In *Proceedings of the IJCAI-09 Workshop on Plan, Activity, and Intent Recognition (PAIR-09)*, Pasadena, CA, July 2009.

Probabilistic Abduction using Markov Logic Networks

Rohit J. Kate Raymond J. Mooney
Department of Computer Sciences
The University of Texas at Austin
1 University Station C0500
Austin, TX 78712-0233, USA
{rjkate,mooney}@cs.utexas.edu

Abstract

Abduction is inference to the best explanation of a given set of evidence. It is important for plan or intent recognition systems. Traditional approaches to abductive reasoning have either used first-order logic, which is unable to reason under uncertainty, or Bayesian networks, which can handle uncertainty using probabilities but cannot directly handle an unbounded number of related entities. This paper proposes a new method for probabilistic abductive reasoning that combines the capabilities of first-order logic and graphical models by using Markov logic networks. Experimental results on a plan recognition task demonstrate the effectiveness of this method.

1 Introduction

Abduction is inference to the best explanation. Its applications include tasks in which observations need to be explained by inferring the best hypothesis, for example, plan or intent recognition, medical diagnosis, fault diagnosis etc. Most previous approaches to automated abduction have been based either on first-order logic and determine a small set of assumptions sufficient to deduce the observations to be explained, or on Bayesian networks and compute the posterior probability of alternative explanations given the observations. The former approaches cannot handle uncertainty in the evidence or background knowledge and are incapable of estimating the likelihood of alternative explanations. While the latter approaches handle uncertainty, they do not directly handle structured representations involving relations amongst multiple entities since Bayesian networks are essentially propositional in nature.

In this paper, we introduce a new method that combines the strengths of these two previous approaches by using probabilistic first-order logic. Our method uses Markov logic networks (MLNs) [Richardson and Domingos, 2006] which combine first-order logic and probabilistic graphical models. However, the inference mechanism in MLNs is based on deduction, not abduction, so we present a method for automatically adapting MLNs to perform a kind of probabilistic abduction using its probabilistic deductive inference mechanism. Since MLNs can be trained using supervised data for a particular task, unlike traditional logic-based approaches, our method has the additional advantage that it can be easily adapted to a particular domain by providing it with training examples. Finally, since we use an existing off-the-shelf framework for MLNs, any progress in efficiency or capability made to MLNs is readily incorporated into our method.

2 Background

This section provides some background on abduction and Markov logic networks (MLNs), in the next section we describe our method of doing probabilistic abduction using MLNs.

2.1 Abduction

The process of finding the best explanation from a set of observations is called *abduction*. In other words, abduction is inferring cause from effect and abductive reasoning works in the reverse direction from deductive reasoning in which effect is inferred from cause. Abduction is widely used in finding explanations for events and actions of agents, scientific theory formulation, medical diagnosis etc. For example, in plan recognition, actions are observed and the underlying plan that the agent is executing is abduced.

Formally, logical abduction is usually defined as follows:

- **Given:** Background knowledge B and observations O, both represented as sets of formulae in first-order logic but O is restricted to only ground formulae.
- **Find:** A hypothesis H, also a set of logical formulae, such that $B \cup H \nvdash \bot$ and $B \cup H \vdash O$.

Here $\vdash$ means logical entailment and $\bot$ means contradiction, i.e. find a hypothesis (a set of assumptions) which is consistent with the given background theory and together with it explains the observations.

In general, there could be multiple hypotheses H that explain a particular set of observations O given the background knowledge B. In these cases, the best hypothesis is selected where quality is typically based on the size or simplicity of the hypothesis, following the principle of Occam's Razor. In order to keep the abduction process computationally tractable, often the background knowledge B is restricted to

a set of Horn clauses [1]. In the rest of the paper we will also make this restriction. In some formalisms, a set of predicates are declared by the user as *abducibles* [Kakas *et al.*, 1993], which are the only predicates that can be used in assumptions.

Several first-order logic based approaches to abduction have been developed [Poole *et al.*, 1987; Stickel, 1988; Ng and Mooney, 1991; Kakas *et al.*, 1993]. These approaches perform first-order logical reasoning to determine the set of assumptions sufficient to deduce the observations. Their search is heuristically guided in accordance with the goodness criteria desired of the set of assumptions. However, a significant limitation of these purely logical approaches is that they are unable to reason under uncertainty or estimate the likelihood of alternative explanations. In many real-world applications, clauses in the provided background knowledge may be true in most cases but sometimes violated. Therefore, it may be desirable to choose an explanation even though it violates some uncertain clauses in the background knowledge but is otherwise a very good explanation. However, in purely logical approaches to abduction, such an explanation is ruled out because it contradicts the background knowledge. A probabilistic form of abduction is needed in order to account for uncertainty in the background knowledge and to handle noisy and incomplete observations.

A widely-used alternative framework for abduction is based on Bayesian networks [Pearl, 1988]. In these networks, the background knowledge with its uncertainties is encoded in a directed graph. Then, given a set of observations, probabilistic inference over the graph structure is done to compute the posterior probability of alternative explanations. However, a major limitation of Bayesian networks is that they are essentially based on propositional logic and cannot handle structured representations, hence preventing their use in situations that involve an unbounded number of entities with a variety of relations between them.

Finally, weighted abduction [Hobbs *et al.*, 1993] is a first-order logic-based approach to abduction which has some desirable features of probabilistic methods. It uses a weighting scheme to incorporate relevance and plausibility for the assumptions it makes and finds the lowest weight explanation. However, unlike probability theory, their weighting scheme does not have any solid theoretical basis and does not lend itself to a complete probabilistic analysis.

2.2 Markov Logic Networks

Markov logic [Richardson and Domingos, 2006] is a recently developed theoretically sound framework for combining first-order logic and probabilistic graphical models. 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 of the first-order logic formula in the knowledge base. Such a set of weighted first-order logic formulae is called a *Markov logic network* (MLN). A formula's weight reflects how strong a constraint it imposes on the set of possible worlds: the higher the weight, the lower the probability of a world that violates it; however, that probability need not be zero. An MLN with all infinite weights reduces to a traditional first-order knowledge base with only hard constraints. Since current MLN implementations do not support logical functions, we will assume function-free formulae.

Formally, an MLN L is a set of formula–weight pairs (F_i, w_i). Given a set of constants, it defines a joint probability distribution over a set of boolean variables $X = (X_1, X_2...)$ corresponding to the possible groundings (using the given constants) of the literals present in the first-order formulae:

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

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.

Semantically, an MLN can be viewed as a set of templates for constructing Markov networks [Pearl, 1988], the undirected counterparts of Bayesian networks. An MLN and a set of constants produce a Markov network in which each ground literal is a node and every pair of ground literals that appear together in some grounding of some formula are connected by an edge. Different sets of constants produce different Markov networks, however, there are certain regularities in their structure and parameters. For example, all groundings of the same formula have the same weight.

Probabilistic inference for an MLN (such as finding the most probable truth assignment for a given set of ground literals, or finding the probability that a particular formula holds) can be performed by first producing the ground Markov network and then using well known inference techniques for Markov networks, like Gibbs sampling. Given a knowledge base as a set of first-order logic formulae, and a database of training examples each consisting of a set of true ground literals, it is also possible to learn appropriate weights for the MLN formulae which maximize the probability of the training data. An open-source software package for MLNs, called Alchemy [2], is also available with many built-in algorithms for performing inference and learning.

3 Probabilistic Abduction using MLNs

In this section, we present a method that adapts Markov logic networks to automatically perform probabilistic abduction using its standard inference mechanism. We make the standard assumption that the initial background knowledge for abduction is specified as a set of Horn clauses. These clauses are then automatically transformed to an augmented set of MLN clauses that support traditional abductive inference. We first illustrate this method through examples and then formally describe the algorithm.

First, note that MLNs do not directly support abductive inference. Logical clauses in MLNs are soft constraints on possible worlds, increasing the probability of worlds in which the clauses are satisfied. Given a clause $P \rightarrow Q$ and an observation Q, we ideally want the probability of the world in

[1] A disjunction of literals with at most one positive literal or equivalently in the form of an implication $p \wedge q \wedge ... \wedge t \rightarrow u$

[2] http://alchemy.cs.washington.edu

which P is true to be higher than the world in which P is false because P is a potential explanation for Q. However, the clause $P \rightarrow Q$ is trivially satisfied when P is false; hence in a possible world in which Q is true, the world has the same probability whether P is true or false (since the clause is satisfied in either case). Therefore, MLNs have no mechanism to infer P from Q in this case. The MLN inference mechanism is thus inherently deductive, not abductive.

In order to adapt MLNs for abductive inference, we need to explicitly include clauses with reverse implications. For example, if we include $Q \rightarrow P$ in the database, specifying that Q is true increases the probability of worlds in which P is true. However, by giving the reverse implications lower weights, abductive rules are treated as "soft" and can be violated if other evidence contradicts their conclusions. Ideally, the proper weights on the reverse rules can be learned from training data.

So far in this description we have ignored the fact that the original Horn clauses can contain multiple literals on their left-hand-side (LHS) and that the arguments of the literals could be variables or constants. As a concrete example, consider the Horn clause:
$\forall x \forall y(mosquito(x) \wedge infected(x, Malaria) \wedge bite(x, y) \rightarrow infected(y, Malaria))$
This states that if x is a mosquito which is infected with malaria and it bites y, then y must also be infected with malaria. In order to adapt MLNs to do abductive inference for this clause, we need to include its reverse implication clause. But note that this clause is universally quantified in its variables and to explain that y has malaria, it is not necessary to assume that y was bitten by every infected mosquito x, however, it must be assumed that y was bitten by at least one infected mosquito x. Hence the appropriate reverse implication clause will be:
$\forall y(infected(y, Malaria) \rightarrow$
$\exists x(mosquito(x) \wedge infected(x, Malaria) \wedge bite(x, y)))$
Hence the general procedure to reverse the implication of a Horn clause requires existentially quantifying the universally quantified variables which appear on the left-hand-side (LHS) of the original clause but not on its right-hand-side (RHS).

But there could be multiple explanations in the background knowledge for the same observation. For example, a background knowledge base may state that someone can also get infected with malaria by having blood transfused from a person who is already infected with malaria:
$\forall x \forall y(infected(x, Malaria) \wedge transfuse(Blood, x, y) \rightarrow infected(y, Malaria))$
Upon reversing its implication, we get:
$\forall y(infected(y, Malaria) \rightarrow$
$\exists x(infected(x, Malaria) \wedge transfuse(Blood, x, y)))$
Now, if we are given the evidence that John is infected with malaria and he had blood transfused from Mary (i.e. $infected(John, Malaria)$ and $transfuse(Blood, Mary, John)$ are true), through the above reverse implication it can be concluded that Mary must have been infected with Malaria. But then we do not want to also conclude that John was bitten by an infected mosquito using the earlier reverse implication clause because Mary being infected by malaria is a better explanation. In general, an observation should only be explained by the best explanation and not by multiple explanations. Hence, in addition to reversing the implications we also need to bias MLNs against generating multiple explanations. In other words, we need the system to support what Pearl calls "explaining away" [Pearl, 1988]. We do this by first adding the reverse implication clause making a disjunction of the possible explanations:
$\forall y(infected(y, Malaria) \rightarrow$
$(\exists x(mosquito(x) \wedge infected(x, Malaria) \wedge bite(x, y))) \vee$
$(\exists x(infected(x, Malaria) \wedge transfuse(Blood, x, y))))$
which states that if y is infected with malaria then at least one of the possible explanations must be true. Next, we add a mutual exclusivity clause:
$\forall y(infected(y, Malaria) \rightarrow$
$\neg(\exists x(mosquito(x) \wedge infected(x, Malaria) \wedge bite(x, y))) \vee$
$\neg(\exists x(infected(x, Malaria) \wedge transfuse(Blood, x, y))))$
which states that both explanations cannot be true simultaneously. But note that because clauses in an MLN are soft constraints, while it would prefer a world in which both explanations are not simultaneously true, it still leaves open the possibility that both may be true (if the remaining clauses or explanations for other observations strongly imply this).

In general, for clauses $P_1 \rightarrow Q$, $P_2 \rightarrow Q$, ..., $P_n \rightarrow Q$ in the background knowledge, we include the reverse implication clause: $Q \rightarrow P_1 \vee P_2 \vee ... \vee P_n$, and a mutual exclusivity clause for every pair of explanations: $Q \rightarrow \neg P_1 \vee \neg P_2$, ..., $Q \rightarrow \neg P_1 \vee \neg P_n$, ..., $Q \rightarrow \neg P_2 \vee \neg P_n$ etc. Note that these are not Horn clauses, but this is not a problem since MLNs are not restricted to Horn clauses. The weights for these clauses can be typically learned from the training examples.

Before we describe our complete algorithm to adapt MLN's deductive inference mechanism for abduction, we want to mention how our approach avoids the potential problem that was pointed out regarding deductive solutions to abduction [Pearl, 1988]. Consider the knowledge base with the two clauses $rained \rightarrow grass\_is\_wet$ and $sprinkler\_was\_on \rightarrow grass\_is\_wet$. If grass is found wet then abductive inference should conclude that it either rained or the sprinkler was on. To perform such an abductive inference deductively, the reverse implications, $grass\_is\_wet \rightarrow rained$ and $grass\_is\_wet \rightarrow sprinkler\_was\_on$, will be added to the knowledge base. Then with such a knowledge base, suppose it is given that the $sprinkler\_was\_on$ is true, a deduction procedure will conclude that $grass\_is\_wet$ is true using $sprinkler\_was\_on \rightarrow grass\_is\_wet$ clause, but next, using the clause $grass\_is\_wet \rightarrow rained$, it will also conclude that $rained$ is true. In effect, it will conclude that it rained because the sprinkler was on, which is absurd. But this absurd reasoning is prevented in our approach because our approach also includes the mutual exclusivity clause $grass\_is\_wet \rightarrow \neg rained \vee \neg sprinkler\_was\_on$. With this, given $sprinkler\_was\_on$ is true, once it is concluded that $grass\_is\_wet$ is true, then both $rained$ and $sprinkler\_was\_on$ cannot be true. Since $sprinkler\_was\_on$ is already true, $rained$ cannot be true.

Now we formally describe our general algorithm for expanding an existing knowledge base to a set of MLN clauses that supports abduction. For all the predicates on the RHS

(i) $u(a,b) \wedge x(c,b) \rightarrow t(a,b,c)$
(ii) $v(a,b,d,e) \rightarrow t(a,b,d)$
(iii) $y(b,c) \rightarrow t(A,b,c)$
(iv) $u(a,B) \wedge z(B,c,f) \rightarrow t(a,B,c)$

Figure 1: A sample background knowledge base of Horn clauses with the same RHS predicate.

of the given Horn clauses, it adds the reverse implication and mutual exclusivity clauses for every suitable variable–constant combination for the arguments of those predicates.

- Given: A set of universally quantified first-order Horn clauses as the background knowledge.
- Partition the background knowledge into sets of Horn clauses with the same right-hand-side (RHS) predicate. For example, Figure 1 shows such a set with the same RHS predicate $t(\_,\_,\_)$. In the figure, variables are shown in lower-case and constants are shown in upper-case. If the user has declared a set of abducible predicates, then only consider these predicates as the potential RHS predicates for this step.
- Do the following for each set $\mathcal{H}$ in the above partition.
- Collect the set of RHS literals $\mathcal{T}$ of $\mathcal{H}$. For example, for the set $\mathcal{H}$ shown in Figure 1, $\mathcal{T} = \{t(a,b,c), t(a,b,d), t(A,b,c), t(a,B,c)\}$.
- For each subset of $\mathcal{T}$, find the most general unifier. Call the resulting set of literals $\mathcal{M}$. For $\mathcal{T}$ above, $\mathcal{M} = \{t(a,b,c), t(A,b,c), t(a,B,c), t(A,B,c)\}$.
- For each literal m in $\mathcal{M}$, consider every Horn clause $\mathcal{L} \rightarrow \mathcal{R}$ in $\mathcal{H}$, where $\mathcal{R}$ is a literal and $\mathcal{L}$ is a conjunction of literals, and $\mathcal{R}$ and m are unifiable, but there is no argument in which $\mathcal{R}$ has a constant and m has a variable. Unify m and $\mathcal{R}$ to obtain a substitution θ. For example, for $m = t(a,b,c)$ and for the Horn clause (ii) in Figure 1, unifying $t(a,b,c)$ and $t(a,b,d)$ gives $\theta = \{d/c\}$. Note that the Horn clause (iii) cannot be used with $m = t(a,b,c)$ because its $\mathcal{R}$, $t(A,b,c)$, has a constant for the first argument where $t(a,b,c)$ has a variable.
- Continuing with the $\mathcal{L} \rightarrow \mathcal{R}$, m and θ from the previous step, let $\mathcal{E}$ be the set of variables present in $\mathcal{L}_\theta$ but not in m. Construct a new clause $m \rightarrow \exists \mathcal{E} \mathcal{L}_\theta$. For the Horn clause (ii) in Figure 1, and for $m = t(a,b,c)$ and $\theta = \{d/c\}$ from the previous step, the set $\mathcal{E}$ is $\{e\}$. Hence, the constructed clause will be $t(a,b,c) \rightarrow \exists e\, v(a,b,c,e)$. [3]
- Using the new clauses obtained through the previous two steps for each m, add the corresponding reverse implication and mutual exclusivity clauses to the background knowledge. [4] Figure 2 shows the clauses that will be added to the knowledge base for each value of m given the initial knowledge base in Figure 1

[3] This step may require renaming variables to avoid name clashes. For example, if the variable e was c, then c would have to be renamed. It is essential to add the existential variable e because the reverse implication need not be true for all values of e but must be true for at least one value of e.

$m = t(a,b,c)$, combine (i) and (ii):

$$t(a,b,c) \rightarrow (u(a,b) \wedge x(c,b)) \vee (\exists e\, v(a,b,c,e))$$
$$t(a,b,c) \rightarrow \neg(u(a,b) \wedge x(c,b)) \vee \neg(\exists e\, v(a,b,c,e))$$

$m = t(A,b,c)$, combine (i), (ii) and (iii):

$$t(A,b,c) \rightarrow (u(A,b) \wedge x(c,b)) \vee (\exists e\, v(A,b,c,e)) \vee (y(b,c))$$
$$t(A,b,c) \rightarrow \neg(u(A,b) \wedge x(c,b)) \vee \neg(\exists e\, v(A,b,c,e))$$
$$t(A,b,c) \rightarrow \neg(u(A,b) \wedge x(c,b)) \vee \neg(y(b,c))$$
$$t(A,b,c) \rightarrow \neg(\exists e\, v(A,b,c,e)) \vee \neg(y(b,c))$$

$m = t(a,B,c)$, combine (i), (ii) and (iv):

$$t(a,B,c) \rightarrow (u(a,B) \wedge x(c,B)) \vee (\exists e\, v(a,B,c,e)) \vee (\exists f\ (u(a,B) \wedge x(B,c,f)))$$
$$t(a,B,c) \rightarrow \neg(u(a,B) \wedge x(c,B)) \vee \neg(\exists e\, v(a,B,c,e))$$
$$t(a,B,c) \rightarrow \neg(u(a,B) \wedge x(c,B)) \vee \neg(\exists f\ (u(a,B) \wedge x(B,c,f)))$$
$$t(a,B,c) \rightarrow \neg(\exists e\, v(a,B,c,e)) \vee \neg(\exists f\ (u(a,B) \wedge x(B,c,f)))$$

$m = t(A,B,c)$, combine (i), (ii), (iii) and (iv):

$$t(A,B,c) \rightarrow (u(A,B) \wedge x(c,B)) \vee (\exists e\, v(A,B,c,e)) \vee (y(B,c)) \vee (\exists f\ (u(A,B) \wedge x(B,c,f)))$$
$$t(A,B,c) \rightarrow \neg(u(A,B) \wedge x(c,B)) \vee \neg(\exists e\, v(A,B,c,e))$$
$$t(A,B,c) \rightarrow \neg(u(A,B) \wedge x(c,B)) \vee \neg(y(B,c))$$
$$t(A,B,c) \rightarrow \neg(u(A,B) \wedge x(c,B)) \vee \neg(\exists f\ (u(A,B) \wedge x(B,c,f)))$$
$$t(A,B,c) \rightarrow \neg(\exists e\, v(A,B,c,e)) \vee \neg(y(B,c))$$
$$t(A,B,c) \rightarrow \neg(\exists e\, v(A,B,c,e)) \vee \neg(\exists f\ (u(A,B) \wedge x(B,c,f)))$$
$$t(A,B,c) \rightarrow \neg(y(B,c)) \vee \neg(\exists f\ (u(A,B) \wedge x(B,c,f)))$$

Figure 2: The abductive MLN clauses constructed for the knowledge base shown in Figure 1.

Once the clauses are added to the background knowledge, their weights can be set manually or can be learned using training examples along with the weights of the original clauses.

4 Experiments

This section describes experiments that evaluated the proposed abductive MLN approach on a previously studied plan-recognition task.

4.1 Dataset

We used a dataset for plan recognition previously used to evaluate abductive systems [5] [Ng and Mooney, 1991; Char-

[4] This step may also require variable re-naming to avoid variable name clashing.

[5] This data can be downloaded from `http://www.cs.utexas.edu/~ml/accel.html`

niak and Goldman, 1991]. In this task, character's higher-level plans must be inferred in order to 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. Some examples of narratives are: "Bill went to the liquor-store. He pointed a gun at the owner."; "Jack went to the supermarket. He found some milk on the shelf. He paid for it."; and "Fred went to the supermarket. He pointed a gun at the owner. He packed his bag. He went to the airport." The dataset consists of 25 development and 25 test examples. The development data was constructed by Goldman [1990] and the testing data was later added by Ng and Mooney [1992]. Their logical representations which form the observations to be explained contain an average of 12.6 literals.

The background knowledge-base was initially constructed for the ACCEL system [Ng and Mooney, 1991] to work with the 25 development examples. It was constructed such that the high-level plans (like shopping and robbing) together with appropriate role-fillers (such as someone being the shopper of a shopping plan or a robber of a robbing plan) imply the input literals representing the observed actions (like going to a store and pointing a gun). The plans in the knowledge base include shopping, robbing, restaurant dining, traveling in a vehicle (bus, taxi or plane), partying and jogging. Each of these plans in turn has subplans, and some of the plans contain recursive subplans. For example, traveling by plane includes a subplan of traveling (in some vehicle) to the airport to catch a plane. There are 107 such knowledge base rules. In addition, there are also consistency checking rules. For example, there has to be a shopper for a shopping plan etc. There are also some rules which ensure unique argument values for some predicates; for example, two different people cannot both be shoppers for the same shopping plan.

Figure 3 (a) shows a sample of the background knowledge base that was used in this task. The first clause says that if s is an instance of a shopping event and g is the going step required for the shopping event s, then g must be an instance of a going event. The next clause says that if, in addition, p is the shopper of the shopping event s, then p must be the goer of the going event g. Similarly, the following clause says that if str is the store of the shopping event s, then it must be the destination of the going event g. The next three clauses analogously relate the shopping event s to the paying event pay.

An example of a narrative text along with its logical representation is shown in Figure 3 (b). Figure 3 (c) shows what assumptions must be true in order to explain the observations shown in Figure 3 (b) using the background knowledge shown in Figure 3 (a). For this example, it can be determined through abduction that there must have been a shopping event, call it $S1$, whose shopper was $Bill$ and who shopped for $Milk$. Note that if instead of "He paid for some milk", the observation was "He pointed a gun", an existence of a robbing event must be assumed whose robber would be $Bill$, instead of a shopping event whose shopper was $Bill$ (the part of the background knowledge about robbing is not shown in Figure 3 (a)).

(a)
$instance\_shopping(s) \wedge go\_step(s, g) \rightarrow instance\_going(g)$

$instance\_shopping(s) \wedge go\_step(s, g) \wedge shopper(s, p) \rightarrow goer(g, p)$

$instance\_shopping(s) \wedge go\_step(s, g) \wedge store(s, str) \rightarrow destination\_go(g, str)$

$instance\_shopping(s) \wedge pay\_step(s, pay) \rightarrow instance\_paying(pay)$

$instance\_shopping(s) \wedge pay\_step(s, pay) \wedge shopper(s, p) \rightarrow payer(pay, p)$

$instance\_shopping(s) \wedge pay\_step(s, pay) \wedge thing\_shopped\_for(s, t) \rightarrow thing\_paid(pay, t)$

(b)
"Bill went to the store. He paid for some milk"
$instance\_going(Go1)$
$goer(Go1, Bill)$
$destination\_go(Go1, Store)$
$instance\_paying(Pay1)$
$payer(Pay1, Bill)$
$thing\_paid(Pay1, Milk)$

(c)
$instance\_shopping(S1)$
$shopper(S1, Bill)$
$go\_step(S1, Go1)$
$pay\_step(S1, Pay1)$
$thing\_shopped\_for(S1, Milk)$

Figure 3: (a) A sample of the background knowledge for the evaluation task. (b) An example of narrative text and its logical representation. (c) The assumptions which explain the observations in (b) using the background knowledge (a).

4.2 Methodology

We used the algorithm described in the previous section to automatically add clauses to the background knowledge base to perform abduction using MLNs. We used the Alchemy software for MLNs. Alchemy requires specifying types for the arguments of the predicates which are then used along with the provided constants to ground the MLN into a Markov network. It is desirable to keep the MLN from generating useless groundings, for example, a constant used to specify a shopper should not be used to specify a store. Hence, we provided very specific types for predicate arguments. We also renamed some predicates in the original knowledge base to allow for even more specific types. For example, we renamed instance(a,action) where action could be shopping, robbing etc. into instance_shopping(s), instance_robbing(r) etc., where the arguments of each predicate could be constrained to a specific type. In the future, one could design

an automatic method for generating the more specific typing in order to improve the efficiency of the resulting MLN.

We found that the 25 development examples were too few to learn weights from for the fairly large knowledge base. Hence, we heuristically set the weights. We put soft weights (i.e. 2) on the reverse implication clauses and hard weights (practically infinite) for all the mutual exclusivity clauses. In addition, we put small negative weights (-1) on unit clauses for all of the predicates in the knowledge base to discourage the system from assuming instances of them to be true unless there is a clear reason to do so. Given a set of observations as ground literals, we use Alchemy's probabilistic inference to determine the most likely truth assignment for the remaining ground literals, i.e the *most probable explanation* (MPE) [Pearl, 1988].

We compared the results of our system with that of ACCEL [Ng and Mooney, 1992] which is a purely logic-based system. It internally uses a metric to guide its search for selecting the best explanation. For the plan recognition task, it can use two different metrics. The first is *simplicity*, which selects the explanation of the smallest size, i.e. the one with the fewest number of assumptions. The second is *coherence*, which selects the explanation that maximally connects the input observations. This second metric is specifically geared towards the task of text interpretation to provide *explanatory coherence* [Ng and Mooney, 1990], i.e., how well the input sentences are tied together in the final interpretation. The metric exploits the well known fact that sentences in natural language text are usually connected in a coherent way. For instance, in the example "John took a bus. He bought milk.", the coherence metric will bias the abduction towards the interpretation in which John took the bus to a store where he bought the milk. Currently, we have not incorporated this bias into the MLN's abduction process.

The development data was also used by Charniak and Goldman [1989; 1991]. They used a Bayesian probabilistic approach for this task. Their approach constructs a Bayesian network for each example. Given the observations, the most probable explanation is selected using the conditional probabilities attached to the Bayesian network. Their method critically depends on the values assigned to various probabilities and these were carefully set manually using common sense knowledge about the various events. ACCEL achieved results similar to their system on the 25 development examples without manually setting any numeric probabilities.

In the traditional definition of abduction, the answer is given in the form of a best set of assumptions that explain the observations, however, the answer given by a deductive system like an MLN will always include other facts that can be deduced from that best set of assumptions. Hence, in order to compare ACCEL and MLNs fairly, we deductively expand ACCEL's answers, i.e. we use the original knowledge base to deduce additional facts that can be inferred from the assumptions and the observations. Similarly, we also deductively expanded the gold-standard answers in the dataset. We measured the *precision* (what fraction of the predicted ground literals are in the gold-standard answers) and *recall* (what fraction of the ground literals in the gold standard answers were predicted). We also computed the *F-measure*, the harmonic mean of precision and recall.

	MLN	ACCEL-Simplicity	ACCEL-Coherence
Development			
Recall	90.10	84.98	93.70
Precision	92.53	91.93	93.65
F-measure	91.30	88.32	93.68
Testing			
Recall	81.44	71.87	88.30
Precision	77.76	88.58	89.06
F-measure	79.56	79.36	88.68

Table 1: Results on abduction for the plan recognition task using MLN and using the two metrics for the ACCEL system.

4.3 Results and Discussion

The results are shown in Table 1. MLN took only 25 seconds to run all the 50 examples on a 2.3GHz 4 GB machine. Results for the two datasets are shown separately because the knowledge base was constructed to work with the development data without any knowledge of the testing data. The results show that the MLN does better than the simplicity metric, particularly on the development data. On the test data, it obtains better recall than the simplicity metric but lags behind in precision. The MLN performs very close to the coherence metric on the development data but does worse on the test data. We again note that the coherence metric was specifically designed to do well on this particular task, but MLNs are more general and are not tailored for the specific task of narrative understanding. When recognizing plans from observed actions rather than from those relayed by a human story teller, the bias for explanations that tie all of the observations together is less useful and simply maximizing the probability of an explanation is more desirable.

We also note that this specific dataset does not really require a full probabilistic treatment, since there is little noise or uncertainty in the knowledge base or the observations, and the correct explanations are usually very clear. We believe that probabilistic abduction using MLNs will have a greater advantage on tasks where uncertainty is more pronounced and where there are multiple plausible explanations that must be evaluated based on the underlying probability of various inferences and assumptions. We also note that an advantage of MLNs is that they do not need specific heuristics to pick the best explanation but simply find the most probable interpretation. Also, given sufficient training examples, clause weights and even additional clauses can be learned, and thus the system can automatically adapt to a specific task. Previous abductive systems like ACCEL and Charniak and Goldman's system do not provide any learning mechanisms.

5 Future Work

An obvious direction for future work is evaluating the MLN approach on a task in which uncertainty plays a greater role. Such a task could more clearly demonstrate the advantages of using a probabilistic framework like MLNs. Also, a larger dataset could be used to learn weights and additional clauses

to automatically adapt the system to a particular domain. Another issue for future work is how task-specific biases, like the coherence metric, could be incorporated into the general MLN abduction framework.

Besides MLNs, there are other frameworks for combining first-order logic and graphical models [Getoor and Taskar, 2007]. For example, Bayesian logic programs (BLPs) [Kersting and De Raedt, 2001] combine Bayesian networks and first-order Horn-clause logic. Therefore, BLPs use a directed graphical model while MLNs use an undirected one. Given that directed models like Bayesian networks are particularly suited for doing abduction, it would be interesting to perform probabilistic abduction using BLPs and compare it with the current MLN approach.

6 Conclusions

We have presented a general method for probabilistic first-order logical abduction using Markov logic networks. The existing deductive inference system of MLNs is employed for abduction by automatically adding suitably reversed implications to the knowledge base. The resulting abductive system offers the advantages of handling uncertainty using probabilities and of handling an unbounded number of related entities using first-order logic. It is also capable of adapting to a domain by learning from training examples. Experiments on a small plan-recognition data set demonstrated that it compares favorably with special-purpose logic-based abductive systems even on a domain which favors those systems.

Acknowledgments

This research was funded by ARO grant W911NF-08-1-0242.

References

[Charniak and Goldman, 1989] Eugene Charniak and Robert P. Goldman. A semantics for probabilistic quantifier-free first-order languages, with particular application to story understanding. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence (IJCAI-89)*, Detroit, MI, 1989.

[Charniak and Goldman, 1991] Eugene Charniak and Robert Goldman. A probabilistic model of plan recognition. In *Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-91)*, pages 160–165, Anaheim, CA, 1991.

[Getoor and Taskar, 2007] L. Getoor and B. Taskar, editors. *Introduction to Statistical Relational Learning*. MIT Press, Cambridge, MA, 2007.

[Goldman, 1990] Robert P. Goldman. *A Probabilistic Approach to Language Understanding*. PhD thesis, Department of Computer Science, Brown University, Providence, RI, December 1990. Technical Report CS-90-34.

[Hobbs *et al.*, 1993] Jerry R. Hobbs, Mark E. Stickel, Douglas E. Appelt, and Paul A. Martin. Interpretation as abduction. *Artificial Intelligence*, 63(1-2):69–142, 1993.

[Kakas *et al.*, 1993] Andonakis C. Kakas, Robert A. Kowalski, and Francesca Toni. Abductive logic programming. *Journal of Logic and Computation*, 2(6):719–770, 1993.

[Kersting and De Raedt, 2001] Kristian Kersting and Luc De Raedt. Towards combining inductive logic programming with Bayesian networks. In *Proceedings of the 11th International Conference on Inductive Logic Programming (ILP-01)*, pages 118–131, Strasbourg, France, September 2001.

[Ng and Mooney, 1990] Hwee Tou Ng and Raymond J. Mooney. The role of coherence in abductive explanation. In *Proceedings of the Eighth National Conference on Artificial Intelligence (AAAI-90)*, pages 337–442, Detroit, MI, July 1990.

[Ng and Mooney, 1991] Hwee Tou Ng and Raymond J. Mooney. An efficient first-order Horn-clause abduction system based on the ATMS. In *Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-91)*, pages 494–499, Anaheim, CA, July 1991.

[Ng and Mooney, 1992] Hwee Tou Ng and Raymond J. Mooney. Abductive plan recognition and diagnosis: A comprehensive empirical evaluation. In *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning*, pages 499–508, Cambridge, MA, October 1992.

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

[Poole *et al.*, 1987] D. L. Poole, R. G. Goebel, and R. Aleliunas. Theorist: A logical reasoning system for defaults and diagnosis. In N. J. Cercone and G. McCalla, editors, *The Knowledge Frontier: Essays in the Representation of Knowledge*, pages 331–352. Springer Verlag, New York, 1987.

[Richardson and Domingos, 2006] Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62:107–136, 2006.

[Stickel, 1988] M. E. Stickel. 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, September 1988.