Focused Grounding for Markov Logic Networks

Michael Glass
Department of Computer Science
University of Texas at Austin
mrglass@cs.utexas.edu

Ken Barker
IBM Watson Research Lab
kjbarker@us.ibm.com

Abstract

Markov logic networks have been successfully applied to many problems in AI. However, the computational complexity of the inference procedures has limited their application. Previous work in lifted inference, lazy inference and cutting plane inference has identified cases where the entire ground network need not be constructed. These approaches are specific to particular inference procedures, and apply well only to certain classes of problems. We introduce a method of focused grounding that can use either general purpose or domain specific heuristics to produce only the most relevant ground formulas. Though a solution to the focused grounding is not, in general, a solution to the complete grounding, we show empirically that the smaller search space of a focused grounding makes it easier to locate a good solution. We evaluate focused grounding on two diverse domains, joint entity resolution and abductive plan recognition. We show improved results and decreased computation cost for the entity resolution domain relative to a complete grounding. Focused grounding in abductive plan recognition produces state of the art results in a domain where complete grounding proved intractable.

Introduction

In recent years formalisms for probabilistic reasoning have found applications in a wide variety of domains. Markov logic networks (MLNs) (Richardson and Domingos 2006) have been applied to entity resolution (Singla and Domingos 2006a), information extraction (Poon and Domingos 2007) and even unsupervised semantic parsing (Poon and Domingos 2009). The ability to consider multiple sources of probabilistic evidence jointly is clearly important in AI (Domingos 2006). But the computational cost of doing so can be prohibitive. The standard inference methods for MLNs all require the network to be grounded. That is, variables in the first order formulas must be replaced with terms from the Herbrand universe. The time required for exact inference is exponential in the number of ground predicates, or atoms. And the number of ground clauses in a complete grounding is exponential in the highest arity first order formula. To address this problem, we use focused grounding to produce a limited set of ground formulas. This permits inference in cases where grounding the entire network would be infeasible. Additionally, inference over this smaller grounding is faster, and may more easily locate a good solution.

Background

Markov Logic Networks are a probabilistic extension of first order logic. An MLN is a set of weighted formulas in first order logic over a finite domain of objects. This formalism allows two types of inference: marginal and Maximum A Posteriori (MAP).

Marginal Inference

Marginal inference can be used to find the probability of individual ground query predicates.

$$P(x) = \frac{1}{Z} \exp\left(\sum_i w_i n_i(x)\right).$$

$P(x)$ is the probability of the truth assignment x, w_i is the weight of the ith formula and $n_i(x)$ is the number of satisfied groundings of formula i given the state x. Z is a normalization constant. Though an exact solution is #P-complete (Roth 1996), sampling methods such as Gibbs and MC-SAT provide a tractable approximation of marginal inference (Poon and Domingos 2006).

MAP Inference

Maximum A Posteriori (MAP) inference finds the most likely possible world $\hat{\mathbf{x}}$. This is the world in which the weight of all the satisfied ground clauses is maximized.

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} \left(\sum_i w_i n_i(\mathbf{x})\right).$$

In cases where there are multiple maximums, any can be returned. Finding the MAP solution is a case of the weighted satisfiability problem, known to be NP-Hard (Roth 1996). It may be solved by conversion to Integer Linear Programming (Taskar 2005), which when computationally tractable, will return an exact solution. It may also be approximated by a local search method, such as MaxWalkSat (Selman, Kautz, and Cohen 1996).

Herbrand Base

The Herbrand Universe H for an MLN M is the set of all terms that can be constructed from the constants and functions in M. If there are no functions, it is simply C, the set of all constants. The Herbrand Base $B(M)$ is often defined as the set of all ground predicates (atoms) that can be constructed using the predicates in M and the terms in H. Here we are more concerned with all the ground formulas that can be constructed from the formulas in M and the terms in H, we call this the Herbrand formula base F_{HB}. The size of the Herbrand formula base is exponential in the number of variables v_i in a formula i.

$$|F_{HB}(M)| = \sum_i v_i^{|H|}.$$

Complete Grounding

Consider the following rule taken from an MLN from the Cora domain about entity resolution for citations (Singla and Domingos 2006a).

$$\begin{aligned} -2.695\ Venue(bc1, v1) \wedge Venue(bc2, v2) \wedge \\ HasWord\_workshop\_Venue(v1) \wedge \\ \neg HasWord\_workshop\_Venue(v2) \Rightarrow \\ SameBib(bc1, bc2) \end{aligned}$$

This rule states that if the venue for one citation contains the word "workshop" and another citation has a venue without that word, then the two citations are probably not the same. There are four variables in this clause so its contribution to the Herbrand formula base is $|C|^4$ groundings, where C is the union of all the venues, authors, titles and citations. In one fifth of the Cora dataset, this is 436. So a completely naïve grounding would result in 36 billion groundings for this one formula. Fortunately, the state of the art is well beyond this naïve approach to grounding. Alchemy (Domingos et al. 2006), a state of the art system for learning and inference in MLNs, implements two domain general controls on grounding and allows for domain specific controls.

First, this MLN is typed. In this domain there is no doubt about what is a venue, title, author or citation, so all predicates have hard type constraints. The $Venue$ predicate relates a citation to its venue. So the number of groundings with typing for this formula is only $|C_{citation}|^2 \cdot |C_{venue}|^2$.

Another control reduces the number of groundings even further. $Venue$ and $HasWord\_workshop\_Venue$ are *evidence predicates* - all the true groundings for these predicates are given as input. If $v1$ is not a venue with "workshop" in it or $v2$ is, the formula is trivially satisfied and need not be grounded. Therefore the number of groundings for this formula is only the number of venues with the word "workshop" times the number of venues without the word "workshop". In this domain, the only non-evidence predicate is $SameBib$.

The final control on the number of groundings for this formula comes from a pre-clustering outside of the MLN. This is a domain specific method and requires a partial solution to the entity resolution problem apart from the MLN. This "umbrella" clustering (McCallum, Nigam, and Ungar 2000) gives a set of plausible $SameBib$ groundings. Groundings not present in this set can be assumed false with high confidence. So the number of groundings can be further reduced by eliminating all groundings where $bc1$ and $bc2$ are not in the same umbrella cluster.

We refer to the set of formulas grounded with these optimizations as the complete, or full grounding. This set is much smaller than the Herbrand formula base. But even with all of these controls on the number of groundings, attempting to ground the network for one fifth of the Cora citation dataset requires more than 2 GB of memory. For one tenth of the dataset, the memory required is 630 MB.

Focused Grounding

In order to address the problems of complete grounding, we developed a method of focused grounding. Our method grounds the most relevant clauses first, then successively more, until a memory or time limit is reached, or until no further groundings score above a threshold. The algorithm also eliminates trivial groundings, and simplifies others, using a set of ground predicates that have known truth values, including evidence predicates and the results of any domain specific pre-processing. Formulas' groundings may be scored for relevance using either domain general or domain specific heuristics.

A key universal constraint is that all formula groundings must be motivated by ground predicates present in the set of current ground predicates, GP. This set initially contains only the evidence. So all formula groundings are directly or indirectly driven by the evidence.

The GROUNDOUT function is the key inner loop of the focused grounding method. It is called for each first order formula, f in the MLN, with an initially empty set of variable bindings b_f. It operates by selecting each predicate in the formula and attempting to unify it with entries in the GP index. The unifiable ground predicates provide bindings for the variables in the formula. The recursion terminates if the formula grounding scores below a threshold (SCORE $<$ MINSCORE), all predicates are processed without a successful grounding (SELECTPREDICATE returns $\perp$), or when all variables are bound and the formula is added to the set of ground formulas GF, and the ground predicates in the formula are added to GP. The notation $p[b_f]$ means the predicate p with its variables replaced by the bindings b_f.

The SCORE function is responsible for providing a low score whenever it is clear that the formula is either trivial (true or false in all possible worlds) or that it will not satisfy the relevancy heuristic for the domain. The SELECTPREDICATE function selects a good predicate with which to continue grounding. We use a simple rule: selecting the predicate with the fewest groundings. If all predicates are marked as missing or complete ($p = \perp$), then grounding terminates.

Working through the example formula, first the base cases are considered. On the first call, the formula is neither below threshold nor fully ground, so a predicate is selected to look up in the GP index. Since initially only evidence predicates are in the GP index, $SameBib(bc1, bc2)$ has zero groundings. It is marked as

Algorithm 1 The focused grounding algorithm

```
GROUNDOUT($f, b_f$)
 1  if SCORE($f, b_f$) < MINSCORE
 2      then return
 3  if ISGROUND($f[b_f]$)
 4      then $GF \leftarrow GF \cup \{f[b_f]\}$
 5              for each $p$ marked missing $\in f$
 6                  do $GP \leftarrow GP \cup \{p[b_f]\}$
 7              return
 8  $p \leftarrow$ SELECTPREDICATE($f$)
 9  if $p = \perp$
10      then return
11  $p$ is marked as complete
12  $B_p \leftarrow$ all possible unifications of $p[b_f]$ with $GP$
13  for each $b_p \in B_p$
14      do GROUNDOUT($f, b_f \cup b_p$)
15  $p$ is marked as missing
16  GROUNDOUT($f, b_f$)
```

missing and GROUNDOUT recurses. Suppose the third predicate, $HasWord\_workshop\_Venue(v1)$ is selected next. From the evidence, all venues with the word "workshop" present are found. Each of these provides a constant to bind to $v1$ and the loop recursively calls our grounding procedure with the predicate marked as completed, and the bindings $\{v1 \rightarrow SomeVenue\}$. Eventually all variables are bound. Then the formula with the bindings substituted for the variables, $f[b_f]$, is added to the ground formula set GF. The missing predicate $SameBib(bc1, bc2)[b_f]$ is also added to the ground predicate index.

The relevancy heuristic is very important in focusing the grounding. A basic, domain general heuristic is to prefer formulas with the fewest missing predicates. This heuristic is motivated by the fact that any formula in conjunctive normal form (CNF) containing a ground predicate appearing nowhere else can be trivially satisfied by setting that ground predicate to whatever truth value is needed to make the formula true. For purposes of counting missing predicates, all closed world ground evidence predicates are considered present. This is the heuristic used in the Cora entity resolution domain.

Another heuristic uses the GP index as a proxy for things that are plausible, and grounds only those formulas where no negated predicates are marked as missing. The rest of the formula can be seen as plausibly true, given that the negated portions are plausibly true. We use this heuristic in a plan recognition domain since some abductive formulas contain many unnegated predicates, and therefore are likely to have many missing predicates.

A separate heuristic, apart from the SCORE function, is present in the outer loop of focused grounding, which simply calls GROUNDOUT for every formula in the knowledge base on each iteration. On the first pass through the first order formula base, only those formulas directly motivated by the evidence are ground. On the nth pass, the formula groundings are at most n steps removed from the evidence.

In addition to this main phase of grounding, where formulas are grounded in approximate order of relevancy, there are two other phases: an initial phase and a final phase.

Because known truth is so valuable in limiting the number of groundings, focused grounding begins with an initial phase in which hard formulas (those with infinite weight) are grounded using predicates that have a known truth. If the result of the simplified formula is a single literal (a negated or unnegated ground predicate), it is added to the known truth table.

A final phase ensures that the most important formulas are grounded. Without this phase the last formula grounded may introduce a new ground predicate. Some other formula may specify as a hard constraint that atoms of this type are mutually exclusive with some other previously grounded atom. But this formula is not grounded with the newly introduced atom, so the result of MAP inference may output both atoms as true. The final phase can be configured based on domain specific heuristics. Currently the only heuristic used is that hard clauses with no missing predicates must be grounded, and clauses with only a single predicate (specifying priors) must be grounded.

Experimental Evaluation

To test the effects of focused grounding on efficiency and correctness, we considered two diverse domains: entity resolution in citations and plan recognition for emergency responses. In each case there were existing probabilistic knowledge bases specifying the domain knowledge and the associated probabilities.

Cora Entity Resolution

Entity resolution is the task of determining for each pair of mentions whether they refer to the same real world entity. In cases where the mentions have structure, like citations, performance on the entity resolution problem can be improved by jointly matching whole records and each component of the records, such as authors, titles and venues. Along with axioms for transitive closure this ensures that each matching decision is influenced by the other relevant matching decisions (Singla and Domingos 2006a).

The Cora dataset contains clusters of citations that all refer to the same paper. There are approximately 1300 citations referring to about 130 distinct papers. The Cora dataset is divided into five parts for five fold cross validation. We further divided each fold, splitting it into two pieces. Each of the citation clusters was randomly assigned to one of the splits, subject to the constraint that the largest split is within a factor of two of the smallest split.

We used MLNs trained on the MLN(B+N+C+T) model, the best performing and one of the most complex models described by Singla and Domingos (2006a). The model includes axioms based on naive Bayes, reverse predicate equivalence, transitive closure, and word mention equivalence using n-grams.

Monroe Plan Recognition

The Monroe dataset is an artificial domain constructed to aid in research on plan recognition (Blaylock and Allen

$$\begin{aligned}
&crew\_get\_to(x_1, x_2) \Rightarrow \\
&\quad fix\_power\_line(x_2) \wedge power\_crew(x_1) \vee \\
&\quad \exists\, loc\ (clean\_up\_hazard(x_2, loc) \wedge hazard\_team(x_1)) \vee \\
&\quad \exists\, loc\ (repair\_pipe(x_2, loc) \wedge water\_crew(x_1)) \vee \\
&\quad \exists\, loc\ (set\_up\_cones(x_2, loc) \wedge work\_crew(x_1)) \vee \\
&\quad \exists\, loc\ (take\_down\_cones(x_2, loc) \wedge work\_crew(x_1)) \vee \\
&\quad \exists\, t\ (clear\_tree(t) \wedge tree\_crew(x_1) \wedge tree(t) \\
&\qquad \wedge object\_atloc(t, x_2)) \vee \\
&\quad \exists\, b\ (remove\_blockage(b) \wedge work\_crew(x_1) \\
&\qquad \wedge object\_atloc(b, x_2)) \vee \\
&\quad \exists\, person\ (emt\_treat(person) \wedge emt\_crew(x_1) \\
&\qquad \wedge person\_atloc(person, x_2))
\end{aligned}$$

Figure 1: The abductive rule for $crew\_get\_to$

2005a). The Monroe knowledge base, built by Raghavan and Mooney (2010), consists of Horn clauses for forward chaining one of ten top level emergency response plans such as fix_water_main. The test data consists of 1000 automatically generated example plan observations, with an average of about 10 observations per example. We used the first twenty as a development set and tested on the next 500. The plan recognition task is: given the observations, identify the top level plan predicate, called the schema, and its parameters.

Alone, the forward planning Horn clauses such as $clean\_up\_hazard(from, to) \wedge hazard\_team(crew) \Rightarrow crew\_get\_to(crew, from)$ in the Monroe dataset do not support abductive plan recognition in an MLN. Reversed formulas must also be added (Kate and Mooney 2009). These formulas are automatically constructed from the forward rules by gathering all possible antecedents that result in a given plan action. The abductive rule is constructed by taking the plan action as the left hand side and the disjunction of all the gathered antecedents as the right hand side. Informally, if a plan action takes place, then one of the higher level plans that causes that plan action is likely to be true.

Critically, the variables appearing in the antecedent of the original planning rule that do not also appear in the consequent must be existentially quantified. When there are many possible higher level plans that can result in a specific lower level plan, and these higher level plans have variables not present in the lower level plan predicate, this conversion can result in formulas with many existentially quantified variables.

Figure 1, the abductive rule for $crew\_get\_to$ shows that there are many reasons for a crew to go to a location in the Monroe planning domain. The multiple existential variables introduced create an exponential growth in the memory requirements for a full grounding of the MLN. This makes MLNs with a complete grounding unsuitable for this domain (Raghavan and Mooney 2010). The complete grounding consumes more than 4 GB of memory.

We added a large negative prior for every top level plan, and a hard rule specifying that there exists some top level plan. Together, these ensured that one and only one plan was abduced.

Results

For the Cora dataset, split into ten pieces, we found dramatic improvements in time, memory and accuracy using focused grounding. The full grounding was performed by Alchemy. On average, the full grounding produced 2.6 million clauses, while we limited our focused grounding to 300 thousand clauses. Both our focused grounding system and Alchemy used MaxWalkSat with one million flips to find the MAP solution. We ignore the large number of false query atoms, citations and citation fields that are different, and report precision and recall over the true query atoms.

	Full MAP	300k Limit
Precision	53.3%	**76.6%**
Recall	21.7%	**87.9%**
F-Score	30.9%	**81.9%**
Total Time	1494 s	**211 s**
Memory	636 MB	**248 MB**

Table 1: Results for different grounding systems on the Cora entity resolution task

Compared to MAP inference over the complete grounding, focused grounding produced better precision, recall, run time and memory usage. We report detailed results on focused grounding only for the limit size of 300 thousand. However, the method proved largely insensitive to the specific number of ground formulas allowed. With one third as many groundings the F-Score was 79.1% and with twice as many the F-Score was 80.9%.

Using Alchemy to perform marginal inference on the complete grounding worked considerably better. Marginal inference approximates the probability of each query atom. By considering every atom above some probability threshold p_t to be true, we can produce a set of predictions and evaluate its precision and recall. Though our method of focused grounding can apply to marginal inference as well, we only implemented MAP inference with MaxWalkSat. Table 2 compares the marginal inference of Alchemy (MCMC) to the MAP inference of our focused grounding system.

	Full Marginal $p_t = 0.5$	300k Limit MAP
Precision	**96.5%**	76.6%
Recall	58.4%	**87.9%**
F-Score	72.8%	**81.9%**
Total Time	9824 s	**211 s**
Memory	763 MB	**248 MB**

Table 2: Results for marginal inference on the Cora entity resolution task

The probability threshold of $p_t = 0.5$ means that we regarded citation and citation fields as the same when the inference procedure reported that they refer to the same entity with at least 50% probability. We selected this threshold to facilitate a comparison between marginal and MAP inference, although the highest F-Score of 94.4% was achieved at the much lower threshold of $p_t = 0.0001$.

In the Monroe domain, we use the metrics of schema convergence, $SchemaConv$, and parameter convergence, $ParamConv$, following Blaylock and Allen (2005b). Schema convergence refers to the percent of top level plans correctly predicted after considering all observations. Parameter convergence refers to the average number of correctly predicted plan parameters for those cases where the schema prediction was correct. Because some sequences of observations do not contain a mention of the correct parameter, and this is the only source of information for the plan recognizer, there is a ceiling of feasible parameter recognition. Therefore we also report parameter convergence divided by the feasible performance ceiling, $AdjParamConv$.

	600k Limit	Blaylock and Allen
$SchemaConv$	94.6%	94.2%
$ParamConv$	**59.6%**	40.6%
$AdjParamConv$	**75.9%**	51.4%

Table 3: Results for focused grounding on the Monroe dataset, compared to results reported by Blaylock and Allen

We achieved similar performance to Blaylock and Allen's system for instantiated plan recognition on schema recognition. The 0.4% difference is not statistically significant. We found substantially better performance on feasible parameter recognition, with a 50% error rate reduction. Using the same knowledge base, a Bayesian Abductive Logic Program (Raghavan and Mooney 2010), achieved 98.8% on schema convergence. Parameter recognition results were not reported.

Related Work

Other research has addressed the computational complexity of grounding and performing inference over the Herbrand formula base.

Lifted inference attempts to avoid many groundings for objects about which identical facts are known. It is possible to construct examples where this method produces exponential savings in time and memory, while preserving correctness (Braz, Amir, and Roth 2005). However, in cases where all objects are distinguishable, the gains from traditional lifted inference are sharply reduced (de Salvo Braz et al. 2009). Lifted inference does not change the optimal solution but is not applicable in all cases. Focused grounding reduces the size of the ground network for arbitrary MLNs, at the cost of possibly changing the optimal solution.

Culotta and McCallum (2005) noted that representing certain aggregate features important in entity resolution, such as the number of distinct strings that are first names for a person, can result in an exponential explosion in the number of groundings. Their method of incremental grounding creates ground predicates as needed for networks too large to instantiate completely. FACTORIE (McCallum, Schultz, and Singh 2009) uses imperatively defined factor graphs to perform marginal inference while only keeping a fraction of the network in memory at any given time. In fact, only a single possible world is represented at a time. The factor graph is described by programmatically specifying how to transition from each possible world to each nearby possible world. In contrast we retain the declarative nature of MLNs.

LazySat (Singla and Domingos 2006b) is a modification of MaxWalkSat to avoid grounding formulas that are never unsatisfied during inference. In many domains the conjunctive normal form (CNF) clauses contain at least one negated predicate and most atoms are false. Therefore most formulas are satisfied, and need not be grounded. LazySat uses this fact by initially assuming only those atoms present in the evidence are true and all others are false. Clauses not satisfied by this assignment become active and are grounded. Unlike our work there is no limit to the number of clauses that may be grounded.

Cutting Plane Inference (CPI) (Riedel 2008) proceeds from some initial grounding and incrementally grounds only those formulas that are violated by the current solution. It can be seen as an extension of LazySat in that formulas are only grounded when they are unsatisfied. But rather than working the grounding into the inference algorithm itself, CPI interleaves the two, allowing it to be applied to any MAP solver. Like our method, CPI produces some initial approximate grounding. This grounding must be engineered to avoid both an empty grounding or a grounding that consumes more memory than is available. One possible use case for focused grounding is to provide an initial grounding for CPI that avoids these two failure modes. Also like our method, CPI can produce approximate groundings by stopping at some iteration before all unsatisfied formulas are grounded. However, focused grounding can use domain specific heuristics to select the most suitable approximate grounding.

These methods take the form of creating groundings on an as needed basis while preserving the exact semantics of the MLN. However, because randomized, approximate inference is needed for large practical problems, the real-world guarantee of focused grounding is not fundamentally different. When using approximate inference, all methods find an approximate solution, in general, different from the solution found with approximate inference on the complete grounding.

Conclusions

Like Riedel (2008), we found that using MaxWalkSat to try to find the MAP solution for the full grounding gives poor results in the Cora domain. Though it may seem strange to run MaxWalkSat for only one million flips on two and a half times as many clauses, we found no improvements with 10 or even 200 million flips. The problem, fully grounded, is simply too difficult to find a reasonable solution.

By limiting to the most relevant 300 thousand ground formulas, we were able to get an F-Score of 81.9%. Using cutting plane inference, Riedel reported an F-Score of 70% using MaxWalkSat as a base solver and 72% using Integer Linear Programming. Since ILP gives an exact solution and CPI inherits its accuracy from its base solver, we must conclude that a better MLN, rather than a better inference procedure is responsible for our higher F-Score.

Unlike MAP inference, marginal inference over the complete grounding performed well, a result established earlier by Singla and Domingos (2006a). There are two possible explanations for this substantial difference. First, while MAP inference attempts to find a complete truth assignment that is most probable, marginal inference finds the probability of each individual query atom. When scoring precision and recall over individual true query atoms, this method maximizes the scoring function. Second, the MLN used for these experiments was engineered and trained with marginal inference in mind.

In the Monroe domain, Blaylock and Allen's (2005b) instantiated plan recognition system operates in two stages: first it predicts the most probable plan schema, then predicts its parameters. Like Raghavan and Mooney (2010), we found that using MLNs with a complete grounding is intractable. With a focused grounding, we were able to use the advantages of jointly considering plan schemas and plan parameters to produce better parameter recognition, while keeping the same accuracy in schema recognition.

Our method of focused grounding constrains the number of formula groundings in a domain general way while also permitting domain specific heuristics. Focused grounding can be applied to any problem represented using a Markov Logic Network and paired with any inference procedure that operates over a set of ground formulas. The smaller set of ground formulas produced by our method allows more efficient and accurate inference in two diverse domains.

Acknowledgments

Research supported in part by Air Force Contract FA8750-09-C-0172 under the DARPA Machine Reading Program.

References

Blaylock, N., and Allen, J. 2005a. Generating artificial corpora for plan recognition. In *International Conference on User Modeling (UM'05)*. Springer.

Blaylock, N., and Allen, J. 2005b. Recognizing instantiated goals using statistical methods. In *G. Kaminka (Ed.), Workshop on Modeling Others from Observations (MOO 2005)*, 79–86.

Braz, R. D. S.; Amir, E.; and Roth, D. 2005. Lifted first-order probabilistic inference. In *Proceedings of IJCAI-05, 19th International Joint Conference on Artificial Intelligence*, 1319–1325. Morgan Kaufmann.

Culotta, A., and Mccallum, A. 2005. Practical Markov Logic containing first-order quantifiers with application to identity uncertainty. Technical report, HLT Workshop on Computationally Hard Problems and Joint Inference in Speech and Language Processing.

de Salvo Braz, R.; Natarajan, S.; Bui, H.; Shavlik, J.; and Russell, S. 2009. Anytime lifted belief propagation. In *Proc. SRL-2009, the International Workshop on Statistical Relational Learning*.

Domingos, P.; Kok, S.; Poon, H.; Richardson, M.; and Singla, P. 2006. Unifying logical and statistical AI. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, 2–7. AAAI Press.

Domingos, P. 2006. What's missing in AI: The interface layer. In Cohen, P., ed., *Artificial Intelligence: The First Hundred Years*. AAAI Press.

Kate, R. J., and Mooney, R. J. 2009. Probabilistic abduction using Markov Logic networks. In *Proceedings of IJCAI 2009 Workshop on Plan, Activity, and Intent Recognition (PAIR)*, 22–28.

McCallum, A.; Nigam, K.; and Ungar, L. H. 2000. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '00, 169–178. New York, NY, USA: ACM.

McCallum, A.; Schultz, K.; and Singh, S. 2009. Factorie: Probabilistic programming via imperatively defined factor graphs. In *Neural Information Processing Systems Conference (NIPS)*.

Poon, H., and Domingos, P. 2006. Sound and efficient inference with probabilistic and deterministic dependencies. In *Proceedings of the 21st national conference on Artificial intelligence - Volume 1*, 458–463. AAAI Press.

Poon, H., and Domingos, P. 2007. Joint inference in information extraction. In *Proceedings of the 22nd national conference on Artificial intelligence - Volume 1*, 913–918. AAAI Press.

Poon, H., and Domingos, P. 2009. Unsupervised semantic parsing. In *EMNLP '09: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, 1–10. Morristown, NJ, USA: Association for Computational Linguistics.

Raghavan, S., and Mooney, R. 2010. Bayesian abductive logic programs. In *Proceedings of the AAAI-10 Workshop on Statistical Relational AI (Star-AI 10)*, 82–87.

Richardson, M., and Domingos, P. 2006. Markov Logic networks. *Machine Learning* 62:107–136.

Riedel, S. 2008. Improving the accuracy and efficiency of MAP inference for Markov Logic. In *Proceedings of the Twenty-Fourth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-08)*, 468–475. Corvallis, Oregon: AUAI Press.

Roth, D. 1996. On the hardness of approximate reasoning. *Artificial Intelligence* 82:273–302.

Selman, B.; Kautz, H. A.; and Cohen, B. 1996. Local search strategies for satisfiability testing. In *DIMACS: Series in Discrete Mathematics and Theoretical Computer Science*, 521–532.

Singla, P., and Domingos, P. 2006a. Entity resolution with markov logic. In *ICDM*, 572–582. IEEE Computer Society Press.

Singla, P., and Domingos, P. 2006b. Memory-efficient inference in relational domains. In *Proceedings of the 21st national conference on Artificial intelligence - Volume 1*, 488–493. AAAI Press.

Taskar, B. 2005. *Learning structured prediction models: a large margin approach*. Ph.D. Dissertation, Stanford, CA, USA. AAI3153077.