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at http://vldb.org/pvldb/format_vol14.html.

1 INTRODUCTION

Motivation and Problem. Structured general-world knowledge is important for many applications like question answering, dialogue agents, and recommendation systems. Building on a long tradition in databases, this kind of knowledge is now often stored in repositories called knowledge bases (KBs), often in the form of `(subject; predicate; object)` triples, such as `(Stephen Hawking; citizenship; U.K.)`. Recent years have seen a rise of interest in the construction, querying, and maintenance of such KBs. They store positive statements and are a key asset for many knowledge-intensive AI applications. A major limitation of most of these KBs is their inability to deal with negative information [5].

Most current KBs contain virtually only positive statements, whereas statements such as "`Hawking did NOT win the Nobel Prize in Physics`", or "`Alan Turing had NO children`" can only be inferred with the major assumption that the KB is complete - the so-called *closed-world assumption* (CWA). Yet as KBs are only pragmatic collections of knowledge, the CWA is not realistic to assume, and there remains uncertainty whether statements not contained in a KB are *false*, or merely *unknown* to the KB. Being able to distinguish whether a statement is *false* or *unknown* is a major challenge for formal data models both in databases and knowledge bases [7, 10, 14]. This becomes apparent, e.g., in structured knowledge exploration, where KBs provide notable but incomplete lists of relevant positive statements. Including *interesting* negative statements could enhance the quality of these summaries. For example, Wikidata [18] lists more than 40 awards that `Hawking` has won, but does not say anything about a salient award he did not win, the `Nobel Prize in Physics`. Another critical application is question answering, where explicit negative statements can reduce the ambiguity, and improve the relevance of answers to queries that involve negation. An example is to query for physicists who did not win the `Nobel Prize in Physics`, where a naive Wikidata query[1] returns 23K unranked names, by simply applying the CWA.

Approach. The system demonstrated in this paper relies on the so-called *peer-based inference methodology* [1]. In particular, it uses information present on related entities to identify statements of interest, for which a *partial-closed world assumption* (PCWA) is reasonable [15]. For instance, most persons in Wikidata have no academic degree recorded, yet this is often just due to the degree not being important, e.g., for many sports people, artists, or politicians of medium to low fame, and hence, the *open-world assumption* (OWA) applies. We can only make the stronger deduction of negation in more specific cases: Looking at `Stephen Hawking`, we find that many entities similar to him (e.g., `Feynman` or `Oppenheimer`) were `U.S.` citizens, but this information is not mentioned for `Hawking`. Moreover, we find that the property `citizenship` for Hawking is populated, i.e., it carries at least one other value (`British`). By these two observation, we can conclude that the PCWA is reasonable to draw for this situation, and hence, that he was truly no `U.S.` citizen. However, his peers could also share other information, such as that many of them have siblings, or many authored literature. To avoid that negations of such incidental information comes first,

Proceedings of the VLDB Endowment, Vol. 14, No. 1 ISSN 2150-8097.
doi:XX.XX/XXX.XX

[1] https://w.wiki/tXQ

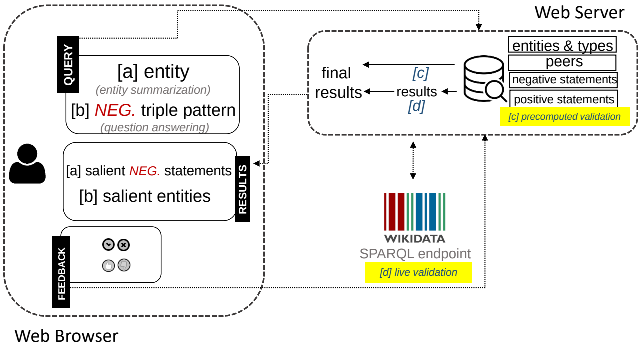

Figure 1: Architecture of Wiki*neg*ata.

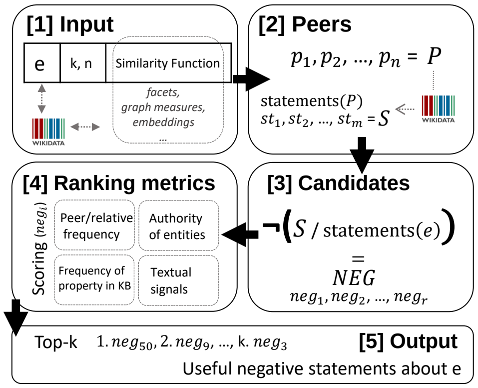

Figure 2: Overview of the peer-based negation inference from [1].

the peer-based inference includes, 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 [1] and in [2].

We present **Wiki*neg*ata** (**NEG**ative statements about **Wikidata** entities), a platform where users can choose different peering functions to explore the peer-based inference methodology, as well as inspect useful negations about Wikidata entities of their choice. The method behind the system is applicable to any other general-purpose KB. The demo is accessible at http://d5demos.mpi-inf.mpg.de/negation, including a demonstrative video on how to use it[2].

2 SYSTEM DESCRIPTION

Figure 1 illustrates the client-server architecture of **Wiki*neg*ata**. On the client side, users enter queries that are sent to the server side, where results are retrieved from the database, then displayed for users. The web interface runs on Apache Tomcat. We used HTML, CSS, and Javascript, to build the interface, JSP as the programming language on the server side, and PostgreSQL to create and manage our database. Positive statements are retrieved from Wikidata.

2.1 Classes of Negative Statements

Our system is able to produce three classes of negations: (i) grounded negative statements ¬`(s; p; o)`, such as ¬`(Hawking; award; Nobel Prize in Physics)`; (ii) universally negative statements ¬∃*x* `(s; p;` *x*`)`, such as ¬∃*x* `(Turing; child;` *x*`)`; and (iii) conditional negative statements ¬∃*x* `(s; p;` *x*`).(`*x*`; p'; o)`, such as ¬∃*x* `(Einstein; studied at;` *x*`).(`*x*`; location; U.S.)`[3].

2.2 Precomputed Peer-based Inference

As peer-based inference is computationally heavy, yet validity of inferences is easy to verify live, this step lends itself to an offline precomputation. For this purpose, we have implemented three orthogonal functions for identifying peers, (i) structured facets of the subject [3], (ii) a graph-based similarity measures (e.g., connectivity [13]), and (iii) embedding-based similarity (e.g., Wikipedia embeddings [19]). 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, as shown in Figure 2, and further detailed in [1]. The total size of our database, indexed using B-tree indexes, is 64GB, including 681[4] million negative and 100 million positive statements.

2.3 Live Validation

Negative statements precomputed offline may turn out incorrect, due to KB completions or real-world changes.

SPARQL Endpoint. Until 2016, `Leonardo Dicaprio` had not won any Oscar, however with his win in that year, in 2020 this assertion is no more true. To address real world changes, we perform a real-time validation using the Wikidata SPARQL endpoint to check that a *precomputed* statement is not contained in Wikidata at *interaction time*.

User Feedback. The feedback feature of the platform is storing up and downvotes on the correctness of the negations displayed. If a negation has at least 3 times more downvotes than upvotes (and has at least 10 downvotes), it is then dropped from the result set.

2.4 Web Interface

Overview. Figure 3 shows the platform with results for `Einstein`. Despite his status as a famous researcher, he truly never formally supervised any PhD students. And unlike many of his peers, including `Max Planck`, he was not a member of the `Russian Academy of Sciences`.

Per-entity Statements. The platform's main function allows users to discover interesting negations about entities of their choice (see Figure 3). The interface has an input entity field **(1)**. One can choose to validate using the Wikidata's live SPARQL endpoint or the prestored positive information **(2)**. This checks real world changes at interaction time. Moreover, one can choose whether to display positive and negative, or only negative statements **(3)**. The similarity function **(4)** is a choice on *how to collect peers for the input entity*. The negation type **(5)** is a decision on which classes of negation to show (*regular* refers to the grounded and universally negative statements, and *conditional* refers to the conditional negative statements). **(6)** is the number of results to display. **(7)** and **(8)** serve as a glimpse into equivalent positive answers for every negated predicate, by creating a Google query for a possible answer, in the case of universally absent negations **(7)**, and querying Wikidata to show objects that hold for the same predicate, in the

[2] Video: https://d5demos.mpi-inf.mpg.de/negation/videos/demo.mp4
[3] "Albert Einstein never studied at any U.S. university".
[4] 600,000 entities × (189 negations on avg.) × 3 similarity functions × 2 negation modes

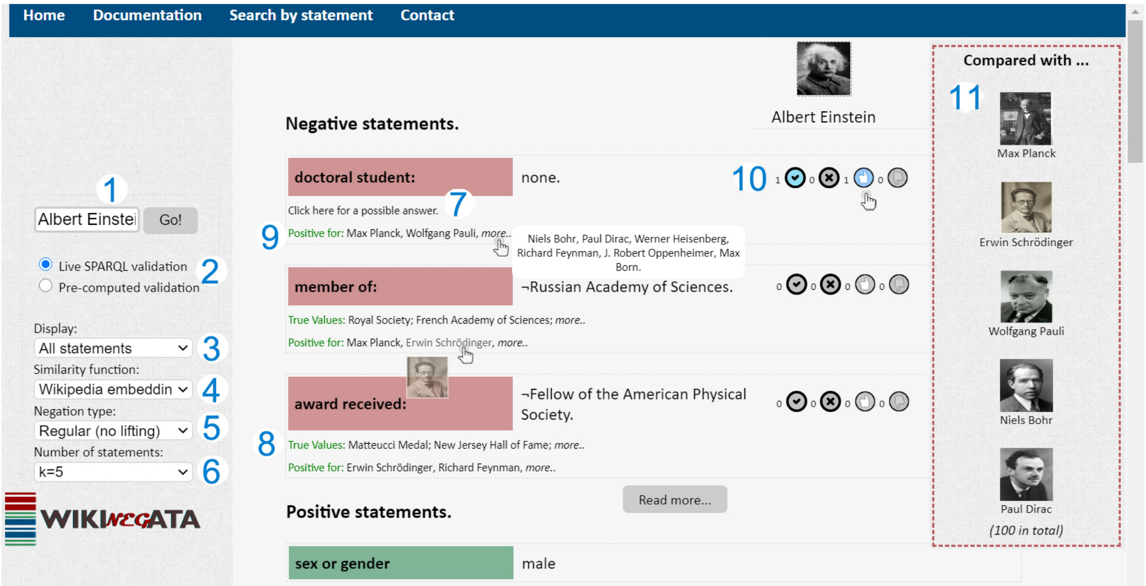

Figure 3: The interface for per-entity statements, showing information for `Einstein`.

case of grounded negations **(8)**. For every result, **(9)** shows the peer entities that the statement *holds* for. Feedback is important to us. One can give signals on correctness and informativeness of results **(10)**. Finally, Under "compared with" **(11)**, the closest peers for the input entity are displayed. By clicking on a peer, a query for that entity is fired. In the unfortunate case where no results are found, a number of alternative queries and features are suggested.

Search by Statement. An additional function allows users to search for entities that share a certain negation, such as "`Physicists who did NOT win the Nobel Prize in Physics`". Unlike existing structured search engines, this function returns a ranked list of entities where the negation is useful and often unexpected. Thus, instead of a random list of physicists, the user is shown a set of *prominent* physicists who did not receive this prize.

The average retrieval time ranges from 4 to 14 seconds. Most of the expensive queries are the ones that include many calls to the SPARQL API, especially for the retrieval of conditional statements.

3 DEMONSTRATION EXPERIENCE

We showcase the **Wiki*neg*ata** platform in three scenarios.

Scenario 1 - Understanding Peer-based Inference. To understand the peer-based inference method, **Wiki*neg*ata** offers various levels of introspection. For each entity, peers are shown at the right side of the screen. Moreover, for each inferred negative statement, the set of peers for which it is positive, is shown below the statement. For instance, suppose the user enters `Steve Carell`, the star of the successful comedy show `The Office`, and learns that he has *not* won an `Emmy Award`. She can explore the reason this negation has been inferred and highly ranked by looking at the peers for which this statement holds, i.e., other comedians such as `Garry Shandling`, as well as other positive values for `Carell` for that predicate, i.e., awards such as the `Golden Globe`, that enabled the partial completeness assumption.

Users can actively influence the produced results, too. Suppose a user enters `Jeff Bezos` as an input entity. She notices that

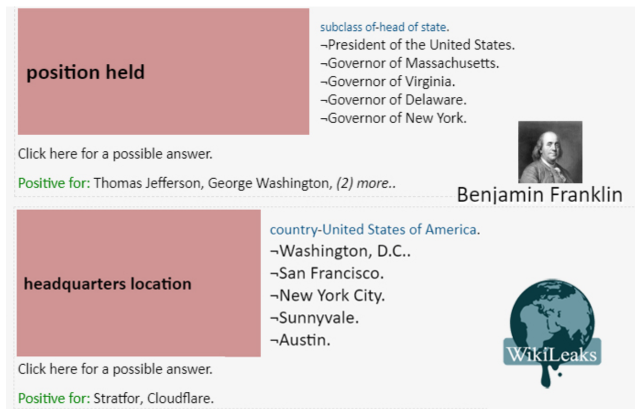

Figure 4: Conditional statements for `Franklin` and `WikiLeaks`.

`Elon Musk` is among his peer when peering via Wikipedia embeddings [19], but not via graph-based measures. This indicates that `Bezos` and `Musk` share latent information, but have few exact predicate-object combinations in common. Different peer groups then also lead to different deductions, embeddings ranking highest that `Musk` is not a writer, graph-based measures ranking highest that he is not a university teacher. More examples are in Table 1.

Using the conditional negative statements, one can explore the lifting technique. With one of the Founding Fathers of the United States as the user's input entity, with *conditional* for negation type, she receives the lifted statement that he *never* held a head of state position. Figure 4 shows that this technique aggregated 5 grounded negative statements, using one shared relevant aspect.

Scenario 2 - Knowledge Exploration. Interested in negative information about `Iceland`, a user enters this country as input entity and leaves the other fields set to their default values, namely Wikipedia embeddings for peering and regular for negation type. She then starts inspecting the results and was surprised to learn that `Iceland` is not a member of the `European Union`. She marks

Entity	Peers	Similarity Function
Oprah Winfrey	Stedman Graham, Barbara Walters, Steve Harvey	Wikipedia emb. [19]
Oprah Winfrey	Maya Angelou, Ellen DeGeneres, Halle Berry	Graph-based measures
Jeff Bezos	Mark Zuckerberg, Larry Page, Bill Gates	Graph-based measures
Jeff Bezos	Elon Musk, Eric Schmidt, Ginni Rometty	Wikipedia emb. [19]
Amazon	Intel, Adobe, Microsoft	Graph-based measures
Amazon	Best Buy, Walmart, eBay	Wikipedia emb. [19]

Table 1: Peers of `Winfrey`, `Bezos`, and `Amazon`, using different peering functions.

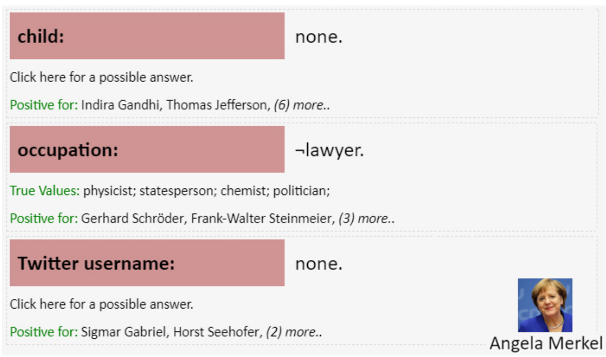

Figure 5: Results for `Angela Merkel`.

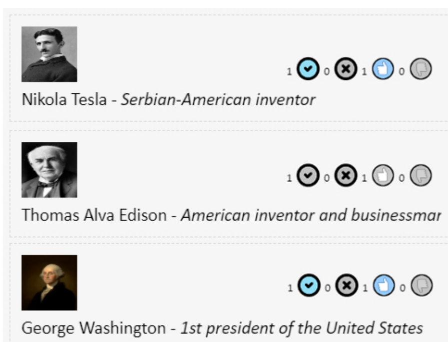

Figure 6: Results for having *no* academic degree.

this negative statement as informative. Next, she enters `Angela Merkel` (Figure 5). She learns some diverse negative information about her, including that she has no children, unlike many world leaders, is not on `Twitter`, and has not studied law.

Scenario 3 - Question Answering. The user wants to find prominent people who have *no* academic degree, using our search by statement function, shown in Figure 6. The figure shows the most salient results, and more can be loaded. She was surprised that two of the most popular `American` inventors and the first `American` President did not receive any formal education.

4 RELATED WORK

Although incompleteness is an established problem in DB research [11, 14], KB construction has focused on positive statements [4, 17], and the problem of compiling interesting negative statements about entities is new. Nevertheless, there are a few related prior works. Among large KBs, Wikidata is a notable exception insofar as it allows to add assertions with an empty object value, corresponding to what we refer to as universally negative statements. In logics and data management, there is work on employing rule mining to predict the completeness of predicates for a given entity [6], and devising a rule mining system that can learn rules with negative atoms in the rule heads (e.g., "`people born in Germany cannot be U.S. president`") [12]. Also related is learning which attributes are mandatory, for only non-mandatory absent predicates are candidates for universally negative statements [9]. Recently, there is a rising interest in discovering *useful* negation in text, such as building an anti-knowledge base containing negations [8] mined from Wikipedia change logs, with a focus on factual mistakes, and obtaining negative samples for commonsense knowledge [16].

5 CONCLUSION

We demonstrated how negative statements can enhance KBs for knowledge exploration and question answering. Related material can be found on our webpage.[5]

REFERENCES

[1] H. Arnaout, S. Razniewski, and G. Weikum. 2020. Enriching Knowledge Bases with Interesting Negative Statements. *AKBC* (2020).
[2] H. Arnaout, S. Razniewski, G. Weikum, and Jeff Z. Pan. 2021. Negative Knowledge for Open-world Wikidata. *Wiki Workshop at WWW* (2021).
[3] V. Balaraman, S. Razniewski, and W. Nutt. 2018. Recoin: Relative Completeness in Wikidata. *Wiki Workshop at WWW* (2018).
[4] Xin Luna Dong et al. 2014. From Data Fusion to Knowledge Fusion. *VLDB* (2014).
[5] Giorgos Flouris, Zhisheng Huang, Jeff Z. Pan, Dimitris Plexousakis, and Holger Wache. 2006. Inconsistencies, Negations and Changes in Ontologies. In *AAAI*.
[6] L. Galárraga, S. Razniewski, A. Amarilli, and F. M Suchanek. 2017. Predicting completeness in knowledge bases. *WSDM* (2017).
[7] Tomasz Imieliński and Witold Lipski Jr. 1989. Incomplete information in relational databases. In *Readings in Artificial Intelligence and Databases*.
[8] G. Karagiannis et al. 2019. Mining an "anti-knowledge base" from Wikipedia updates with applications to fact checking and beyond. *VLDB* (2019).
[9] J. Lajus and F. M Suchanek. 2018. Are all people married? Determining obligatory attributes in knowledge bases. *WWW* (2018).
[10] Willis Lang, Rimma V Nehme, Eric Robinson, and Jeffrey F Naughton. 2014. Partial results in database systems. In *SIGMOD*.
[11] Amihai Motro. 1989. Integrity= validity+ completeness. *TODS* (1989).
[12] S. Ortona, V. Meduri, and P. Papotti. 2018. RuDiK: rule discovery in knowledge bases. *VLDB* (2018).
[13] M. Ponza, P. Ferragina, and S. Chakrabarti. 2017. A Two-Stage Framework for Computing Entity Relatedness in Wikipedia. In *CIKM*.
[14] Simon Razniewski, Flip Korn, Werner Nutt, and Divesh Srivastava. 2015. Identifying the extent of completeness of query answers over partially complete databases. *SIGMOD* (2015).
[15] Yuan Ren, Jeff Z. Pan, and Yuting Zhao. 2010. Closed World Reasoning for OWL2 with NBox. *Journal of Tsinghua Science and Technology* 6 (2010).
[16] Tara Safavi and Danai Koutra. 2020. Generating Negative Commonsense Knowledge. *arXiv* (2020).
[17] J. Shin, S. Wu, F. Wang, C. De Sa, C. Zhang, and C. Ré. 2015. Incremental knowledge base construction using DeepDive. *VLDB* (2015).
[18] D. Vrandečić and M. Krötzsch. 2014. Wikidata: A Free Collaborative Knowledge base. *CACM* (2014).
[19] I. Yamada et al. 2018. Wikipedia2Vec: An Optimized Tool for Learning Embeddings of Words and Entities from Wikipedia. *arXiv* (2018).

[5] Homepage: https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/knowledge-base-recall/interesting-negations-in-kbs