Negative Statements Considered Useful

Hiba Arnaout[a,*], Simon Razniewski[a], Gerhard Weikum[a], Jeff Z. Pan[b]

[a] *Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken 66123, Germany*
[b] *School of Informatics, The University of Edinburgh, Informatics Forum, Edinburgh EH8 9AB, Scotland*

Abstract

Knowledge bases (KBs) about notable entities and their properties are an important asset in applications such as search, question answering and dialogue. All popular KBs capture virtually only positive statements, and abstain from taking any stance on statements not stored in the KB. This paper makes the case for explicitly stating salient statements that do *not* hold. Negative statements are useful to overcome limitations of question answering, and can often contribute to informative summaries of entities. Due to the abundance of such invalid statements, any effort to compile them needs to address ranking by saliency. We present a statistical inference method for compiling and ranking negative statements, based on expectations from positive statements of related entities in peer groups. Experimental results, with a variety of datasets, show that the method can effectively discover notable negative statements, and extrinsic studies underline their usefulness for entity summarization. Datasets and code are released as resources for further research.

Keywords: knowledge bases, negative statements, information extraction, statistical inference, ranking

1. Introduction

Motivation and Problem. Structured knowledge is crucial in a range of applications like question answering, dialogue agents, and recommendation systems. The required knowledge is usually stored in KBs, and recent years have seen a rise of interest in KB construction, querying and maintenance, with notable projects being Wikidata [46], DBpedia [3], Yago [43], or the Google Knowledge Graph [42]. These KBs store positive statements such as *"Renée Zellweger won the 2020 Oscar for the best actress"*, and are a key asset for many knowledge-intensive AI applications.

A major limitation of all these KBs is their inability to deal with negative information. At present, most major KBs only contain positive statements, whereas statements such as that *"Tom Cruise did not win an Oscar"* could 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 positive statements, the CWA is not realistic to assume, and there remains uncertainty whether statements not contained in a KBs are false, or truth is merely unknown to the KB.

Not being able to formally distinguish whether a statement is false or unknown poses challenges in a variety of applications. In medicine, for instance, it is important to distinguish between knowing about the absence of a biochemical reaction between substances, and not knowing about its existence at all. In corporate integrity, it is important to know whether a person was never employed by a certain competitor, while in anti-corruption investigations, absence of family relations needs to be ascertained. In data science and machine learning, on-the-spot counterexamples are important to ensure the correctness of learned extraction patterns and associations.

State of the Art and its Limitations. This has consequences for usage of KBs: for instance, today's *question answering* (QA) systems are well geared for positive questions, and questions where exactly one answer should be returned (e.g., quiz questions or reading comprehension tasks) [17, 51]. In contrast, for answering negative questions like *"Actors without Oscars"*, QA systems lack a data basis. Similarly, they struggle with positive questions that have no answer, like *"Children of Angela Merkel"*, too often still returning a best-effort answer even if it is incorrect. Materialized negative information would allow a better treatment of both cases.

Approach and Contribution. In this paper, we make the case that important negative knowledge should be explicitly materialized. We motivate this selective materialization with the challenge of overseeing a near-infinite space of false statements, and with the importance of explicit negation in search and question answering.

We consider three classes of negative statements: grounded negative statements *"Tom Cruise is not a British citizen"*, conditional negative statements *"Tom Cruise has not won an award from the Oscar categories"* and universal negative statements *"Tom Cruise is not member of any political party"*. In a nutshell, given a knowledge base and an entity e, we select highly related entities to e (we call

*Corresponding author
Email addresses: harnaout@mpi-inf.mpg.de (Hiba Arnaout), srazniew@mpi-inf.mpg.de (Simon Razniewski), weikum@mpi-inf.mpg.de (Gerhard Weikum), j.z.pan@ed.ac.uk (Jeff Z. Pan)

them *peers*). We then use these peers to derive positive expectations about e, where the absence of these expectations might be interesting for e. In this approach, we are assuming completeness within a group of peers. More precisely, if the knowledge base does not mention the *Nobel Prize in Physics* as an award won by *Stephen Hawking*, but does mention it for at least one of his peers, it is assumed to be false for *Hawking*, and not a missing statement. This is followed by a ranking step where we use predicate and object prominence, frequency, and textual context in a learning-to-rank model.

The salient contributions of this paper are:

1. We make the first comprehensive case for materializing useful negative statements, and formalize important classes of such statements.

2. We present a judiciously designed method for collecting and ranking negative statements based on knowledge about related entities.

3. We show the usefulness of our models in use cases like entity summarization, decision support, and question answering.
   Experimental datasets and code are released as resources for further research[1].

The present article extends the earlier conference publication [2] in several directions:

1. We extend the statistical inference to ordered sets of related entities, thereby removing the need to select a single peer set, and obtaining finer-grained contextualizations of negative statements. (Section 5);

2. To bridge the gap between overly fine-grained grounded negative statements and coarse universal negative statements, we introduce a third notion of negative statement, *conditional negative statements*, and show how to compute them post-hoc (Section 6);

3. We evaluate the value of negative statements in an additional use case, with hotels from booking.com (Section 7.4).

2. State of the Art

2.1. Negation in Existing Knowledge Bases

Deleted Statements. Statements that were once part of a KB but got subsequently deleted are promising candidates for negative information [45]. As an example, we studied deleted statements between two Wikidata versions from 1/2017 and 1/2018, focusing in particular on statements for people (close to 0.5M deleted statements). On a random sample of 1K deleted statements, we found that over 82% were just caused by ontology modifications, granularity changes, rewordings, or prefix modifications. Another 15% were statements that were actually restored a year later, so presumably reflected erroneous deletions. 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.

In Wikidata, erroneous changes can also be directly recorded via the deprecated rank feature [29]. Yet again we found that this mostly relates to errors coming from various import sources, and did concern the active collection of interesting negations, as advocated in this article.

Count and Negated Predicates. Another way of expressing negation is via counts matching with instances, for instance, storing 5 children statements for Trump and statement `(number of children; 5)` allows to infer that anyone else is not a child of Trump. 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 [21].

Moreover, some KBs contain relations that carry a negative meaning. For example, DBpedia has predicates like *carrier never available* (for phones), or *never exceed alt* (for airplanes), Knowlife [15] contains medical predicates like *is not caused by* and *is not healed by*, and Wikidata contains *does not have part* and *different from*. Yet these present very specific pieces of knowledge, and do not generalize. Although there have been discussions to extend the Wikidata data model to allow generic opposites[2], these have not been worked out so far.

Wikidata No-Values. Wikidata can capture statements about *universal absence* via the "no-value" symbol [16]. This allows KB editors to add a statement where the object is empty. For example, what we express as ¬`(Angela Merkel; child; _)`, the current version of Wikidata allows to be expressed as `(Angela Merkel; child; no-value)`[3]. As of 8/2019, 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).

2.2. Negation in Logics and Data Management

Negation has a long history in logics and data management. Early database paradigms usually employed the closed-world assumption (CWA), i.e., assumed that all statements not stated to be true were false [41], [31]. On the Semantic Web and for KBs, in contrast, the open-world assumption (OWA) has become the standard. The OWA

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

[2] `https://www.wikidata.org/wiki/Wikidata:Property_proposal/fails_compliance_with`

[3] `https://www.wikidata.org/wiki/Q567`

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.

In limited domains, logical rules and constraints, such as Description Logics [4], [10] or OWL, can be used to derive negative statements. An example is the statement that every person has only one birth place, which allows to deduce with certainty that a given person who was born in France was not born in Italy. OWL also allows to explicitly assert negative statements [30], 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). Different levels of negations and inconsistencies in Description Logic-based ontologies are proposed in a general framework [18].

The AMIE framework [19] employed rule mining to predict the completeness of properties for given entities. This corresponds to learning whether the CWA holds in a local part of the KB, inferring that all absent values for a subject-predicate pair are false. For our task, this could be a building block, but it does not address the inference of *useful* negative statements.

RuDiK [35] is a rule mining system that can learn rules with negative atoms in rule heads (e.g., people born in Germany 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. The same work also proposed a precision-oriented variant of the CWA that assumes negation only if subject and object are connected by at least one other relation. Unfortunately, this condition is rarely met in interesting cases. For instance, none of the negative statements in Table 6 have alternative connections between subject and object in Wikidata.

2.3. Related Areas

Linguistics and Textual Information Extraction (IE). Negation is an important feature of human language [32]. 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 [12], [48].

A body of work targets negation in medical data and health records. [13] developed a supervised system for detecing negation, speculation and their scope in biomedical data, based on the annotated BioScope corpus [44]. [22] specifically focuses on negations via "not". The challenge here is the right scoping, e.g., "Examination could not be performed due to the Aphasia" does not negate the medical observation that the patient has Aphasia. In [9], a rule-based approach based on NegEx [11], and a vocabulary-based approach for prefix detection were introduced. PreNex [8] also deals with negation prefixes. The authors propose to break terms into prefixes and root words to identify this kind of negation. They rely on a pattern matching approach over medical documents. In [26], an anti-knowledge base containing negations is mined from Wikipedia change logs, with the focus however being again on factual mistakes, and precision, not interestingness is employed as main evaluation metric. We explore text extraction more in the proposed *pattern-based query log extraction* method in our earlier conference publication [2].

Statistical Inference and KB Completion. As text extraction often has limitations, data mining and machine learning are frequently used on top of extracted or user-built KBs, in order to detect interesting patterns in existing data, or in order to predict statements not yet contained in a KB. There exist at least three popular approaches, rule mining, tensor factorization, and vector space embeddings [47]. Rule mining is an established, interpretable technique for pattern discovery in structured data, and has been successfully applied to KBs for instance by the AMIE system [28]. Tensor factorization and vector space embeddings are latent models, i.e., they discover hidden commonalities by learning low-dimensional feature vectors [37]. To date, all these approaches only discover positive statements.

Ranking KB Statements. Ranking statements is a core task in managing access to KBs, with techniques often combining generative language-models for queries on weighted and labeled graphs [27, 49, 1]. In [6], the authors propose a variety of functions to rank values of type-like predicates. These algorithms include retrieving entity-related texts, binary classifiers with textual features, and counting word occurrences. [24] focuses on identifying the informativeness of statements within the context of the query, by exploiting deep learning techniques.

3. Model

For the remainder we assume that a KB is a set of statements, each being a triple $(s; p; o)$ of subject s, property p and object o.

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 [40]. We distinguish two 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* K^i.
2. *A universally negative statement* $\neg\exists o : (s, p, o)$ *is satisfied if there exists no* o *such that* $(s; p; o) \in K^i$.

An example of a grounded negative statement is that *"Samuel L. Jackson never won an Oscar for Best Actor"*, and is expressed as ¬(`Samuel L. Jackson; award; Oscar for Best Actor`). And an example of a universally negative statement is that *"Leonardo DiCaprio has never been married"*, expressed as $\neg\exists o$:(`Leonardo DiCaprio; spouse; o`). Both types of negative statements represent standard logical constructs, and could also be expressed in the OWL ontology language. Grounded negative statements could be expressed via negative property statements (e.g., `NegativeObjectPropertyStatement(:hasWife :Bill :Mary)`), while universally negative statements could be expressed via `owl:complementOf` and `ObjectAllValuesFrom` [16]. Without further constraints, for these classes of negative statements, 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, 36].

Yet compiling negative statements faces two other challenges. First, being not in conflict with positive statements is a necessary but not a sufficient condition for correctness of negation, due to the OWA. In particular, 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. Second, the set of correct negative statements is near-infinite, especially for grounded negative statements. Thus, unlike for positive statements, negative statement construction/extraction needs a tight coupling with ranking methods.

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

4. Peer-based Statistical Inference

We next present a method to derive useful negative statements by combining information from similar entities ("peers") with supervised calibration of ranking heuristics. The intuition behind this method is that similar entities can give cues towards what expectations regarding relevant statements for an entity are. For instance, several entities similar to the physicist *Stephen Hawking* have won the *Nobel in Physics*. We may thus conclude that him not winning this prize could be an especially useful statement. Yet related entities also share other traits, e.g., many famous physicists are *U.S.* citizens, while *Hawking* is *British*. 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, in the first stage, we compute a candidate set of negative statements using the CWA, to be ranked in the second stage. Given a subject e, we proceed in three steps:

1. *Obtain peers:* We collect entities that set expectations for statements that e could have, the so-called *peer groups* of e. These groups can be based on (i) structured facets of the subject [5], such as *occupation, nationality*, or *field of work* for people, or classes/types for other entities, (ii) graph-based measures such as distance or connectivity [38], or (iii) entity embeddings such as TransE [7], possibly in combination with clustering, thus reflecting latent similarity.
2. *Count statements:* We count the relative frequency of all predicate-object-pairs (i.e., (_,p,o)) and predicates (i.e., (_,p,_)) within the peer groups, and retain the maxima, if candidates occur in several groups. In this way, statements are retained if they occur frequently in at least one of the possibly orthogonal peer groups.
3. *Subtract positives:* We remove those predicate-object-pairs and predicates that exist for e.

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 $\boldsymbol{P}[]= peer\_groups(e,s)$; // collecting peer groups
3 $N[]=$; // final list of scored negative statements
4 **for** $P_i \in \boldsymbol{P}$ **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
19 $candidates{+}{=}collect(pe,p,o)$; // 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 $N{-}{=}inKB(e,N)$; // remove statements e already has
20 return $max(N,k)$

Algorithm 1 shows the full procedure of the peer-based inference method. In line 2, peers are selected based on some blackbox function *peer_groups*. Subsequently, for each peer group, we collect all statements and properties that these peers have, and rank them 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 entity e.

Example 1. Consider the entity e=*Brad Pitt*. Table 1 shows a few examples of his peers and candidate negative statements.

We instantiate the peer group choice to be based on structured information, in particular, shared occupations with the subject, as in Recoin [5]. In Wikidata, Pitt has 8 occupations *(actor, film director, model, ...)*, thus we would obtain 8 peer groups of entities sharing one of these with Pitt. For readability, let us consider statements derived from only one of these peer groups, *actor*. Let us assume 3 entities in that peer group, *Russel Crowe, Tom Hanks, and Denzel Washington*. The list of negative candidates, *candidates*, are all the predicate and predicate-object pairs shown in the columns of the 3 actors. And in this particular example, N is just *ucandidates* with scores for only the *actor* group, namely `(award; Oscar for Best Actor)`: 1.0, `(citizen; New Zealand)`: 0.33, `(child; _)`: 1.0, `(occupation; screenwriter)`: 1.0, `(convicted; _)`: 0.33, and `(citizen; U.S.)`: 0.67. Positive candidates of *Brad Pitt* are then dropped from N, namely `(citizen; U.S.)`: 0.67 and `(child; _)`: 1.0. The top-k of the rest of candidates in N are then returned. For k=3 for example, the top-k negative statements are ¬`(award; Oscar for Best Actor)`, ¬`(occupation; screenwriter)`, and ¬`(citizen; New Zealand)`.

Note that without proper thresholding, the candidate set grows very quickly, for instance, if using only 30 peers, the candidate set for *Brad Pitt* on Wikidata is already about 1500 statements.

Ranking Negative Statements. Given potentially large candidate sets, in a second step, ranking methods are needed. Our rationale in the design of the following four ranking metrics is to combine frequency signals with popularity and probabilistic likelihoods in a *learning-to-rank model*.

1. *Peer frequency (PEER):* The statement discovery procedure already provides a relative frequency, e.g., 0.9 of a given actor's peers are married, but only 0.1 are political activists. The former is an immediate candidate for ranking.
2. *Object popularity (POP):* When the discovered statement is of the form ¬`(s; p; o)`, its relevance might be reflected by the popularity[4] of the `Object`. For example, ¬`(Brad Pitt; award; Oscar for Best Actor)` would get a higher score than ¬`(Brad Pitt; award; London Film Critics' Circle Award)`, because of the high popularity of the *Academy Awards* over the latter.
3. *Frequency of the Property (FRQ):* When the discovered statement has an empty `Object` ¬`(s; p; _)`, the frequency of the `Property` will reflect the authority of the statement. To compute the frequency of a `Property`, we refer to its frequency in the KB. For example, ¬`(Joel Slater; citizen; _)` will get a higher score (3.2M citizenships in Wikidata) than ¬`(Joel Slater; twitter; _)` (160K twitter usernames).
4. *Pivoting likelihood (PIVO):* In addition to these frequency/view-based metrics, we propose to consider textual background information about e in order to better decide whether a negative statement is relevant. To this end, we build a set of statement pivoting classifier [39], i.e., classifiers that decide whether an entity has a certain statement (or property), each trained on the Wikipedia embeddings [50] of 100 entities that have a certain statement (or property), and 100 that do not[5]. To score a new statement (or property) candidate, we then use the pivoting score of the respective classifier, i.e., the likelihood of the classifier to assign the entity to the group of entities having that statement (or property).

The final score of a candidate statement is then computed as follows.

Definition 2 (Ensemble Ranking Score).

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

Hereby λ_1, λ_2, λ_3, and λ_4 are hyperparameters to be tuned on data withheld from training.

5. Order-oriented Peer-based Inference

In the previous section, we assume a binary peer relation as the basis of peer group computation. In other words, for each entity, any other entity is either a peer, or is not. Yet in expressive knowledge bases, relatedness is typically graded and multifaceted, thus reducing this to a binary notion risks losing valuable information. We therefore investigate, in this section, how negative statements can be computed while using ordered peer set.

Orders on peers arise naturally when using real-valued similarity functions, such as Jaccard-similarity, or cosine distance of embedding vectors. An order also naturally arises when one uses temporal or spatial features for peering. Here are some examples:

1. *Spatial:* Considering the class *national capital*, the peers closest to *London* are *Brussels* (199 miles), *Paris* (213 miles), *Amsterdam* (223 miles), etc.
2. *Temporal*: The same holds for temporal orders on attributes, e.g., via his role as president, the entities most related to *Trump* are *Obama* (predecessor), *Bush* (pre-predecessor), *Clinton* (pre-pre-predecessor), etc.

[4]Wikipedia page views.

[5]On withheld data, linear regression classifiers achieve 74% avg. accuracy on this task.

Table 1: Discovering candidate statements for *Brad Pitt* from one peer group with 3 peers.

Russel Crowe	Tom Hanks	Denzel Washington	Brad Pitt	Candidate statements
(award; Oscar for Best Actor) (citizen; New Zealand) (child; _) (occup.; screenwriter) (convicted; _)	(award; Oscar for Best Actor) (citizen; U.S.) (child; _) (occup.; screenwriter)	(award; Oscar for Best Actor) (citizen; U.S.) (child; _) (occup.; screenwriter)	(citizen; U.S.) (child; _)	¬(award; Oscar for Best Actor), 1.0 ¬(occup.; screenwriter), 1.0 ¬(citizen; New Zealand), 0.33 ¬(convicted; _), 0.33

Formalization. Given a target entity e_0, a similarity function $sim(e_a, e_b) \rightarrow \mathcal{R}$, and a set of candidate peers $E = \{e_1, ..., e_n\}$, we can sort E by sim to derive an ordered list of sets $L = [S_1, ..., S_n]$, where each S_i is a subset of E.

Example 2. Let us consider temporal recency of having won the *Oscars for Best Actor/Actress* as similarity function w.r.t. the target entity *Olivia Colman*. The ordered list of closest peer sets S is `[{Frances McDormand, Gary Oldman}, {Emma Stone, Casey Affleck}, {Brie Larson, Leonardo DiCaprio}, {Julianne Moore, Eddie Redmayne}.., {Janet Gaynor, Emil Jannings}]`.

Given an index of interest m ($m \leq n$), we have a prefix list $S_{[1,m]}$ of such an ordered peer set list L. For any negative statement candidate *stmt*, we can compute two ranking features:

1. *Prefix-volume (VOL)*: The prefix volume denotes the size of the prefix in terms of peer entities considered, i.e., $VOL = |S_1 \cup ... \cup S_m|$. Note that the volume should not be mixed with the length m of the prefix, which does not allow easy comparison, as sets may contain very different numbers of members.

2. *Peer frequency (PEER)*: As in Section 4, *PEER* denotes the fraction of entities in $S_1 \cup ... \cup S_m$ for which *stmt* holds, i.e., FRQ / VOL, where FRQ is the number of entities sharing the statement.

Note that these two ranking features change values with prefix length. In addition, we can also consider static features like *POP* and *PIVO*, as introduced before.

Consider the entity e=*Olivia Colman* from our example, with prefix length 3. For the statement `(citizen of; U.S.)`, FRQ is 5 and VOL is 6, i.e., unlike *Olivia Colman*, 5 out of the 6 winners of the previous 3 years are U.S. citizens. Now considering prefix length 2, for the statement `(occupation; director)`, FRQ is 1 and VOL is 4, i.e., unlike *Olivia Colman*, 1 out of the 4 winners of the previous 2 years are directors.

Ranking. Now that we have an ordered list of peer sets, the next question is now how to identify informative negative statements. It is easy to see that a statement is better than another, if it has both a higher peer frequency (*PEER*) and prefix volume (*VOL*). For example, the statement `¬(citizen of; U.S.)` above is preferable over `¬(occupation; director)`, due to it being both reported on a larger set of peers, and with higher relative frequency. Yet statements can be incomparable along these two metrics, and this problem even arises when comparing a statement with itself over different prefixes: Is it more helpful if 3 out of the previous 4 winners are *U.S.* citizens, or 7 out of the previous 10?

To resolve such situations, we propose to map the two metrics into a single one, as follows:

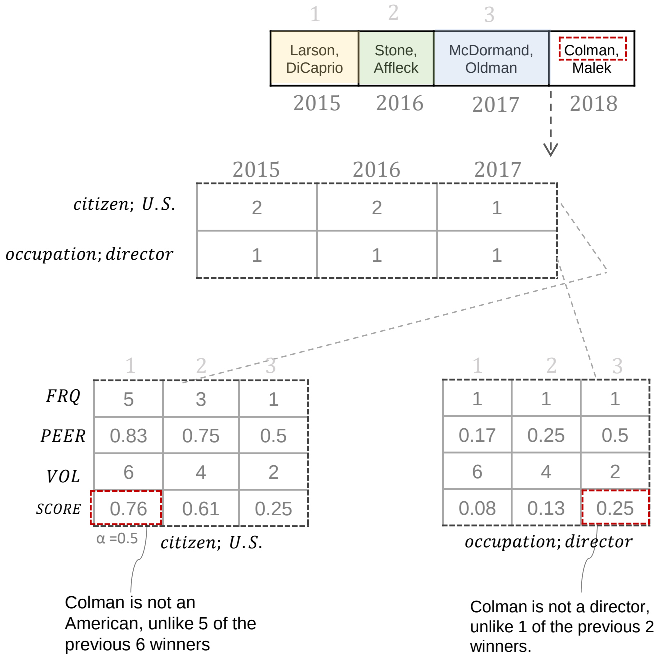
$$score(stmt, L, m) = \lambda \cdot PEER + (1 - \lambda) \cdot log(FRQ) \quad (1)$$

where λ is again a hyperparameter allowing to trade off the effects of the two variables. Note that we propose a logarithmic contribution of *FRQ* - this is based on the rationale that larger number of peers is preferable. For example, for the same *PEER* value 0.5, we can have a statement with 5 peers out of 10 and 1 peer out of 2.

Given the above example, the score for *Olivia Colman*'s negative statement `¬(citizen of; U.S.)` at prefix length 3 and $\alpha = 0.5$ is 0.76, with verbalization as "unlike 5 of the previous 6 winners". The same statement with prefix length 2 will receive a score of 0.61, with verbalization as "unlike 3 of the previous 4 winners". As for `¬(occupation; director)` at prefix length 3 and $\alpha = 0.5$ is 0.08, with verbalization as "unlike 1 of the previous 6 winners". The same statement with prefix length 2 will receive a score of 0.13, with verbalization as "unlike 1 of the previous 4 winners". This example is illustrated in Figure 1.

Computation. Having defined how statements over ordered peer sets can be ranked, we now present an efficient algorithm, Algorithm 2, to compute the optimal prefix length per statement candidate, based on a single pass over the prefix. Given the entity *Olivia Colman* as the input entity e, ordered sets of her peers are collected (line 2), for example *winners of Oscar for Best Actor/Actress.* All statements of the peers are then retrieved from the KB (for loop at line 8). For every candidate statement *st*, the score(s) of the statement is computed with different prefix lengths, starting with 1 (position of e in the ordered set) and stopping with prefix length of *pos* (if e is at position 5, then the last prefix length is of size 5). The maximum score is then returned with its corresponding values of *FRQ* and *VOL*, i.e., max_frq and max_vol (line 31). The returned candidate statement with its highest score is compared across ordered sets of peers, in order to either be replaced or disregarded from the final set. The last two steps (line 20 and 21) are again to drop any candidate negative statements that holds for e and then return the top-k salient negative statements about e.

Figure 1: Retrieving useful negative statements about *Olivia Colman*.



6. Conditional Negative Statements

In the previous section, we have presented a method for generating two classes of negative statements, grounded negative statements, and universally negative statements. These two classes represent extreme cases: each grounded statement negates just a single assertion, while each universally negative statement negates all possible assertions for a property. Consequently, grounded statements may make it difficult to be concise, while universally negative statements do not apply whenever at least one positive statement exists for a property. A compromise between these extremes is to restrict the scope of universal negation. For example, it is cumbersome 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*. We call these statements *conditional negative statements*, as they represent a conditional case of universal negation. In principle, the conditions used to constrain the object could take the form of arbitrary logical formulas. For proof of concept, we focus here on conditions that take the form of a single triple pattern.

Definition 3. *A conditional negative statement takes the form ¬∃ o: (s; p; o), (o; p'; o'). It is satisfied if there exists no o such that (s; p; o) and (o; p'; o') are in* K^i.

In the following, we call the property p' the aspect of the conditional negative statement.

Example 3. Consider the statement that Einstein did not study at any *U.S.* university could be written as $\neg\exists o$: `(Einstein; educated at; o), (o; located in; U.S.)`. It is true, as *Einstein* only studied at *ETH Zurich, Luitpold-Gymnasium, Alte Kantonsschule Aarau*, and *University of Zurich*, located in *Switzerland* and *Germany*. Another possible conditional negative statement is $\neg\exists o$: `(Einstein; educated at; o), (o; instance of; private University)`.

Algorithm 2: Order-oriented peer-based candidate retrieval algorithm.

Input : knowledge base KB, entity e, ordered peers function *ordered_peers*, number of results k, hypeparameter of scoring function α
Output: top-k negative statement candidates for e

```
L[]= ordered_peers(e) ;  // collecting ordered sets of peers
N[]= ;  // final list of scored negative statements
for L_i ∈ L do
    candidates = [] ; // predicate and predicate-object pairs of ordered set L_i
    ucandidates=[] ;  // unique values of candidates
    pos=position(L_i, e) ;  // position of e in the ordered set
    for pe ∈ L_i do
        candidates+=collect(pe, p, _) ;  // pe: peer, p: predicate
        candidates+=collect(pe, p, o) ;  // o: object
    end
    ucandidates = unique(candidates)

    for st ∈ ucandidates do
        sc = scoring(st, L_i, e, pos, α) ;  // scoring statements with different prefix lengths, st: statement
        if getscore(st, N) < sc.max then
            setscore(st, sc, N)
        end
    end
end
N-=inKB(e, N) ;  // remove statements e already has
return max(N, k)

Function scoring(st, S, e, pos, α):
    max = - inf; frq = 0; vol=0; max_frq = - inf; max_vol = - inf;
    for j = pos; j >= 1; j−− do
        vol += countentities(S[i]) ;  // number of entities at position i
        frq += count(st, candidates, S[i]) ; // number of entities at position i sharing the candidate statement
        sc = α * frq/vol + (1 − α) * log(frq);
        if sc > max then
            max = sc; max_frq = frq; max_vol = vol;
        end
    end
return max, max_frq, max_vol
```

As before, the challenge is that there is a near-infinite set of true conditional negative statements, so a way to identify interesting ones is needed. For example, *Einstein* also did not study at any *Jamaican* university, nor did he study at any university that *Richard Feynman* studied at, etc. One way to proceed would be to traverse the space of possible conditional negative statements, and score them with another set of metrics. Yet 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 propose to make use of previously generated grounded negative statements: In a nutshell, the idea is first generate grounded negative statements, then in a second step, to *lift* subsets of these into more expressive conditional negative statements. A crucial step is to define this lifting operation, and what the search space for this operation is.

With the *Einstein* example, shown in Table 2, we could start from three relevant grounded negative statements that *Einstein* did not study at *MIT, Stanford*, and *Harvard*. One option is to lift them based on aspects they all share: their locations, their types, or their memberships. The values for these aspects are then automatically retrieved: they are all located in the *U.S.*, they are all private universities, they are all members of the *Digital Library Federation*, etc., however, not all of these may be interesting. So instead we propose to *pre-define* possible aspects for lifting, either using manual definition, or using methods for facet discovery, e.g., for faceted interfaces [34]. For manual definition, we assume the condition to be in the form of a single triple pattern. A few samples are shown in Table 3. For `educated at`, it would result in statements like "e was not educated in the *U.K.*" or "e was not educated at a public university"; for `award received`, like "e did not win any category of *Nobel Prize*"; and for `position held`, like "e did not hold any position in the *House of Representatives*".

Algorithm 3 shows the process of lifting a set of ground negative statements. Consider e as *Einstein*, the set of possible aspects ASP for lifting containing only two properties for `educated at` for readability `[(educated at: located in, instance of)]`, NEG as `[¬(educated at: MIT, Stanford, Harvard)]`, and k = 2. The loop at line 2 considers every property in NEG with its corresponding objects(s), whereas the nested loops at line 3 and 4 will visit every aspect referring the this property and look for aspect values all (or a number of) these corresponding objects share. For example, one aspect (manually pre-defined) is *location* and one aspect value (automatically retrieved) is *U.S.*, shared by all objects, *MIT, Stanford*, and *Harvard*. Hence this instance of the aspect, `(located in; U.S.)` receives a score of 3, and is added to the conditional negation result set, at line 4. After scoring all the aspects and their possible values, top-2 (k=2) conditional negative statements are returned. In this example, the score of `¬(educated at - (located in; U.S.))` is 3, and of `¬(educated at - (instance of; private university))` is 3 as well.

Algorithm 3: Lifting grounded negative statements algorithm.

Input : knowledge base KB, entity e, aspects ASP $= [(x_1\text{: } y_1, y_2, ..), ..., (x_n\text{: } y_1, y_2, ..)]$, grounded negative statements about e NEG $= [\neg(p_1\text{: } o_1, o_2, ..), ..., (p_m\text{: } o_1, o_2, ..)]$, set of conditional statements about e $cond\_NEG = []$, number of results k

Output: k-most frequent conditional negative statements for e

1 **for** $neg.p \in NEG$ **do**
2   **for** $asp \in ASP$ **do**
3   **for** $o \in neg.o$ **do**
4             $cond\_NEG$ += KB.get(`(o; asp; ?aspvalue)`)
5   **end**
6   **end**
7 **end**
8 $cond\_NEG$-=$inKB(e, cond\_NEG)$
9 return $max(cond\_NEG, k)$

7. Experimental Evaluation

7.1. Peer-based Inference

Setup. We instantiate the peer-based inference method with 30 peers, popularity based on Wikipedia page views, and peer groups based on entity occupations. The choice of this simple peering function is inspired by Recoin [5]. In order to further ensure relevant peering, we also only considered entities as candidates for peers, if their Wikipedia viewcount was at least a quarter of that of the subject entity. We randomly sample 100 popular Wikidata people. For each of them, we collect 20 negative statement candidates: 10 with the highest *PEER* score, 10 being chosen at random from the rest of retrieved candidates. We then used crowdsourcing[6] to annotate each of these 2000 statements on whether it was interesting enough to be added to a biographic summary text (Yes/Maybe/No). Each task was given to 3 annotators. Interpreting the answers as numeric scores (1/0.5/0), we found a standard deviation of 0.29, and full agreement of the 3 annotators on 25% of the questions. Our final labels are the numeric averages among the 3 annotations.

Hyperparameter Tuning. To learn optimal hyperparameters for the ensemble ranking function (Definition 2), we trained a linear regression model using 5-fold crossvalidation on the 2000 labels for usefulness. Four example rows are shown in Table 4. Note that the ranking metrics were normalized using a ranked transformation to obtain a uniform distribution for every feature.

[6] `https://www.mturk.com`

Table 2: Negative statements about *Einstein*, before and after lifting.

Grounded negative statements	**Conditional negative statements**
¬(educated at; MIT)	¬ educated at - (*located in; U.S.*)
¬(educated at; Stanford)	¬ educated at - (*instance of; private uni.*)
¬(educated at; Harvard)	

Table 3: A few samples of property aspects.

Property	**Aspect(s)**
educated at	located in; instance of;
award received	subclass of;
position held	part of;

The average obtained optimal hyperparameter values were -0.03 for *PEER*, 0.09 for *FRQ(p)*, -0.04 for *POP(o)*, and 0.13 for *PIVO*, and a constant value of 0.3., with a 71% out-of-sample precision.

Ranking Metric. To compute the ranking quality of our method against a number of baselines, we use the Discounted Cumulative Gain (DCG) [25], which is a measure that takes into consideration the rank of relevant statements and can incorporate different relevance levels. DCG is defined as follows:

$$DCG(i) = \begin{cases} G(1) & if \text{ i=1} \\ DCG(i-1) + \frac{G(i)}{log(i)} & otherwise \end{cases}$$

where i is the rank of the result within the result set, and $G(i)$ is the relevance level of the result. We set $G(i)$ to a value between 1 and 3, depending on the annotator's assessment. We then average, for each result (statement), the ratings given by all annotators and use this as the relevance level for the result. Dividing the obtained DCG by the DCG of the ideal ranking, we obtained the normalized DCG (nDCG), which accounts for the variance in performance among queries (entities).

Baselines. We use three *baselines*: As a naive baseline, we randomly order the 20 statements per entity. This baseline gives a lower bound on what any ranking model should exceed. We also use two competitive embedding-based baselines, TransE [7] and HolE [33]. For these two, we used pretrained models, from [23], on Wikidata (300K statements) containing prominent entities of different types, which we enriched with all the statements about the sampled entities. We plug their prediction score for each candidate grounded negative statement.[7]

Results. Table 5 shows the average *nDCG* over the 100 entities for top-k negative statements for k equals 3, 5, 10, and 20. As one can see, our ensemble outperforms the best baseline by 6 to 16% in NDCG. The coverage column reflects the percentage of statements that this model was able to score. For example, for the *Popularity of Object*, $POP(o)$ metric, a universally negative statement will not be scored. The same applies to TransE and HolE.

Ranking with the *Ensemble* and ranking using the *Frequency of Property* outperformed all other ranking metrics and the three baselines, with an improvement over the random baseline of 20% for k=3 and k=5. Examples of ranked top-3 negative statements for *Albert Einstein* are shown in Table 6. That *Einstein* notably refused to work on the *Manhattan* project, and was suspected of communist sympathies is noteworthy. Also, despite his status as famous researcher, he truly never formally supervised any PhD student.

Correctness Evaluation. We used crowdsourcing to assess the correctness of results from the peer-based method. We collected 1K negative statements belonging to the three types, namely people, literature work, and organizations. Every statement was annotated 3 times as either correct, incorrect, or ambiguous. 63% of the statements were found to be correct, 31% were incorrect, and 6% were ambiguous. Interpreting the scores numerical (0/0.5/1), annotations showed a standard deviation of 0.23.

PCA vs. CWA For a sample of 200 statements about people (10 each for 20 entities), half generated only relying on the CWA, half additionally filtered to satisfy the PCA (subject has at least one other object for that property [20]), we manually checked correctness. We observed 84% accuracy for PCA-based statements, and 57% for CWA-based statements. So the PCA yields significantly more correct negative statements, though losing the ability to predict universally negative statements.

Subject coverage. In column 2 of Table 5, we measure the number of entities in a KB that the method can infer negative statements about. Our peer-based inference method offers a very high subject coverage and is able to discover negative statements about almost any existing entity in a given KB, whereas for pre-trained embedding-based baselines, many subjects are out-of-vocabulary, or come with too little information to predict statements.

7.2. Inference with Ordered Peers

In the following, we use temporal order on specific roles, or of specific attribute values, to compute ordered peer sets. In particular, we use two common forms of temporal information in Wikidata to compute such peer groups:

- **Time-based Qualifiers (TQ)**: Temporal qualifiers are time signals associated with statements about

[7] Note that both models are not able to score statements about universal absence, a trait shared with the object popularity heuristic in our ensemble.

Table 4: Data samples for illustrating hyperparameter tuning.

Statement	PEER	FRQ(p)	POP(o)	PIVO	Label
¬(Bruce Springsteen; award; Grammy Lifetime Achievement Award)	0.8	0.8	0.55	0.25	0.83
¬(Gordon Ramsay; lifestyle; mysticism)	0.3	0.8	0.8	0.65	0.33
¬(Albert Einstein; doctoral student; _)	0.85	0.9	0.15	0.4	0.66
¬(Celine Dion; educated at; _)	0.95	0.95	0.25	0.95	0.5

Table 5: Ranking metrics evaluation results for peer-based inference.

Ranking Model	Coverage(%)	$nDCG_3$	$nDCG_5$	$nDCG_{10}$	$nDCG_{20}$
Random	100	0.37	0.41	0.50	0.73
TransE [7]	31	0.43	0.47	0.55	0.76
HolE [33]	12	0.44	0.48	0.57	0.76
Property Frequency	11	**0.61**	**0.61**	**0.66**	**0.82**
Object Popularity	89	0.39	0.43	0.52	0.74
Pivoting Score	78	0.41	0.45	0.54	0.75
Peer Frequency	100	**0.54**	**0.57**	**0.63**	**0.80**
Ensemble	100	**0.60**	**0.61**	**0.67**	**0.82**

entities. In Wikidata, some of those qualifiers are *point in time* (P585), *start time* (P580), and *end time* (P582). A few samples are shown in Table 7.

- **Time-based Properties (TP)**: Temporal properties are properties like *follows* (P155) and *followed by* (P156) indicating a chain of entities, ordered from oldest to newest, or from newest to oldest. For example, `1st Academy Awards; followed by; .. 54th Academy Awards; followed by; 55th Academy Awards; ..`

We create TQ groups from aggregating information about people sharing the same statements. For example, `position held; President of the U.S.` is one TQ group, where members will have a *start time* for this position, as well as an *end time*. In case of absence of an *end time*, this implies that the statement holds to this day (`Donald Trump`'s statement in Table 7). In other words, we aggregate entities sharing the same predicate-object pair, which will be treated as the peer group's title, and rank them in ascending order of time qualifiers. For the *point in time* qualifier, we simply rank the dates from oldest to newest, and for the *start/end date*, we rank the end date from oldest to newest. If the *end date* is missing, the entity will be moved to the newest slot.

We collect a total of 19.6K TQ groups (13.6K using the *start/end date* qualifier and 6K using the *point in time* qualifier). Based on a manual analysis of a random sample of 100 groups of different sizes, we only consider time series with at least 10 entities[8]. We create TP groups by first collecting all entities reachable by one of the transitive properties, *follows* (P155) and *followed by* (P156). Considering each of the collected entities as a source entity, we compute the longest possible path of entities with only transitive properties. This path consists an ordered set of peers. To avoid the problem of double-branching (one entity followed by two entities), we consider the two directions separately. Again, one path will be chosen at the end; the one with maximum length. The total number of TP groups is 19.7K groups. We limit the size of the groups to at least 10 and at most 150[9].

Setup and Baseline. We choose 100 entities, that belongs to at least one ordered set of peers, from Wikidata: 50 people and 50 literature works. We collect top-5 negative statements for each of those entities (for people, we consider TQ groups, and for literature works, TP groups). We made this choice because of the lack of entities of type person with transitive properties. In case an entity belongs to several groups, we merge all the results it is receiving from different groups, rank them, and retrieve the top-5 statements. Similarly, as a baseline, using the peer-based inference method of Section 4, instantiated with cosine similarity on Wikipedia embeddings [50] as similarity function, we collect top-5 negative statements for the same entities. We end up with 1K statements, 500 inferred by each model.

Correctness Evaluation. We randomly retrieve 400 negative statements from the 1K statements collected above, 200 from each model (100 people, and 100 literature works). We then assess the correctness of each method, using crowdsourcing, we show each statement to 3 annotators, asking

[8]This variable can be easily adjusted depending on the preference of the developers and/or the purpose of the application.

[9]We do not truncate the groups, we simply disregard any group smaller or larger than the thresholds.

Table 6: Top-3 results for *Albert Einstein* using 3 ranking metrics.

Random rank	**Property frequency**	**Ensemble**
¬(instagram; _)	¬(doctoral student; _)	¬(occup.; astrophysicist)
¬(child; Tarek Sharif)	¬(candidacy in election; _)	¬(party; Communist Party USA)
¬(award; BAFTA)	¬(noble title; _)	¬(doctoral student; _)

Table 7: Samples of temporal information in Wikidata.

Statement	**Time-based qualifier(s)**
(Barack Obama; position held; U.S. senator)	*start time*: 3 January 2005; *end time*: 16 November 2008
(Maya Angelou; award received; Presidential Medal of Freedom)	*point in time*: 2010
(Donald Trump; spouse; Melania Trump)	*start time*: 22 January 2005

Table 8: Correctness of order-oriented and peer-based methods.

	People	**Literature Work**
Peer-based inference		
	%	%
Correct	81	88
Incorrect	18	12
Ambiguous	1	0
Order-oriented inference		
	%	%
Correct	**91**	**91**
Incorrect	9	7
Ambiguous	0	2

them to choose whether this statement is correct, incorrect, or ambiguous. Results are shown in Table 8. Our order-oriented inference method clearly infer less incorrect statements by 9 percentage points for people, and 5 for literature works. It also provide more correct statements for people by 10 percentage points, and literature work by 3. The percentage of queries with full agreement in this task is 37%. Also, annotations shows a standard deviation of 0.17.

Subject Coverage. To assess the subject coverage of the order-oriented method, we randomly sample 1K entities from each dataset, and test whether it is a member of at least one ordered set, thus the ability to infer useful negative statements about it. For TQ groups, we randomly sample 1K people, which results in a coverage of 54%. And for TP groups, we randomly sample 1K literature works, and also receive a coverage of 54%.

Usefulness. To assess the quality of our inferred statements from the order-oriented inference method against the baseline (the peer-based inference method), we present to the annotators two sets of top-5 negative statements about a given entity, and ask them to choose the more interesting set. The total number of opinions collected, given 2 models, 100 entities, 3 annotations each, is 600. To avoid biases, we repeatedly switch the position of the sets. Results are shown in Table 10. Overall results show that our method is preferred more by 10% of the entities for both domains. The standard deviation of this task is 0.24 and the percentage of queries with full agreement is 18%.

Evaluation of Verbalizations. One main contribution that our order-oriented inference method offers are *verbalizations* produced with every inferred negative statement. In other words, it can, unlike the peer-based inference method, provide more concrete explanations of the usefulness of the inferred negations. For example, the inferred negative statement ¬`(Abraham Lincoln; cause of death; natural causes)` was inferred by both of our methods. However, each method offers a different verbalization. For the peer-based method, the verbalization is "unlike 10 of 30 similar people", and for the order-oriented method is "unlike 12 of the previous 12 presidents of the U.S.". To assess the quality of the verbalizations more formally, we conduct a crowdsourcing task with 100 useful negations that were inferred by both methods from our previous experiment. For every negative statement, the annotator will see two different verbalizations on "why is this negative statement noteworthy". We ask the annotator to choose the better verbalization, she can choose Verbalization1, Verbalization2, or Either/Neither. Results show that verbalizations provided by our order-oriented inference method were chosen 76% of the time, by the peer-based inference method 23% of the time, and the either or neither option only 1% of the time. The standard deviation is 0.23, and the percentage of queries with full agreement is 20%. Table 9 shows a number of examples, using different grouping functions for the peer-based method.

7.3. Conditional Negative Statements Evaluation

We evaluate our lifting technique to retrieve useful conditional negative statements, based on three criteria: (i) compression, (ii) correctness, and (iii) usefulness. We collect the top-200 negative statements about 100 entities (people, organizations, and art work), and then apply lifting on them.

Table 9: Negative statements and their verbalizations using peer-based and order-oriented methods.

Statement	Order-oriented *Unlike..*	Peer-based *Unlike..*	Peering
¬(Emmanuel Macron; member; National Assembly)	29 of 36 members of La République En Marche party	70 of 100 similar people	WP embed. [50]
¬(Tim Berners-Lee; citizenship; U.S.)	101 of previous 115 winners of the MacArthur Fellowship	53 of 100 sim. comp. scientists	Structured facets
¬(Michael Jordan; occupation; basketball coach)	27 of prev. 49 winners of the NBA All-Defensive Team	31 of 100 sim. people	WP embed. [50]
¬(Theresa May; position; Opposition Leader)	11 of prev. 14 Leaders of the Conservative Party	10 of 100 sim. people	WP embed. [50]
¬(Cristiano Ronaldo; citizenship; Brazil)	4 of prev. 7 winners of the Ballon d'Or	20 of 100 sim. football players	Structured facets

Table 10: Usefulness of order-oriented and peer-based methods.

	People	Literature Work
	%	%
Peer-based inference	42	44
Order-oriented inference	**52**	**54**
Both	6	2

Table 11: Usefulness of conditional negative statements.

Preferred	(%)
Conditional negative statements	**70**
Grounded and universally negative statements	25
Either or neither	5

Compression. On average, 200 statements are reduced to 33, which means that lifting compresses the result set by a factor of 6.

Correctness. We ask the crowd to assess the correctness of 100 conditional negative statements (3 annotations per statement), chosen randomly. To make it easier for annotators who are unfamiliar with RDF triples[10], we manually convert them into natural language statements, for example "*Bing Crosby did not play any keyboard instruments*". Results show that 57% were correct, 23% incorrect, and 20% were uncertain. The standard deviation of this task is 0.24 and the percentage of queries with full agreement is 18%.

Usefulness. For every entity, we show 3 annotators 2 sets of top-3 negative statements: a grounded and universally negative statements set and a conditional negative statement set, and ask them to choose the one with more interesting information. Results are shown in Table 11. The conditional statements were chosen 45 percentage points more than the grounded and universally negative statements. The standard deviation of this task is 0.22 and the percentage of queries with full agreement is 21%.

An example is shown in Table 12, with entity $e =$*Leonardo Dicaprio*, and property *occupation*. Even though he is one of the most accomplished actors in the world, unlike many of his peers, he never attempted directing any kind of creative work (films, plays, television shows, etc..).

[10]Especially because of the triple-pattern condition.

7.4. Extrinsic Evaluation

We next highlight the relevance of negative statements for:

- Entity summarization on Wikidata [46].
- Decision support with hotel data from Booking.com.
- Question answering on various structured search engines.

Entity Summarization. In this experiment we analyze whether mixed positive-negative statement set can compete with standard positive-only statement sets in the task of entity summarization. In particular, we want to show that the addition of negative statements will *increase the descriptive power* of structured summaries.

We collect 100 Wikidata entities from 3 diverse types: 40 people, 30 organizations (including publishers, financial institutions, academic institutions, cultural centers, businesses, and more), and 30 literary works (including creative work like poems, songs, novels, religious texts, theses, book reviews, and more). On top of the negative statements that we infer, we collect relevant positive statements about those entities.[11] We then compute for each entity e a sample of 10 positive-only statements, and a mixed set of 7 positive and 3 *correct*[12] negative statements, produced by the peer-based method. We rely on peering using Wikipedia embeddings [50]. Annotators were then asked to decide which set contains more new or unexpected information about e. More particularly, for every entity, we ask workers to assess the sets (flipping the position of our set to avoid biases), leading to a total number of 100 tasks for 100 entities. We collect 3 opinions per task. Overall results show that mixed sets with negative information were preferred for 72% of the entities, sets with positive-only statements were preferred for 17% of the entities, and the option "both or neither" was chosen for 11% of the entities. Table 14 shows results per each considered type. The standard deviation is 0.24, and the percentage of queries with full agreement is 22%. Table 13 shows three diverse examples. The first one is *Daily Mirror*. One particular noteworthy negative statement in this case is that the newspaper is not owned by the "*News U.K.*" publisher which owns a number of of *British* newspapers like

[11]We define a number of common/useful properties to each of type, e.g., for people, "position held"is a relevant property for positive statements.

[12]We manually check the correctness of these negative statements.

Table 12: Negative statements about *Leonardo Dicaprio*, before and after lifting.

Negative statements	**Conditional negative statements**
¬(occupation; film director) ¬(occupation; theater director) ¬(occupation; television director)	¬ occupation - (*subclass of; director*)

The Times, The Sunday Times, and The Sun. The second entity is *Peter the Great* who died in *Saint Petersburg* and not *Moscow*, and who did not receive the *Order of St Alexander Nevsky* which was first established by his wife, a few months after his death. And the third entity is *Twist and Shout*. Although it is a known song by *The Beatles*, however, they were *not* its composers, writers, or original performers.

Decision Support. Negative statements are highly important also in specific domains. In online shopping, characteristics not possessed by a product, such as the *IPhone 7* not having a headphone jack, are a frequent topic highly relevant for decision making. The same applies to the hospitality domain: the absence of features such as free WiFi or gym rooms are important criteria for hotel bookers, although portals like Booking.com currently only show (sometimes overwhelming) positive feature sets.

To illustrate this, based on a comparison of 1.8K hotels in India, as per their listing on Booking.com, using the peer-based method, we infer useful negative features. For peering, we consider all other hotels in India, and for ranking, we compute peer frequencies (*PEER*). We then use crowdsourcing over the results of 100 hotels. We ask annotators to check two sets of features about a given hotel, one set containing 5 random positive-only features, and one set containing a mix of 3 positive and 2 negative features. Their task was to choose which set of features will help them more in deciding whether to say in this hotel or not. They can choose one of the sets, or both. For every hotel, we request 3 annotators.

Table 15 show that sets with negative features were chosen 16 percentage points more than the positive-only sets. The standard deviation of this task is 0.22 and the percentage of queries with full agreement is 28%. Table 16 shows three hotels with useful negative features. Although the *Hotel Asia The Dawn* lists 64 positive features, negative information such as that it does not offer air conditioning and free Wifi may provide important cues for decision making.

Moreover, we collect 20 pairs of hotels from the same dataset, and show every pair's Booking.com pages to 3 annotators. We ask them to choose the better hotel for them. Then we show them negative features about the pair, and ask them whether this new information would change their mind on their initial decision. A screenshot of the task is shown in Figure 2. 42% changed their pick after negative features were revealed. The standard deviation on this task is 0.15 and the full agreement of the 3 annotators is 35%, where all annotators changed their choice (or did not) after negative features were revealed, regardless of the hotel chosen before and after, and a full agreement of 30%, where all annotators chose the same hotel at the end.

Question Answering. In this experiment we compare the results to negative questions over a diverse set of sources. We manually compiled 20 questions that involve negation, such as *"Actors without Oscars"*[13]. We compare them over a four highly diverse sources: Google Web Search (increasingly returning structured answers from the Google knowledge graph [42]), WDAqua [14] (an academic state-of-the-art KBQA system), the Wikidata SPARQL endpoint (direct access to structured data), and our peer-based method. For Google Web Search and WDAqua, we submit the queries in their textual form, and consider answers from Google if they come as structured knowledge panels. For Wikidata and peer-based inference, we transform the queries into SPARQL queries[14], which we either fully execute over the Wikidata endpoint, or execute the positive part over the Wikidata endpoint, while evaluating the negative part over a dataset produced by our peer-based inference method. For each method, we then self-evaluate the number of results, the correctness and relevance of the (top-5) results. All methods are able to return highly correct statements, yet Google Web Search and WDAqua provide no answers to answer 18 and 16 of the queries at all.

We continue the assessment over a sample of 5 queries. Wikidata SPARQL returns by far the highest number of results, 250K on average, yet does not perform ranking, thus returns results that are hardly relevant (e.g., a local Latvian actor to the Oscar question). The peer-based inference outperforms it by far in terms of relevance (72% vs. 44% for Wikidata SPARQL), and we point out that although Wikidata SPARQL results appear highly correct, this has no formal foundation, due to the absence of a stance of OWA KBs towards negative knowledge.

8. Resources

Negative Statement Datasets for Wikidata. We publish the first datasets that contain dedicated *useful* negative statements about entities in Wikidata: (i) Peer-based and order-oriented inference data: 7.9M negative

[13]Sample textual queries: "actors with no Oscars", "actors with no spouses", "film actors who are not film directors", "football players with no Ballon d'Or", "politicians who are not lawyers".

[14]SPARQL queries: w.wiki/A6r, w.wiki/9yk, w.wiki/9yn, w.wiki/9yp, w.wiki/9yq

Table 13: Results for the entities *Daily Mirror*, *Peter the Great*, and *Twist and Shout*.

Daily Mirror	
Pos-only	**Pos-and-neg**
(owned by; Reach plc)	¬***(newspaper format; broadsheet)***
(newspaper format; tabloid)	(newspaper format; tabloid)
(country; United Kingdom)	¬***(country; U.S.)***
(language of work or name; English)	(language of work or name; English)
(instance of; newspaper)	¬***(owned by; News U.K.)***
...	...
Peter the Great	
Pos-only	**Pos-and-neg**
(military rank; general officer)	(military rank; general officer)
(owner of; Kadriorg Palace)	(owner of; Kadriorg Palace)
(award; Order of the Elephant)	¬***(place of death; Moscow)***
(award; Order of St. Andrew)	(award; Order of St. Andrew)
(father; Alexis of Russia)	¬***(award; Knight of the Order of St. Alexander Nevsky)***
...	...
Twist And Shout	
Pos-only	**Pos-and-neg**
(composer; Phil Medley)	¬***(composer; Paul McCartney)***
(performer; The Beatles)	(performer; The Beatles)
(producer; George Martin)	¬***(composer; John Lennon)***
(instance of; musical composition)	(instance of; musical composition)
(lyrics by; Phil Medley)	¬***(lyrics by; Paul McCartney)***
...	...

Table 14: Positive-only vs. pos-and-neg statements.

Preferred Choice	**Person (%)**	**Organization (%)**	**Literary work (%)**
Pos-and-neg	**71**	**77**	**66**
Pos-only	22	10	17
Both or neither	7	13	17

Figure 2: Extrinsic use-case: decision support on hotel data.

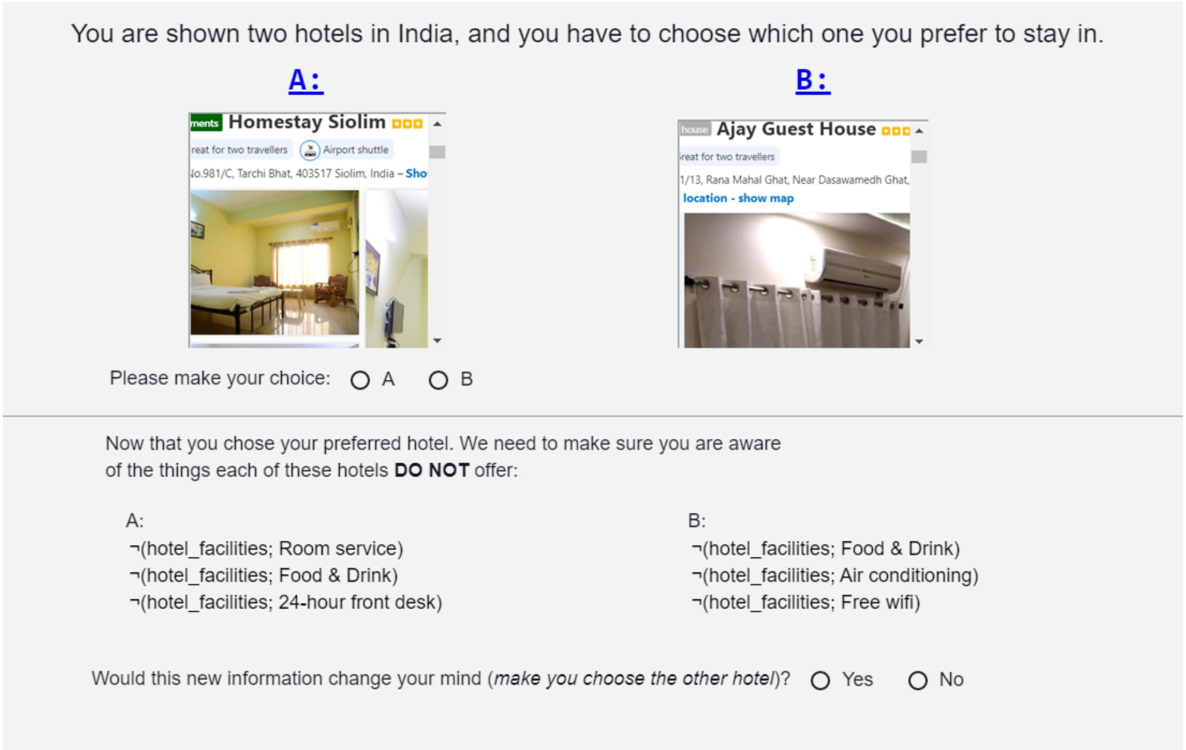


Table 15: Usefulness of hotel features.

Preferred Choice	(%)
Pos-and-neg	**54**
Pos-only	38
Either or neither	8

statements about popular 600K entities from various types, (ii) release the mturk-annotated on the correctness of 1K negative statements of Section 7.1, and (iii) 40K ordered set of peers introduced in Section 7.2.

Open-source Code. We make our peer-based inference method available for users to try it on their own datasets.

Demo. We are currently developing a web-based browsing interface for useful negative statements about Wikidata entities.[15] A screenshot of the homepage is shown in Figure 3.

All the material related to this work can be found on our webpage.[16]

9. Conclusion & Future Directions

This article has made the first comprehensive case for explicitly materializing useful negative statements in KBs. We have introduced a statistical inference approach on retrieving and ranking candidate negative statements, based on expectations set by highly related peers. We have also released a number of resources to encourage further research.

In future work we would like to explore a number of research directions:

1. Missing vs negative statements: How to allow maximal trade-ability between fewer highly correct statements, and larger sets of interesting negation candidates.

2. Mining more complex negations. Our focus was on simple - grounded and universal negation, with a hint at more complex conditional statements, but it is open to extend that to automatically finding aspects, further joins *"did not study at a university which was graduating any Nobel prize winner"*, negation on sets of entities instead of entity-centric *"no African country has hosted any Olympic games"*, etc.

3. Exploring textual information extraction for implicit negations, like *"Theresa May is an only child."* can be expressed using the KB statement ¬(`sibling; _`), and *"Angela Merkel is childless"* as ¬(`child; _`).

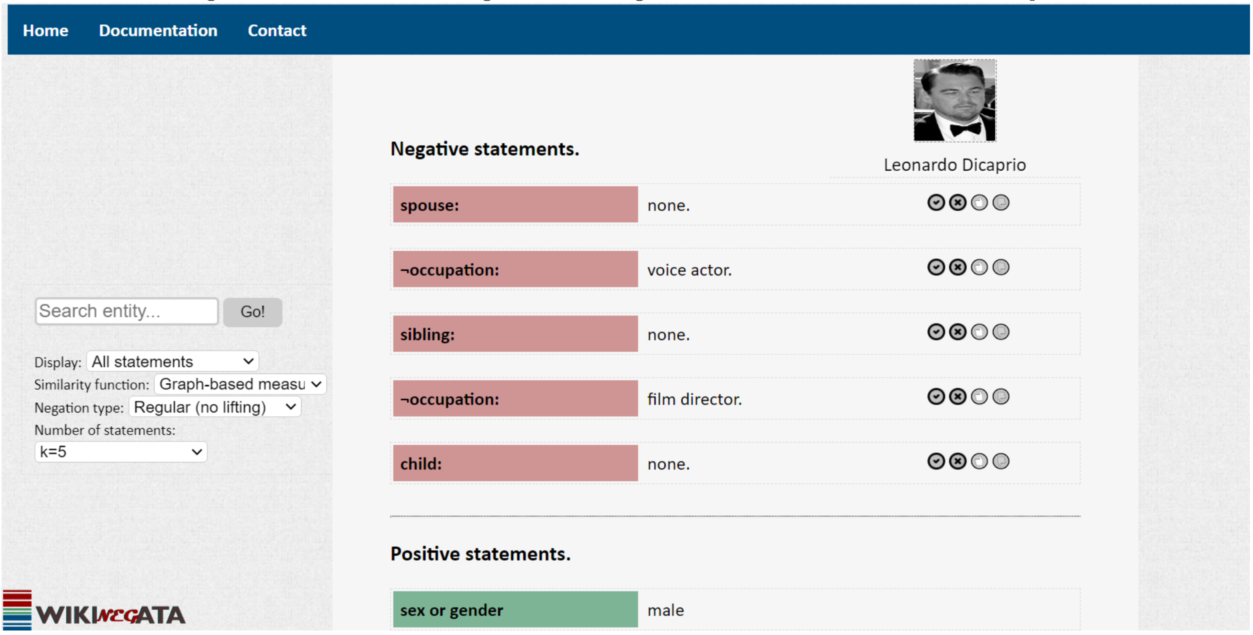
[15] A demonstrative video is shown at `https://www.mpi-inf.mpg.de/fileadmin/inf/d5/research/negation_in_KBs/wikinegata_overview.mp4`

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

Table 16: Negative statements for hotels in India.

Hotel	Number of positive features	Top-3 negative features
The Sultan Resort	106	¬ Parking; ¬ Fan; ¬ Newspapers
Vista Rooms at Mount Road	28	¬ Room service; ¬ Food & Drink; ¬ 24-hour front desk
Hotel Asia The Dawn	64	¬ Air conditioning; ¬ Free Wifi; ¬ Free private parking

Figure 3: Interface for Wikinegata - useful negative statements about *Leonardo DiCaprio*.



References

[1] Arnaout, H., Elbassuoni, S.: Effective searching of RDF knowledge graphs. JWS (2018)

[2] Arnaout, H., Razniewski, S., Weikum, G.: Enriching knowledge bases with interesting negative statements. In: AKBC (2020)

[3] Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z., et al.: DBpedia: A nucleus for a web of open data. In: ISWC (2007)

[4] Baader, F., Calvanese, D., Mcguinness, D., Nardi, D., F. Patel-Schneider, P.: The Description Logic Handbook. Cambridge University Press (2007)

[5] Balaraman, V., Razniewski, S., Nutt, W.: Recoin: Relative completeness in Wikidata. In: Wiki Workshop at WWW (2018)

[6] Bast, H., Buchhold, B., Haussmann, E.: Relevance scores for triples from type-like relations. In: SIGIR (2015)

[7] Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: NIPS (2013)

[8] Bărbântan, I., Potolea, R.: Exploiting word meaning for negation identification in electronic health records. In: AQTR (2014)

[9] Bărbântan, I., Potolea, R.: Towards knowledge extraction from electronic health records - automatic negation identification. In: IFMBE (2014)

[10] Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Tractable reasoning and efficient query answering in description logics: The DL-Lite family. Journal of Automated Reasoning (2007)

[11] Chapman, W., Bridewell, W., Hanbury, P., Cooper, G., Buchanan, B.: A simple algorithm for identifying negated findings and diseases in discharge summaries. Journal of Biomedical Informatics (2001)

[12] Chapman, W., Hillert, D., Velupillai, S., Kvist, M., Skeppstedt, M., Chapman, B., Conway, M., Tharp, M., Mowery, D., Deleger, L.: Extending the NegEx lexicon for multiple languages. Studies in health technology and informatics (2013)

[13] Cruz Díaz, N.P.: Detecting negated and uncertain information in biomedical and review texts. In: RANLP (2013)

[14] Diefenbach, D., Singh, K., Maret, P.: WDAqua-core0: A Question Answering Component for the Research Community (2017)

[15] Ernst, P., Siu, A., Weikum, G.: Knowlife: a versatile approach for constructing a large knowledge graph for biomedical sciences. BMC bioinformatics (2015)

[16] Erxleben, F., Günther, M., Krötzsch, M., Mendez, J., , Vrandečić, D.: Introducing Wikidata to the linked data web. In: ISWC (2014)

[17] Fader, A., L., Z., O., E.: Open question answering over curated and extracted knowledge bases. In: KDD (2014)

[18] Flouris, G., Huang, Z., Pan, J.Z., Plexousakis, D., Wache, H.: Inconsistencies, negations and changes in ontologies. In: AAAI (2006)

[19] Galárraga, L., Razniewski, S., Amarilli, A., Suchanek, F.M.: Predicting completeness in knowledge bases. In: WSDM (2017)

[20] Galárraga, L., Teflioudi, C., Hose, K., Suchanek, F.M.: Fast rule mining in ontological knowledge bases with AMIE+. VLDB (2015)

[21] Ghosh, S., Razniewski, S., Weikum, G.: Uncovering hidden semantics of set information in knowledge bases. JWS (2020)

[22] Goldin, I., Chapman, W.: Learning to detect negation with "Not" in medical texts. In: SIGIR (2003)

[23] Ho, V., Stepanova, D., Gad-Elrab, M., Kharlamov, E., Weikum, G.: Rule learning from knowledge graphs guided by embedding models. In: ISWC (2018)

[24] Huang, S., Liu, J., Korn, F., Wang, X., Wu, Y., Markowitz, D., Yu, C.: Contextual fact ranking and its applications in table synthesis and compression. In: KDD (2019)

[25] Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of IR techniques. Trans. Inf. Syst. (2002)

[26] Karagiannis, G., Trummer, I., Jo, S., Khandelwal, S., Wang, X., Yu, C.: Mining an "anti-knowledge base" from Wikipedia updates with applications to fact checking and beyond. PVLDB (2019)
[27] Kasneci, G., Suchanek, F.M., Ifrim, G., Ramanath, M., Weikum, G.: NAGA: Searching and ranking knowledge. In: ICDE (2008)
[28] Lajus, J., Galárraga, L., Suchanek, F.: Fast and exact rule mining with AMIE 3. In: ESWC (2020)
[29] Malyshev, S., Krötzsch, M., González, L., Gonsior, L., Bielefeldt, A.: Getting the most out of Wikidata: Semantic technology usage in Wikipedia's knowledge graph. In: ISWC (2018)
[30] McGuinness, D., Van Harmelen, F., et al.: OWL web ontology language overview. W3C recommendation (2004)
[31] Minker, J.: On indefinite databases and the closed world assumption. In: 6th Conference on Automated Deduction (1982)
[32] Morante, R., Sporleder, C.: Modality and negation: An introduction to the special issue. Comput. Linguist. (2012)
[33] Nickel, M., Rosasco, L., Poggio, T.: Holographic embeddings of knowledge graphs. In: AAAI (2016)
[34] Oren, E., Delbru, R., Decker, S.: Extending faceted navigation for RDF data. In: ISWC (2006)
[35] Ortona, S., Meduri, V., Papotti, P.: RuDiK: rule discovery in knowledge bases. VLDB (2018)
[36] Pan, J.Z., Calvanese, D., Eiter, T., Horrocks, I., Kifer, M., Lin, F., (Eds), Y.Z.: Reasoning Web: Logical Foundation of Knowledge Graph Construction and Query Answering. Springer (2017)
[37] Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: EMNLP (2014)
[38] Ponza, M., Ferragina, P., Chakrabarti, S.: A two-stage framework for computing entity relatedness in Wikipedia. In: CIKM (2017)
[39] Razniewski, S., Balaraman, V., Nutt, W.: Doctoral advisor or medical condition: Towards entity-specific rankings of knowledge base properties. In: ADMA (2017)
[40] Razniewski, S., Nutt, W.: Completeness of queries over incomplete databases. VLDB (2011)
[41] Reiter, R.: On Closed World Data Bases (1978)
[42] Singhal, A.: Introducing the knowledge graph: things, not strings. `https://www.blog.google/products/search/introducing-knowledge-graph-things-not` (2012)
[43] Suchanek, F., Kasneci, G., , Weikum, G.: Yago: A core of semantic knowledge. In: WWW (2007)
[44] Szarvas, G., Vincze, V., Farkas, R., Csirik, J.: The BioScope corpus: Annotation for negation, uncertainty and their scope in biomedical texts. BioNLP (2008)
[45] Tanon, T., Suchanek, F.: Querying the edit history of Wikidata. In: ESWC (2019)
[46] Vrandečić, D., Krötzsch, M.: Wikidata: A free collaborative knowledge base. CACM (2014)
[47] Wang, Q., Mao, Z., Wang, B., Guo, L.: Knowledge graph embedding: A survey of approaches and applications. IEEE TKDE (2017)
[48] Wu, S., Miller, T., Masanz, J., Coarr, M., Halgrim, S., Carrell, D., Clark, C.: Negation's not solved: generalizability versus optimizability in clinical natural language processing. PloS one (2014)
[49] Yahya, M., Barbosa, D., Berberich, K., Wang, Q., Weikum, G.: Relationship queries on extended knowledge graphs. In: WSDM (2016)
[50] Yamada, I., Asai, A., Shindo, H., Takeda, H., Takefuji, Y.: Wikipedia2Vec: An optimized tool for learning embeddings of words and entities from wikipedia. arXiv (2018)
[51] Yang, Y., Yih, W., Meek, C.: WikiQA: A challenge dataset for open-domain question answering. In: EMNLP (2015)