Knowledge Verification for Long-Tail Verticals

Furong Li[†] Xin Luna Dong[‡] Anno Langen[§] Yang Li[§]
[†]National University of Singapore [‡]Amazon [§]Google Inc.
furongli@comp.nus.edu.sg lunadong@amazon.com {arl, ngli}@google.com

ABSTRACT

Collecting structured knowledge for real-world entities has become a critical task for many applications. A big gap between the knowledge in existing knowledge repositories and the knowledge in the real world is the knowledge on *tail* verticals (*i.e.*, less popular domains). Such knowledge, though not necessarily globally popular, can be personal hobbies to many people and thus collectively impactful. This paper studies the problem of *knowledge verification for tail verticals*; that is, deciding the correctness of a given triple.

Through comprehensive experimental study we answer the following questions. *1) Can we find evidence for tail knowledge from an extensive set of sources, including knowledge bases, the web, and query logs? 2) Can we judge correctness of the triples based on the collected evidence? 3) How can we further improve knowledge verification on tail verticals?* Our empirical study suggests a new knowledge-verification framework, which we call FACTY, that applies various kinds of evidence collection techniques followed by knowledge fusion. FACTY can verify 50% of the (correct) tail knowledge with a precision of 84%, and it significantly outperforms state-of-the-art methods. Detailed error analysis on the obtained results suggests future research directions.

1. INTRODUCTION

Collecting structured knowledge for real-world entities has become a critical task for many applications, such as semantic search, query answering and machine reading. Both academia and industry have spent considerable efforts on constructing large-scale knowledge bases (KBs), such as YAGO [35], NELL [7], Knowledge Vault [11], DeepDive [30], DBpedia [1], Probase [41], Google Knowledge Graph [20], and Microsoft Satori [34].

Knowledge is usually stored as (subject, predicate, object) *triples*, where each triple states a fact of some entity. To exemplify, a triple (Kobe Bryant, profession, basketball player) means that Kobe Bryant's profession is basketball player. Triples in knowledge bases are often organized into *verticals*, where each vertical describes a set of entities in the same domain sharing common attributes (*i.e.*, predicates). For instance, the vertical of *athletes* contains triples regarding different athletes, and describes each athlete by *profession, country, team*, and so on. Verticals may have hierarchy; for example, *basketball players* is a sub-vertical of *athletes*.

A big gap between the knowledge in existing knowledge bases and the knowledge in the real world is the knowledge on *tail* verticals. Roughly speaking, a vertical is a tail vertical if its subject entities are not globally popular (for example, in terms of search query volume); the number of triples in a tail vertical is usually not huge (below millions). In contrast to head (popular) verticals such as *music, movies* and *celebrities*, examples of tail verticals include *gym exercises, yoga poses, cheese varieties*, and *tomato varieties*. Although existing knowledge bases contain billions of triples, their information on tail verticals is still limited. For instance, we found that in *Freebase* [4] about 40% entities have no *factual* triples (*i.e.*, triples that state some properties of an entity), but only triples about their names, types, and descriptions; the majority of such entities belong to some tail verticals. As another example, we manually collected triples for the four aforementioned tail verticals from up to three manually selected authoritative sources, and observed that in total only about 150 triples exist in *Freebase*, and the coverage of their subject entities is below 10%. Although a tail vertical may not be popular by itself, given the large number of tail verticals, they can be collectively impactful.

Collecting knowledge for tail verticals is hard. On the one hand, there can be millions of tail verticals and their attributes are highly diverse, so manual curation cannot scale. On the other hand, automatic extractions fall short both because we lack good training data, and because reconciliation (*i.e.*, deciding if two mentions refer to the same entity) on tail entities can be error-prone. We thus tried a different approach: we identified a set of tail verticals and a few data sources for each vertical[1], and then asked the crowd to extract triples from these given sources through annotation tools and hand-crafted patterns [9]. Although the results are much cleaner than those from automatic extraction systems, there can still be remnant extraction errors [14] and imprecise information from the sources. Thus, it is critical to verify the correctness of the collected knowledge before populating knowledge bases.

There exist two approaches for knowledge verification. First, one can search the subject and object of a triple on the web, and then apply a classifier to decide if the triple is true based on the search results [25]. However, this approach obtains poor results on tail verticals: we can verify only 19% of the tail knowledge with a precision of 22% (*i.e.*, for every 100 triples that we verified as true, only 22 are actually true) in our experiments. This is because tail knowledge is not globally popular on the web and search results can be very noisy. Another solution is to apply supervised knowledge extraction [15] on the web, and consider a triple as verified if it

[1]Vertical discovery and source selection are very important problems but are out of the scope of this paper.

Proceedings of the VLDB Endowment, Vol. 10, No. 11
Copyright 2017 VLDB Endowment 2150-8097/17/07.

can be extracted. Unfortunately, this solution usually leads to a low recall on tail verticals because it cannot extract any triple whose subject or object is unknown to existing knowledge bases.

In this paper we investigate a third approach that first leverages both search-based and extraction-based techniques to find supporting evidence for each triple, and subsequently predicates the correctness of each triple based on the evidence. Our investigation tries to answer the following questions:

- How can we find evidence for the tail triples, and what are the sources that we can use?
- Can we judge the correctness of the triples based on the collected evidence?
- How can we further improve knowledge verification on tail verticals?

This paper makes four contributions. First, we explored an extensive set of sources to collect supporting evidence for a triple. We start with existing knowledge bases, which provide highly reliable information but with limited coverage on tail verticals. We then expand our search space to the web, which has a much higher coverage but can be noisy. Further, we enrich the evidence by analysing search query logs, which reflect users' perspectives on the world. In total we tried seven approaches to extract evidence from these sources. Overall we found evidence for 60% of the correct triples on over 96% of the entities in the tail verticals we examined. However, there are evidence for wrong triples too. We provide a detailed study to compare various sources and approaches that we used.

Second, we investigate how knowledge fusion [12] can be applied to distinguish correct triples from wrong ones based on the collected evidence. Knowledge fusion [12, 13] is recently invented to decide the correctness of an extracted triple based on the data sources that provide the triple and the extractors that obtain the triple. We tried both single-truth methods [10, 12], and multi-truth methods [32, 45]; the former assume that there is only one true value for an entity, while the latter allow the existence of multiple true values (*e.g.*, a book can have multiple authors). Our experiments show that single-truth methods usually give the highest precision, and multi-truth methods usually lead to the highest recall. We then propose a hybrid approach that combines the strengths of them and thus balances the precision and recall.

Third, this paper is the first to propose an end-to-end knowledge verification framework that performs evidence collection followed by knowledge fusion. Our framework, which we call FACTY, can verify 50% of the (correct) tail knowledge with a precision of 84%, significantly better than existing approaches.

Finally, we conducted a detailed error analysis on the obtained results, and suggest several directions for improving both knowledge verification and knowledge curation in general.

The rest of the paper is organized as follows. Section 2 defines the problem and describes our experiment datasets. Section 3 describes how we collect evidence from different sources. Section 4 studies how knowledge fusion can be applied to decide the correctness of a triple. Section 5 presents results obtained by knowledge fusion. Section 6 compares our framework with existing knowledge-verification systems, and Section 7 discusses future directions. Section 8 reviews related work, and Section 9 concludes the paper.

2. PROBLEM DEFINITION

Triple, vertical. A *triple* is in the form of (subject, predicate, object), where subject is an entity, predicate is an attribute of the entity, and object is a value for the attribute of the entity. An object may be an entity, an arbitrary string, a number, and so on. A subject entity could have several values for an attribute, and there is a triple for each value. For example, Table 1 has three triples (t_3-t_5) regarding *ice hockey equipments*, each for an equipment. We consider factual triples and say a triple is *true* if it conforms to the real world; for example, (ice hockey, equipment, stick) is true, while (ice hockey, equipment, board) is false. If a (subject, predicate) pair has only one true triple (*e.g.*, date-of-birth), we call the case *single-truth*; otherwise, we call it *multi-truth*.

Table 1: Sample triples for the vertical *Winter sports*.

	subject	predicate	object
t_1	skiing	equipment	boots
t_2	snowboarding	equipment	board
t_3	ice hockey	equipment	helmet
t_4	ice hockey	equipment	stick
t_5	ice hockey	equipment	neck guard
t_6	ice hockey	venue	hockey rink
t_7	skiing	venue	outdoor

A *vertical* is a collection of triples whose subjects are entities in the same domain and have a set of predicates in common. Table 1 exemplifies a small set of triples in the vertical *Winter sports*; it contains seven triples for three winter sports on two predicates.

We can now formally define the problem we study in this paper.

DEFINITION 2.1 (KNOWLEDGE VERIFICATION). *Given a set $\mathcal{T}$ of triples in a vertical,* knowledge verification *decides if each triple in $\mathcal{T}$ is true.* □

Experiment dataset. We experimented on four verticals: *Cheese varieties, Tomato varieties, Gym exercises* and *Yoga poses*. We chose these four verticals because they represent verticals with different characteristics. For each vertical, we manually collected triples from up to three carefully selected authoritative and comprehensive sources, and kept those for which we can manually validate the correctness as true triples.

Then for experimental purpose, we generate false triples as follows. Given a true triple, we consider triples that share the same subject and predicate, but have different objects as its *alternatives*; for instance, (ice hockey, venue, outdoor) is an alternative of the triple t_6 in Table 1. All alternatives that are not contained in our input are considered false; this is known as the Local Closed-World Assumption (LCWA) and is commonly used for generating negative examples for knowledge extraction and fusion [11]. More precisely, given a set $\mathcal{T}$ of true triples, for each triple $t = (s, p, o) \in \mathcal{T}$, let $\mathcal{O}$ be the set of objects associated with p. We generate a false triple $t' = (s, p, o')$ for each $o' \in \mathcal{O}$ if $t' \notin \mathcal{T}$. In our experiments, for a true triple, we generate at most 20 false triples by considering only popular objects for the predicate. Then the input of the knowledge-verification task contains both the true triples and the false triples. We chose the top-20 popular objects because they typically occur more often on the web and thus increased the hardness of the problem. We also note that we found very few false negatives since the data sources we selected have high coverage for these domains.

Table 2 shows the statistics of the verticals, where each vertical contains hundreds of entities and thousands of true triples. The number of predicates ranges from 7 to 17. The ratio between false triples to true triples ranges from 5 to 14. In the first three verticals, the majority of the (subject, predicate) pairs have a single object as the true value, whereas in the *Yoga* vertical, most (subject, predicate) pairs have multiple truths.

3. COLLECTING EVIDENCE

We first try a simple approach for knowledge verification: consider any triple as correct if we can find some kind of evidence that supports the triple. In this section we answer the following

Table 2: Statistics for each tail vertical.

Vertical	#entities	#preds	#true triples	#false triples	%multi-truth
Cheese	420	17	4,753	68,480	2.3%
Tomato	574	14	5,464	64,935	3.7%
Gym	931	7	7,114	48,348	7.4%
Yoga	123	10	1,826	9,759	59.0%

questions: *1) Is this approach adequate? 2) How many true triples can we find evidence for (*i.e.*, recall)? 3) Will we also find evidence for false triples (*i.e.*, precision)? 4) Which sources contain rich evidence for tail knowledge and which extraction methods are effective?* We explain where and how we collect evidence in Sections 3.1 and 3.2, and present results in Section 3.3.

3.1 Sources for evidence collection

We first define the concept of evidence. *Evidence* of a triple is a piece of information that supports the triple. Evidence can be a triple in a knowledge base, a sentence in a document, a row in a web table, and so on. For instance, *"Besides ice skates and sticks, hockey players are usually equipped with..."* from *Wikipedia* is considered evidence for the triple (ice hockey, equipment, ice skates).

We consider three types of sources to find evidence for triples. First, we consider existing *knowledge bases*. As data in knowledge bases are structured and often of high quality, they should provide fairly reliable evidence. However, their coverage on tail knowledge can be low. Second, we consider the *web*, which contains rich data. However, the web data are often unstructured and thus make the evidence collection harder. There may also be errors on the web [13], reducing the trustworthiness of the evidence. Third, we consider *query logs*. As reported by previous research, 71% of search queries contain named entities [17]. The entities extracted from query logs can be different from those appear in web documents, as query logs model the world from user perspectives [18].

3.2 Techniques for evidence collection

Ideally, the evidence of a triple should mention the subject, predicate and object in some way. Usually one can recognize the mentions of subjects or objects through string matching or entity linking [19]. However, it is hard to identify the mentions of a predicate, because predicates often appear in distinct forms (*e.g.*, *birthday* vs. *was born on*), and sometimes even do not appear (*e.g.*, search queries may contain only subject and object). Explicitly looking for evidence containing a particular predicate usually leads to a low recall. Therefore, when collecting evidence, we only require subject and object matching, but relax on predicate. With no surprise, not requiring predicate matching causes errors; we compensate this by recording the matching information in pattern (we explain shortly) and leverage it in knowledge fusion (Section 5).

To facilitate the matching between entities, we conduct reconciliation [19] to map all entities to a unified repository (*Freebase* in particular). More precisely, if the subject/object of a triple exists in *Freebase*, we record its entity ID (known as *mid*); otherwise, the subject/object remains in its raw form.

For book-keeping purpose, we record the *provenance* of each piece of evidence. We write $\langle$url, system, pattern$\rangle$ as provenance, where url is the webpage on which the evidence is found, system is the system that finds the evidence, and pattern is the pattern used in evidence discovery (*e.g.*, a pattern for extracting sport equipments from texts can be *"$\langle$sport$\rangle$ players are equipped with $\langle$noun$\rangle$"*). Note that url may be set to null if the evidence is not from a webpage (*e.g.*, query log), and pattern can be null if we are not aware of the pattern used to discover the evidence. As we show later, such provenance information can help significantly in knowledge fusion (Sections 4-5)

We next describe how we find evidence from each type of data sources in detail.

3.2.1 Collecting evidence from knowledge bases

We consider two types of knowledge bases for evidence collection: manually curated KBs and automatically generated KBs.

Freebase: *Freebase* is a human curated knowledge base consisting of 50M entities and 1,500 verticals[2]. All subjects in the triples are entity IDs, and objects are either entity IDs or values from a fixed domain (*e.g.*, date, number). The predicates come from a predefined ontology.

Given an input triple $t = (s, p, o)$, we consider a *Freebase* triple (s', p', o') as evidence for t if $s = s'$ and $o = o'$ (recall that predicate matching is not required). Two subjects are the same if they are the same ID; two objects are the same if they are the same ID, date, or number (possibly in different formats). The provenance of this evidence is $\langle$null, Freebase, p'$\rangle$; url is set to null as it typically is not recorded in *Freebase*; pattern is set to p', meaning that we discover the evidence by considering triples with predicate p'.

KV [11]: Knowledge Vault (KV) is an automatically generated knowledge base containing 2.8B triples, among which 90M high-probability (≥ 0.7) triples are not in Freebase. A KV triple is in the same format as Freebase, and in addition has a probability indicating the likelihood of the triple being true, and a list of URLs where the triple is extracted.

We consider a *KV* triple $t'=(s', p', o')$ as evidence of $t = (s, p, o)$ if $s = s'$ and $o = o'$, and $Pr(t') \geq 0.7$ (0.7 is a threshold suggested in [11]). For each webpage url from which t' is extracted, we output a provenance $\langle$url, KV, p'$\rangle$.

3.2.2 Collecting evidence from the web

The web contains rich information in various formats, including web tables (structured), DOM trees (semi-structured), and texts (unstructured). We applied a wide spectrum of techniques to collect evidence from the web, ranging from sophisticated supervised learning to simple co-occurrences of entity names.

Web tables: Previous research [5] shows that web tables contain a vast amount of structured information about entities. Here we consider webpages where some keywords of the vertical (*e.g.*, *"winter sports"*) are mentioned. Then, following the techniques in [6], we extract from these webpages all instances under the *table* tag, and further identify relational tables in contrast to tables for layout (1.3% relational tables among all raw tables [5]).

Next we extract a set of triples from each relational table. We distinguish *vertical tables* and *horizontal tables* (the "vertical" here should not be confused with that in "tail verticals"). A vertical table (*e.g.*, Wikipedia infobox [40]) describes an entity in two columns, where the first column gives the attributes and the second column gives the values. We then generate a triple from each row, where the subject entity is identified from the table header or surrounding texts. A horizontal table often contains multiple entities, where each row describes an entity, each column represents an attribute, and each cell gives the value of the column attribute for the row entity. We train a classifier to decide the entity column, from which we extract subjects; we extract predicates from table header; and we generate a triple from each remaining cell. Whenever possible, we reconcile the subjects and objects to *Freebase mids*.

We consider an extracted triple (s', p', o') as evidence of an input triple (s, p, o) if they match on both subjects and objects. We say s' and s (respectively, o' and o) match if (1) their mids are identical, or (2) their Jaccard string similarity is above a threshold θ. The

[2] `https://developers.google.com/freebase/data`

predicates extracted from web tables are strings, so can have much higher variety than KB triples. Hence instead of tracking each predicate as a pattern, we compare p with p' and decide whether or not they match (string similarity above θ). The provenance of the evidence is $\langle$url, Webtables, pred-match/pred-unmatch$\rangle$, where url is the webpage that the evidence is extracted from.

Closed IE: We apply *closed information extraction* [15] techniques to extract triples from DOM trees and texts. The general idea is to learn a set of patterns from training data for each targeted relation (*i.e.*, predicate), and then apply the learnt patterns to extract new instances. We say this extraction method is *closed* since it is restricted to pre-defined predicates and vocabularies.

In particular, we take our true triples as training examples, and apply *distant supervision* [29] to learn extraction patterns for each predicate. We then use the learnt patterns to extract triples from the web. In addition to phrases, the patterns we learned include two types of information for the purpose of improving extraction precision. First, we apply Natural Language Processing (NLP) techniques to parse texts [8], and the patterns may contain sentence structure. Second, we annotate *Freebase* entities in web documents [19], and the patterns contain the types of annotated entities; as such, we cannot extract triples whose subjects or objects are unknown to *Freebase*.

We consider an extracted triple (s', p', o') as evidence of (s, p, o) if $s{=}s', p{=}p'$ and $o{=}o'$ (we require $p{=}p'$ here because triples extracted by Closed IE are guaranteed to have the same predicates as our input). For each webpage url and extraction pattern pattern, we have a provenance $\langle$url, ClosedIE, pattern$\rangle$ for the evidence.

Open IE: *Open IE* [2] systems can extract triples on any domain without specifying vocabulary, thereby called *open*. We apply Open IE on web texts; our extraction system works in a similar way as TEXTRUNNER [2] and REVERB [16]. Basically it leverages NLP annotations to identify a relation phrase (*e.g.*, *equipped with*) and a pair of entities from surrounding texts, and then generates a triple correspondingly. Both the relations and the entities can be arbitrary strings; but we reconcile them to *Freebase mids* whenever possible.

The way we identify an Open IE extraction as evidence of a triple is exactly the same as we do for web-table extractions. We do not repeat the details here.

Web co-occurrences: Given a triple $t = (s, p, o)$, we find the *co-occurrences* of the subject s and the object o on webpages as evidence for t. To allow for small variations, for each entity we (1) look up its aliases in *Freebase* if possible, and (2) use the n-grams of its name and aliases, where each n-gram contains n consecutive words in the string. To improve accuracy, (1) we allow a maximum number of 30 words between the two n-grams; (2) in case of free texts (instead of DOM trees), we further restrict the co-occurrence to the same sentence.

We consider the pair (s-n-gram, o-n-gram) as evidence of t if they co-occur in at least three webpages (so it is less likely to be random noise). For each webpage url, we output a provenance $\langle$url, Web co-occur, null$\rangle$; we do not require a predicate match here in order to improve recall, and we set extraction pattern to null.

3.2.3 Collecting evidence from query logs

We annotate each query in Google's search log with *Freebase* entities following the techniques in [17]. We consider a query Q as evidence of a triple $t = (s, p, o)$, if (1) s has a corresponding *Freebase mid*, and Q contains an annotation for this *mid*, and (2) Q contains the name, alias, or *Freebase* annotation of o. We do not check predicates here since they rarely occur in a query where both subjects and objects occur; we require the subject to match on *mid* to reduce noise.

Table 3: Summary of different evidence-collection approaches.

Source	Technique	Closed	Open	Co-occur
Knowledge bases	Freebase	√		
	KV	√		
Web	Web tables		√	
	Closed IE	√		
	Open IE		√	
	Web co-occur			√
Query log	Query logs			√

Table 4: Evidence collection results on each vertical.

	Cheese	Tomato	Gym	Yoga	Average
# URLs	20.6M	2.1M	4.6M	16.4M	9.2M
Precision	0.183	0.181	0.244	0.188	0.201
Recall	0.518	0.642	0.548	0.833	0.595
F1	0.270	0.282	0.338	0.307	0.300

We distinguish two patterns for the evidence: entity-to-entity if the object o and Q match on *Freebase mids*, and entity-to-text otherwise. The provenance, thus, is $\langle$null, Query logs, entity-to-entity/entity-to-text$\rangle$. We distinguish these two patterns since they often lead to different evidence quality (see Section 5).

3.2.4 Summary

Table 3 summarizes the different evidence-collection approaches. Horizontally the approaches are divided into three classes based on the type of the data sources. Vertically, there are also three categories: *Freebase*, *KV* and *Closed IE* are "closed" since they are restricted to known entities and relations; *Open IE* and *Web tables* are considered "open" since they can recognize unknown entities and relations; *Web co-occurrences* and *Query logs* are based on the "co-occurrences" of subject and object.

3.3 Results and discoveries

We next examine the evidence we obtained through the above techniques. Empirically, we set the string-similarity threshold $\theta = 0.35$, and use 4-gram for Web co-occurrences (we discuss parameter setting shortly). We use *precision* and *recall* to quantify the quality of the collected evidence. Let $Evidence$ be the set of triples that we found evidence for, and $Truth$ be the set of true triples. We define:

$$Precision = \frac{|Evidence \cap Truth|}{|Evidence|};$$

$$Recall = \frac{|Evidence \cap Truth|}{|Truth|}.$$

Further we have $F\text{–}measure = \frac{2*Precision*Recall}{Precision+Recall}$.

Quality of evidence. Table 4 shows the evidence collection results on each vertical. We find evidence from millions of URLs; there are more webpages about cheese and yoga poses, while fewer about tomato and gym exercises. The precision of the evidence is fairly similar among different verticals (between 0.18 and 0.25), while the recall ranges from 0.51 to 0.84. On average the obtained evidence has a precision of 0.20 and recall of 0.60, and covers 96% of the entities (we discuss the recall loss in Section 7.1).

Interestingly, based on the results, a more widely mentioned vertical may not have a higher recall in evidence collection. For instance, the vertical *Cheese* obtains evidence from the largest number of URLs, but it has the lowest recall. A detailed examination shows that in this vertical, the evidence for triples about different predicates differ greatly. For predicates like *cheese_type* and *state_of_origin*, we obtain a recall over 0.8 and each triple has hundreds of pieces of evidence; in contrast, for predicates like *taste* and *pairings*, the recall is around 0.1 and each triple has only a few pieces of evidence.

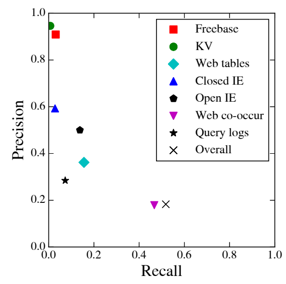

Figure 1: Quality of the evidence from different systems on *Cheese* vertical.

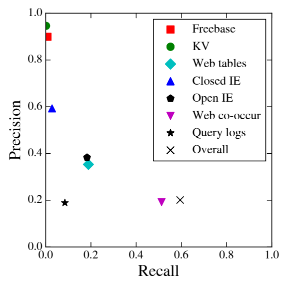

Figure 2: Average quality of the evidence from different systems on all verticals.

Table 5: The recall of evidence obtained by different methods from various sources; *singleton rate* is the percentage of triples for which only the particular method can find evidence.

	Closed	Open	Co-occur
Knowledge bases	3.3%	-	-
Web	2.8%	19.7%	46.7%
Query logs	-	-	7.3%
Singleton rate	0.0%	4.4%	29.0%

Comparison of different strategies. Using the *Cheese* vertical as an example, we further evaluate the performance of different evidence collection approaches. Figure 1 plots the precision and recall of each approach.

We can see that the two *knowledge bases* (*Freebase* and *KV*) provide evidence with the highest precision but the lowest recall, exactly as expected. Both *Closed IE* and *Query logs* have low recall (but higher than the knowledge bases), as they provide evidence only for *Freebase* entities, which is a very small portion of the input. *Query logs* also has quite low precision, indicating that understanding short query phrases is hard. *Open IE* and *Web tables* in a sense both apply open extractions; the former on texts and the latter on tables. They have medium precision and fairly low recall, but better than any closed system. *Web co-occurrences* has the highest recall but the lowest precision. This is not surprising as it purely relies on string matching and ignores any possible hint from structural information.

Figure 2 shows the average performance on the four verticals. We observe very similar trend, except that the quality of *Open IE* and *Web tables*, both being open extraction systems but applied on different types of data, are even closer.

In Table 5 we compare the recall of different methods. If we rely on closed techniques only, we obtain a recall of 0.04 in total: most of the evidence obtained by *Closed IE* are already contained in existing knowledge bases, and all evidence obtained by closed techniques can also be found through other approaches (singleton rate = 0). After including open techniques and relax the requirement for predicate matching, we obtain a recall of only 0.2; 4.4% of the true triples can only gain evidence through open extraction techniques. *Web co-occurrences* is the major contributor, with a singleton rate of 29%; the quality of all collected evidence is more or less dominated by it. However, because removing evidence collected by other approaches will decrease the recall of the final results by about 15% (by 30% in the vertical *Tomato*), to maintain the same precision. This justifies the need of the larger spectrum of extraction techniques we used. The majority of the evidence are collected from the web, which is not surprising.

Difference between true triples and false triples. As we have shown, we can find evidence for not only true triples, but also false triples. We are now wondering if the amount of evidence for a true triple is different from that of a false triple. We show results on *Cheese* vertical in Table 6 (we observe similar patterns on other verticals).

Table 6: Comparing evidence for true triples and evidence for false triples. *%True* and *%False* are the percentage of true triples that have evidence, and respectively, the percentage of false triples that have evidence. *#forTrue* and *#forFalse* are the number of pieces of evidence for a true triple, and respectively, for a false triple; *ratio* is the ratio between them.

	%True	%False	#forTrue	#forFalse	ratio
Freebase	3.1%	0.02%	1.5	1.1	1.4
KV	0.7%	0.003%	4.3	2.0	2.1
Web tables	15.6%	1.9%	4.6	2.4	1.9
Closed IE	2.8%	0.1%	43.1	6.7	6.4
Open IE	13.8%	1.0%	21.3	3.9	5.5
Web co-occur	46.7%	14.8%	209.4	16.2	12.9
Query logs	7.3%	0.9%	2.7	2.0	1.4
Overall	51.8%	17.3%	117.7	13.4	8.8

We first compare the percentage of true triples with evidence against that of false triples. On average, we were able to find evidence for 52% of the true triples and 17% of the false triples; all systems are more likely to find evidence for true triples rather than false triples. Next we compare the amount of evidence for a true triple against that for a false triple. We observe that the number of pieces of evidence for a true triple is 8.8 times as many as that for a false triple on average.

For all evidence-collection approaches, there are more evidence for true triples than for false triples, which conforms to our intuition. Among different approaches, although *Web co-occur* finds evidence for 14.8% false triples, its evidence is the most distinguishable: the number of pieces of evidence for a true triple is 13 times as many as that for a false triple. *Closed IE* and *Open IE* also give reasonable results, where the evidence for true triples is six times as much as that for false triples. *Web tables* and *Query logs* are less effective on distinguishing true triples from false triples (with a ratio below 2). Although the ratio for *Freebase* and *KV* are not high, they rarely provide evidence for false triples (*i.e.*, with low %False), even though we do not require predicate match.

Parameter tunning. We found that setting $\theta = 0.35$ and using 4-gram for Web co-occurrences obtain the best results for all verticals. A higher parameter value would inevitably reduce recall. Interestingly, a lower parameter value can increase recall and F-measure for evidence collection, but introduce noise in the evidence, and thus hurt the precision and F-measure of the final results from knowledge fusion, which we describe in the next section.

Summary. By exploiting a wide spectrum of techniques, we were able to collect evidence for 60% of the (correct) tail knowledge. However, we also found evidence for much more false triples, showing that simply looking for supporting evidence is not a reliable approach for knowledge verification. Among different strategies, we found that the web is the best source for finding evidence, co-occurence-based approaches have the highest recall, but open techniques make the balance between precision and recall.

4. OVERVIEW OF KNOWLEDGE FUSION

The previous section shows that although we found evidence for both true triples and false triples, the amount of evidence for a true triple is usually much more than that for a false triple. Also, the evidence obtained by different approaches often have different qualities. These two observations inspire us to apply knowledge fusion [12, 13] to decide the truthfulness of each triple based on the collected evidence.

Table 7: Triples regarding *ice hockey equipment*. $\surd/\times$ indicates the correctness of a triple.

	subject	predicate	object	source
$\surd\ c_1$	ice hockey	equipment	helmet	s_1, s_3
$\surd\ c_2$	ice hockey	equipment	stick	s_1, s_2
$\times\ c_3$	ice hockey	equipment	boots	s_2
$\times\ c_4$	ice hockey	equipment	board	s_3
$\surd\ c_5$	ice hockey	equipment	neck guard	

Table 8: Notations used in Section 4.

Notation	Description
d	a data item
v	a value
S	a source that provides values
Φ	mapping between values and sources for a data item
$\Phi(S)$	the set of values provided by S on a data item
$\mathcal{O}$	a sequence of values that have been selected as truths
$\perp$	"there is no more truth"

Knowledge fusion is a research topic that predicts the correctness of knowledge triples by examining extractions made by multiple systems from various sources. We review the existing knowledge fusion methods and propose a new approach in Section 4.1, and then study their performance in Section 4.2.

4.1 Review of knowledge fusion methods

In the context of knowledge verification, each (subject, predicate) pair is considered as a *data item*, and each object is considered as a *value*. For simplicity, we follow [12] and consider an evidence provenance as a source that provides the triple (our experiments did not show significant gain by separating sources and extractors as in [13]). For instance, Table 7 shows five triples regarding the data item (ice hockey, equipment). There are three sources (s_1, s_2 and s_3) that provide four values for this data item, while "neck guard" is not provided by any source.

Given a data item d and a set $\bar{S}$ of sources that provide values on d, knowledge fusion aims to compute a probability $p(v)$ for each value $v \in \mathcal{V}$ being true, where $\mathcal{V}$ denotes all possible values for d ($\mathcal{V}$ may contain values not provided by any source). Let Φ denote the mapping between $\bar{S}$ and $\mathcal{V}$, and $\Phi(S)$ denote the values provided by a source $S \in \bar{S}$ on d (we dismiss d in the notation for simplicity). Our goal then becomes computing the posterior probability $p(v|\Phi)$ for v being true based on the observations from the sources. Table 8 summarizes the notations. Note that in this paper we focus on the case where the sources are independent of each other; one can incorporate with the techniques from [10, 32] to address the correlations between sources.

We categorize the existing knowledge fusion methods into two classes: single-truth models [24], and multi-truth models [32, 45]. Our observation from Table 2 shows that the majority of our data items have a single truth, but there are also cases with multiple truths. We thus propose a hybrid method that combines the strengths of existing methods and meanwhile takes into consideration the prior value of the number of truths. We next introduce each category respectively. While our review mainly focuses on Bayesian-based approaches, there are also graphical-model approaches [31] and optimization-based approaches [23] that share similar intuitions.

4.1.1 *Single-truth models*

Single-truth models [24, 26] assume that there is only one true value for a data item, and thus $\sum_{v\in\mathcal{V}} p(v|\Phi) = 1$. The value v with the highest probability $p(v|\Phi)$ is then selected as the truth.

The intuition behind the single-truth models is that values provided by more sources and higher-quality sources are more likely to be true. The quality of a source S is measured by its *accuracy* $A(S)$, which is the probability that a value provided by S is true.

We now explain how to compute $p(v|\Phi)$ using the ACCU [10] model, as other models share a lot of commonalities. Let $\Phi(S)$ denote the values provided by a source S on d. Under the source-independence assumption and applying Bayesian analysis, we have

$$p(v|\Phi) = \frac{\Pi_{S\in\bar{S}}\, p(\Phi(S)|v)\cdot\lambda}{\sum_{v'\in\mathcal{V}} \Pi_{S\in\bar{S}}\, p(\Phi(S)|v')\cdot\lambda}. \quad (1)$$

Here $p(\Phi(S)|v)$ is the probability of observing $\Phi(S)$ given that v is the truth. λ is the *a priori* probability that a value v is true; we usually assume λ is the same for all values in $\mathcal{V}$, so can cross it out from the numerator and the denominator.

Assuming there are n false values in the domain ($n = |\mathcal{V}| - 1$), S provides a true value with probability $A(S)$, and a *particular* false value with probability $\frac{1-A(S)}{n}$. Thus we have:

$$p(\Phi(S)|v) = \begin{cases} A(S) & \text{if } v \in \Phi(S); \\ \frac{1-A(S)}{n} & \text{if } v \notin \Phi(S). \end{cases} \quad (2)$$

EXAMPLE 4.1. *Consider the triples c_1-c_5 in Table 7. Suppose $n = 10$ and $A = 0.6$ for all sources.*

Triples c_1 and c_2 are provided by 2 sources, so we have $\Pi_{S\in\bar{S}}\, p(\Phi(S)|c_1) = \Pi_{S\in\bar{S}}\, p(\Phi(S)|c_2) = 0.6^2 \times \frac{1-0.6}{10} = 0.0144$. *Similarly,* $\Pi_{S\in\bar{S}}\, p(\Phi(S)|c_3) = \Pi_{S\in\bar{S}}\, p(\Phi(S)|c_4) = 0.001$.

From Eq. (1) *we compute triple probabilities as follows:* $p(c_1) = p(c_2) = \frac{0.0144}{0.0144+0.0144+0.001+0.001} = 0.47$*;* $p(c_3) = p(c_4) = 0.03$*, and* $p(c_5) = 0$ *as it is not provided by any source.*

*We see that the probabilities of all values add up to 1, so even true values (*helmet *and* stick*) have rather low probabilities.* □

Obviously, the limitation of single-truth models is that when multiple truths exist, they at best find one of them.

4.1.2 *Multi-truth models*

Multi-truth models [32, 38, 45] allow the existence of multiple truths. They compute the probability of each value separately. Hence, they do not require $\sum_{v\in\mathcal{V}} p(v|\Phi) = 1$, but only $p(v|\Phi) + p(\neg v|\Phi) = 1$, where $\neg v$ denotes that v is a false value. A value v is considered true if $p(v|\Phi) > p(\neg v|\Phi)$; that is, $p(v|\Phi) > 0.5$.

An *unknown semantics* is used to capture the nature of multi-truth: if a source S does not provide the value v on d, S means that it *does not know* whether or not v is correct (instead of saying v is *incorrect*). Accordingly, multi-truth methods capture the quality of a source S by two metrics: the *precision* $P(S)$, which is the same as the accuracy in ACCU, and the *recall* $R(S)$, which is the probability of a truth is provided by S. Intuitively, values provided by high-precision sources are likely to be true, and values not provided by high-recall sources are likely to be false.

The PRECREC method [32] computes $p(v|\Phi)$ as follows:

$$p(v|\Phi) = \frac{\Pi_{S\in\bar{S}}\, p(\Phi(S)|v)\cdot\lambda}{\Pi_{S\in\bar{S}}\, p(\Phi(S)|v)\cdot\lambda + \Pi_{S\in\bar{S}}\, p(\Phi(S)|\neg v)\cdot(1-\lambda)}. \quad (3)$$

We now explain the computation of $p(\Phi(S)|v)$ and $p(\Phi(S)|\neg v)$. First, from $P(S)$ and $R(S)$, we derive the *false positive rate* $Q(S)$ of S, as $Q(S) = \frac{\lambda}{1-\lambda}\cdot\frac{1-P(S)}{P(S)}\cdot R(S)$ according to [32]. Then S provides v with probability $R(S)$ if v is true, and with probability $Q(S)$ if v is false:

$$p(\Phi(S)|v) = \begin{cases} R(S) & \text{if } v \in \Phi(S); \\ 1 - R(S) & \text{if } v \notin \Phi(S). \end{cases} \quad (4)$$

$$p(\Phi(S)|\neg v) = \begin{cases} Q(S) & \text{if } v \in \Phi(S); \\ 1 - Q(S) & \text{if } v \notin \Phi(S). \end{cases} \tag{5}$$

EXAMPLE 4.2. *Again, consider triples c_1-c_5 in Table 7. Suppose $R = 0.5$ and $Q = 0.33$ for all sources. Then for c_1, we have*
$\Pi_{S \in \bar{\mathcal{S}}}\ p(\Phi(S)|c_1) = R(s_1)(1 - R(s_2))R(s_3) = 0.125$*;*
$\Pi_{S \in \bar{\mathcal{S}}}\ p(\Phi(S)|\neg c_1) = Q(s_1)(1 - Q(s_2))Q(s_3) = 0.074$*.*
Assuming λ=0.5, from Eq. (3) *we have*
$p(c_1) = \frac{0.125 \times 0.5}{0.125 \times 0.5 + 0.074 \times (1 - 0.5)} = 0.63$*. Similarly,*
$p(c_2) = 0.63$*,* $p(c_3) = p(c_4) = 0.54$*, and* $p(c_5) = 0.28$*.*
Therefore, all provided values are considered true under PRECREC*, resulting in false positives (*i.e.*,* boots *and* board*).* □

In practice, even multi-truth items often have only a few true values instead of infinite number of truths. Existing multi-truth models cannot capture this because they decide the truthfulness of each value independently, without considering other values in $\mathcal{V}$ and thus lack a *global view* of a data item.

4.1.3 Hybrid models

To gain a global view in truth finding while allowing identifying multiple truths, we propose a hybrid model, called HYBRID. HYBRID considers the conflicts between values as important evidence for ruling out wrong values, while keeping the flexibility of allowing multiple truths. In addition, HYBRID considers the prior value of the number of truths in the model.

HYBRID makes two decisions for a data item: (i) how many truths there are, and (ii) what they are. Essentially, it interleaves the two decisions and finds the truths one by one: conditioning on a sequence $\mathcal{O}$ of true values that have been selected previously, it decides (1) the probability of a value v being the next truth, denoted by $p(v|\mathcal{O}, \Phi)$, and (2) the probability that there is no more truth, denoted by $p(\perp|\mathcal{O}, \Phi)$. These are disjoint decisions so their probabilities sum up to 1: $\sum_{v \in \mathcal{V} \setminus \mathcal{O}} p(v|\mathcal{O}, \Phi) + p(\perp|\mathcal{O}, \Phi) = 1$.

Thus, when *selecting the next truth*, HYBRID basically applies a single-truth model. However, when deciding *whether there is any more truth* (*i.e.*, $p(\perp|\mathcal{O}, \Phi)$), HYBRID incorporates the *unknown semantics* used in multi-truth models.

Moreover, HYBRID leverages the typical number of truths for each type of data items. For example, a person typically has 2 parents and 1-5 children. HYBRID allows incorporating such knowledge as *a priori* probability of $p(\perp|\mathcal{O}, \Phi)$.

Source quality metrics. As HYBRID jointly models the number of truths as well as the true values, it captures the quality of a source with two sets of metrics: that for deciding whether there exists a truth, and that for deciding the true values.

The first set of metrics enables the *unknown semantics* for multi-truth, and it includes two measures: (1) *precision* $P(S)$, the probability that when S provides a value, there indeed exists a truth; (2) *recall* $R(S)$, the probability that when there exists a truth, S provides a value. Note that our $P(S)$ and $R(S)$ are different from the same notions in PRECREC: we only measure how well S predicts whether or not there exists a truth, but not how well S predicts what the truth is; in other words, we do not require the value provided by S to be the same as the truth.

The second set of metrics follows single-truth models to address the conflicts between values. It contains one measure: *accuracy* $A(S)$, the probability that a value provided by S for a "real" truth slot is true (*i.e.*, S provides a true value after it has correctly predicted the existence of a truth slot). Note that values provided for non-existing truth slots, which are absolutely false, are not counted here, as they have been captured by $P(S)$.

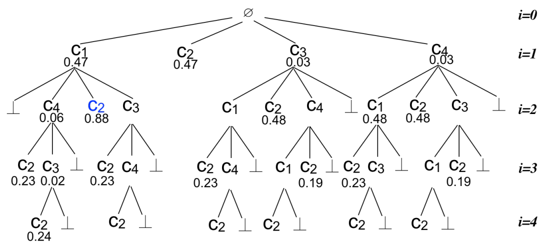

Figure 3: Tree structure for computing $p(c_2)$ in Table 7. We omit triples without any source for simplicity.

Table 9: Value probabilities computed by different fusion methods for data item *(ice hockey, equipment)*.

	helmet	stick	boots	board	neck guard
Single-truth [10]	0.47	0.47	0.03	0.03	0.00
Multi-truth [32]	0.63	0.63	0.54	0.54	0.28
HYBRID	0.92	0.92	0.08	0.08	0.00

Value probability computation. We next describe how to obtain $p(v|\Phi)$. As we select the truths one by one, there can be various sequences of truths (of any length below $|\mathcal{V}|$) that are selected before v (we may select the value v after selecting v_1, or after selecting $v_1 v_2$, *etc.*). We call each sequence $\mathcal{O}$ a *possible world* and denote by Ω all possible worlds. Then the probability of v is the weighted sum of its probability $p(v|\mathcal{O}, \Phi)$ in each possible world $\mathcal{O}$:

$$p(v|\Phi) = \sum_{\mathcal{O} \in \Omega} p(v|\mathcal{O}, \Phi) \cdot p(\mathcal{O}|\Phi) \tag{6}$$

where $p(\mathcal{O}|\Phi)$ is the probability of entering the possible world $\mathcal{O}$.

Let $\mathcal{O} = v_1 v_2 \ldots v_{|\mathcal{O}|}$, $v \notin \mathcal{O}$, denote a possible world with the sequence $v_1, v_2, \ldots, v_{|\mathcal{O}|}$ of values selected as truths. Let $\mathcal{O}_j$ denote a prefix of $\mathcal{O}$ with length j and $\mathcal{O}_0 = \varnothing$. Applying the chain rule leads us to:

$$p(\mathcal{O}|\Phi) = \prod_{j=1}^{|\mathcal{O}|} p(v_j|\mathcal{O}_{j-1}, \Phi). \tag{7}$$

Now the only piece missing from Eqs (6-7) is the conditional probability $p(v|\mathcal{O}, \Phi)$. They are computed according to the three quality metrics and we refer readers to [21] for details.

EXAMPLE 4.3. *One way to think about the computation in* HYBRID *is through a tree structure (See Figure 3 as an example for c_2 in Table 7). The root of the tree represents having not selected any value. A path from the root to a node v represents a possible way to select v; for example, the path c_1-c_4-c_2 corresponds to the case where we select c_2 after selecting c_1 and c_4 sequentially ($\mathcal{O}$=$c_1 c_4$). The children of a node represent candidates for the next truth, containing all unselected values in $\mathcal{V}$ and $\perp$.*

The number under each node v is the probability $p(v|\mathcal{O}, \Phi)$. For example, following the path $\varnothing$-c_1-c_4-c_2, we have $p(c_2|c_1 c_4, \Phi) =$ 0.23 (see [21] for details on how this number is obtained). Probability $p(\mathcal{O})$ is given by the product of the numbers along the path. For example, $p(c_1 c_4 c_2|\Phi) = 0.47 \times 0.06 \times 0.23 = 0.007$. Then the probability $p(v|\Phi)$ of v being true is thus the sum of the probabilities of all paths ending with v, according to Eq. (6)*. In our example, we can reach c_2 through 16 paths, and we obtain $p(c_2) = 0.92$.* □

While computing value probabilities by enumerating all possible worlds is expensive, we develop an efficient algorithm that runs in polynomial time and has an approximation bound of 1/6 (see [21] for details).

Table 9 compares the probabilities computed by different fusion models on the data item *(ice hockey, equipment)*. We can see that

Table 10: Statics of the Book data.

#entities	#triples	#sources	precision	recall	%multi-truth
1,263	6,139	876	0.62	0.98	57%

Table 11: Results on Book data. HYBRID **obtains the highest recall and F-measure.**

	Precision	Recall	F1
ACCU	**0.990**	0.532	0.692
ACCU_LIST	0.974	0.801	0.879
LTM	0.911	**0.973**	0.941
PRECREC	0.965	0.931	0.947
HYBRID	0.941	**0.973**	**0.957**

HYBRID clearly gives high probabilities for true values and low probabilities for false values.

4.2 Performance study of different methods

Before showing the results on tail verticals, we first compare the various methods, including the newly proposed HYBRID, on a widely used dataset, as well as synthetic data, to gain insights on their performance. We compare HYBRID with a single-truth approach ACCU [10], and two multi-truth approaches PRECREC [32] and LTM [45] (LTM shares the same intuition as PRECREC but uses a graphical model).

We implemented all methods in Java. Following previous works, we set $n = 10$, and $\lambda = 0.5$. We initialize the source quality metrics as $A(S) = P(S) = 0.8$ and $Q(S) = 0.2$ for every source S. We then iteratively compute triple probabilities and source qualities for up to 5 iterations.

We report the *precision*, *recall* and *F1* for each method. *Precision* measures among all values predicted as true, the percentage that are indeed true. *Recall* measures among all true values, the percentage that are predicted as true.

4.2.1 Results on Book data

We first use the Book data from [43], which has been widely used for knowledge-fusion experiments. As shown in Table 10, it contains 6,139 book-author triples on 1,263 books from 876 retailers. The gold standard consists of authors for 100 randomly sampled books, where the authors were manually identified from book covers. According to the gold standard, 62% of the provided authors are correct, and 98% of the true values are provided by some source. There are 57% of the books that have multiple authors.

In addition to the five fusion methods we listed before, we also compared with ACCU_LIST, which applies ACCU but considers the full list of authors as a whole [10, 43].

Table 11 shows the results. Not surprisingly, ACCU has the highest precision but the lowest recall as it only finds one author for a book; even though ACCU_LIST treats each author list as a whole, its recall is still low. The two multi-truth models, LTM and PRECREC, have similar F-measure, while PRECREC appears to be more conservative (a higher precision but lower recall). In this dataset many sources only provide the first author of a book and this explains the low recall; the high precision is because the sources rarely provide the same wrong value. HYBRID obtains a higher F-measure than existing single-truth and multi-truth models. By considering both conflicts between values and the possibility of having multiple truths, it is able to identify more true values without sacrificing much of precision.

4.2.2 Results on synthetic data

To better understand the performance of different methods in various situations, we next compare them on synthetic data where we vary the number of truths and the quality of sources. We generated 10 data sources providing values on 100 data items, where wrong values are randomly sampled from a domain of 100 values.

The data generation is controlled by two sets of parameters. The first parameter is the *number of truths* for a data item. It ranges from 1 to 10, and by default follows a Gaussian distribution with mean = 6 and standard deviation = 1. The second set of parameters control the values of a source S. Given the ground truth $\mathcal{T}$ of a data item, for each $v \in \mathcal{T}$, S has a probability of $\%cover$ to provide a value, and with probability $\%correct$ this provided value is v (otherwise it is a wrong value). Meanwhile, for each $v \in \mathcal{T}$, S has a probability of $\%extra$ to provide a random wrong value. Hence, $\%cover$ and $\%extra$ control the number of values provided by a source, while $\%correct$ controls the correctness of a provided value. All the three parameters range from 0.2 to 1; by default we set $\%cover$ and $\%correct$ to 0.7, and $\%extra$ to 0.2. All experiments were repeated 100 times and we report the average performance.

Figure 4 shows the results when we vary the number of truths for a data item. We can see that HYBRID can fairly well "guess" the number of truths and consistently outperforms the others. As the number of truths increases, the precision of HYBRID remains high, while the precision of PRECREC drops. This is because the *extra ratio* is fixed; so when there are more truths, more wrong values will be provided by the sources, and PRECREC is more sensitive to noise. Again, LTM gives a low precision but a high recall, consistent with our observation on the Book data. ACCU usually has the highest precision but the lowest recall. However, its precision can be low when the number of truths is small; this is because ACCU always finds a truth for each data item, while other methods may not output any truth (when all the value probabilities are below 0.5).

Figure 5 shows the F-measure of all methods when we vary $\%cover$, $\%correct$, and $\%extra$ of the sources (see [21] for precision and recall). Not surprisingly, all methods obtain better results when the sources are more complete (higher $\%cover$) and accurate (higher $\%correct$). On the contrary, the performance of all methods drop when $\%extra$ increases. We observe that HYBRID has the highest F-measure in general and it is the most robust; it typically outperforms others when the source observations are noisy. PRECREC obtains the worst results when $\%cover$ and $\%correct$ are medium (0.4-0.6); this is because in this case the sources have similar probabilities to provide a true value and a false value, and PRECREC is unable to distinguish them.

Summary. HYBRID outperforms existing single-truth and multi-truth models and is the most robust under varying source qualities.

5. APPLYING KNOWLEDGE FUSION

We now apply knowledge fusion on tail verticals. As shown in Figure 2 and Table 10, the quality of the collected evidence on tail verticals is much lower than that of the Book data. Furthermore, the long-tail data has a higher variety in terms of the number of truths and the number of sources. Hence, in this set of experiments we seek to answer the following questions: *1) Is knowledge fusion effective on datasets that are highly incomplete and noisy? 2) Among different fusion models, which one performs the best on tail data?*

We point out that the effectiveness of knowledge fusion relies on the fact that the set of triples are in the same vertical, so that we can estimate the quality of the sources meaningfully. We also point out that the predicate matching information recorded in `pattern` is leveraged in knowledge fusion as we evaluate the quality of each provenance.

5.1 Implementations

We compare the following three knowledge fusion algorithms: the single-truth model ACCU, the multi-truth model PRECREC, and

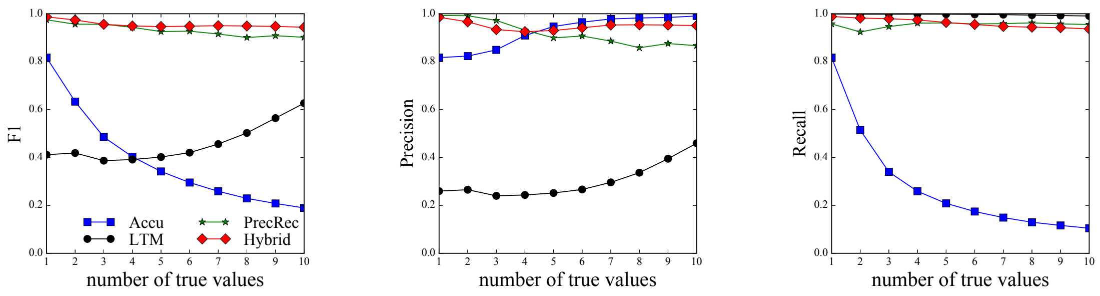

Figure 4: Varying the number of truths on synthetic data. HYBRID **improves over other models when the number of truths is large.**

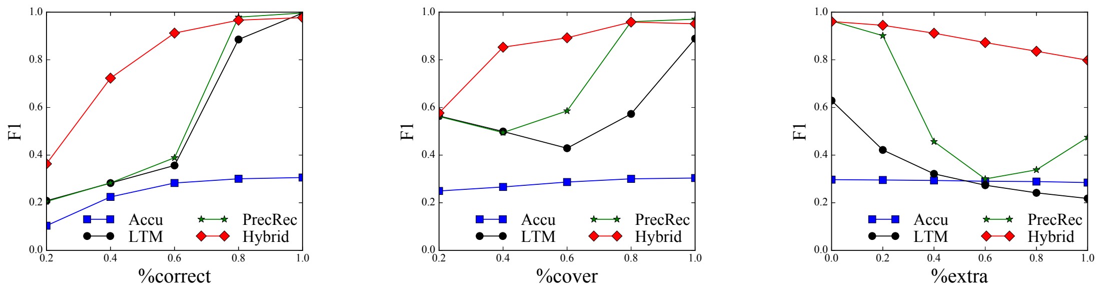

Figure 5: Varying source quality on synthetic data. HYBRID **performs best and is the most robust among all methods.**

the hybrid model HYBRID. We skip LTM because it does not scale on this dataset. We test two settings of the algorithms: (1) assume each source has the same quality, and then iteratively compute triple probabilities and source qualities; (2) initialize source quality by comparing its provided values against the gold standard[3], and run the fusion algorithms in one-pass. We distinguish these two options by appending "+" to the latter (*e.g.*, ACCU+ for quality bootstrap). By default, we use HYBRID+ for our framework.

We remove evidence from sources that provides no evidence for any true triple, as these are likely to be irrelevant sources. Our experimental results show slightly better results and we skip the details because of lack of space.

5.2 Results

High-level results. We first evaluate the effectiveness of knowledge fusion on tail verticals. Note that knowledge fusion models predict the probability of a triple being true. We thus order triples in decreasing order of the predicted probabilities; then as we gradually consider triples above a probability threshold, we plot the precision *vs.* recall (known as the PR-curve) obtained by HYBRID+ in Figure 6. In a PR-curve, the highest point corresponds to the results when considering triples with probabilities above 0.99, while the lowest point corresponds to all triples with evidence.

We observe that knowledge fusion effectively filters false triples. For instance, in the *Cheese* vertical, while the raw evidence has an F-measure of 0.27 (precision = 0.18, recall = 0.52, corresponding to the lowest point in the curve), the highest F-measure obtained after knowledge fusion is 0.62 (precision = 0.84, recall = 0.49). With only 3% recall loss, the precision is 4.6 times higher.

Among different verticals, we obtain fairly similar precision (between 0.81 and 0.86), but the recall vary from 0.37 to 0.61. Interestingly, while the vertical *Yoga* obtains the lowest recall after knowledge fusion, its recall on *evidence collection* (the rightmost point of the red curve) is considerably higher than the others. A

[3]In practice, we can use a gold standard on a small sample to initialize source quality. Our experiments on a 10% sample obtains similar results.

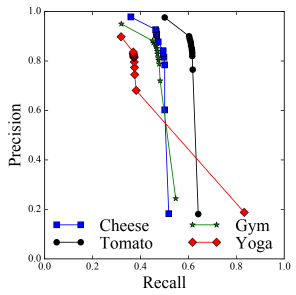

Figure 6: PR-curves of knowledge fusion on tail verticals.

Table 12: Requiring a precision of 0.9, we achieve a recall of 47.1%, and we can provide at least three sources for 44.8% of the true triples.

	Cheese	Tomato	Gym	Yoga	Average
recall@0.9	46.7%	61.7%	40.0%	31.9%	47.1%
#URLs $\geq$ 3	45.0%	60.1%	38.8%	21.7%	44.8%

closer examination reveals that in *Yoga* the collected evidence is very noisy: we find evidence for 67% false triples, and a true triple has only *twice* as much evidence as a false triple. In contrast, on the *Cheese* vertical a true triple has eight times as much evidence as a false triple. Therefore the knowledge fusion is less effective on *Yoga* since the collected evidence is less distinguishable.

In practice we require a high precision such that the approach can be deemed reliable. As shown in Table 12, when we require a precision of 0.9, we were able to verify 47.1% of the true triples, and for 44.8% of the true triples we can find at least three websites that contain supporting evidence, showing that they are *public* knowledge.

Effect of triple types. Using the *Cheese* vertical as an example, we next study how the type of a triple may affect the performance of knowledge fusion. We distinguish triples based on whether or not their objects can be reconciled to *Freebase mids*, calling them *entity-obj* triples and *string-obj* triples correspondingly. Figure 7 shows the PR-curve of each category.

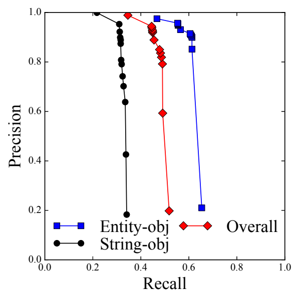

Figure 7: Effect of triple types.

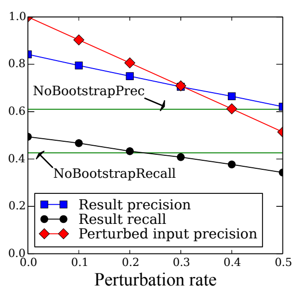

Figure 8: Effect of source quality bootstrap.

Table 13: Knowledge fusion results on four verticals.

	P	R	F	P	R	F
	Cheese			Tomato		
ACCU	0.637	0.790	**0.705**	0.725	0.828	**0.773**
PRECREC	0.597	0.820	0.691	0.701	0.836	0.763
HYBRID	0.610	0.822	0.700	0.706	0.853	**0.773**
ACCU+	0.850	0.923	0.885	0.852	0.944	0.896
PRECREC+	0.791	0.975	0.873	0.826	0.970	0.892
HYBRID+	0.842	0.954	**0.894**	0.851	0.958	**0.901**
	Gym			Yoga		
ACCU	0.669	0.704	**0.686**	0.514	0.233	0.321
PRECREC	0.642	0.712	0.675	0.388	0.212	0.275
HYBRID	0.652	0.724	**0.686**	0.482	0.253	**0.332**
ACCU+	0.838	0.841	0.839	0.786	0.353	0.487
PRECREC+	0.796	0.929	**0.857**	0.720	0.709	**0.715**
HYBRID+	0.829	0.861	0.845	0.813	0.448	0.577

With no surprise, *entity-obj* obtained much higher precision and recall than *string-obj*: the higher recall is because reconciled objects are typically more popular and thus widely mentioned; the higher precision is due to the more reliable evidence from structured sources like *Freebase* and *KV*.

Comparing different knowledge fusion methods. Table 13 compares the performance of different approaches. To focus on the fusion models, we report the *relative recall*; that is, the recall on triples for which at least some evidence is collected.

We observe that HYBRID (or HYBRID+) has slightly better results than ACCU and PRECREC. Comparing with ACCU, it has similar precision but higher recall; it is able to find 10% more truths on predicates where over 10% data items have multiple truths. Comparing with PRECREC, it has higher precision and slightly lower recall, but higher F-measure; the precision gain is more pronounced for predicates with a lot of noise, where the precision of HYBRID can be twice as much as that of PRECREC.

An exception is vertical *Yoga*, where about 60% data items have multiple truths. Although HYBRID+ has a 13% higher precision than PRECREC+, its recall is 35% lower. This is because most of the provenances contribute evidence for a single triple of a data item, so HYBRID+ often predicts that there is only one truth, especially given that the evidence for this vertical is very noisy. Interestingly, without source-quality bootstrap, HYBRID outperforms PRECREC by 24% on precision and by 19% on recall, showing that HYBRID is more robust in absence of bootstrap.

Effect of source quality bootstrap. As shown in Table 13, by initializing source quality using the gold standard before fusion, we obtained significantly better results. We next examine the effect of this bootstrap process. We ask two questions: *(1) does this bootstrap always work? (2) if our prior knowledge on source quality is inaccurate, how robust are the results?*

Table 14: Results on knowledge verification. Each column evaluates the output of a component (*e.g.*, Evidence collection and Knowledge fusion for FACTY). The highlighted column of each method gives the final results; FACTY outperforms T-VERIFIER significantly.

		FACTY		T-VERIFIER	
		Step I Evidence	Step II KFusion	Step I Search	Step II Ranking
Cheese	P	0.183	**0.842**	0.066	**0.163**
	R	0.518	**0.494**	0.984	**0.159**
	F	0.270	**0.623**	0.124	**0.161**
Tomato	P	0.181	**0.851**	0.129	**0.131**
	R	0.642	**0.614**	0.979	**0.218**
	F	0.282	**0.713**	0.228	**0.163**
Gym	P	0.244	**0.849**	0.147	**0.279**
	R	0.548	**0.466**	0.898	**0.232**
	F	0.338	**0.602**	0.252	**0.253**
Yoga	P	0.188	**0.813**	0.161	**0.245**
	R	0.833	**0.373**	0.947	**0.093**
	F	0.307	**0.512**	0.275	**0.135**
Overall	P	0.201	**0.843**	0.123	**0.223**
	R	0.595	**0.506**	0.947	**0.190**
	F	0.301	**0.633**	0.218	**0.205**

We perturbed the gold standard that we use to initialize source quality: given a percentage of data items, we replace a triple in gold standard with a wrong triple. As such, the initial source quality we obtained according to this perturbed gold standard would be different from the real quality of the sources. We show results on the *Cheese* vertical by HYBRID+ in Figure 8 (similar trends are observed in other verticals). We see that both the precision and the recall of the fusion results drop when the initial source quality becomes inaccurate. However, comparing with the dropping rate on the perturbed gold standard, the dropping rate of result precision is 44% and that of result recall is 30%. Further, the result precision is always higher than that without bootstrap.

Summary. Our experiments show that knowledge fusion can effectively identify the correct triples based on the evidence, improving precision from 0.20 to 0.84, without sacrificing much of the recall. In general HYBRID is adequate for both single-truth and multi-truth cases; however, comparing with the results on synthetic data, the improvement of HYBRID is limited because of the sparsity of evidence we can automatically collect for tail verticals. In cases when the majority of data items have multiple truths and we have reliable data to estimate source quality, applying PRECREC may better leverage the bootstrap and improve recall.

6. OVERALL EVALUATION OF KNOWLEDGE VERIFICATION

We finally evaluate the overall performance of knowledge verification. We compare the following methods.

- FACTY, our framework that performs evidence collection followed by knowledge fusion. It applies HYBRID+ to decide the correctness of each triple.
- T-VERIFIER [25], the state-of-the-art approach that verifies a triple by searching the web. For each candidate triple, it first searches its subject and object using *Google*. It then ranks the candidates of each data item based on features such as the number of results returned by the query, distance between keywords in the result snippet, ranking given by Google, and so on. A weight vector is learnt to combine the features. Among different candidates, it selects the triple with the highest score as true.

Table 14 reports the precision, recall and F1 of each method. On average, FACTY obtains a precision of 0.843 and a recall of

0.506 (in column "KFusion"), whereas T-VERIFIER gives much lower precision and recall (in column "Ranking"). In the first step, T-VERIFIER obtains very noisy results (low precision in column "Search"); this is because a false triple whose object contains popular words or phrases can occur much more often than a true triple. Unfortunately, none of the features considered in the ranking step is able to detect this, resulting in a low precision. In contrast, the knowledge-fusion step in FACTY dramatically increases the precision over raw evidence. Note that the quality of T-VERIFIER results we obtained is also lower than that reported in [25], indicating that it is not suitable for verifying long-tail triples.

We also note that evidence collected by FACTY are more precise than that by web search in T-VERIFIER. This is because our evidence collection approaches usually exploit stricter rules than web search. However, the search results obtained by T-VERIFIER have a very high recall (95% on average); this is not surprising since for most queries, the search engines will return some results, although they may not be relevant.

7. FUTURE DIRECTIONS

Our experimental results show that by employing various evidence collection techniques and applying knowledge fusion on the obtained evidence, we can obtain reasonable results for knowledge verification, and it significantly outperforms the existing approach. However there are still much room for improvement. Next, through a series of error analysis, we discuss possible future directions.

7.1 Improving recall

As shown in Figure 2, our evidence collection has a recall of 60%, which means that there are 40% of correct triples for which we cannot find any evidence. We randomly selected 10 such triples, and manually constructed various keyword search queries for each of them and tried to find evidence from top-20 search results. We were able to find evidence for 8 of them. For the two we cannot find, one has a very ambiguous entity name and it is even hard to find the entity itself; the other is not mentioned in top-20 results of various keyword search. This investigation suggests several possibilities for improving the recall of evidence collection.

Open extraction for DOM trees. Among the eight triples, six triples exist in DOM data. Recall that we have open extractions for web tables and texts, but not for DOM trees. Indeed, we are not aware of any such technique, since DOM trees have neither grammar structure nor table structure to indicate the semantics. There are rich data in DOM trees, and many of them are regarding entities and attributes not in existing knowledge bases. We believe that extracting knowledge in this form will dramatically enrich existing knowledge bases.

Allowing long-distance co-occurrences. In our examination, four triples have the subject mentioned in article (or DOM tree) title, while the predicate or object mentioned in the texts. Extractions, including *Web co-occurrences*, fail because of the long distance between subjects and objects. An easy fix would be increasing the window size when looking for co-occurrences, but this may hurt the precision. A more advanced approach is to treat the article title and the texts differently.

Using embedding methods. Another possibility is to jointly embed web documents and the existing knowledge-base triples into the same vector space [28, 39]. In this way, we represent predicates with dense vectors instead of lexical words, and thus largely reduce the dimensionality. Then a candidate triple is true if its subject-vector can be transformed to object-vector through predicate-vector. While prior research on this direction is restricted to pre-known predicates, extending to unknown predicates is an interesting problem.

Extractions of dates and numbers. In general we miss a lot of dates and numbers (often with units) because they can appear in many different formats, especially for range numbers. Sometimes the number is not exact (such as *population*), which makes it hard to verify. This applies to one triple in our sample. Possible solutions include defining numerical value mapping rules and improving extraction of numerical relations [27].

7.2 Improving precision

Matching Predicates. Recall that when we look for evidence, we require subject match and object match, but being tolerant on predicate semantics; actually, *Web co-occurrences*, our major evidence contributor, completely ignores predicates. This strategy significantly increases recall, but fails in two cases (the vertical *Yoga* has both cases, making the collected evidence very noisy):

1. When we have two predicates with related but different (or even contradictory) semantics, such as "major source" versus "other sources", or "friends" versus "foes", the objects of the two predicates are from the same domain, and are thus indistinguishable without considering the predicates.
2. When the subjects and objects of a predicate are in the same domain, such as the "follow-up posture" of a posture in exercises, *Web co-occurrences* may simply return all postures in a list.

We may improve precision by requiring the predicate to also co-occur with the subject and object for such special cases, or use embedding instead of lexical words to represent the predicates.

Removing noise from highly frequent names. Another place where we make mistakes is when an object name is very popular and thus ambiguous, so false triples may obtain more evidence due to the high frequency of the entity name. How to incorporate the *DF* (*i.e.*, document frequency) in evidence collection is a topic of future research.

8. RELATED WORK

Knowledge verification. To the best of our knowledge, T-VERIFI ER [25] is the first system on knowledge-verification, and we have reviewed it in Section 6. A recently proposed supervised approach, ClaimEval [33], aims to decide the correctness of a *binary* claim, such as "yoghurt is healthy food", and "SIGMOD is a top CS conference". It also finds evidence for each claim through web search; then it trains a classifier for each category of claims (*e.g.*, classifiers for "healthy food" and "top CS conferences" respectively). This method works only for IS-A relationships whereas we consider all possible relationships.

Tylenda et al. [36] study how to find evidence for a triple inside a given document, but focus on the cases where entities and predicates all exist in some knowledge base; in our datasets all predicates are new and no more than 10% of the subject entities exist in *Freebase*. Knowledge verification has also been studied in the NLP community with emphasis on linguistic features [44]. Our framework allows applying extraction techniques proposed in [36, 44] whenever applicable.

Knowledge fusion. Besides the methods we have reviewed and evaluated in Section 4, a recent method TEM [46] considers in addition if the truth for a data item exists at all (*i.e.*, *date-of-death* does not exist for an alive person). It is mainly designed for single-truth scenario. The work in [22] studies the case where most sources provide very few triples, and thus source quality cannot be reliably estimated. Our framework can be enhanced by these techniques.

Knowledge collection on the long-tail. Our work is generally related to knowledge collection on the long-tail. A concept expansion system [37] finds entities belonging to a tail vertical by leveraging both structured and unstructured signals in web tables. An open IE system called ReNoun [42] focuses on extracting nominal attributes on the long-tail. The work in [3] answers queries regarding long-tail knowledge through log mining and crowdsourcing. Our work differs in that we consider the context of collecting knowledge by crowdsourcing, and focus on verifying the collected knowledge.

9. CONCLUSIONS

This paper studies the problem of verifying knowledge for long-tail verticals. We investigated seven evidence-collection approaches and four knowledge-fusion algorithms, and provided a detailed analysis of their performance on various long-tail verticals. Our experimental results inspire the FACTY knowledge-verification framework that first finds supporting evidence for a triple from various sources, and then applies knowledge fusion to predict the correctness of a triple based on the evidence. We showed that our framework significantly outperforms existing knowledge-verification techniques. Finally, a detailed loss analysis suggests future directions to improve knowledge verification for tail verticals.

10. REFERENCES

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. DBpedia: A nucleus for a Web of Open Data. *The Semantic Web*, 2007.

[2] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni. Open information extraction for the web. In *IJCAI*, 2007.

[3] M. S. Bernstein, J. Teevan, S. Dumais, D. Liebling, and E. Horvitz. Direct answers for search queries in the long tail. In *CHI*, 2012.

[4] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *SIGMOD*, 2008.

[5] M. J. Cafarella, A. Halevy, D. Z. Wang, E. Wu, and Y. Zhang. Webtables: Exploring the power of tables on the web. *PVLDB*, 2008.

[6] M. J. Cafarella, A. Y. Halevy, Y. Zhang, D. Z. Wang, and E. Wu. Uncovering the relational web. In *WebDB*, 2008.

[7] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka Jr, and T. M. Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, 2010.

[8] M.-C. De Marneffe, B. MacCartney, C. D. Manning, et al. Generating typed dependency parses from phrase structure parses. In *LREC*, 2006.

[9] X. L. Dong. Leave no valuable data behind: The crazy ideas and the business. *PVLDB*, 2016.

[10] X. L. Dong, L. Berti-Equille, and D. Srivastava. Integrating conflicting data: The role of source dependence. *PVLDB*, 2009.

[11] X. L. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang. Knowledge Vault: A web-scale approach to probabilistic knowledge fusion. In *KDD*, 2014.

[12] X. L. Dong, E. Gabrilovich, G. Heitz, W. Horn, K. Murphy, S. Sun, and W. Zhang. From data fusion to knowledge fusion. *PVLDB*, 2014.

[13] X. L. Dong, E. Gabrilovich, K. Murphy, V. Dang, W. Horn, C. Lugaresi, S. Sun, and W. Zhang. Knowledge-based trust: Estimating the trustworthiness of web sources. *PVLDB*, 2015.

[14] J. S. Downs, M. B. Holbrook, S. Sheng, and L. F. Cranor. Are your participants gaming the system?: Screening Mechanical Turk workers. In *CHI*, 2010.

[15] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates. Web-scale information extraction in KnowItAll. In *WWW*, 2004.

[16] A. Fader, S. Soderland, and O. Etzioni. Identifying relations for open information extraction. In *EMNLP*, 2011.

[17] J. Guo, G. Xu, X. Cheng, and H. Li. Named entity recognition in query. In *SIGIR*, 2009.

[18] A. Jain and M. Pennacchiotti. Open entity extraction from web search query logs. In *COLING*, 2010.

[19] H. Ji. Entity linking and wikification reading list, 2014. `http://nlp.cs.rpi.edu/kbp/2014/elreading.html`.

[20] Introducing the knowledge graph: Things, not strings. `https://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html`.

[21] F. Li, X. L. Dong, A. Langen, and Y. Li. Discovering multiple truths with a hybrid model. *CoRR*, abs/1705.04915, 2017.

[22] Q. Li, Y. Li, J. Gao, L. Su, B. Zhao, M. Demirbas, W. Fan, and J. Han. A confidence-aware approach for truth discovery on long-tail data. *PVLDB*, 2014.

[23] Q. Li, Y. Li, J. Gao, B. Zhao, W. Fan, and J. Han. Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation. In *SIGMOD*, 2014.

[24] X. Li, X. L. Dong, K. Lyons, W. Meng, , and D. Srivastava. Truth finding on the Deep Web: Is the problem solved? *PVLDB*, 2013.

[25] X. Li, W. Meng, and C. Yu. T-verifier: Verifying truthfulness of fact statements. In *ICDE*, 2011.

[26] Y. Li, J. Gao, C. Meng, Q. Li, L. Su, B. Zhao, W. Fan, and J. Han. A survey on truth discovery. *SIGKDD Exploration Newsletter*, 2016.

[27] A. Madaan, A. Mittal, G. R. Mausam, G. Ramakrishnan, and S. Sarawagi. Numerical relation extraction with minimal supervision. In *AAAI*, 2016.

[28] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013.

[29] M. Mintz, S. Bills, R. Snow, and D. Jurafsky. Distant supervision for relation extraction without labeled data. In *ACL*, 2009.

[30] F. Niu, C. Zhang, C. Ré, and J. W. Shavlik. DeepDive: Web-scale knowledge-base construction using statistical learning and inference. *VLDS*, 2012.

[31] J. Pasternack and D. Roth. Latent credibility analysis. In *WWW*, 2013.

[32] R. Pochampally, A. Das Sarma, X. L. Dong, A. Meliou, and D. Srivastava. Fusing data with correlations. In *SIGMOD*, 2014.

[33] M. Samadi, P. Talukdar, M. Veloso, and M. Blum. ClaimEval: Integrated and flexible framework for claim evaluation using credibility of sources. In *AAAI*, 2016.

[34] Understand your world with bing. `https://blogs.bing.com/search/2013/03/21/understand-your-world-with-bing`.

[35] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A core of semantic knowledge. In *WWW*, 2007.

[36] T. Tylenda, Y. Wang, and G. Weikum. Spotting facts in the wild. In *Workshop on Automatic Creation and Curation of Knowledge Bases*, 2014.

[37] C. Wang, K. Chakrabarti, Y. He, K. Ganjam, Z. Chen, and P. A. Bernstein. Concept expansion using web tables. In *WWW*, 2015.

[38] X. Wang, Q. Z. Sheng, X. S. Fang, L. Yao, X. Xu, and X. Li. An integrated Bayesian approach for effective multi-truth discovery. In *CIKM*, 2015.

[39] Z. Wang, J. Zhang, J. Feng, and Z. Chen. Knowledge Graph and text jointly embedding. In *EMNLP*, 2014.

[40] F. Wu, R. Hoffmann, and D. S. Weld. Information extraction from wikipedia: Moving down the long tail. In *KDD*, 2008.

[41] W. Wu, H. Li, H. Wang, and K. Q. Zhu. Probase: A probabilistic taxonomy for text understanding. In *SIGMOD*, 2012.

[42] M. Yahya, S. E. Whang, R. Gupta, and A. Halevy. Renoun: Fact extraction for nominal attributes. In *EMNLP*, 2014.

[43] X. Yin, J. Han, and P. S. Yu. Truth discovery with multiple conflicting information providers on the web. In *KDD*, 2007.

[44] D. Yu, H. Huang, H. J. Taylor Cassidy, C. Wang, S. Zhi, J. Han, C. Voss, and M. Magdon-Ismail. The wisdom of minority: Unsupervised slot filling validation based on multi-dimensional truth-finding. In *COLING*, 2014.

[45] B. Zhao, B. I. P. Rubinstein, J. Gemmell, and J. Han. A bayesian approach to discovering truth from conflicting sources for data integration. *PVLDB*, 2012.

[46] S. Zhi, B. Zhao, W. Tong, J. Gao, D. Yu, H. Ji, and J. Han. Modeling truth existence in truth discovery. In *KDD*, 2015.