Advances in Inference Methods for Markov Logic Networks

Deepak Venugopal

***Abstract*—Markov Logic Networks (MLNs) are expressive models that can be used to specify complex and uncertain background knowledge in the form of weighted first-order logic formulas. However, inference in MLNs is highly challenging since the underlying probabilistic model can be very large even for simple MLN structures. Lifted inference has emerged as dominant approach for probabilistic inference in MLNs, where the idea is to exploit symmetries in the MLN for scalable inference. In this paper, we provide an overview of MLNs, and major advances in inference techniques for MLNs over the last several years.**

***Index Terms*—Markov Logic Networks, Statistical Relational Learning, Probabilistic Graphical Models, Probabilistic Inference.**

I. Introduction

STATISTICAL Relational AI [1] unifies two corner-stones of Artificial Intelligence, namely, first-order logic and probabilities, to represent relational knowledge in the presence of uncertainty. Several notable SRL models have been proposed over the last several years including Markov Logic Networks (MLNs) [2], [3], Bayesian Logic (BLOG) [4], probabilistic soft logic (PSL) [5] and ProbLog [6]. MLNs are arguably one of the most popular models for SRL, and combine first-order logic with undirected probabilistic graphical models also known as Markov networks [7]. Specifically, an MLN is a set of first-order logic formulas with real-valued weights attached to each formula. The first-order formulas encode knowledge corresponding to an application domain, while the weights represent uncertainty associated with that knowledge. The larger the weight of a formula, greater is our belief in that formula, and vice-versa. Thus, MLNs soften the semantics of first-order logic (where formulas are either true/false). More specifically, MLNs are essentially template models that can encode different probability distributions based on the instantiations of its first-order formulas. Given the constants in a domain-of-interest, the probability distribution in an MLN is represented in factored form as a Markov network. Note that by combining the compactness of first-order logic and Markov networks, MLNs are capable of representing large, complex, uncertain relational knowledge in a succinct manner. Therefore, they have been used in diverse areas including NLP [8], computer vision [9], intelligent tutoring systems [10] and health informatics [11].

However, the expressiveness of MLNs comes at the cost of increased complexity of probabilistic inference, and consequently learning, which typically uses inference as a sub-step. Specifically, with just a few compact formulas, MLNs are able to represent extremely large Markov networks containing thousands of variables and factors. For instance, consider a simple MLN that models the transitive relationship $\texttt{Friends}\ x,$ $\texttt{Friends}\ z, x$ with weight . Every possible instantiation or *grounding* of the MLN formula for a given domain, represents a factor in the Markov network. That is, suppose we consider 1000 people in our domain, the Markov network underlying our example MLN has 1 billion factors and 1 million variables. Performing inference on MLNs is infeasible using traditional inference algorithms for graphical models, which we refer to as *propositional* methods, since they work on the Markov network representation of the first-order MLN. Thus, the challenge is to perform inference by taking advantage of the *lifted* representation in MLNs.

An interesting aspect about the MLN representation is that the number of weights in the MLN is typically much smaller than the number of factors in the underlying Markov network. In other words, all instantiations of a formula *share* the same weight. This induces symmetries in the probability distributions encoded by an MLN. Therefore, a significant amount of research in MLNs has aimed towards exploiting these symmetries to improve scalability. In particular, starting with the pioneering work by Poole [12], the predominant method for inference is the idea of *lifted inference*, which performs reasoning over groups of indistinguishable variables in the model. For example, if $\texttt{Friends}\ \mathit{Alice}, \mathit{Bob}$ and $\texttt{Friends}\ \mathit{Bob}, \mathit{Carl}$ have the same distributions, then inference results for $\texttt{Friends}\ \mathit{Alice}, \mathit{Bob}$ can be re-used for $\texttt{Friends}\ \mathit{Bob}, \mathit{Carl}$. The main challenge in developing efficient lifted inference algorithms is to efficiently compute groups of symmetric variables at a first-order level, without explicitly grounding the MLN, which could potentially create an extremely large Markov network.

The aim of this paper is to provide readers an overview of MLNs in general, and in particular, to summarize major advances in inference over the last few years. Fast and scalable inference algorithms are critical to the success of not only MLNs, but the general field of Statistical Relational AI. With a growing interest in Statistical Relational AI due to the expressiveness, and explainability of its models [13], we believe that developments in this area should be of interest to the intelligent systems community in general.

II. Background

A. First-order Logic

The language of first-order logic (cf. [14]) consists of quantifiers (and), logical variables, constants, predicates,

Deepak Venugopal is with the Department of Computer Science, University of Memphis, TN, 38152 USA e-mail: (dvngopal@memphis.edu).

and logical connectives (, , , , and). A predicate is a relation that takes a specific number of arguments as input and outputs either TRUE (synonymous with) or FALSE (synonymous with). The arity of a predicate is the number of its arguments. A first-order formula connects various predicates through the logical connectives. A first-order knowledge base (KB) is a set of first-order formulas. We denote logical variables in a KB by lower case letters (e.g., x, , z) and constants, which model objects in the real-world domain, by strings that begin with an uppercase letter (e.g., *A*, *Ana*, *Bob*).

B. Markov Logic Networks (MLNs)

One of the problems with first-order logic is that it cannot represent uncertainty, i.e., formulas are either true or false. MLNs soften the constraints expressed by each formula, by attaching a weight to it. Higher the weight, higher is our belief of the formula being satisfied, all other things being equal. In MLNs, we assume a restricted version of first-order logic with *Herbrand* semantics. Specifically, we assume that each argument of each predicate is typed and can only be assigned to a finite set of constants. By extension, each logical variable in each formula is also typed. Given a domain of constants, a *ground* atom is obtained by substituting all the variables in a predicate by constants. Similarly, a ground formula is obtained by replacing all variables in the formula with constants. A possible world, denoted by ω, is a truth assignment to all ground atoms in the MLN.

MLNs can also be seen as a first-order template for generating large Markov networks[15], [7], which is an undirected probabilistic graphical model. To illustrate MLNs, we consider the prototypical "friends-smokers" social network domain. We can represent common-sense knowledge that "smoking causes cancer" and "smokers tend to have similar smoking habits" using the following weighted formulas: (i) x `Smokes` x `Cancer` x ; and (ii) x, `Smokes` x `Friends` x, `Smokes` where and are the weights. Weights lie between and + and reflect the strength of the constraint. Positive (negative) weights represent that the worlds satisfying the formula have higher (lower) probability than worlds not satisfying the formula. MLNs generalize first-order logic in the sense that weights that are equal to infinity represent hard constraints.

Given a set of constants that represent objects in the domain (e.g. people in the social-network), the Markov network has one random variable for each grounding of each predicate (one for each instantiation of each logical variable in the predicate by a constant) and one feature for each possible grounding of each first-order formula. The weight attached to the feature is the weight attached to the corresponding first-order formula. For instance, given two constants *Ana* and *Bob*, the first first-order formula in the friends-smokers MLN yields the following two ground formulas having the same weight : (i) `Smokes` *Ana* `Cancer` *Ana* ; and (ii) `Smokes` *Bob* `Cancer` *Bob* . . Similarly, the second first-order formula with the same constants will yield four ground formulas. Formally, given a set of weighted first-order formulas f, and a set of constants, the probability of a world ω, which is a truth-assignment to all the ground atoms, is given by the following log-linear expression: $\omega = Z$ $N\ \omega$ where $N\ \omega$ is the number of groundings of f that are true in ω and Z is a normalization constant, also called the partition function.

Important inference queries in MLNs are computing the partition function, finding the marginal probability of an atom given evidence (an assignment to a subset of variables) and finding the most probable assignment to all atoms given evidence (MAP inference). All these problems are computationally intractable. Therefore, typically approximate inference algorithms are used to solve these problems in practical MLNs. In a typical use case of MLNs, the application designer writes first-order logic formulas that encode prior knowledge about the domain, and then relies on domain independent techniques implemented in software packages such as Alchemy [16] and Tuffy [17] to solve two key tasks: *probabilistic inference* – answering queries (making predictions) given the learned MLN and observations (evidence), and *weight learning* – learning the weights attached to the formulas from data. Weight learning internally uses inference within each sub-step, and therefore developing efficient inference methods is one of the key problems in MLNs.

III. Exact Lifted Inference

Lifted inference in MLNs can be viewed as the probabilistic equivalent of reasoning in first-order logic, i.e., theorem proving. Specifically, just as theorem proving does not convert first-order formulas in a knowledge base to propositional formulas but instead reasons directly on the first-order representation, lifted inference aims to perform probabilistic reasoning without creating the full Markov network from the ground formulas. The concept of *domain liftable* MLNs was introduced in [18], [19], which refers to MLN structures on which the complexity of exact inference is polynomial in the number of domain objects. Notable lifted inference algorithms that peform domain-lifted exact inference, include lifted factor-graphs [12], First-order Variable Elimination (FOVE) [20], Weighted First-Order Model Counting (WFOMC) [21] and Probabilistic theorem Proving (PTP) [22]. Next, we will briefly review PTP which is one of the most popular exact lifted inference methods for MLNs.

PTP lifts *weighted model counting* [23] to the first-order. It turns out that the weighted model counting problem is equivalent to computing the partition function of the MLN (cf. [22], [23]). PTP computes the partition functions using two lifting rules, namely, lifted decomposition and lifted conditioning. Just like the well-known DPLL algorithm [24] for SAT solving, PTP recursively applies the lifting rules on the input MLN. Below, we give a informal summary of each lifting rule, and refer the reader to [22], [25] for details.

Lifted Decomposition identifies identical and independent components in the underlying Markov network by only looking at the first-order structure. We illustrate this with a simple example. Consider the MLN , `Strong`(x) `Wins`(x). Given the domain, $\Delta = X, X$, the Markov network defined over `Strong`(X) `Wins`(X) is identical and independent of the Markov network defined over `Strong`(X) `Wins`(X). Thus, $Z = Z$ $Z = Z$.

Lifted Conditioning conditions on a first-order predicate. Conditioning removes the predicate from the MLN by creating MLNs corresponding to each possible assignment to that predicate. Clearly, if a predicate R has d ground atoms, the total number of possible assignments to the ground atoms of R is equal to . However, in some cases, it is possible to group the set of assignments such that in any group, all the MLNs that are generated by the assignments in a group are equivalent to each other, i.e., they have the same partition function. For example, consider the MLN Smokes x Friends x, Asthma x . Here, conditioning on Smokes x implicitly conditions on $^{\Delta}$ different assignments to the groundings of Smokes x . However, it turns out that, in this case, we can form Δ + groups, where any assignment within a group yields the same partition function after conditioning. The grouping can be performed based on the number of atoms among the Δ that are set to true in an assignment. We can then choose one representative from each group, condition on it, and multiply the partition function of the conditioned MLN by group-size. Note that, in general, this rule can only be applied to singleton atoms, namely, atoms whose arity is equal to 1.

In practice, apart from the two lifting rules, PTP leverages advanced SAT techniques such as unit propagation and caching to greatly improve the performance of lifted inference. For more details on these extensions, refer to [22].

IV. Approximate Lifted Inference

Exact lifted inference is highly scalable when the MLN structure has symmetries that can be exploited by algorithms such as PTP. However, as shown in [18], [19], a very restrictive set of MLNs exhibit such symmetries. Specifically, according to current complexity results, MLNs are liftable only if each formula has a maximum of two variables. Therefore, for most practical MLNs, exact inference is unlikely to scale up. Thus, several well-known propositional approximate inference algorithms have been lifted to the first-order level. Next, we will review a few notable ones.

Lifted Belief Propagation. Singla and Domingos [26] lifted belief propagation[27] in MLNs to the first-order level. Specifically, in loopy belief propagation, the MLN is encoded as a factor-graph, where the atoms are the variables, and the ground formulas are the factors. The sum-product algorithm computes the marginals of all variables in the factor graphs by passing messages between the nodes/variables and features/factors that relate the nodes. The message from nodes to features is a product of all incoming messages from other features that the node is connected to, with the variable corresponding to the summed-out from the product. Similarly, the message from a feature to a node is a product of all the messages coming into a feature from nodes connected to the feature. In Lifted Belief Propagation, the main idea is to identify messages that are identical, and send a single aggregate message instead of individual messages. To do this, Singla and Domingos proposed the creation of super-nodes and super-features, which correspond to groups of nodes and features that emit common messages. The grouping of nodes into super-nodes and features into super-features is performed incrementally by observing the messages in BP that are identical to each other.

Lifted Sampling-based Inference. In sampling-based inference methods we draw samples from the target distribution, and compute inference queries as statistics on the drawn samples. Note that in the case of MLNs, sampling worlds directly from the MLN distribution is hard, since the partition function is intractable to compute. In IS, we perform approximate inference by sampling worlds from an easier-to-sample *proposal distribution*. However, to compensate for the fact that we sampled from the approximate distribution, we weigh each sample, and compute statistics over the weighted samples. The quality of estimates from IS depends upon how close the proposal distribution is to the true distribution. Gogate et al. [28] proposed a lifted Importance Sampling (LIS) algorithm, where the main idea is to exploit symmetries to create a more-informed proposal distribution. Specifically, they grouped together symmetric worlds, and sampled a single world from each group, which consequently increases the effective sample-size, and yields lower-variance estimates of the computed inference queries. In order to create a proposal distribution which is tractable to sample from, Gogate et al. relied on lifting rules of PTP [22]. Specifically, in PTP, the lifting rules are applicable only for specific MLN structures. In LIS, the pre-conditions for applying the lifting rules are relaxed, and thus, the lifting rules are applied approximately to non-liftable MLN structures. Samples from the proposal distribution are generated by sampling from a symbolic execution trace of PTP. Further, the proposal distribution is adaptively improved based on prior samples such that the distribution moves closer to the true MLN distribution.

An alternative approach to IS, is to construct a Markov Chain whose stationary distribution is equivalent to the MLN's true distribution. We can then sample from this chain, and answer inference queries based on statistics obtained from the samples. A lifted MCMC method can be visualized as one that works in a *lifted state-space*. That is, we construct a state-space of the sampler that does not explicitly enumerate every possible state but instead groups these states based on symmetries. For instance, a propositional sampler on a MLN with n atoms works in a state-space states, while a lifted state-space can have far fewer number of states. A Markov chain defined on the lifted states can typically make larger jumps as compared to Markov chains defined on a propositional space, and in many cases larger jumps can avoid being struck in regions of local optima. Niepert [29] proposed a lifted MCMC method by grouping together states based on symmetries detected from automorphism groups computed from the MLN's graph structure. Venugopal et al. [30] lifted the blocked Gibbs sampling algorithm [31], which is an advanced variant of the Gibbs sampling [32] algorithm, which is arguably one of the most popular MCMC methods. Lifted Blocked Gibbs (LBG) partitions the atoms in the MLN into domain-liftable blocks, i.e., exact lifted inference must be tractable within each block. Further, the LBG sampler maintains a lifted state-space within each block, where the assignments to all ground atoms within a block are not stored, but sufficient statistics related to these assignments are stored.

This improves the convergence of the sampler, as well as the estimates derived from the sampler.

Lifted MAP Inference. MAP inference is an optimization problem that can be solved using popular randomized local-search solvers such as MaxWalkSAT [33]. These techniques are propositional since they work on the Markov network underlying the MLN. Sarkhel et al. [34] proposed an approach to lift such propositional MAP solvers by pre-processing the MLN, and reducing the size of its underlying Markov network. Specifically, they considered a specific subset of MLNs called *non-shared* MLNs, where no variables are shared across atoms in a formula, and showed that the MAP solution in these MLNs is independent of the number of domain objects. For example, the MLN R x S is equivalent to R S, where, if the assignment to R (or S) in the MAP solution is 1 (or 0), then all ground atoms of R x (or S x) have an assignment equal to 1 (or 0). Using this property, MAP inference on non-shared MLNs can be reduced to propositional MAP inference, where each first-order predicate is replaced by a single propositional variable, since the MAP assignment to all groundings of the predicate are symmetric to each other. Other approaches for MAP inference have lifted Linear Programming solvers based on symmetries [35].

V. Exploiting Approximate Symmetries

One of the key problems with lifted inference methods that exploit exact symmetries is that they are ineffective when the structure of the MLN is complex (e.g. transitive formula) or evidence is presented to the MLN, since evidence typically breaks symmetries. For example, consider the toy MLN given in Figure 1(a). When no evidence is present, all ground atoms of Wins x, have the same marginal probability (exact symmetry). Given evidence (see Figure 1(b)), the marginal probabilities are no longer the same as shown in Figure 1(c) and there are no exact symmetries. In such cases, lifted inference algorithms will ground the MLN, and lifted inference is almost the same as propositional inference. More generally, Broeck and Darwiche [36] showed theoretical results that, in the presence of evidence on binary or higher-arity atoms, MLNs are no longer domain-liftable. Similarly, Kersting et al. [37] showed that as the amount of *symmetry-breaking* evidence increases, the benefits of lifted inference diminishes. Unfortunately, most real-world applications require inference algorithms that can reason in the presence of evidence, and in such cases lifted inference is more or less equivalent to propositional inference. This problem can be averted using approximations which group together variables that are similar, but not identical. For example, from Figure 1(c), we can see that the marginal probabilities of the first three atoms and the last two atoms are roughly the same and they can be treated as indistinguishable for all practical purposes. Below, we highlight some specific approaches that are designed to scale up inference by exploiting approximate symmetries when exact symmetries are absent.

Over-Symmetric Approximation. Broeck and Darwiche [36] proposed the idea of *smoothing* the evidence by introducing more symmetries in the model. In this way, lifted inference methods will have a better chance of finding these symmetries. For example, if we consider the evidences on a predicate Linked, as Linked P , P , Linked P , P , Linked P , P , Linked P , P , the evidence on P and P is not symmetrical. however, by removing Linked P , P and adding Linked P , P , the evidence becomes more symmetrical. Broeck and Darwiche modeled this as a factorization problem in a boolean matrix. Specifically, binary evidence is represented as a boolean matrix, and the idea is to come up with a reduced-rank approximation of this matrix, which in-turn yields more symmetric evidence that is better suited for lifted inference. However, changing the evidence would change the MLN distribution, and therefore, inference results computed from this approximate distribution will not have strong guarantees associated with it. To obtain such guarantees, Niepert and Broeck [38] used the over-symmetric approximation to as a proposal distribution for MCMC algorithms instead of computing inference results directly from the approximate distribution.

Evidence-based Clustering. The key idea here is to pre-process the MLN by reducing the number of objects, replacing several roughly symmetric objects by a single (meta) object. We then run lifted inference using these new, much smaller set of objects. A key challenge is how to find objects that are similar to each other and thus partitioning the set of objects into symmetric subsets. To solve this problem, we defined a distance (similarity) function that takes two objects as input and outputs a number that describes how symmetric or similar the two objects are (smaller the number, greater the chance that the two objects are similar). The problem is now reduced to a standard clustering problem, and algorithms such as K-means can be used to solve it. Venugopal and Gogate [39] proposed a distance function which is based on common sense knowledge that objects having similar neighborhood constraints (Markov blanket) tend to be symmetric in the sense that the marginal probabilities of atoms containing those objects will be roughly the same. Formally, the distance function developed by Venugopal and Gogate is given by: $d\ X, X = U \quad U$ where X and X are constants (objects) that belong to the same domain equivalence class (see section 2) and $U = c \ , \ldots, c$ and $U = c \ , \ldots, c$ are m-dimensional vectors where m is the number of formulas and c is the number of groundings of the formula f that evaluate to true in the MLN obtained from the original MLN by grounding all logical variables having the corresponding object type with X and instantiating evidence. One can think of U as a feature vector describing the neighborhood of the object X in the MLN given evidence. Since computing the number of groundings is a #P-hard problem, the approach by Venugopal and Gogate proposed to approximate the counts by decomposing large formulas into smaller ones. However, one of the major problems with using clustering methods such as K-Means is that the optimal number of clusters is hard to compute. For instance, for some domains with greater symmetry among objects, a small set of meta-objects may suffice, while for other domains, we may require more meta-objects. Venugopal et al. [40] extended the aforementioned approach using a non-parametric clustering method called DP-Means [41], where they computed the optimal number of

Wins(A,A)	0.56
Wins(A,B)	0.56
Wins(A,C)	0.56
Wins(B,A)	0.56
Wins(B,B)	0.56
Wins(B,C)	0.56
Wins(C,A)	0.56
Wins(C,B)	0.56
Wins(C,C)	0.56

(a) Original Marginals

Strong(C)
Wins(A,C)
Wins(B,B)
Wins(B,C)
Wins(C,A)

(b) Evidence

Wins(A,A)	0.6
Wins(A,B)	0.6
Wins(B,A)	0.63
Wins(C,B)	0.85
Wins(C,C)	0.85

(c) New Marginals

Fig. 1. Effect of evidence on an MLN with one formula, Strong( ) Wins(,). The marginal probabilities which were equal in (a) become unequal in (c) due to evidence (b).

clusters for a given bound on the error in samples generated from the approximated MLN.

Apart from the above approaches, other methods have also been proposed for exploiting approximate symmetries in specific inference tasks. Specifically, Sarkhel et al. [42] proposed a refinement approach for MAP inference by adding equality constraints to the MLN when objects are approximately symmetric to each other. For the marginal inference problem, Singla et al. [43] considered approximately similar messages in belief propagation as equivalent messages, and constructed a lifted belief network that is much smaller than the lifted network constructed from exactly symmetric messages.

VI. Exploiting MLN Structure

Since MLNs are deined as logical formulas, propositional inference algorithms can use approaches that exploit MLN structure to avoid constructing the ground Markov network during inference. Along these lines, Shavlik and Natarajan [44] proposed an approach called FROG (Fast Reduction Of Grounded networks). The idea was to pre-process the MLN, and reduce the size of the ground network by efficiently computing formulas that are satisfied due to the logical structure. For example, in R x, S , z T z, x , if we know that majority of the groundings of R x, are false, then majority of the formula groundings are true irrespective of the states of the groundings of S , z and T , z . FROG maintains a representation of these non-satisfied formula groundings, and just stores statistics on the satisfied groundings, which is sufficient for performing inference. Similarly, Poon and Domingos [45] developed an approach to perform *lazy grounding* in MLNs. Here, only a subset of variables and ground formulas are kept in memory, and as inference proceeds, grounding is performed as-needed. The main idea behind this approach is that, many propositional inference algorithms work on one variable of the MLN at a time, and updates to the variable depends upon a small subset of other variables and formulas. More recently, Venugopal et al. [46] proposed an approach to scale up local-search algorithms such as MaxWalkSAT and sampling-based inference algorithms such as Gibbs sampling using efficient counting algorithms. Specifically, a counting problem that MaxWalkSAT or Gibbs Sampling needs to solve is, counting the satisfied groundings of a first-order formula f, given a world ω. This problem is known to be computationally hard [3], and inference algorithms need to solve this problem not just once but several times over thousands of iterations. Venugopal et al. encoded the counting problem as a problem of computing the partition function of a graphical model. Specifically, given that f has k variables, they encoded a graphical model with k nodes, and derived the factors in the graphical model from ω. Importantly, if the *tree-width* of the encoded graphical model is small, then the counting can be performed exactly using methods such as junction-trees [47]. For larger treewidth models, off-the-shelf algorithms such as generalized belief propagation [27] to approximate the counts in a scalable manner.

VII. Joint Inference Applications

MLNs have been used extensively to model joint inference tasks in complex problems. As compared to Integer Linear Prgramming (ILP) formulations which were previously used to model joint inference [48], the first-order structure of MLNs helps us model joint dependencies more compactly. Singla and Domingos [49] developed one of the earliest MLN-based joint inference models for the entity resolution task on the cora and bibserv citation datasets. Poon and Domingos [8] developed an MLN model for information extraction utilizing entity resolution within the model, to jointly segment citation fields in the cora and bibserv datasets. Poon and Domingos [50] also developed an unsupervised model for joint coreference resolution, where they used a predicate to specify clustering of mentions that correspond to a common entity. Poon and Vanderwende [51] developed a joint model for Biomedical event extraction on the Genia dataset, where they detected triggers and arguments jointly. Venugopal et al. [52] further improved the performance of MLNs on the same event extraction task by leveraging linguistic features. Specifically, encoding such features directly as MLN formulas makes learning and inference infeasible due to their high-dimensionality, which results in a large number of ground formulas. To add such features to the MLN, Venugopal et al. used the output of SVMs learned from the high-dimensional features to generate priors for the MLN distribution. Lu et

al. [53] designed an MLN for event coreference resolution that jointly performs four tasks in the pipeline: trigger identification and subtyping, argument identification and role determination, entity coreference resolution and event coreference resolution. Similar to Venugopal et al.'s strategy, Lu et al. used an SVM to incorporate high-dimensional features into their MLN.

VIII. Discussion and Conclusion

Though plenty of progress has been made in MLNs over the last several years, there is quite a lot of work that needs to be done to make MLNs a "black-box" for application designers. The blow-up in the size of the probabilistic model, with increased data-size makes inference and learning very hard, and therefore application designers find it hard to use MLNs using existing open-source systems MLNs as-is. Particularly, if MLNs are to work with *big-data problems*, inference algorithms need to become far more scalable than current state-of-the-art. Further, explaining the results of inference is an area that requires active future research in MLNs. Due to their basis in first-order logic, MLNs seem to be a promising candidate to develop interpretable Machine learning models. Integrating MLNs with other models such as deep-models in order to facilitate relational deep learning is also an area where future research seems to be headed.

In this paper, we provided a brief overview of MLNs and major advances in MLN inference methods. Particularly, the idea of lifted inference has received a lot of interest from the research community, where symmetries (both exact and approximate) in the MLN distribution are exploited to perform scalable inference. As MLNs continue to be applied in varied domains, advances in this area should be of interest to the field of Statistical Relational AI, and the Intelligent Systems community in general.

References

[1] L. Getoor and B. Taskar, Eds., *Introduction to Statistical Relational Learning*. MIT Press, 2007.

[2] M. Richardson and P. Domingos, "Markov Logic Networks," *Machine Learning*, vol. 62, pp. 107–136, 2006.

[3] P. Domingos and D. Lowd, *Markov Logic: An Interface Layer for Artificial Intelligence*. San Rafael, CA: Morgan & Claypool, 2009.

[4] B. Milch, B. Marthi, and S. Russell, "BLOG: Relational Modeling with Unknown Objects," in *Proceedings of the ICML-2004 Workshop on Statistical Relational Learning and its Connections to Other Fields*. Banff, Canada: IMLS, 2004, pp. 67–73.

[5] S. H. Bach, M. Broecheler, B. Huang, and L. Getoor, "Hinge-loss markov random fields and probabilistic soft logic," *Journal of Machine Learning Research*, vol. 18, pp. 109:1–109:67, 2017.

[6] L. De Raedt, A. Kimmig, and H. Toivonen, "ProbLog: A Probabilistic Prolog and Its Application in Link Discovery," in *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 2007, pp. 2462–2467.

[7] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[8] H. Poon and P. Domingos, "Joint Inference in Information Extraction," in *Proceedings of the 22nd National Conference on Artificial Intelligence*. Vancouver, Canada: AAAI Press, 2007, pp. 913–918.

[9] S. D. Tran and L. S. Davis, "Event modeling and recognition using markov logic networks," in *ECCV*, 2008.

[10] D. Venugopal and V. Rus, "Joint inference for mode identification in tutorial dialogues," in *COLING 2016, 26th International Conference on Computational Linguistics*, 2016, pp. 2000–2011.

[11] S. Natarajan, V. Bangera, T. Khot, J. Picado, A. Wazalwar, V. S. Costa, D. Page, and M. Caldwell, "Markov logic networks for adverse drug event extraction from text," *Knowl. Inf. Syst.*, vol. 51, no. 2, pp. 435–457, 2017.

[12] D. Poole, "First-Order Probabilistic Inference," in *Proceedings of the 18th International Joint Conference on Artificial Intelligence*. Acapulco, Mexico: Morgan Kaufmann, 2003, pp. 985–991.

[13] L. D. Raedt, K. Kersting, S. Natarajan, and D. Poole, *Statistical Relational Artificial Intelligence: Logic, Probability, and Computation*, ser. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2016.

[14] M. R. Genesereth and E. Kao, *Introduction to Logic, Second Edition*. Morgan & Claypool Publishers, 2013.

[15] A. Darwiche, *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2009.

[16] S. Kok, M. Sumner, M. Richardson, P. Singla, H. Poon, D. Lowd, J. Wang, and P. Domingos, "The Alchemy System for Statistical Relational AI," Department of Computer Science and Engineering, University of Washington, Seattle, WA, Tech. Rep., 2008, http://alchemy.cs.washington.edu.

[17] F. Niu, C. Ré, A. Doan, and J. W. Shavlik, "Tuffy: Scaling up statistical inference in markov logic networks using an rdbms," *PVLDB*, vol. 4, no. 6, pp. 373–384, 2011.

[18] G. V. den Broeck, "On the completeness of first-order knowledge compilation for lifted probabilistic inference," in *NIPS*, 2011, pp. 1386–1394.

[19] M. Jaeger, "Lower Complexity Bounds for Lifted Inference," *CoRR*, vol. abs/1204.3255, 2012.

[20] R. de Salvo Braz, "Lifted First-Order Probabilistic Inference," Ph.D. dissertation, University of Illinois, Urbana-Champaign, IL, 2007.

[21] G. Van den Broeck, N. Taghipour, W. Meert, J. Davis, and L. De Raedt, "Lifted Probabilistic Inference by First-Order Knowledge Compilation," in *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, 2011, pp. 2178–2185.

[22] V. Gogate and P. Domingos, "Probabilistic Theorem Proving," in *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 2011, pp. 256–265.

[23] M. Chavira and A. Darwiche, "On probabilistic inference by weighted model counting," *Artificial Intelligence*, vol. 172, no. 6-7, pp. 772–799, 2008.

[24] M. Davis and H. Putnam, "A Computing Procedure for Quantification Theory," *Journal of the Association of Computing Machinery*, vol. 7, no. 3, 1960.

[25] A. Jha, V. Gogate, A. Meliou, and D. Suciu, "Lifted Inference from the Other Side: The tractable Features," in *Proceedings of the 24th Annual Conference on Neural Information Processing Systems (NIPS)*, 2010, pp. 973–981.

[26] P. Singla and P. Domingos, "Lifted First-Order Belief Propagation," in *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*. Chicago, IL: AAAI Press, 2008, pp. 1094–1099.

[27] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Generalized Belief Propagation," in *Advances in Neural Information Processing Systems 13*, T. Leen, T. Dietterich, and V. Tresp, Eds. Cambridge, MA: MIT Press, 2001, pp. 689–695.

[28] V. Gogate, A. Jha, and D. Venugopal, "Advances in Lifted Importance Sampling," in *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[29] M. Niepert, "Markov chains on orbits of permutation groups." in *UAI*. AUAI Press, 2012, pp. 624–633.

[30] D. Venugopal and V. Gogate, "On lifting the gibbs sampling algorithm," in *Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS)*, 2012, pp. 1664–1672.

[31] C. S. Jensen, U. Kjaerulff, and A. Kong, "Blocking Gibbs Sampling in Very Large Probabilistic Expert Systems," *International Journal of Human Computer Studies. Special Issue on Real-World Applications of Uncertain Reasoning*, vol. 42, pp. 647–666, 1993.

[32] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 721–741, 1984.

[33] B. Selman, H. Kautz, and B. Cohen, "Local Search Strategies for Satisfiability Testing," in *Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge*, D. S. Johnson and M. A. Trick, Eds. Washington, DC: American Mathematical Society, 1996, pp. 521–532.

[34] S. Sarkhel, D. Venugopal, P. Singla, and V. Gogate, "Lifted MAP inference for Markov Logic Networks," *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics (AISTATS-14)*, 2014.

[35] U. Apsel, K. Kersting, and M. Mladenov, "Lifting relational map-lps using cluster signatures," in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014, pp. 2403–2409.
[36] G. van den Broeck and A. Darwiche, "On the complexity and approximation of binary evidence in lifted inference," in *Advances in Neural Information Processing Systems 26*, 2013, pp. 2868–2876.
[37] K. Kersting, B. Ahmadi, and S. Natarajan, "Counting Belief Propagation," in *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, 2009, pp. 277–284.
[38] G. Van den Broeck and M. Niepert, "Lifted probabilistic inference for asymmetric graphical models," in *Proceedings of the 29th Conference on Artificial Intelligence (AAAI)*, 2015.
[39] D. Venugopal and V. Gogate, "Evidence-based clustering for scalable inference in markov logic," in *ECML PKDD*, 2014.
[40] D. Venugopal, S. Sarkhel, and K. Cherry, "Non-parametric domain approximation for scalable gibbs sampling in mlns," in *Proceedings of the 32nd Conference on Uncertainty In Artificial Intelligence*, 2016, pp. 745–755.
[41] B. Kulis and M. I. Jordan, "Revisiting k-means: New algorithms via bayesian nonparametrics," *ICML*, 2012.
[42] S. Sarkhel, D. Venugopal, T. A. Pham, P. Singla, and V. Gogate, "Scalable training of markov logic networks using approximate counting," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, 2016, pp. 1067–1073.
[43] P. Singla, A. Nath, and P. Domingos, "Approximate Lifting Techniques for Belief Propagation," in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014, pp. 2497–2504.
[44] J. W. Shavlik and S. Natarajan, "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*, 2009, pp. 1951–1956.
[45] H. Poon, P. Domingos, and M. Sumner, "A General Method for Reducing the Complexity of Relational Inference and its Application to MCMC," in *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, 2008, pp. 1075–1080.
[46] D. Venugopal, S. Sarkhel, and V. Gogate, "Just count the satisfied groundings: Scalable local-search and sampling based inference in mlns," in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.
[47] S. L. Lauritzen and D. J. Spiegelhalter, "Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 50, no. 2, pp. 157–224, 1988.
[48] D. Roth and W. Yih, "Integer Linear Programming Inference for Conditional Random Fields," in *Proceedings of the Twenty-Second International Conference on Machine Learning*, 2005, pp. 736–743.
[49] P. Singla and P. Domingos, "Memory-Efficient Inference in Relational Domains," in *Proceedings of the Twenty-First National Conference on Artificial Intelligence*. Boston, MA: AAAI Press, 2006, this volume.
[50] H. Poon and P. Domingos, "Joint Unsupervised Coreference Resolution with Markov Logic," in *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*. Honolulu, HI: ACL, 2008, pp. 649–658.
[51] H. Poon and L. Vanderwende, "Joint inference for knowledge extraction from biomedical literature." in *HLT-NAACL*, 2010.
[52] D. Venugopal, C. Chen, V. Gogate, and V. Ng, "Relieving the computational bottleneck: Joint inference for event extraction with high-dimensional features," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. ACL, 2014, pp. 831–843.
[53] J. Lu, D. Venugopal, V. Gogate, and V. Ng, "Joint inference for event coreference resolution," in *COLING 2016, 26th International Conference on Computational Linguistics*, 2016, pp. 3264–3275.