**From:**"Guest Editors of ILP 2014" <em@editorialmanager.com>  
**To:**"Zhensong Qian" <zqian@sfu.ca>  
**Sent:**Tuesday, February 17, 2015 4:50:17 AM  
**Subject:**Decision on your manuscript #MACH-D-14-00294

CC: d.margin@comcast.net, jesse.davis@cs.kuleuven.be, jan.ramon@cs.kuleuven.be

Dear Mr. Zhensong Qian,

The reports from the reviewers of your manuscript, "Fast Learning of Relational Dependency Networks", which you submitted to Machine Learning, have now been received.

Based on the advice received, your manuscript requires major revisions in order to be acceptable for publication.  It is the policy of Machine Learning not to accept manuscripts requiring major revisions.  Instead, authors choosing to revise their manuscript according to the reviewers' comments must resubmit a revised manuscript, which usually will be reassessed by the same action editor.

When preparing your revised manuscript, you should carefully consider the reviewers' comments, and submit a point wise list of responses to the comments.  Your list of responses should be uploaded as a separate file in addition to your revised manuscript.

Among other points, we would encourage you to address the following:

1. Improving some of the motivation for tackling the problem in this manner  
2. Discussing the implications of the learning bias  
3. Improving the empirical evaluation section along the lines suggested by the reviewers  
4. Cleaning up some of the technical details

Finally, as this is a special issue we would appreciate receiving your revisions within 12 weeks. Please email us and confirm if this time frame seems feasible or not.

Please make sure to submit your editable source files (i. e. Word, TeX).

In order to submit your revised manuscript electronically, please access the following web site:

http://mach.edmgr.com/

Your username is: zqian@sfu.ca

To access your password, please complete the following steps:

   1.        Open the URL:  http://mach.edmgr.com/  
   2.        Click the LOGIN button on the banner.  
   3.        Click "Send Username/Password".    
   4.        Complete the required information (First Name, Last Name, Email Address).   
   5.        Click "Send Username and Password".

Your Password will be sent to you by email.

Please click "Author Login" to submit your revision.

Thank you.

Best regards,

Jesse Davis and Jan Ramon  
Guest Editors of ILP 2014  
Guest Editor  
Machine Learning

COMMENTS FOR THE AUTHOR:

Reviewer #1: This paper considers the problem of learning Relational Dependency Networks. To this effect, the authors propose a three step process of first learning a Bayes net, then converting it into a dependency and finally, lifting this to a RDN. The claim is that this can scale to very large amounts of data. The experimental results compared to a recent boosting algorithm backs up this claim.

The idea in the paper is quite simple and the paper is written in a  clear manner to explain "what" happens in the algorithm. So from that perspective the paper is pretty well written. The experimental results are quite compelling (though I have some questions) to back up the key claims in the paper. Things I like about the paper are the simplicity of the idea, its relatively clear explanation, nice related work section that discusses the different methods quite diligently and finally compelling experimental results.

I do have some key issues with the paper that need to clarified/fixed before fully accepting the paper.

My key issue with the paper is the motivation itself. I am quite baffled with the claim that learning Bayes net is somehow easier than learning a dependency network. More pertinently learning Bayes net can scale somehow more than a DN. We all know that learning a full BN is NP-Complete (Chickering 96). In such cases, claiming that learning a BN with acyclicity constraints is somehow easier than learning several local models (boosted or not) is quite unintuitive. While experimental results show some idea, it is clear that the issue is with the number of predicates. If the number of variables increases, clearly the number of parameters increases as well and clearly this method cannot scale.

The second (and probably more confusing issue) for me is that why learn a RDN when you have a chance of learning a RBN. RBNs are fairly better interpretable and much tighter in semantics. Not much work has been done in SRL for learning directed models effectively. If your method can learn a BN successfully and in larger scale why bother with converting it to an approximate model instead of lifting it directly? This motivation is unclear and confusing to me. So this process and the lack of motivation is quite confusing to me.

What is the learning algorithm for BNs? This needs more explanation. What are the properties of the data that will make this learning effective? Why is a NP-complete problem an easy one for you? Is this assuming independence of variables or is it the fact that you consider a small number of predicates? This needs to be better explained for things to be clear.

Which brings me to the experimental setting. If the data sets used a smaller number of predicates to learn from, I can easily see why boosting will not perform as well. But the numbers reported in this work are well-below the ones reported in the prior work that the authors have cited. Why not include another part that shows the scale with the number of predicates?

In your learning of the parameters, it is unclear how the probabilities are propagated to newly introduced bi-directional links. For existing links, I can see a weighted count being the new potential function. It appears that the Gibbs sampling comes in when you are computing the fractions (feature functions), but this is not explained quite clearly. This explanation must be improved. Back to the original question, if the original link is B->A in the BN, when you create a A->B link, how can you simply use the parameters of the original link? This needs more explanation and is unclear. Simply stating that this is same as moralizing does not seem sufficient in the case of bi-directional case. So please improve the explanation of this part. Also please do explain if the equation on Page 8, lines 12/13 are the ones obtained by Gibbs sampling.

Another issue seems to be inventing new names. Why call something as Gibbs conditional probabilities? What does this even mean? These are local distributions over a particular variable and you perform Gibbs sampling in this space because these local distributions may not be consistent. This terminology of Gibbs conditional probabilities only adds to more confusion and does not clarify anything in particular.

The experimental setup needs to be better explained. The results are explained clearly. I am not sure the claim that previous methods work for small data sets is quite correct. if you mean small in terms of the number of instances, then a better citation to the original papers may be needed. I checked the original paper and they handle the 0.1M data set size that you mention. So how is this claim justified? Their AUC-PR values seem much higher in the original paper. Can you speculate why this difference? While the claim of interpretability of a set of regression trees is a valid criticism and in fact a key reason why boosting must not be the first choice for several problems, I am not sure I agree with the idea about extending learning to more than two values. We all know that many of these n-class learning problems can be seen as n-binary learning problems and so this does not seem a big issue.

Over all, I think the paper has promise in terms of the idea. It lacks motivation, details on when things work and certainly better experimental justification.

Reviewer #2: The paper shows how to learn relational dependency networks  
via Bayesian networks. The experimental comparison compares  
to state of the art and indicates competitive performance.

Overall, I like the paper. However, in its current form,   
there are some downsides:

- no intuition provided  
- unclear learning bias  
- a little bit sloppy on some technical details

The paper is a little bit too loose about some technical details.   
For instance, on page 2, you argue that dependency networks can   
derive the joint distribution using Gibbs sampling. This, however,   
is only true if you run an unordered Gibbs sampler, see

Yoshua Bengio, Eric Laufer, Guillaume Alain, Jason Yosinski:   
Deep Generative Stochastic Networks Trainable by Backprop.   
ICML 2014: 226-234

The two references you give [9,17] also touch upon this in that  
they speak up pseudo Gibbs sampler. This sheets also some light  
on Theorem 1. First if all, we may not need the proposed refinement  
when sticking to an unordered Gibbs sample (see Bengio et al.).   
Nevertheless, it is nice that since we start with a BN, we can  
construct a consistent RDN.

On page 2, you speak of functors. Technically speaking a functor is not a predicate as claimed.   
Functors can only be used to form compound terms and not atoms.   
Moreover, as far as I understand your notation, you are actually   
not allowing for compound terms. Hence, I would suggest to just call   
this a predicate and subsequently of atoms. That is atoms   
denote parameterized RVs. On page 3, you say that one just has   
to ground the parents and the child. While this is technically true,   
the success of relational dependency networks critically depends on   
using aggregation functions (min, max, count, or any other „function"   
defined in the background knowledge). With this, however, one cannot   
say anymore that we simply have to ground parents and children.

More interestingly is the idea of learning a relational DN from  
a (relational) BN. For the propositional case, there seems to be not  
major benefits of doing so. Actually, the opposite is more appropriate,  
see also

G. Hulten, D.M. Chickering, D. Heckerman. Learning Bayesian Networks from Dependency Networks: A Preliminary Study. In Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics, Key West, FL, January 2003,

which should be cited. Hence, the authors should discuss this situation  
since it is covered by RDN\_Bayes. Generally, the authors do  
not discuss the intuition underlying their approach. This, however,  
is critical to understand the conversion to RDNs starting on page 5   
and consequently the experimental results. To be more precise, why do  
we consider the expected log-conditional probability? A relational  
dependency network can in principle have different parameters per   
ground atom. The proposed method, however, implicitly couples all  
ground atoms of a predicate (parameterized RV). Please note, I am not  
saying that this is a bad idea, but it is not clear to me whether this  
learning bias is always what we want; a discussion is in place.   
This is related to the another major downside: the learning of the  
Bayesian structure is not explained at all, instead it is referred  
to [23]. The authors should extend the current paper by a brief discussion  
of the learn-and-join algorithm, in particular w.r.t. to the „issue"  
raised above. As far as I read [23], only most general atoms are  
considered, i.e., all arguments are variables.

Why is this important? Well the baseline approaches consider  
the considerably large search space where arbitrary atoms (arguments  
can be constants) are considered. In turn, it is not clear whether  
the novel learning approach gives the performance gain or the implicitly  
encoded learning bias (no constants in atoms). This makes learning  
considerably faster and provides a regularization, which may lead  
to better performance. At least this has to be discussed in detail.   
The movieLens experiments seem to support that it is mainly the  
implicitly regularization since with large dataset size the boosting   
methods start to outperform the proposed RDN\_Bayes in terms of CLL.  
Please specify exactly the predicates you have used for the AUC comparison.  
Looks like Table 4 is presenting AUC averaged over different predicates.  
Also, do the running times include the time spend on learning the Bayes  
structure? Although I guess they are neglect able.   
The experimental results should also include the performance  
of the Bayes structure to see what the conversion to an RDN is   
giving in terms of performance gain.

Finally, please note that boosting is not creating discriminative   
models. They are just estimating RDNs, a joint model using the   
pseudo-loglikelhood factorization. And, you do not have to analyze an  
ensemble of regression trees. They can always collapsed  
after training is finished into a single tree if you like.

Nevertheless, the contributions are indeed interesting. The most important  
question is about the learning bias. That is, the authors should provide  
some intuition about what makes the learning better and faster.   
The additional number of predicates used seems not to provide   
a complete answer, at least the Bayes structure is not factored in  
at all in this answer.

Reviewer #3: This paper presents a method for quickly learning relational dependency networks (RDNs) via first learning Bayesian networks (BNs).

The rationale for "compiling" to RDNs is to make use of their "advantages of inference" (page 1, line 30). However, these advantages are not apparent in the context of the empirical evaluation, which predicts the probability of a target ground predicate given all other ground predicates as evidence (page 11, line 16). In this setting, wouldn't the inference in BNs be easy too? To demonstrate the value of the paper's proposed algorithm, there should be experiments comparing the performance of the learned BNs to that of the RDNs (that are derived from the BNs). Only if the RDNs perform better, will the overhead of "compiling" them be justified.

Questions

\* Page 5, Algorithm 1, line 8  
By restricting F to containing only true relationships, is it true that that you are confining features to vary over values of attributes (e.g., gender(A)=M, gender(B)=M)? Why do you need to make such a restriction? And what happens if you don't?

\* Page 6, Definition 1.  
The second summation ranges over all values of u\_pa. However, when u is a child of T, u\_pa contains T, and its value must be tied to T\*=t. Could the authors make this condition clearer? (Table 1 illustrates this situation clearly.)

\* Page 12, Table 4.  
Why is RDN\_Bayes doing so poorly relative to RDN\_Boost and MLN\_Boost on the Mutagenesis dataset? What about Mutagenesis that is tripping up RDN\_Bayes?