Large-Scale Knowledge Graph Identification using PSL

Jay Pujara JAY@CS.UMD.EDU
Hui Miao HUI@CS.UMD.EDU
Lise Getoor GETOOR@CS.UMD.EDU
Department of Computer Science University of Maryland College Park, MD 20742

William Cohen WCOHEN@CS.CMU.EDU
Machine Learning Department, Carnegie Mellon University Pittsburgh, PA 15213

Abstract

Building a web-scale knowledge graph, which captures information about entities and the relationships between them, represents a formidable challenge. While many large-scale information extraction systems operate on web corpora, the candidate facts they produce are noisy and incomplete. To remove noise and infer missing information in the knowledge graph, we propose *knowledge graph identification*: a process of jointly reasoning about the structure of the knowledge graph, utilizing extraction confidences and leveraging ontological information. Scalability is often a challenge when building models in domains with rich structure, but we use *probabilistic soft logic* (PSL), a recently-introduced probabilistic modeling framework which easily scales to millions of facts. In practice, our method performs joint inference on a real-world dataset containing over 1M facts and 80K ontological constraints in 12 hours and produces a high-precision set of facts for inclusion into a knowledge graph.

1. Introduction

The web is a vast repository of knowledge, but automatically extracting that knowledge, at scale, has proven to be a formidable challenge. A number of recent evaluation efforts have focused on automatic knowledge base population (Ji et al., 2011; Artiles & Mayfield, 2012), and many well-known broad domain and open information extraction systems exist, including the Never-Ending Language Learning (NELL) project (Carlson et al., 2010), OpenIE (Etzioni et al., 2008), and efforts at Google (Pasca et al., 2006), which use a variety of techniques to extract new knowledge, in the form of facts, from the web. These facts are interrelated, and hence, recently this extracted knowledge has been referred to as a *knowledge graph* (Singhal, 2012). Unfortunately, most web-scale extraction systems do not take advantage of the knowledge graph. Millions of facts and the many dependencies between them pose a scalability challenge. Accordingly, web-scale extraction systems generally consider extractions independently, ignoring the dependencies between facts or relying on simple heuristics to enforce consistency.

However, reasoning jointly about facts shows promise for improving the quality of the knowledge graph. Previous work (Jiang et al., 2012) chooses candidate facts for inclusion in a knowledge base with a joint approach using Markov Logic Networks (MLNs) (Richardson & Domingos, 2006). Jiang et al. provide a straightforward codification of ontological relations and candidate facts found in a knowledge base as rules in first-order logic and use MLNs to formulate a probabilistic model. However, due to the combinatorial explosion of Boolean assignments to random variables, inference and learning in MLNs pose intractable optimization problems. Jiang et al. limit the candidate facts they consider, restricting their dataset to a 2-hop neighborhood around each fact, and use a sampling approach to inference, estimating marginals using MC-SAT. Despite these approximations, Jiang et al. demonstrate the utility of joint reasoning in comparison to a baseline that considers each fact independently.

Our work builds on the foundation of Jiang et al. by providing a richer model for knowledge bases and vastly improving scalability. Using the noisy input from an information extraction system, we define the problem of jointly inferring the entities, relations and attributes comprising a knowledge graph as *knowledge graph identification*. We leverage dependencies in the

ICML workshop on Structured Learning: Inferring Graphs from Structured and Unstructured Inputs (SLG 2013).

knowledge graph expressed through ontological constraints, and perform entity resolution allowing us to reason about co-referent entities. We also take advantage of uncertainty found in the extracted data, using continuous variables with values derived from extractor confidence scores. Rather than limit inference to a predefined test set, we employ lazy inference to produce a broader set of candidates.

To support this representation, we use a continuous-valued Markov random field and use the probabilistic soft logic (PSL) modeling framework (Broecheler et al., 2010). Inference in our model can be formulated as a convex optimization that scales linearly in the number of variables (Bach et al., 2012), allowing us to handle millions of candidate facts.

Our work:

- Defines the knowledge graph identification problem;
- Uses soft-logic values to leverage extractor confidences;
- Formulates knowledge graph inference as convex optimization;
- Evaluates our proposed approach on extractions from NELL, a large-scale operational knowledge extraction system;
- Produces high-precision results on millions of candidate extractions on the scale of hours.

2. Background

Probabilistic soft logic (PSL) (Broecheler et al., 2010; Kimmig et al., 2012) is a recently-introduced framework which allows users to specify rich probabilistic models over continuous-valued random variables. Like other statistical relational learning languages such as MLNs, it uses first-order logic to describe features that define a Markov network. In contrast to other approaches, PSL: 1) employs continuous-valued random variables rather than binary variables; and 2) casts MPE inference as a convex optimization problem that is significantly more efficient to solve than its combinatorial counterpoint (polynomial vs. exponential).

A PSL model is composed of a set of weighted, first-order logic rules, where each rule defines a set of features of a Markov network sharing the same weight. Consider the formula

$$\mathrm{P}(A,B) \wedge \mathrm{Q}(B,C) \overset{w}{\Rightarrow} \mathrm{R}(A,B,C)$$

which is an example of a PSL rule. Here w is the weight of the rule, A, B, and C are universally-quantified variables, and P, Q and R are predicates. A *grounding* of a rule comes from substituting constants for universally-quantified variables in the rule's atoms. In this example, assigning constant values a, b, and c to the respective variables in the rule above would produce the ground atoms P(a,b), Q(b,c), R(a,b,c). Each ground atom takes a soft-truth value in the range $[0, 1]$.

PSL associates a numeric *distance to satisfaction* with each ground rule that determines the value of the corresponding feature in the Markov network. The distance to satisfaction is defined by treating the ground rule as a formula over the ground atoms in the rule. In particular, PSL uses the *Lukasiewicz t-norm* and *co-norm* to provide a relaxation of the logical connectives, AND ($\wedge$), OR($\vee$), and NOT($\neg$), as follows (where relaxations are denoted using the $\sim$ symbol over the connective):

$$p\tilde{\wedge}q = \max(0, p+q-1)$$
$$p\tilde{\vee}q = \min(1, p+q)$$
$$\tilde{\neg}p = 1-p$$

This relaxation coincides with Boolean logic when p and q are in $\{0, 1\}$, and provides a consistent interpretation of soft-truth values when p and q are in the numeric range $[0, 1]$.

A PSL program, $\mathbf{P}$, consisting of a model as defined above, along with a set of constants (or facts), produces a set of ground rules, R. If I is an interpretation (an assignment of soft-truth values to ground atoms) and r is a ground instance of a rule, then the distance to satisfaction $\phi_r(I)$ of r is simply the soft-truth value from the Lukasiewicz t-norm. We can define a probability distribution over interpretations by combining the weighted degree of satisfaction over all ground rules, R, and normalizing, as follows:

$$f(I) = \frac{1}{Z}\exp[\sum_{r\in R} w_r \phi_r(I)]$$

Here Z is a normalization constant and w_r is the weight of rule r. Thus, a PSL program (set of weighted rules and facts) defines a probability distribution from a logical formulation that expresses the relationships between random variables.

MPE inference in PSL determines the most likely soft-truth values of unknown ground atoms using the values of known ground atoms and the dependencies between atoms encoded by the rules, corresponding to inference of random variables in the underlying Markov network. PSL atoms take soft-truth values in the interval $[0, 1]$, in contrast to MLNs, where atoms take Boolean values. MPE inference in MLNs requires optimizing over the combinatorial assignment of Boolean truth values to random variables. In contrast, the relaxation to the continuous domain greatly changes the tractability of computations in PSL: finding the most probable interpretation given a set of weighted rules is equivalent to solving a convex optimization problem. Recent work from (Bach et al., 2012) introduces a consensus optimization method applicable to PSL models; their results suggest consensus optimization scales linearly in the number of random variables in the model.

3. Knowledge Graph Identification

Our approach to constructing a consistent knowledge base uses PSL to represent the candidate facts from an information extraction system as a *knowledge graph* where entities are nodes, categories are labels associated with each node, and relations are directed edges between the nodes. Information extraction systems can extract such candidate facts, and these extractions can be used to construct a graph. Unfortunately, the output from an information extraction system is often incorrect; the graph constructed from it has spurious and missing nodes and edges, and missing or inaccurate node labels. Our approach, *knowledge graph identification* combines the tasks of entity resolution, collective classification and link prediction mediated by ontological constraints. We motivate the necessity of our approach with examples of challenges taken from a real-world information extraction system, the Never-Ending Language Learner (NELL) (Carlson et al., 2010).

One common problem is entity extraction. Many textual references that initially look different may correspond to the same real-world entity. For example, NELL's knowledge base contains candidate facts which involve the entities "kyrghyzstan", "kyrgzstan", "kyrgystan", "kyrgyz republic", "kyrgyzstan", and "kyrgistan" which are all variants or misspellings of the same country, Kyrgyzstan. In the extracted knowledge graph, these correspond to different nodes, which is incorrect. Our approach uses *entity resolution* to determine co-referent entities in the knowledge graph, producing a consistent set of labels and relations for each resolved node.

Another challenge in knowledge graph construction is inferring labels consistently. For example, NELL's extractions assign Kyrgyzstan the attributes "country" as well as "bird". Ontological information can be used to infer that an entity is very unlikely to be both a country and a bird at the same time. Using the labels of other, related entities in the knowledge graph can allow us to determine the correct label of an entity. Our approach uses *collective classification* to label nodes in manner which takes into account ontological constraints and neighboring labels.

A third problem commonly encountered in knowledge graphs is finding the set of relations an entity participates in. NELL also has many facts relating the location of Kyrgyzstan to other entities. These candidate relations include statements that Kyrgyzstan is located in Kazakhstan, Kyrgyzstan is located in Russia, Kyrgyzstan is located in the former Soviet Union, Kyrgyzstan is located in Asia, and that Kyrgyzstan is located in the US. Some of these possible relations are true, while others are clearly false and contradictory. Our approach uses *link prediction* to predict edges in a manner which takes into account ontological constraints and the rest of the inferred structure.

Refining a knowledge graph becomes even more challenging as we consider the interaction between the predictions and take into account the confidences we have in the extractions. For example, as mentioned earlier, NELL's ontology includes the constraint that the attributes "bird" and "country" are mutually exclusive. Reasoning collectively allows us to resolve which of these two labels is more likely to apply to Krygyzstan. For example, NELL is highly confident that the Kyrgyz Republic has a capital city, Bishkek. The NELL ontology specifies that the domain of the relation "hasCapital" has label "country". Entity resolution allows us to infer that "Kyrgyz Republic" refers to the same entity as "Kyrgyzstan". Deciding whether Kyrgyzstan is a bird or a country now involves a prediction where we include the confidence values of the corresponding "bird" and "country" facts from co-referent entities, as well as collective features from ontological constraints of these co-referent entities, such as the confidence values of the "hasCapital" relations.

We refer to this process of inferring a knowledge graph from a noisy extraction graph as knowledge graph identification. Knowledge graph identification builds on ideas from *graph identification* (Namata et al., 2011); like graph identification, three key components to the problem are entity resolution, node labeling and link prediction. Unlike earlier work on graph identification, we use a very different probabilistic framework, PSL, allowing us to incorporate extractor confidence values and also support a rich collection of ontological constraints.

4. Knowledge Graph Identification Using PSL

Knowledge graphs contain three types of facts: facts about entities, facts about entity labels and facts about relations. We represent entities with the logical predicate Ent(E). We represent labels with the logical predicate Lbl(E,L) where entity E has label L. Relations are represented with the logical predicate Rel(E_1,E_2,R) where the relation R holds between the entities E_1 and E_2, eg. `R(`E_1`,`E_2`)`.

In knowledge graph identification, our goal is to identify a true set of predicates from a set of noisy extractions. Our method for knowledge graph identification incorporates three components: capturing uncertain extractions, performing entity resolution, and enforcing ontological constraints. We show how we create a PSL program that encompasses these three components, and then relate this PSL program to a distribution over possible knowledge graphs.

4.1. Representing Uncertain Extractions

We relate the noisy extractions from an information extraction system to the above logical predicates by introducing *candidate* predicates, using a formulation similar to (Jiang et al., 2012).

For each candidate entity, we introduce a corresponding predicate, CandEnt(E). Labels or relations generated by the information extraction system correspond to predicates, CandLbl(E,L) or CandRel(E_1,E_2,R) in our system. Uncertainty in these extractions is captured by assigning these predicates a soft-truth value equal to the confidence value from the extractor. For example, the extraction system might generate a relation, `teamPlaysSport`(`Yankees`,`baseball`) with a confidence of .9, which we would represent as CandRel(`Yankees`,`baseball`,`teamPlaysSport`).

Information extraction systems commonly use many different extraction techniques to generate candidates. For example, NELL produces separate extractions from lexical, structural, and morphological patterns, among others. We represent metadata about the technique used to extract a candidate by using separate predicates for each technique T, of the form CandRel$_T$ and CandLbl$_T$. These predicates are related to the true values of attributes and relations we seek to infer using weighted rules.

$$\textsc{CandRel}_T(E_1, E_2, R) \overset{w_{CR-T}}{\Rightarrow} \textsc{Rel}(E_1, E_2, R)$$
$$\textsc{CandLbl}_T(E, L) \overset{w_{CL-T}}{\Rightarrow} \textsc{Lbl}(E, L)$$

Together, we denote the set of candidates, generated from grounding the rules above using the output from the extraction system, as the set $\mathcal{C}$.

In addition to the candidate facts in an information extraction system, we sometimes have access to background knowledge or previously learned facts. Background knowledge that is certain can be represented using the Lbl and Rel predicates. Often the background knowledge included in information extraction settings is generated from the same pool of noisy extractions as the candidates, and is considered uncertain. For example, NELL uses a heuristic formula to "promote" candidates in each iteration of the system, however these promotions are often noisy so the system assigns each promotion a confidence value. Since these promotions are drawn from the best candidates in previous iterations, they can be a useful addition to our model. We incorporate this uncertain background knowledge as hints, denoted $\mathcal{H}$, providing a source of weak supervision through the use of additional predicates and rules as follows:

$$\textsc{HintRel}(E_1, E_2, R) \overset{w_{HR}}{\Rightarrow} \textsc{Rel}(E_1, E_2, R)$$
$$\textsc{HintLbl}(E, L) \overset{w_{HL}}{\Rightarrow} \textsc{Lbl}(E, L)$$

The weights w_{HR} and w_{HL} allow the system to specify how reliable this background knowledge is as a source of weak supervision, while treating these rules as constraints is the equivalent of treating the background knowledge as certain knowledge.

4.2. Entity Resolution

While the previous PSL rules provide the building blocks of predicting links and labels using uncertain information, knowledge graph identification employs entity resolution to pool information across co-referent entities. A key component of this process is identifying possibly co-referent entities and determining the similarity of these entities. We use the SameEnt predicate to capture the similarity of two entities. While any similarity metric can be used, we compute the similarity of entities using a process of mapping each entity to a set of Wikipedia articles and then computing the Jaccard index of possibly co-referent entities, which we discuss in more detail in Section 5.

To perform entity resolution using the SameEnt predicate we introduce three rules, whose groundings we refer to as $\mathcal{R}$, to our PSL program:

$$\textsc{SameEnt}(E_1, E_2)\tilde{\wedge}\textsc{Lbl}(E_1, L) \Rightarrow \textsc{Lbl}(E_2, L)$$
$$\textsc{SameEnt}(E_1, E_2)\tilde{\wedge}\textsc{Rel}(E_1, E, R) \Rightarrow \textsc{Rel}(E_2, E, R)$$
$$\textsc{SameEnt}(E_1, E_2)\tilde{\wedge}\textsc{Rel}(E, E_1, R) \Rightarrow \textsc{Rel}(E, E_2, R)$$

These rules define an equivalence class of entities, such that all entities related by the SameEnt predicate must have the same labels and relations. The soft-truth value of the SameEnt, derived from our similarity function, mediates the strength of these rules. When two entities are very similar, they will have a high truth value for SameEnt, so any label assigned to the first entity will also be assigned to the second entity. On the other hand, if the similarity score for two entities is low, the truth values of their respective labels and relations will not be strongly constrained. While we introduce these rules as constraints to the PSL model, they could be used as weighted rules, allowing us to specify the reliability of the similarity function.

4.3. Enforcing Ontological Constraints

In our PSL program we also leverage rules corresponding to an ontology, the groundings of which are denoted as $\mathcal{O}$. Our ontological constraints are based on the logical formulation proposed in (Jiang et al., 2012). Each type of ontological relation is represented as a predicate, and these predicates represent ontological knowledge of the relationships between labels and relations. For example, the constraints Dom(`teamPlaysSport`, `sportsteam`) and Rng(`teamPlaysSport`, `sport`) specify that the relation `teamPlaysSport` is a mapping from entities with label `sportsteam` to entities with label `sport`. The constraint Mut(`sport`, `sportsteam`) specifies that the labels `sportsteam` and `sport` are mutually exclusive, so that an entity cannot have both the labels `sport` and `sportsteam`. We similarly use constraints for subsumption of labels (Sub) and inversely-related functions (Inv). To use this ontological knowledge, we introduce rules relating each ontological relation to the predicates representing our knowledge graph. We specify seven types of ontological constraints in our experiments:

$$
\begin{array}{llll}
\textsc{Dom}(R,L) & \tilde{\wedge}\ \textsc{Rel}(E_1,E_2,R) & \Rightarrow & \textsc{Lbl}(E_1,L) \\
\textsc{Rng}(R,L) & \tilde{\wedge}\ \textsc{Rel}(E_1,E_2,R) & \Rightarrow & \textsc{Lbl}(E_2,L) \\
\textsc{Inv}(R,S) & \tilde{\wedge}\ \textsc{Rel}(E_1,E_2,R) & \Rightarrow & \textsc{Rel}(E_2,E_1,S) \\
\textsc{Sub}(L,P) & \tilde{\wedge}\ \textsc{Lbl}(E,L) & \Rightarrow & \textsc{Lbl}(E,P) \\
\textsc{Mut}(L_1,L_2) & \tilde{\wedge}\ \textsc{Lbl}(E,L_1) & \Rightarrow & \neg\textsc{Lbl}(E,L_2) \\
\textsc{RMut}(R,S) & \tilde{\wedge}\ \textsc{Rel}(E_1,E_2,R) & \Rightarrow & \neg\textsc{Rel}(E_1,E_2,S)
\end{array}
$$

These ontological rules are specified as constraints to our PSL model. When optimizing the model, PSL will only consider truth-value assignments or interpretations that satisfy all of these ontological constraints.

4.4. Probability Distribution Over Uncertain Knowledge Graphs

The logical formulation introduced in this section, together with ontological information and the outputs of an information extraction system define a PSL program $\mathbf{P}$. The corresponding set of ground rules, R, consists of the union of groundings from uncertain candidates, $\mathcal{C}$, and hints, $\mathcal{H}$, co-referent entities, $\mathcal{R}$, and ontological constraints, $\mathcal{O}$. The distribution over interpretations, I, generated by PSL corresponds to a probability distribution over knowledge graphs, G:

$$P_{\mathbf{P}}(G) = f(I) = \frac{1}{Z}\exp[\sum_{r\in R} w_r\phi_r(I)]$$

The results of inference provide us with the most likely interpretation, or soft-truth assignments to entities, labels and relations that comprise the knowledge graph. By choosing a threshold on the soft-truth values in the interpretation, we can select a high-precision set of facts to construct our knowledge graphs.

5. Experimental Evaluation

We evaluate our method on data from the Never-Ending Language Learning (NELL) project (Carlson et al., 2010). Our goal is to demonstrate that Knowledge Graph Identification can remove noise from uncertain extractions, producing a high-precision set of facts for inclusion in a knowledge graph. A principal concern in this domain is scalability; we show that using PSL for MPE inference is a practical solution for knowledge graph identification at a massive scale.

NELL iteratively generates a knowledge base: in each iteration NELL uses facts learned from the previous iteration and a corpus of web pages to generate a new set of candidate facts. NELL selectively promotes those candidates that have a high confidence from the extractors and obey ontological constraints with the existing knowledge base to build a high-precision knowledge base. We present experimental results on the 194^{th} iteration of NELL, using the candidate facts, promoted facts and ontological constraints that NELL used during that iteration. We summarize the important statistics of this dataset in Table 1.

In addition to data from NELL, we use data from the YAGO database (Suchanek et al., 2007) as part of our entity resolution approach. The YAGO database contains entities which correspond to Wikipedia articles, variant spellings and abbreviations of these entities, and associated WordNet categories. To correct against the multitude of variant spellings found in NELL's data, we use a mapping technique from NELL's entities to Wikipedia articles. We then define a similarity function on the article URLs, using the similarity as the soft-truth value of the SameEnt predicate.

When mapping NELL entities to YAGO records, We perform selective stemming on the NELL entities, em-

	Iter194
Date Generated	1/2011
Cand. Label	1M
Cand. Rel	530K
Promotions	300K
Dom, Rng, Inv	660
Sub	520
Mut	29K
RMut	58K

Table 1. Summary of dataset statistics

Percentile	1%	2%	5%	10%	25%
Precision	.96	.95	.89	.90	.74

Table 2. Precision of knowledge graph identification at top soft truth value percentiles

ploy blocking on candidate labels, and use a case-insensitive string match. Each NELL entity can be mapped to a set of YAGO entities, and we can generate a set of Wikipedia URLs that map to the YAGO entities. To generate the similarity of two NELL entities, we compute a set-similarity measure on the Wikipedia URLs associated with the entities. For our similarity score, we use the Jaccard index, the ratio of the size of the set intersection and the size of the set union.

The ultimate goal of knowledge graph identification is to generate a high-precision set of facts for inclusion in a knowledge base. We evaluated the precision of our method by sampling facts from the top 1%, 2%, 5%, 10% and 25% of inferred facts ordered by soft truth value. We sampled 120 facts at each cutoff, split evenly between labels and relations, and had a human judge provide appropriate labs. Table 2 shows the precision of our method at each cutoff. The precision decays gracefully from .96 at the 1% cutoff to .90 at the 10% cutoff, but has a substantial drop-off at the 25% cutoff.

Performing the convex optimization for MPE inference in knowledge graph identification is efficient. The optimization problem is defined over 22.6M terms, consisting of potential functions corresponding to candidate facts and dependencies among facts generated by ontological constraints. Performing this optimization on a six-core Intel Xeon X5650 CPU at 2.66 GHz with 32GB of RAM took 44300 seconds.

6. Conclusion

We have described how to formulate the problem of *knowledge graph identification*, jointly inferring a knowledge graph from the noisy output of an information extraction system through a combined process of determining co-referent entities, predicting relational links, collectively classifying entity labels, and enforcing ontological constraints. Using PSL, we illustrate the scalability benefits of our approach on a large-scale dataset from NELL, while producing high-precision results.

Acknowledgments This work was partially supported by NSF CAREER grant 0746930 and NSF grant IIS1218488.

References

Artiles, J and Mayfield, J (eds.). *Workshop on Knowledge Base Population*, 2012.

Bach, S. H, Broecheler, M, Getoor, L, and O'Leary, D. P. Scaling MPE Inference for Constrained Continuous Markov Random Fields with Consensus Optimization. In *NIPS*, 2012.

Broecheler, M, Mihalkova, L, and Getoor, L. Probabilistic Similarity Logic. In *UAI*, 2010.

Carlson, A, Betteridge, J, Kisiel, B, Settles, B, Hruschka, E. R, and Mitchell, T. M. Toward an Architecture for Never-Ending Language Learning. In *AAAI*, 2010.

Etzioni, O, Banko, M, Soderland, S, and Weld, D. S. Open Information Extraction from the Web. *Communications of the ACM*, 51(12), 2008.

Ji, H, Grishman, R, and Dang, H. Overview of the Knowledge Base Population Track. In *Text Analysis Conference*, 2011.

Jiang, S, Lowd, D, and Dou, D. Learning to Refine an Automatically Extracted Knowledge Base Using Markov Logic. In *ICDM*, 2012.

Kimmig, A, Bach, S. H, Broecheler, M, Huang, B, and Getoor, L. A Short Introduction to Probabilistic Soft Logic. In *NIPS Workshop on Probabilistic Programming*, 2012.

Namata, G. M, Kok, S, and Getoor, L. Collective Graph Identification. In *KDD*, 2011.

Pasca, M, Lin, D, Bigham, J, Lifchits, A, and Jain, A. Organizing and Searching the World Wide Web of Facts-Step One: the One-million Fact Extraction Challenge. In *AAAI*, 2006.

Richardson, M and Domingos, P. Markov Logic Networks. *Machine Learning*, 62(1-2), 2006.

Singhal, A. Introducing the Knowledge Graph: Things, Not Strings, 2012. Official Blog (of Google).

Suchanek, F. M, Kasneci, G, and Weikum, G. Yago: A Core of Semantic Knowledge. In *WWW*, 2007.