Knowledge Graph Identification

Jay Pujara[1], Hui Miao[1], Lise Getoor[1], and William Cohen[2]

[1] Dept of Computer Science, University of Maryland, College Park, MD 20742
{jay,hui,getoor}@cs.umd.edu
[2] Machine Learning Dept, Carnegie Mellon University, Pittsburgh, PA 15213
wcohen@cs.cmu.edu

Abstract. Large-scale information processing systems are able to extract massive collections of interrelated facts, but unfortunately transforming these candidate facts into useful knowledge is a formidable challenge. In this paper, we show how uncertain extractions about entities and their relations can be transformed into a *knowledge graph*. The extractions form an *extraction graph* and we refer to the task of removing noise, inferring missing information, and determining which candidate facts should be included into a knowledge graph as *knowledge graph identification*. In order to perform this task, we must reason jointly about candidate facts and their associated extraction confidences, identify co-referent entities, and incorporate ontological constraints. Our proposed approach uses probabilistic soft logic (PSL), a recently introduced probabilistic modeling framework which easily scales to millions of facts. We demonstrate the power of our method on a synthetic Linked Data corpus derived from the MusicBrainz music community and a real-world set of extractions from the NELL project containing over 1M extractions and 70K ontological relations. We show that compared to existing methods, our approach is able to achieve improved AUC and F1 with significantly lower running time.

1 Introduction

The web is a vast repository of knowledge, but automatically extracting that knowledge at scale has proven to be a formidable challenge. Recent evaluation efforts have focused on automatic knowledge base population [1, 2], and many well-known broad domain and open information extraction systems exist, including the Never-Ending Language Learning (NELL) project [3], OpenIE [4], and efforts at Google [5], 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 [6].

A key challenge in producing the knowledge graph is incorporating noisy information from different sources in a consistent manner. Information extraction systems operate over many source documents, such as web pages, and use a collection of strategies to generate candidate facts from the documents, spanning syntactic, lexical and structural features of text. Ultimately, these extraction systems produce candidate facts that include a set of entities, attributes of these entities, and the relations between these entities which we refer to as the extraction graph. However errors in the extraction process introduce inconsistencies in the extraction graph, which may contain duplicate entities and violate key ontological constraints such as subsumption, mutual exclusion, inverse, domain and range constraints. Such noise obscures the true knowledge graph, which captures a consistent set of entities, attributes and relations.

Our work infers the knowledge graph from the extraction graph generated by an information extraction system. We demonstrate that the errors encountered by information extraction systems require jointly reasoning over candidate facts to construct a consistent knowledge graph. Our approach performs entity resolution, collective classification and link prediction while also enforcing global constraints on the knowledge graph, a process which we refer to as *knowledge graph identification*.

In order to implement knowledge graph identification, we use probabilistic soft logic (PSL) [7], a recently introduced framework for reasoning probabilistically over continuously-valued random variables. PSL provides many advantages: models are easily defined using declarative rules with first-order logic syntax, continuously-valued variables provide a convenient representation of uncertainty, weighted rules and weight learning capture the importance of model rules, and advanced features such as set-based aggregates and hard constraints are supported. In addition, inference in PSL is a convex optimization that is highly scalable allowing us to handle millions of facts in minutes.

We develop a PSL model for knowledge graph identification that both captures probabilistic dependencies between facts and enforces global constraints between entities and relations. Through this model, we define a probability distribution over interpretations - or truth value assignments to facts - each of which corresponds to a possible knowledge graph. By performing inference using the extraction graph and an ontology, we are able to find the most probable knowledge graph. We establish the benefits of our approach on two large datasets: a synthetic dataset derived from the MusicBrainz community and ontological relationships defined in the Music Ontology as well as noisy extractions from NELL, a large-scale operational knowledge extraction system.

Our contributions in this work are 1) formulating the knowledge graph identification problem that supports reasoning about multiple, uncertain extractor sources in the presence of ontological constraints; 2) solving knowledge graph identification efficiently with convex optimization using PSL; and 3) demonstrating the power of knowledge graph identification by presenting results on benchmark datasets that are superior to state-of-the-art methods and generating massive knowledge graphs on the scale of minutes that are infeasible to compute in competing systems.

2 Related Work

Early work on the problem of jointly identifying a best latent KB from a collection of noisy facts was considered by Cohen et al. [8], however they considered only a small subset of KB errors. More recently, Jiang et al. [9] perform knowledge base refinement at a broader scope by using an ontology to relate candidate extractions and exploring many different modeling choices with Markov Logic

Networks (MLNs) [10]. Jiang et al. provide a crisp codification of ontological constraints and candidate facts found in a knowledge base as rules in first-order logic, contributing an attractive abstraction for knowledge bases which we adopt in our modeling. However, the choice of MLNs as a modeling framework comes with certain limitations. In MLNs, all logical predicates must take Boolean truth values, making it difficult to incorporate the confidence values. Moreover, the combinatorial explosion of Boolean assignments to random variables makes inference and learning in MLNs intractable optimization problems. Jiang et al. surmount these obstacles with a number of approximations and demonstrate the utility of joint reasoning in comparison to a baseline that considers each fact independently. By using PSL we can avoid these representational and scalability limitations, and we build on and improve the model of Jiang et al. by including multiple extractors in our model and reasoning about co-referent entities.

Other research has used relevant techniques for problems related to knowledge graph identification. Namata et al. [11] introduced the problem of graph identification to uncover the true graph from noisy observations through entity resolution, collective classification, and link prediction. However, Namata's approach considered these tasks iteratively and could not easily support logical constraints such as those found in an ontology. Memory et al. [12] also use PSL to resolve confounding evidence. Their model performs graph summarization across multiple ontologies and uses inference only for inferring missing links. Work by Yao et al. [13] employs joint reasoning at the extractor level by using conditional random fields to learn selectional preferences for relations.

3 Motivation: Knowledge Graph Identification

In this work, we 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 an extraction graph. Unfortunately, the extraction graph is often incorrect, with errors such as spurious and missing nodes and edges, and missing or inaccurate node labels. Our approach, knowledge graph identification (KGI) combines the tasks of entity resolution, collective classification and link prediction mediated by rules based on ontological information. 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) [3].

Entity extraction is a common problem: many textual references that initially look different may refer to the same real-world entity. For example, NELL's knowledge base contains candidate facts involving the entities "kyrghyzstan", "kyrgzstan", "kyrgystan", "kyrgyz republic", "kyrgyzstan", and "kyrgistan" which are all variants or misspellings of the country Kyrgyzstan. In the extracted knowledge graph, these incorrectly correspond to different nodes. 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.

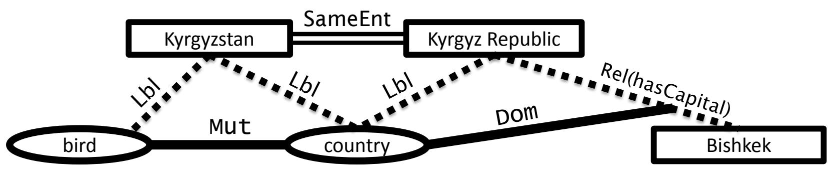

Fig. 1. An illustration of the example showing how knowledge graph identification can resolve conflicting information in an extraction graph. Entities are shown in rectangles, dotted lines represent uncertain information, solid lines show ontological constraints and double lines represent co-referent entities found with entity resolution.

Another challenge in knowledge graph construction is inferring labels consistently. For example, NELL's extractions assign Kyrgyzstan the labels "country" as well as "bird." Ontological information suggests that an entity is very unlikely to be both a country and a bird at the same time. Using the labels of 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 information and neighboring labels.

A third problem commonly encountered in knowledge graphs is determining the relationships between entities. 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 information and the rest of the inferred structure.

Refining an extraction graph becomes even more challenging as we consider the interaction between the predictions and take into account the confidences we have in the extractions. Figure 1 illustrates such a complex example. As mentioned earlier, NELL's ontology includes the constraint that the labels "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 relationships 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. Unlike earlier work on graph identification and knowledge base refinement, we use a very different probabilistic framework, PSL, allowing us to jointly infer a knowledge graph while incorporating extractor confidence values and supporting a rich collection of ontological constraints.

4 Background: Probabilistic Soft Logic

Probabilistic soft logic (PSL) [7, 14] 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 Markov Logic Networks (MLNs), it uses first-order logic to describe features that define a Markov network. In contrast to other approaches, PSL employs continuous-valued random variables rather than binary variables and casts most probable explanation (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

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

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, $\mathbf{\Pi}$, consisting of a model as defined above, along with a set of facts, F, 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 $1 - T_r(I)$, where $T_r(I)$ is 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:

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

Here Z is a normalization constant, w_r is the weight of rule r, and p in $\{1, 2\}$ allows a linear or quadratic combination of rules. 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 combinatorial assignments of Boolean truth values. 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 [15] introduces a consensus optimization method applicable to PSL models; their results suggest consensus optimization scales linearly with the number of ground rules in the model.

5 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) and 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 atoms 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.

5.1 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 [9]. 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, `hasCapital`(`kyrgyzstan`, `Bishkek`) with a confidence of .9, which we would represent as CANDREL(-`kyrgyzstan`,`Bishkek`, `hasCapital`) and assign it a truth value of .9.

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 $\textsc{CandRel}_T$ and $\textsc{CandLbl}_T$. 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 $\mathcal{C}$.

5.2 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, which we discuss in detail in Section 6. We use the $\textsc{SameEnt}$ predicate to capture the similarity of two entities, for example $\textsc{SameEnt}$(`kyrgyzstan`, `kyrgz republic`).

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

$$\textsc{SameEnt}(E_1, E_2)\tilde{\wedge}\textsc{Lbl}(E_1, L) \overset{w_{EL}}{\Rightarrow} \textsc{Lbl}(E_2, L)$$

$$\textsc{SameEnt}(E_1, E_2)\tilde{\wedge}\textsc{Rel}(E_1, E, R) \overset{w_{ER}}{\Rightarrow} \textsc{Rel}(E_2, E, R)$$

$$\textsc{SameEnt}(E_1, E_2)\tilde{\wedge}\textsc{Rel}(E, E_1, R) \overset{w_{ER}}{\Rightarrow} \textsc{Rel}(E, E_2, R)$$

These rules define an equivalence class of entities, such that all entities related by the $\textsc{SameEnt}$ predicate must have the same labels and relations. The soft-truth value of the $\textsc{SameEnt}$, 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 $\textsc{SameEnt}$, 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. We introduce these rules as weighted rules in the PSL model, where the weights can capture the reliability of the similarity function.

5.3 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 rules are based on the logical formulation proposed in [9]. 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 ontological predicates $\textsc{Dom}$(`hasCapital`, `country`) and $\textsc{Rng}$(`hasCapital`, `city`) specify that the relation `hasCapital` is a mapping from entities with label `country` to entities with label `city`. The predicate $\textsc{Mut}$(`country`, `city`) specifies that the labels `country` and `city` are mutually exclusive, so that an entity cannot have both the labels `country` and `city`. We similarly use predicates for subsumption of labels (SUB) and relations(RSUB), and inversely-related functions (INV). To use this ontological knowledge, we introduce rules relating each ontological predicate to the predicates representing our knowledge graph. We specify seven types of ontological constraints in our experiments using weighted rules:

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

5.4 Probability Distribution Over Uncertain Knowledge Graphs

Combining the logical rules introduced in this section with atoms, such as candidates from the information extraction system (e.g. CANDREL(`kyrgyzstan`, `Bishkek`, `hasCapital`)), co-reference information from an entity resolution system (e.g. SAMEENT(`kyrgyzstan`, `kyrgz republic`)) and ontological information (e.g. DOM(`hasCapital`, `country`)) we can define a PSL program, $\mathbf{\Pi}$. The inputs to this program instantiate a set of ground rules, R, that consists of the union of groundings from uncertain candidates, $\mathcal{C}$, co-referent entities, $\mathcal{S}$, and ontological relationships, $\mathcal{O}$. The distribution over interpretations, I, generated by PSL corresponds to a probability distribution over knowledge graphs, G:

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

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 a knowledge graph.

6 Experimental Evaluation

6.1 Datasets and Experimental Setup

We evaluate our method on two different datasets: a synthetic knowledge base derived from the LinkedBrainz project [16], which maps data from the MusicBrainz community using ontological information from the MusicOntology [17] as well as web-extraction data from the Never-Ending Language Learning (NELL) project [3]. Our goal is to assess the utility of knowledge graph identification, formulated as a PSL model, at inferring a knowledge graph from noisy data. Additionally,

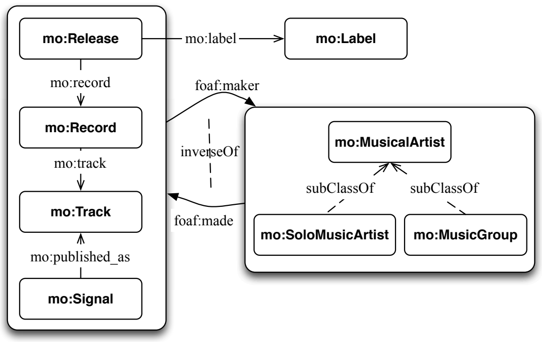

Fig. 2. Subset of Music Ontology mapped using LinkedBrainz for MusicBrainz data in our synthetic dataset

we contrast two very different evaluation settings. In the first, as used in previous work [9] inference is limited to a subset of the knowledge graph generated from the test or query set. In the second evaluation setting, inference produces a complete knowledge graph, which is not restricted by the test set but employs a soft-truth threshold for atoms. We provide documentation, code and datasets to replicate our results on GitHub[3].

MusicBrainz MusicBrainz is a community-driven, open-source, structured database for music metadata, including information about artists, albums, and tracks, The Music Ontology is built on top of many well known ontologies, such as FRBR [18] and FOAF [19], and has been used widely, for instance in BBC Music Linked Data sites [20]. However, the relational data available from MusicBrainz are expressed in a proprietary schema that does not map directly to the Music Ontology. To bridge this gap, the LinkedBrainz project publishes an RDF mapping between the freely available MusicBrainz data and the Music Ontology using D2RQ [21]. A summary of the labels and relations we use in our data is show in Figure 2. We use an intuitive mapping of ontological relationships to the PSL predicates, using ontological information from FRBR and FOAF classes used by the Music Ontology. Specifically we convert `rdfs:domain` to Dom, `rdfs:range` to Rng, `rdfs:subClassOf` to Sub, `rdfs:subPropertyOf` to RSub, `owl:inverseOf` to Inv, and `owl:disjointWith` to Mut.

Our synthetic knowledge graph uses a sample of data from the LinkedBrainz mapping of the MusicBrainz project[4] and adds noise to generate a realistic data set. To generate a subset of the LinkedBrainz data, we use snowball sampling from a set of tracks in the MusicBrainz dataset to produce a set of recordings, releases, artists and labels. Next, we introduce noise into this graph by randomly removing known facts and adding inconsistent facts as well as generating random confidence values for these facts. This noise can be interpreted as errors intro-

[3] `https://github.com/linqs/KnowledgeGraphIdentification`
[4] `http://linkedbrainz.c4dmpresents.org/content/rdf-dump`

duced by a MusicBrainz user misspelling artist names, accidentally switching input fields, or omitting information when contributing to the knowledge base.

We model these errors by distorting a percentage of the true input data. For labels, we omit known labels and introduce spurious labels for 25% of the facts in the input data. When dealing with relations, we focus on the `foaf:maker` and `foaf:made` relations between artists and creative works. We randomly remove one of these pair of relations 25% of the time. Finally, 25% of the time we remove the relationship between a work and its artist, and insert a new relationship between the work and a generated artist, adding a SameEnt for these two artists. The confidence values for facts found in the input are generated from a Normal$(.7, .2)$ distribution while inconsistent facts have lower confidence values generated from a Normal$(.3, .2)$ distribution. The high variance in these distributions ensures a significant overlap. For the SameEnt the similarity values are generated from a Normal$(.9, .1)$ distribution. In all cases, the distribution is thresholded to the $[0, 1]$ range.

We summarize important data statistics in Table 1. In our experiments, we represent the noisy relations and labels of the knowledge graph as candidate facts in PSL with the predicates CandLbl and CandRel. During evaluation, we use the PSL program for knowledge graph identification to infer the most probable knowledge graph. In this setting, we use quadratic combinations of static weights for all rules, where $w_{CL} = w_{CR} = 1$, $w_{EL} = w_{ER} = 25$ and $w_O = 100$. We evaluate our results by comparing to the true knowledge graph used to generate the data, and include false labels corresponding to spurious data we introduce.

NELL The goal of NELL is to iteratively generate 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 165th iteration of NELL, using the candidate facts, promoted facts and ontological relationships that NELL used during that iteration. We summarize the important statistics of this dataset in Table 1. Due to the diversity of the web, the data from NELL is larger, includes more types of relations and categories, and has more ontological relationships than our synthetic data.

NELL uses diverse extraction sources, and in our experiments we use distinct predicates CandLbl$_T$ and CandRel$_T$ for the sources CBL, CMC, CPL, Morph, and SEAL while the remaining sources, which do not contribute a significant number of facts, are represented with CandLbl and CandRel predicates. In addition to candidate facts, NELL uses a heuristic formula to “promote” candidates in each iteration of the system into a knowledge base, however these promotions are often noisy so the system assigns each promotion a confidence value. We represent these promoted candidates from previous iterations as an additional source with corresponding candidate predicates.

In addition to data from NELL, we use data from the YAGO database [22] as part of our entity resolution approach. Our model uses a SameEnt predicate to capture the similarity of two entities. To correct against the multitude of variant spellings found in the 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.

The YAGO database contains entities which correspond to Wikipedia articles, variant spellings and abbreviations of these entities, and associated WordNet categories. Our approach to entity resolution matches entity names in NELL with YAGO entities. We perform selective stemming on the NELL entities, employ blocking on candidate labels, and use a case-insensitive string match to find corresponding YAGO entities. Once we find a matching set of YAGO entities, we can generate a set of Wikipedia URLs that map to the corresponding NELL entities. We can judge the similarity of two entities by computing 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.

In our experiments using NELL, we consider two scenarios. The first is similar to experimental setup in [9] where rule weights are learned using training data and predictions are made on a limited neighborhood of the test set. The neighborhood used in this previous work attempts to improve scalability by generating a grounding of the test set and only including atoms that are not trivially satisfied in this grounding. In practice, this produces a neighborhood that is distorted by omitting atoms that may contradict those in the test set. For example, if ontological relationships such as Sub(`country`,`location`) and Mut(`country`, `city`) are present, the test set atom Lbl(`kyrgyzstan`,`country`) would not introduce Lbl(`kyrgyzstan`,`city`) or Lbl(`country`,`location`) into the neighborhood, even if contradictory data were present in the input candidates. By removing the ability to reason about contradictory information, we believe this evaluation setting diminishes the true difficulty of the problem. We validate our approach on this setting, but also present results from a more realistic setting. In the second scenario we perform inference independently of the test set, lazily generating truth values for atoms supported by the input data, using a soft-truth value threshold of .01. This second setting allows us to infer a complete knowledge graph similar to the MusicBrainz setting.

6.2 Knowledge Graph Identification Results for MusicBrainz

Our experiments on MusicBrainz data attempt to recover the complete knowledge graph despite the addition of noise which introduces uncertainty for facts, removes true information and adds spurious labels and relations. We evaluate a number of variants on their ability to recover this knowledge graph. We measure performance using a number of metrics: the area under the precision-recall curve (AUC), as well as the precision, recall and F1 score at a soft-truth threshold of .5, as well as the maximum F1 score on the dataset. Due to the high variance of confidence values and large number of true facts in the ground truth, the maxi-

Table 1. Summary of dataset statistics for NELL and MusicBrainz, including (a) the number of candidate facts in input data, the distinct relations and labels present, and (b) the number of ontological relationships defined between these relations and labels

(a)

	NELL	MusicBrainz
Cand. Label	1.2M	320K
Cand. Rel	100K	490K
Promotions	440K	0
Unique Labels	235	19
Unique Rels	221	8

(b)

	NELL	MusicBrainz
Dom	418	8
Rng	418	8
Inv	418	2
Mut	17.4K	8
RMut	48.5K	0
Sub	288	21
RSub	461	2

Table 2. A comparison of knowledge graph identification methods on MusicOntology data shows knowledge graph identification effectively combines the strengths of graph identification and reasoning with ontological information and produces superior results.

Method	AUC	Prec	Recall	F1	Max F1
Baseline	0.672	0.946	0.477	0.634	0.788
PSL-EROnly	0.797	0.953	0.558	0.703	0.831
PSL-OntOnly	0.753	0.964	0.605	0.743	0.832
PSL-KGI-Complete	**0.901**	**0.970**	**0.714**	**0.823**	**0.919**

mum F1 value occurs at a soft-truth threshold of 0, where recall is maximized, in all variants. These results are summarized in Table 2.

The first variant we consider uses only the input data, setting the soft-truth value equal to the generated confidence value as an indicator of the underlying noise in the data. The baseline results use only the candidate rules we introduced in subsection 5.1. We improve upon this data by adding either the entity resolution rules introduced in subsection 5.2, which we report as PSL-EROnly, or with weighted rules capturing ontological constraints introduced in subsection 5.3. Finally, we combine all the elements of knowledge graph identification introduced in section 5 and report these results as PSL-KGI-Complete. The results on the baseline demonstrate the magnitude of noise in the input data; less than half the facts in the knowledge graph can be correctly inferred. Reasoning jointly about co-referent entities, as in graph identification, improves results. Using ontological constraints, as previous work in improving extraction in this domain has, also improves results as well. Comparing these two improvements, adding entity resolution has a higher AUC, while ontological constraints show a greater improvement in F1 score. However, when these two approaches are combined, as they are in knowledge graph identification, results improve dramatically. Knowledge graph identification increases AUC, precision, recall and F1 substantially over the the other variants, improving AUC and F1 over 10% compared to the more competitive baseline methods. Overall, we are able to infer 71.4% of true relations while maintaining a precision of .97. Moreover, a high AUC of .901 suggests that knowledge graph identification balances precision and recall for a wide range of parameter values.

6.3 Knowledge Graph Identification Results for NELL

Comparison to Previous Work While results on data with synthetic noise confirm our hypothesis, we are particularly interested in the results on a large, noisy real-world dataset. We compare our method to data from iteration 165 of NELL using previously reported results on a manually-labeled evaluation set [9]. A summary of these results is shown in Table 3. The first method we compare to is a baseline similar to the one used in the MusicBrainz results where candidates are given a soft-truth value equal to the extractor confidence (averaged across extractors when appropriate). Results are reported at a soft-truth threshold of .45 which maximizes F1.

We also compare the default strategy used by the NELL project to choose candidate facts to include in the knowledge base. Their method uses the ontology to check the consistency of each proposed candidate with previously promoted facts already in the knowledge base. Candidates that do not contradict previous knowledge are ranked using a heuristic rule based on the confidence scores of the extractors that proposed the fact, and the top candidates are chosen for promotion subject to score and rank thresholds. Note that the NELL method includes judgments for all input facts, not just those in the test set.

The third method we compare against is the best-performing MLN model from [9], that expresses ontological constraints, and candidate and promoted facts through logical rules similar to those in our model. The MLN uses additional predicates that have confidence values taken from a logistic regression classifier trained using manually labeled data. The MLN uses hard ontological constraints, learns rule weights considering rules independently and using logistic regression, scales weights by the extractor confidences, and uses MC-Sat with a restricted set of atoms to perform approximate inference, reporting output at a .5 marginal probability cutoff, which maximizes the F1 score. The MLN method only generates predictions for a 2-hop neighborhood generated by conditioning on the values of the query set, as described earlier.

Our method, PSL-KGI, uses PSL with quadratic, weighted rules for ontological constraints, entity resolution, and candidate and promoted facts as well as incorporating a prior. We also incorporate the predicates generated for the MLN method for a more equal comparison. We learn weights for all rules, including the prior, using a voted perceptron learning method. The weight learning method generates a set of target values by running inference and conditioning on the training data, and then chooses weights that maximize the agreement with these targets in absence of training data. Since we represent extractor confidence values as soft-truth values, we do not scale the weights of these rules. Using the learned weights, we perform inference on the same neighborhood defined by the query set that is used by the MLN method. We report these results, using a soft-truth threshold of .55 to maximize F1, as PSL-KGI. As Table 3 shows, knowledge graph identification produces modest improvements in both F1 and AUC.

Table 3. Comparing against previous work on the NELL dataset, knowledge graph identification using PSL demonstrates a substantive improvement.

Method	AUC	Prec	Recall	F1
Baseline	0.873	0.781	0.881	0.828
NELL	0.765	0.801	0.580	0.673
MLN	0.899	**0.837**	0.837	0.836
PSL-KGI	**0.904**	0.777	**0.944**	**0.853**

Table 4. Comparing variants of PSL graph identification show the importance of ontological information, but the best performance is achieved when all of the components of knowledge graph identification are combined.

Method	AUC	Prec	Recall	F1
PSL-NoSrcs	0.900	0.770	**0.955**	0.852
PSL-NoER	0.899	0.778	0.944	**0.853**
PSL-NoOnto	0.887	**0.813**	0.839	0.826
PSL-KGI	**0.904**	0.777	0.944	**0.853**

Table 5. Producing a complete knowledge graph reduces performance on the test set, suggesting that the true complexity of the problem is masked when generating a limited set of inferences.

Method	AUC	Prec	Recall	F1
NELL	**0.765**	**0.801**	0.580	0.673
PSL-KGI-Complete	0.718	0.709	**0.929**	**0.804**
PSL-KGI	0.904	0.777	0.944	0.853

Analyzing Variations of Knowledge Graph Identification To better understand the contributions of various components of our model, we explore variants that omit one aspect of the knowledge graph identification model. PSL-NoSrcs removes predicates $\textsc{CandLbl}_T$ and $\textsc{CandRel}_T$ for different candidate sources, replacing them with a single $\textsc{CandLbl}$ or $\textsc{CandRel}$ with the average confidence value across sources. PSL-NoER removes rules from subsection 5.2 used to reason about co-referent entities. PSL-NoOnto removes rules from subsection 5.3 that use ontological relationships to constrain the knowledge graph. While source information and entity resolution both provide benefits, ontological information is clearly a principal contributor to the success of knowledge graph identification. One drawback of our comparisons to previous work is the restriction of the model to a small set of inference targets. The construction of this set obscures some of the challenges presented in real-world data, such as conflicting evidence. To assess the performance of our method in a setting where inference targets do not restrict potentially contradictory inferences, we also ran knowledge graph identification using the same learned weights but with no predefined set of targets, allowing lazy inference to produce a complete knowledge graph. The resulting inference produces a total of 4.9M total facts, which subsumes the test set. We report results on the test set as PSL-KGI-Complete. Allowing the model to optimize on the full knowledge graph instead of just the test set reduced the performance as measured by the particular test set, suggesting that the noise introduced by conflicting evidence does have a significant impact on results. Compared to the NELL scoring method, KGI has lower AUC and precision but higher recall and F1. One possible explanation for this lackluster performance may be the use of weights learned for a different setting. For example, during weight learning the weights for the Mut rule dropped significantly. However, as results on the MusicBrainz data show, knowledge graph identification can be very powerful at recovering a full knowledge graph.

Scalability One advantage of using PSL for knowledge graph identification is the ability to frame complex joint reasoning as a convex optimization. Knowledge graph identification implemented in PSL can handle problems from real-world datasets like NELL, which include millions of candidate facts. Inference when an explicit query set of 70K facts is given (PSL-KGI) requires a mere 10 seconds. The MLN method we compare against takes a few minutes to an hour to run for the same setting. When inferring a complete knowledge graph without known query targets, as in the LinkedBrainz and complete NELL experiments, inference with MLNs is infeasible. In contrast, knowledge graph identification on the NELL dataset can produce the complete knowledge graph containing 4.9M facts in only 130 minutes. The ability to produce complete knowledge graphs in these realistic settings is an important feature of our implementation of knowledge graph identification.

7 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 benefits of our approach on two knowledge graph inference problems: synthetic data from MusicBrainz and noisy, real-world web extractions from NELL. On both datasets, knowledge graph identification produces superior results by combining the strengths of ontological reasoning with graph identification. Moreover, our method is solved through efficient convex optimization allowing previously infeasible problems to be solved on the order of minutes. In the future, we hope to apply knowledge graph identification to larger, more varied problems with richer ontological relationships.

Acknowledgments We would like to thank Shangpu Jiang and Daniel Lowd for sharing their data and offering enthusiastic assistance. This work was partially supported by NSF CAREER grant 0746930 and NSF grants IIS1218488 and CCF0937094. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

1. Ji, H., Grishman, R., Dang, H.: Overview of the Knowledge Base Population Track. In: Text Analysis Conference. (2011)
2. Artiles, J., Mayfield, J., eds.: Workshop on Knowledge Base Population. In Artiles, J., Mayfield, J., eds.: Text Analysis Conference. (2012)
3. Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka, E.R., Mitchell, T.M.: Toward an Architecture for Never-Ending Language Learning. In: AAAI. (2010)
4. Etzioni, O., Banko, M., Soderland, S., Weld, D.S.: Open Information Extraction from the Web. Communications of the ACM **51**(12) (2008)
5. Pasca, M., Lin, D., Bigham, J., Lifchits, A., Jain, A.: Organizing and Searching the World Wide Web of Facts-Step One: the One-million Fact Extraction Challenge. In: AAAI. (2006)
6. Singhal, A.: Introducing the Knowledge Graph: Things, Not Strings (2012) Official Blog (of Google), see: `http://goo.gl/zivFV`.
7. Broecheler, M., Mihalkova, L., Getoor, L.: Probabilistic Similarity Logic. In: UAI. (2010)
8. Cohen, W., McAllester, D., Kautz, H.: Hardening Soft Information Sources. In: KDD. (2000)
9. Jiang, S., Lowd, D., Dou, D.: Learning to Refine an Automatically Extracted Knowledge Base Using Markov Logic. In: ICDM. (2012)
10. Richardson, M., Domingos, P.: Markov Logic Networks. Machine Learning **62**(1-2) (2006)
11. Namata, G.M., Kok, S., Getoor, L.: Collective Graph Identification. In: KDD. (2011)
12. Memory, A., Kimmig, A., Bach, S.H., Raschid, L., Getoor, L.: Graph Summarization in Annotated Data Using Probabilistic Soft Logic. In: Workshop on Uncertainty Reasoning for the Semantic Web (URSW). (2012)
13. Yao, L., Riedel, S., McCallum, A.: Collective Cross-Document Relation Extraction Without Labelled Data. In: EMNLP. (2010)
14. Kimmig, A., Bach, S.H., Broecheler, M., Huang, B., Getoor, L.: A Short Introduction to Probabilistic Soft Logic. In: NIPS Workshop on Probabilistic Programming. (2012)
15. Bach, S.H., Broecheler, M., Getoor, L., O'Leary, D.P.: Scaling MPE Inference for Constrained Continuous Markov Random Fields with Consensus Optimization. In: NIPS. (2012)
16. Dixon, S., Jacobson, K.: LinkedBrainz - A project to provide MusicBrainz NGS as Linked Data see `http://linkedbrainz.c4dmpresents.org/`.
17. Raimond, Y., Abdallah, S., Sandler, M.: The Music Ontology. In: International Conference on Music Information Retrieval. (2007)
18. Davis, I., Newman, R., DArcus, B.: Expression of Core FRBR Concepts in RDF (2005) see `http://vocab.org/frbr/core.html`.
19. Brickley, D., Miller, L.: FOAF Vocabulary Specification 0.98 (2010) see `http://xmlns.com/foaf/spec/20100809.html`.
20. Kobilarov, G., Scott, T., Raimond, Y., Oliver, S., Sizemore, C., Smethurst, M., Bizer, C., Lee, R.: Media Meets Semantic Web–How The BBC uses DBpedia and Linked Data to Make Connections. In: ESWC. (2009)
21. Bizer, C., Seaborne, A.: D2RQ–Treating Non-RDF Databases as Virtual RDF Graphs. In: ISWC. (2004)
22. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: A Core of Semantic Knowledge. In: WWW. (2007)