Knowledge Intensive Learning: Directed vs Undirected SRL Models

Sriraam Natarajan NATARASR@BIOSTAT.WISC.EDU

Prasad Tadepalli* TADEPALL@CS.ORST.EDU

Gautam Kunapuli KUNAPG@BIOSTAT.WISC.EDU

Jude Shavlik SHAVLIK@CS.WISC.EDU

University of Wisconsin Madison USA

* Oregon State University USA

1. Introduction

The principal attraction of statistical relational models is that they are more succinct than their propositional counterparts, leading to easier specification of their structure by the domain experts and faster learning of their parameters. However, different proposed models are good at expressing different kinds of knowledge, making it difficult to compare their empirical performance. For example, Markov Logic Networks (MLNs)[1] express the knowledge as a set of weighted formulas, while the models based on directed graphs such as PRMs [3] and BLPs [7] use variablized nodes and edges with shared parameters. While PRMs [3] use aggregators max, min, avg etc to combine the influences due to several parents, other formalisms such as BLPs [7] and RBNs[5] use rules such as Noisy-OR to combine distributions.

The goal in this work is to present work in progress of comparing two kinds of models in their ability to use prior knowledge: directed models with combining rules and undirected models based on weighted logics. More precisely, we consider the following questions about different formalisms:

1. What kind of prior knowledge is most suitable for different models?
2. Given the same structural prior knowledge, how do different models differ in learning speed and generalization?
3. Which representation allows the experts to express their knowledge more easily for a given performance?
4. How does each system's performance vary with increased fine tuning of its knowledge and its representation?

Answering these questions in their full generality is a long-term goal. We naturally expect different models and systems to have different spheres of applicability and dominance. In this paper, we compare learning algorithms for first-order conditional influence (FOCI) statements [8] to the Alchemy system[1] for MLNs for answering these questions. We begin to answer the first question empirically in two domains by starting with very simple prior knowledge. We compare the performance of the two models in the context of a small number of equivalent sets of rules. We show that the directed models perform better than undirected models when there are a small number of appropriately parameterized influence rules. On the other hand, we expect the undirected models to dominate when there are a large number of weakly predictive rules.

The second question demands a deeper study of how to express equivalent pieces of prior knowledge in the two kinds of models.The final two questions seek to address the issues of ease of encoding of knowledge by human experts and the sensitivity of the model to changes in the knowledge and its representation. Answering these questions requires a user study and a more thorough evaluation than what we offer in this paper.

2. Experimental Setup

In this paper, we compare the undirected models represented by MLNs with parametric directed models represented by FOCI statements. FOCI statements specify directed graphical models with shared parameters and employ a variety of combining rules to combine the target conditional distributions. They are similar to a number of different representations including PRMs, BLPs, and RBNs that are lifted versions of Bayesian networks.

We compare the two systems in two domains: *UW* dataset and *Cora* dataset. For UW-dataset, the goal is to predict if a student was advised by a professor. The database consists of 278 faculty members and students. We used 2 rules: (1) If a student has co-authored publications with a professor, the student is likely to be advised by the professor. (2) If a student has been a TA for a course that a Professor teaches, the student is likely to be advised by the Professor. These rules when translated to the FOCI syntax [8] are:

```
CR{
  If {student(S), professor(P), course(C)}
   then taughtBy(P,C,Q), ta(S,C,Q)
```

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

```
    Qinf (Mean) advisedBy(S,P).
  If {student(S), professor(P)}
   then publication(P,W), publication(S,W)
    Qinf (Mean) advisedBy(S,P).}
```

Associated with each rule is a conditional probability distribution (CPT) that predicts the target given the influencing attributes. The predicates inside the *If* statements serve as conditions in which the rules are fired. The distributions due to the different instantiations of the same rule (multiple publications or multiple courses) are combined using the *Mean* combining rule (shown as (Mean) above). The different rules are combined using either Noisy-Or or Weighted Mean combining rule (shown as CR). The rules used for MLNs are:

`w1: student(S) ∧ professor(P) ∧ course(C) ∧ taughtBy(P,C,Q) ∧ ta(S,C,Q) :- advisedBy(S,P).`

`w2: student(S) ∧ professor(P) ∧ publication(P,W) ∧ publication(S,W) :- advisedBy(S,P).`

The other domain we used is the `citeseer` domain, where the goal is to predict if 2 citations refer to the same one. The data consisted of 4300 pairs of similar publications. We selected 500 of them at random. Each pair was labeled as positive with a probability of 0.7. Similar to the other dataset, we used 2 rules: (1) If two publications have similar title and same venue, they are likely to refer to the same citation. (2) The second rule is the transitive rule on the same citation. Converting them to our syntax,

```
CR{
 If {pub(P1), pub(P2)} then
   similarTitle(P1,P2,T1,T2), sameVenue(P1,P2)
    Qinf (Mean) sameBib(P1,P2).
 If {pub(P1), pub(P2), pub(P3)} then
     sameBib(P1,P3), sameBib(P3,P2)
      Qinf (Mean) sameBib(P3,P2).
}
```

Similar to the earlier one, mean is used for multiple instantiations of the same rule while weighted mean or Noisy-Or is used for combining different rules. The same rules when translated to MLNs will be

`w1: similarTitle(P1,P2,T1,T2) ∧ sameVenue(P1,P2) :- sameBib(P1,P2).`

`w2: sameBib(P1,P3) ∧ sameBib(P3,P2) :- sameBib(P1,P2).`

Our learning algorithms would learn the parameters of the CPTs and the weights of the weighted mean combining rule. Alchemy was used to learn the weights of the two clauses. We used 5-fold cross-validation where the parameters (resp. the weights) were learned using 4 folds and used to obtain a distribution over the target in the test-fold. We measured the average likelihood of the target over the test folds.

3. Results and Analysis

Algorithm	UW	Citeseer
CR-Learner	0.74	0.70
MLN-2	0.5	0.34
MLN-N	0.52	0.40

Table 1. Results of the learning algorithms. CR-Learner: Gradient-descent algorithm that uses combining rules. MLN-2: Alchemy with 2 clauses. MLN-N: Alchemy that uses clauses for all possible combinations of values for the predicates.

We had earlier derived Gradient-Descent and EM based algorithms for both the weighted mean as well as the Noisy-OR combining rule[8]. We present the results in Table 1 for the algorithm that optimizes the mean-squared error of the gradient (CR-Learner in the table) along with the results of MLN weight learning (MLN-2 in the table). For the MLN weight learning algorithm, we used the discriminative learning option (*-d*) and the rescaled conjugate gradient descent (*-dCG*) learning algorithm. To our understanding, we have used the state-of-the art discriminative weight learning algorithms for MLNs. We used the MCSAT algorithm for obtaining the distribution in the test fold. We also experimented with different priors and different settings of the parameters and present the best results of Alchemy runs.

The results clearly demonstrate that in both domains, likelihood of MLN-2 is not comparable to that of the CR-Learner. While the CPTs of each rule has 8 entries(2 values corresponding to each predicate in the influents and resultant), a single weight for each clause is insufficient to capture the CPT of a directed model. Hence, we included clauses for each of the combination of the truth values of the predicates of the antecedent of each rule. This is to say that if a rule's set of influents contains N literals, we consider using each literal as is or negated, producing 2^N rules. The results are presented as MLN-N in the table. Though the likelihood for MLN-N increases marginally, it is still not comparable with that of the directed models. We tried different number of clauses, different versions of the learning algorithms and different parameter settings but the results were not too different.

We speculate a few reasons for this. First, it appears that MLNs do not perform very well in the presence of a small number of rules. The directed models have more free parameters per rule and can learn a better model. We included 7 more rules provided in the UW dataset that used the *advisedBy* predicate and the performance improved (the average likelihood was close to 0.57). We also evaluated using most of the clauses

(about 55 of them) in the UW-dataset where we excluded the clauses that used existential quantifiers and the ones that used the predicates such as *samePerson*, *sameCourse*, etc. for efficiency. This resulted in a much better performance where the average likelihood was 0.67. This suggests that the performance of MLNs is highly sensitive to the number and form of the rules. Also, for MLNs, it appears necessary to use knowledge engineering to learn the target concept. It seems to be an iterative process (at least in our experience) where the domain expert begins with a certain number of clauses, learns weights, evaluates them, adds more clauses if needed etc. While this is unquestionably better than developing domain-specific solutions, it still requires a significant amount of effort from a domain expert to elucidate all the rules in the model.

Secondly, it has been shown in [6] that the weight learning algorithms have to be modified in order to capture a coherent distribution. More precisely, [6] states and proves that when the weight of a unit clause is changed, the weights of the other clauses where the predicate is present must also be adjusted by a certain factor. Although they argue this case for generative learning, we suspect that this could be the case with discriminative learning as well. We are currently pursuing research in this direction.

Thirdly, we observe that Alchemy drives the weights of most clauses towards zero[1]. This is a very good solution when the clauses are being automatically learned from data (a.k.a. structure learning). This has been demonstrated by Huynh and Mooney [4] for discriminative learning where they use L_1 regularization that uses a laplacian prior with zero mean on each weight. But when provided with minimal number of rules from a domain expert, the weights should not be driven to zero. Alchemy weight learning algorithms use a gaussian prior with zero mean (L_2 regularization), but still appear to drive weights to zero. We are currently experimenting with different priors for the weights.

Yet another issue could be the fact that Alchemy uses MAP inference and uses the counts in MAP state to approximate the expected counts for gradient computation. It is not clear if this is an issue in our domains, since the number of clauses is very small. But, on the other hand, the number of objects (citations, courses, students, etc.) is high and this could lead to a very large ground network. Hence, the approximation may not be reflective of the expected counts. It will be interesting to replace the inner loop of learning with exact inference.

Finally, the problem of representing arbitrary distributions using a minimal MLN remains an open problem. Use of complex combining rules such as Noisy-Or makes the problem of capturing arbitrary distributions more challenging. In addition, the conditions can be factored out of the CPT and made separate like in logical Bayesian networks [2] and as shown earlier using our abstract syntax. This will greatly reduce the number of combinations in the CPT. For the current implementation of Alchemy, it is also necessary to convert each function to n predicates, where n is the range of the function. This will lead to an exponential blow up in the number of clauses.

4. Conclusions

Our objective in this work is to motivate a systematic study of different SRL models to understand their strengths and weaknessses with the long-term goal of developing a variety of models that can capture, integrate, and refine prior knowledge effectively. Our initial experiments show that with small number of rules, the directed models have a better performance over MLNs as they learn more free parameters for the same set of rules. There have been several fast inference algorithms proposed based on lifted methods, lazy methods, preprocessing methods etc that can improve the quality of MAP estimates and can be integrated to improve learning quality. The other questions raised in the introduction including the ability of human users to express their knowledge succinctly in a given formalism, the efficiency of learning with different kinds of prior knowledge, and the sensitivity of learning to the representation of prior knowledge also deserve serious study.

References

[1] P. Domingos and D. Lowd. *Markov Logic: An Interface Layer for AI.* Morgan and Claypool, 2008.

[2] D. Fierens, H. Blockeel, M. Bruynooghe, and J. Ramon. Logical bayesian networks and their relation to other probabilistic logical models. In *ILP*, 2005.

[3] L. Getoor, N. Friedman, D. Koller, and A. Pfeffer. Learning probabilistic relational models. *Relational Data Mining, S. Dzeroski and N. Lavrac, Eds.*, 2001.

[4] T. N. Huynh and R. J. Mooney. Discriminative structure and parameter learning for markov logic networks. In *ICML*, 2008.

[5] M. Jaeger. Relational Bayesian networks. In *Proceedings of UAI-97*, 1997.

[6] D. Jain, B. Kirchlechner, and M. Beetz. Extending markov logic to model probability distributions in relational domains. In *KI*, 2007.

[7] K. Kersting and L. De Raedt. Bayesian logic programs. In *ILP*, 2000.

[8] S. Natarajan, P. Tadepalli, T. G Dietterich, and A. Fern. Learning first-order probabilistic models with combining rules. *Special Issue on Probabilistic Relational Learning, Annals of Mathematics and AI*, 2009.