Tractable Learning of Liftable Markov Logic Networks

Jan Van Haaren JAN.VANHAAREN@CS.KULEUVEN.BE
Guy Van den Broeck GUY.VANDENBROECK@CS.KULEUVEN.BE
Wannes Meert WANNES.MEERT@CS.KULEUVEN.BE
Jesse Davis JESSE.DAVIS@CS.KULEUVEN.BE
Department of Computer Science, KU Leuven, Belgium

Abstract

Markov logic networks (MLNs) are a popular statistical relational learning formalism that combine Markov networks with first-order logic. Unfortunately, inference and maximum-likelihood learning with MLNs is highly intractable. For inference, this problem is addressed by lifted algorithms, which speed up inference by exploiting symmetries. State-of-the-art lifted algorithms give tractability guarantees for broad classes of MLNs and inference tasks. For learning, we showed in recent work how to use lifted inference techniques for efficient maximum-likelihood parameter learning. In this paper, we propose the first lifted structure learning algorithm that guarantees that the learned MLNs are liftable, and thus tractable for certain queries. Our work is among the first to apply the tractable learning paradigm to statistical relational models. Moreover, it is the first structure learning algorithm that exactly optimizes the likelihood of the MLN. An empirical evaluation on three real-world datasets shows that our algorithm learns accurate models, both in terms of likelihood and prediction quality. Furthermore, our tractable learner outperforms intractable models on prediction tasks suggesting that liftable models are a powerful hypothesis space, which may be sufficient for many standard learning problems.

1. Introduction

Statistical relational learning (SRL) (Getoor & Taskar, 2007) and probabilistic logic learning (De Raedt et al., 2008) seek to develop representations that combine the benefits of *probabilistic* models, such as Markov or Bayesian networks, with those of *relational* representations, such as first-order logic. Markov logic networks (MLNs), which combine first-order logic with Markov networks, are one of the most widely used SRL formalisms (Richardson & Domingos, 2006). MLNs attach a weight to each first-order logic formula in a theory. Given a set of objects, they compactly specify how to construct a very large propositional Markov network. When learning MLNs from data, two tasks are considered. The *parameter learning* task is to learn the weights associated with each formula in a given theory. This is an analogous problem to that of learning feature weights in a log-linear propositional Markov network. For the *structure learning* task, one also learns the first-order logic formulas. In both cases, the data consist of a set of relational databases.

Markov logic networks pose a great challenge for inference and learning: using classical algorithms, these tasks reduce to inference and learning in densely connected Markov networks with millions of random variables. The intractability of reasoning with SRL models motivated a new class of *lifted inference* algorithms (Poole, 2003; Kersting, 2012), which exploit the abundant *symmetries* in relational representations to speed up probabilistic inference. For large classes of liftable models and queries, these algorithms perform efficient inference without ever grounding to a propositional Markov network (Jaeger & Van den Broeck, 2012). Whereas lifted inference deals with the intractability of reasoning, the intractability of *learning* is what motivated us in Van den Broeck et al. (2013), where we showed that lifted inference techniques improve *parameter learning* of liftable models, in terms of learning time and quality of the learned model.

In this paper, we seek to efficiently learn models that themselves are tractable for inference, in the sense that they permit lifted inference. We perform *generative* learning, where the objective is to learn a model that maximizes the probability of observing the data. We focus on techniques from the *exact* lifted inference literature, where liftable model and query classes were defined. Moreover, this will

Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP volume 32.

allow us to compare the exact likelihoods of the learned MLNs, which is the natural evaluation measure for generative learning (Darwiche, 2009; Koller & Friedman, 2009; Murphy, 2012).

Our first contribution is a *lifted structure learning* algorithm that learns liftable MLN theories. It uses the lifted parameter learning algorithm of Van den Broeck et al. (2013) as a subroutine, and therefore also optimizes the exact training set likelihood. This contrasts with existing MLN structure learners which resort to optimizing pseudolikelihood. Moreover, the learned structures are guaranteed to support certain types of queries efficiently, including, for example, conditional probability queries with bounded rank. Our work thus follows in a long tradition of tractable structure learning algorithms for probabilistic graphical models (e.g., Chechetka & Guestrin 2007), and is among the first *tractable learning* algorithms for *statistical relational* representations.

Our second contribution is an *extensive empirical evaluation* of lifted structure learning on three standard real-world SRL datasets. When learning tractable structures, our lifted learning algorithm outperforms existing learners in terms of likelihood, but also in terms of conditional likelihood and area under the precision-recall curve on prediction tasks. We even find that our tractable structure learner outperforms off-the-shelf intractable learners on prediction tasks, suggesting that liftable models are a powerful hypothesis space, which is sufficient for many standard learning problems. It shows that liftability and symmetry is an effective regularization technique, and that the support for lifted maximum-likelihood parameter learning outweighs the reduced expressivity of liftable structures.

2. Background

We build on statistical relational representations, tasks, and algorithms, which we now briefly review.

2.1. Markov Logic

Markov networks are undirected probabilistic graphical models that represent a joint probability distribution over a set of random variables $X_1, \ldots X_n$ (Della Pietra et al., 1997). Each clique of variables $\mathbf{X}_k$ in the graph has an associated potential function $\phi_k(\mathbf{X}_k)$. The probability of a possible world $\mathbf{x}$ represented by a Markov network is $\Pr(\mathbf{x}) = \frac{1}{Z}\prod_k \phi_k(\mathbf{x}_k)$, where $\mathbf{x}_k$ is the state of the kth clique (i.e., the state of the variables that appear in that clique), and Z is a normalization constant. Markov networks are often conveniently represented as *log-linear models*, where clique potentials are replaced by an exponentiated weighted sum of features of the state: $\Pr(\mathbf{x}) = \frac{1}{Z}\exp\left(\sum_i w_i f_i(\mathbf{x})\right)$. A feature f_i may be any real-valued function of the state.

Markov logic networks (MLNs) (Richardson & Domingos, 2006) combine Markov networks with first-order logic. MLNs soften logic by associating a weight with each formula. Worlds that violate formulas become less likely, but not impossible. Formally, an MLN is a set of pairs, (F_i, w_i), where F_i is a first-order formula and $w_i \in \mathbb{R}$. As w_i increases, so does the strength of the constraint F_i imposed on the world.

MLNs provide a template for constructing Markov networks. When given a finite set of constants (the domain), the MLN formulas define a Markov network. Nodes in the network, representing random variables, are the ground instances of the atoms in the formulas. Edges connect literals that appear in the same ground instance of a formula. An MLN induces the following probability distribution over relational databases db:

$$\Pr(db) = \frac{1}{Z}\exp\left(\sum_j^{|F|} w_i n_i(db)\right) \tag{1}$$

where F is the set of formulas in the MLN, w_i is the weight of the i^{th} formula, and $n_i(db)$ is the number of true groundings of formula F_i in databases db.

2.2. Lifted Inference and Tractability

Statistical relational languages such as Markov logic have motivated a new class of *lifted inference* algorithms (Poole, 2003). SRL models with large domains lead to very large graphical models, causing inference to become intractable. Lifted algorithms mitigate this cost, by exploiting the high-level structure and symmetries of the first-order logic formulas to speed up inference (Kersting, 2012). Surprisingly, they perform tractable inference even in the absence of conditional independencies (Niepert & Van den Broeck, 2014). They scale up to millions of random variables, in cases where classical algorithms require exponential time.

Recently, efforts were made to understand lifted inference at a theoretical level, and to delineate the classes of MLNs and inference tasks[1] for which lifted inference is tractable. The intuition that lifted algorithms should efficiently deal with large domains is formalized by the notion of domain-lifted inference (Van den Broeck, 2011). An inference algorithm is domain-lifted when it runs in time polynomial in the domain size, that is, the number of objects in the world. Note that domain-lifted algorithms can be exponential in other parameters, such as the number of formulas. Based

[1]Note that tractability of probabilistic models is always w.r.t. a class of inference tasks. For example, polytrees, arithmetic circuits, and sum-product networks are generally considered to be tractable, but (partial) MAP inference in them is still NP-complete (Park, 2002).

on this notion of tractability, several classes of MLNs and inference tasks were shown to support lifted inference.

Tractable *classes of MLNs* include monadic MLNs, where all predicates have a single argument (Jaeger & Van den Broeck, 2012), and two-variable MLNs, where all features (formulas) have at most two logical variables (Van den Broeck, 2011; Taghipour et al., 2013). Any combination of universal and existential quantification is liftable (Van den Broeck et al., 2014). This list of tractable classes is far from exhaustive, and many more complex MLNs are liftable, including ones with more than two variables. The MLNs considered in this paper are generally not in the classes above, yet they are still individually liftable. Tractable *classes of inference tasks* include partition functions, single marginal probabilities (Van den Broeck, 2011), and expected counts of MLN formulas (Van den Broeck et al., 2013). Conditional probability queries are liftable given unary evidence atoms (Van den Broeck & Davis, 2012; Bui et al., 2012) and binary evidence atoms of bounded Boolean rank (Van den Broeck & Darwiche, 2013).

2.3. Weight Learning

The weight learning task for MLNs (Singla & Domingos, 2005; Richardson & Domingos, 2006; Lowd & Domingos, 2007; Huynh & Mooney, 2009) uses data to automatically learn the weight associated with each feature by optimizing a given objective function.

Generative learning typically optimizes the log-likelihood. For MLNs, the log-likelihood is a convex function of the weights and learning can be solved via convex optimization. The gradient for each feature F_j in the MLN can be computed by simply taking the difference between the number of true groundings of F_j in the data and its expectation according to the current model. Thus, each iteration of weight learning must perform inference on the current model to compute the expectations. This is often computationally infeasible and hence the default generative weight learning approach for MLNs is to optimize the pseudo-likelihood (Besag, 1975), which is an approximation of the likelihood, but is more efficient to compute.

Lifted Weight Learning (LWL) We have recently shown that it is possible to use exact lifted inference to learn maximum likelihood weights (Van den Broeck et al., 2013). This approach works by compiling an MLN theory to a FO d-DNNF circuit (Van den Broeck et al., 2011), which is able to efficiently compute both the partition function and the expected number of true groundings of each formula. Moreover, this can be done in time polynomial in the size of the relational database whereas traditional inference techniques are in general exponential in the size of the training databases. The benefit of FO d-DNNF compilation for weight learning is that compilation only needs to be performed once per MLN structure. In each iteration of lifted weight learning, the compiled circuit can be reused with updated weights to compute a new likelihood and its gradient. Previous work had explored the use of lifted belief propagation, which is an approximate lifted inference algorithm, for both generative (Jaimovich et al., 2007) and discriminative (Ahmadi et al., 2012) weight learning.

2.4. Structure Learning

The structure learning task is to learn both the formulas and their associated weights from data. Structure learning is an incredibly challenging problem as there is a huge number of candidate clauses and an even larger space of candidate models. Although many structure learning approaches have been proposed in recent years (Biba et al., 2008; Huynh & Mooney, 2008; Kok & Domingos, 2009; Khot et al., 2011; Dinh et al., 2011), the structure of a MLN is typically learned by greedily adding one clause at a time to the MLN. While multiple MLN structure learning approaches exist, they can be broadly be divided into two categories: top-down and bottom-up.

MSL (Kok & Domingos, 2005) is a canonical example of a top-down approach. MSL begins with a MLN that only contains the unit clauses. MSL starts by constructing all clauses of length two. It then runs a beam search to find the current best clause and adds it to the network. In each iteration, MSL constructs new candidate clauses by adding literals to the best clauses in the beam. The search iterates until no clause improves the score of the MLN. To evaluate the merit of each clause, MSL uses weighted pseudo-log-likelihood (WPLL), which is an extension of pseudo-log-likelihood that diminishes the importance of predicates with a large number of groundings by attaching a weight to each predicate (Kok & Domingos, 2005). To avoid overfitting, each clause receives a penalty term proportional to the number of literals that differ between the current clause and the initial clause.

A second category of structure learners adopts a bottom-up approach (e.g., Mihalkova & Mooney 2007; Kok & Domingos 2010), using the data to restrict the search space. BUSL (Mihalkova & Mooney, 2007) is a two-step algorithm that follows this paradigm. In the first step, it constructs a template Markov network from a ground Markov network by discovering recurring paths of true atoms. In the second step, it transforms the template Markov network into candidate clauses. It greedily iterates through the set of candidate clauses. It adds the clause to the MLN that most improves the score of the model. The search terminates when no clause improves the model's score. Again, BUSL uses WPLL to evaluate the merit of a candidate clause.

2.5. Tractable Learning

Learning tractable probabilistic models, that is, models that always permit efficient inference for certain queries is an emerging area of research. The largest body of work restricts the structure of the learned models (e.g., Chechetka & Guestrin 2007). One way to do this is to only consider models with a low tree-width (Narasimhan & Bilmes, 2004; Chechetka & Guestrin, 2007). Another body of work looks at simultaneously learning either a Bayesian network or a Markov network as well as an alternative representation (typically an arithmetic circuit) of the model that permits efficient inference. Then the model is penalized by the cost of inference, which can be calculated based on well-defined properties of the representation. By penalizing the circuit size of the model, it is possible to bias the learning algorithm towards models where efficient inference is possible (Lowd & Domingos, 2008; Lowd & Rooshenas, 2013; Gens & Domingos, 2013). Our work fits within this framework since we propose tractable structure learning towards models that allow lifted inference. This guarantees tractable inference for the types of queries identified in Section 2.2. While tractable statistical relational languages have been investigated before (Domingos & Webb, 2012), we believe our work is among the first to consider the problem of learning such tractable representations.

3. Lifted Structure Learning

This section presents an algorithm that learns liftable MLN models. The ideal way to learn a liftable model is to design a search space that only contains liftable models, such as the tractable classes in Section 2.2. However, it is difficult to design that search space, because we currently lack a full characterization of which models are liftable. While any model where each formula contains at most two distinct logical variables is always liftable, this class of models may be too restrictive. Many models that contain more expressive formulas are also liftable. This process is complicated by the fact that two formulas, considered independently, may be liftable, but combining them into a single model results in a theory that is not liftable.

We propose integrating a check into the search procedure that verifies whether each candidate theory is liftable. Our algorithm evaluates each MLN model by compiling it into a FO d-DNNF circuit (Van den Broeck et al., 2011), which guarantees domain-lifted inference. Only those models that are compilable will be evaluated. Hence, non-liftable candidate theories are discarded from the search space. Furthermore, even if a model is liftable, the circuit may be too big to fit in memory or time-consuming to evaluate. A bias can be inserted into the search to avoid liftable theories that are too big to fit in memory or too complex to be evaluated in practice. By having the FO d-DNNF circuit of the MLN available, we thus guarantee tractability. An additional benefit is that the learner can use the FO d-DNNF circuit to directly optimize the exact likelihood instead of an approximation such as pseudo-likelihood.

Algorithm 1 LIFTEDSTRUCTURELEARNING(CFS, DB)

Input:
CFS: A set of candidate MLN formulas
DB: A set of training databases

Supporting functions:
LIFTEDWEIGHTLEARNING Compute formula weights
LOGLIKELIHOOD Compute likelihood

Function:

```
T ← ∅                                   // Initialize theory
TLL ← 0                                 // Theory likelihood
while |CFS| > 0 do
    BCT ← ∅                             // Best candidate theory
    BCF ← ∅                             // Best candidate formula
    BCTLL ← 0                           // Best candidate theory likelihood
    for each CF ∈ CFS do
        CT ← T ∪ CF                     // Compose candidate theory
        WCT ← LIFTEDWEIGHTLEARNING(CT,DB)
        WCTLL ← LOGLIKELIHOOD(WCT,DB)
        if WCTLL > BCTLL then
            BCT ← WCT                   // Update best candidate theory
            BCF ← CF                    // Update best formula
            BCTLL ← WCTLL               // Update best likelihood
        end if
    end for
    T ← BCT                             // Replace theory by best candidate theory
    CFS ← CFS \ {BCF}                   // Remove best candidate formula from candidate set
end while
return T
```

Algorithm 1 outlines our Lifted Structure Learning (LSL) approach. LSL takes a set of MLN formulas and training databases as input and returns a theory of MLN formulas and their associated weights. The algorithm optimizes the training set log-likelihood by iteratively adding formulas to an initially empty theory. The initial set of candidate formulas can be constructed either by running the candidate formula construction step of an off-the-shelf structure learning algorithm or by greedily enumerating all valid formulas satisfying certain constraints. In our experimental evaluation (see Section 4), we enumerate all formulas having at most three literals and at most three distinct object variables. We only consider "connected" formulas for which a path via arguments exists between any two literals.

In each iteration, the algorithm performs three steps. First, the algorithm builds a set of candidate theories by adding

each candidate formula to a distinct copy of the current theory (line 8). Second, the algorithm learns the associated formula weights (line 9) and computes the training data log-likelihood (line 10) for each candidate theory. This step is allowed a user-specified amount of time to complete. Each candidate formula that is not compilable or whose weight cannot be learned within the allotted time, is discarded. By biasing the search process towards formulas that can easily be compiled, the final theory's liftability is ensured. Third, the algorithm replaces the current theory by the best candidate theory in terms of training data log-likelihood if that theory yields a log-likelihood improvement (line 17). The algorithm ends when no more candidate formulas are available or none of the remaining candidate theories yields a log-likelihood improvement over the current best theory.

Note that the tractable models learned by lifted structure learning typically have a *treewidth* that is at least linear in the domain size, and have almost no conditional independencies. The MLNs learned in Section 4, for example, will have a treewidth of at least 1000. The tractability of these models therefore essentially depends on the ability to perform lifted inference. This ability is witnessed in our algorithm by the availability of a FO d-DNNF circuit.

4. Empirical Evaluation

In this section, we evaluate our lifted structure learning (LSL) approach.[2] We now introduce the datasets, discuss the methodology, and address the following questions:

- Q1: How does LSL compare to off-the-shelf structure learners in terms of test-set likelihood when learning tractable models?
- Q2: How does LSL compare to off-the-shelf structure learners in terms of AUC and CLL when learning tractable models?
- Q3: How does LSL compare to off-the-shelf structure learners in terms of AUC and CLL?

4.1. Datasets

We use three real-world datasets in our empirical evaluation. The first dataset, **IMDb**,[3] comes from the IMDb.com website. The dataset contains information about attributes (e.g., gender) and relationships among actors, directors, and movies. The data is divided into five different folds. The second dataset, **UWCSE**,[3] contains information about the University of Washington CSE Department. The data contains information about students, professors and classes, and models relationships (e.g., TAs and advisor) among these entities. The data consists of five folds, each one corresponding to a different group in the CSE Department. The third dataset, **WebKB**,[4] consists of Web pages from the computer science departments of four universities (Mihalkova & Mooney, 2007). The data has information about labels of pages (e.g., student and course). There are four folds, one for each university.

4.2. Methodology

In this evaluation, we compare tractable models learned by LSL with both tractable and intractable models learned by BUSL and MSL. We learned these models as follows:

Tractable LSL models: LSL is run with all valid "connected" MLN formulas having at most three literals and three distinct logical variables as input. To enforce tractability, LSL discards any candidate theory for which the LWL subroutine fails to find weights within the allotted time limit of five minutes.

Tractable BUSL and MSL models + PLL: To enforce tractability, we found it is sufficient to restrict BUSL and MSL to learn formulas containing at most four literals and three distinct logical variables. The weights for the models are learned using the generative weight learning algorithm in the Alchemy package (Kok & Domingos, 2005), which optimizes the pseudo-likelihood.

Tractable BUSL and MSL models + LWL: To enforce tractability, we again restrict BUSL and MSL to formulas containing at most four literals and three distinct logical variables. The weights for the models are learned using LWL, which optimizes the likelihood. These algorithms thus search for MLN structures that optimize pseudo-likelihood, but then use lifted weight learning to optimize the exact likelihood.

Intractable BUSL and MSL models: BUSL and MSL are run with their default parameter settings, which allows them to learn formulas containing up to five literals and five distinct logical variables. The weights for the models are learned using the generative weight learning algorithm in the Alchemy package,[5] which optimizes the pseudo-likelihood.

[2] Our implementation is available on http://dtai.cs.kuleuven.be/wfomc as part of the WFOMC package.

[3] The IMDb and UWCSE datasets are available on http://alchemy.cs.washington.edu/data.

[4] The WebKB dataset is available on http://www.cs.cmu.edu/~webkb.

[5] The Alchemy package is available on http://alchemy.cs.washington.edu.

4.3. Research Questions

Q1: How does LSL compare to off-the-shelf structure learners in terms of test-set likelihood when learning tractable models?

The goal of this question is to investigate whether models learned by LSL yield better test-set likelihoods than tractable models learned by off-the-shelf structure learning algorithms. We compare the tractable LSL models with the tractable BUSL and MSL models to answer this question.

Table 1 reports test-set likelihoods for tractable models learned by BUSL and LSL on all three datasets. When comparing the LSL models with the BUSL+PLL models, LSL outperforms BUSL in all 14 settings. When comparing the LSL models with the BUSL+LWL models, LSL still outperforms BUSL in all 14 settings.

Table 2 reports test-set likelihoods for tractable models learned by MSL and LSL on all three datasets. When comparing the LSL models with the MSL+PLL models, LSL outperforms MSL in all 14 settings. When comparing the LSL models with the MSL+LWL models, LSL outperforms MSL in 13 of the 14 settings, doing only marginally worse than M+LWL on one fold of the UWCSE dataset.

These results show that our lifted structure learning algorithm learns considerably better tractable models in terms of test-set likelihood than the off-the-shelf structure learners. The fact that LSL performs better than the models whose weights are learned using LWL, and these models in turn perform better than the models whose weights are learned using PLL, shows that both LWL and LSL yield significant improvements.

Q2: How does LSL compare to off-the-shelf structure learners in terms of AUC and CLL when learning tractable models?

The goal of this question is to investigate whether models learned by LSL are better at answering queries than tractable models learned by off-the-shelf structure learning algorithms. We compare the tractable LSL models with the tractable BUSL and MSL models whose weights are learned using LWL to answer this question.

In each domain, we predict the marginal probabilities of each predicate given evidence about all other predicates. Since lifted inference approaches cannot efficiently handle arbitrary binary evidence (Van den Broeck & Darwiche, 2013), we use MC-SAT, which is part of the Alchemy package, to compute the probabilities. We use a burn-in of 10,000 samples and compute the probabilities with the following 100,000 samples.

We measure the area under the precision-recall curve (AUC) and the test-set conditional likelihood (CLL) for the predicate of interest. AUC is insensitive to the large number of true negatives in the datasets, whereas CLL measures the quality of the probability estimates. AUC and CLL are both skew-dependent metrics. The skew of a predicate varies both across different predicates and different databases, simply averaging AUCs and CLLs, as has commonly been done in the past, is incorrect (Boyd et al., 2012). Therefore, we report the number of wins, loses, and ties for each algorithm.

Table 3 reports the number of times LSL wins, loses, and ties compared to BUSL as well as MSL. When comparing LSL and BUSL in terms of AUC, LSL wins in 45 of the 95 settings, ties in 14 settings, and loses in 36 settings. When comparing LSL and BUSL in terms of CLL, LSL wins in 67 of the 95 settings, ties in 1 setting, and loses in 27 settings. When comparing LSL and MSL in terms of AUC, LSL wins in 49 of the 95 settings, ties in 13 settings, and loses in 33 settings. When comparing LSL and MSL in terms of CLL, LSL wins in 56 of the 95 settings, ties in 1 setting, and loses in 38 settings.

LSL is consistently leading to better models. While it achieves more wins than both BUSL and MSL for both metrics, it does particularly well at CLL. Although we used LWL to relearn the weights for the BUSL and MSL models, during structure learning these algorithms optimized WPLL, which is very similar to this inference setting and thus should be to their advantage.

Q3: How does LSL compare to off-the-shelf structure learners in terms of AUC and CLL?

The goal of this question is to investigate whether *tractable* models learned by LSL are better at answering queries than *intractable* models learned by off-the-shelf structure learning algorithms. We compare LSL models with the intractable BUSL and MSL models to answer this question. We use the same inference set up as in Q2 and again measure the area under the precision-recall curve and the test-set conditional likelihood for the predicate of interest.

Table 4 reports the number of times LSL wins, loses, and ties compared to BUSL as well as MSL. When comparing LSL and BUSL in terms of AUC, LSL wins in 57 of the 95 settings, ties in 6 settings, and loses in 32 settings. When comparing LSL and BUSL in terms of CLL, LSL wins in 71 of the 95 settings, ties in 0 settings, and loses in 24 settings. When comparing LSL and MSL in terms of AUC, LSL wins in 43 of the 95 settings, ties in 8 settings, and loses in 44 settings. When comparing LSL and MSL in terms of CLL, LSL wins in 62 of the 95 settings, ties in 0 settings, and loses in 33 settings.

Surprisingly, BUSL and MSL do not perform better at answering queries when they are no longer bound to learning

Table 1. Comparison of test-set likelihoods of tractable LSL and tractable BUSL models. LSL outperforms both the models whose weights are learned using PLL (B+PLL) and the models whose weights are learned using LWL (B+LWL) in all 14 settings.

	IMDb			**UWCSE**			**WebKB**		
	B+PLL	B+LWL	LSL	B+PLL	B+LWL	LSL	B+PLL	B+LWL	LSL
Fold 1	-548	-378	**-306**	-1,860	-1,524	**-1,447**	-858	-778	**-774**
Fold 2	-689	-390	**-309**	-594	-535	**-511**	-1,422	-1,331	**-1,306**
Fold 3	-1,157	-851	**-733**	-1,462	-1,245	**-1,167**	-717	-702	**-672**
Fold 4	-415	-285	**-224**	-2,820	-2,510	**-2,442**	-1,224	-1,052	**-1,043**
Fold 5	-413	-267	**-216**	-2,763	-2,357	**-2,227**			

Table 2. Comparison of test-set likelihoods of tractable LSL and tractable MSL models. LSL outperforms the models whose weights are learned using PLL (M+PLL) in all 14 settings and the models whose weights are learned using LWL (M+LWL) in 13 of the 14 settings.

	IMDb			**UWCSE**			**WebKB**		
	M+PLL	M+LWL	LSL	M+PLL	M+LWL	LSL	M+PLL	M+LWL	LSL
Fold 1	-831	-440	**-306**	-1,705	-1,469	**-1,447**	-868	-797	**-774**
Fold 2	-944	-477	**-309**	-574	**-509**	-511	-1,426	-1,324	**-1,306**
Fold 3	-1,576	-909	**-733**	-1,358	-1,198	**-1,167**	-711	-677	**-672**
Fold 4	-393	-315	**-224**	-2,758	-2,449	**-2,442**	-1,207	-1,054	**-1,043**
Fold 5	-388	-353	**-216**	-2,582	-2,254	**-2,227**			

Table 3. Comparison of AUC and CLL results for learning tractable models. In comparison to BUSL, LSL wins in 112 of the 190 settings, ties in 15 settings, and loses in 63 settings. In comparison to MSL, LSL wins in 105 of the 190 settings, ties in 14 settings, and loses in 71 settings.

		IMDb			**UWCSE**			**WebKB**		
		Win	Loss	Tie	Win	Loss	Tie	Win	Loss	Tie
LSL vs. BUSL	AUC	**11**	9	10	**24**	18	3	**10**	9	1
LSL vs. BUSL	CLL	**18**	11	1	**36**	9	0	**13**	7	0
LSL vs. MSL	AUC	8	**11**	**11**	**32**	11	2	9	**11**	0
LSL vs. MSL	CLL	12	**17**	1	**31**	14	0	**13**	7	0

Table 4. Comparison of AUC and CLL results. In comparison to BUSL, LSL wins in 128 of the 190 settings, ties in 6 settings, and loses in 56 settings. In comparison to MSL, LSL wins in 105 of the 190 settings, ties in 8 settings, and loses in 77 settings.

		IMDb			**UWCSE**			**WebKB**		
		Win	Loss	Tie	Win	Loss	Tie	Win	Loss	Tie
LSL vs. BUSL	AUC	**15**	10	5	**27**	17	1	**15**	5	0
LSL vs. BUSL	CLL	**24**	6	0	**31**	14	0	**16**	4	0
LSL vs. MSL	AUC	9	**14**	7	**25**	19	1	9	**11**	0
LSL vs. MSL	CLL	**18**	12	0	**31**	14	0	**13**	7	0

tractable models. Their performance remains roughly the same with BUSL winning in even fewer settings than when learning tractable models. These results show that learning longer, more complex formulas does not necessarily lead to better inference results. A possible explanation is that more complex models may fit the data better but also lead to more complicated inference tasks, which in turn leads to a decreased predictive performance.

5. Conclusion

We investigated efficiently learning models that are tractable for inference by permitting lifted inference. Specifically, we investigate generative learning, where the goal is to maximize the probability of observing the data, in the context of Markov logic networks (MLNs). We present two contributions. Our first contribution is a *lifted structure learning* algorithm that learns liftable MLN theories.

In contrast to existing MLN structure learners, which resort to optimizing pseudo-likelihood, it optimizes the exact likelihood and guarantees the learned structures to support certain types of queries efficiently. Our second contribution is an *extensive empirical evaluation* of lifted structure learning on three standard real-world SRL datasets. When learning tractable structures, our lifted learning algorithm outperforms existing learners in terms of likelihood, but also in terms of conditional likelihood and area under the precision-recall curve on prediction tasks. We even find that our tractable structure learner outperforms off-the-shelf intractable learners on prediction tasks, suggesting that liftable models are a powerful hypothesis space, which is sufficient for many standard learning problems.

Acknowledgments

Jan Van Haaren is supported by the Agency for Innovation by Science and Technology in Flanders (IWT). Guy Van den Broeck is supported by the Research Foundation-Flanders (FWO-Vlaanderen). Jesse Davis is partially supported by the research fund KU Leuven (OT/11/051), EU FP7 Marie Curie Career Integration Grant (#294068) and FWO-Vlaanderen (G.0356.12).

References

Ahmadi, B., Kersting, K., and Natarajan, S. Lifted online training of relational models with stochastic gradient methods. In *Proceedings of ECML/PKDD*, 2012.

Besag, J. Statistical Analysis of Non-Lattice Data. *The Statistician*, 24:179–195, 1975.

Biba, Marenglen, Ferilli, Stefano, and Esposito, Floriana. Discriminative Structure Learning of Markov Logic Networks. In *Proceedings of the 18th International Conference on Inductive Logic Programming*, pp. 59–76. Springer, 2008.

Boyd, Kendrick, Santos Costa, Vitor, Davis, Jesse, and Page, David. Unachievable region in precision-recall space and its effect on empirical evaluation. In *Proceedings of 29th International Conference on Machine Learning*, pp. 1–9, 2012.

Bui, Hung B, Huynh, Tuyen N, and de Salvo Braz, Rodrigo. Exact lifted inference with distinct soft evidence on every object. In *AAAI*, 2012.

Chechetka, Anton and Guestrin, Carlos. Efficient principled learning of thin junction trees. In *Advances in Neural Information Processing Systems*, pp. 273–280, 2007.

Darwiche, Adnan. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2009.

De Raedt, Luc, Frasconi, Paolo, Kersting, Kristian, and Muggleton, Stephen (eds.). *Probabilistic inductive logic programming: theory and applications*. Springer-Verlag, Berlin, Heidelberg, 2008.

Della Pietra, S., Della Pietra, V., and Lafferty, J. Inducing Features of Random Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:380–392, 1997.

Dinh, Quang-Thang, Exbrayat, Matthieu, and Vrain, Christel. Generative Structure Learning for Markov Logic Networks Based on Graph of Predicates. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pp. 1249–1254. AAAI Press, 2011.

Domingos, Pedro and Webb, W Austin. Tractable markov logic. *Proceedings of AAAI*, 2012.

Gens, Robert and Domingos, Pedro. Learning the structure of sum-product networks. In *Proceedings of The 30th International Conference on Machine Learning*, pp. 873–880, 2013.

Getoor, Lise and Taskar, Ben (eds.). *An Introduction to Statistical Relational Learning*. MIT Press, 2007.

Huynh, Tuyen and Mooney, Raymond. Discriminative Structure and Parameter Learning for Markov Logic Networks. In *Proceedings of the 25th International Conference on Machine Learning*, pp. 416–423. ACM, 2008.

Huynh, Tuyen N. and Mooney, Raymond J. Max-margin weight learning for markov logic networks. In *Proceedings of ECML/PKDD*, pp. 564–579, 2009.

Jaeger, Manfred and Van den Broeck, Guy. Liftability of probabilistic inference: Upper and lower bounds. In *Proceedings of the 2nd International Workshop on Statistical Relational AI*, 2012.

Jaimovich, Ariel, Meshi, Ofer, and Friedman, Nir. Template based inference in symmetric relational Markov random fields. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence*, pp. 191–199, 2007.

Kersting, Kristian. Lifted probabilistic inference. In *Proceedings of European Conference on Artificial Intelligence (ECAI)*, 2012.

Khot, Tushar, Natarajan, Sriraam, Kersting, Kristian, and Shavlik, Jude. Learning Markov Logic Networks via Functional Gradient Boosting. In *Proceedings of the 11th International Conference on Data Mining*, pp. 320–329. IEEE, 2011.

Kok, S. and Domingos, P. Learning the structure of Markov logic networks. In *Proceedings of the International Conference on Machine Learning*, pp. 441–448, 2005.

Kok, S. and Domingos, P. Learning Markov logic networks using structural motifs. In *Proceedings of the 27th International Conference on Machine Learning*, pp. 551–558, 2010.

Kok, Stanley and Domingos, Pedro. Learning Markov Logic Network Structure via Hypergraph Lifting. In *Proceedings of the 26th International Conference on Machine Learning*, pp. 505–512. ACM, 2009.

Koller, D. and Friedman, N. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

Lowd, Daniel and Domingos, Pedro. Efficient weight learning for Markov logic networks. In *Proceedings of the 11th Conference on the Practice of Knowledge Discovery in Databases*, pp. 200–211, 2007.

Lowd, Daniel and Domingos, Pedro. Learning arithmetic circuits. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence*, 2008.

Lowd, Daniel and Rooshenas, Amirmohammad. Learning Markov networks with arithmetic circuits. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, pp. 406–414, 2013.

Mihalkova, L. and Mooney, R. J. Bottom-Up Learning of Markov Logic Network Structure. In *Proceedings of the 24th International Conference on Machine Learning*, pp. 625–632, 2007.

Murphy, Kevin P. *Machine learning: a probabilistic perspective*. MIT Press, 2012.

Narasimhan, Mukund and Bilmes, Jeff. Pac-learning bounded tree-width graphical models. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pp. 410–417, 2004.

Niepert, Mathias and Van den Broeck, Guy. Tractability through exchangeability: A new perspective on efficient probabilistic inference. *Proceedings of AAAI*, 2014.

Park, James D. Map complexity results and approximation methods. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, pp. 388–396. Morgan Kaufmann Publishers Inc., 2002.

Poole, David. First-Order Probabilistic Inference. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 985–991, 2003.

Richardson, Matthew and Domingos, Pedro. Markov Logic Networks. *Machine Learning*, 62(1):107–136, 2006.

Singla, Parag and Domingos, Pedro. Discriminative training of markov logic networks. In *20th National Conference on Artificial Intelligence*, pp. 868–873, 2005.

Taghipour, Nima, Fierens, Daan, Van den Broeck, Guy, Davis, Jesse, and Blockeel, Hendrik. Completeness results for lifted variable elimination. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, pp. 572–580, 2013.

Van den Broeck, Guy. On the Completeness of First-Order Knowledge Compilation for Lifted Probabilistic Inference. In *Annual Conference on Neural Information Processing Systems (NIPS)*, December 2011.

Van den Broeck, Guy and Darwiche, Adnan. On the complexity and approximation of binary evidence in lifted inference. In *Advances in Neural Information Processing Systems 26 (NIPS),*, December 2013.

Van den Broeck, Guy and Davis, Jesse. Conditioning in first-order knowledge compilation and lifted probabilistic inference. In *Proceedings of AAAI*, 2012.

Van den Broeck, Guy, Taghipour, Nima, Meert, Wannes, Davis, Jesse, and De Raedt, Luc. Lifted Probabilistic Inference by First-Order Knowledge Compilation. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pp. 2178–2185, 2011.

Van den Broeck, Guy, Meert, Wannes, and Davis, Jesse. Lifted Generative Parameter Learning. In *3rd International Workshop on Statistical Relational AI (StaRAI)*, 2013.

Van den Broeck, Guy, Meert, Wannes, and Darwiche, Adnan. Skolemization for weighted first-order model counting. In *Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning (KR)*, 2014.