ProPPR: Efficient First-Order Probabilistic Logic Programming for Structure Discovery, Parameter Learning, and Scalable Inference

William Yang Wang and **Kathryn Mazaitis** and **William W. Cohen**
School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213 USA.
{yww,krivard,wcohen}@cs.cmu.edu

Abstract

A key challenge in statistical relational learning is to develop a semantically rich formalism that supports efficient probabilistic reasoning using large collections of extracted information. This paper presents a new, scalable probabilistic logic called ProPPR, which further extends stochastic logic programs (SLP) to a framework that enables efficient learning and inference on graphs: using an abductive second-order probabilistic logic, we show that first-order theories can be automatically generated via parameter learning; that in parameter learning, weight learning can be performed using parallel stochastic gradient descent with a supervised personalized PageRank algorithm; and that most importantly, queries can be approximately grounded with a small graph, and inference is independent of the size of the database.

Introduction

One key problem for scaling up first-order probabilistic first-order logics is that queries are typically answered by "grounding" the query—i.e., mapping it to a propositional representation, and then performing propositional inference—and for many logics, the size of the "grounding" can be extremely large for large databases.

We propose a first-order probabilistic language which is well-suited to approximate "local" grounding: we present an extension to *stochastic logic programs* (SLP) (Cussens 2001) that is biased towards short derivations, and show that this is related to *personalized PageRank* (PPR) (Page et al. 1998; Chakrabarti 2007) on a linearized version of the proof space. Based on the connection to PPR, we develop a proveably-correct approximate inference scheme, and an associated proveably-correct approximate grounding scheme: specifically, we show that it is possible to prove a query, or to build a graph which contains the information necessary for weight-learning, in time $O(\frac{1}{\alpha\epsilon})$, where α is a reset parameter associated with the bias towards short derivations, and ϵ is the worst-case approximation error across all intermediate stages of the proof. This means that both inference and learning can be approximated in time *independent of the size of the underlying database*—a surprising and important result. The ability to locally ground queries has another important consequence: it is possible to decompose the problem of weight-learning to a number of moderate-size subtasks (in fact, tasks of size $O(\frac{1}{\alpha\epsilon})$ or less) which are weakly coupled. Based on this we outline a parallelization scheme, which in our initial implementation provides an order-of-magnitude speedup in learning time.

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

```
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).
```

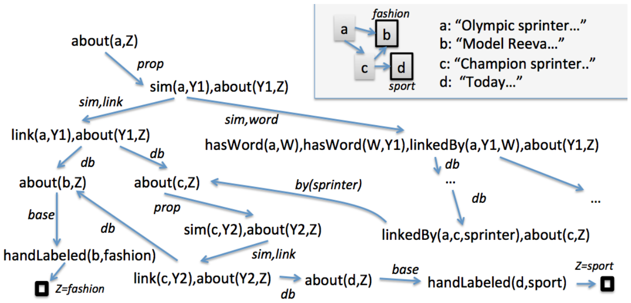

Figure 1: A partial proof graph for the query *about(a,Z)*. The upper right shows the link structure between documents a, b, c, and d, and some of the words in the documents.

ProPPR[1]

Table 1 and Figure 1 illustrate a ProPPR theory and a corresponding proof graph. We refer the reader to prior papers (Wang, Mazaitis, and Cohen 2013; Wang et al. 2014) for a detailed explanation of ProPPR's semantics: briefly, a ProPPR theory T is a Horn clause (Prolog) theory, where each clause c is associated with a function ϕ_c which computes a set of features. ProPPR builds a graph rooted at the query goal, where each node corresponds to a conjunction of

[1] Programming with Personalized PageRank

goals $A_1, \dots, A_k$ and a substitution θ that imply the query goal (i.e. $T \wedge (A_1, \dots, A_k\theta) \rightarrow Q\theta$), and each edge corresponds to the application of a clause c. The feature-functions ϕ_c are simple: each feature corresponds to a Prolog goal, which may include bound variables from the head of the clause, the bindings of which are supplied when ϕ_c is invoked. A node corresponding to an the empty conjunction (denoted $\Box$) is a leaf of the graph, are corresponds to a completed proof of the query goal. Every edge created by the clause c is labeled with the features produced by ϕ_c for that application of c. Note that each different $\Box$ node corresponds to a different proof for Q, and different proofs may be associated with different substitutions θ and hence different solutions to the query Q.

Finally, we associated a score with each node in the graph by performing a personalized PageRank (PPR) (aka "random walk with restart") process, where transition probabilities are based on edge weights, which are in turn determined by a weighted function of the features.

ProPPR allows a fast approximate proof procedure, in which only a small subset of the full proof graph is generated. In particular, if α upper-bounds the reset probability, and d upperbounds the degree of nodes in the graph, then one can efficiently find a subgraph with $O(\frac{1}{\alpha\epsilon})$ nodes which approximates the weight for every node within an error of $d\epsilon$ (Wang, Mazaitis, and Cohen 2013), using a variant of the PageRank-Nibble algorithm of Reid *et al* (Andersen, Chung, and Lang 2008).

Experiments

Parameter Learning

ProPPR's parameter learning framework is implemented using a parallel stochastic gradient descent variant to optimize the log loss using the supervised personalized PageRank algorithm (Backstrom and Leskovec 2011): given the training queries, we perform a random walk with restart process, and upweight the edges that are more likely to end up with a known positive solution. In experiments (Wang, Mazaitis, and Cohen 2013), we show that learning in ProPPR helps improving the AUC of an entity resolution experiment from 0.68 to 0.8 on the CORA dataset; and that significant speedup in parallel learning can be achieved by multithreading. We also demonstrate that, for a bag-of-words classification task on the WebKB dataset, parameter learning for ProPPR is able to improve the AUC from 0.5 to 0.797; and that the average training time for 16 threads is only about 2 minutes.

Scalable Inference

To demonstrate the effectiveness of ProPPR's inference scheme, we also consider the inference time of ProPPR on the Cora entity resolution dataset, comparing to various MLN inference methods. In our CIKM paper (Wang, Mazaitis, and Cohen 2013), we fix the amount of test queries, and increase the size of the database for entity resolution of the CORA dataset. We empirically show that the inference time in ProPPR is independent of the size of the database, whereas all the inference methods (Gibbs sampling, Lifted Belief Propagation, and MAP) in MLN depend on the size of the dataset. Similar results are also observed in the WebKB experiments. In recent work, we also study the joint inference task (Wang et al. 2014) on the 1M-facts subsets of the NELL KB, using recursive first-order theories. Comparing ProPPR to MC-SAT (MLN) in our large-scale inference experiments on the NELL datasets (Wang et al. 2014), we also notice the significant improvement in the inference time, when varying the amount the entities in the database.

Structure learning

ProPPR's structure learning scheme is based on a meta-interpretive program that resembles Metagol (Muggleton and Lin 2013). The idea is that: using an abductive second-order logic, one can construct the hypothesis space for first-order theories, and thus structure learning can be relaxed to parameter learning. In our preliminary experiments, we find that in a KB completion task using Geoff Hinton's kinship dataset, neither FOIL nor Alchemy's MLN structure learning method outperform our simple baseline, and our iterated strutural gradient approach further improve the baseline. We then further validate the result on additional datasets, including a Alyawarra kinship dataset (Denham 1973), a UMLS dataset (McCray et al. 2001; McCray 2003), and two NELL subsets. These additional experimental results show that our proposed method consistently outperforms Alchemy's MLN on various problems.

References

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

Backstrom, L., and Leskovec, J. 2011. Supervised random walks: predicting and recommending links in social networks. In *WSDM*.

Chakrabarti, S. 2007. Dynamic personalized PageRank in entity-relation graphs. In *WWW*.

Cussens, J. 2001. Parameter estimation in stochastic logic programs. *Machine Learning* 44(3):245–271.

Denham, W. 1973. The detection of patterns in alyawarra nonverbal behavior. *PhD thesis, University of Washington*.

McCray, A. T.; Burgun, A.; Bodenreider, O.; et al. 2001. Aggregating umls semantic types for reducing conceptual complexity. *Studies in health technology and informatics* (1):216–220.

McCray, A. T. 2003. An upper-level ontology for the biomedical domain. *Comparative and Functional Genomics* 4(1):80–84.

Muggleton, S., and Lin, D. 2013. Meta-interpretive learning of higher-order dyadic datalog: Predicate invention revisited. In *IJCAI*.

Page, L.; Brin, S.; Motwani, R.; and Winograd, T. 1998. The PageRank citation ranking: Bringing order to the web. In *TR, Computer Science department, Stanford*.

Wang, W. Y.; Mazaitis, K.; Lao, N.; Mitchell, T.; and Cohen, W. W. 2014. Efficient inference and learning in a large knowledge base: Reasoning with extracted information using a locally groundable first-order probabilistic logic. *arXiv preprint arXiv:1404.3301*.

Wang, W. Y.; Mazaitis, K.; and Cohen, W. W. 2013. Programming with personalized pagerank: a locally groundable first-order probabilistic logic. In *CIKM*.