Crowd-sourced knowledge graph extension: a belief revision based approach

Artem Revenko and **Albin Ahmeti** and **Martin Schauer**
Semantic Web Company

Marta Sabou
Technical University of Vienna

Abstract

Knowledge graphs are gaining popularity as key ingredients of many advanced applications. For many applications there is a need of having the common sense knowledge that is not domain specific, and, therefore, can be provided by non-experts. In this paper we introduce a novel crowd-sourcing approach that allows the crowdworkers to provide their update in a simplistic intuitive form without having the information about the knowledge already contained in the graph. The approach roots in belief revision theory and is capable of analyzing the user input, identifying the compliance with the existing structure and singling out new suggestions. When providing the update and upon submission the crowdworkers obtain intuitive color-coded feedback on their input w.r.t. consistency and discrepancies with the existing knowledge. This feedback enables the educational aspect of the approach. The approach guarantees the consistency of the crowd-sourced knowledge when it is being integrated into the knowledge graph.

Introduction

Knowledge graphs (KG) are a novel kind of data structures that enable the creation of intelligent applications such as advanced search engines, recommender systems and question answering systems. Recently, knowledge graphs were defined as "a set of interconnected typed entities and their attributes" (Pan et al. 2017), where an ontology defines the vocabulary of the graph. For example, box 4 in Figure 1, shows an example knowledge graphs including entities such as *bat* or *cat*, which can be of type *Mammal*. Lines between the entities denote the type relation (*cat* is of type *Mammal*) or other relations. Some representative examples of knowledge graphs include DBpedia[1], which is a structured representation of Wikipedia data, or EuroVoc[2], a multi-lingual thesaurus of all activities of the European Union.

A critical problem in the life-cycle of a KG is extending and keeping it up-to-date. This is a costly and time-consuming task that is hard to achieve within the boundaries of one organization. Therefore, in this paper, we investigate the following research question:

RQ1: How to use crowd-sourcing to extend a large KG?

Involving crowds into the extension of knowledge structures leads to opportunities in terms of educating them in the subject domain covered by the knowledge structure. Therefore, an additional research question addressed is:

RQ2: How to educate crowdworkers about the subject domain of the KG while they are extending it?

We address these research questions as part of the European PROFIT project[3], where a platform to promote financial awareness and stability is developed, and we are designing a web-based system which collects extensions to a large knowledge graph from a crowd of citizens which use this platform[4]. To address RQ2, the tool provides the users feedback about discrepancies between their vision of the domain and the existing knowledge graph of the domain.

Our approach to ensure this capability is the use of belief revision theory (Gärdenfors 2003). Accordingly, the problem is translated into the belief revision problem where the existing knowledge graph is "mapped" to the world W, and the model created by the user is "mapped" to the update U. Therefore, we enable the analysis of the differences and distances between the user provided update and the existing knowledge graph (world).

Novelty in the proposed work is the use of Semantic Web technologies to formally represent the knowledge structure that is extended. This enables the system to automatically reason upon user suggestions to judge their correctness, which is a pre-requisite to providing feedback to users (thus educating them) as well as to integrating this knowledge in the KG in a way that it remains correct (i.e., consistent). The use of belief revision theory to inform the reasoning mechanisms is another novelty. As an important consequence the tool allows for an additional *implicit* voting mechanism by comparing the overlapping parts in the users' updates. Overall, the tool illustrates the use of Semantic Web reasoning capabilities to support a Human Computation task, a research line which has only been weakly covered so far (Sabou et al. 2018a).

In the rest of the paper we detail the problem setting and sketch the general workflow followed by the tool, highlight-

[1] dbpedia.org

[2] eurovoc.europa.eu

[3] platform.projectprofit.eu

[4] A demo of the system is available: research.semantic-web.com/crowd-sourcing/

ing the role and benefits of using belief revision.

Problem Setting

A core component of the knowledge graph that is to be extended is an ontology $\mathcal{O}$. As ontologies are often encoded in terms of the OWL[5] knowledge representation language, we define the ontology by relying on the OWL terminology. The ontology holds definitions of classes and relations between them. Let A and B be two *classes*. Let a be an *instance*. The statement $a \in A$ is interpreted as a is of type class A and is called a *class assertion*. Let $R \subseteq A \times B$ be a *relation* between the two classes. For $a \in A,\ b \in B$ one may assert $R(a, b)$, i.e. a and b are in relation R; this is called a *relation assertion*. Moreover, every instance can have *attributes* whose values are constants (integers, strings, dates, etc). Statements about class, relation or attribute assertions are atomic knowledge structures that we refer to as *triples*.

The ontology $\mathcal{O}$ is pre-defined and fixed for the crowdsourcing process, i.e. the users cannot suggest new classes or new relations. Nevertheless, users can suggest new class assertions, new relation assertions, new attribute values.

The basis of our ontology is the Simple Knowledge Organization Scheme[6]. Instances are called *concepts* in SKOS notation. SKOS allows for defining a thesaurus with hierarchical relations *broader* `skos:broader` and *narrower* `skos:narrower`. Moreover, in a SKOS thesaurus every instance may have different labels which denote synonyms of that instance. These labels are important in several advanced applications where they support tasks such as finding instance mentions in text or disambiguation. In the developed crowd-sourced application the users can provide suggestions on new instance labels as well.

Approach

The typical workflow of our approach consists of the following phases, as illustrated in Fig. 1:

Collect The user inputs their update U (box 1 in Fig.1). The proposed tool allows users to provide input without referring to the existing knowledge graph, i.e. the user is not forced into any particular vision of the subject domain. Users are encouraged to convey their input in a free form, starting from an empty canvas and creating new triples.

In order to enable such freedom and flexibility it is necessary to (1) identify and resolve inconsistencies between U and W and (2) compute overlaps, contradictions and novelties w.r.t. the existing knowledge. This is performed in the analysis phase, described next.

Analyze and Provide Feedback The user's update U is analyzed against the world W (box 2 in Fig. 1) in order to identify *new triple suggestions* and update *the trust thresholds* of these triples, as we will discuss in more detail in the next section on inconsistency detection.

The user's input is analyzed in real time and all the inconsistencies in his provided knowledge are highlighted. The user is not able to finalize the input unless he resolves all the intrinsic inconsistencies in U. Each inconsistency features a description for user convenience. Upon submitting the input, for educational purposes, the user obtains color-coded feedback on his submission in terms of new (blue), confirming (green) and contradicting (red) triples in his input. This consistency checking mechanisms is employed during both the **collect** and the **integrate** phases of the workflow.

Vote The users vote on triples suggested by other users (box 3 in Fig. 1). Voting mechanisms are introduced as an answer to RQ2 since they initiate interaction and opinion exchange with other users and/or experts in the field. Two types of voting are implemented. First, in the dedicated page every authorized user can vote *explicitly*. The user can vote on triples contributed by others only once. The user can change the vote (from upvote to downvote and vice versa) or withdraw the vote. If different users suggest the same new triple then an *implicit* voting mechanism gets activated. When the difference between upvotes and downvotes reaches the trust threshold the triple becomes *accepted* and the **integrate** gets activated.

The users cannot upvote or downvote their own triples.

Integrate The new and verified crowd-sourced knowledge is integrated into the world W (box 4 in Fig. 1).

Inconsistency detection and management

Core to our approach is identifying differences between the existing (W) and newly contributed (U) knowledge and assessing whether inconsistencies arise, as these should be avoided. An *inconsistency* is defined as a violation of axioms. Since the ontology is defined using SKOS, we take SKOS axioms into account[7]. Of all axioms the following two could be violated by the user input:

1. "Disjointness of `skos:related` and `skos:broaderTransitive`. This specification treats the hierarchical and associative relations as fundamentally distinct in nature. Therefore a clash between hierarchical and associative links is not consistent with the SKOS data model." In other words, if instance a is `skos:broader` of b then the two instances cannot be `skos:related`
2. "Cycles in the Hierarchical Relation (`skos:broaderTransitive` and Reflexivity)". For example, a `skos:broader` b and b `skos:broader` a. We prohibit this kind of hierarchical cycles for our application.

Furthermore we introduce two additional axioms and we do not allow to submit the update unless it is free from these two types of inconsistencies:

3. In U there should not be any disconnected instances. We introduce this requirement to avoid abandoned instances.

[5] `www.w3.org/OWL`

[6] `www.w3.org/2004/02/skos`

[7] `www.w3.org/TR/skos-reference/#semantic-relations`

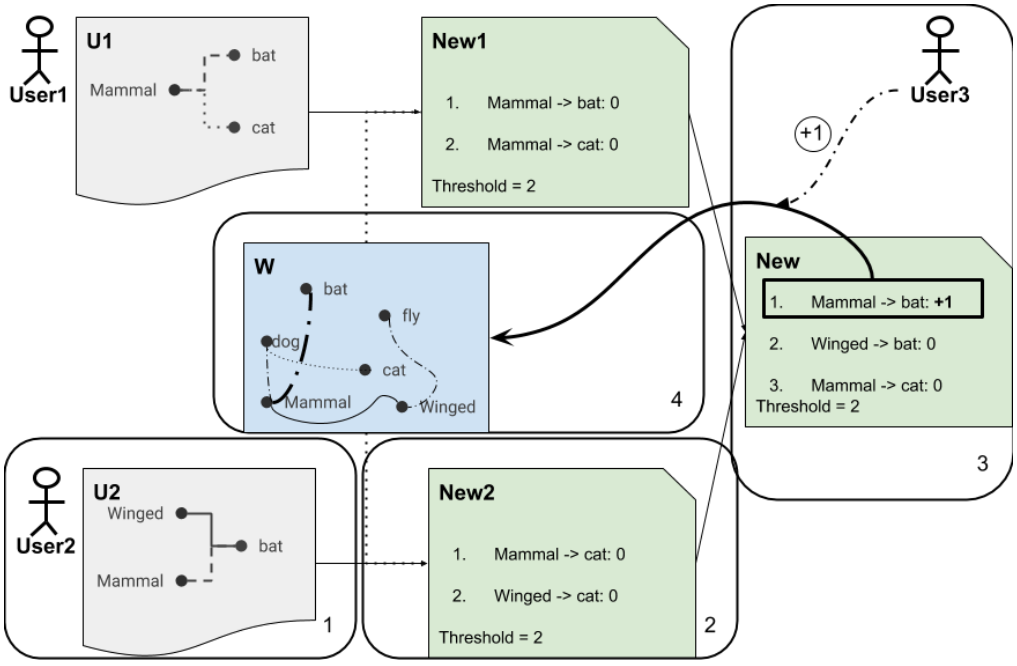

Figure 1: Crowd-sourcing workflow. User 1 and User 2 submit their updates (**collect**). Let the threshold needed to accept each new suggestion be 2 (**analyze**). Both updates contain two new suggestions that extend the world. One suggestion is overlapping in the updates (S_1:= Mammal $\rightarrow$ bat), it is implicitly upvoted (**vote**). User 3 upvotes the same suggestion S_1 explicitly through the user interface (**vote**), therefore S_1 gets 2 upvotes, reaches the threshold and it is added to the world (**integrate**).

4. Every new instance in U should have a broader instance. This condition requires every new instance to be integrated into the hierarchical structure.

We distinguish between two sources of inconsistencies:

- *intrinsic inconsistency*, an inconsistency in the update itself; any of the four identified inconsistency types above may appear here;
- *general inconsistency*, an inconsistency that is only present in the union of W and U and does not appear neither in W alone nor in U alone; only violation of axioms 1 and 2 may appear as general inconsistencies.

For the sake of identifying the discrepancies between W and U only the general inconsistencies are taken into account. As follows from the definitions of axioms 1 and 2, it is always possible to identify the triples in U that cause these inconsistencies; these triples form the set of *contradicting* triples $\mathcal{T}_{contra}$. The set of *confirming* triples $\mathcal{T}_{conf}$ contains the triples contained in both W and U. The set of new triples $\mathcal{T}_{new}$ contains all the triples that are contained in U but not in W.

The new, confirming, and contradicting sets of triples enable us to give the user a feedback on his input w.r.t. existing knowledge and quantify the correspondence between the update and the world. Moreover, we can relate the updates of different users and enable implicit voting between updates in case the sets of new triples overlap. Now we are in position to compute a distance between U and W and introduce a threshold for accepting the new triples.

Threshold

The threshold t, denoting the trust level of a triple, depends on the number of contradicting triples $|\mathcal{T}_{contra}|$ and confirming triples $|\mathcal{T}_{conf}|$. In order to encourage users to provide larger input and avoid updates with only new facts we introduce a "penalty" p; if the user uses less than p triples from the existing knowledge graph then the user's threshold is increased. Moreover, each contradicting triple indicates a deviation from the existing knowledge, hence the triples from the update need to obtain additional support from other users to get accepted. Finally, in order to prevent any update to be accepted automatically we add 1 to the resulting threshold. The resulting formula:

$$t = \max(0, p - |\mathcal{T}_{conf} \cup \mathcal{T}_{contra}|) + 2 * |\mathcal{T}_{contra}| + 1 \quad (1)$$

Example 1 *Let $p = 5$, $|\mathcal{T}_{conf}| = 3$ and $|\mathcal{T}_{contra}| = 1$, i.e. the user has provided 3 confirming triples and 1 contradicting. Then $t = \max(0, 5 - (3 + 1)) + 2 * 1 + 1 = 4$, i.e. at least 4 upvotes are needed to accept the new triples.*

Future Work

In the future we plan to improve the usability and personalization of the tool by enabling users to start with a pre-filled canvas. The pre-filled canvas may contain the triples of most interest to the crowd-sourcing process or to the user. To that end, we will reuse principles outlined in (Wohlgenannt, Sabou, and Hanika 2016) and (Sabou et al. 2018b).

References

Gärdenfors, P. 2003. *Belief revision*, volume 29. Cambridge University Press.

Pan, J. Z.; Vetere, G.; Gomez-Perez, J. M.; and Wu, H. 2017. *Exploiting Linked Data and Knowledge Graphs in Large Organisations*. Springer, 1st edition.

Sabou, M.; Aroyo, L.; Bozzon, A.; and Qarout, R. K. 2018a. Semantic Web and Human Computation: the Status of an Emerging Field. *Semantic Web* 9(3):1–12.

Sabou, M.; Winkler, D.; Biffl, S.; and Penzerstadler, P. 2018b. Verifying conceptual domain models with human computation: A case study in software engineering. In *The sixth AAAI Conference on Human Computation and Crowdsourcing*.

Wohlgenannt, G.; Sabou, M.; and Hanika, F. 2016. Crowd-based ontology engineering with the uComp Protégé plugin. *Semantic Web* 7(4):379–398.