Top K Hypotheses Selection on a Knowledge Graph

Kexuan Sun Krishna Akhil Maddali Shriraj Salian T. K. Satish Kumar
Information Sciences Institute, University of Southern California, Marina del Rey, CA 90292, USA
{kexuansu, maddali, salian}@usc.edu, tkskwork@gmail.com

Abstract

A Knowledge Graph (KG), popularly used in both industry and academia, is an effective representation of knowledge. It consists of a collection of knowledge elements, each of which in turn is extracted from the web or other sources. Information extractors that use natural language processing techniques or other complex algorithms are usually noisy. That is, the vast number of knowledge elements extracted from the web may not only be associated with different confidence values but may also be inconsistent with each other. Many applications such as question answering systems that are built on top of large-scale KGs are required to reason efficiently about these confidence values and inconsistencies. In addition, they are required to incorporate ontological constraints in their reasoning. One way to do this is to extract a subgraph of a KG that is consistent with the ontological constraints and is of maximum total confidence value. Such a subgraph is referred to as the top hypothesis and is combinatorially hard to find. In this paper, we introduce an algorithmic framework for efficiently addressing the combinatorial hardness and selecting the top K hypotheses. Our approach is based on powerful algorithmic techniques recently invented in the context of the Weighted Constraint Satisfaction Problem (WCSP).

Introduction

A *Knowledge Graph (KG)* has been widely accepted as a prominent and effective representation of knowledge since its debut in 2012. KGs allow users to search for "things" instead of just "strings" (Singhal 2012). Therefore, they have been used to enhance search engines. Conceptually, a KG is a graphical representation of facts where nodes represent entities and directed edges represent relations between entities. The types and properties of entities and relations are defined in an ontology that covers various topics (Paulheim 2017).

In a KG, the facts can come from knowledge bases such as DBpedia (Lehmann et al. 2014) or from web data, e.g., NELL (Carlson et al. 2010).

Despite their prominence, KGs bear a few fundamental obstacles: noisy information extractors and unreliable and inconsistent knowledge embodied in documents (Bordes and Gabrilovich 2014). These obstacles severely undermine our ability to reason at a corpus level. Therefore, it is imperative for us to design strong inference algorithms.

One approach that renders KGs useful is to annotate a confidence value with each fact and ontological constraint. A calculus of confidence values can then be formalized to include reasoning about uncertain facts and hard constraints. In our framework, the task of reasoning with uncertainties, noise and inconsistencies is reformulated to be the combinatorial task of extracting a subgraph of the KG that is consistent with the ontological constraints and is of maximum total confidence value. Such a subgraph is referred to as the top hypothesis. Finding the top K hypotheses qualifies as a task in *knowledge graph refinement*, a topic of research that focuses on improving the consistency and correctness of a KG despite the noise of the extracted facts (Paulheim 2017).

Unfortunately, finding the top hypothesis is combinatorially NP-hard; finding the top K hypotheses is even harder. Nonetheless, finding the top K hypotheses is useful for generating alternative interpretations. Moreover, it also allows users to pick or reject a hypothesis with a declared reason that can then be incorporated for further reasoning.

In this paper, we introduce an algorithmic framework for efficiently selecting the top K hypotheses in a KG. In our approach, the top K hypotheses correspond to the top K solutions of a *Weighted Constraint Satisfaction Problem (WCSP)* instance in which Boolean variables correspond to edges in the KG and weighted constraints model uncertainties and ontological constraints. We use a solver called `WCSPLift` (Xu, Koenig, and Kumar 2017) that constructs the *Constraint Composite Graph (CCG)* (Kumar 2008a; 2008b) of a WCSP instance, compiles a substrate *Minimum Weighted Vertex Cover (MWVC)* problem, and then uses powerful algorithmic techniques such as the Nemhauser-Trotter reduction and message passing for solving WCSPs (Xu, Kumar, and Koenig 2017; Fioretto et al. 2018). Using this novel approach, we discuss preliminary results supplemented by a case study.

Related Work

Many methods for *knowledge graph refinement* have been proposed to resolve inconsistencies on KGs. (Paulheim 2017) divides these methods into two broad categories: *completion* and *correction*. The former includes completing type assertions and predicting relations, while the latter is mainly about detecting errors. (Namata, Kok, and Getoor 2011) introduces *graph identification* to address the problem of identifying "true" underlying networks from noisy data. After that, the related problem of *knowledge graph identification* from noisy facts is addressed in (Pujara et al. 2013) using *Probabilistic Soft Logic (PSL)* (Bröcheler, Mihalkova, and Getoor 2012). A similar problem of automatically cleaning a noisy knowledge base is addressed in (Jiang, Lowd, and Dou 2012) using *Markov Logic Networks (MLNs)* (Richardson and Domingos 2006). These problems correspond to the identification of the top hypothesis in our framework. Our approach is different from these methods not only because it generates the top K hypotheses but also because it uses algorithmic techniques that focus on kernelization instead of relaxation. Methods based on relaxation may not keep proper semantics because of the fractional values of variables.

Background on WCSPs

The WCSP (Bistarelli et al. 1999) is a combinatorial optimization problem that generalizes the *Constraint Satisfaction Problem (CSP)*. It is formally characterized by a triplet $\mathcal{B} = (\mathcal{X}, \mathcal{D}, \mathcal{C})$ where $\mathcal{X} = \{X_1, \cdots X_N\}$ is a set of N variables with an associated set of domains $\mathcal{D} = \{D_1, \cdots D_N\}$. $\mathcal{C} = \{C_1, \cdots C_M\}$ is a set of M weighted constraints. For each weighted constraint $C_i \in \mathcal{C}$ defined on a subset of variables $\mathcal{S}_i \subseteq \mathcal{X}$, each possible combination of values to variables in $\mathcal{S}_i$ has an associated non-negative weight (cost). The goal is to find an assignment of values to all variables in $\mathcal{X}$ from their respective domains such that the sum of the weights identified by each constraint in $\mathcal{C}$ is minimized. This task is equivalent to calculating $\arg\min_{a \in \mathcal{A}(\mathcal{X})} \sum_{C_i \in \mathcal{C}} E_{C_i}(a|C_i)$, where $\mathcal{A}(\mathcal{X})$ represents the set of all complete assignments, i.e., the Cartesian product of the domains of all variables $|D_1| \times \cdots |D_N|$. $a|C_i$ represents the projection of a complete assignment a onto the subset of variables in C_i. E_{C_i} is a function that maps each assignment $a|C_i$ to its associated weight. WCSPs are NP-hard to solve in general. Boolean WCSPs are WCSPs in which all variables are Boolean. They are representationally as powerful as WCSPs and are also NP-hard to solve.

Exploiting Structure in WCSPs

Although WCSPs are NP-hard, special structure in individual instances can be exploited for computational benefits. There are two kinds of structure in WCSP instances: (a) *micro structure* (*numerical structure*) that captures *how* variables interact with each other; and (b) *macro structure* (*graphical structure*) that captures *which* variables interact with each other (Dechter 1992). These two kinds of structure have been traditionally exploited in different ways.

The idea of the CCG provides a unifying framework for simultaneously exploiting the graphical structure of the variable interactions in a Boolean WCSP and the numerical structure of the weighted constraints in it. The task of solving the Boolean WCSP can be reformulated as the task of finding an MWVC on its associated CCG. Under this reformulation, various properties of the MWVC problem can be computationally leveraged. For example, the half-integrality property of the MWVC problem can be converted to a kernelization procedure, a maxflow-based polynomial-time preprocessing algorithm that fixes the optimal values of a large subset of the variables before search starts.

This and other techniques for solving WCSPs are implemented in `WCSPLift`. In this paper, we use it to generate the top K hypotheses for a KG after casting them as the top K solutions of a WCSP.

Hypotheses Selection Using WCSPs

The nodes of a KG can represent different objects, such as real-world entities, events or constants like certain numbers and strings. The directed edges between nodes of the KG represent the semantic relations between them. Edges can be of two kinds: (a) from an entity or event node to a constant node and (b) from an event node to an entity node. An edge of the first kind describes the entity or the event from which it emanates. For example, an edge labeled IS_A from an entity node with label Mike to a constant node with label Person indicates that Mike is a Person. An edge of the second kind indicates that the entity node is an argument for the event node. For example, an edge labeled Life.Born_Person from an event node to an entity node with label Mike indicates that Mike is an object of the argument Life.Born_Person for this event.

Hypotheses selection in our framework consists of three steps: (1) represent uncertainties in facts; (2) represent ontological constraints; and (3) use `WCSPLift` to generate and score the top K hypotheses.

Representing Uncertainties in Facts

A KG can contain edges that represent uncertain facts. For example, suppose that the KG contains an edge annotated with confidence value 0.8 and label Life.Born_Person. If this edge is from an event node with label Born to an entity node with label Mike, it represents the fact that the person born in this event is Mike with confidence value 0.8.

Our approach does not naively deem an uncertain fact in the KG true in a hypothesis. This is because it has a certain annotated confidence value. Instead, a hypothesis decides the truth value of each fact. Therefore, each uncertain fact can be treated as a Boolean variable where "0" and "1" represent "False" and "True," respectively. For each Boolean variable v corresponding to an uncertain fact with confidence value p, say, treated as a probability, we add a unary weighted constraint with weights $-\ln(1-p)$ and $-\ln(p)$ against the assignments $v = 0$ and $v = 1$, respectively. For the uncertain fact associated with the edge labeled Life.Born_Person in the previous example, Table 1a shows the unary weighted constraint created for it.

Representing Ontological Constraints

To resolve inconsistencies between uncertain facts, we also need to consider ontological constraints. Typically, we enforce ontological constraints based on the corresponding ontology of the KG. For example, two common types of ontological constraints are cardinality and domain constraints.

- *Cardinality Constraint*: The arguments for some types of events can only be present a limited number of times. For example, suppose the argument Life.Born_Person

Table 1: Shows the weighted constraints created in our framework. (a) shows a unary weighted constraint representing an uncertain fact and (b) shows a binary weighted constraint representing an ontological constraint.

(a) v represents the edge labeled Life.Born_Person from Born to Mike.

v	
True	$-\ln(0.8) = 0.2$
False	$-\ln(0.2) = 1.6$

(b) v_1 and v_2 represent the edges labeled Life.Born_Person from Born to Mike and Mary, respectively.

v_1 \ v_2	True	False
True	∞	$\ln(3) = 1.1$
False	$\ln(3) = 1.1$	$\ln(3) = 1.1$

can ontologically be present only once for an event. This means that there can be at most one edge labeled Life.Born_Person emanating from an event node.

- *Domain Constraint*: Each argument of an event can only be specific types of objects. For example, the argument Life.Born_Person can only be an object with type Person. Suppose there is an edge labeled Life.Born_Person from an event node to an entity node. Any edge labeled IS_A emanating from the entity node needs to terminate at a constant node labeled Person.

Such ontological constraints can be encoded as weighted constraints in our framework. Consider an ontological constraint involving Boolean variables $v_1, \cdots v_k$, where each v_i corresponds to an uncertain fact in the KG. For example, the cardinality constraint can enforce a mutual exclusion between the edges labeled Life.Born_Person from the event Born to the entities Mike and Mary. A corresponding weighted constraint that is built on the same variables resembles a truth table on these variables. In other words, the truth table entries that evaluate to "False" are given an infinite weight, and those that evaluate to "True" are given a uniform positive weight. This positive weight depends on the semantics and calculus of confidence values. If treated as probabilities, then the positive weight is set to $\ln(s)$ where s is the number of truth table entries that evaluate to "True." This value is derived from the principle of insufficient reason. Table 1b shows the weighted constraint created from the mutual exclusion involving Mike and Mary.

Sometimes, ontological constraints are also uncertain. For example, we might be interested in reasoning with general knowledge that restricts any person to have at most two jobs but only with confidence value 0.9. Such uncertain ontological constraints can also be encoded as weighted constraints in our framework by assigning a uniform positive finite weight to all truth table entries that evaluate to "True" and a different uniform positive finite weight to all truth table entries that evaluate to "False".

Generating and Scoring the Top K Hypotheses

As described above, each uncertain fact and each ontological constraint in a KG can be reformulated to a weighted constraint. Suppose the confidence values are treated as probabilities and the weights are obtained by taking the negative natural logarithm of the probabilities. Maximization over the product of probabilities is mapped to minimization over the sum of weights, and all weights are non-negative as all probabilities are in the interval $[0, 1]$. For a different semantics of the confidence values, a different formulation can be used.

Since the top hypothesis of a KG corresponds to the optimal solution of its associated WCSP, we can invoke `WCSPLift` to efficiently perform combinatorial search in the space of all hypotheses. However, we are often interested in generating the top K hypotheses for a specified value of K. This is important because it generates alternative interpretations of KGs and allows users to pick or reject hypotheses with additional reasons. Such reasons can also be incorporated back into the KG for further inference.

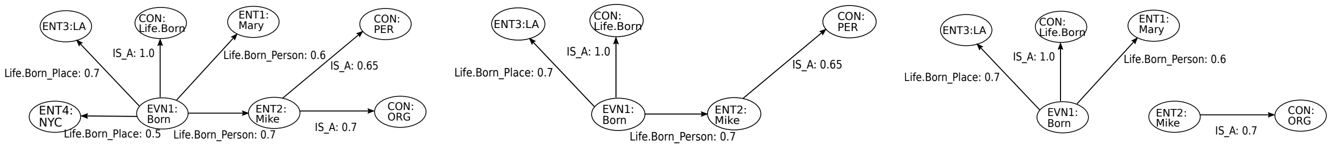
To generate the k^{th} hypothesis, we construct a new WCSP instance that includes all of the original weighted constraints as well as some additional constraints that disallow the previously generated $k - 1$ hypotheses. Therefore, K WCSP instances are solved in sequence to generate the top K hypotheses. The score of the k^{th} hypothesis can be defined as e^{-w}, where w is the total weight of the optimal solution of the k^{th} WCSP instance. The exponential transformation is used to revert the negative natural logarithm transformation.

Figure 1 shows a small example. The KG contains uncertain facts with different confidence values. The event of a birth occurs either in LA or in NYC. The person born is either Mike or Mary. The entity Mike is an organization (ORG) or a person (PER). The following ontological constraints are used: (a) Life.Born_Person(EVN1, ENT1) and Life.Born_Person(EVN1, ENT2) cannot both be "True" because EVN1 refers to a specific person; (b) Life.Born_Place(EVN1, ENT3) and Life.Born_Place(EVN1, ENT4) cannot both be "True" because EVN1 occurs in at most one city; (c) IS_A(ENT2, PER) and IS_A(ENT2, ORG) cannot both be "True" because ENT2 is either a person or an organization; and (d) IS_A(ENT2, ORG) and Life.Born_Person(EVN1, ENT2) cannot both be "True" because the argument Life.Born_Person can only be a person (PER). Figure 1 also shows the top 2 hypotheses on the KG enforcing these ontological constraints.

Case Study

The task of selecting the top K hypotheses on a KG is central to DARPA's Active Interpretation of Disparate Alternatives (AIDA) program (Onyshkevych 2017). This program is intended to generate alternative interpretations of structured and unstructured data. Generating the top K hypotheses is critical for maintaining and refining understanding of events, situations and trends in a variety of domains.

In the first stage, the KG is constructed by applying multiple natural language processing methods. For example, named entities and nominal mentions are extracted using BiLSTM (Hochreiter and Schmidhuber 1997), and events and arguments are extracted using GAIL (Ho and Ermon 2016). In the second stage, various techniques such as heuristic rules and connected component analysis are applied to address problems related to document-level coreference resolution and corpus-level entity linking. The details of these methods can be found in (Zhang et al. 2018). `WCSPLift`



(a) The given KG

(b) The top hypothesis

(c) The second best hypothesis

Figure 1: Illustrates top K hypotheses selection on a KG. Each edge is annotated with a label as well as a confidence value. Event, entity and constant nodes have labels with prefix EVN, ENT and CON, respectively.

runs successfully on the KG produced after the first two stages to generate the top K hypotheses as required in the program. The KG contains about 1760682 nodes and 1922206 edges; and `WCSPLift` takes about 30 minutes to generate each of the top K hypotheses.

Conclusions and Future Work

In this paper, we introduced the idea of using recent developments in WCSP technology to generate the top K hypotheses on a KG. In particular, we used `WCSPLift`, a powerful solver that can quickly explore the combinatorial space of all hypotheses and return the top K of them. Future work involves applying our method to KGs from various domains.

Acknowledgment This material is based on research sponsored by DARPA under agreement number FA8750-18-2-0014. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government. We also thank Hong Xu and Zhi Wang for their helpful comments.

References

Bistarelli, S.; Montanari, U.; Rossi, F.; Schiex, T.; Verfaillie, G.; and Fargier, H. 1999. Semiring-based CSPs and valued CSPs: Frameworks, properties, and comparison. *Constraints* 4(3):199–240.

Bordes, A., and Gabrilovich, E. 2014. Constructing and mining web-scale knowledge graphs: KDD 2014 tutorial. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1967–1967.

Bröcheler, M.; Mihalkova, L.; and Getoor, L. 2012. Probabilistic similarity logic. *CoRR* abs/1203.3469.

Carlson, A.; Betteridge, J.; Wang, R. C.; Hruschka, Jr., E. R.; and Mitchell, T. M. 2010. Coupled semi-supervised learning for information extraction. In *ACM International Conference on Web Search and Data Mining*, 101–110.

Dechter, R. 1992. Constraint networks. *Artificial Intelligence* 49:61–95.

Fioretto, F.; Xu, H.; Koenig, S.; and Kumar, T. K. S. 2018. Solving multiagent constraint optimization problems on the constraint composite graph. In *International Conference on Principles and Practice of Multi-Agent Systems*, 106–122.

Ho, J., and Ermon, S. 2016. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems 29*. Curran Associates, Inc. 4565–4573.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural Comput.* 9(8):1735–1780.

Jiang, S.; Lowd, D.; and Dou, D. 2012. Learning to refine an automatically extracted knowledge base using Markov logic. In *IEEE International Conference on Data Mining*, 912–917.

Kumar, T. K. S. 2008a. A framework for hybrid tractability results in Boolean weighted constraint satisfaction problems. In *International Conference on Principles and Practice of Constraint Programming.*, 282–297.

Kumar, T. K. S. 2008b. Lifting techniques for weighted constraint satisfaction problems. In *International Symposium on Artificial Intelligence and Mathematics*.

Lehmann, J.; Isele, R.; Jakob, M.; Jentzsch, A.; Kontokostas, D.; Mendes, P.; Hellmann, S.; Morsey, M.; Van Kleef, P.; Auer, S.; and Bizer, C. 2014. DBpedia - a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web Journal* 6.

Namata, G. M.; Kok, S.; and Getoor, L. 2011. Collective graph identification. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 87–95.

Onyshkevych, B. 2017. Active interpretation of disparate alternatives. https://www.darpa.mil/program/active-interpretation-of-disparate-alternatives.

Paulheim, H. 2017. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web* 8:489–508.

Pujara, J.; Miao, H.; Getoor, L.; and Cohen, W. 2013. Knowledge Graph Identification. In *International Semantic Web Conference*.

Richardson, M., and Domingos, P. 2006. Markov logic networks. *Mach. Learn.* 62(1-2):107–136.

Singhal, A. 2012. Introducing the knowledge graph: Things, not strings. http://goo.gl/zivFV.

Xu, H.; Koenig, S.; and Kumar, T. K. S. 2017. A constraint composite graph-based ILP encoding of the Boolean weighted CSP. In *International Conference on Principles and Practice of Constraint Programming*, 630–638.

Xu, H.; Kumar, T. K. S.; and Koenig, S. 2017. The Nemhauser-Trotter reduction and lifted message passing for the weighted CSP. In *International Conference on Integration of Artificial Intelligence and Operations Research Techniques in Constraint Programming*, 387–402.

Zhang, T.; Subburathinam, A.; Shi, G.; Huang, L.; Lu, D.; Pan, X.; Li, M.; Zhang, B.; Wang, Q.; Whitehead, S.; Ji, H.; Zareian, A.; Akbari, H.; Chen, B.; Zhong, R.; Shao, S.; Allaway, E.; Chang, S.-F.; McKeown, K.; Li, D.; Huang, X.; Sun, K.; Peng, X.; Gabbard, R.; Freedman, M.; Kejriwal, M.; Nevatia, R.; Szekely, P.; Kumar, T. S.; Sadeghian, A.; Bergami, G.; Dutta, S.; Rodriguez, M.; and Wang, D. Z. 2018. GAIA - a multi-media multi-lingual knowledge extraction and hypothesis generation system. In *Text Analysis Conference Knowledge Base Population Workshop*.