Efficient Markov Logic Inference for Natural Language Semantics

Islam Beltagy
Department of Computer Science
The University of Texas at Austin
Austin, Texas 78712
beltagy@cs.utexas.edu

Raymond J. Mooney
Department of Computer Science
The University of Texas at Austin
Austin, Texas 78712
mooney@cs.utexas.edu

Abstract

Using Markov logic to integrate logical and distributional information in natural-language semantics results in complex inference problems involving long, complicated formulae. Current inference methods for Markov logic are ineffective on such problems. To address this problem, we propose a new inference algorithm based on SampleSearch that computes probabilities of complete formulae rather than ground atoms. We also introduce a modified closed-world assumption that significantly reduces the size of the ground network, thereby making inference feasible. Our approach is evaluated on the recognizing textual entailment task, and experiments demonstrate its dramatic impact on the efficiency of inference.

1 Introduction

A novel approach to natural-language semantics that uses Markov Logic Networks (MLNs) (Richardson and Domingos 2006) to integrate logical and distributional approaches to linguistic meaning was recently proposed by Beltagy et al. (2013). However, using this approach to solve problems in recognizing textual entailment (RTE) (i.e. determining whether one natural-language sentence reasonably implies or contradicts another) (Dagan et al. 2013) results in complex inference problems that are intractable for existing MLN inference methods.

Given a set of weighted formulae in first-order logic, MLNs use them as a template to construct a Markov network that defines a probability distribution over possible worlds. MLN inference calculates the probability of query Q given a set of evidence E and a set of weighted clauses R in first-order logic. MLN inference is computationally difficult, and various sampling algorithms have been proposed for approximate inference (Domingos and Lowd 2009).

The inference task addressed by Beltagy et al. (2013) has two distinct features that tend to violate the implicit assumptions behind the currently available MLN inference algorithms and their implementations. First, the query is a complex logical formula, while available MLN tools calculate marginal probabilities of ground atoms only. In this paper, we introduce an inference algorithm based on SampleSearch (Gogate and Dechter 2011) that directly calculates the probability of complete formulae.

The second problem with the resulting inference tasks is the large number of predicates and long formulae. This results in an extremely large number of ground clauses that existing inference algorithms fail to handle efficiently. Even recent advancements in lifted inference (Singla and Domingos 2008; Gogate and Domingos 2011) are not able to reduce the size of the ground network to a feasible size. Lifted inference relies on finding structural regularities (symmetries) in the MLN and avoiding grounding repeated structures. However, for these regularities to be frequent enough, the implicit assumption is that formulae are typically short and the set of evidence is large. These assumptions do not fit the inference problems we are addressing, which involve limited number of ground facts but complex logical formulae, which makes lifting ineffective.

To address this problem, we propose a modified closed-world assumption that helps reduce the size of the ground network and make inference tractable. Any ground atom that is *unreachable* from the evidence, is considered to be False. A ground atom is said to be *reachable* from the evidence if there is a way to propagate the evidence through the formulae and reach this ground atom. Note that this notion of reachability is different from reachability in the graph theoretic sense. The intuition behind our modified closed-world assumption is related to the idea of imposing a low prior on all predicates, which means that all ground atoms are effectively assumed to be False by default unless they can be inferred from the available evidence and background knowledge. However, by explicitly providing evidence that unreachable atoms are False, the size of the ground network is dramatically reduced.

The rest of the paper is organized as follows, section 2 is the background section, section 3 discusses an inference algorithm that supports complex query formulae, section 4 explains the modified closed-world assumption, section 5 is the experimental evaluation, and sections 6 and 7 discuss future work and conclusions, respectively.

2 Background

2.1 Logical Semantics

Logic-based representations of natural language meaning map a natural sentence into a logical form (Montague 1970). They handle many complex semantic phenomena such as relational propositions, logical operators, and quantifiers; however, they cannot handle "graded" aspects of meaning in language because they are binary by nature.

Boxer (Bos 2008) is an example of a wide-coverage logical-semantic tool that maps a natural sentence to first-order logic. For example, consider the two sentences "A man is driving a car", and "A guy is driving a vehicle". They become:

$T: \exists x, y, z.\ \ man(x) \wedge agent(y, x) \wedge drive(y) \wedge patient(y, z) \wedge car(z)$

$H: \exists x, y, z.\ \ guy(x) \wedge agent(y, x) \wedge drive(y) \wedge patient(y, z) \wedge vehicle(z)$

2.2 Distributional Semantics

Distributional models use statistics on large corpora to predict semantic similarity of words and phrases (Turney and Pantel 2010; Mitchell and Lapata 2010), based on the observation that semantically similar words occur in similar contexts. Distributional models capture the graded nature of meaning, but do not adequately capture logical structure (Grefenstette 2013).

Distributional semantic knowledge can be encoded as weighted inference rules. For the example above, relevant rules could look like:

$r_1: \forall x.\, man(x) \Rightarrow guy(x) \mid w_1$

$r_2: \forall x.\, car(x) \Rightarrow vehicle(x) \mid w_2$

where w_1, w_2 are weights calculated from the distributional similarity between the rule's antecedent and consequent.

2.3 Recognizing Textual Entailment

Recognizing Textual Entailment (RTE) (Dagan et al. 2013) is the task of determining whether one natural language text, the *premise* T, Entails, Contradicts, or not related (Neutral) to another, the *hypothesis* H.

Beltagy et al. (2013) proposed a framework to perform the RTE task using MLNs. Given T and H in first-order logic, and given the knowledge base KB generated from distributional semantics, MLN inference is then used to compute $P(H|T, KB)$, which is then used as a measure of the degree to which T entails H.

2.4 Markov Logic Network

Markov Logic Networks (MLNs) (Richardson and Domingos 2006) are a framework for probabilistic logic that employ weighted formulae in first-order logic to compactly encode complex undirected probabilistic graphical models (i.e., Markov networks). MLNs define a probability distribution over possible worlds. Probability of a given world x is denoted by:

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

where Z is the partition function, i ranges over all formulae F_i is the MLN, w_i is the weight of F_i and n_i is the number of true groundings of F_i. MLN's marginal inference calculates the probability $P(Q|E, R)$, where Q is a query, E is the evidence set, and R is the set of weighted formulae.

Alchemy (Kok et al. 2005) is the most widely used MLN implementation. It is a software package that contains implementations of a variety of MLN inference and learning algorithms.

2.5 MC-SAT

MC-SAT (Poon and Domingos 2006) is a popular and generally effective MLN inference algorithm. It combines ideas from MCMC and satisfiability. MC-SAT generates samples from the probability distribution specified by the weighted rules, and use these samples to estimate marginal probabilities of query ground atoms. For each sample, MC-SAT randomly selects a set of weighted clauses to sample from, where the probability of a clause to be added to the set exponentially increases with the rule's weight. Hard clauses are always included in this set. Once a set of clauses is selected, clauses are sampled using SampleSAT (Wei, Erenrich, and Selman 2004). SampleSAT uses WalkSAT (Selman et al. 1993), an efficient satisfiability solver, to efficiently generate nearly uniform samples.

2.6 Lifted Inference

The grounding process of a first-order formula generates a large number of ground atoms and ground clauses. Lifting techniques exploit the logical structure of the problem and the similarities in the ground network to reduce computation and the size of the ground network.

Lifted First-Order Belief Propagation proposed in (Singla and Domingos 2008) is a lifted version of the well known inference algorithm, Belief Propagation. It starts by building a *lifted network* where similar ground atoms of a first-order atom are grouped in one *supernode*, and similar ground clauses of a first-order clause are grouped in one *superfeature*. Ground atoms are considered similar (and merged in the same supernode) if they send and receive the same messages at each step of belief propagation. Similarly, ground clauses are considered similar (and merged in the same superfeature) if the ground clauses send and receive the same messages at each step of belief propagation. The minimum size of the lifted network is the size of the MLN, and the maximum size is that of a full ground Markov network. After building the lifted network, belief propagation proceeds normally with different messages that take lifting into account.

Another lifting technique is proposed in (Gogate and Domingos 2011). It starts by showing that inference in the propositional case (full ground network) is equivalent to "weighted model counting", then suggests an algorithm for weighted model counting that involves two basic steps, a decomposition and a splitting step. Then, it shows that inference in the first-order case is equivalent to "lifted weighted model counting", and shows how to lift the basic decomposition and splitting steps.

2.7 SampleSearch

SampleSearch (Gogate and Dechter 2011) is an importance sampling algorithm for graphical models that have a mix of probabilistic and deterministic constraints. Importance sampling in general is problematic in the presence of determinism, because many of the generated samples violate the deterministic constraints, and they get rejected. Instead, SampleSearch combines the sampling process with a backtracking search that uses a SAT solver to modify the generated sample if it violates the deterministic constraints.

SampleSearch works on top of a base-level sampler. The base sampler generates samples, and then SampleSearch makes sure the final samples satisfy all hard constraints. We use an implementation of SampleSearch that uses a generalized belief propagation algorithm called Iterative Join-Graph Propagation (IJGP)(Dechter, Kask, and Mateescu 2002). This version is available online (Gogate 2014).

3 Inference with Complex Query Formulae

Current implementations of MLNs like Alchemy (Kok et al. 2005) do not allow queries to be complex formulae, they can only calculate probabilities of ground atoms. This section discusses an inference algorithm for arbitrary query formulae.

3.1 Standard Work-Around

Although current MLN implementations can only calculate probabilities of ground atoms, they can be used to calculate the probability of a complex formula through a simple work-around. The complex query formula Q is added to the MLN using the hard formula:

$$Q \leftrightarrow result(D) \mid \infty \tag{2}$$

where $result(D)$ is a new ground atom that is not used anywhere else in the MLN. Then, inference is run to calculate the probability of $result(D)$, which is equal to the probability of the formula Q.

However, this approach can be very inefficient for some queries. For example, consider the query Q,

$$\begin{aligned} Q:\ \exists x,y,z.\ & man(x) \wedge agent(y,x) \wedge drive(y) \\ & \wedge patient(y,z) \wedge car(z) \end{aligned} \tag{3}$$

This form of existentially quantified formula with a list of conjunctively joined atoms, is very common in the inference problems we are addressing, so it is important to have efficient inference for such queries. However, using this Q in equation 2 results in a very inefficient MLN. The direction $Q \leftarrow result(D)$ of the double-implication in equation 2 is very inefficient because the existentially quantified formula is replaced with a large disjunction over all possible combinations of constants for variables x, y and z (Gogate and Domingos 2011). Generating this disjunction, converting it to clausal form, and running inference on the resulting ground network becomes increasingly intractable as the number of variables and constants grows.

3.2 New Inference Method

Instead, we propose an inference algorithm to directly calculate the probability of complex query formulae. The probability of a formula is the sum of the probabilities of the possible worlds that satisfy it. Gogate and Domingo (2011) show that to calculate the probability of a formula Q given a probabilistic knowledge base K, it is enough to compute the partition function Z of K with and without Q added as a hard formula:

$$P(Q \mid K) = \frac{Z(K \cup \{(Q,\infty)\})}{Z(K)} \tag{4}$$

Therefore, all we need is an appropriate algorithm to estimate the partition function Z of a Markov network. Then, we construct two ground networks, one with the query and one without, and estimate their Zs using that estimator. The ratio between the two Zs is the probability of Q.

We tried to estimate Z using a harmonic-mean estimator on the samples generated by MC-SAT, but we found that the estimates are highly inaccurate as shown in (Venugopal and Gogate 2013). So, the partition function estimator we use is SampleSearch (Gogate and Dechter 2011). SampleSearch has been shown to be an effective sampling algorithm when there is a mix of probabilistic and deterministic (hard) constraints, a fundamental property of the inference problems we address.

For the example Q in equation 3, in order to avoid generating a large disjunction because of the existentially quantified variables, we replace Q with its negation $\neg Q$, so the existential quantifiers are replaced with universals, which are easier to ground and perform inference upon. Finally, we compute the probability of the query $P(Q) = 1 - P(\neg Q)$. Note that replacing Q with $\neg Q$ cannot make inference with the technique discussed in section 3.1 faster, because with $\neg Q$, the direction $\neg Q \rightarrow result(D)$ suffers from the same problem of the existential quantifiers instead of the other direction $\neg Q \leftarrow result(D)$.

4 Modified Closed-World Assumption

This section explains the problem with the explosive size of the ground network, and why lifting techniques are not enough to solve it. Next it discusses the relationship between the traditional low prior on predicates, and our modified closed-world assumption. Finally, it defines our modified closed-world assumption and describes how it is implemented.

4.1 Problem Description

In the inference problems we address, typically formulae are long, especially the query formula. First-order formulae result in an exponential number of ground clauses, where the number of ground clauses of a formula is $O(c^v)$, where c is number of constants in the domain, and v is number of variables in the formula. For any moderately long formula, the number of resulting ground clauses is infeasible to process in any reasonable time using available inference algorithms. Even recent lifting techniques (Singla and Domingos 2008; Gogate and Domingos 2011) that try to group similar ground clauses to reduce the total number of nodes in the ground network, are not applicable here. Lifting techniques implicitly assume that c is large compared to v, and number of ground clauses is large because c is large. In our case, c and v are typically in the same range, and v is large, and this makes lifting algorithms fail to find similarities to lift.

4.2 Low prior

In the inference problems we address, as in most MLN applications, all atoms are initialized with a low prior. This low prior means that, by default, all groundings of an atom have very low probability, unless they can be inferred from the evidence and knowledge base. However, we found that a large fraction of the ground atoms cannot be inferred, and their probabilities remain very low. This suggests that these ground atoms can be identified and removed in advance with very little impact on the approximate nature of the inference. As the number of such ground atoms is large, this has the potential to dramatically decrease the size of the ground network. Our modified closed-world assumption was created to address this issue.

4.3 Definition

Closed-world, open-world and our modified closed-world assumptions are different ways of specifying what ground atoms are initialized to True, False or Unknown. True and False ground atoms are used to construct the appropriate network but are not part of the final ground Markov network. Only Unknown ground atoms participate in probabilistic inference. All ground atoms specified as evidence are known (True or False). The difference between the three assumptions is in the non-evidence ground atoms. With a closed-world assumption, non-evidence ground atoms are all False. In case of the open-world assumption, non-evidence ground atoms are all Unknown and they are all part of the inference task. In case of our modified closed-world atom, non-evidence ground atoms are False by default, unless they are *reachable* from any of the evidence, or from a ground atom in an input formula.

4.4 Reachability

A ground atom is said to be *reachable* from the evidence if there is a way to propagate the evidence through the formulae and reach this ground atom. The same applies for ground atoms specified in an input formula. For example, consider the evidence set E, and clauses r_1, r_2:

$$
\begin{aligned}
E &: \{\, g(C_1), h(C_2) \,\} \\
r_1 &: \forall x, y.\ g(x) \vee h(y) \vee i(x, y) \\
r_2 &: \forall x, y.\ j(x) \vee k(y) \vee i(x, y)
\end{aligned}
$$

From r_1, variables x, y can be assigned the constants C_1, C_2 respectively because of the evidence $g(C_1)$, $h(C_2)$. Then, this evidence gets propagated to $i(C_1, C_2)$, so the ground atom $i(C_1, C_2)$ is Unknown. From r_2, the variables x, y can be assigned the constants C_1, C_2 respectively because of the Unknown ground atom $i(C_1, C_2)$, and this gets propagated to $j(C_1)$, $k(C_2)$, so ground atoms $j(C_1)$, $k(C_2)$ are also Unknown. All other ground atoms, except the evidence $g(C_1)$ and $h(C_2)$, are False because they are not reachable from any evidence.

Note that the definition of reachability here (mcw-reachable) is different from the definition of reachability in graph theory (graph-reachable). Nodes can be graph-reachable but not mcw-reachable. For the example above, consider the full ground network of E and r_1, which contains 8 nodes, and 4 cliques. It is a connected graph, and all nodes are graph-reachable from each others. However, as explained in the example, $i(C_1, C_2)$ is the only mcw-reachable node.

Algorithm 1 Grounding with modified closed-world assumption

Input R: $\{K \cup Q\}$ set of first-order clauses, where K is the set of clauses from the input MLN, and Q is the set of clauses from the query.
Input E: set of evidence (list of ground atoms)
Output : a set of ground clauses with the modified closed-world assumption applied

Add all E to the $reachable$ ground atoms
Add all ground atoms in R to $reachable$
repeat
  for all $r \in R$ **do**
    p = propagate $reachable$ ground atoms between predicates sharing the same variable
    add propagated ground atoms (p) to $reachable$
    if p not empty **then**
      $changed = true$
    end if
  end for
until not $changed$
Generate $False$ evidence for ground atoms $\notin reachable$ and add them to E
GC = Use MLN's grounding process to ground clauses R
for all $gc \in GC$ **do**
  gc = gc after substituting values of known ground atoms in E
  if $gc = True$ **then**
    drop gc
  else if $gc = False$ **then**
    if gc is a grounding of one of Q's clauses **then**
      Terminate inference with $Z = 0$
    else
      if gc is hard clause **then**
        Error inconsistent MLN
      else
        drop gc
      end if
    end if
  else
    keep gc in GC
  end if
end for
return GC

4.5 Algorithm and Implementation

Algorithm 1 describes the details of the grounding process with the modified closed-world assumption applied. Lines 1 and 2 initialize the reachable set with the evidence and any ground atom in R. Lines 3-11 repeatedly propagate evidence until there is no change in the reachable set. Line 12 generates False evidence for all unreachable ground atoms. Line 13 generates all ground clauses, then lines from 14-31 substitute values of the known ground atoms in the ground clauses. Alchemy drops all True and False ground clauses, but this does not work when the goal of the inference algorithm is to calculate Z. Lines from 16-30 describe the change. True ground clauses are dropped, but not False ground clauses. If a False ground clause is a grounding of one of Q's clauses, then $Z = 0$ and there is no need to perform inference since there is no way to satisfy Q given E and R. If there is False hard clause, then this MLN is inconsistent. Otherwise, the False ground clause can be dropped. The resulting list of ground clauses GC are then passed to the inference algorithm to estimate Z.

5 Evaluation

This section evaluates the two techniques proposed in this paper using the RTE task, demonstrating the effectiveness of both components.

5.1 Task

Given the two natural-language sentences T and H represented in first-order logic, and given the background knowledge KB generated as in (Beltagy et al. 2013), two inferences are run, $P(H|T, KB)$ and $P(H|\neg T, KB)$.

The dataset used in our evaluation is "Sentences Involving Compositional Knowledge" (SICK) (Marelli et al. 2014). SICK is a new RTE dataset collected for the SemEval 2014 competition. Only the "training set" is available at this point, which consists of 5,000 pairs of sentences.

5.2 Systems Compared

- **mln**: This system uses MC-SAT (Richardson and Domingos 2006) for inference without any modifications. It uses the work-around explained in section 3.1 to calculate the probability of a complex query formula, and uses an open-world assumption.
- **mln+qf**: This system uses our SampleSearch inference to directly calculate the probability of a query formula (qf), while making an open-world assumption.
- **mln+mcw**: This system uses MC-SAT with the work-around for computing the probability of a complex query formula, but uses our modified closed-world (mcw) assumption.
- **mln+qf+mcw**: This is our proposed technique, inference that supports a query formula (qf) and makes a modified closed-world (mcw) assumption.

For all systems, we run the two inferences $P(H|T, KB)$ and $P(H|\neg T, KB)$, then train a classifier that maps their outputs to one of the classes: Entails, Contradicts or Neutral.

	Accuracy	CPU Time	Timeouts
mln	56.94%	2min 27s	9,578
mln+qf	68.74%	1min 51s	2,964
mln+mcw	65.80%	10s	252
mln+qf+mcw	71.80%	7s	212

Table 1: Systems' performance, accuracy, CPU Time for completed runs only, and number of Timeouts out of 10,000 runs

We use a 30 minute timeout for each MLN inference problem in order to make the experiments tractable. If the system times out, it outputs -1 indicating an error, and and the final classifier learns to assign it to one of the three RTE classes. Usually, because the Neutral class is the largest, timeouts are classified as Neutral.

5.3 Metrics

- Accuracy: Percentage of correct classifications (Entail, Contradict, or Neutral) using 10-fold cross validation.
- CPU Time (completed runs): Average CPU time per run for the completed runs only, i.e. timed out runs are not included.
- Timeouts: Number of inferences that timeout after 30 minutes. Total number of runs is 10,000.

5.4 Results and Discussion

Table 1 summarizes the results of the experiments. First, for the four different systems, the CPU time (average time per run for completed runs only) is very short compared to the length of the timeout (30 minutes). This shows the exponential nature of the inference algorithms, either the problem is small enough to finish in few minutes, or if it is slightly larger, it fails to finish in reasonable time.

Comparing the systems, the results clearly show that the base system, (**mln**), is not effective for the type of inference problems that we are addressing, almost all of the runs timed out. System **mln+qf** shows the impact of being able to calculate the probability of a complex query directly. It significantly improves the accuracy, and it lowers the number of timeouts; however, the number of timeouts is still large. System **mln+mcw** shows the impact of the modified closed-world assumption, demonstrating that makes inference significantly faster, since the number of unreachable ground atoms in our application is large compared to the total number of ground atoms. However, the accuracy of **mln+mcw** is lower than that of **mln+qf**, since calculating the probability of a query directly is more accurate than the standard work-around. Finally, **mln+qf+mcw** is both more accurate and faster than the other systems, clearly demonstrating the effectiveness of our overall proposed approach.

6 Future Work

An obvious extension to this work is a better algorithm for computing the probability of an arbitrary formula. Instead of making two separate runs of SampleSearch to estimate two different Zs, it would be helpful to exploit the similarities between the two Markov networks (one with Q and one without Q) to reduce the amount of repeated computation. Also, it should be possible to optimize the calculations, or simplify them, knowing that we are really only interested in the ratio between the two Zs and not their individual values.

7 Conclusion

This paper has addressed the problem of making MLN inference feasible for textual inference that combines logical and distributional semantics, where queries are long, complex logical formulae (not just single ground atoms). We proposed a simple MLN inference algorithm based on SampleSearch to compute probabilities of formulae, and proposed a modified closed-world assumption that can dramatically reduce the size of the ground network. Experiments on a recent SemEval task in textual entailment demonstrated the effectiveness of these techniques.

Acknowledgments

We would like to thank Deepak Venugopal and Vibhav Gogate for insightful discussions.

This research was supported by the DARPA DEFT program under AFRL grant FA8750-13-2-0026. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the view of DARPA, DoD or the US government. Some experiments were run on the Mastodon Cluster supported by NSF Grant EIA-0303609.

References

Beltagy, I.; Chau, C.; Boleda, G.; Garrette, D.; Erk, K.; and Mooney, R. 2013. Montague meets Markov: Deep semantics with probabilistic logical form. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics (*SEM-13)*.

Bos, J. 2008. Wide-coverage semantic analysis with Boxer. In *Proceedings of Semantics in Text Processing (STEP-08)*.

Dagan, I.; Roth, D.; Sammons, M.; and Zanzotto, F. M. 2013. Recognizing textual entailment: Models and applications. *Synthesis Lectures on Human Language Technologies*.

Dechter, R.; Kask, K.; and Mateescu, R. 2002. Iterative join-graph propagation. In *Proceedings of the Eighteenth conference on Uncertainty in Artificial Intelligence (UAI-02)*.

Domingos, P., and Lowd, D. 2009. *Markov Logic: An Interface Layer for Artificial Intelligence*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.

Gogate, V., and Dechter, R. 2011. Samplesearch: Importance sampling in presence of determinism. *Artificial Intelligence* 175(2):694–729.

Gogate, V., and Domingos, P. 2011. Probabilistic theorem proving. In *27th Conference on Uncertainty in Artificial Intelligence (UAI-11)*.

Gogate, V. 2014. IJGP-sampling and samplesearch. http://www.hlt.utdallas.edu/~vgogate/ijgp-samplesearch.html.

Grefenstette, E. 2013. Towards a formal distributional semantics: Simulating logical calculi with tensors. In *Proceedings of Second Joint Conference on Lexical and Computational Semantics (*SEM-13)*.

Kok, S.; Singla, P.; Richardson, M.; and Domingos, P. 2005. The Alchemy system for statistical relational AI. Technical report, Department of Computer Science and Engineering, University of Washington. http://alchemy.cs.washington.edu/.

Marelli, M.; Menini, S.; Baroni, M.; Bentivogli, L.; Bernardi, R.; and Zamparelli, R. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *LREC-2014*.

Mitchell, J., and Lapata, M. 2010. Composition in distributional models of semantics. *Journal of Cognitive Science*.

Montague, R. 1970. Universal grammar. *Theoria* 36:373–398.

Poon, H., and Domingos, P. 2006. Sound and efficient inference with probabilistic and deterministic dependencies.

Richardson, M., and Domingos, P. 2006. Markov logic networks. 62:107–136.

Selman, B.; Kautz, H.; Cohen, B.; et al. 1993. Local search strategies for satisfiability testing. *Cliques, coloring, and satisfiability: Second DIMACS implementation challenge*.

Singla, P., and Domingos, P. 2008. Lifted first-order belief propagation. In *Proceedings of Association for the Advancement of Artificial Intelligence(AAAI-08)*.

Turney, P., and Pantel, P. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research* 37(1):141–188.

Venugopal, D., and Gogate, V. 2013. Giss: Combining samplesearch and importance sampling for inference in mixed probabilistic and deterministic graphical models. In *Proceedings of Association for the Advancement of Artificial Intelligence(AAAI-13)*.

Wei, W.; Erenrich, J.; and Selman, B. 2004. Towards efficient sampling: Exploiting random walk strategies. In *Proceedings of Association for the Advancement of Artificial Intelligence(AAAI-04)*.