Proceedings of the 25th AAAI Conference on Artificial Intelligence (AAAI-2011) (2011), pp. 1069-1075.

Abductive Markov Logic for Plan Recognition

Parag Singla **Raymond J. Mooney**
Department of Computer Science
University of Texas at Austin
1616 Guadalupe, Suite 2.408
Austin, TX USA 78701
{*parag,mooney*}*@cs.utexas.edu*

Abstract

Plan recognition is a form of abductive reasoning that involves inferring plans that best explain sets of observed actions. Most existing approaches to plan recognition and other abductive tasks employ either purely logical methods that do not handle uncertainty, or purely probabilistic methods that do not handle structured representations. To overcome these limitations, this paper introduces an approach to abductive reasoning using a first-order probabilistic logic, specifically Markov Logic Networks (MLNs). It introduces several novel techniques for making MLNs efficient and effective for abduction. Experiments on three plan recognition datasets show the benefit of our approach over existing methods.

Introduction

Abduction, inference to the best explanation, is a well-studied problem with a wide variety of applications ranging from plan and activity recognition to natural language understanding and diagnosis. Many existing solutions employ a purely logical framework (Pople 1973; Kakas, Kowalski, and Toni 1993) and hence, cannot judge the probability of alternative explanations nor handle uncertainty in the requisite knowledge or data. An alternative approach uses Bayesian networks and computes the posterior probability of possible explanations given the observations (Pearl 1988). Although this approach naturally handles uncertainty, it is propositional in nature and unable to handle structured data.

The last decade has seen rapid growth in the area of Statistical Relational AI, which uses well-founded probabilistic methods while maintaining the representational power of first-order logic. One of the most widely used formalisms is Markov Logic Networks (MLNs) (Domingos and Lowd 2009), which attaches real-valued weights to formulas in first order logic in order to represent their certainty. MLNs effectively use logic as a compact template for constructing complex ground Markov networks. In this work, we adapt MLNs to perform abduction, thereby incorporating the advantages of both logical and probabilistic approaches.

In MLNs, the probability of a possible world increases with the total weight of the satisfied formulae. Since an implication is satisfied whenever its consequent is true, an MLN is unable to abductively infer the antecedent of a rule from its consequent. Kate and Mooney (2009) presented the first attempt to enhance MLNs for abduction. However, their approach has serious inefficiencies that prevent it from scaling beyond fairly simple problems. Building on their framework, we propose a Hidden Cause (HC) model, which simplifies the resulting network by introducing a hidden cause for each rule antecedent. However, the HC model still leads to unnecessarily complex networks for many abductive problems. Therefore, we also propose a novel model construction procedure based on abduction. The HC model together with abductive model construction produces an effective MLN formulation that generally outperforms existing approaches on three plan-recognition datasets.

The remainder of the paper is organized as follows. First we provide some background on abduction and Markov logic. Next, we present our Hidden Cause model and abductive model construction procedure. Finally, we present an experimental evaluation on three benchmark datasets, followed by conclusions and future work.

Background

Abduction

In a logical framework, abduction is usually defined as follows (Pople 1973):

- **Given:** Background knowledge B and observations O, both represented as sets of formulae in first-order logic, where O is typically restricted to a set of ground literals.
- **Find:** A hypothesis H, also a set of logical formulae, such that $B \cup H \not\models \bot$ and $B \cup H \models O$.

Here $\models$ means logical entailment and $\bot$ means false, i.e. find a set of assumptions that is consistent with the background theory and explains the observations. There are generally many hypotheses H that explain a particular set of observations O. Following Occam's Razor, the best hypothesis is typically defined as the one that minimizes $|H|$. It is also common to assume that B is a set of Horn clauses and that H is a set of ground atoms. Given a set of observations O_1, O_2, ..., O_n, the set of abductive proof trees is computed by recursively backchaining on each O_i until every literal in the proof is either proven or assumed. Logical abduction has been applied to tasks such as plan recognition and diagnosis,

e.g. Ng and Mooney(1992). The primary problem with the logical approach is that it does not handle uncertainty.

Markov Logic Networks

Markov logic (Domingos and Lowd 2009) is a framework for combining first-order logic and undirected probabilistic graphical models (Markov networks). A traditional first-order knowledge base can be seen as a set of hard constraints on the set of possible worlds: if a world violates even one formula, its probability is zero. In order to soften these constraints, Markov logic attaches a weight to each formula in the knowledge base. A formula's weight reflects how strong a constraint it imposes on the set of possible worlds. Formally, an MLN is a set of pairs (F_i, w_i), where F_i is a first-order formula and w_i is a real number. A *hard clause* has an infinite weight and acts as a logical constraint; otherwise, it is a *soft clause*. Given a set of constants, an MLN defines a ground Markov network with a node in the network for each ground atom and a feature for each ground clause. The joint probability distribution over a set of boolean variables $X = (X_1, X_2...)$ corresponding to the nodes in the ground Markov network (i.e. ground atoms) is defined as:

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

where $n_i(x)$ is the number of true groundings of F_i in x and Z is a normalization term obtained by summing $P(X = x)$ over all values of X.

An MLN can be viewed as a set of templates for constructing ground Markov networks. Different sets of constants produce different Markov networks; however, there are certain regularities in their structure and parameters determined by the underlying first-order theory. MPE (most probable explanation) inference finds the joint assignment of values to unobserved nodes in the network that has maximum posterior probability given the values of a set of observed nodes. Marginal inference finds the probability of true assignment for each of unobserved nodes given the values of observed nodes. Standard inference techniques such as belief propagation (MPE and marginal) or Gibbs sampling (marginal) can be used. MLN weights can be learned by maximizing the conditional log-likelihood of training data supplied in the form of a database of true ground literals. A number of efficient inference and learning algorithms that exploit the structure of the network have also been proposed. Domingos and Lowd (2009) provide details on these and many other aspects of MLNs.

Markov Logic for Abduction

Traditional MLNs do not support logical abduction. Given the rule $P \Rightarrow Q$ and the observation that Q is true, we would like to abduce P as a possible cause for Q. But once the consequent (Q) is true, the clause is satisfied independent of the value of the antecedent (P) and hence, does not give any information about the truth value of the antecedent.

In this section, we describe three ideas for extending MLNs with abduction, each building on the previous ones. First, we describe the Pairwise Constraint (PC) model proposed by Kate and Mooney (2009). This is followed by the Hidden Cause (HC) model which alleviates some of the inefficiencies of the PC model. Finally, we present an abductive model construction procedure that produces a simpler, more effective ground Markov network.

There has been some recent work using MLNs for intent recognition in the multi-agent setting of the game Capture the Flag (Sadilek and Kautz 2010). The domain knowledge is completely hand-crafted to predict the predicates of interest. We take a different approach which only assumes the existence of a knowledge-base capturing the causal structure of the domain. No explicit knowledge-engineering is required to adapt planning knowledge for plan recognition. This avoids costly human labor since, unlike the approach taken by Sadilek and Kautz (2010), it allows the same knowledge base to be used for both planning *and* plan recognition. Directly comparing the two approaches is a direction for future work.

Pairwise Constraint Model

Kate and Mooney (2009) presented the first attempt to extend MLNs with abduction, which we will call the Pairwise Constraint (PC) model. The key idea is to introduce explicit reversals of the implications appearing in the original knowledge base. Multiple possible explanations for the same observation are supported by having a disjunction of the potential explanations in the reverse implication. "Explaining away" (Pearl 1988) (inferring one cause eliminates the need for others) is achieved by introducing a mutual-exclusivity constraint between every pair of possible causes for an observation. Given the set of Horn clauses: $P_1 \Rightarrow Q, P_2 \Rightarrow Q, \cdots P_n \Rightarrow Q$, a reverse implication: $Q \Rightarrow P_1 \vee P_2 \cdots \vee P_n$, and a set of mutual-exclusivity constraints: $Q \Rightarrow \neg P_1 \vee \neg P_2, \cdots Q \Rightarrow \neg P_{n-1} \vee \neg P_n$ for every pair of explanations, are introduced. The weights on these clauses control the strength of the abductive inference and the typical number of alternate explanations, respectively.

For first-order Horn clauses, all variables not appearing in the head of the clause become existentially quantified in the reverse implication. Kate & Mooney (2009) give the details of the conversion process. Here is a concrete example motivated by one of our evaluation benchmarks, the emergency response domain introduced by Blaylock & Allen (2005) (by default variables are universally quantified):

$\texttt{hvy\_snow}(\texttt{loc}) \wedge \texttt{drive\_hzrd}(\texttt{loc}) \Rightarrow \texttt{blk\_rd}(\texttt{loc})$
$\texttt{acdnt}(\texttt{loc}) \wedge \texttt{clr\_wrk}(\texttt{crew}, \texttt{loc}) \Rightarrow \texttt{blk\_rd}(\texttt{loc})$

These rules give two explanations for a road being blocked at a location: 1) there has been heavy snow resulting in hazardous driving conditions, and 2) there has been an accident and the crew is clearing the wreck. Given the observation that a road is blocked, we should be able to abductively infer one of these causes as the explanation. The final combined reverse implication and pairwise constraint clauses are:

$\texttt{blk\_rd}(\texttt{loc}) \Rightarrow (\texttt{hvy\_snow}(\texttt{loc}) \wedge \texttt{drive\_hzrd}(\texttt{loc})) \vee$
$\quad (\exists \texttt{crew acdnt}(\texttt{loc}) \wedge \texttt{clr\_wrk}(\texttt{crew}, \texttt{loc}))$
$\texttt{blk\_rd}(\texttt{loc}) \Rightarrow \neg(\texttt{hvy\_snow}(\texttt{loc}) \wedge \texttt{drive\_hzrd}(\texttt{loc})) \vee$

$$\neg(\exists \texttt{crew acdnt}(\texttt{loc}) \wedge \texttt{clr\_wrk}(\texttt{crew}, \texttt{loc}))$$

The first rule introduces the two possible explanations and the second rule constrains them to be mutually exclusive.

The PC model constructs an unnecessarily complex network, in part because including multiple clause bodies in the reverse implication makes it quite long. If there are n possible causes for an observation and each of the corresponding Horn clause has k literals in its body, then the reverse implication has $O(nk)$ literals. This in turn results in cliques of size $O(nk)$ in the ground network. This significantly increases computational complexity since probabilistic inference is *exponential* in the treewidth of the graph which in turn is at least the size of the maximum clique (Koller and Friedman 2009). The PC model also introduces $O(n^2)$ pairwise constraints, which can result in a large number of ground clauses.

Hidden Cause Model

The Hidden Cause (HC) model fixes some of the inefficiencies of the PC model by introducing a hidden cause node for each possible explanation. The joint constraints can then be expressed in terms of these hidden causes, thereby reducing the size of the reverse implication (and hence, the corresponding clique size) to $O(n)$. The need for the pairwise constraints is eliminated by specifying a low prior on all hidden causes. A low prior indicates that in absence of any reason to be true, each of the hidden causes is most likely to be false. Hence, in the presence of an observation, inferring one cause obviates the need for the others. We now describe the HC model more formally. We first consider the propositional case for ease of explanation. It is straightforwardly extended to first-order Horn clauses. Consider, the following set of rules describing the possible explanations for a proposition Q:.

$$P_{i1} \wedge P_{i2} \wedge \cdots \wedge P_{ik_i} \Rightarrow Q, \forall i, (1 \leq i \leq n)$$

For each rule we introduce a hidden cause C_i and add the following rules to the MLN:

- $P_{i1} \wedge P_{i2} \wedge \cdots \wedge P_{ik_i} \Leftrightarrow C_i, \forall i, (1 \leq i \leq n)$
- $C_i \Rightarrow Q, \forall i, (1 \leq i \leq n)$
- $Q \Rightarrow C_1 \vee C_2 \cdots C_n$ (reverse implication)
- $\texttt{true} \Rightarrow C_i, \forall i, (1 \leq i \leq n)$ (negatively weighted)

The first set of rules are soft clauses with high positive weights. This allows the antecedents to sometimes fail to cause the consequent (and vice-versa). The next two sets of rules are hard clauses (in effect, they implement a deterministic-or function between the consequent and the hidden causes). The last rule implements a low prior (by having a negative MLN weight) on the hidden causes. These low priors discourage inferring multiple hidden causes for the same consequent ("explaining way"), and the strength of the prior determines the degree to which multiple explanations are allowed. Different sets of weights on the biconditional in the first set of rules implement different ways of combining multiple explanations. For example, a noisy-or (Pearl 1988) can be implemented by modeling the implication from antecedents to the hidden cause as a soft constraint and the reverse direction as a hard constraint. The weight w_i for the soft-constraint is set to $log[(1 - p_{f_i})/p_{f_i}]$, where p_{f_i} is the failure probability for cause i. This formulation has some similarity to Natarajan *et al.*'s (2010) implementation of combining functions in Markov logic; however, their work does not directly concern abduction or inferring causes from observed effects. There has been prior work on automatically detecting the hidden structure in a domain (e.g. (Davis et al. 2007), (Kok and Domingos 2007)). In our case, we can directly construct the requisite hidden predicates from the existing clauses, thereby eliminating the need for such explicit predicate invention. Whether such techniques could automatically induce such hidden structure from data alone is a potential direction for future work.

For first-order Horn clauses, variables present in the antecedents but not in the consequent become existentially quantified in the reverse implication, as in the PC model. But unlike the PC model, the reverse implication expression is much simpler as it only involves one predicate (the hidden cause) for each rule implying the consequent. Revisiting the blocked road example, we introduce two hidden causes corresponding to the two rules:

$$\texttt{hvy\_snow}(\texttt{loc}) \wedge \texttt{drive\_hzrd}(\texttt{loc}) \Leftrightarrow \texttt{rb\_C1}(\texttt{loc})$$
$$\texttt{acdnt}(\texttt{loc}) \wedge \texttt{clr\_wrk}(\texttt{crew}, \texttt{loc}) \Leftrightarrow \texttt{rb\_C2}(\texttt{crew}, \texttt{loc})$$

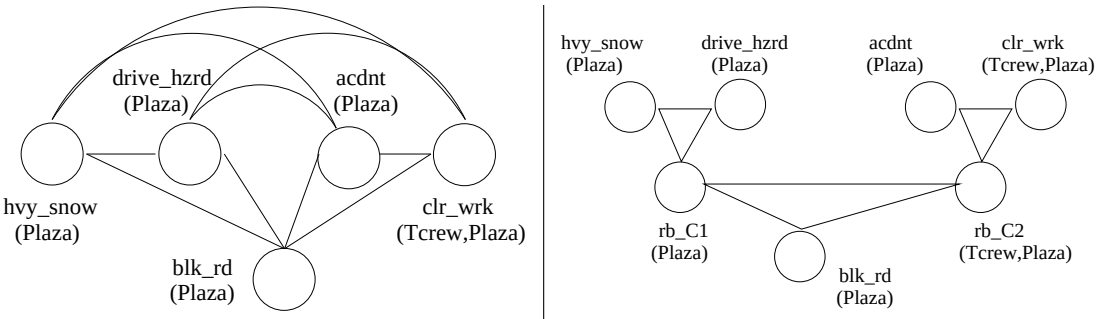
Note that each hidden cause contains all variables present in the antecedent of the rule. These hidden causes are combined with the original consequent as follows:

$$\texttt{rb\_C1}(\texttt{loc}) \Rightarrow \texttt{blk\_rd}(\texttt{loc})$$
$$\texttt{rb\_C2}(\texttt{crew}, \texttt{loc}) \Rightarrow \texttt{blk\_rd}(\texttt{loc})$$
$$\texttt{blk\_rd}(\texttt{loc}) \Rightarrow \texttt{rb\_C1}(\texttt{loc}) \vee \exists \texttt{crew}(\texttt{rb\_C2}(\texttt{crew}, \texttt{loc}))$$

In addition, there are unit clauses specifying low priors on the hidden causes.

Figure 1 shows the ground network constructed by the two models when `loc` is bound to `Plaza` and `crew` to `Tcrew`. The PC model results in a fully connected graph (maximum clique size is 5), whereas the HC model is much sparser (maximum clique size is 3). Consequently, inference in the HC model is significantly more efficient.

Algorithm 1 presents the pseudocode for constructing the abductive MLN given a Horn-clause knowledge base. First (lines 2 to 8), hidden causes are created for each possible explanation of each consequent (line 5). A biconditional is introduced between the hidden causes and the corresponding antecedents (line 6). These are modeled as soft clauses in the MLN. Each hidden cause is also linked to the corresponding consequent via a hard clause (line 7). The next part (lines 9 to 24) combines the hidden causes for each of the consequents into a single reverse implication. The rules are partitioned according to the first-order predicate appearing in the consequent (line 9). For each partition (line 10), each possible instantiation of the consequent predicate appearing the underlying rules is considered (lines 11 to 13). For instance, given the rule: $\texttt{h}_1(\texttt{x}, \texttt{y}) \Rightarrow \texttt{q}(\texttt{y})$ and another: $\texttt{h}_2(\texttt{x}, \texttt{Const}) \Rightarrow \texttt{q}(\texttt{Const})$, we need to consider each of the instantiations $\texttt{q}(\texttt{x})$ and $\texttt{q}(\texttt{Const})$ separately. For each such instantiation c (line 14), we consider the rules which could result in the consequent c being true (line 15). Technically, these are the rules whose consequents subsume c, i.e. there



Pairwise Constraint (PC) Hidden Cause (HC)

Figure 1: Ground networks for the two models for the road blocked example

exists a substitution θ_i such that $c = C(r_i)\theta_i$ where $C(r_i)$ is the consequent of rule r_i. These are the rules which could possibly cause c when bound by the substitution θ_i. For each such rule r_i (line 17), substitution θ_i is applied to the corresponding hidden cause $H(r_i)$ (line 18). Then, the set of free variables in the hidden cause (i.e. the variables not appearing in c) is extracted (line 19). These variables are existentially quantified in the reverse implication (next step). We then introduce a reverse implication saying that c implies at least one of the consequents (amongst those that subsume c) (line 21). This reverse implication is made a hard clause and added to the MLN (line 22). Finally, a low prior is introduced for each hidden cause (lines 25 to 27).

Abductive Model Construction

Abduction using Markov logic consists of the following 3 steps: 1) Generate the abductive MLN, 2) Construct the ground Markov network (model construction), 3) Perform learning/inference over the resulting ground network. The standard MLN model construction process uses the set of all possible ground atoms (the Herbrand base) and the set of all possible ground clauses using these atoms. Using logical rules to construct a graphical model is generally referred to as *knowledge-based model construction* (KBMC), originally proposed by Ngo and Haddawy (1997). In abduction, where we are looking for explanations of a set of observations, the set of all possible ground atoms and clauses may not be needed to explain the observations. Considering the fully-grounded network leads to increased time and memory complexity of learning and inference. For instance, in the road blocked example, if the observation of interest is $\texttt{blk\_rd}(\texttt{Plaza})$, then we can ignore groundings where the location is not `Plaza`.

We propose an alternative model-construction procedure that uses logical abduction to determine the set of *relevant* ground atoms. The procedure first constructs the set of abductive proof trees for the observations and then uses only the ground atoms in these proofs instead of the full Herbrand base. The ground Markov network is then constructed by instantiating the formulae in the abductive MLN using this reduced set of ground atoms. We refer to the set of ground atoms (Markov network) thus constructed as the abductive ground atoms (Markov network). First, given a set of Horn rules and a set of observations, the rules for the abductive MLN are constructed using the HC model. Next, the set of most-specific abductive proof trees for the observations are computed using the method of Stickel (1988). The atoms in these proofs form the set of abductive ground atoms. For each formula in the abductive MLN, the set of all ground formulae whose atoms appear in the abductive ground atoms are added to the ground Markov network. While handling existentials, only those disjuncts which belong to the set of abductive ground atoms are used. Learning and inference are then performed over the resulting network.

In general, abductive model construction results in a ground network that is substantially different (and usually much simpler) than that constructed using the full Herbrand base. It also differs from the network constructed by starting KBMC from the query/observations (Domingos and Lowd 2009) because of the use of the backward chaining and unification during the abductive model construction. Consequently, the probabilistic inferences supported by this model can be different from that of the traditional MLN model. This also makes the abductive process different from other pre-processing approaches such as Shavlik and Natarajan (2009), or existing lifted inference techniques such as Singla and Domingos (2008), both of which produce a network that is probabilistically equivalent to the original. As shown in our experiments, by focusing on the relevant ground atoms, abductive model construction significantly improves the performance of abductive MLNs *both* in terms of time and memory efficiency as well as predictive accuracy. Further, lifted inference could still be applied by constructing lifted network over the nodes/clauses present in the abductive network. We will refer to the Hidden Cause model followed by abductive model construction as the HCAM model.

Experiments

This section presents experiments comparing the three MLN models and several existing plan-recognition systems. These include Blaylock and Allen's (2005) method that learns n-gram models to separately predict the plan schema (predicate) and its arguments. This contrasts with the joint prediction performed by MLNs. We also compare to ACCEL (Ng and Mooney 1992), a purely logic-based system, and Bayesian Abductive Logic Programs (BALPs) (2010), which combine Bayesian networks and logical abduction.

Algorithm 1 GenAbductiveMLN(**KB**)

inputs: **KB**, a Horn clause knowledge base
output: $\mathbf{M}$, set of rules in the abductive MLN

$\mathbf{M} \leftarrow \{\}$
for all $r \in$ **KB do**
  $A(r) \leftarrow$ antecedent in r
  $C(r) \leftarrow$ consequent in r
  $H(r) \leftarrow$ hidden cause for r
  $\mathbf{M} \leftarrow \mathbf{M} \cup \{A(r) \Leftrightarrow H(r)\}$
  $\mathbf{M} \leftarrow \mathbf{M} \cup \{H(r) \Rightarrow C(r)\}$
end for
Part(**KB**) $\leftarrow$ partition of **KB** into sets or rules with same (first-order) predicate in the consequent
for all set of rules $R \in$ Part(**KB**) **do**
  Let $R = \{r_1, r_2, \cdots r_m\}$
  $C(R) \leftarrow \bigcup_{i=1}^{m} \{C(r_i)\}$
  ($C(R)$ is set of unique consequents appearing in R)
  for all $c \in C(R)$ **do**
    $R_c \leftarrow \{r_i \in R \mid \exists \theta_i, c = C(r_i)\theta_i\}$
    (R_c is set of rules whose consequents subsume c)
    for all $r_i \in R_c$ **do**
      $H_{\theta_i}(r_i) \leftarrow H(r_i)\theta_i$
      $\{v_{i_1}, v_{i_2}, \cdots v_{i_k}\} \leftarrow$ variables appearing in $H_{\theta_i}(r_i)$ but not in c
    end for
    $I_r(R_c) \leftarrow (c \Rightarrow \bigvee_{i=1}^{m} \exists v_{i_1}, v_{i_2}, \cdots v_{i_k} H_{\theta_i}(r_i))$
    $\mathbf{M} \leftarrow \mathbf{M} \cup \{I_r(R_c)\}$
  end for
end for
for all $r \in$ **KB do**
  $\mathbf{M} \leftarrow \mathbf{M} \cup \{\texttt{true} \Rightarrow H(r)$ (negatively weighted) $\}$
end for
return M

Datasets

Story Understanding Our first dataset was previously used to evaluate abductive story understanding systems (Ng and Mooney 1992; Charniak and Goldman 1991). In this task, character's higher-level plans must be inferred which explain their observed actions described in a narrative text. A logical representation of the literal meaning of the narrative text is given for each example. An example of a narrative text is: "Bill went to the liquor-store. He pointed a gun at the owner." The high-level plans in the knowledge base include shopping, robbing, restaurant dining, traveling in a vehicle, partying and jogging. Some narratives involve more than a single plan. We used the knowledge-base initially constructed for the ACCEL system (Ng and Mooney 1992). The knowledge-base contains a total of 126 Horn rules. The dataset contains 25 development and 25 test examples containing an average of 12.6 literals each.

Monroe/Modified-Monroe We also used the Monroe dataset, an artificially generated plan-recognition dataset in the emergency response domain by Blaylock and Allen (2005). This domain includes 10 top-level plans like setting up a temporary shelter, clearing a road wreck, and providing medical attention to victims. The task is to infer a single top level plan based on a set of observed actions generated by a *hierarchical transition network* (HTN) planner. We used the logical clauses constructed by Raghavan and Mooney (2010) encoding the knowledge in the HTN. The knowledge-base contains a total of 153 Horn rules. We generated 1,000 plans containing an average of 10.19 literals each.

Many rules in the Monroe domain contain multiple existentials in the reverse implications. This results in very complex networks for both the PC and HC models, leading to memory overflow and/or intractable inference. In order to compare their approach with PC MLN's, Raghavan and Mooney (2010) created a slightly altered domain, *Modified-Monroe*, designed to eliminate this problem. The resulting dataset contains the same number of rules as the original Monroe and 1,000 examples with an average of 10.56 observations.

Linux Linux is another plan recognition dataset created by Blaylock and Allen (2005). Human users were asked to perform various tasks in the Linux operating system and their commands were recorded. Given the sequence of individual commands, the task is to predict the correct top level plan. The 19 top level plans include tasks such as moving files based on their extension. Logical clauses were created to encode the effects of the relevant Linux commands. The knowledge-based contained a total of 50 Horn rules. There are 457 examples with an average of 6.1 literals each.

Methodology

All MLN models were implemented using Alchemy (Kok et al. 2010), an open source software package for learning and inference in Markov logic. We used the logical-abduction software developed by Raghavan and Mooney (2010) in our abductive model construction. For the HC and HCAM models, noisy-or was used to combine multiple explanations. Since the training data only provides supervision for top-level plans but not for subgoals or hidden causes, learning methods that support partially observed training data are required. We used a version of the gradient-based voted-perceptron algorithm (Singla and Domingos 2005) modified for partially observed data as discussed in Chapter 20 (Section 3.3.1) of Koller & Friedman (2009). Weights were initialized using hand-tuning on development data. Due to computational limitations, we were only able to learn weights for the HCAM model. For Monroe, we performed 10-fold cross validation. Since training Alchemy on the full training set was still intractable, we were able to train on at most one partition (1/10 of the overall data). In each fold, we trained on each one of the 9 training partitions separately using a range of learning rates. Each model was validated on the remaining training partitions, and the result for the best training-partition/learning-rate combination was picked. For Linux, we used 4-fold cross validation with a similar methodology. Developing online learning methods that allow scaling to larger training sets is an important direction for future work. Huynh and Mooney (2011)) present online weight-learning methods for MLNs, but they assume fully observable data (no hidden variables). For Story Understanding, weights were learned on the development set.

	Precision	Recall	F-measure
ACCEL-Sim	66.45	52.32	58.54
ACCEL-Coh	89.39	89.39	89.39
BALP	72.07	85.57	78.24
MLN-PC	67.31	68.10	67.70
MLN-HC	67.08	78.94	72.53
MLN-HCAM	69.13	75.32	72.10

Table 1: Results for Story Understanding

For the HC model, unit clause weights were hand-tuned on development data and noisy-or parameters were set to 0.1. The PC model's weights were set manually as by Kate and Mooney (2009).

To measure accuracy, we compared inferred plans to the correct plans. Partial credit was given for predicting the correct plan predicate with only a subset of its correct arguments. A point was rewarded for inferring the correct plan predicate, then, given the correct predicate, an additional point was rewarded for each correct argument. For example, if the correct plan was $plan_1(a_1, a_2)$ and the inferred plan was $plan_1(a_1, a_3)$, the score is 66.67%.

Results and Discussion

Story Understanding We used exact MPE inference with the cutting-plane method (Riedel 2008) to infer the set of plans in this domain. Accuracy is measured using *precision* (the fraction of inferred plans that are correct), *recall* (the fraction of correct plans that are inferred) and *F-measure* (the harmonic mean of precision and recall). We compared all three MLN models and with ACCEL using two explanation-selection metrics 1) *Simplicity* (Sim), which selects the explanation with the fewest assumptions, and 2) *Coherence* (Coh), which selects the explanation that maximally connects the input observations. We also compared with BALPs.

Table 1 shows the results. [1] The MLN-based models perform better than ACCEL-Sim. However, ACCEL-Coh gives the best results. The coherence metric incorporates extra criteria specific to story understanding. Incorporating this bias into a probabilistic model is difficult since it concerns a global graph-theoretic property of a complete explanation. However, the coherence metric is specific to narrative interpretation and not applicable to plan recognition in general. MLN-HCAM and MLN-HC have similar accuracy and outperform both ACCEL-Sim and MLN-PC. Ground networks for each example are relatively small, so abductive model construction does not provide an advantage on this dataset. BALP does somewhat better than MLN-HCAM.

Monroe and Linux These domains involve inferring a single top level plan. Therefore, we computed the marginal probability of each plan using MC-SAT (Poon and Domingos 2006) and picked the most probable one. We were unable to get reasonable results for either the PC or HC models on these datasets since the PC model exhausted memory and the HC model resulted in intractable inference. HCAM was the only MLN approach that scaled to these large datasets.

[1] Training time for the HCAM model was less than 10 minutes.

	Monroe	Linux
Blaylock	94.20	36.10
BALP	98.80	-
MLN-HCAM	97.30	38.94

Table 2: Schema Accuracy for Monroe and Linux

	100%	75%	50%	25%
MLN-PC	79.13	36.83	17.46	06.91
MLN-HC	88.18	46.63	21.11	15.15
MLN-HCAM	94.80	66.05	34.15	15.88
BALP	91.80	56.70	25.25	09.25

Table 3: Modified-Monroe Results Varying Observability

We compare its performance with both Blaylock and Allen's system (Blaylock) and BALPs. Published results only allow us to directly compare results for *schema accuracy*, i.e. the percentage of plan predicates that are correctly predicted, ignoring arguments.[2] Results are shown in Table 2. [3] MLN-HCAM is more accurate than Blaylock on both datasets. Published results for BALPs are only available for Monroe, where is it marginally better than MLN-HCAM.

Modified-Monroe This domain is used to effectively compare all three MLN methods on a large dataset that is still tractable for the PC and HC models. Since weight learning was intractable for models other than HCAM, to provide a fair comparison, we used weights that were hand-tuned on disjoint data. Results for BALPs also used weights set using a similar approach. MC-SAT was again used to compute marginal probabilities. We also evaluate predictive accuracy when only some fraction of the overall actions in a plan are observed, since in many applications not all actions are available at prediction time. We present results for several levels of observability, from 100% (full observability) to 25%, where for $k\%$ observability, only the first $k\%$ of the actions in the plan are available to the recognition system. Table 3 shows the results on predictive accuracy. They clearly demonstrate the continual improvement in performance from the PC model, to the HC model, to the full HCAM model. Also, MLN-HCAMs are more accurate than BALPs at all levels of observability.

Inference Time Results Table 4 compares inference time for the three MLN models on Story Understanding and Modified-Monroe. Results are averaged over 25 examples for Story Understanding and 1,000 examples for Modified-Monroe. Similar to the accuracy results, there is a continual improvement in efficiency as one moves from PC to HC to HCAM.

Summary Overall, the results demonstrate that the hidden cause model combined with abductive model construction allow an MLN approach to effectively scale to complex plan recognition problems, unlike the previously pro-

[2] Blaylock and Allen also report results on predicting arguments but using a different methodology that prevents direct comparisons.

[3] Training MLN-HCAM on Monroe and Linux took on the order of 3 days and 5 hrs, respectively.

	Story	Modified-Monroe
MLN-PC	2.93	252.13
MLN-HC	0.93	91.06
MLN-HCAM	0.25	2.27

Table 4: Average inference time in seconds

posed PC model. Also, MLN-HCAM outperforms a recent plan recognition system (Blaylock) on two datasets specifically developed for its evaluation. Finally, it is competitive with an alternative state-of-the-art approach, BALPs, which is slightly more accurate on Story Understanding and Monroe, but less accurate on Modified-Monroe.

Conclusions and Future Work

We have presented novel methods that enhance MLNs to perform effective abductive reasoning, producing an approach that combines the advantages of probabilistic and logical methods. The key ideas involve introducing a hidden predicate for each potential cause, followed by an abductive model construction procedure. Experiments on three plan recognition datasets demonstrate the benefit of our approach over existing methods.

Directions for future work include experimenting on a wider variety of domains, comparing to additional approaches, online learning of clause weights, and learning abductive rules from data.

Acknowledgements

We are thankful to Pedro Domingos for helpful discussions, and to Sindhu Raghavan for helpful discussions and for sharing the datasets and logical-abduction software. This research was partly funded by ARO grant W911NF-08-1-0242. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ARO or the United States Government.

References

Blaylock, N., and Allen, J. 2005. Recognizing instantiated goals using statistical methods. In *G. Kaminka (Ed.), Workshop on Modeling Others from Observations (MOO-05)*, 79–86.

Charniak, E., and Goldman, R. 1991. A probabilistic model of plan recognition. In *Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-91)*, 160–165.

Davis, J.; Ong, I.; Struyf, J.; Costa, V. S.; Burnside, E.; and Page, D. 2007. Change of representation for statistical relational learning. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-2007)*.

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

Huynh, T. N., and Mooney, R. J. 2011. Online max-margin weight learning for Markov logic networks. In *Proceedings of the Eleventh SIAM International Conference on Data Mining (SDM-11)*. To appear.

Kakas, A. C.; Kowalski, R. A.; and Toni, F. 1993. Abductive logic programming. *Journal of Logic and Computation* 2(6):719–770.

Kate, R. J., and Mooney, R. J. 2009. Probabilistic abduction using Markov logic networks. In *Proceedings of the IJCAI-09 Workshop on Plan, Activity, and Intent Recognition (PAIR-09)*.

Kok, S., and Domingos, P. 2007. Statistical predicate invention. In *Proceedings of 24th International Conference on Machine Learning (ICML-2007)*.

Kok, S.; Sumner, M.; Richardson, M.; Singla, P.; Poon, H.; Lowd, D.; Wang, J.; Nath, A.; and Domingos, P. 2010. The Alchemy system for statistical relational AI. Technical report, Department of Computer Science and Engineering, University of Washington.

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

Natarajan, S.; Khot, T.; Kersting, K.; Tadepalli, P.; and Shavlik, J. 2010. Exploiting causal independence in Markov logic networks: Combining undirected and directed models. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD-10)*, 434–450.

Ng, H. T., and Mooney, R. J. 1992. Abductive plan recognition and diagnosis: A comprehensive empirical evaluation. In *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning*, 499–508.

Ngo, L., and Haddawy, P. 1997. Answering queries from context-sensitive probabilistic knowledge bases. *Theoretical Computer Science* 171:147–177.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo,CA: Morgan Kaufmann.

Poon, H., and Domingos, P. 2006. Sound and efficient inference with probabilistic and deterministic dependencies. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI-06)*.

Pople, H. E. 1973. On the mechanization of abductive logic. In *Proceedings of the Third International Joint Conference on Artificial Intelligence (IJCAI-73)*, 147–152.

Raghavan, S., and Mooney, R. J. 2010. Bayesian abductive logic programs. In *AAAI-2010 Workshop on Statistical Relational AI*.

Riedel, S. 2008. Improving the accuracy and efficiency of MAP inference for Markov logic. In *Proceedings of 24th Conference on Uncertainty in Artificial Intelligence (UAI-2008)*, 468–475.

Sadilek, A., and Kautz, H. 2010. Modeling and reasoning about success, failure, intent of multi-agent activities. In *Proceedings of the UbiComp 2010 workshop on mobile context-awareness*.

Shavlik, J., and Natarajan, S. 2009. Speeding up inference in Markov logic networks by preprocessing to reduce the size of the resulting grounded network. In *Proceedings of the Twenty First International Joint Conference on Artificial Intelligence (IJCAI-2009)*, 1951–1956.

Singla, P., and Domingos, P. 2005. Discriminative training of Markov logic networks. In *Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI-05)*, 868–873.

Singla, P., and Domingos, P. 2008. Lifted first-order belief propagation. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI-08)*, 1094–1099.

Stickel, M. E. 1988. A Prolog-like inference system for computing minimum-cost abductive explanations in natural-language interpretation. Technical Report Technical Note 451, SRI International, Menlo Park, CA.