Evidence-Based Clustering for Scalable Inference in Markov Logic

Deepak Venugopal and Vibhav Gogate

Computer Science Department
The University of Texas at Dallas
Richardson, USA
`{dxv021000,vibhav.gogate}@utdallas.edu`

Abstract. Markov Logic is a powerful representation that unifies first-order logic and probabilistic graphical models. However, scaling-up inference in Markov Logic Networks (MLNs) is extremely challenging. Standard graphical model inference algorithms operate on the propositional Markov network obtained by grounding the MLN and do not scale well as the number of objects in the real-world domain increases. On the other hand, algorithms which perform inference directly at the first-order level, namely *lifted inference algorithms*, although more scalable than propositional algorithms, require the MLN to have specific symmetric structure. Worse still, evidence breaks symmetries, and the performance of lifted inference is the same as propositional inference (or sometimes worse, due to overhead). In this paper, we propose a general method for solving this "evidence" problem. The main idea in our method is to approximate the given MLN having, say, n objects by an MLN having k objects such that $k << n$ and the results obtained by running potentially much faster inference on the smaller MLN are as close as possible to the ones obtained by running inference on the larger MLN. We achieve this by finding clusters of "similar" groundings using standard clustering algorithms (e.g., K-means), and replacing all groundings in the cluster by their cluster center. To this end, we develop a novel distance (or similarity) function for measuring the similarity between two groundings, based on the evidence presented to the MLN. We evaluated our approach on many different benchmark MLNs utilizing various clustering and inference algorithms. Our experiments clearly show the generality and scalability of our approach.

1 Introduction

Markov Logic Networks (MLNs) [18, 4] unify first-order logic and probabilistic models and are arguably the most popular representation for statistical relational learning. They have been used in a wide variety of application domains including natural language understanding [17], computer vision [22] and planning [21]. Just as in conventional probabilistic models such as Bayesian networks and Markov networks, the key challenge in MLNs is to develop scalable inference algorithms. However, this challenge is more pronounced in MLNs because MLNs are template models, compactly specified using a first-order logic representation and as a result even a seemingly simple MLN can yield an arbitrary large (propositional) probabilistic model as the number of objects in the real-world domain increases.

Existing MLN inference algorithms can be broadly classified into two categories, propositional algorithms, which operate on the Markov network obtained by grounding the MLN and lifted algorithms, which operate directly on the first-order representation, grounding only as necessary. Propositional algorithms such as Gibbs Sampling [5] and Belief Propagation [28] do not scale well as the number of objects gets large, because they perform inference over the Markov network obtained by grounding the MLN, which for large domain-sizes can be huge. On the other hand, lifted inference algorithms [15, 3, 6, 26, 20, 10, 7, 2, 27, 13] either directly operate on the first-order structure or exploit symmetries in the propositional model and can therefore, in principle, scale significantly better than propositional inference algorithms.

Lifted inference algorithms typically suffer from two problems. First, they require MLNs to have a specific symmetric structure [3, 9, 23], which is not always the case in real-world applications. For example, to apply certain inference operations, the MLN needs to be composed of purely singleton atoms [9]. Second, a far more serious problem is that, in the presence of evidence most MLNs are not liftable because evidence breaks symmetries. As a concrete example, the symmetrical marginal probabilities in Fig. 1 (a) are broken with evidence (b). Therefore, a lifted algorithm that could potentially exploit the symmetry in (a) can no longer do so in (c). Thus, in the presence of evidence, lifted inference algorithms often resort to grounding the MLN. This is problematic because most interesting inference problems are almost always of the form $P(Q|E)$, i.e., computing the probability of a query given evidence. Therefore, there is a pressing need for inference algorithms that work without restrictions on the MLN structure or evidence. The main contribution of this paper is presenting one such method.

Our main idea is to reduce the number of objects in the domain of the MLN, thereby approximating it by a much smaller MLN such that the results obtained by performing inference on the smaller MLN are as close as possible to the ones obtained by running an expensive inference algorithm on the original MLN. To achieve this domain-reduction, we pre-process the MLN utilizing standard clustering algorithms such as K-means to merge together objects that are in some sense “similar” to each other from an inference perspective. Importantly, this pre-processing step allows us to plug-in existing grounded/lifted inference algorithms where the sampling-space (for sampling-based inference) or search-space (for search-based inference) can be controlled, which makes inference feasible even when the original MLN’s domain is extremely large.

In order to obtain an accurate domain-reduced approximation of the original MLN, we specify a novel distance function that measures similarity based on the evidence presented to the MLN. This distance function helps cluster together objects having similar evidence-structure. The inherent symmetry in MLN representation makes it more likely that similar evidence structure translates to approximately similar marginal probabilities. Thus, we compute the marginal probability for a single element of the cluster and project the same results to all elements in the cluster, thereby drastically reducing the complexity of inference.

We evaluated our approach on several benchmark MLNs available on the Alchemy website [11]. Also, in our experiments, we leverage a number of clustering algorithms from data-mining/machine learning literature implemented in Weka [8] to scale-up inference to very large domain-sizes. We experimented with two inference algorithms,

$\texttt{Wins}(A,A)$	0.56
$\texttt{Wins}(A,B)$	0.56
$\texttt{Wins}(A,C)$	0.56
$\texttt{Wins}(B,A)$	0.56
$\texttt{Wins}(B,B)$	0.56
$\texttt{Wins}(B,C)$	0.56
$\texttt{Wins}(C,A)$	0.56
$\texttt{Wins}(C,B)$	0.56
$\texttt{Wins}(C,C)$	0.56

(a) Original Marginals

$\texttt{Strong}(C)$
$\texttt{Wins}(A,C)$
$\texttt{Wins}(B,B)$
$\texttt{Wins}(B,C)$
$\texttt{Wins}(C,A)$

(b) Evidence

$\texttt{Wins}(A,A)$	0.6
$\texttt{Wins}(A,B)$	0.6
$\texttt{Wins}(B,A)$	0.63
$\texttt{Wins}(C,B)$	0.85
$\texttt{Wins}(C,C)$	0.85

(c) New Marginals

Fig. 1. Effect of evidence on an MLN with one formula, $1.75\ \texttt{Strong}(x) \Rightarrow \texttt{Wins}(x,y)$. The marginal probabilities which were equal in (a) become unequal in (c) due to evidence (b).

a propositional sampling-based algorithm, Gibbs sampling [5] and a lifted message-passing algorithm, Lifted Belief Propagation [20] to show the generality of our approach. Our results clearly illustrate that, using a fraction of the true groundings, we are able to approximate the marginal probabilities quite consistently on a wide variety of MLN structures with arbitrary evidence.

2 Preliminaries

First-order logic (FOL) consists of predicates (e.g., Friends) that represent relations between objects, logical connectives (e.g., $\vee$, $\neg$, etc.) and quantifiers ($\forall$, $\exists$). Each predicate has a parenthesized list of arguments which can be substituted by a term which can either be a logical variable (x), a constant (X) or a function. A formula in first order logic is a predicate (atom), or any complex sentence that can be constructed from atoms using logical connectives and quantifiers. For example, the formula $\forall x\ \texttt{Smokes}(x) \Rightarrow \texttt{Asthma}(x)$ states that all persons who smoke have asthma. A *ground* atom corresponding to a predicate is one where each term is substituted by a constant symbol.

We use a strict subset of FOL. Specifically, we make the following assumptions. First, we assume that there is a one-to-one mapping between the constant symbols and objects (Herbrand semantics). This means that any possible *world* is simply an assignment of True or False to every distinct *ground* atom. Second, we assume a function-free language where each variable is typed and the number of constant symbols is finite. Therefore, for any variable x, we can define a finite set Δ_x (domain of x) which consists of all the constant symbols that can be substituted for x. We refer to the constants corresponding to a domain as the domain's groundings. A ground formula is a formula obtained by substituting all of its variables with a constant. A ground KB is a KB containing all possible groundings of all of its formulas. For example, the grounding of a KB containing one formula, $\texttt{Smokes}(x) \Rightarrow \texttt{Asthma}(x)$ where $\Delta_x = \{Ana, Bob\}$, is a KB containing two ground formulas: $\texttt{Smokes}(Ana) \Rightarrow \texttt{Asthma}(Ana)$ and $\texttt{Smokes}(Bob) \Rightarrow \texttt{Asthma}(Bob)$.

Markov logic [4] extends FOL by softening the hard constraints expressed by the formulas. A soft formula or a weighted formula is a pair (f, w) where f is a formula in FOL and w is a real-number. A MLN denoted by $\mathcal{M}$, is a set of weighted formulas (f_i, w_i). Given a set of constants that represent objects in the domain, a Markov logic network defines a Markov network or a log-linear model. The Markov network is obtained by grounding the weighted first-order knowledge base and represents the following probability distribution.

$$P_{\mathcal{M}}(\omega) = \frac{1}{Z(\mathcal{M})} \exp\left(\sum_i w_i N(f_i, \omega)\right) \tag{1}$$

where ω is a world, $N(f_i, \omega)$ is the number of groundings of f_i that evaluate to `True` in the world ω and $Z(\mathcal{M})$ is a normalization constant or the partition function.

In this paper, we assume that the input MLN to our algorithm is in normal form [9, 12]. A *normal* MLN [9] is an MLN that satisfies the following two properties: (1) There are no constants in any formula, and (2) If two distinct atoms with the same predicate symbol have variables x and y in the same position then $\Delta_x = \Delta_y$. An important distinction here is that, unlike in previous work on lifted inference that use normal forms [9, 6] which require the MLN along with the associated evidence to be normalized, here we only require the MLN in normal form.

The two main inference problems in MLNs are computing the partition function and the marginal probabilities of query atoms given evidence. In this paper, we focus on the latter.

3 Domain Clustering

3.1 Problem Formulation

Let $\mathcal{M}$ denote an MLN with M predicates R_1, R_2, . . ., R_M, and N weighted formulas f_1, f_2, . . ., f_N. Let $G_{\mathcal{M}}$ denote the propositional Markov network obtained by grounding all the formulas in $\mathcal{M}$. Let **E** = $\{E_k\}_{k=1}^S$ be the set of *evidences*. Each $E_k \in \mathbf{E}$ represents a single ground atom that is known to be either `True` or `False`. Let **I** be a set of indices of the form (i, j) such that $1 \leq i \leq M$, $1 \leq j \leq A_i$, where A_i is the arity of the i-th predicate. In other words, (i, j) is an index to the j-th argument of the i-th predicate in $\mathcal{M}$.

Let R be a binary relation on **I** such that $(i, j)\ R\ (a, b)$ iff there exists a formula $f \in \mathcal{M}$ such that: (1) f contains atoms having predicate symbols indexed by i and a, and (2) a logical variable x of f appears as the j-th argument and as the b-th argument of atoms having predicate symbols indexed by i and a respectively. Clearly, R is symmetric and reflexive. Let R^+ be the transitive closure of R on **I**. R^+ is an equivalence relation on **I**. Let $\mathcal{I} = \{\mathcal{I}_1\, \mathcal{I}_2 \ldots \mathcal{I}_P\}$ denote the set of equivalence classes of **I** due to the equivalence relation R^+. Let $\Delta_{\mathcal{I}_k}$ denote the domain (possible groundings) of an element of $\mathcal{I}_k$. Note that since we assume that the MLN is in normal form, all elements of $\Delta_{\mathcal{I}_k}$ have the same domain.

Example 1. Let $\mathcal{M}$ contain exactly one formula $\mathsf{R}_1(x,y) \wedge \mathsf{R}_2(y,z) \Rightarrow \mathsf{R}_3(z,x)$. Let $\Delta_x = \Delta_y = \Delta_z = \{A,B\}$. $\mathcal{I} = \{\{(1,1), (3,2)\}, \{(1,2), (2,1)\}, \{(2,2), (3,1)\}\}$. $\Delta_{\mathcal{I}_1} = \{A,B\}$ and grounding $\mathcal{I}_1$ with A, yields the partially ground formula, $\mathsf{R}_1(A,y) \wedge \mathsf{R}_2(y,z) \Rightarrow \mathsf{R}_3(z,A)$.

To reduce the total number of formulas in $G_{\mathcal{M}}$, we reduce the number of groundings in each $\mathcal{I}_k \in \mathcal{I}$ independently. Specifically, for each $\Delta_{\mathcal{I}_k}$, we learn a new domain, $\hat{\Delta}_{\mathcal{I}_k}$ and a surjective mapping $\zeta : \Delta_{\mathcal{I}_k} \rightarrow \hat{\Delta}_{\mathcal{I}_k}$, i.e., $\forall\, \mu \in \hat{\Delta}_{\mathcal{I}_k}$, $\exists\, C \in \Delta_{\mathcal{I}_k}$ such that $\zeta(C) = \mu$. We formulate this domain-reduction problem ($|\hat{\Delta}_{\mathcal{I}_k}| << |\Delta_{\mathcal{I}_k}|$) as a standard clustering problem below.

Definition 1. *Given a distance measure d between any two groundings of $\mathcal{I}_k \in \mathcal{I}$ and the number of clusters for $\mathcal{I}_k$ equal to r_k, we define the clustering problem as,*

$$\min_{\mathbf{C}_1 \ldots \mathbf{C}_P} \sum_{k=1}^{P} \sum_{j=1}^{r_k} \sum_{C_{kj} \in \mathbf{C}_{kj}} d(C_{kj}, \mu_{kj}) \tag{2}$$

where $\mathbf{C}_{kj}$ corresponds to all groundings of $\mathcal{I}_k$ that are placed in cluster j, μ_{kj} is the cluster-center of $\mathbf{C}_{kj}$, i.e., it represents the "average grounding" for that cluster, $\zeta^{-1}(\mu_{kj}) = \mathbf{C}_{kj}$.

Each cluster-center in some sense "compresses" the original domain, and we generate a new MLN $\hat{\mathcal{M}}$ from $\mathcal{M}$ by replacing each $\Delta_{\mathcal{I}_k}$ with $\hat{\Delta}_{\mathcal{I}_k} = \{\mu_{kj}\}_{j=1}^{r_k}$. Importantly, the formulation in Eq. (2) allows us control the inference-complexity in $\hat{\mathcal{M}}$ even when $G_{\mathcal{M}}$ is extremely large. This is important because, for arbitrary MLN structures or for inference with evidence, even state-of-the-art inference techniques end up working on a model whose size is comparable to $G_{\mathcal{M}}$ and in most cases, $G_{\mathcal{M}}$ grows rapidly with domain-size. For example, consider the MLN, $\mathsf{R}(x, y) \wedge \mathsf{S}(y, z) \Rightarrow \mathsf{T}(z, x)$, even for $\Delta_x = \Delta_y = \Delta_z = \Delta_u = 100$, the number of formulas in $G_{\mathcal{M}}$ is already one million. Further, the search space (for search-based algorithms) or the sampling space (for sampling-based algorithms) is massive, i.e., exponential in the total number of ground atoms in the MLN. By clustering, we are essentially compressing this large space and now any existing inference algorithm can be used to solve large problems as they implicitly work in this reduced space. The key advantage is that this space complexity can now be controlled based on the cluster-size. Specifically,

Proposition 1. *The number of ground atoms in $\hat{\mathcal{M}}$ is $O(Mr^A)$, where M is the number of predicates in $\mathcal{M}$, $r = \max_k r_k$ and A is the maximum arity of a predicate in $\mathcal{M}$.*

Clearly, the ground atoms in $\hat{\mathcal{M}}$ are different from those in $\mathcal{M}$. Specifically, an atom in $\hat{\mathcal{M}}$ is ground with cluster-centers rather than concrete objects of the original MLN. Thus, one ground atom in $\hat{\mathcal{M}}$ implicitly corresponds to multiple ground atoms in $\mathcal{M}$. This also means that in $\hat{\mathcal{M}}$, the original evidence **E** needs to be modified because it is specified on the ground atoms of $\mathcal{M}$. Therefore, we approximate **E** with $\hat{\mathbf{E}}$ which specifies the evidence on atoms ground with cluster-centers instead of the original objects in $\mathcal{M}$. To specify this, we define the *expansion* of a ground atom in $\hat{\mathcal{M}}$ as the set of all groundings that it represents in $\mathcal{M}$. Formally,

Definition 2. *The* expansion *of the j-th ground atom corresponding to the i-th predicate ($\mathsf{R}_i(\mu_{i_1 j_1}, \ldots \mu_{i_{A_i} j_{A_i}})$) in $\hat{\mathcal{M}}$ is denoted by π_{ij} and consists of all distinct ground atoms of the form $\mathsf{R}_i(C_1, \ldots, C_{A_i})$ where $C_k \in \zeta^{-1}(\mu_{i_k j_k})$.*

Clearly, if we assert in $\hat{\mathbf{E}}$ that a ground atom in $\hat{\mathcal{M}}$ is `True` (or `False`), this implicitly asserts that every grounding in its expansion is `True` (or `False`). Given a clustering of the domains, in order to best approximate $\mathbf{E}$ for this clustering, we minimize the approximation error as follows.

$$\min_{\hat{\mathbf{E}}} |\mathbf{E} \triangle \bar{\pi}(\hat{\mathbf{E}})| \tag{3}$$

where $\hat{\mathbf{E}}$ is a subset of the ground atoms in $\hat{\mathcal{M}}$ and each grounding is assigned a sign (positive/`True` or negative/`False`), $\bar{\pi}(\hat{\mathbf{E}})$ expands every grounding in $\hat{\mathbf{E}}$ and assigns each grounding in the expansion the same sign as its corresponding grounding in $\hat{\mathbf{E}}$. The $\triangle$ operator computes the symmetric difference between $\mathbf{E}$ and $\bar{\pi}(\hat{\mathbf{E}})$. (Note that a grounding with different signs is treated as distinct elements for our purpose). $\hat{\mathbf{E}}$ can be optimally chosen using the following proposition.

Proposition 2. *Let π_{ij} be the expansion of one grounding ($\hat{E}$) in $\hat{\mathbf{E}}$. Let n_+ be the count of positive-sign elements and n_-, the count of negative-sign elements in $\pi_{ij} \cap \mathbf{E}$. Eq.* (3) *is optimized by assigning $\hat{E}$ as positive (negative) if $n_+ \geq \frac{|\pi_{ij}|}{2}$ $\left(n_- \geq \frac{|\pi_{ij}|}{2}\right)$.*

Algorithm 1 shows a schematic illustration of our algorithm to compute the marginal probabilities in an MLN given evidence. Algorithm 1 needs three other algorithms to be specified namely, the distance function, clustering algorithm and the inference algorithm. The amount of reduction applied to each domain is specified as the cluster-bound α. The algorithm starts by computing the partition $\mathcal{I}$ from the term dependencies in $\mathcal{M}$. Next, to each $\mathcal{I}_k \in \mathcal{I}$, the clustering algorithm $\mathcal{L}$ is applied which outputs the clustered domain $\Delta_{\mathcal{I}_k}$ as well as the mapping function ζ. $\Delta_{\mathcal{I}_k}$ is now replaced by its approximation in the new MLN $\hat{\mathcal{M}}$. Once all the domains are suitably reduced, the next step is to approximate the evidence based on the reduced domains. Using Proposition 2, for every grounding of every atom in $\hat{\mathcal{M}}$, we make a decision as to whether it is to be considered positive evidence, negative evidence or treated as a grounding whose truth value is unknown. This yields the approximate evidence set $\hat{\mathbf{E}}$. We then invoke the inference algorithm $\mathcal{F}$ to compute the marginals in $\hat{\mathcal{M}}$. Finally, we project the results obtained on $\hat{\mathcal{M}}$ back to the original domains. Specifically, if a grounding in $\hat{\mathcal{M}}$ has a marginal probability p, then each grounding in its expansion is assigned the same probability.

3.2 Distance Function

The distance function is a key parameter that affects the quality of the generated clusters in Eq. (2) and in turn the inference results computed in Algorithm 1. The advantage of our formulation is that it is quite easy to plug-in a new distance function and generate "new" inference algorithms targeted towards specific applications or datasets. Here, we develop a generic distance measure using the evidence specified on the MLN.

Algorithm 1: Compute-Marginals

```
Input: MLN $\mathcal{M}$, Evidence E, set of query predicates Q, Distance function $d$, Clustering
       function $\mathcal{L}$, Inference algorithm $\mathcal{F}$, cluster-bound $\alpha$
Output: Marginal probabilities $\mathcal{P}$ for each ground atom corresponding to a predicate in Q
1  Compute the partition $\mathcal{I}$ from $\mathcal{M}$
2  $\hat{\mathcal{M}} = \mathcal{M}$
3  for $\mathcal{I}_k \in \mathcal{I}$ do
4      $numclusters = \alpha \times \Delta_{\mathcal{I}_k}$
5      $(\hat{\Delta}_{\mathcal{I}_k}, \zeta) = \mathcal{L}(numclusters, d)$
6      Replace $\Delta_{\mathcal{I}_k}$ with $\hat{\Delta}_{\mathcal{I}_k}$ in $\hat{\mathcal{M}}$
7  Construct $\hat{\mathbf{E}}$ based on Proposition 2
8  $\hat{\mathcal{P}} = \mathcal{F}(\hat{\mathcal{M}}, \hat{\mathbf{E}}, \mathbf{Q})$
9  for Each $\mathsf{R}_k \in \mathbf{Q}$ do
10     for Each $j$, where $j$ indexes the possible groundings of $\mathsf{R}_k$ in $\hat{\mathcal{M}}$ do
11         for Each $t$, where $t$ indexes the possible groundings of $\mathsf{R}_k$ in the expansion $\pi_{kj}$
           do
12             $\mathcal{P}(\mathsf{R}_k, t) = \hat{\mathcal{P}}(\mathsf{R}_k, j)$
13 return $\mathcal{P}$
```

Example 2. Consider the MLN with one formula $\mathsf{R}(x) \Rightarrow \mathsf{S}(x,y)$ with weight 1.75 and domain $\Delta_x = \{A, B, C\}$. Let the evidence $\mathbf{E} = \{\mathsf{R}(A), \mathsf{R}(B)\}$. The task is to compute the marginal probabilities of all groundings of $\mathsf{S}(x,y)$ which we refer to as the query. The exact marginal probabilities for the query are, $\mathsf{S}(A,y) = \mathsf{S}(B,y) = 0.5$, $\mathsf{S}(C,y) = 0.56$. Thus, an ideal distance function should give us a clustering of Δ_x where A and B are placed in the same cluster as they have the same marginals w.r.t the query variable. To do this, we observe that the evidence on $\mathsf{R}(A) \Rightarrow \mathsf{S}(A,y)$ and $\mathsf{R}(B) \Rightarrow \mathsf{S}(B,y)$ are "symmetrical", i.e., they satisfy the same number of groundings and consequently the number of groundings that are left unsatisfied in both the formulas is the same. In other words, when $x = A$, the relevant evidence yields MLN $\mathcal{M}'$ and when $x = B$, its yields $\mathcal{M}''$ and if $\mathcal{M}'$ is sufficiently close to $\mathcal{M}''$, we would want all the groundings where $x = A$ clustered together with the groundings where $x = B$ because they are likely to have the same marginal probabilities. We formalize this intuitive idea below.

Let $\mathcal{M}_{C_{kj}}$ represent the MLN obtained after grounding $\mathcal{I}_k$ with the j-th constant in $\Delta_{\mathcal{I}_k}$. Clearly, in the general case, for any two distinct j_1, j_2, $\mathcal{M}_{C_{kj_1}}$ and $\mathcal{M}_{C_{kj_2}}$ are not necessarily independent MLNs as there may be atoms in $\mathcal{M}_{C_{kj_1}}$ that are also present in $\mathcal{M}_{C_{kj_2}}$. However, in our distance function, we relax the constraints/dependencies between $\mathcal{M}_{C_{kj_1}}$, $\mathcal{M}_{C_{kj_2}}$ and assume these to be independent MLNs and compute the distance between these two MLNs. Specifically, we define a feature vector $\mathbf{U}_{C_{kj}} = c_{f_1}$, $\ldots c_{f_N}$, where c_{f_k} is the number of groundings in formula f_k of MLN $\mathcal{M}_{C_{kj}}$ satisfied due to the evidence $\mathbf{E}$. The distance is computed as $d(C_{kj_1}, C_{kj_2}) = ||\mathbf{U}_{C_{kj_1}} - \mathbf{U}_{C_{kj_2}}||$.

Even though the above distance function seems like an intuitive and reasonable heuristic, it turns out that computing the distance function efficiently is infeasible in the general case because computing the counts in $\mathbf{U}_{C_{kj}}$ is a hard problem when **E** is large. Formally, the following result has been shown in [4],

Theorem 1. *Computing the number of satisfied groundings of a first-order clause in a database is $\#P$-complete in the length of the clause.*

Using database terminology, computing the counts in $\mathbf{U}_{C_{kj}}$ requires computing joins over an arbitrary number of relations (or tables). Therefore, we further relax the constraints/dependencies within the atoms in a formula to guarantee feasibility of computing $\mathbf{U}_{C_{kj}}$ when the size of the evidence-set is very large. In order to formalize this clearly, we specify the ground atoms using a relational database. Further, we also assume that each formula is reduced to a clausal form. The i-th predicate R_i is stored as a relational database table R_i with $A_i + 1$ columns (A_i is the arity), namely, id_1,, id_2 ... id_{A_i} and val. The first A_i columns correspond to a specific grounding and the val column specifies whether that ground atom is `True` ($val = 1$), `False` ($val = 0$) or unknown ($val = -1$). Given such a database, computing the feature vector involves counting the number of groundings of the formulas in $\mathcal{M}_{C_{kj}}$ that are satisfied by the evidence, which according to Theorem 1 is a hard problem in the general case. Though Theorem 1 is not an issue when the number of evidence atoms is small, to scale-up inference to arbitrarily large evidence-sets, we adopt the following approach. Instead of computing the exact number of groundings for a formula satisfied by the evidence, which involves an arbitrary number of joins over the relations in the formula, we approximate this with a vector of counts, where each count is computed on a subset of relations and the computation involves a bounded number of joins over these relations.

Example 3. Let $\mathcal{M}$ contain one formula, $\neg\mathsf{R}(x, y) \vee \neg\mathsf{S}(y,z) \vee \mathsf{T}(z,x)$, where $\Delta_x = \{A, B, C\}$. To compute the count of satisfied groundings for $x = A$, we compute its inverse, i.e., the number of unsatisfied groundings for $x = A$. The satisfied count is simply the difference between the total number of groundings and the number of unsatisfied groundings. Since the total number of groundings $\Delta_y \times \Delta_z$ is a constant for all groundings of x, it does not affect the clustering and we simply ignore it. The unsatisfied groundings for $x = A$ is given by the following relational algebra expression

$$\sigma_{R.val=1 \wedge S.val=1 \wedge T.val=0}\big((\sigma_{R.id_1=A}(R) \bowtie_{R.id_2=S.id_1} S) \\ \bowtie_{S.id_2=T.id_1 \wedge R.id_1=T.id_2} T\big) \tag{4}$$

where σ is the selection operator and $\bowtie$ is the join operator. Clearly, the above expression has two joins. However, if we impose a constraint that no joins are allowed during the computation of the feature vector, we approximate Eq. (4) by implicitly assuming that each predicate in the formula is independent i.e. we ignore the joins to obtain a vector of counts by counting the tuples returned by 3 separate queries, $\sigma_{R.val=1 \wedge R.id_1=A}(R)$, $\sigma_{S.val=1}(S)$ and $\sigma_{T.val=0 \wedge T.id_2=A}(T)$. An alternate distance function can be obtained if we only allow exactly one join in a query. In this case, we can get a better approximation of Eq. (4) by considering two queries,

$$\sigma_{R.id_1=A \wedge R.val=1}(R) \bowtie_{R.id_2=S.id_1} \sigma_{S.val=1}(S)$$

$$\sigma_{S.val=1}(S) \bowtie_{S.id_2=T.id_1} \sigma_{T.val=0 \wedge T.id_2=A}(T)$$

```
Algorithm 1: Build-Query
  Input: Clausal formula $f_t$
  Output: Relational-Algebra expression $\mathcal{Q}$
1 $\mathcal{Q} = \emptyset$
2 for $R_i \in f_t$ do
3     $Rvalue = 1$
4     if $R_i$ is positive then
5         $Rvalue = 0$
6     if $\mathcal{Q} = \emptyset$ then
7         $\mathcal{Q} = \mathcal{Q} + \sigma_{R_i.val=Rvalue}(R_i)$
8     else
9         $\mathcal{Q} = \mathcal{Q} \bowtie_\theta \sigma_{R_i.val=Rvalue}(R_i)$
```

Fig. 2. Building a query for the feature vector

The algorithm illustrated in Fig. 3 generalizes the idea in the above example and computes the feature vectors for a specific $\mathcal{I}_k \in \mathcal{I}$. The algorithm generates multiple queries corresponding to each grounding of $\mathcal{I}_k$ such that the number of joins in each query is lesser than J. For this, we go over each formula f_t, and first check if f_t is *relevant* to $\mathcal{I}_k$, i.e., if f_t contains at least one atom corresponding to R_i such that for some p, $(i, p) \in \mathcal{I}_k$, then f_t is a relevant formula for clustering $\mathcal{I}_k$, otherwise, we ignore f_t. This is because, the features from f_t which are not relevant to $\mathcal{I}_k$ remains identical for every grounding of x and therefore never affects the clustering. For every relevant f_t, we first build the complete query which is a sequence of θ-joins on the tables corresponding to every atom in f_t. The query selects the the groundings of f_t that are not satisfied by the evidence. The θ in the join specifies variables shared among atoms in f_t. For e.g. In a formula $\neg\mathsf{R}(x) \vee \mathsf{S}(x)$, the θ-join is specified as $\sigma_{R.val=1}(R) \bowtie_{R.id_1=S.id_1} \sigma_{S.val=0}(S)$. Once we build the full query, we simply walk through the query executing no more than J joins at a time. For each atom which has a variable that corresponds to some element of $\mathcal{I}_k$, we ground the variable by enforcing the select condition in line 11 of the algorithm. We execute the partial query $\mathcal{Q}'$ with a maximum of J joins and store the result (count) in the feature vector. Next, we remove $\mathcal{Q}'$ from $\mathcal{Q}$ and relax the next θ- join condition as follows. Among all the tables mentioned in $\mathcal{Q}'$, we select one table R_s, that participates in the next join operation in $\mathcal{Q} - \mathcal{Q}'$. We only retain the join conditions related to R_s in the next join in $\mathcal{Q} - \mathcal{Q}'$ and remove the rest of the conditions. We continue until we empty the original query $\mathcal{Q}$. Finally, we return the vector of counts accumulated across all queries for each grounding of $\mathcal{I}_k$.

4 Related Work

Several previous approaches have been suggested for improving the scalability of inference in MLNs. Most of these approaches can be termed as lifted inference algorithms since they either use rules that can be directly applied on the first-order structure or identify symmetries in the ground representation to perform efficient inference. Both

Algorithm 1: Compute-Features

```
Input: $\mathcal{M}$ and its associated relational DB, join-bound $J$, $\mathcal{I}_k \in \mathcal{I}$
Output: Feature vector set $\{\mathbf{U}_{C_{kj}}\}_{j=1}^{\Delta_{\mathcal{I}_k}}$
1  U = $\emptyset$
2  for $C_{kj} \in \Delta_{\mathcal{I}_k}$ do
3      $\mathbf{U}_{C_{kj}}$ = $\emptyset$
4      for $f_t \in$ F do
5          if $f_t$ is not relevant to $\mathcal{I}_k$ then
6              continue
7          $\mathcal{Q}$ = Build-Query($f_t$)
8          while $\mathcal{Q}$ not empty do
9              $\mathcal{Q}'$ = Select a sub-query containing up to the first $J$ joins in $\mathcal{Q}$
10             for $R_i \in \mathcal{Q}'$ do
11                 if $\exists$ $p$ such that $(i, p) \in \mathcal{I}_k$ then
12                     Wrap a select ($\sigma_{R_i.id_p = C_{kj}}$) around $R_i$
13             $\mathbf{U}_{C_{kj}}$.append(Count($\mathcal{Q}'$))
14             Let $R_s$ be a table in $\mathcal{Q}'$ whose attribute participates in the $\theta$-join after $\mathcal{Q}'$
15             if $R_s$ = $\emptyset$ then
16                 $\mathcal{Q}$ = $(\mathcal{Q} - \mathcal{Q}')$
17             else
18                 Relax the $\theta$-join and include only those constraints involving $R_s$
19                 $\mathcal{Q}$ = $R_s \bowtie_\theta (\mathcal{Q} - \mathcal{Q}')$
20     U.append($\mathbf{U}_{C_{kj}}$)
21 return U
```

Fig. 3. Algorithm to compute the feature vectors.

exact [3, 6, 26, 1] as well as approximate [20, 10, 7, 13, 27, 2] lifted algorithms have been developed that can greatly improve scalability. However, all these algorithms are efficient only when given the right MLN structure/evidence. Specifically, [1, 25] show that efficient inference is possible when presented with specific evidence-structure. More recently, [24] have proposed to counter the evidence-problem by adding more symmetries that make the MLN liftable. Specifically, they compute a low-rank boolean matrix factorization of the evidence matrix which implicitly induces a clustering whereas we explicitly cast it as a clustering problem thereby allowing us the flexibility to use a range of clustering algorithms and also better control of the inference-complexity. Further, [24] handles only binary evidence while our approach is much more general. Finally, our approach of pre-processing the MLN is related to [19] which develops a systematic grounding procedure that can reduce the ground MLN size in many cases, and our approach of leveraging databases for MLN inference is related to [14].

5 Experiments

5.1 Setup

We evaluate our approach on 4 benchmark MLNs available in Alchemy [11], namely Entity Resolution (ER), Segmentation (Seg), Web Linkage analysis (WebKB) and Protein Interaction (Protein). Additionally, we added two new MLNs that have different structures called Student ($\texttt{Teaches}(i, c) \wedge \texttt{Prereq}(c, c1) \Rightarrow \texttt{Takes}(s, c1)$) and Relation ($\texttt{Related}(i, j) \wedge \texttt{Friends}(j, k) \Rightarrow \texttt{Loves}(k, i)$). For our experiments, we implemented our system using MySQL. To speed up query processing, we created n indexes for a table corresponding to a n-ary predicate, where the column corresponding to each argument of a predicate is indexed separately. For the distance function, we limit the number of joins (J) to 1. In the inference subroutine, we used two algorithms, a lifted algorithm based on message passing, Lifted-BP [20] and a propositional algorithm based on sampling, Gibbs sampling [5]. We used the implementation of both these algorithms from Alchemy. For the clustering subroutine, we experimented with four different algorithms available in Weka [8] namely, KMeans++ (KM), Expectation-Maximization (EM), Hierarchical clustering (HC) and XMeans (XM). We ran all our experiments on a quad-core Ubuntu machine with 6 GB RAM.

5.2 Approximation results on benchmarks

Fig. 4 illustrates the approximation error on each benchmark for all combinations of the two inference and four clustering algorithms. The *x-axis* plots the inverse compression ratio $ICR = \frac{N_C}{N_G}$, where N_C is the total number of ground formulas in the approximated MLN after clustering and N_G is the total number of ground formulas in the original MLN. The *y-axis* shows the approximation error calculated as follows. $Err = \frac{\sum_{g \in G} D_{KL}(P_g || P'_g)}{|G|}$, where D_{KL} is the standard KL-Divergence distance measure, G refers to all groundings of a query predicate, P_g is the marginal distribution of g computed from the original MLN and P'_g is computed from the approximate MLN using clustered domains. For fairness, both marginals are computed using the same inference algorithm. We set 50% of arbitrary groundings as evidence, where 25% are `True` and 25% are `False`.

Fig. 4 illustrates the trade-off between accuracy and complexity. As ICR increases, the complexity increases, however, the approximation error reduces because we map the original domains to a larger set thereby reducing the difference between the original MLN and the approximate MLN. The structure of the MLN also plays an important role in determining the accuracy of the approximation. For some cases such as Student in Fig. 4 (a), (g) the error goes down quite rapidly initially and stays consistently low afterwards. In some other cases such as ER, Fig. 4 (f), (l), the change is more gradual. This is because ER contains complex formulas with multiple self-joins such as the transitive relation which make it harder to approximate. In almost all cases for Lifted-BP (except ER), the approximation error was below 0.2 for even small compression ratios. For Gibbs sampling though, in general, it took slightly larger compression ratios before the approximation was close to ground inference such as in the benchmarks Protein (k) and ER (l). One of the reasons for this could be that deterministic dependencies or hard constraints tend to be problematic for Gibbs sampling [16], i.e., if the probabilities lie at the extremes then the Gibbs sampler mixes very slowly. Therefore the approximations given by Gibbs sampling are not very accurate even for the fully ground model. Among the clustering algorithms, KM and HC clearly outperformed XM and EM. In almost all cases, KM and HC produced clusterings that produced more stable and consistent results compared to EM and XM. For example, in (a), (d) and (h) the EM algorithm gave poor results while in (i), XM gave poor results.

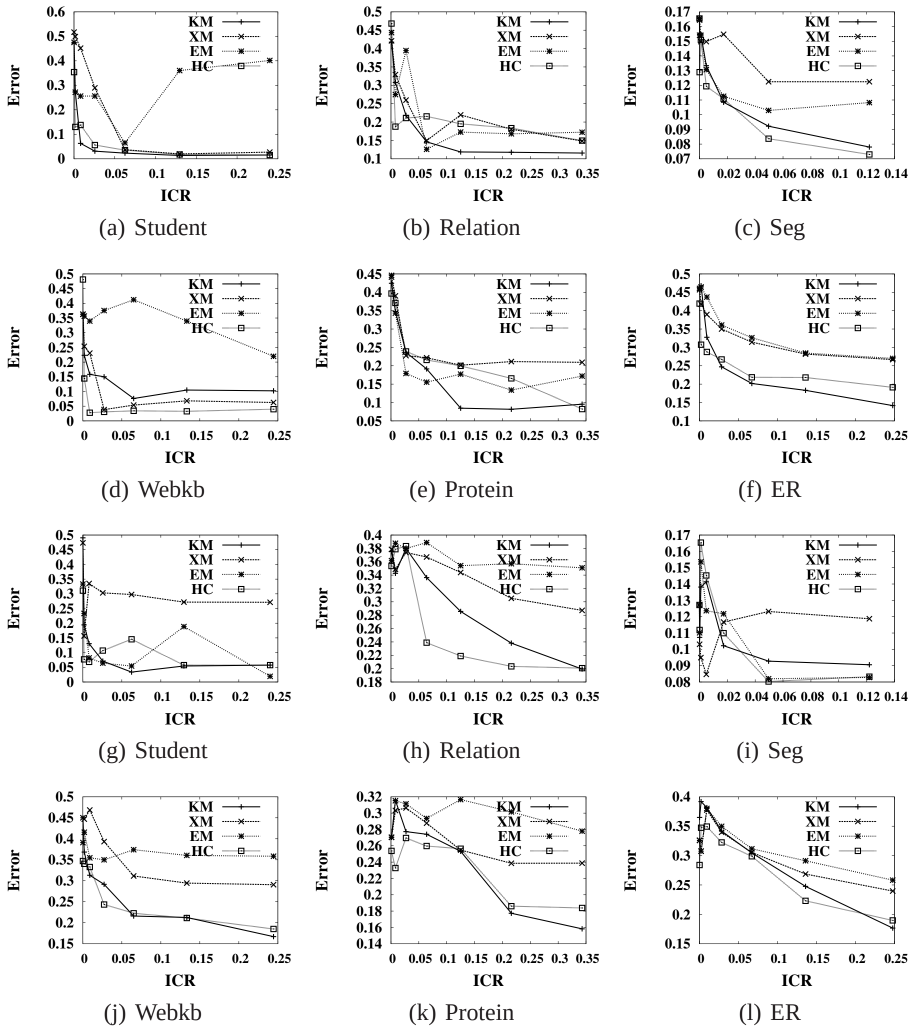
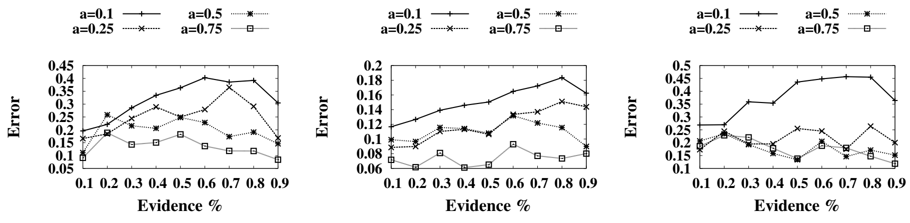


Fig. 4. Approximation-error vs ICR. The y-axis shows the average KL-Divergence of the marginals computed on the clustered MLN from the marginals computed on the original MLN (smaller is better). (a) - (f) show the results using Lifted-BP, (g) - (l) using Gibbs sampling.

5.3 Effect of Evidence

Fig. 5 illustrates the error for different values of cluster-bounds (α) and varying amount of evidence. The results shown Fig. 5 use K-Means++ for clustering and Lifted-BP for inference. As expected, using a larger value of α in most cases leads to lower errors due to a better approximation of the original MLN. Also, it can be seen that in most of the cases illustrated in Fig. 5, for very small or very large amounts of evidence, the errors seem to go down. This is quite consistent with the effect that evidence has on MLNs as previously shown in the introduction. Evidence breaks symmetries in the MLN and thus if very few groundings or nearly all groundings are evidence, as there are more symmetries, the inference algorithms tend to give us better approximations (for all α values) than the cases shown in middle portion of the graphs where the random evidence makes inference more challenging.

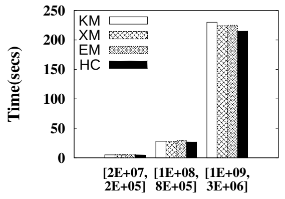


(a) Relation (b) Seg (c) WebKB

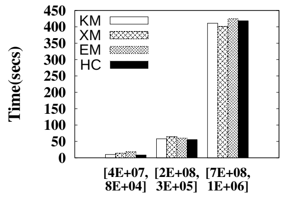
Fig. 5. Illustrating the effect of evidence. The x-axis varies the amount of evidence on the atoms in the MLN. The y-axis plots the approximation error for varying cluster-bounds. The experiment is run using K-Means for clustering and Lifted-BP for inference.

5.4 Scalability

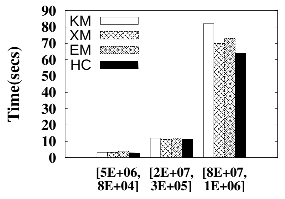
Fig. 6 illustrates the scalability of our approach when handling large domain-sizes. For different domain-sizes, we show the time in seconds it takes to compute the approximate MLN after clustering. We used an α value of 0.25 for these experiments and introduced 50% random evidence with half of them `True` and the other half `False`. As expected, the time taken to compute the approximate MLN increases as the domain-size grows. However, it should be noted that none of the MLNs in Fig. 6 could be processed by existing ground/lifted inference algorithms in Alchemy before running out of memory as the number of ground formulas is extremely large. For example, one instance of the Relation MLN in Fig. 6 (a) has one billion groundings. Thus, without approximating the MLN, there is no feasible approach to inference in such large models. As shown by our results, we were able to complete processing the MLN in a reasonable amount of time even when the groundings reached a trillion as in Fig. 6 (e). Also, the number of first-order formulas and their structure play a role in determining the complexity due to the distance function computation. Recall that we compute a vector for every formula in the MLN. Therefore, a larger number of formulas mean more computations on the database. For instance, Fig. 6 (e) has just one formula while (f) has 8 formulas which have more complex structure. Therefore, even though the number of ground formulas in (e) is a trillion while in (f) it is a billion, we took more time to process (f). Further, it can be seen that for each of the benchmarks, the third instance (the largest MLN) takes a visibly longer time when compared to the first two instances. This is expected because, when the size of the database grows really large as is the case for very large domain-sizes, it typically requires many more hard disk accesses for query processing which causes it to slow down. Finally, as seen in the results, the type of clustering has minimal impact on the time taken to process the MLN, i.e., nearly all clustering methods took approximately the same amount of time.



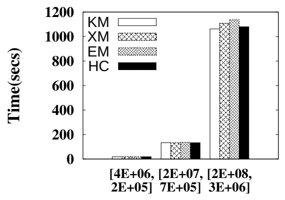
(a) Relation



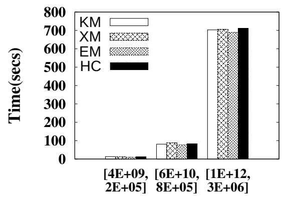
(b) Webkb



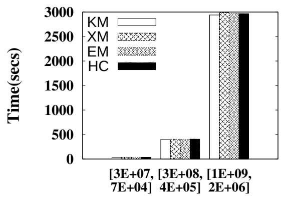
(c) Protein



(d) ER



(e) Student



(f) Seg

Fig. 6. Scalability experiments. The y-axis shows the time taken to form the approximate MLN and the x-axis shows $[N_f, N_a]$, where N_f is the number of ground formulas and N_a is the number of ground atoms.

6 Conclusion

In this paper, we presented an approach for scaling up inference in MLNs. Existing approaches either ground the MLN which makes it too large to process or use rules to identify symmetries and perform lifted inference. However, lifting rules are applicable only in certain specific, symmetric cases and more importantly, in the presence of evidence, these symmetries are broken, rendering lifted inference powerless. To achieve scalable inference for such hard cases in which we can have arbitrary MLN structures with arbitrary evidence, we proposed to compress the original MLN. Specifically, we defined a novel distance function that is sensitive to the evidence presented to the MLN and used it to replace groups of similar objects in the MLN by their cluster centers. Our experimental results on several benchmark MLNs clearly illustrated the high accuracy and scalability of our approach.

Acknowledgements This research was partly funded by ARO MURI grant W911NF-08-1-0242, by the AFRL under contract number FA8750-14-C-0021 and by the DARPA Probabilistic Programming for Advanced Machine Learning Program under AFRL prime contract number FA8750-14-C-0005. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of DARPA, AFRL, ARO or the US government.

References

1. Bui, H., Huynh, T., de Salvo Braz, R.: Exact lifted inference with distinct soft evidence on every object. In: Proceedings of the 26th AAAI Conference on Artificial Intelligence. AAAI Press (2012)
2. Bui, H., Huynh, T., Riedel, S.: Automorphism groups of graphical models and lifted variational inference. In: Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence. pp. 132–141. AUAI Press (2013)
3. de Salvo Braz, R.: Lifted First-Order Probabilistic Inference. Ph.D. thesis, University of Illinois, Urbana-Champaign, IL (2007)
4. Domingos, P., Lowd, D.: Markov Logic: An Interface Layer for Artificial Intelligence. Morgan & Claypool, San Rafael, CA (2009)
5. Geman, S., Geman, D.: Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. IEEE Transactions on Pattern Analysis and Machine Intelligence 6, 721–741 (1984)
6. Gogate, V., Domingos, P.: Probabilistic Theorem Proving. In: Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence. pp. 256–265. AUAI Press (2011)
7. Gogate, V., Jha, A., Venugopal, D.: Advances in Lifted Importance Sampling. In: Proceedings of the 26th AAAI Conference on Artificial Intelligence. AAAI Press (2012)
8. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: An update. SIGKDD Explor. Newsl. 11(1), 10–18 (2009)
9. Jha, A., Gogate, V., Meliou, A., Suciu, D.: Lifted Inference from the Other Side: The tractable Features. In: Proceedings of the 24th Annual Conference on Neural Information Processing Systems (NIPS). pp. 973–981 (2010)
10. Kersting, K., Ahmadi, B., Natarajan, S.: Counting Belief Propagation. In: Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence. pp. 277–284. AUAI Press (2009)
11. Kok, S., Sumner, M., Richardson, M., Singla, P., Poon, H., Lowd, D., Wang, J., Domingos, P.: The Alchemy System for Statistical Relational AI. Tech. rep., Department of Computer Science and Engineering, University of Washington, Seattle, WA (2008), http://alchemy.cs.washington.edu
12. Milch, B., Zettlemoyer, L.S., Kersting, K., Haimes, M., Kaelbling, L.P.: Lifted Probabilistic Inference with Counting Formulas. In: Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence. pp. 1062–1068 (2008)
13. Niepert, M.: Markov chains on orbits of permutation groups. In: Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence. pp. 624–633. AUAI Press (2012)
14. Niu, F., Ré, C., Doan, A., Shavlik, J.W.: Tuffy: Scaling up statistical inference in markov logic networks using an rdbms. PVLDB 4(6), 373–384 (2011)

15. Poole, D.: First-Order Probabilistic Inference. In: Proceedings of the 18th International Joint Conference on Artificial Intelligence. pp. 985–991 (2003)
16. Poon, H., Domingos, P.: Sound and Efficient Inference with Probabilistic and Deterministic Dependencies. In: Proceedings of the 21st National Conference on Artificial Intelligence. pp. 458–463. AAAI Press (2006)
17. Poon, H., Domingos, P.: Joint Unsupervised Coreference Resolution with Markov Logic. In: Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing. pp. 649–658. ACL (2008)
18. Richardson, M., Domingos, P.: Markov Logic Networks. Machine Learning 62, 107–136 (2006)
19. Shavlik, J.W., Natarajan, S.: Speeding up inference in markov logic networks by preprocessing to reduce the size of the resulting grounded network. In: Proceedings of the 21st International Joint Conference on Artificial Intelligence. pp. 1951–1956 (2009)
20. Singla, P., Domingos, P.: Lifted First-Order Belief Propagation. In: Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence. pp. 1094–1099. AAAI Press (2008)
21. Singla, P., Mooney, R.J.: Abductive Markov Logic for Plan Recognition. In: Proceedings of the 25th AAAI Conference on Artificial Intelligence. pp. 1069–1075. AAAI Press (2011)
22. Tran, S.D., Davis, L.S.: Event modeling and recognition using markov logic networks. In: 10th European Conference on Computer Vision. pp. 610–623 (2008)
23. van den Broeck, G.: On the completeness of first-order knowledge compilation for lifted probabilistic inference. In: Proceedings of the 25th Annual Conference on Neural Information Processing Systems (NIPS). pp. 1386–1394 (2011)
24. van den Broeck, G., Darwiche, A.: On the complexity and approximation of binary evidence in lifted inference. In: Proceedings of the 27th Annual Conference on Neural Information Processing Systems (NIPS). pp. 2868–2876 (2013)
25. van den Broeck, G., Davis, J.: Conditioning in first-order knowledge compilation and lifted probabilistic inference. In: Proceedings of the 26th AAAI Conference on Artificial Intelligence. AAAI Press (2012)
26. van den Broeck, G., Taghipour, N., Meert, W., Davis, J., De Raedt, L.: Lifted Probabilistic Inference by First-Order Knowledge Compilation. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence. pp. 2178–2185 (2011)
27. Venugopal, D., Gogate, V.: On lifting the gibbs sampling algorithm. In: Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS). pp. 1664–1672 (2012)
28. Yedidia, J.S., Freeman, W.T., Weiss, Y.: Generalized Belief Propagation. In: Proceedings of the 14th Annual Conference on Neural Information Processing Systems (NIPS). pp. 689–695 (2000)