Joint Information Extraction and Reasoning: A Scalable Statistical Relational Learning Approach

William Yang Wang
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
yww@cs.cmu.edu

William W. Cohen
Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213, USA
wcohen@cs.cmu.edu

Abstract

A standard pipeline for statistical relational learning involves two steps: one first constructs the knowledge base (KB) from text, and then performs the learning and reasoning tasks using probabilistic first-order logics. However, a key issue is that information extraction (IE) errors from text affect the quality of the KB, and propagate to the reasoning task. In this paper, we propose a statistical relational learning model for joint information extraction and reasoning. More specifically, we incorporate context-based entity extraction with structure learning (SL) in a scalable probabilistic logic framework. We then propose a latent context invention (LCI) approach to improve the performance. In experiments, we show that our approach outperforms state-of-the-art baselines over three real-world Wikipedia datasets from multiple domains; that joint learning and inference for IE and SL significantly improve both tasks; that latent context invention further improves the results.

1 Introduction

Information extraction (IE) is often an early stage in a pipeline that contains non-trivial downstream tasks, such as question answering (Mollá et al., 2006), machine translation (Babych and Hartley, 2003), or other applications (Wang and Hua, 2014; Li et al., 2014). Knowledge bases (KBs) populated by IE techniques have also been used as an input to systems that learn rules allowing further inferences to be drawn from the KB (Lao et al., 2011), a task sometimes called KB completion (Socher et al., 2013; Wang et al., 2014; West et al., 2014). Pipelines of this sort frequently suffer from error cascades, which reduces performance of the full system[1].

In this paper, we address this issue, and propose a joint model system for IE and KB completion in a statistical relational learning (SRL) setting (Sutton and McCallum, 2006; Getoor and Taskar, 2007). In particular, we outline a system which takes as input a partially-populated KB and a set of relation mentions in context, and jointly learns: 1) how to extract new KB facts from the relation mentions, and; 2) a set of logical rules that allow one to infer new KB facts. Evaluation of the KB facts inferred by the joint system shows that the joint model outperforms its individual components. We also introduce a novel extension of this model called Latent Context Invention (LCI), which associates latent states with context features for the IE component of the model. We show that LCI further improves performance, leading to a substantial improvement over prior state-of-the-art methods for joint relation-learning and IE.

To summarize our contributions:

- We present a joint model for IE and relational learning in a statistical relational learning setting which outperforms universal schemas (Riedel et al., 2013), a state-of-the-art joint method;
- We incorporate latent context into the joint SRL model, bringing additional improvements.

In next section, we discuss related work. We describe our approach in Section 3. The details of the datasets are introduced in Section 4. We show experimental results in Section 5, discuss in Section 6, and conclude in Section 7.

[1]For example, KBP slot filling is known for its complex pipeline, and the best overall F1 scores (Wiegand and Klakow, 2013; Angeli et al., 2014) for recent competitions are within the range of 30-40.

2 Related Work

In NLP, our work clearly aligns with recent work on joint models of individual text processing tasks. For example, Finkel and Manning (2009) work on the problem of joint IE and parsing, where they use tree representations to combine named entities and syntactic chunks. Recently, Devlin et al. (Devlin et al., 2014) use a joint neural network model for machine translation, and obtain an impressive 6.3 BLEU point improvement over a hierarchical phrase-based system.

In information extraction, weak supervision (Craven et al., 1999; Mintz et al., 2009) is a common technique for extracting knowledge from text, without large-scale annotations. In extracting Infobox information from Wikipedia text, Wu and Weld (2007; 2010) also use a similar idea. In an open IE project, Banko et al. (2007) use a seed KB, and utilize weak supervision techniques to extend it. Note that weakly supervised extraction approaches can be noisy, as a pair of entities in context may be associated with one, none, or several of the possible relation labels, a property which complicates the application of distant supervision methods (Mintz et al., 2009; Riedel et al., 2010; Hoffmann et al., 2011; Surdeanu et al., 2012).

Lao et al. (2012) learned syntactic rules for finding relations defined by "lexico-semantic" paths spanning KB relations and text data. Wang et al. (2015) extends the methods used by Lao et al. to learn mutually recursive relations. Recently, Riedel et al. (2013) propose a matrix factorization technique for relation embedding, but their method requires a large amount of negative and unlabeled examples. Weston et al. (2013) connect text with KB embedding by adding a scoring term, though no shared parameters/embeddings are used. All these prior works make use of text and KBs. Unlike these prior works, our method is posed in an SRL setting, using a scalable probabilistic first-order logic, and allows learning of relational rules that are mutually recursive, thus allowing learning of multi-step inferences. Unlike some prior methods, our method also does not require negative examples, or large numbers of unlabeled examples.

3 Our Approach

In this section, we first briefly review the semantics, inference, and learning procedures of a newly proposed scalable probabilistic logic called ProPPR (Wang et al., 2013; Wang et al., 2014). Then, we describe the joint model for information extraction and relational learning. Finally, a latent context invention theory is proposed for enhancing the performance of the joint model.

```
about(X,Z) :- handLabeled(X,Z)          # base.
about(X,Z) :- sim(X,Y),about(Y,Z)       # prop.
sim(X,Y) :- links(X,Y)                  # sim,link.
sim(X,Y) :-
   hasWord(X,W),hasWord(Y,W),
   linkedBy(X,Y,W)                      # sim,word.
linkedBy(X,Y,W) :- true                 # by(W).
```

Table 1: A simple program in ProPPR. See text for explanation.

3.1 ProPPR: Background

Below we will give an informal description of ProPPR, based on a small example. More formal descriptions can be found elsewhere (Wang et al., 2013).

ProPPR (for Programming with Personalized PageRank) is a stochastic extension of the logic programming language Prolog. A simple program in ProPPR is shown in Table 1. Roughly speaking, the upper-case tokens are variables, and the ":-" symbol means that the left-hand side (the *head* of a rule) is implied by the conjunction of conditions on the right-hand size (the *body*). In addition to the rules shown, a ProPPR program would include a *database* of *facts*: in this example, facts would take the form *handLabeled(page,label)*, *hasWord(page,word)*, or *linkedBy(page1,page2)*, representing labeled training data, a document-term matrix, and hyperlinks, respectively. The condition "true" in the last rule is "syntactic sugar" for an empty body.

In ProPPR, a user issues a query, such as "about(a,X)?", and the answer is a set of possible bindings for the free variables in the query (here there is just one such varable, "X"). To answer the query, ProPPR builds a *proof graph*. Each node in the graph is a list of conditions $R_1, \ldots, R_k$ that remain to prove, interpreted as a conjunction. To find the children of a node $R1, \ldots, Rk$, you look for either

1. database facts that match R_1, in which case the appropriate variables are bound, and R_1 is removed from the list, or;

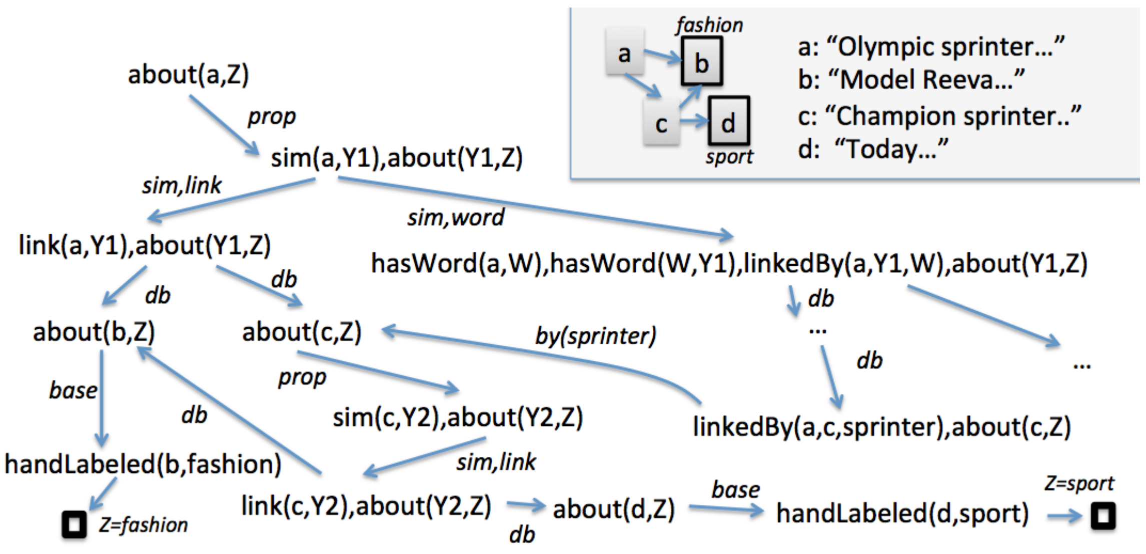

Figure 1: A partial proof graph for the query *about(a,Z)*. The upper right shows the link structure between documents a, b, c, and d, and some of the words in the documents. Restart links are not shown.

2. a rule $A \leftarrow B_1, \ldots, B_m$ with a head A that matches R_1, in which case again the appropriate variables are bound, and R_1 is replaced with the body of the rule, resulting in the new list $B_1, \ldots, B_m, R_2, \ldots, R_k$.

The procedures for "matching" and "appropriately binding variables" are illustrated in Figure 1.[2] An empty list of conditions (written $\Box$ in the figure) corresponds to a complete proof of the initial query, and by collecting the required variable bindings, this proof can be used to determine an answer to the initial query.

In Prolog, this proof graph is constructed on-the-fly in a depth-first, left-to-right way, returning the first solution found, and backtracking, if requested, to find additional solutions. In ProPPR, however, we will define a *stochastic process on the graph*, which will generate a score for each node, and hence a score for each answer to the query. The stochastic process used in ProPPR is *personalized PageRank* (Page et al., 1998; Csalogny et al., 2005), also known as random-walk-with-restart. Intuitively, this process upweights solution nodes that are reachable by *many short proofs* (i.e., short paths from the query node.) Formally, personalized PageRank is the fixed point of the iteration

$$\mathbf{p}^{t+1} = \alpha\chi_{v_0} + (1-\alpha)W\mathbf{p}^t \tag{1}$$

[2]The edge annotations will be discussed later.

where $\mathbf{p}[u]$ is the weight assigned to u, v_0 is the seed (i.e., query) node, χ_{v_0} is a vector with $\chi_{v_0}[v_0] = 1$ and $\chi_{v_0}[u] = 0$ for $u \neq v$, and W is a matrix of transition probabilities, i.e., $W[v,u]$ is the probability of transitioning from node u to a child node v. The parameter α is the reset probability, and the transition probabilities we use will be discussed below.

Like Prolog, ProPPR's proof graph is also constructed on-the-fly, but rather than using depth-first search, we use PageRank-Nibble, a fast approximate technique for incrementally exploring a large graph from a an initial "seed" node (Andersen et al., 2008). PageRank-Nibble takes a parameter ϵ and will return an approximation $\hat{\mathbf{p}}$ to the personalized PageRank vector $\mathbf{p}$, such that each node's estimated probability is within ϵ of correct.

We close this background section with some final brief comments about ProPPR.

Scalability. ProPPR is currently limited in that it uses memory to store the fact databases, and the proof graphs constructed from them. ProPPR uses a special-purpose scheme based on sparse matrix representations to store facts which are triples, which allows it to accomodate databases with hundreds of millions of facts in tens of gigabytes.

With respect to run-time, ProPPR's scalability is improved by the fast approximate inference scheme used, which is often an order of magnitude faster than power iteration for moderate-sized problems (Wang et al., 2013). Experimen-

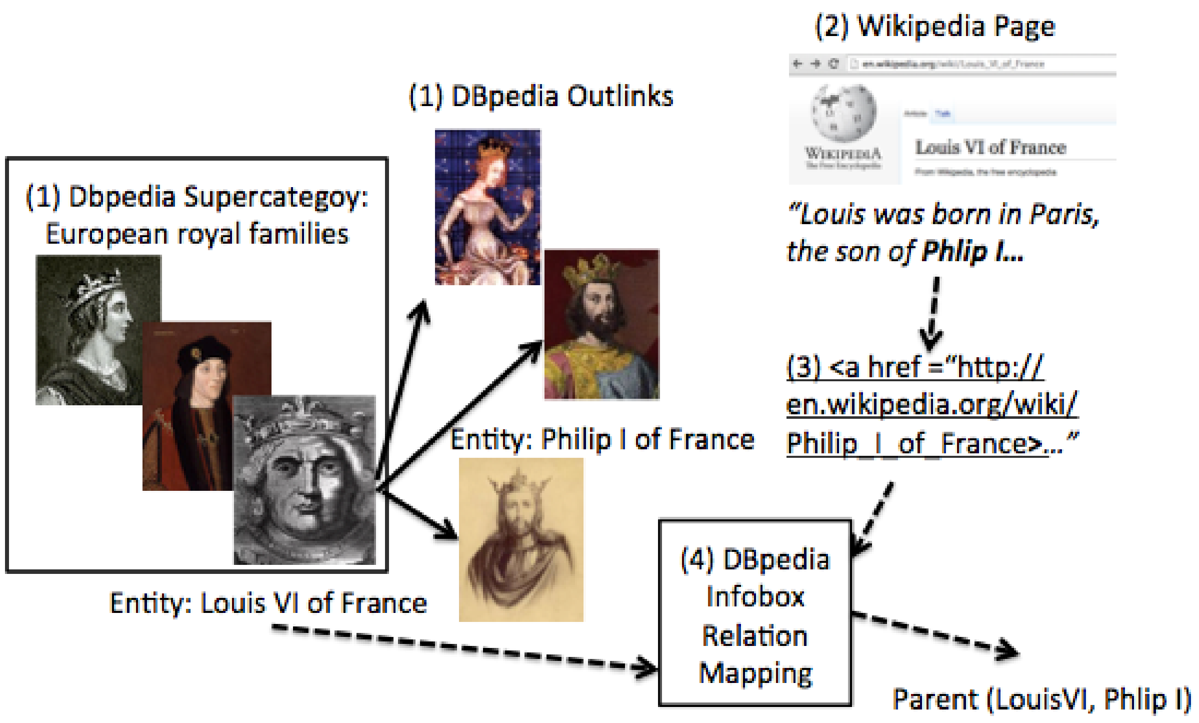
(2) Wikipedia Page
(1) DBpedia Outlinks
Louis VI of France
(1) Dbpedia Supercategoy: European royal families
"Louis was born in Paris, the son of Phlip I...
(3) <a href ="http://en.wikipedia.org/wiki/Philip_I_of_France>..."
Entity: Philip I of France
(4) DBpedia Infobox Relation Mapping
Entity: Louis VI of France
Parent (LouisVI, Phlip I)

Figure 2: The data generation example as described in subsection 3.2.

tation and learning are also sped up because with PageRank-Nibble, each query is answered using a "small"—size $O(\frac{1}{\alpha\epsilon})$—proof graph. Many operations required in learning and experimentation can thus be easily parallized on a multi-core machine, by simply distributing different proof graphs to different threads.

Parameter learning. Personalized PageRank scores are defined by a transition probability matrix W, which is parameterized as follows. ProPPR allows "feature generators" to be attached to its rules, as indicated by the code after the hashtags in the example program.[3] Since edges in the proof graph correspond to rule matches, the edges can also be labeled by features, and a weighted combination of these features can be used to define a total weight for each edge, which finally can be normalized used to define the transition matrix W. Learning can be used to tune these weights to data; ProPPR's learning uses a parallelized SGD method, in which inference on different examples is performed in different threads, and weight updates are synchronized.

Structure learning. Prior work (Wang et al., 2014) has studied the problem of learning a ProPPR theory, rather than simply tuning parameters in an existing theory, a process called *structure learning* (SL). In particular, Wang et al. (2014) propose a scheme called the *structural gradient* which scores rules in some (possibly large) user-defined space $\mathcal{R}$ of potential rules, which can be viewed as instantiations of rule templates, such as the ones shown in the left-hand side of Table 2.

For completeness, we will summarize briefly the approach used in (Wang et al., 2014). The space of potential rules $\mathcal{R}$ is defined by a "second-order abductive theory", which conceptually is an interpreter that constructs proofs using all rules in $\mathcal{R}$. Each rule template is mapped to two clauses in the interpreter: one simulates the template (for any binding), and one "abduces" the specific binding (facts) from the KB. Associated with the use of the abductive rule is a feature corresponding to a particular binding for the template. The gradient of these features indicates which instantiated rules can be usefully added to the theory. More details can be found in (Wang et al., 2014).

[3]For instance, when matching the rule "sim(X,Y) :- links(X,Y)" to a condition such as "sim(a,X)" the two features "sim" and "link" are generated; likewise when matching the rule "linkedBy(X,Y,W) :- true" to the condition "linkedBy(a,c,sprinter)" the feature "by(sprinter)" is generated.

	Rule template	ProPPR clause
Structure learning		
(a)	P(X,Y) :- R(X,Y)	interp(P,X,Y) :- interp0(R,X,Y),abduce_if(P,R). abduce_if(P,R) :- true # f_if(P,R).
(b)	P(X,Y) :- R(Y,X)	interp(P,X,Y) :- interp0(R,Y,X),abduce_ifInv(P,R). abduce_ifInv(P,R) :- true # f_ifInv(P,R).
(c)	P(X,Y) :- R1(X,Z),R2(Z,Y)	interp(P,X,Y) :- interp0(R1,X,Z),interp0(R2,Z,Y), abduce_chain(P,R1,R2). abduce_chain(P,R1,R2) :- true # f_chain(P,R1,R2).
	base case for SL interpreter	interp0(P,X,Y) :- rel(R,X,Y).
	insertion point for learned rules	interp0(P,X,Y) :- *any rules learned by SL.*
Information extraction		
(d)	R(X,Y) :- link(X,Y,W), indicates(W,R).	interp(R,X,Y) :- link(X,Y,W),abduce_indicates(W,R). abduce_indicates(W,R) :- true #f_ind1(W,R).
(e)	R(X,Y) :- link(X,Y,W1), link(X,Y,W2), indicates(W1,W2,R).	interp(R,X,Y) :- link(X,Y,W1),link(X,Y,W2), abduce_indicates(W1,W2,R). abduce_indicates(W1,W2,R) :- true #f_ind2(W1,W2,R).
Latent context invention		
(f)	R(X,Y) :- latent(L), link(X,Y,W), indicates(W,L,R)	interp(R,X,Y) :- latent(L),link(X,Y,W),abduce_latent(W,L,R). abduce_latent(W,L,R) :- true #f_latent1(W,L,R).

(g)	R(X,Y) :- latent(L1),latent(L2)
	link(X,Y,W),
	indicates(W,L1,L2,R)
		interp(R,X,Y) :- latent(L1),latent(L2),link(X,Y,W),
		abduce_latent(W,L1,L2,R).
		abduce_latent(W,L1,L2,R) :- true #f_latent2(W,L1,L2,R).

Table 2: The ProPPR template and clauses for joint structure learning and information extraction.

3.2 Joint Model for IE and SRL

Dataset Generation The KBs and text used in our experiments were derived from Wikipedia. Briefly, we choose a set of closely-related pages from a hand-selected Wikipedia list. These pages define a set of entities $\mathcal{E}$, and a set of commonly-used Infobox relations $\mathcal{R}$ between these entities define a KB. The relation mentions are hyperlinks between the pages, and the features of these relation mentions are words that appear nearby these links. This information is encoded in a single relation *link(X,Y,W)*, which indicates that there is hyperlink between Wikipedia pages X to Y which is near the context word W. The Infobox relation triples are stored in another relation, *rel(R,X,Y)*. [4]

Figure 2 shows an example. We first find the "*European royal families*" to find a list of entities $\mathcal{E}$. This list contains the page "*Louis VI of France*", the *source entity*, which contains an outlink to the *target entity page* "*Philip I of France*". On the source page, we can find the following text: "*Louis was born in Paris, the son of **Philip I** and his first wife, Bertha of Holland.*" From Infobox data, we also may know of a relationship between the source and target entities: in this case, the target entity is the *parent* of the source entity.

Theory for Joint IE and SL The structure learning templates we used are identical to those used in prior work (Wang et al., 2014), and are summarized by the clauses (a-c) in Table 2. In the templates in the left-hand side of the table, P, R, $R1$ and $R2$ are variables in the template, which will be bound to specific relations found to be useful in prediction. (The interpreter rules on the right-hand side are provided for completeness, and can be ignored by readers not deeply familiar with the work of (Wang et al., 2014).)

The second block of the table contains the templates used for IE. For example, to understand template (d), recall that the predicate *link* indicates a hyperlink from Wikipedia page X to

[4]In more detail, the extraction process was as follows. (1) We used a DBpedia dump of categories and hyperlink structure to find pages in a category; sometimes, this included crawling a supercategory page to find categories and then entities. (2) We used the DBpedia hyperlink graph to find the target entity pages, downloaded the most recent (2014) version of each of these pages, and collected relevant hyperlinks and anchor text, together with 80 characters of context to either side.

Y, which includes the context word W between two entities X and Y. The abductive predicate *abduce_indicates* activates a feature template, in which we learn the degree of association of a context word and a relation from the training data. These rules essentially act as a trainable classifier which classifies entity pairs based on the hyperlinks they that contain them, and classifies the hyperlinks according to the relation they reflect, based on context-word features.

Notice that the learner will attempt to tune word associations to match the gold *rel* facts used as training data, and that doing this does not require assigning labels to individual links, as would be done in a traditional distant supervision setting: instead these labels are essentially left latent in this model. Similar to "deep learning" approaches, the latent assignments are provided not by EM, but by hill-climbing search in parameter space.

A natural extension to this model is to add a bilexical version of this classifier in clause (e), where we learn a feature which conjoins word $W1$, word $W2$, and relation R.

Combining the clauses from (a) to (e), we derive a hybrid theory for joint SL and IE: the structure learning section involves a second-order probabilistic logic theory, where it searches the relational KB to form plausible first-order relational inference clauses. The information extraction section from (d) to (e) exploits the distributional similarity of contextual words for each relation, and extracts relation triples from the text, using distant supervision and latent labels for relation mentions (which in our case are hyperlinks). Training this theory as a whole trains it to perform joint reasoning to facts for multiple relations, based on relations that are known (from the partial KB) or inferred from the IE part of the theory. Both parameters for the IE portion of the theory and inference rules between KB relations are learned.[5]

Latent Context Invention Note that so far both the IE clauses (d-e) are fully observable: there are no latent predicates or variables. Recent work (Riedel et al., 2013) suggests that learning latent representations for words improves performance in predicting relations. Perhaps this is because such latent representations can better model the semantic information in surface forms, which are often ambiguous.

We call our method latent context invention (LCI), and it is inspired from literature in predicate invention (Kok and Domingos, 2007).[6] LCI applies the idea of predicate invention to the context space: instead of inventing new predicates, we now invent a *latent context property* that captures the regularities among the similar relational lexical items. To do this, we introduce some additional rules of the form *latent(1) :- true*, *latent(2) :- true*, etc, and allow the learner to find appropriate weights for pairing these arbitrarily-chosen values with specific words. This is implemented by template (f) in Table 2. Adding this to the joint theory means that we will learn to map surface-level lexical items (words) to the "invented" latent context values and also to relation.

Another view of LCI is that we are learning a latent embedding of words jointly with relations. In template (f) we model a single latent dimension, but to model higher-dimensional latent variables, we can add the clauses such as (g), which constructs a two-dimensional latent space. Below we will call this variant method hLCI.

4 Datasets

Using the data generation process that we described in subsection 3.2, we extract two datasets from the supercategories of "*European royal families*" and "*American people of English descent*, and third geographic dataset using three lists: "*List of countries by population*", "*List of largest cities and second largest cities by country*" and "*List of national capitals by population*".

For the *royal* dataset, we have 2,258 pages with 67,483 source-context-target mentions, and we use 40,000 for training, and 27,483 for testing. There are 15 relations[7]. In the *American* dataset, we have 679 pages with 11,726 mentions, and we use 7,000 for training, and 4,726 for testing. This dataset includes 30 relations[8]. As for the *Geo* dataset, there are 497

[5]In in addition to finding rules which instantiate the templates, weights on these rules are also learned.

[6]To give some background on this nomenclature, we note that the SL method is inspired by Cropper and Muggleton's Metagol system (Cropper and Muggleton, 2014), which includes predicate invention. In principle predicates could be invented by SL, by extending the interpreter to consider "invented" predicate symbols as binding to its template variables (e.g., P and R); however, in practice invented predicates leads to close dependencies between learned rules, and are highly sensitive to the level of noise in the data.

[7]birthPlace, child, commander, deathPlace, keyPerson, knownFor, monarch, parent, partner, predecessor, relation, restingPlace, spouse, successor, territory

[8]architect, associatedBand, associatedMusicalArtist, au-

pages with 43,475 mentions, and we use 30,000 for training, and 13,375 for testing. There are 10 relations[9]. The datasets are freely available for download at `http://www.cs.cmu.edu/~yww/data/jointIE+Reason.zip`.

5 Experiments

To evaluate these methods, we use the setting of Knowledge Base completion (Socher et al., 2013; Wang et al., 2014; West et al., 2014). We randomly remove a fixed percentage of facts in a training knowledge base, train the learner from the partial KB, and use the learned model to predict facts in the test KB. KB completion is a well-studied task in SRL, where multiple relations are often needed to fill in missing facts, and thus reconstruct the incomplete KB. Following prior work (Riedel et al., 2013; Wang et al., 2013), we use mean average precision (MAP) as the evaluation metric.

5.1 Baselines

To understand the performance of our joint model, we compare with three prior methods. **Structure Learning (SL)** includes the second-order relation learning templates (a-c) from Table 2. **Information Extraction (IE)** includes only templates (d) and (e). **Markov Logic Networks (MLN)** is the Alchemy's implementation[10] of Markov Logic Networks (Richardson and Domingos, 2006), using the first-order clauses learned from SL method[11]. We used conjugate gradient weight learning (Lowd and Domingos, 2007) with 10 iterations. Finally, **Universal Schema** is a state-of-the-art matrix factorization based universal method for jointly learning surface patterns and relations. We used the code and parameter settings for the best-performing model (NFE) from (Riedel et al., 2013).

As a final baseline method, we considered a simpler approach to clustering context words, which we called **Text Clustering**, which used the following template:

R(X,Y) :-
clusterID(C),link(X,Y,W),
cluster(C,W),related(R,W).

Here surface patterns are grouped to form latent clusters in a relation-*independent* fashion.

5.2 The Effectiveness of the Joint Model

Our experimental results are shown in 3. The left-most part of the table concerns the Royal dataset. We see that the universal schema approach outperforms the MLN baseline in most cases, but ProPPR's SL method substantially improves over MLN's conjugated gradient learning method, and the universal schema approach. This is perhaps surprising, as the universal schema approach is also a joint method: we note that in our datasets, unlike the New York Times corpus used in (Riedel et al., 2013), large numbers of unlabeled examples are not available. The unigram and bilexical IE models in ProPPR also perform well—better than SL on this data. The joint model outperforms the baselines, as well as the separate models. The difference is most pronounced when the background KB gets noisier: the improvement with 10% missing setting is about 1.5 to 2.3% MAP, while with 50% missing data, the absolute MAP improvement is from 8% to 10%.

In the next few columns of Table 3, we show the KB completion results for the *Geo* dataset. This dataset has fewer relations, and the most common one is *country*. The overall MAP scores are much higher than the previous dataset. MLN's results are good, but still generally below the universal schema method. On this dataset, the universal schema method performs better than the IE only model for ProPPR in most settings. However, the ProPPRjoint model still shows large improvements over individual models and the baselines: the absolute MAP improvement is 22.4%.

Finally, in the rightmost columns of Table 3, we see that the overall MAP scores for the *American* dataset are relatively lower than other datasets, perhaps because it is the smallest of the three. The universal schema approach consistently outperforms the MLN model, but not ProPPR. On this dataset the SL-only model in ProPPR outperforms the IE-only models; however, the joint models still outperform individual ProPPR models from 1.5% to 6.4% in MAP.

thor, birthPlace, child, cinematography, deathPlace, director, format, foundationOrganisation, foundationPerson, influenced, instrument, keyPerson, knownFor, location, musicComposer, narrator, parent, president, producer, relation, relative, religion, restingPlace, spouse, starring, successor, writer

[9]archipelago, capital, country, daylightSavingTimeZone, largestSettlement, leaderTitle, mottoFor, timeZone, twinCity, twinCountry

[10]http://alchemy.cs.washington.edu/

[11]We also experimented with Alchemy's structure learning, but it was not able to generate results in 24 hours.

	Royal					Geo					American				
% missing	10%	20%	30%	40%	50%	10%	20%	30%	40%	50%	10%	20%	30%	40%	50%
Baselines															
MLN	60.8	43.7	44.9	38.8	38.8	80.4	79.2	68.1	66.0	68.0	54.0	56.0	51.2	41.0	13.8
Universal Schema	48.2	53.0	52.9	47.3	41.2	82.0	84.0	75.7	77.0	65.2	56.7	51.4	55.9	54.7	51.3
SL	79.5	77.2	74.8	65.5	61.9	83.8	80.4	77.1	72.8	67.2	73.1	70.0	71.3	67.1	61.7
IE only															
IE (U)	81.3	78.5	76.4	75.7	70.6	83.9	79.4	73.1	71.6	65.2	63.4	61.0	60.2	61.4	54.4
IE (U+B)	81.1	78.1	76.2	75.5	70.3	84.0	79.5	73.3	71.6	65.3	64.3	61.2	61.1	62.1	55.7
Joint															
SL+IE (U)	82.8	80.9	79.1	77.9	78.6	89.5	89.4	89.3	88.1	87.6	74.0	73.3	73.7	70.5	68.0
SL+IE (U+B)	83.4	82.0	80.7	79.7	80.3	89.6	89.6	89.5	88.4	87.7	**74.6**	73.5	74.2	70.9	68.4
Joint + Latent															
Joint + Clustering	**83.5**	82.3	81.2	80.2	80.7	89.8	89.6	89.5	88.8	88.4	**74.6**	73.9	74.4	71.5	69.7
Joint + LCI	**83.5**	**82.5**	81.5	80.6	81.1	**89.9**	**89.8**	**89.7**	89.1	89.0	**74.6**	74.1	74.5	72.3	70.3
Joint + LCI + hLCI	**83.5**	**82.5**	**81.7**	**81.0**	**81.3**	**89.9**	89.7	**89.7**	**89.6**	**89.5**	**74.6**	**74.4**	**74.6**	**73.6**	**72.1**

Table 3: The MAP results for KB completion on three datasets. U: unigram. B: bigram. Best result in each column is highlighted in **bold**.

The averaged training runtimes on an ordinary PC for unigram joint model on the above *Royal, Geo, American* datasets are 38, 36, and 29 seconds respectively, while the average testing times are 11, 10, and 9 seconds. For bilexical joint models, the averaged training times are 25, 10, and 10 minutes respectively, whereas the testing times are 111, 28, and 26 seconds respectively.

5.3 The Effectiveness of LCI

Finally we consider the latent context invention (LCI) approach. The last three rows of Table 3 show the performances of LCI and hHCI. We compare it here with the best previous approach, the joint IE + SL model, and text clustering approach.

For the *Royal* dataset, first, the LCI and hLCI models clearly improve over joint IE and SL. In noisy conditions of missing 50% facts, the biggest improvement of LCI/hLCI is 2.4% absolute MAP.

From the *Geo* dataset, we see that the joint models and joint+latent models have similar performances in relatively clean conditions (10%-30%) facts missing. However, in noisy conditions, we the LCI and hLCI model has an advantage of between 1.5% to 1.8% in absolute MAP.

Finally, the results for the *American* dataset show a consistent trend: again, in noisy conditions (missing 40% to 50% facts), the latent context models outperform the joint IE + SL models by 2.9% and 3.7% absolute MAP scores.

Although the LCI approach is inspired by predicate invention in inductive logic programming, our result is also consistent with theories of generalized latent variable modeling in probabilistic graphical models and statistics (Skrondal and Rabe-Hesketh, 2004): modeling hidden variables helps take into account the measurement (observation) errors (Fornell and Larcker, 1981) and results in a more robust model.

6 Discussions

Compared to state-of-the-art joint models (Riedel et al., 2013) that learn the latent factor representations, our method gives strong improvements in performance on three datasets with various settings. Our model is also trained to retrieve a target entity from a relation name plus a source entity, and does not require large samples of unlabeled or negative examples in training.

Another advantage of the ProPPR model is that they are explainable. For example, below are the features with the highest weights after joint learning from the *Royal* dataset, written as predicates or rules:

indicates("mother",parent)
indicates("king",parent)
indicates("spouse",spouse)
indicates("married",spouse)
indicates("succeeded",successor)
indicates("son",successor)

parent(X,Y) :- successor(Y,X)
successor(X,Y) :- parent(Y,X)
spouse(X,Y) :- spouse(Y,X)
parent(X,Y) :- predecessor(X,Y)
successor(Y,X) :- spouse(X,Y)
predecessor(X,Y) :- parent(X,Y)

Here we see that our model is able to learn that the keywords "mother" and "king" that are indicators of the relation *parent*, that the keywords “spouse” and “married” indicate the relation *spouse*, and the keywords “succeeded” and “son” indicate the relation *successor*. Interestingly, our joint model is also able to learn the inverse relation *successor* for the relation *parent*, as well as the similar relational predicate *predecessor* for *parent*.

7 Conclusions

In this paper, we address the issue of joint information extraction and relational inference. To be more specific, we introduce a holistic probabilistic logic programming approach for fusing IE contexts with relational KBs, using locally groundable inference on a joint proof graph. We then propose a latent context invention technique that learns relation-specific latent clusterings for words. In experiments, we show that joint modeling for IE and SRL improves over prior state-of-the-art baselines by large margins, and that the LCI model outperforms various fully baselines on three real-world Wikipedia dataset from different domains. In the future, we are interested in extending these techniques to also exploit unlabeled data.

Acknowledgment

This work was sponsored in part by DARPA grant FA87501220342 to CMU and a Google Research Award.

References

Reid Andersen, Fan R. K. Chung, and Kevin J. Lang. 2008. Local partitioning for directed graphs using pagerank. *Internet Mathematics*, 5(1):3–22.

Gabor Angeli, Julie Tibshirani, Jean Y Wu, and Christopher D Manning. 2014. Combining distant and partial supervision for relation extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Bogdan Babych and Anthony Hartley. 2003. Improving machine translation quality with automatic named entity recognition. In *Proceedings of the 7th International EAMT workshop on MT and other Language Technology Tools, Improving MT through other Language Technology Tools: Resources and Tools for Building MT*, pages 1–8. Association for Computational Linguistics.

Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction for the web. In *IJCAI*, volume 7, pages 2670–2676.

Mark Craven, Johan Kumlien, et al. 1999. Constructing biological knowledge bases by extracting information from text sources. In *ISMB*, volume 1999, pages 77–86.

Andrew Cropper and Stephen H Muggleton. 2014. Can predicate invention in meta-interpretive learning compensate for incomplete background knowledge? *Proceedings of the 24th International Conference on Inductive Logic Programming*.

Kroly Csalogny, Dniel Fogaras, Balzs Rcz, and Tams Sarls. 2005. Towards scaling fully personalized PageRank: Algorithms, lower bounds, and experiments. *Internet Mathematics*, 2(3):333–358.

Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. 2014. Fast and robust neural network joint models for statistical machine translation. In *52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MD, USA, June*.

Jenny Rose Finkel and Christopher D Manning. 2009. Joint parsing and named entity recognition. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 326–334. Association for Computational Linguistics.

Claes Fornell and David F Larcker. 1981. Evaluating structural equation models with unobservable variables and measurement error. *Journal of marketing research*, pages 39–50.

Lise Getoor and Ben Taskar. 2007. *Introduction to statistical relational learning*. MIT press.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 541–550. Association for Computational Linguistics.

Stanley Kok and Pedro Domingos. 2007. Statistical predicate invention. In *Proceedings of the 24th international conference on Machine learning*, pages 433–440. ACM.

Ni Lao, Tom M. Mitchell, and William W. Cohen. 2011. Random walk inference and learning in a large scale knowledge base. In *EMNLP*, pages 529–539. ACL.

Ni Lao, Amarnag Subramanya, Fernando C. N. Pereira, and William W. Cohen. 2012. Reading the web with learned syntactic-semantic inference rules. In *EMNLP-CoNLL*, pages 1017–1026. ACL.

Jiwei Li, Alan Ritter, and Eduard Hovy. 2014. Weakly supervised user profile extraction from twitter. ACL.

Daniel Lowd and Pedro Domingos. 2007. Efficient weight learning for markov logic networks. In *Knowledge Discovery in Databases: PKDD 2007*, pages 200–211. Springer.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Diego Mollá, Menno Van Zaanen, and Daniel Smith. 2006. Named entity recognition for question answering. *Proceedings of ALTW*, pages 51–58.

Larry Page, Sergey Brin, R. Motwani, and T. Winograd. 1998. The PageRank citation ranking: Bringing order to the web. In *Technical Report, Computer Science department, Stanford University*.

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

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *Proceedings of NAACL-HLT*, pages 74–84.

Anders Skrondal and Sophia Rabe-Hesketh. 2004. *Generalized latent variable modeling: Multilevel, longitudinal, and structural equation models*. CRC Press.

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems*, pages 926–934.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 455–465. Association for Computational Linguistics.

Charles Sutton and Andrew McCallum. 2006. An introduction to conditional random fields for relational learning. *Introduction to statistical relational learning*, pages 93–128.

William Yang Wang and Zhenhao Hua. 2014. A semiparametric gaussian copula regression model for predicting financial risks from earnings calls. In *Proceedings of the 52th Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, Baltimore, MD, USA, June. ACL.

William Yang Wang, Kathryn Mazaitis, and William W Cohen. 2013. Programming with personalized pagerank: a locally groundable first-order probabilistic logic. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 2129–2138. ACM.

William Yang Wang, Kathryn Mazaitis, and William W Cohen. 2014. Structure learning via parameter learning. *Proceedings of the 23rd ACM International Conference on Information and Knowledge Management (CIKM 2014)*.

William Yang Wang, Kathryn Mazaitis, Ni Lao, Tom Mitchell, and William W Cohen. 2015. Efficient inference and learning in a large knowledge base: Reasoning with extracted information using a locally groundable first-order probabilistic logic. *Machine Learning Journal*.

Robert West, Evgeniy Gabrilovich, Kevin Murphy, Shaohua Sun, Rahul Gupta, and Dekang Lin. 2014. Knowledge base completion via search-based question answering. In *Proceedings of the 23rd international conference on World wide web*, pages 515–526. International World Wide Web Conferences Steering Committee.

Jason Weston, Antoine Bordes, Oksana Yakhnenko, Nicolas Usunier, et al. 2013. Connecting language and knowledge bases with embedding models for relation extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1366–1371.

Benjamin Roth Tassilo Barth Michael Wiegand and Mittul Singh Dietrich Klakow. 2013. Effective slot filling based on shallow distant supervision methods. *Proceedings of NIST KBP workshop*.

Fei Wu and Daniel S Weld. 2007. Autonomously semantifying wikipedia. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 41–50. ACM.

Fei Wu and Daniel S Weld. 2010. Open information extraction using wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 118–127. Association for Computational Linguistics.